INTERNATIONAL BACCALAUREATE DIPLOMA PROGRAMME

INTERNAL ASSESSMENT MATHEMATIQUES HIGHER LEVEL

THEME DE RECHERCHE :

ÉTUDE ET MODELISATION DE LA TRAJECTOIRE D'UNE FUSÉE PAR RÉSOLUTION NUMÉRIQUE ET ANALYTIQUE D'ÉQUATIONS DIFFÉRENTIELLES

Pierre-Ange Delbary Rouillé

Nombre de mots : 3030

1. Introduction

1.1 Contexte et motivation

L'exploration spatiale repose sur une maîtrise extrême des lois de la physique et des mathématiques. Les équations différentielles jouent un rôle central dans la modélisation des mouvements des fusées. En effet, la trajectoire d'une fusée est soumise à de nombreux paramètres physiques tels que la gravité, la poussée, la résistance de l'air et la variation de masse due à la consommation de carburant.

Dans ce projet, nous nous intéressons à la modélisation mathématique de la trajectoire d'une fusée en utilisant des équations différentielles linéaires ou non. L'objectif est de comparer ces modèles en fonction de leur précision.

1.2 Objectifs du projet

Le projet vise à :

1. Construire un modèle simplifié de la trajectoire d'une fusée et utiliser une équation différentielle linéaire pour le résoudre,
2. Améliorer le modèle en ajoutant des paramètres physiques plus réalistes. L'équation différentielle devient non linéaire et nécessite une résolution numérique,
3. Coder plusieurs algorithmes de trajectoires selon différentes méthodes de résolution numérique (utilisation du logiciel Matlab),
4. Effectuer des calculs d'erreur et analyser la stabilité des méthodes,
5. Évaluer la pertinence des résultats des méthodes utilisées pour la modélisation de la réalité.

1.3 Introduction aux équations différentielles dans la dynamique des fusées

Les équations différentielles permettent de décrire l'évolution d'un système dynamique en fonction de paramètres, le temps en particulier. Pour une fusée, sa position, sa vitesse et son accélération sont régies par des équations qui font intervenir les forces appliquées.

En négligeant la variation de masse et la résistance de l'air, l'équation du mouvement d'une fusée, est donnée par la seconde loi de Newton :

$$\sum_i \vec{F_i} = m.\vec{a}$$

Soit, pour une fusée à trajectoire verticale :

Équation	Variables
$m\frac{dv}{dt} = F - mg$	m est la masse de la fusée, v est la vitesse, F est la force de poussée du moteur, g est l'accélération due à la gravité terrestre.

En ajoutant la consommation de carburant, la masse m varie dans le temps, et l'équation devient non linéaire. Ces aspects seront étudiés progressivement.

2. Modèle Simplifié

Dans cette section, nous allons établir un premier modèle de trajectoire de fusée basé sur une équation différentielle linéaire. Ce modèle est volontairement simplifié pour obtenir une solution analytique exploitable.

2.1 Hypothèses du modèle simplifié

Dans un premier temps, nous posons les hypothèses suivantes :

Hypothèses
La fusée est à masse constante (pas de consommation de carburant),
La poussée du moteur est constante (force égale tout au long du vol)
La résistance de l'air est négligée
La fusée suit une trajectoire strictement verticale

2.2 Établissement de l'équation différentielle

Selon la seconde loi de Newton appliquée à la fusée :

Équations
$$m\frac{dv}{dt} = F - mg$$ $$\Leftrightarrow \quad \frac{dv}{dt} = \frac{F}{m} - g$$

Il s'agit d'une équation différentielle linéaire du premier ordre.

2.3 Résolution de l'équation différentielle

On peut l'intégrer directement en fonction de t :

Équation	Variable
$v(t) = \left(\frac{F}{m} - g\right)t + v_0$	v_0 est la vitesse initiale de la fusée au moment du décollage.

En intégrant à nouveau pour obtenir la position $h(t)$:

Équation	Variable
$h(t) = \left(\frac{F}{m} - g\right)\frac{t^2}{2} + v_0 t + h_0$	h_0 est l'altitude initiale.

Ces équations permettent de calculer la vitesse et la position de la fusée en fonction du temps dans ce modèle simplifié.

3. Complexification du Modèle

Nous allons maintenant affiner le modèle de la trajectoire de la fusée en intégrant la masse variable et la résistance de l'air.

3.1 Ajout de la variation de masse

Précédemment, nous avons supposé la masse de la fusée constante. En réalité, la fusée brûle du carburant, ce qui réduit sa masse dans le temps.

La loi de Tsiolkovski[1], qui décrit cette perte de masse, donne :

Équation	Variable
$\frac{dm}{dt} = -\dot{m}$	$m(t)$ est la masse de la fusée à l'instant t, $\dot{m}$ est le débit massique d'éjection du carburant (kg/s).

En tenant compte de cette variation, l'équation du mouvement devient :

Équations	Variable
$m(t)\frac{dv}{dt} = F - m(t).g$ Et $m(t) = m_0 - \dot{m}t$	m_0 est la masse initiale de la fusée.

En remplaçant $m(t)$, on obtient :

Équations
$\frac{dv}{dt} = \frac{F}{m_0 - \dot{m}t} - g$ $\Leftrightarrow \quad dv = \left(\frac{F}{m_0 - \dot{m}t} - g\right) dt$

[1] **Tsiolkovski, K.E. :** *Exploration de l'espace cosmique par des engins à réaction*, 1903.

En intégrant en fonction du temps:

$$\int dv = \int \left(\frac{F}{m_0 - \dot{m}t} - g\right) dt$$

$$\Leftrightarrow v = -\frac{F}{\dot{m}} \ln(m_0 - \dot{m}t) - gt + c \quad \text{avec} \quad (m_0 - \dot{m}t) > 0 \quad \forall\, t$$

Cette équation est encore linéaire.

Mais la réalité des fusées complique rapidement la variation de m (voire de la poussée).

Quelques exemples :

La masse dépend de la vitesse

Par exemple :

$$m(v) = m_0 - \alpha v^2$$

$$\Rightarrow \frac{dv}{dt} = \frac{F}{m_0 - \alpha v^2} - g$$

$\Rightarrow$ **non-linéarité structurelle.**

La masse dépend de l'altitude

Même raisonnement :

$$\Rightarrow \frac{dv}{dt} = \frac{F}{m(h)} - g$$

Et si $h(t)$ dépend de $v(t)$ alors on a une **non-linéarité** indirecte dans l'équation.

La poussée dépend de la masse

Par exemple :

$$F(t) = k.\dot{m}(t)$$

Et dans ce cas :

$$\Rightarrow \frac{dv}{dt} = \frac{-\dot{m}(t)v_e}{m(t)} - g$$

Apparaissent des termes comme $\frac{\dot{m}(t)}{m(t)}$, ou même plus compliqué si $\dot{m}$ n'est pas constante. Ce n'est pas forcément non-linéaire en v, sauf si $\dot{m}$ dépend de v.

C'est pourquoi nous traiterons cette équation dans son cas le plus défavorable, c'est-à-dire non-linéaire, et donc via une intégration numérique.

3.2 Ajout de la résistance de l'air qui varie en fonction de l'altitude

L'air exerce une force opposée au mouvement de la fusée appelée force de traînée aérodynamique :

Équation	Variable
$F_{Traînée} = \frac{1}{2} C_d \rho A v^2$	C_d est le coefficient de traînée (dépend de la forme de la fusée), ρ est la densité de l'air (diminue avec l'altitude), A est la section frontale de la fusée, v est la vitesse de la fusée.

L'équation différentielle devient :

Équation
$\frac{dv}{dt} = \frac{F}{m_0 - \dot{m}t} - g - \frac{1}{2}\frac{C_d \rho A v^2}{m_0 - \dot{m}t}$

4 Analyse mathématique rigoureuse des méthodes numériques

4.1 Définition formelle des méthodes

4.1.1 Introduction aux méthodes d'intégration numérique[2]

Les équations différentielles ordinaires (EDO) permettent la modélisation mathématique de phénomènes dynamiques. Cependant, dans de nombreux cas, il est impossible de trouver une solution analytique explicite. On peut alors utiliser des méthodes numériques pour approximer la solution.

Considérons une EDO sous la forme générale :

Équation
$\frac{dy}{dt} = f(t, y), \quad$ avec $\quad y(0) = y_0$

Où :

Conditions	
$\frac{dy}{dt} \neq f(y).g(t)$	$\Leftrightarrow$N'est pas une équation séparable
$f(\lambda t, \lambda y) \neq \lambda^n f(t, y) \quad \forall \lambda \in \mathbb{R}$	$\Leftrightarrow$N'est pas une équation homogène
$\nexists F(t, y)$ telle que : • $\frac{\delta F}{\delta t} = M(t, y)$ et $\frac{\delta F}{\delta y} = N(t, y)$ • Et $\frac{\delta M}{\delta y} = \frac{\delta N}{\delta t}$ $(\delta \, étant \, la \, dérivée \, partielle)$	$\Leftrightarrow$N'est pas une équation exacte
$\frac{dy}{dt} + P(t)y \neq Q(t)y^n$	$\Leftrightarrow$ N'est pas une équation de Bernoulli (non-linéaire sauf si $n = 0$ ou 1, mais pouvant être résolue par changement de variable)
	N'est pas une EDO autonome traitable pas séparation des variables.

[2] **Meyer, Y.** - *Analyse numérique* (Tome 1). Éditions Dunod, 2012.

Les méthodes d'intégration numérique cherchent à approximer la solution en avançant par pas de temps dt :

Équation
$$y_{n+1} = y_n + \Delta y$$

Le choix de la méthode influe sur la précision de l'approximation et sur la stabilité de la solution.

Nous présenterons plus loin trois méthodes : Euler**,** Heun (Euler Amélioré) et Runge-Kutta d'ordre 4 (RK4).

4.1.2 Approximation de Taylor et lien avec les erreurs numériques

L'approximation de Taylor permet de comprendre comment les méthodes numériques approximent une solution continue. Son principe est d'approximer la valeur d'une fonction en un point voisin grâce à des dérivées successives. Elle exprime une fonction différentiable sous forme d'une somme infinie de monômes qui sont ses termes dérivés. Les méthodes d'Euler, Heun et RK4 en sont issues.

Considérons une équation différentielle du type :

Équation
$$\frac{dy}{dt} = f(t,y) \text{ et } y(t_0) = y_0$$

Le développement de Taylor de la solution $y(t)$ autour du point t, donne :

Équation
$$y(t+dt) = y(t) + dtf(t,y) + \frac{dt^2}{2}f'(t,y) + \frac{dt^3}{6}f''(t,y) + \frac{dt^4}{24}f'''(t,y) + \cdots + O(dt^n)$$

Où :

Fonction	Explications
$f(t,y) = \frac{dy}{dt}$	Est l'expression donnée par l'équation différentielle,
$f'(t,y) = \frac{d^2y}{dt^2}$	Est la dérivée temporelle de f
$O(dt^n)$	Désigne le reste de l'approximation, c'est-à-dire l'erreur qui apparaît quand on néglige les termes d'ordre n ou supérieurs.

La notation « grand O » signifie que le reste est borné par une constante multipliée par $(dt)^n$ lorsque $dt \to 0$. Cela permet de caractériser l'ordre d'erreur locale : plus n est grand, plus l'erreur diminue rapidement.

L'ordre de l'erreur de chaque méthode dépend du nombre de termes inclus dans son approximation.

Nota : Taylor vs Maclaurin

Nous avons étudié lors de nos cours le développement de Maclaurin. C'est un cas particulier du développement de Taylor lorsqu'on le développe autour du point 0.

On a alors :

$$y(t) = y(0) + y'(0)t + \frac{1}{2}y''(0)t^2 + \cdots$$

En ce qui nous concerne, nous ne développons pas autour du point 0 seul, mais autour du point t qui évolue à chaque itération de pas. C'est pourquoi avec les méthodes numériques que nous aborderons, nous utiliserons un développement de Taylor.

4.1.3 Méthode d'Euler (ordre 1)

La méthode d'Euler est la plus simple. Elle approxime la dérivée par une tangente :

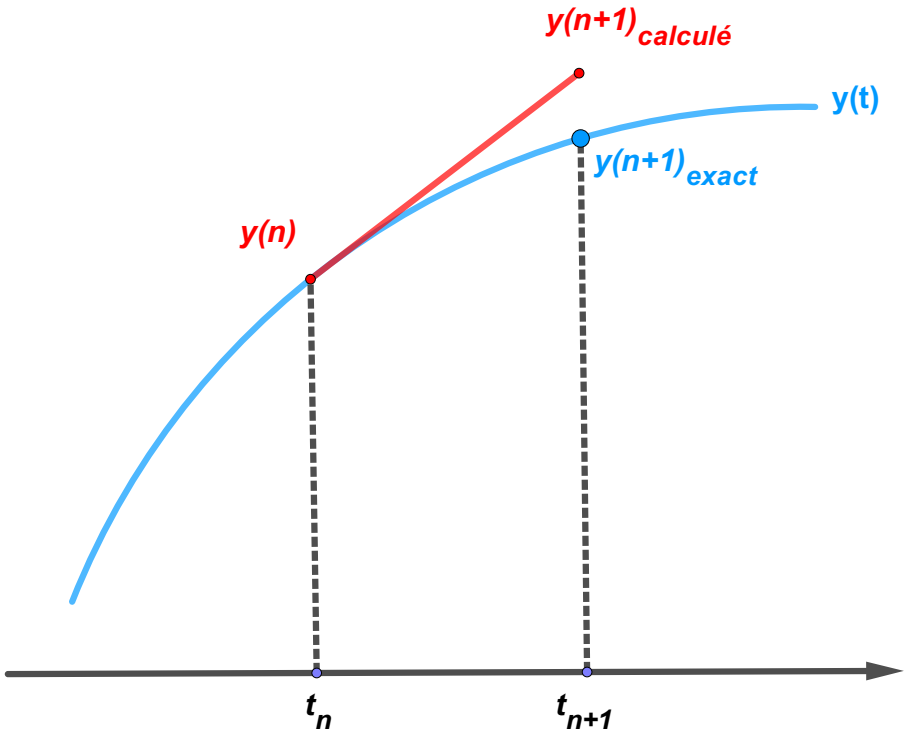


Figure 1: Principe de la méthode d'Euler

C'est une approximation de Taylor d'ordre 1. Effectivement, la prochaine valeur de y est obtenue en multipliant la dérivée de $f(t, y)$ par le pas de temps dt :

Équation
$$y_{n+1} = y_n + dt.f(t_n, y_n)$$

On a donc une erreur liée à l'approximation de Taylor :

Équation
$$y(t,n) = y(t) + dt.f(t,y) + O(dt^2)$$

Euler néglige les termes d'ordre $O(dt^2)$ pour ne tenir compte que du terme linéaire dt. Après N itérations ($N = \frac{T}{dt}$), l'erreur globale est donc proportionnelle à $O(dt)$.

4.1.4 Méthode de Heun (ordre 2)

Aussi appelée méthode du point milieu, la méthode de Heun améliore Euler, faisant une moyenne de la pente sur l'intervalle :

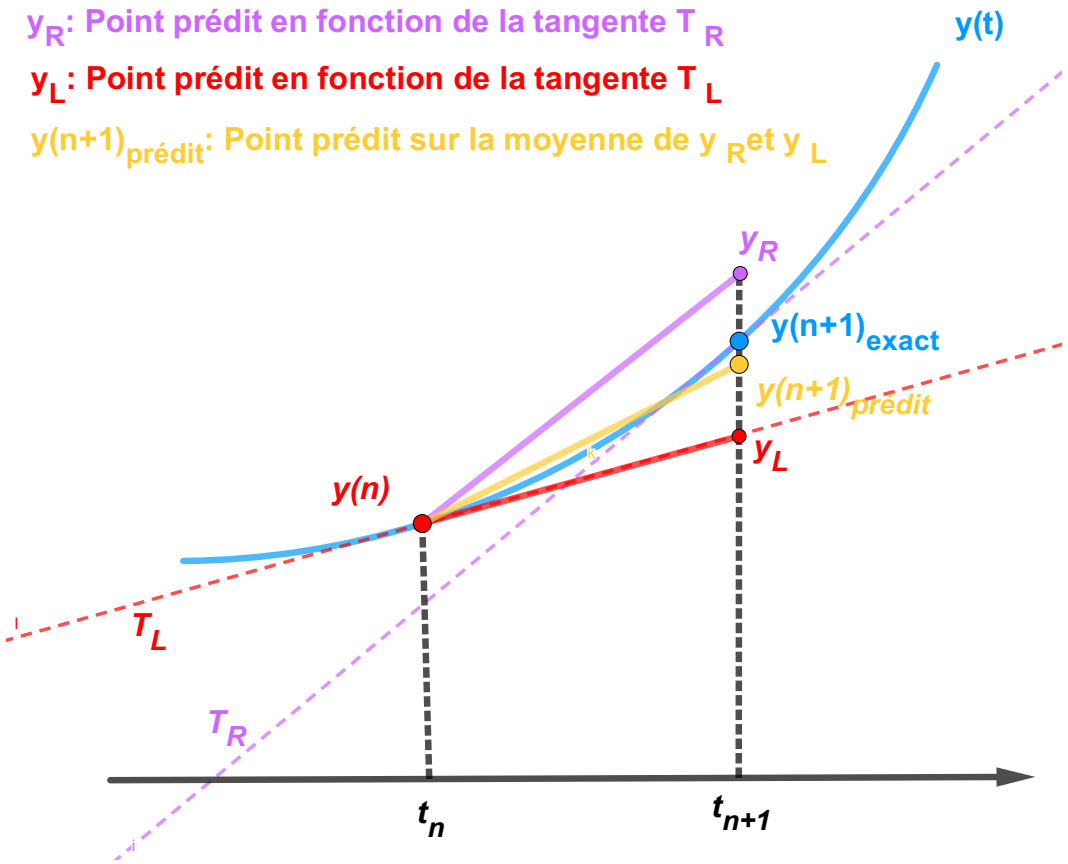


Figure 2 : Principe de la méthode de Heun

On effectuera une correction basée sur la pente de la courbe en partant de la prédiction d'Euler :

Équation
$$y_{pred} = y_n + dt.f(t_n, y_n)$$

La correction de la pente est calculée ainsi :

Équation
$$y_{n+1} = y_n + \frac{dt}{2}\left[f(t_n, y_n) + f(t_{n+1}, y_{pred})\right]$$

La méthode fait la moyenne de deux pentes, et engendre donc une meilleure approximation du développement de Taylor :

Équation
$$y(t + dt) = y(t) + dt.f(t, y) + \frac{dt^2}{2}f'(t, y) + O(dt^3)$$

Le terme d'ordre dt^2 étant pris en compte, l'erreur est réduite à $O(dt^2)$.

4.1.5 Méthode de Runge-Kutta d'ordre 4, RK4 (ordre 4)

La méthode RK4 améliore de façon drastique les méthodes précédentes. Elle évalue la pente en plusieurs points intermédiaires pour obtenir une meilleure approximation :

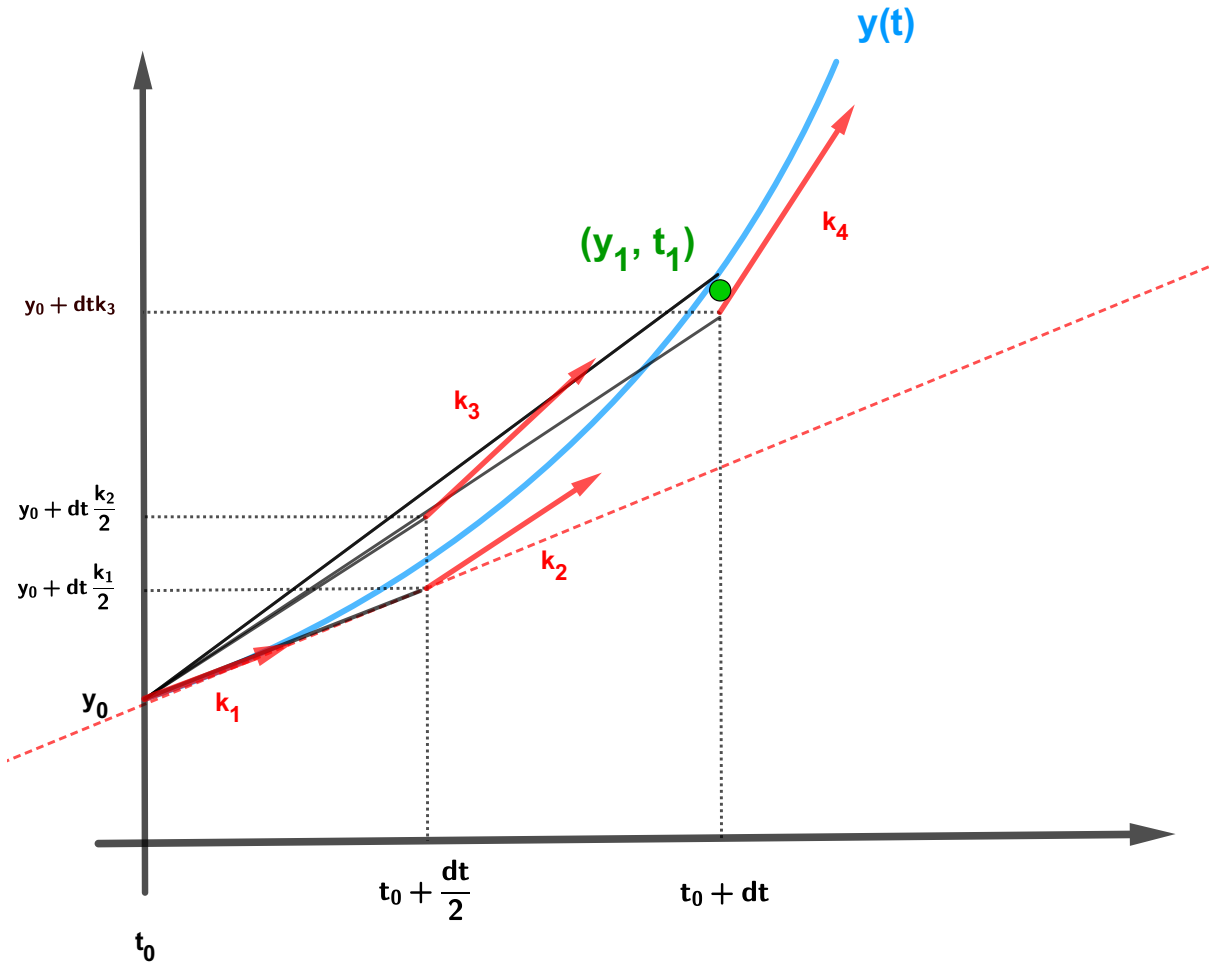


Figure 3: Principe de détermination des coefficients RK4

On effectuera 4 évaluations de f pour obtenir une approximation plus précise :

Équation
$$y(t+dt) = y(t) + dt.f(t,y) + \frac{dt^2}{2}f'(t,y) + \frac{dt^3}{6}f''(t,y) + \frac{dt^4}{24}f'''(t,y) + O(dt^5)$$

(k_1, k_2, k_3, k_4) étant les quatre estimations pondérées de la pente, l'erreur est réduite à l'ordre $O(dt^4)$.

4.1.6 Comparaison théorique des erreurs

Le tableau suivant dresse les différences entre les 3 méthodes :

Méthode	Approximation	Ordre	Pourquoi ?	Stabilité
Euler	$y_{n+1} = y_n + dt.f_n$	$O(dt)$	Néglige $O(dt^2)$	**Mauvaise**
Heun	Moyenne de 2 pentes	$O(dt^2)$	Prend en compte $O(dt^2)$	**Meilleure**
RK4	Moyenne pondérée de 4 pentes	$O(dt^4)$	Taylor d'ordre $O(dt^4)$	**Très bonne**

4.2 Étude de la convergence des méthodes numériques[3]

4.2.1 Introduction à la convergence des méthodes numériques

Une méthode numérique est convergente si, le pas de temps dt tendant vers 0, la solution de l'intégration numérique approchée y_n obtenue tend vers la solution exacte $y(t_n)$ de l'équation différentielle originale.

Cela signifie que l'erreur globale $E_n = |y(t_n) - y_n|$ tend vers 0 lorsque $dt \to 0$.

On définit l'ordre de convergence d'une méthode par la relation suivante :

Relation	Ordre
$E_n = O(dt^p)$	Où p est l'ordre de la méthode.

[3] **Godlewski, E., & Raviart, P.-A.** - *Analyse numérique des équations différentielles*. Éditions Ellipses, 1991.

Nous allons maintenant tester la convergence des 3 méthodes étudiées et vérifier si elles vérifient leurs ordres théoriques $O(dt), O(dt^2)$ et $O(dt^4)$.

4.2.2 Vérification expérimentale de la convergence

Nous allons comparer les solutions numériques à une solution analytique exacte dans un cas simplifié. Revenons à l'équation différentielle sans traînée ni variation de masse :

Équations
$$\frac{dv}{dt} = \frac{F}{m} - g$$ $Dont\ la\ solution\ analytique\ donne$: $$v(t) = v_0 + \left(\frac{F}{m} - g\right)t$$ et $$h(t) = h_0 + v_0.t + \frac{1}{2}\left(\frac{F}{m} - g\right)t^2$$

Nous allons implémenter les calculs d'altitude et de vitesse selon les 4 méthodes (analytique, Euler, Heun et RK4).

Le code Matlab est disponible en ANNEXE 1 / COMPARAISON ANALYTIQUE VS METHODES EULER, HEUN ET RK4

4.2.3 Interprétation des résultats

Nous allons tracer un graphique logarithmique de l'erreur en fonction de dt. Les pentes des courbes doivent correspondre aux ordres supposés ci-dessus :

Équation	Variables
$\log(E) = p.\log(dt) + C$	$\boldsymbol{p}$ • p vaudra 1 pour la méthode d'Euler, • p vaudra 2 pour la méthode de Heun, • p vaudra 4 pour la méthode RK4, $\boldsymbol{C}$: constante d'ajustement qui dépendra de la méthode utilisée (C représente un décalage vertical dans l'échelle logarithmique, et sa valeur sera ajustée empiriquement en ajustant une droite sur le graphique de E en fonction de dt)

Le choix d'un tracé logarithmique (afin d'avoir une vue plus claire des variations sur plusieurs ordres de grandeurs) est lié au fait qu'il permet de bien visualiser les ordres de convergence de la méthode utilisée. Sur un graphique linéaire, les courbes seraient rapprochées au début et risqueraient « d'exploser » ensuite, rendant les visuels difficilement exploitables.

Transformation logarithmique :

Équation	Variables
$E_n = C.dt^p$ $\Rightarrow \log(E_n) = p.\log(dt) + \log(C)$	p la pente de la droite $\log(C)$ une constante d'ajustement

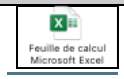
Si les résultats ne respectaient pas les tendances d'erreur, cela signifierait que :

- dt est trop grand, ce qui génère des erreurs de troncature trop importantes,
- La méthode implémentée contient une erreur,
- La précision des calculs est limitée par des erreurs d'arrondi.

4.2.3 Conclusion

Notre étude va vérifier la validité des méthodes numériques et leur ordre de convergence théorique. Nous justifierons ainsi l'utilisation de méthodes avancées telle RK4 pour une simulation qui demande une haute précision.

Dans le cas du modèle ultra simplifié (pas de variation de masse, aucune contrainte), nous disposons :

Données	Lien
Tableau Excel représentant l'erreur entre les différentes méthodes.	Double-cliquer sur l'icône 
Courbes, Analyses et Synthèses des mesures.	Annexe 2

Qui montre, pour ce cas simple, que la méthode RK4 est des 3 méthodes d'intégration numérique, celle qui se rapproche le plus de la mesure analytique.

4.3 Étude de la stabilité numérique

4.3.1 Introduction à la stabilité numérique

Outre sa convergence, la stabilité d'une méthode la rend bonne pour une utilisation en simulation : elle consiste à ce que la méthode n'amplifie pas les erreurs numériques au bout de n itérations.

Une méthode instable, va par exemple faire « exploser » ses erreurs, la rendant inutilisable. L'instabilité d'une méthode dépend de plusieurs facteurs, dont la nature de l'équation différentielle ; et en ce qui nous concerne, du pas de temps dt, voire ultérieurement de la variation d'altitude dh.

Nous étudierons donc la stabilité de nos méthodes, et évaluerons leur comportement lorsque dt varie.

4.3.2 Définition mathématique de la stabilité

Une équation différentielle de la forme :

$$\frac{dy}{dt} = \lambda y$$

Sera stable, si pour un pas dt, sa solution numérique y_n est bornée lorsque $n \to +\infty$:

$$\Leftrightarrow \forall n \in \mathbb{N}, |y_n| \leq +\infty$$

Sinon, elle est instable.

4.3.3 Analyse de la stabilité pour les méthodes numériques

4.3.3.1 Stabilité de la méthode d'Euler

La méthode d'Euler s'écrit :

Équations
$$y_{n+1} = y_n + dt.f(t_n, y_n)$$
$Considérons$:
$$\lambda y = \frac{dy}{dt}$$
$$\Rightarrow \lambda y_n = \frac{dy_n}{dt}$$
Et
$$f(t_n, y_n) = \frac{dy_n}{dt}$$
$Alors$
$$y_{n+1} = y_n + dt\frac{dy_n}{dt}$$
$$\Leftrightarrow y_{n+1} = y_n + dt.\lambda y_n$$
$$\Leftrightarrow y_{n+1} = (1 + \lambda dt)y_n$$

C'est une suite géométrique de raison $1 + \lambda dt$. L'erreur se propage de la même façon. La méthode sera donc stable si :

Condition

$$|1 + \lambda dt| < 1$$

$$\Rightarrow dt < \frac{2}{|\lambda|}$$

Si le pas dt choisi est trop grand, la méthode d'Euler sera instable et l'erreur explosera.

4.3.3.2 Stabilité de la méthode de Heun

La méthode de Heun s'écrit :

Équations

$$y_{n+1} = y_n + \frac{dt}{2}(f_n + f_{n+1})$$

Avec un raisonnement analogue à la méthode d'Euler, on obtient :

$$y_{n+1} = y_n + \frac{dt}{2}(\lambda y_n + \lambda y_{n+1})$$

$$\Leftrightarrow y_{n+1} - \frac{dt}{2}\lambda y_{n+1} = y_n + \frac{dt}{2}\lambda y_n$$

$$\Leftrightarrow \left(1 - \frac{dt}{2}\lambda\right) y_{n+1} = \left(1 + \frac{dt}{2}\lambda\right) y_n$$

$$\Leftrightarrow y_{n+1} = \frac{1 + \frac{dt}{2}\lambda}{1 - \frac{dt}{2}\lambda} y_n \qquad (suite\ géométrique)$$

Ce qui implique la condition de stabilité suivante :

Condition

$$\left|\frac{1+\frac{dt}{2}\lambda}{1-\frac{dt}{2}\lambda}\right| < 1$$

$$\Rightarrow dt < \frac{4}{|\lambda|}$$

La contrainte est donc moins stricte que pour la méthode d'Euler (le pas de temps est 2 fois plus grand). Heun est plus stable et peut donc supporter des valeurs de dt plus grandes avant de devenir instable.

4.3.3.3 Stabilité de la méthode RK4

La méthode RK4 calcule 4 pentes intermédiaires :

Équations

$$k_1 = f(t_n, y_n)$$

$$k_2 = f\left(t_n + \frac{dt}{2}, y_n + \frac{dt}{2}k_1\right)$$

$$k_3 = f\left(t_n + \frac{dt}{2}, y_n + \frac{dt}{2}k_2\right)$$

$$k_4 = f(t_n + dt, y_n + dtk_3)$$

$$y_{n+1} = y_n + \frac{dt}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

En partant de $\lambda y = \frac{dy}{dt}$, on obtient :

Équation
$$y_{n+1} = G(dt, \lambda).y_n$$

Où $G(dt, \lambda)$ est un facteur qui dépend de dt et λ.

$G(dt, \lambda)$ peut être approximé par un développement de Taylor (ou de Maclaurin [4]) :

Équation
$$G(dt, \lambda) = 1 + \lambda dt + \frac{(\lambda dt)^2}{2!} + \frac{(\lambda dt)^3}{3!} + \frac{(\lambda dt)^4}{4!} + O((\lambda dt)^5)$$

Ce développement est à peu de choses près celui de l'exponentielle :

$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!}$$

D'où :

Équation
$$e^{\lambda dt} = 1 + \lambda dt + \frac{(\lambda dt)^2}{2!} + \frac{(\lambda dt)^3}{3!} + \frac{(\lambda dt)^4}{4!} + O((\lambda dt)^5)$$ $Donc$: $$G(dt, \lambda) \approx e^{\lambda dt}$$

On voit alors que RK4 suit le développement de l'exponentielle jusqu'à l'ordre 4, ce qui est bien meilleur que les méthodes d'Euler ou Heun.

Ce qui implique la condition de stabilité suivante :

[4] **Source** : https://fr.wikipedia.org/wiki/S%C3%A9rie_de_Taylor#D.C3.A9veloppements_en_s.C3.A9rie_de_Maclaurin_des_fonctions_usuelles

Condition
$\left\|e^{\lambda dt}\right\| \leq 1$ $Implique, pour\ \lambda \in \mathbb{R}$: $\lambda dt \leq 0$

Cela signifie que :

- Si $\lambda < 0$ (cas d'un système dissipatif – l'énergie totale diminue au fil du temps, lié à différentes forces de dissipation telle la friction ou autre – avec une équation de type $\frac{dy}{dt} = -10\lambda$), alors RK4 est toujours stable pour tout dt.
- Si $\lambda > 0$, alors RK4 devient instable pour de grands dt , mais reste beaucoup plus stable que les 2 autres méthodes.

Contrairement à Euler (qui impose $dt < \frac{2}{|\lambda|}$) et Heun ($dt < \frac{4}{|\lambda|}$), RK4 tolère un pas plus important avant de devenir instable. En pratique, RK4 est stable pour des pas environ 10 fois plus grands que pour les 2 autres méthodes[5].

4.3.4 Pour complexifier : Influence de la traînée variable $\rho(h)$ sur la stabilité

Précédemment, notre analyse de stabilité a porté sur des méthodes numériques en fonction de dt appliquées à des EDO. Dans notre cas particulier de la trajectoire d'une fusée, une nouvelle force intervient, qui dépend de l'altitude : la force de traînée aérodynamique. Celle-ci dépend de la densité de l'air $\rho(h)$ qui varie selon h.

La densité de l'air est donnée par la formule suivante :

Formule	Variables
$\rho(h) = \rho_0 e^{-\frac{h}{H}}$	ρ_0 est la densité de l'air au niveau de la mer, H est l'échelle de hauteur atmosphérique (8km sur Terre).

La force de traînée intervenant directement dans l'équation de mouvement, sa variation rapide avec l'altitude pourrait affecter la stabilité numérique des méthodes choisies.

[5] J. C. Butcher, Numerical Methods for Ordinary Differential Equations, Wiley, 2008.

Nous testerons la stabilité au travers de l'équation différentielle du mouvement pour la vitesse v de la fusée qui devient :

Équation
$$\frac{dv}{dt} = \frac{F}{m} - g - \frac{1}{2} C_d . \rho(h) . A . v^2$$

Nous :

- Simulerons la trajectoire pour les 3 méthodes pour différents pas dt,
- Comparerons la stabilité des méthodes en observant la cohérence de la vitesse et de l'altitude,
- Testerons la stabilité d'Euler pour certaines valeurs de dt du fait de la dépendance de $\rho(h)$,
- Observerons si RK4 est meilleure.

Attendus et hypothèses :

- Euler pourrait devenir instable en cas de forte variation de $\rho(h)$, notamment en descente où la traînée augment brusquement,
- Heun devrait mieux gérer les variations
- RK4 étant d'ordre 4 devrait encore mieux gérer les variations, voire rester stable.

4.3.5 Effet des variations brusques sur la stabilité numérique

Les méthodes numériques partent du principe que la fonction est régulière, dérivable, avec des dérivées continues. Mais lorsque la poussée motrice s'arrête, l'accélération chute brutalement ce qui affecte la méthode.

Effectivement, les méthodes numériques sont basées sur le développement de Taylor qui suppose une régularité de la fonction $f(t, y)$ et de ses dérivées. Une variation brutale, discontinue de l'accélération la rend irrégulière. Les méthodes peuvent alors devenir instables.

4.3.6 Conclusion

Nous avons examiné si la stabilité des méthodes est impactée par la force de traînée qui dépend de l'altitude. Si RK4 sort vainqueur, cela confirmera son utilisation pour les simulations de trajectoire de fusée.

De plus, nous serions tentés d'étendre l'analyse en intégrant l'accélération gravitationnelle g qui dépend également de l'altitude ($g(h) = g_0.\left(\frac{R_T}{R_T+h}\right)$).

5 Comparaison Euler vs Heun vs RK4

5.1 Comparaison visuelle des trajectoires Heun vs RK4

5.1.1 Introduction

Nous allons maintenant comparer les performances des méthodes en traçant les trajectoires obtenues dans le cadre de la simulation de la fusée.

5.1.2 Méthodologie

Nous allons comparer les résultats issus de trois équations de mouvement montrant une complexité progressive :

5.1.2.1 Modèle simplifié : équation différentielle linéaire du premier ordre

Dans un premier temps, le modèle étudié négligera la force de traînée et la variation de masse ne sera pas prise en compte. L'équation est donc :

Équation
$m\frac{dv}{dt} = F - mg$ (*Cf. : Modèle Simplifié*)

La résolution de cette équation permettra d'avoir une première approximation de la trajectoire, et de tester la convergence des méthodes sur un cas simple.

5.1.2.2 Modèle intermédiaire : équation du mouvement ne variant que dans le temps

Au travers de ce modèle, seule la variation de masse est ajoutée au premier modèle. La masse, devient en fonction du temps :

Équation
$$m(t) = m_0 - \dot{m}.t$$

L'équation de mouvement devient alors :

Équation
$$(m_0 - \dot{m}.t)\frac{dv}{dt} = F - (m_0 - \dot{m}.t)g$$ (*Cf. : Modèle amélioré*)

Cette équation, une fois résolue permettra d'observer l'influence de la variation de masse sur la trajectoire et sur les méthodes d'intégration.

5.1.2.3 Modèle complet : équation du mouvement dépendant du temps et de l'altitude

Ce modèle plus réaliste introduit l'effet de la traînée aérodynamique et de la gravité qui dépendent de l'altitude. L'équation devient donc :

Équation
$(m_0 - \dot{m}.t)\frac{dv}{dt} = F - (m_0 - \dot{m}.t)g(h) - \frac{1}{2}C_d\rho(h).A.v^2$ (*Cf. :Traînée et Gravité*) $Avec$: $$\frac{dh}{dt} = v$$

Ce modèle permet d'observer les différences de trajectoire en tenant compte de plusieurs phénomènes physiques importants.

5.1.3 Comparaisons

Nous allons :

- Tracer les trajectoires selon les 3 méthodes d'Euler, Heun et RK4,
- Observer l'influence des paramètres dynamiques sur les 3 méthodes,
- Tester plusieurs pas de temps dt pour voir comment cela impacete les résultats.

5.1.4 Résultats attendus

Nous formulons les conjectures suivantes :

- La méthode d'Euler va probablement avoir une trajectoire décalée et instable pour de grandes valeurs de dt,
- Heun sera sûrement plus précise, avec des écarts si dt était trop grand,
- RK4, d'ordre 4, devrait montrer des trajectoires plus proches de la réalité, même pour un grand dt.

5.1.5 Algorithmes :

Les 3 algorithmes sont disponibles en Annexe :

Méthode	**Algorithme**
Euler	Annexe 1 / MODELE SIMPLIFIE
Heun	Annexe 1 / MODELE AVANCE
RK4	Annexe 1 / MODELE COMPLET

5.1.6 Courbes & Données :

Type	**Lien**
Calculs de variables (h, v, a, etc.)	Annexe 3 - Courbes
Données Excel	Double-cliquer sur l'icône 

5.2 Calcul de l'erreur entre Euler, Heun et RK4

5.2.1 Introduction

Après avoir tracé les trajectoires, nous allons maintenant effectuer les calculs d'erreurs entre les 3 méthodes dans le but de mesurer l'erreur relative entre chacune. Cela nous permettra d'évaluer leur précision.

Nous comparerons :

- Euler vs Heun afin de constater ou non l'amélioration de la précision sur une méthode d'ordre 2,
- Euler vs RK4 lors d'un passage de l'ordre 1 à l'ordre 4,
- Heun vs RK4 pour évaluer si le gain de l'ordre 2 à l'ordre 4 est significatif.

La comparaison se fera sur l'altitude, la vitesse et l'accélération.

5.2.2 Méthodologie

Le calcul de l'erreur relative entre la méthode A et la méthode B pour la mesure M se calcule de la sorte :

Formule
$$Erreur\ relative = \frac{\lvert M_{MéthodeA} - M_{Méthode\ B}\rvert}{M_{Méthode\ B}} * 100$$

Nous allons donc :

- Calculer les erreurs entre les 3 méthodes sur les 3 modèles,
- Tracer les courbes d'erreur en fonction du temps pour chaque mesure (altitude, vitesse, accélération),
- Exporter les résultats sous Excel.

5.2.3 Résultats attendus

Nous formulons les conjectures suivantes :

- La méthode d'Euler devrait montrer des erreurs significatives comparé à Heun ou RK4,

- Heun devrait approcher RK4,
- RK4 étant d'ordre 4, ses erreurs devraient être minimales.

5.2.4 Algorithme :

Méthode	Algorithme
Valable pour toutes	Annexe 1 - CALCUL D'ERREUR

5.2.5 Courbes & Données :

Type	Lien
Synthèse & Courbes Modèle Simplifié	Annexe 4 – Synthèse - Modèle Simplifié
Synthèse & Courbes Modèle Avancé	Annexe 4 – Synthèse – Modèle Avancé
Synthèse & Courbes Modèle Complet	Annexe 4 – Synthèse – Modèle Complet
Données Excel	Double-cliquer sur l'icône 

5.2.6 Synthèse des résultats

L'analyse des erreurs montre des éléments importants :

- Euler est très imprécise pour tous les modèles, surtout lorsque dt est grand ($dt = 0{,}1s$). Son erreur devient trop importante sur la vitesse et l'accélération.
- Heun est une bonne approximation de RK4, mais avec un écart d'environ 4% à 5% sur l'altitude et la vitesse pour $dt = 0{,}1s$. Cet écart diminue fortement à $dt = 0{,}01s$.
- RK4 maximise les résultats, avec une erreur minimale. Il est cependant sensible aux transitions (poussée/chute libre).
- L'erreur diminue plus dt est petit : on observe une forte amélioration lorsque l'on passe de $dt = 0{,}1s$ à $dt = 0{,}05s$, puis à $dt = 0{,}01s$. À $dt = 0{,}01s$, Heun devient une très bonne alternative à RK4, avec une erreur de l'ordre de 1% à 2% seulement.
- Les erreurs explosent autour de $t \approx 60s$, en raison de la transition entre la phase de poussée et la chute libre. Cette instabilité est particulièrement marquée pour Euler et Heun. Même RK4 est impactée, bien que d'ordre 4 ; cela confirme ce que nous évoquions plus haut : une transition lors de laquelle la fonction n'est plus régulière engendre une augmentation significative de l'erreur.

Cette étude met en évidence l'impact crucial du choix du pas et de la méthode numérique sur la précision de la simulation.

5.3 Influence du pas dt sur les écarts entre les méthodes

5.3.1 Introduction

Après avoir analysé les erreurs entre les méthodes numériques, nous nous intéressons maintenant à l'influence du pas de temps dt sur ces écarts. L'objectif est de déterminer à partir de quelle valeur de dt les erreurs deviennent acceptables et quelle méthode reste stable même pour des pas plus grands.

Nous allons comparer les écarts relatifs entre Euler, Heun et RK4 pour plusieurs valeurs de dt :

- $dt = 0{,}1s$
- $dt = 0{,}05s$
- $dt = 0{,}01s$

Cette analyse permettra d'identifier le meilleur compromis entre précision et coût de calcul.

5.3.2 Influence du pas dt sur l'altitude

dt	Observations
0,1s	Euler a une erreur supérieure à 10% en fin de vol, et devient inutilisable. Heun est meilleur, mais présente un écart de 4-5% avec RK4.
0,05s	L'erreur Euler-RK4 diminue mais reste importante (~5-6%). Heun s'améliore avec une erreur de 3-4%.
0,01s	L'erreur Euler-RK4 devient inférieure à 3%, et Heun se stabilise à 1-2%, c'est une bonne alternative à RK4.

Conclusion : À $dt = 0{,}01s$, les écarts sont acceptables, et Heun est une bonne alternative à RK4.

5.3.3 Influence du pas dt sur la Vitesse

dt	Observations
0,1s	Les erreurs Euler-RK4 dépassent 15%, et Heun-RK4 reste élevé (~5%).
0,05s	On observe une certaine amélioration, mais Euler est instable.

0,01s	Les erreurs sont très faibles (<3%), Heun est proche de RK4.

Conclusion : Pour un suivi précis de la vitesse, il vaut mieux choisir $dt \leq 0{,}05s$.

5.3.4 Influence du pas dt sur l'accélération

dt	**Observations**
0,1s	Les erreurs Euler-RK4 explosent (>100%) ; Euler est inutilisable.
0,05s	Les erreurs baissent mais sont encore élevées (~20-30%).
0,01s	Heun-RK4 se stabilise sous 5% et Euler-RK4 sous 10%.

Conclusion : Seul $dt = 0{,}01s$ permet d'avoir une bonne stabilité sur l'accélération.

5.3.5 Synthèse et recommandations finales

L'analyse des erreurs en fonction de dt nous amène à conclure que :

- Euler est inutile pour $dt < 0{,}05s$, surtout pour la vitesse et l'accélération.
- Heun est un très bon compromis dès que $dt = 0{,}01s$, avec une erreur relative inférieure à 2%.
- RK4 est la méthode la plus fiable. Cependant, pour un coût de calcul plus faible, Heun est une bonne alternative.
- Un pas $dt = 0{,}01s$ est recommandé pour des simulations précises et stables.

6 Conclusion et perspectives

6.1 Synthèse des principaux résultats

A travers ce projet, nous avons étudié et comparé 3 méthodes numériques d'intégration d'équations différentielles pour la trajectoire d'une fusée. Partis d'un modèle simplifié, nous avons ajouté en deux temps des forces afin d'atteindre un certain réalisme tel que la variation de masse et la traînée aérodynamique ; le modèle complet intégrant la gravité et résistance de l'air en fonction de l'altitude.

L'analyse des altitudes, vitesses et accélérations, combinée à celle des erreurs entre les méthodes d'Euler, Heun et RK4 a montré que :

- Euler est une méthode simple mais imprécise ; elle devient instable pour les grands pas de temps.

- Heun est un bon compromis entre précision et coût de calcul ; les pas de temps doivent tout de même rester sobres.
- RK4 est la méthode la plus précise, même pour des pas plus grands.
- D'importantes erreurs numériques apparaissent dans toutes les méthodes dès qu'il y a de fortes variations d'accélération.
- Plus le pas dt est petit, meilleurs sont les résultats, tant en coûts de calculs que de stabilité.

Enfin, certaines phases du vol (comme la coupure de moteur entraînant une brusque variation de l'accélération) perturbent les méthodes numériques ; ceci pourrait justifier l'usage de pas adaptatifs (réduction automatique de dt) ou de méthodes spécifiques capables de gérer ces changements de régime.

6.2 Limites de l'étude et pistes d'amélioration

Bien que notre étude ait fourni des résultats intéressants, certaines limitations se sont présentées :

- Les forces latérales et la rotation ne sont pas prises en compte.
- Les effets du vent et des perturbations atmosphériques n'ont pas été inclus.
- L'analyse est limitée à des trajectoires verticales.
- Les calculs d'erreur ont été faits sans solution analytique de référence pour le modèle complet.

Il serait intéressant de prolonger cette étude en intégrant les 3 dimensions spatiales, et en menant une analyse plus approfondie de la stabilité numérique.

6.3 Ouverture sur d'autres analyses possibles

Les modèles étudiés restent simples. Pour approfondir cette étude, il serait intéressant d'envisager d'intégrer :

- La poussée vectorielle grâce à un moteur orientable,
- L'influence du vent sur la trajectoire,
- L'ajout de la force de Coriolis,
- La comparaison avec des données de vol d'une fusée réelle.

Passionné d'aéronautique, cette première étude m'a enchanté et m'invite à découvrir ces nombreuses pistes ! À titre personnel, elle m'a ouvert les yeux à plus d'un titre sur de nombreux phénomènes et leurs conséquences. Plus encore, elle renforce mon souhait de faire des études en aéronautique.

Vous pouvez retrouver la simulation visuelle JavaScript (en cours de développement) de deux modèles de fusées sur ma page https://padr.fr/rocket .

Travaux étudiés

Meyer, Y. : ***Analyse numérique*** **(Tome 1). Éditions Dunod, 2012.**

Godlewski, E., & Raviart, P.-A. : ***Analyse numérique des équations différentielles*****. Éditions Ellipses, 1991.**

Blanchard, P., Devaney, R.L., & Hall, G.R. : ***Equations différentielles*****. Éditions De Boeck, 2006.**

Guitart, D. : ***Initiation aux équations différentielles et à leurs applications*****. Éditions Ellipses, 2015.**

Siegmund, S., Lemoine, C. : ***Méthodes numériques pour les équations différentielles ordinaires*****. Dunod, 2019.**

ESA – Agence Spatiale Européenne : ***Lancer une fusée – Guide pédagogique pour l'enseignement secondaire*****, 2016.**

Tsiolkovski, K.E. : ***Exploration de l'espace cosmique par des engins à réaction*****, 1903.**

Butcher J. C. : ***Numerical Methods for Ordinary Differential Equations*****, Wiley, 2008.**

ANNEXES

Annexe 1 : CODE INFORMATIQUE

Comparaison analytique vs méthodes Euler, Heun et RK4

```
function simulate_ultra_simple_model_vs_analytic(dt)
    % Paramètres physiques
    g = 9.81;         % Gravité (m/s²)
    F = 5000;         % Poussée (N) appliquée pendant 50 secondes
    m = 150;          % Masse constante (100 kg vide + 50 kg carburant)
    burn_time = 50;   % Durée de la poussée (s)

    % Définition du répertoire de sauvegarde
    base_dir = 'Ultra Simple Model';
    dt_dir = sprintf('Pas-%.3f', dt);
    full_path = fullfile(base_dir, dt_dir);

    % Création des répertoires si non existants
    if ~exist(full_path, 'dir')
        mkdir(full_path);
    end

    % Conditions initiales
    t = 0;
    s_Euler = 0; s_Heun = 0; s_RK4 = 0;
    alt_Euler = 0; alt_Heun = 0; alt_RK4 = 0;

    % Stockage des résultats
    time = [];
    speed_Euler = []; speed_Heun = []; speed_RK4 = []; speed_analytic = [];
    altitude_Euler = []; altitude_Heun = []; altitude_RK4 = []; altitude_analytic = [];
    error_speed_EA = []; error_speed_HA = []; error_speed_RA = [];
    error_alt_EA = []; error_alt_HA = []; error_alt_RA = [];

    % Simulation jusqu'à retour à altitude 0
    while alt_Euler >= 0
        % Enregistrement du temps
        time = [time; t];

        % Détermination de la poussée en fonction du temps
        if t < burn_time
            F_thrust = F;
```

```
else
    F_thrust = 0;
end

% Solution analytique
if t < burn_time
    s_exact = (F/m) * t - g * t;
    alt_exact = (F/(2*m)) * t^2 - (g/2) * t^2;
else
    t_burnout = burn_time;
    s_burnout = (F/m) * t_burnout - g * t_burnout;
    alt_burnout = (F/(2*m)) * t_burnout^2 - (g/2) * t_burnout^2;

    s_exact = s_burnout - g * (t - t_burnout);
    alt_exact = alt_burnout + s_burnout * (t - t_burnout) - (g/2) * (t - t_burnout)^2;
end

% Enregistrement des valeurs analytiques
speed_analytic = [speed_analytic; s_exact];
altitude_analytic = [altitude_analytic; alt_exact];

% Intégration par méthode d'Euler
acc_Euler = (F_thrust / m) - g;
s_Euler = s_Euler + dt * acc_Euler;
alt_Euler = alt_Euler + dt * s_Euler;

% Intégration par méthode de Heun (double application)
s_pred = s_Heun + dt * acc_Euler;

% Déterminer la poussée prédite pour Heun
if t + dt < burn_time
    F_thrust_pred = F;
else
    F_thrust_pred = 0;
end

acc_pred = (F_thrust_pred / m) - g;
s_Heun = s_Heun + (dt/2) * (acc_Euler + acc_pred);

alt_pred = alt_Heun + dt * s_pred;
alt_Heun = alt_Heun + (dt/2) * (s_Heun + s_pred);

% Intégration par méthode RK4 (double application)
k1_v = dt * ((F_thrust / m) - g);
k2_v = dt * (((F_thrust_pred / m) - g) + k1_v / 2);
k3_v = dt * (((F_thrust_pred / m) - g) + k2_v / 2);
k4_v = dt * (((F_thrust_pred / m) - g) + k3_v);
s_RK4 = s_RK4 + (1/6) * (k1_v + 2*k2_v + 2*k3_v + k4_v);

k1_h = dt * s_RK4;
k2_h = dt * (s_RK4 + k1_h / 2);
```

```
        k3_h = dt * (s_RK4 + k2_h / 2);
        k4_h = dt * (s_RK4 + k3_h);
        alt_RK4 = alt_RK4 + (1/6) * (k1_h + 2*k2_h + 2*k3_h + k4_h);

        % Enregistrement des valeurs numériques
        speed_Euler = [speed_Euler; s_Euler];
        speed_Heun = [speed_Heun; s_Heun];
        speed_RK4 = [speed_RK4; s_RK4];
        altitude_Euler = [altitude_Euler; alt_Euler];
        altitude_Heun = [altitude_Heun; alt_Heun];
        altitude_RK4 = [altitude_RK4; alt_RK4];

        % Calcul des erreurs relatives par rapport à l'analytique
        error_speed_EA = [error_speed_EA; abs(s_Euler - s_exact) / max(abs(s_exact), 1e-6) *
100];
        error_speed_HA = [error_speed_HA; abs(s_Heun - s_exact) / max(abs(s_exact), 1e-6) *
100];
        error_speed_RA = [error_speed_RA; abs(s_RK4 - s_exact) / max(abs(s_exact), 1e-6) *
100];

        error_alt_EA = [error_alt_EA; abs(alt_Euler - alt_exact) / max(abs(alt_exact), 1e-6) * 100];
        error_alt_HA = [error_alt_HA; abs(alt_Heun - alt_exact) / max(abs(alt_exact), 1e-6) * 100];
        error_alt_RA = [error_alt_RA; abs(alt_RK4 - alt_exact) / max(abs(alt_exact), 1e-6) * 100];

        % Mise à jour du temps
        t = t + dt;
    end

    % Création des graphiques d'erreur
    figure;
    subplot(2,1,1);
    plot(time, error_alt_EA, 'r', time, error_alt_HA, 'b', time, error_alt_RA, 'g', 'LineWidth', 1.5);
    xlabel('Temps (s)');
    ylabel('Erreur (%)');
    title("Erreur sur l'altitude (Comparaison avec l'analytique)");
    legend('Euler-Analytique', 'Heun-Analytique', 'RK4-Analytique');
    grid on;

    subplot(2,1,2);
    plot(time, error_speed_EA, 'r', time, error_speed_HA, 'b', time, error_speed_RA, 'g',
'LineWidth', 1.5);
    xlabel('Temps (s)');
    ylabel('Erreur (%)');
    title("Erreur sur la vitesse (Comparaison avec l'analytique)");
    legend('Euler-Analytique', 'Heun-Analytique', 'RK4-Analytique');
    grid on;

    % Sauvegarde des résultats et des graphiques
    filename = fullfile(full_path, sprintf('UltraSimpleModel_Erreurs-vs-Analytic-pas%.3f.xlsx', dt));
    writetable(table(time, error_alt_EA, error_alt_HA, error_alt_RA, error_speed_EA,
error_speed_HA, error_speed_RA), filename);
```

```
    saveas(gcf, fullfile(full_path, 'Erreurs_Altitude_Vitesse.png'));
    saveas(gcf, fullfile(full_path, 'Erreurs_Altitude_Vitesse.svg'));

    disp(['Résultats enregistrés dans : ', full_path]);
end
```

S’appelle via la commande :

simulate_ultra_simple_model_vs_analytic(<le pas choisi>)

Modèle Simplifié

```
function simulate_simplified_model(dt)
    % Paramètres physiques
    g = 9.81;         % Gravité (m/s²) supposée constante
    F = 5000;         % Poussée (N) appliquée pendant 50 secondes (consommation de 1kg/s)
    m = 150;          % Masse totale de la fusée (100 kg vide + 50 kg carburant)
    A = 1;            % Surface frontale (m²)
    Cd = 0.5;         % Coefficient de traînée
    rho = 1.225;      % Densité de l'air (kg/m³) constante

    % Conditions initiales
    t = 0;
    s_Euler = 0; s_Heun = 0; s_RK4 = 0;     % Vitesses selon les 3 méthodes
    alt_Euler = 0; alt_Heun = 0; alt_RK4 = 0;     % Altitude selon les 3 méthodes

    % Stockage des résultats
    time = []; speed_Euler = []; speed_Heun = []; speed_RK4 = [];
    altitude_Euler = []; altitude_Heun = []; altitude_RK4 = [];
    acceleration_Euler = []; acceleration_Heun = []; acceleration_RK4 = [];
    thrust = []; gravity = []; mass = []; drag_force = [];

    % Simulation jusqu'à retour à altitude 0
    while alt_Euler >= 0
        % Enregistrement des données
        time = [time; t];
        speed_Euler = [speed_Euler; s_Euler];
        speed_Heun = [speed_Heun; s_Heun];
        speed_RK4 = [speed_RK4; s_RK4];
        altitude_Euler = [altitude_Euler; alt_Euler];
        altitude_Heun = [altitude_Heun; alt_Heun];
        altitude_RK4 = [altitude_RK4; alt_RK4];
        gravity = [gravity; g];
        mass = [mass; m];
        thrust = [thrust; (t < 50) * F];

        % Calcul de la traînée pour chaque méthode
        drag_Euler = 0.5 * Cd * rho * A * s_Euler^2 * sign(s_Euler);
        drag_Heun = 0.5 * Cd * rho * A * s_Heun^2 * sign(s_Heun);
        drag_RK4 = 0.5 * Cd * rho * A * s_RK4^2 * sign(s_RK4);
        drag_force = [drag_force; drag_Euler];

        % Calcul des accélérations pour chaque méthode
        acc_Euler = ((t < 50) * F - m * g - drag_Euler) / m;
        acc_Heun = ((t < 50) * F - m * g - drag_Heun) / m;
        acc_RK4 = ((t < 50) * F - m * g - drag_RK4) / m;

        acceleration_Euler = [acceleration_Euler; acc_Euler];
        acceleration_Heun = [acceleration_Heun; acc_Heun];
        acceleration_RK4 = [acceleration_RK4; acc_RK4];

        % Intégration par méthode d'Euler
        s_Euler = s_Euler + dt * acc_Euler;
        alt_Euler = alt_Euler + dt * s_Euler;
```

```
        % Intégration par méthode de Heun
        s_pred = s_Heun + dt * acc_Heun;
        acc_pred = ((t + dt < 50) * F - m * g - drag_Heun) / m;
        s_Heun = s_Heun + (dt/2) * (acc_Heun + acc_pred);
        alt_Heun = alt_Heun + (dt/2) * (s_Heun + s_pred);

        % Sauvegarde de la nouvelle accélération pour Heun après correction
        acc_Heun = ((t < 50) * F - m * g - 0.5 * Cd * rho * A * s_Heun^2 * sign(s_Heun)) / m;
        acceleration_Heun(end) = acc_Heun;

        % Intégration par méthode RK4
        k1_v = dt * acc_RK4;
        k2_v = dt * (((t + dt/2 < 50) * F - m * g - drag_RK4) / m);
        k3_v = dt * (((t + dt/2 < 50) * F - m * g - drag_RK4) / m);
        k4_v = dt * (((t + dt < 50) * F - m * g - drag_RK4) / m);

        s_RK4 = s_RK4 + (1/6) * (k1_v + 2*k2_v + 2*k3_v + k4_v);

        k1_h = dt * s_RK4;
        k2_h = dt * (s_RK4 + k1_h/2);
        k3_h = dt * (s_RK4 + k2_h/2);
        k4_h = dt * (s_RK4 + k3_h);

        alt_RK4 = alt_RK4 + (1/6) * (k1_h + 2*k2_h + 2*k3_h + k4_h);
        %alt_RK4 = alt_RK4 +s_RK4*dt;

        % Sauvegarde de la nouvelle accélération pour RK4 après correction
        acc_RK4 = ((t < 50) * F - m * g - 0.5 * Cd * rho * A * s_RK4^2 * sign(s_RK4)) / m;
        acceleration_RK4(end) = acc_RK4;

        % Mise à jour du temps
        t = t + dt;
    end

    % Sauvegarde des résultats avec la nouvelle fonction
    save_results("Simplified Model", dt, time, altitude_Euler, altitude_Heun, altitude_RK4, ...
                 speed_Euler, speed_Heun, speed_RK4, ...
                 acceleration_Euler, acceleration_Heun, acceleration_RK4, ...
                 thrust, gravity, mass, drag_force);
end
```

S’appelle via la commande :

simulate_simplified_model(<le pas choisi>)

Modèle Avancé

```
function simulate_advanced_model(dt)
    % Paramètres physiques
    g = 9.81;         % Gravité (m/s²) supposée constante
    F = 5000;         % Poussée (N) appliquée tant qu'il y a du carburant
    m_empty = 100;    % Masse à vide de la fusée (kg)
    m_fuel = 50;      % Masse initiale du carburant (kg)
    dm = 1;           % Débit de consommation du carburant (kg/s)
    A = 1;            % Surface frontale (m²)
    Cd = 0.5;         % Coefficient de traînée
    rho = 1.225;      % Densité de l'air (kg/m³) constante

    % Conditions initiales
    t = 0;
    s_Euler = 0; s_Heun = 0; s_RK4 = 0;
    alt_Euler = 0; alt_Heun = 0; alt_RK4 = 0;
    m = m_empty + m_fuel; % Masse initiale de la fusée

    % Stockage des résultats
    time = []; speed_Euler = []; speed_Heun = []; speed_RK4 = [];
    altitude_Euler = []; altitude_Heun = []; altitude_RK4 = [];
    acceleration_Euler = []; acceleration_Heun = []; acceleration_RK4 = [];
    thrust = []; gravity = []; mass = []; drag_Euler = []; drag_Heun = []; drag_RK4 = [];

    % Simulation jusqu'à retour à altitude 0
    while alt_Euler >= 0
        % Mise à jour de la masse (tant qu'il y a du carburant)
        if m > m_empty
            m = max(m_empty, m - dm * dt);
            current_thrust = F;
        else
            current_thrust = 0; % Plus de poussée une fois le carburant épuisé
        end

        % Calcul des forces de traînée pour chaque méthode
        drag_Euler_val = 0.5 * Cd * rho * A * s_Euler^2 * sign(s_Euler);
        drag_Heun_val  = 0.5 * Cd * rho * A * s_Heun^2 * sign(s_Heun);
        drag_RK4_val   = 0.5 * Cd * rho * A * s_RK4^2 * sign(s_RK4);

        % Calcul des accélérations pour chaque méthode
        acc_Euler = (current_thrust - m * g - drag_Euler_val) / m;
        acc_Heun  = (current_thrust - m * g - drag_Heun_val) / m;
        acc_RK4   = (current_thrust - m * g - drag_RK4_val) / m;

        % Enregistrement des données
        time = [time; t];
        speed_Euler = [speed_Euler; s_Euler];
        speed_Heun = [speed_Heun; s_Heun];
        speed_RK4 = [speed_RK4; s_RK4];
        altitude_Euler = [altitude_Euler; alt_Euler];
        altitude_Heun = [altitude_Heun; alt_Heun];
        altitude_RK4 = [altitude_RK4; alt_RK4];
        gravity = [gravity; g];
        mass = [mass; m];
```

```
    thrust = [thrust; current_thrust];
    drag_Euler = [drag_Euler; drag_Euler_val];
    drag_Heun = [drag_Heun; drag_Heun_val];
    drag_RK4 = [drag_RK4; drag_RK4_val];
    acceleration_Euler = [acceleration_Euler; acc_Euler];
    acceleration_Heun = [acceleration_Heun; acc_Heun];
    acceleration_RK4 = [acceleration_RK4; acc_RK4];

    % Intégration par méthode d'Euler
    s_Euler = s_Euler + dt * acc_Euler;
    alt_Euler = alt_Euler + dt * s_Euler;

    % Intégration par méthode de Heun
    s_pred = s_Heun + dt * acc_Heun;
    acc_pred = (current_thrust - m * g - drag_Heun_val) / m;
    s_Heun = s_Heun + (dt/2) * (acc_Heun + acc_pred);
    alt_Heun = alt_Heun + (dt/2) * (s_Heun + s_pred);

    % Mise à jour de l'accélération pour Heun après correction
    acc_Heun = (current_thrust - m * g - 0.5 * Cd * rho * A * s_Heun^2 * sign(s_Heun)) / m;
    acceleration_Heun(end) = acc_Heun;

    % Intégration par méthode RK4 (double intégration pour v et h)
    k1_v = dt * acc_RK4;
    k2_v = dt * ((current_thrust - m * g - drag_RK4_val) / m);
    k3_v = dt * ((current_thrust - m * g - drag_RK4_val) / m);
    k4_v = dt * ((current_thrust - m * g - drag_RK4_val) / m);
    s_RK4 = s_RK4 + (1/6) * (k1_v + 2*k2_v + 2*k3_v + k4_v);

    k1_h = dt * s_RK4;
    k2_h = dt * (s_RK4 + k1_h/2);
    k3_h = dt * (s_RK4 + k2_h/2);
    k4_h = dt * (s_RK4 + k3_h);
    alt_RK4 = alt_RK4 + (1/6) * (k1_h + 2*k2_h + 2*k3_h + k4_h);

    % Mise à jour de l'accélération pour RK4 après correction
    acc_RK4 = (current_thrust - m * g - 0.5 * Cd * rho * A * s_RK4^2 * sign(s_RK4)) / m;
    acceleration_RK4(end) = acc_RK4;

    % Mise à jour du temps
    t = t + dt;
  end

  % Sauvegarde des résultats avec la fonction `save_results`
  save_results("Advanced Model", dt, time, altitude_Euler, altitude_Heun, altitude_RK4, ...
          speed_Euler, speed_Heun, speed_RK4, ...
          acceleration_Euler, acceleration_Heun, acceleration_RK4, ...
          thrust, gravity, mass, drag_Euler);
end
```

S'appelle via la commande :

simulate_advanced_model(<le pas choisi>)

Modèle Complet

```
function simulate_full_model(dt)
    % Paramètres physiques
    g0 = 9.81;       % Gravité au niveau de la mer (m/s²)
    RT = 6371000;    % Rayon de la Terre (m)
    F = 5000;        % Poussée (N)
    m_empty = 100;   % Masse à vide de la fusée (kg)
    m_fuel = 50;     % Masse initiale du carburant (kg)
    dm = 1;          % Débit de consommation du carburant (kg/s)
    A = 1;           % Surface frontale (m²)
    Cd = 0.5;        % Coefficient de traînée
    rho0 = 1.225;    % Densité de l'air au niveau de la mer (kg/m³)
    H = 8000;        % Échelle de hauteur atmosphérique (m)

    % Conditions initiales
    t = 0;
    s_Euler = 0; s_Heun = 0; s_RK4 = 0;
    alt_Euler = 0; alt_Heun = 0; alt_RK4 = 0;
    m = m_empty + m_fuel; % Masse initiale de la fusée

    % Stockage des résultats
    time = []; speed_Euler = []; speed_Heun = []; speed_RK4 = [];
    altitude_Euler = []; altitude_Heun = []; altitude_RK4 = [];
    acceleration_Euler = []; acceleration_Heun = []; acceleration_RK4 = [];
    thrust = []; gravity = []; mass = []; drag_Euler = []; drag_Heun = []; drag_RK4 = [];

    % Simulation jusqu'à retour à altitude 0
    while alt_Euler >= 0
        % Mise à jour de la masse (tant qu'il y a du carburant)
        if m > m_empty
            m = max(m_empty, m - dm * dt);
            current_thrust = F;
        else
            current_thrust = 0; % Plus de poussée une fois le carburant épuisé
        end

        % Calcul de la gravité variable en fonction de l'altitude
        g_Euler = g0 * (RT / (RT + alt_Euler))^2;
        g_Heun  = g0 * (RT / (RT + alt_Heun))^2;
        g_RK4   = g0 * (RT / (RT + alt_RK4))^2;

        % Calcul de la densité de l'air variable en fonction de l'altitude
        rho_Euler = rho0 * exp(-alt_Euler / H);
        rho_Heun  = rho0 * exp(-alt_Heun / H);
        rho_RK4   = rho0 * exp(-alt_RK4 / H);

        % Calcul des forces de traînée pour chaque méthode
        drag_Euler_val = 0.5 * Cd * rho_Euler * A * s_Euler^2 * sign(s_Euler);
        drag_Heun_val  = 0.5 * Cd * rho_Heun  * A * s_Heun^2  * sign(s_Heun);
        drag_RK4_val   = 0.5 * Cd * rho_RK4   * A * s_RK4^2   * sign(s_RK4);

        % Calcul des accélérations pour chaque méthode
        acc_Euler = (current_thrust - m * g_Euler - drag_Euler_val) / m;
        acc_Heun  = (current_thrust - m * g_Heun  - drag_Heun_val) / m;
```

```
    acc_RK4   = (current_thrust - m * g_RK4   - drag_RK4_val) / m;

    % Enregistrement des données
    time = [time; t];
    speed_Euler = [speed_Euler; s_Euler];
    speed_Heun = [speed_Heun; s_Heun];
    speed_RK4 = [speed_RK4; s_RK4];
    altitude_Euler = [altitude_Euler; alt_Euler];
    altitude_Heun = [altitude_Heun; alt_Heun];
    altitude_RK4 = [altitude_RK4; alt_RK4];
    gravity = [gravity; g_Euler]; % On stocke la valeur d'Euler pour référence
    mass = [mass; m];
    thrust = [thrust; current_thrust];
    drag_Euler = [drag_Euler; drag_Euler_val];
    drag_Heun = [drag_Heun; drag_Heun_val];
    drag_RK4 = [drag_RK4; drag_RK4_val];
    acceleration_Euler = [acceleration_Euler; acc_Euler];
    acceleration_Heun = [acceleration_Heun; acc_Heun];
    acceleration_RK4 = [acceleration_RK4; acc_RK4];

    % Intégration par méthode d'Euler
    s_Euler = s_Euler + dt * acc_Euler;
    alt_Euler = alt_Euler + dt * s_Euler;

    % Intégration par méthode de Heun
    s_pred = s_Heun + dt * acc_Heun;
    acc_pred = (current_thrust - m * g_Heun - drag_Heun_val) / m;
    s_Heun = s_Heun + (dt/2) * (acc_Heun + acc_pred);
    alt_Heun = alt_Heun + (dt/2) * (s_Heun + s_pred);

    % Mise à jour de l'accélération pour Heun après correction
    acc_Heun = (current_thrust - m * g_Heun - 0.5 * Cd * rho_Heun * A * s_Heun^2 * sign(s_Heun)) / m;
    acceleration_Heun(end) = acc_Heun;

    % Intégration par méthode RK4 (double intégration pour v et h)
    k1_v = dt * acc_RK4;
    k2_v = dt * ((current_thrust - m * g_RK4 - drag_RK4_val) / m);
    k3_v = dt * ((current_thrust - m * g_RK4 - drag_RK4_val) / m);
    k4_v = dt * ((current_thrust - m * g_RK4 - drag_RK4_val) / m);
    s_RK4 = s_RK4 + (1/6) * (k1_v + 2*k2_v + 2*k3_v + k4_v);

    k1_h = dt * s_RK4;
    k2_h = dt * (s_RK4 + k1_h/2);
    k3_h = dt * (s_RK4 + k2_h/2);
    k4_h = dt * (s_RK4 + k3_h);
    alt_RK4 = alt_RK4 + (1/6) * (k1_h + 2*k2_h + 2*k3_h + k4_h);

    % Mise à jour de l'accélération pour RK4 après correction
    acc_RK4 = (current_thrust - m * g_RK4 - 0.5 * Cd * rho_RK4 * A * s_RK4^2 * sign(s_RK4)) / m;
    acceleration_RK4(end) = acc_RK4;

    % Mise à jour du temps
    t = t + dt;
end

% Sauvegarde des résultats avec la fonction `save_results`
```

```
    save_results("Full Model", dt, time, altitude_Euler, altitude_Heun, altitude_RK4, ...
                 speed_Euler, speed_Heun, speed_RK4, ...
                 acceleration_Euler, acceleration_Heun, acceleration_RK4, ...
                 thrust, gravity, mass, drag_Euler);
end
```

S’appelle via la commande :

simulate_full_model(<le pas choisi>)

Calcul d’erreur

```
function compute_errors(model_name, dt)
    % Chargement des données depuis le fichier Excel
    dt_folder = sprintf("%s/Pas-%.3f", model_name, dt);
    filename = sprintf("%s/%s-pas%.3f.xlsx", dt_folder, model_name, dt);

    if ~isfile(filename)
        error("Fichier non trouvé : %s", filename);
    end

    data = readtable(filename);
    time = data.time;

    % Extraction des variables
    altitude_Euler = data.altitude_Euler;
    altitude_Heun = data.altitude_Heun;
    altitude_RK4 = data.altitude_RK4;

    speed_Euler = data.speed_Euler;
    speed_Heun = data.speed_Heun;
    speed_RK4 = data.speed_RK4;

    acceleration_Euler = data.acceleration_Euler;
    acceleration_Heun = data.acceleration_Heun;
    acceleration_RK4 = data.acceleration_RK4;

    % Calcul des erreurs relatives
    error_alt_Euler_Heun = abs(altitude_Euler - altitude_Heun) ./ abs(altitude_Heun) * 100;
    error_alt_Euler_RK4 = abs(altitude_Euler - altitude_RK4) ./ abs(altitude_RK4) * 100;
    error_alt_Heun_RK4 = abs(altitude_Heun - altitude_RK4) ./ abs(altitude_RK4) * 100;

    error_speed_Euler_Heun = abs(speed_Euler - speed_Heun) ./ abs(speed_Heun) * 100;
    error_speed_Euler_RK4 = abs(speed_Euler - speed_RK4) ./ abs(speed_RK4) * 100;
    error_speed_Heun_RK4 = abs(speed_Heun - speed_RK4) ./ abs(speed_RK4) * 100;

    error_acc_Euler_Heun = abs(acceleration_Euler - acceleration_Heun) ./ abs(acceleration_Heun) * 100;
    error_acc_Euler_RK4 = abs(acceleration_Euler - acceleration_RK4) ./ abs(acceleration_RK4) * 100;
    error_acc_Heun_RK4 = abs(acceleration_Heun - acceleration_RK4) ./ abs(acceleration_RK4) * 100;

    % Appel de la fonction pour sauvegarder les résultats
    save_error_results(model_name, dt, time, ...
                       error_alt_Euler_Heun, error_alt_Euler_RK4, error_alt_Heun_RK4, ...
                       error_speed_Euler_Heun, error_speed_Euler_RK4, error_speed_Heun_RK4, ...
                       error_acc_Euler_Heun, error_acc_Euler_RK4, error_acc_Heun_RK4);
end
```

S’appelle via la commande :

compute_errors(<nom du modèle>, <le pas choisi>)

Sauvegarde et Export des résultats de simulation

```
function save_results(model_name, dt, time, altitude_Euler, altitude_Heun, altitude_RK4, ...
                      speed_Euler, speed_Heun, speed_RK4, ...
                      acceleration_Euler, acceleration_Heun, acceleration_RK4, ...
                      thrust, gravity, mass, drag_force)

    % Création du répertoire principal si inexistant
    if ~exist(model_name, 'dir')
        mkdir(model_name);
    end

    % Création d'un sous-répertoire pour chaque valeur de dt
    dt_folder = sprintf("%s/Pas-%.3f", model_name, dt);
    if ~exist(dt_folder, 'dir')
        mkdir(dt_folder);
    end

    % Sauvegarde des résultats dans un fichier Excel
    filename = sprintf("%s/%s-pas%.3f.xlsx", dt_folder, model_name, dt);
    T = table(time, altitude_Euler, altitude_Heun, altitude_RK4, ...
              speed_Euler, speed_Heun, speed_RK4, ...
              acceleration_Euler, acceleration_Heun, acceleration_RK4, ...
              thrust, gravity, mass, drag_force);
    writetable(T, filename);

    % Paramètres de style d'affichage
    styleEuler = 'ro-';  % Cercles rouges pour Euler
    styleHeun = 'b';     %'bs-';   % Carrés bleus pour Heun
    styleRK4 = 'g';      %'g^-';   % Triangles verts pour RK4

    % Fonction pour générer et sauvegarder chaque graphique individuellement avec unités
    function save_plot(x, y_Euler, y_Heun, y_RK4, title_text, x_label, y_label, filename, close_figure)
        figure('Position', [100, 100, 800, 600]); % Haute résolution
        plot(x, y_Euler, styleEuler, 'LineWidth', 2); hold on;
        plot(x, y_Heun, styleHeun, 'LineWidth', 2);
        plot(x, y_RK4, styleRK4, 'LineWidth', 2);
        title(title_text);
        xlabel(x_label);
        ylabel(y_label);
        grid on;
        legend('Euler', 'Heun', 'RK4');

        % Sauvegarde en SVG et PNG
        saveas(gcf, sprintf("%s/%s.svg", dt_folder, filename));
        saveas(gcf, sprintf("%s/%s.png", dt_folder, filename));

        % Fermeture de la figure uniquement si nécessaire
        if close_figure
            close(gcf);
        end
    end

    % Sauvegarde individuelle des graphiques avec unités
    save_plot(time, altitude_Euler, altitude_Heun, altitude_RK4, 'Altitude', 'Temps (s)', 'Altitude (m)',
```

```
'Altitude', true);
    save_plot(time, speed_Euler, speed_Heun, speed_RK4, 'Vitesse', 'Temps (s)', 'Vitesse (m/s)', 'Vitesse',
true);
    save_plot(time, acceleration_Euler, acceleration_Heun, acceleration_RK4, 'Accélération', 'Temps (s)',
'Accélération (m/s²)', 'Acceleration', true);
    save_plot(time, thrust, thrust, thrust, 'Poussée', 'Temps (s)', 'Poussée (N)', 'Poussee', true);
    save_plot(time, gravity, gravity, gravity, 'Gravité', 'Temps (s)', 'Gravité (m/s²)', 'Gravite', true);
    save_plot(time, mass, mass, mass, 'Masse', 'Temps (s)', 'Masse (kg)', 'Masse', true);
    save_plot(time, drag_force, drag_force, drag_force, 'Force de Traînée', 'Temps (s)', 'Force de Traînée
(N)', 'Trainee', true);

    % Création du graphique global regroupant toutes les courbes avec unités
    figure('Position', [100, 100, 1200, 900]);

    subplot(3,3,1); plot(time, altitude_Euler, styleEuler, 'LineWidth', 2); hold on;
    plot(time, altitude_Heun, styleHeun, 'LineWidth', 2);
    plot(time, altitude_RK4, styleRK4, 'LineWidth', 2);
    title('Altitude'); xlabel('Temps (s)'); ylabel('Altitude (m)'); grid on; legend('Euler', 'Heun', 'RK4');

    subplot(3,3,2); plot(time, thrust, 'k', 'LineWidth', 2);
    title('Poussée'); xlabel('Temps (s)'); ylabel('Poussée (N)'); grid on;

    subplot(3,3,3); plot(time, gravity, 'b', 'LineWidth', 2);
    title('Gravité'); xlabel('Temps (s)'); ylabel('Gravité (m/s²)'); grid on;

    subplot(3,3,4); plot(time, mass, 'g', 'LineWidth', 2);
    title('Masse'); xlabel('Temps (s)'); ylabel('Masse (kg)'); grid on;

    subplot(3,3,5); plot(time, speed_Euler, styleEuler, 'LineWidth', 2); hold on;
    plot(time, speed_Heun, styleHeun, 'LineWidth', 2);
    plot(time, speed_RK4, styleRK4, 'LineWidth', 2);
    title('Vitesse'); xlabel('Temps (s)'); ylabel('Vitesse (m/s)'); grid on; legend('Euler', 'Heun', 'RK4');

    subplot(3,3,6); plot(time, acceleration_Euler, styleEuler, 'LineWidth', 2); hold on;
    plot(time, acceleration_Heun, styleHeun, 'LineWidth', 2);
    plot(time, acceleration_RK4, styleRK4, 'LineWidth', 2);
    title('Accélération'); xlabel('Temps (s)'); ylabel('Accélération (m/s²)'); grid on; legend('Euler', 'Heun',
'RK4');

    subplot(3,3,7); plot(time, drag_force, 'y', 'LineWidth', 2);
    title('Force de Traînée'); xlabel('Temps (s)'); ylabel('Force de Traînée (N)'); grid on;

    % Sauvegarde du graphique global
    saveas(gcf, sprintf("%s/%s-TrajectoiresGlobales.svg", dt_folder, model_name));
    saveas(gcf, sprintf("%s/%s-TrajectoiresGlobales.png", dt_folder, model_name));

    % Fermeture de la figure globale
    close(gcf);
end
```

Sauvegarde et Export des calculs d'erreur

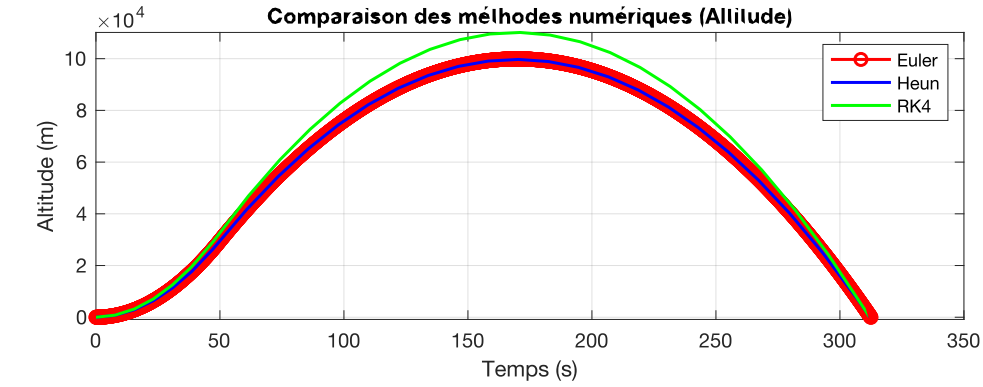
```
function save_error_results(model_name, dt, time, ...
                    error_alt_Euler_Heun, error_alt_Euler_RK4, error_alt_Heun_RK4, ...
                    error_speed_Euler_Heun, error_speed_Euler_RK4, error_speed_Heun_RK4, ...
                    error_acc_Euler_Heun, error_acc_Euler_RK4, error_acc_Heun_RK4)

    % Création du répertoire de stockage des erreurs
    dt_folder = sprintf("%s/Pas-%.3f", model_name, dt);
    if ~exist(dt_folder, 'dir')
        mkdir(dt_folder);
    end

    % Sauvegarde des erreurs dans un fichier Excel
    error_filename = sprintf("%s/%s-erreurs-pas%.3f.xlsx", dt_folder, model_name, dt);
    T_errors = table(time, ...
                error_alt_Euler_Heun, error_alt_Euler_RK4, error_alt_Heun_RK4, ...
                error_speed_Euler_Heun, error_speed_Euler_RK4, error_speed_Heun_RK4, ...
                error_acc_Euler_Heun, error_acc_Euler_RK4, error_acc_Heun_RK4);
    writetable(T_errors, error_filename);

    % Paramètres de style pour les graphiques
    styleEulerHeun = 'r';    %ro-';  % Cercles rouges pour Euler-Heun
    styleEulerRK4  = 'b';    % Ligne bleue pour Euler-RK4
    styleHeunRK4   = 'g';    % Ligne verte pour Heun-RK4

    % Création et sauvegarde des graphiques d'erreurs
    figure('Position', [100, 100, 1200, 900]);

    % Graphiques pour l'altitude
    subplot(3,2,1);
    semilogy(time, error_alt_Euler_Heun, styleEulerHeun, 'LineWidth', 2); hold on;
    semilogy(time, error_alt_Heun_RK4, styleHeunRK4, 'LineWidth', 2);
    title('Erreur Altitude (%) - Euler-Heun & Heun-RK4'); xlabel('Temps (s)'); ylabel('Erreur (%)'); grid on;
    legend('Euler-Heun', 'Heun-RK4');

    subplot(3,2,2);
    semilogy(time, error_alt_Euler_RK4, styleEulerRK4, 'LineWidth', 2); hold on;
    semilogy(time, error_alt_Heun_RK4, styleHeunRK4, 'LineWidth', 2);
    title('Erreur Altitude (%) - Euler-RK4 & Heun-RK4'); xlabel('Temps (s)'); ylabel('Erreur (%)'); grid on;
    legend('Euler-RK4', 'Heun-RK4');

    % Graphiques pour la vitesse
    subplot(3,2,3);
    semilogy(time, error_speed_Euler_Heun, styleEulerHeun, 'LineWidth', 2); hold on;
    semilogy(time, error_speed_Heun_RK4, styleHeunRK4, 'LineWidth', 2);
    title('Erreur Vitesse (%) - Euler-Heun & Heun-RK4'); xlabel('Temps (s)'); ylabel('Erreur (%)'); grid on;
    legend('Euler-Heun', 'Heun-RK4');

    subplot(3,2,4);
    semilogy(time, error_speed_Euler_RK4, styleEulerRK4, 'LineWidth', 2); hold on;
    semilogy(time, error_speed_Heun_RK4, styleHeunRK4, 'LineWidth', 2);
    title('Erreur Vitesse (%) - Euler-RK4 & Heun-RK4'); xlabel('Temps (s)'); ylabel('Erreur (%)'); grid on;
    legend('Euler-RK4', 'Heun-RK4');
```

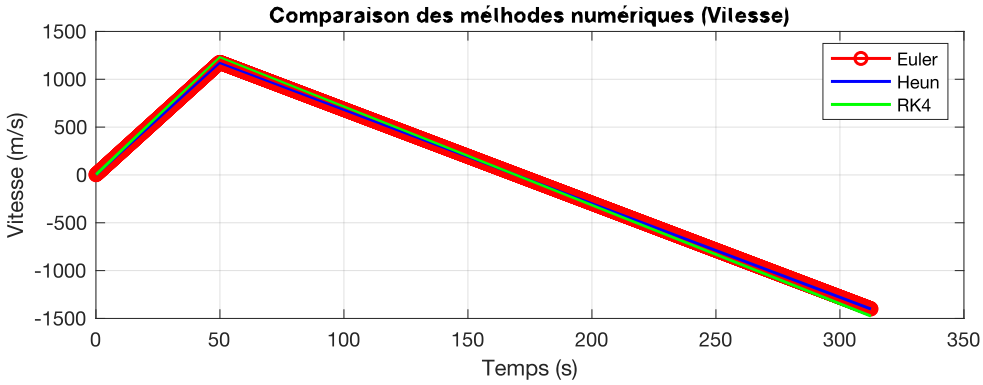
```
    % Graphiques pour l'accélération
    subplot(3,2,5);
    semilogy(time, error_acc_Euler_Heun, styleEulerHeun, 'LineWidth', 2); hold on;
    semilogy(time, error_acc_Heun_RK4, styleHeunRK4, 'LineWidth', 2);
    title('Erreur Accélération (%) - Euler-Heun & Heun-RK4'); xlabel('Temps (s)'); ylabel('Erreur (%)'); grid
on;
    legend('Euler-Heun', 'Heun-RK4');

    subplot(3,2,6);
    semilogy(time, error_acc_Euler_RK4, styleEulerRK4, 'LineWidth', 2); hold on;
    semilogy(time, error_acc_Heun_RK4, styleHeunRK4, 'LineWidth', 2);
    title('Erreur Accélération (%) - Euler-RK4 & Heun-RK4'); xlabel('Temps (s)'); ylabel('Erreur (%)'); grid on;
    legend('Euler-RK4', 'Heun-RK4');

    % Sauvegarde des graphiques
    saveas(gcf, sprintf("%s/%s-Erreurs-pas%.3f.svg", dt_folder, model_name, dt));
    saveas(gcf, sprintf("%s/%s-Erreurs-pas%.3f.png", dt_folder, model_name, dt));

    % Fermeture de la figure
    close(gcf);
end
```

Annexe 2 : Altitudes et Vitesses. Synthèse des erreurs.

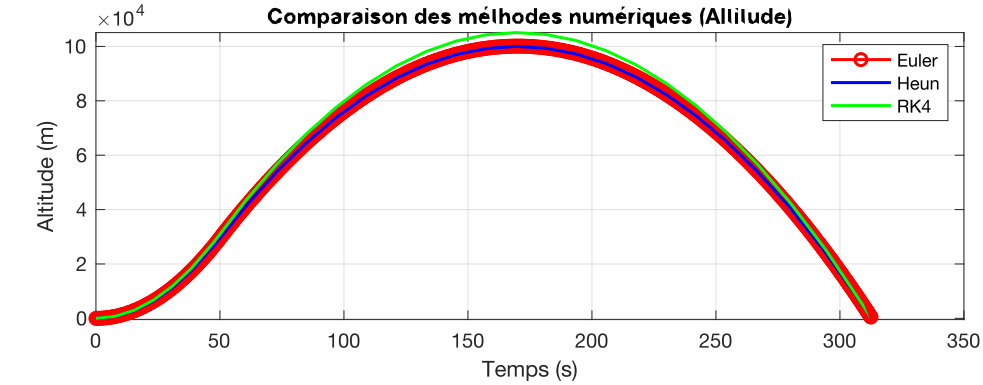
Altitudes et Vitesses

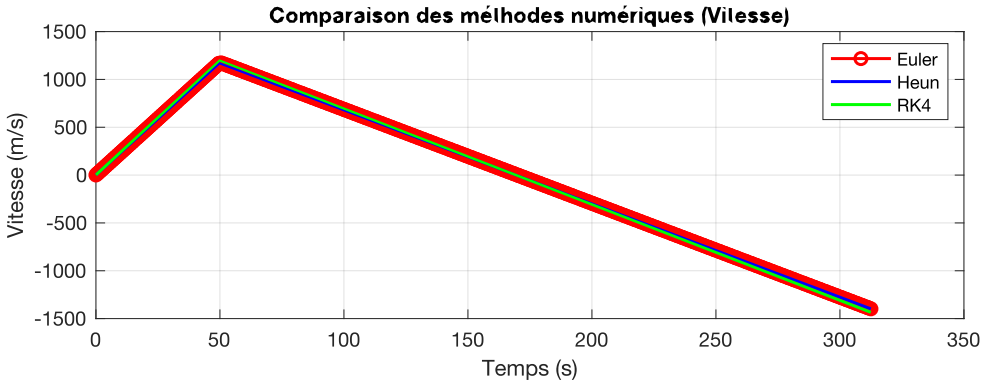
Altitude et vitesse selon les méthodes pour $dt = 0{,}1s$



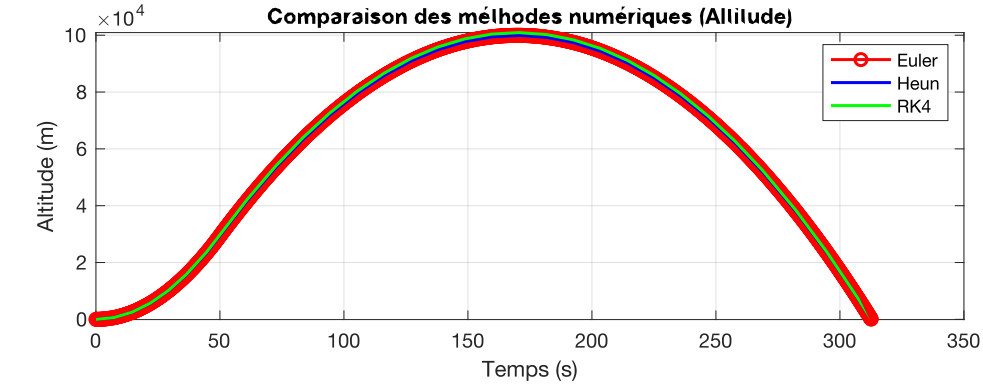


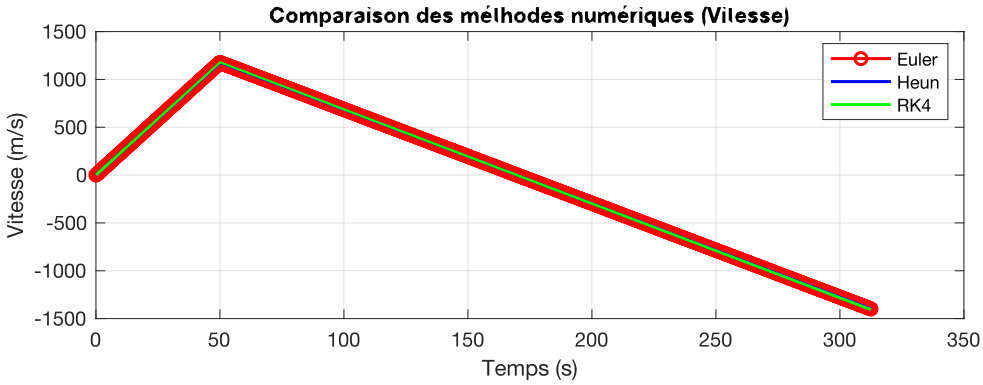
Altitude et vitesse selon les méthodes pour $dt = 0{,}05s$



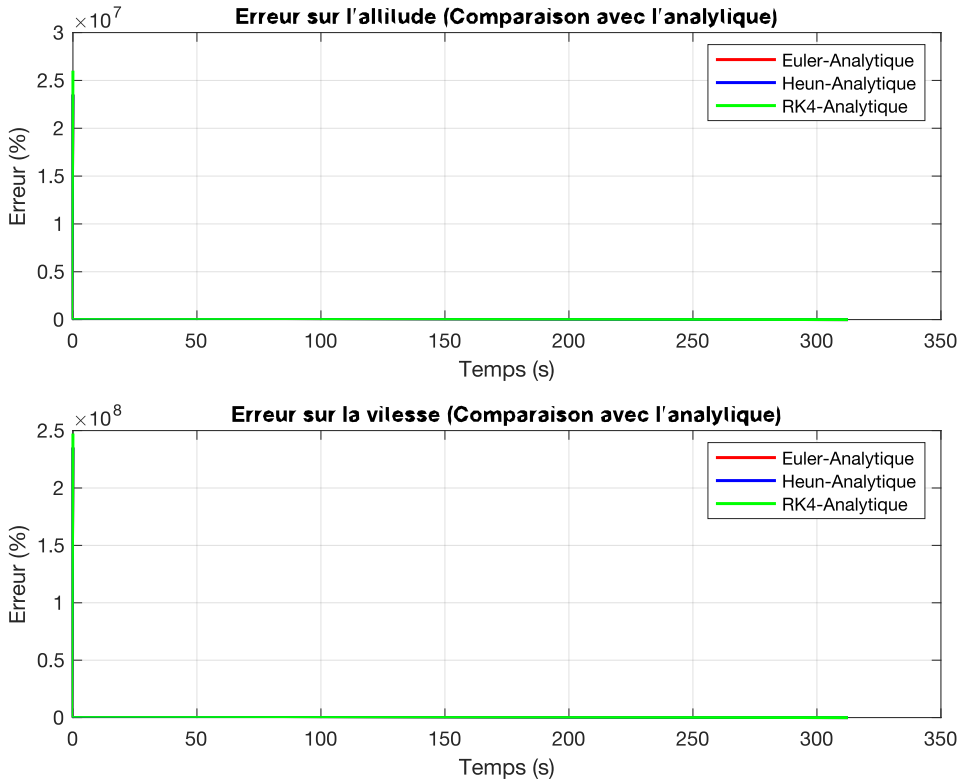


Altitude et vitesse selon les méthodes pour $dt = 0{,}01s$

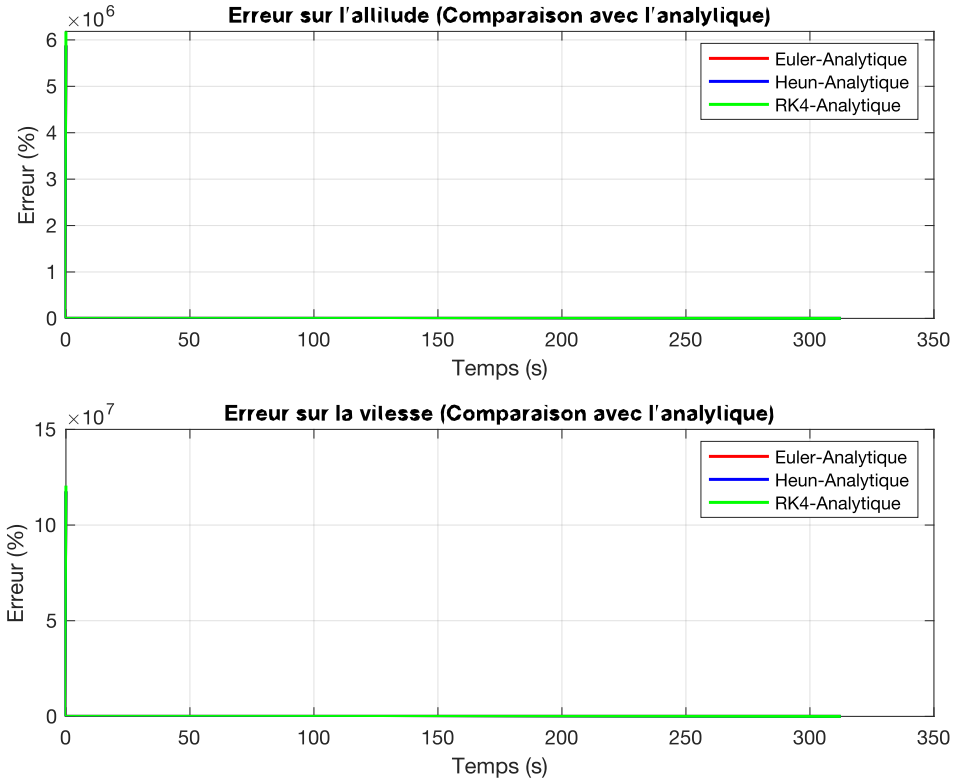




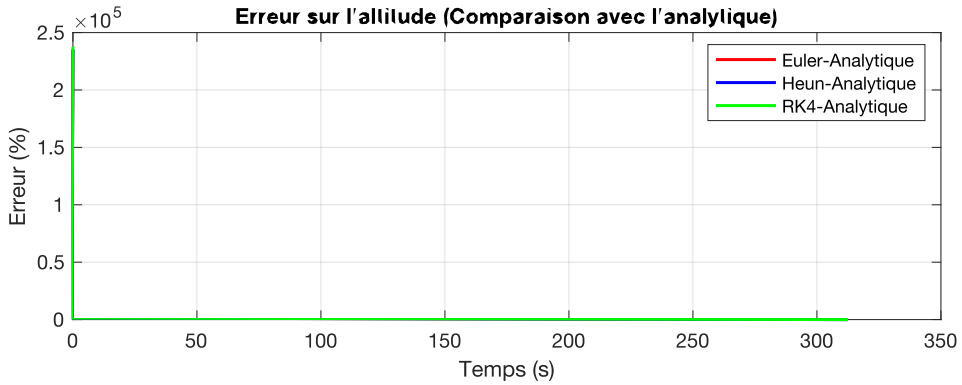
Erreurs selon les méthodes pour $dt = 0{,}1s$

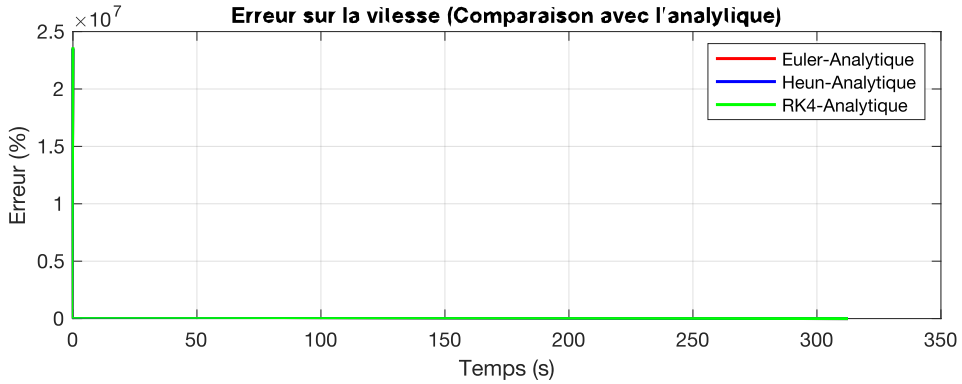


Erreurs selon les méthodes pour $dt = 0{,}05s$



Erreurs selon les méthodes pour $dt = 0{,}01s$





Synthèse des erreurs - $dt = 0{,}1s$

Erreurs sur l'altitude

Méthode	Tendance
Euler	Erreur initiale très élevée, diminue progressivement
Heun	Réduit légèrement l'erreur, proche d'Euler
RK4	Méthode la plus précise, erreur moindre

Erreurs sur la vitesse

Méthode	Tendance
Euler	Erreur importante au début, diminue avec le temps
Heun	Même tendance qu'Euler mais avec une erreur plus faible
RK4	Erreur plus basse et meilleure précision

Conclusion

Synthèse
Euler présente des erreurs importantes au début, mais elles diminuent progressivement.
Heun est meilleure, mais reste assez proche d'Euler.
RK4 est la méthode la plus précise et stable avec les erreurs les plus faibles.

Synthèse des erreurs - $dt = 0{,}05s$

Erreurs sur l'altitude

Méthode	Tendance
Euler	Erreur initiale très élevée, diminue plus rapidement que dt=0,1s
Heun	Améliore la précision avec une erreur plus faible qu'Euler
RK4	Méthode la plus précise, erreur fortement réduite

Erreurs sur la vitesse

Méthode	Tendance
Euler	Erreur importante, mais réduite par rapport à dt=0,1s
Heun	Même tendance qu'Euler mais avec une meilleure précision
RK4	Méthode la plus précise et stable

Conclusion

Synthèse
Euler montre des erreurs significatives, mais elles diminuent plus rapidement.
Heun améliore Euler, avec une meilleure précision.
RK4 est la méthode la plus précise et stable avec une erreur encore plus réduite.

Synthèse des erreurs - $dt = 0{,}01s$

Erreurs sur l'altitude

Méthode	Tendance
Euler	Erreur encore présente mais bien réduite
Heun	Tendance similaire à Euler mais plus précise

RK4	Méthode la plus stable et précise, erreur très faible

Erreurs sur la vitesse

Méthode	Tendance
Euler	Erreur au départ, mais fortement réduite
Heun	Proche de RK4, converge mieux
RK4	Erreur minimale et stable tout au long du vol

Conclusion

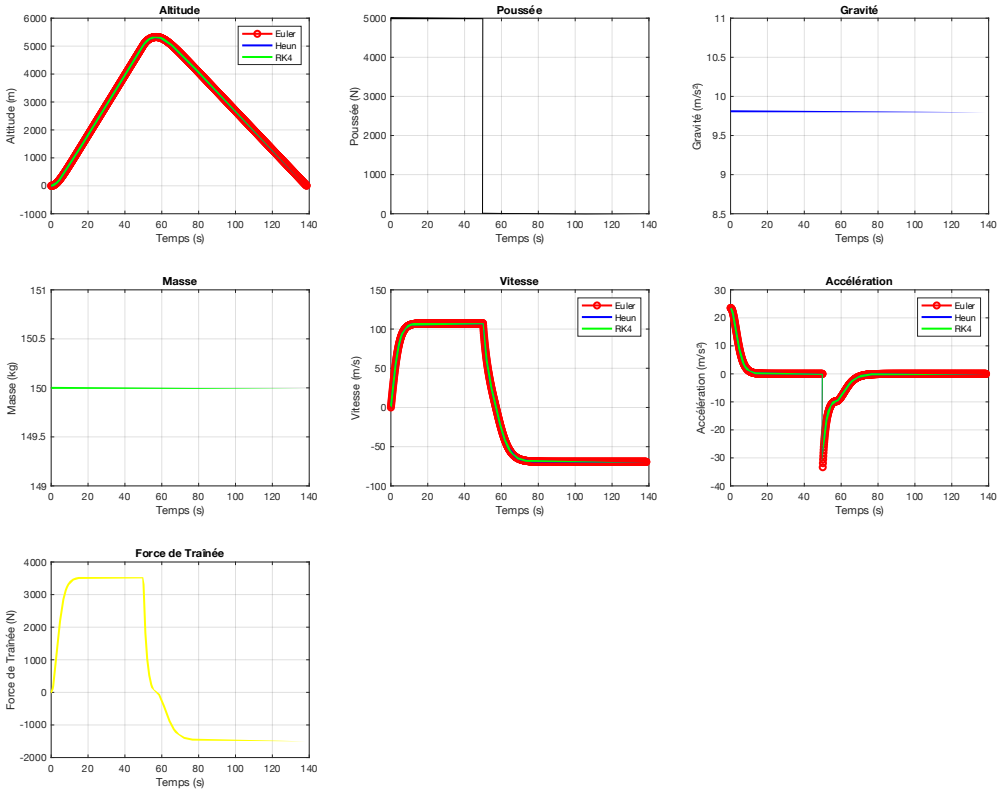
Synthèse
Avec un pas plus petit, les erreurs sont moindres.
Heun est une alternative à RK4, avec une bonne précision.
Euler est la méthode la moins précise, mais son erreur devient acceptable.

Source de données disponible ici

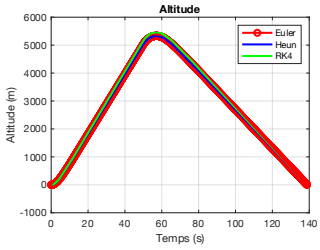
Annexe 3 – Courbes

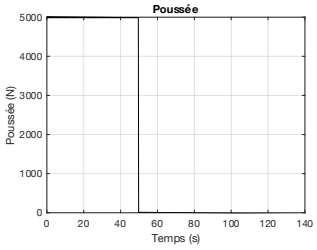
Modèle Simplifié

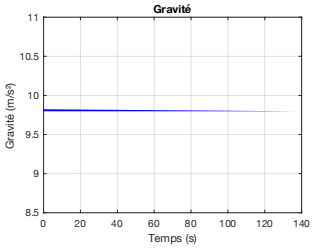
Courbes pour $dt = 0{,}1s$

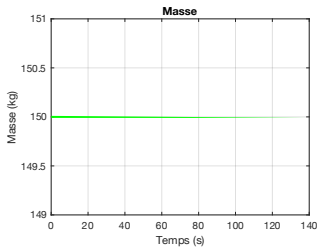


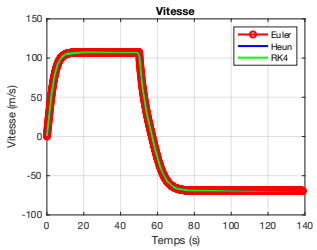
Courbes pour $dt = 0{,}05s$

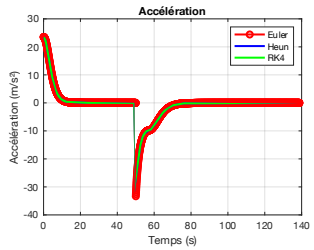


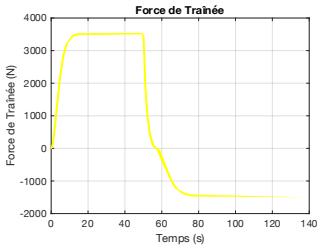




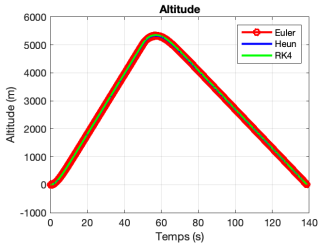


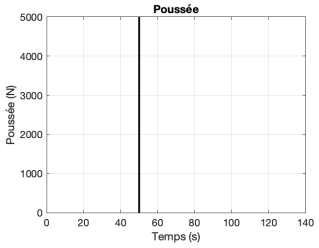


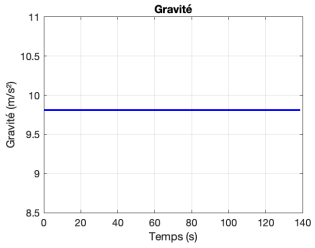


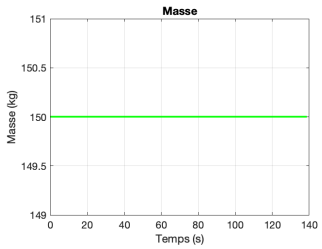


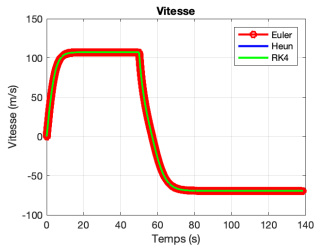
Courbes pour $dt = 0{,}01s$

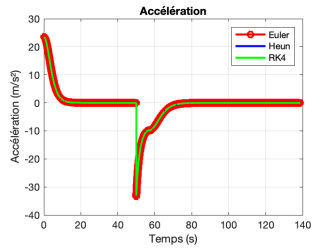


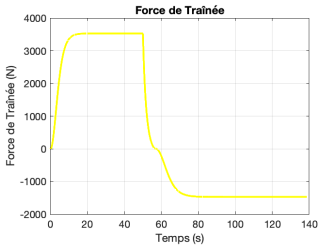






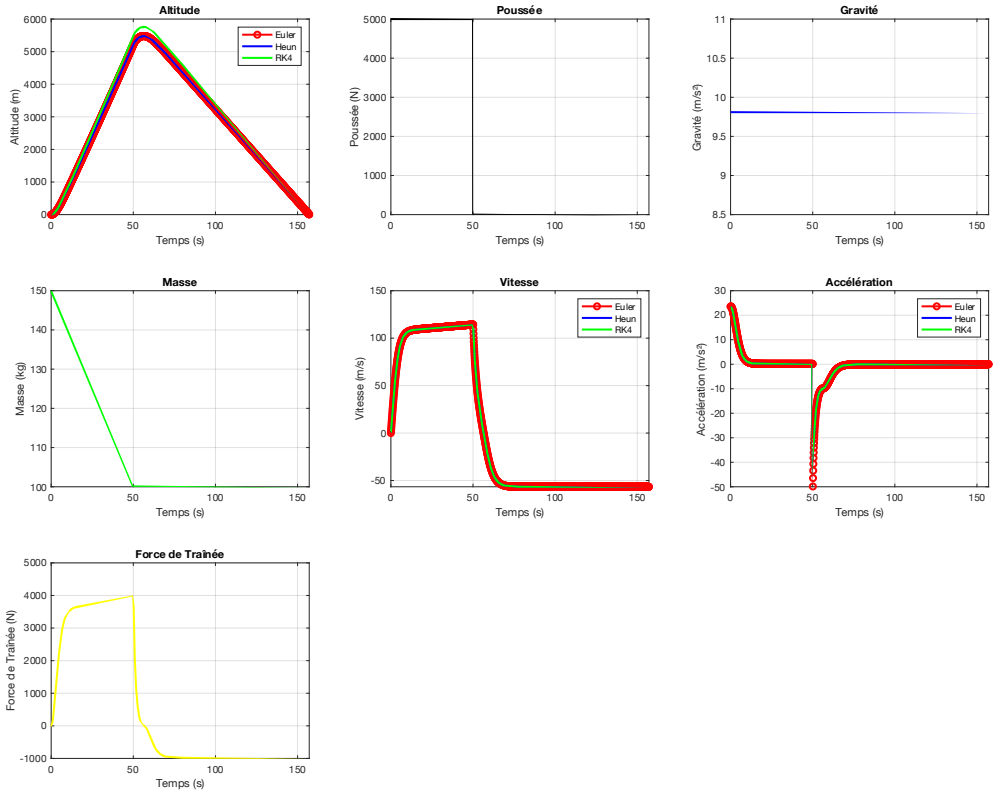




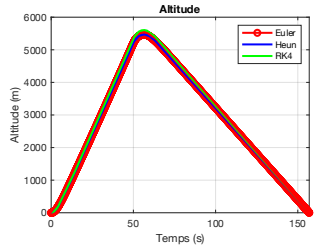


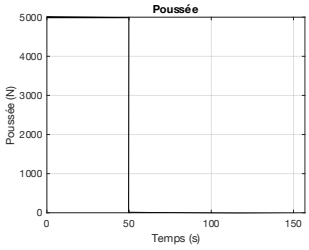
Modèle Avancé

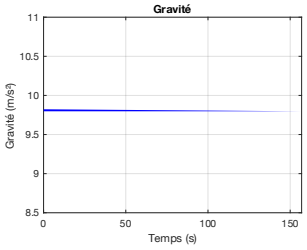
Courbes pour $dt = 0{,}1s$

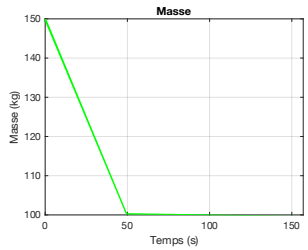


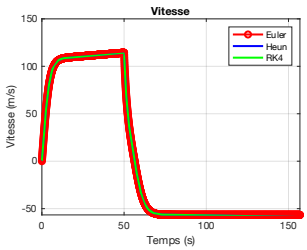
Courbes pour $dt = 0{,}05s$

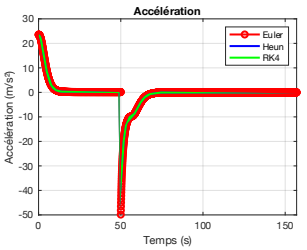


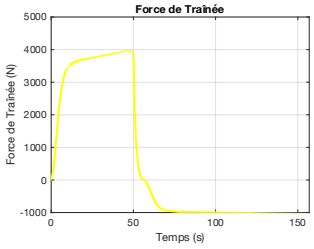




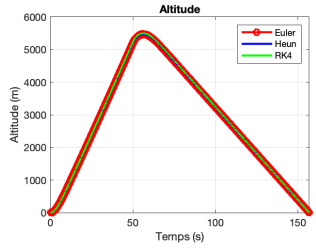


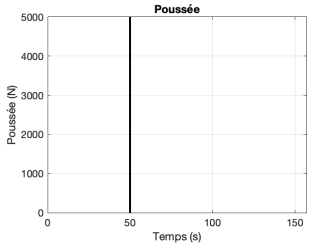


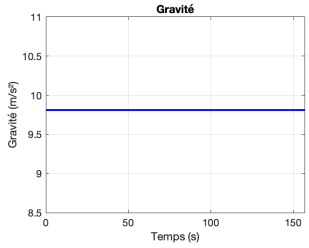


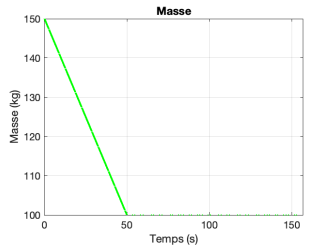


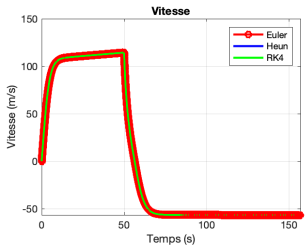
Courbes pour $dt = 0{,}01s$

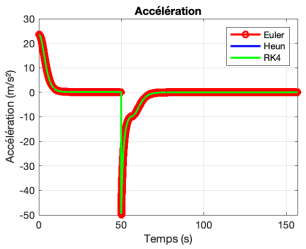


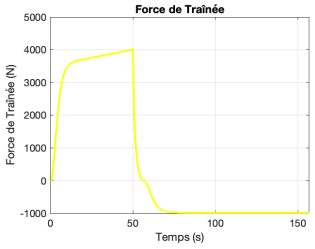






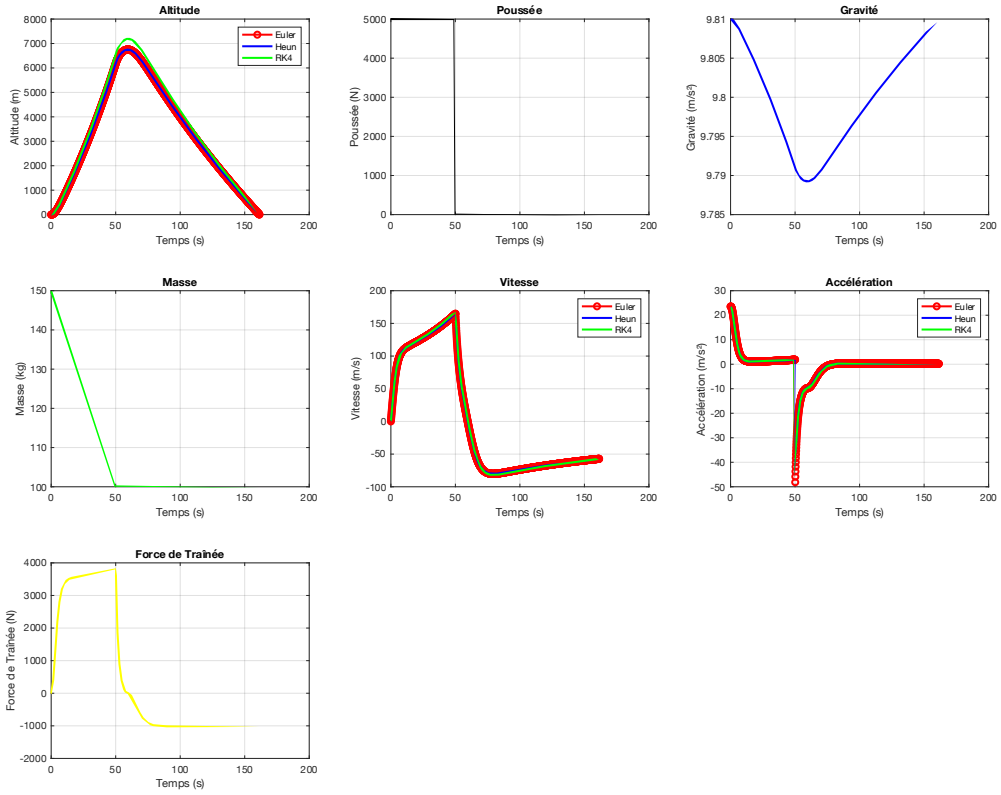




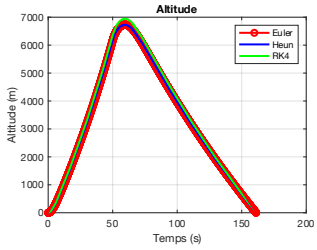


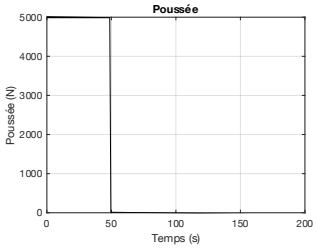
Modèle Complet

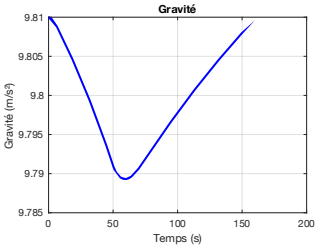
Courbes pour $dt = 0{,}1s$

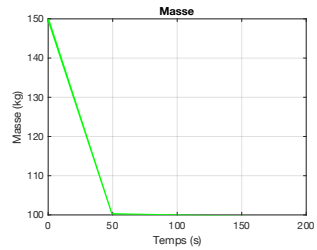


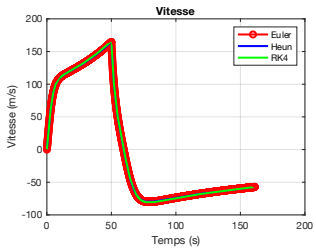
Courbes pour $dt = 0{,}05s$

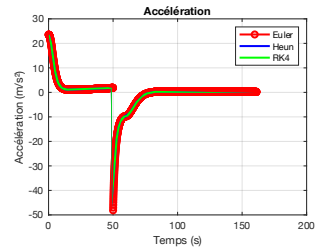


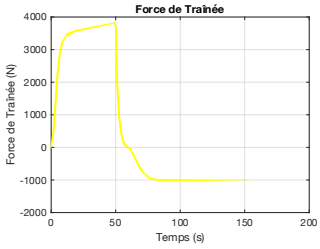




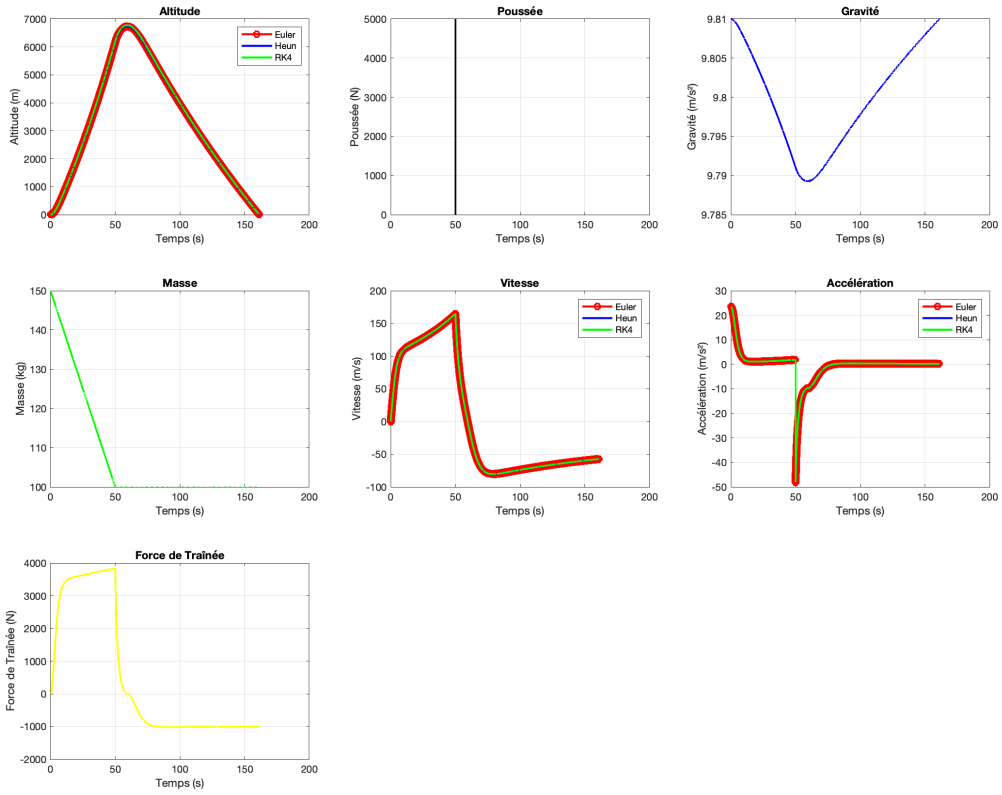








Courbes pour $dt = 0{,}01s$



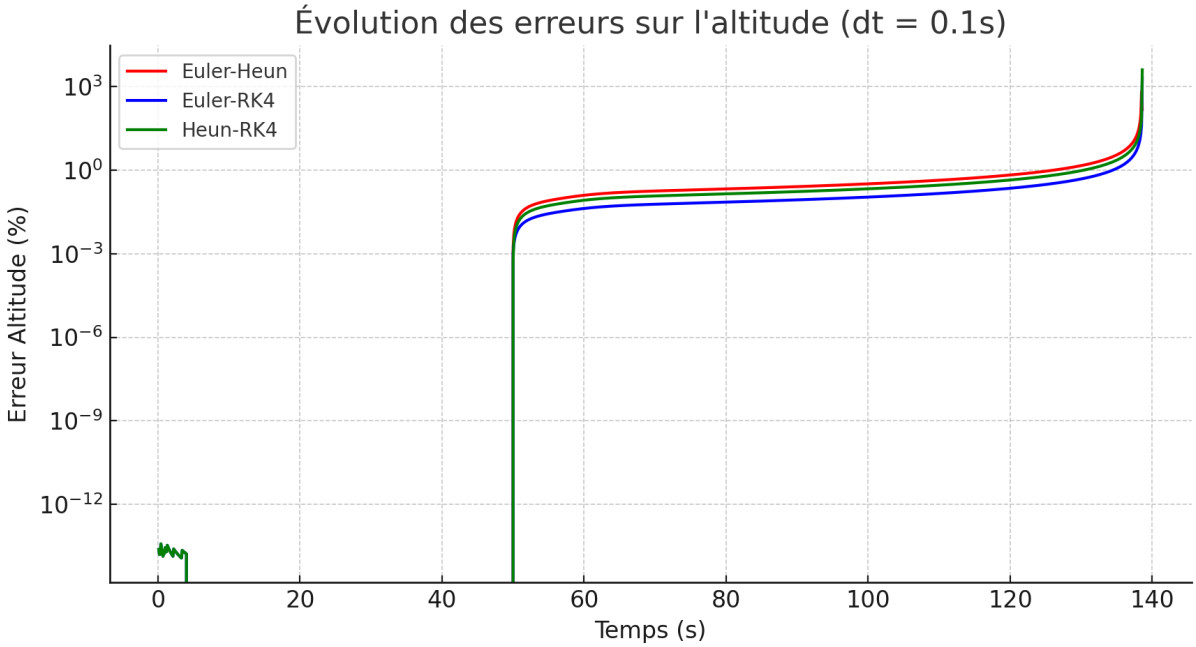
Conclusion

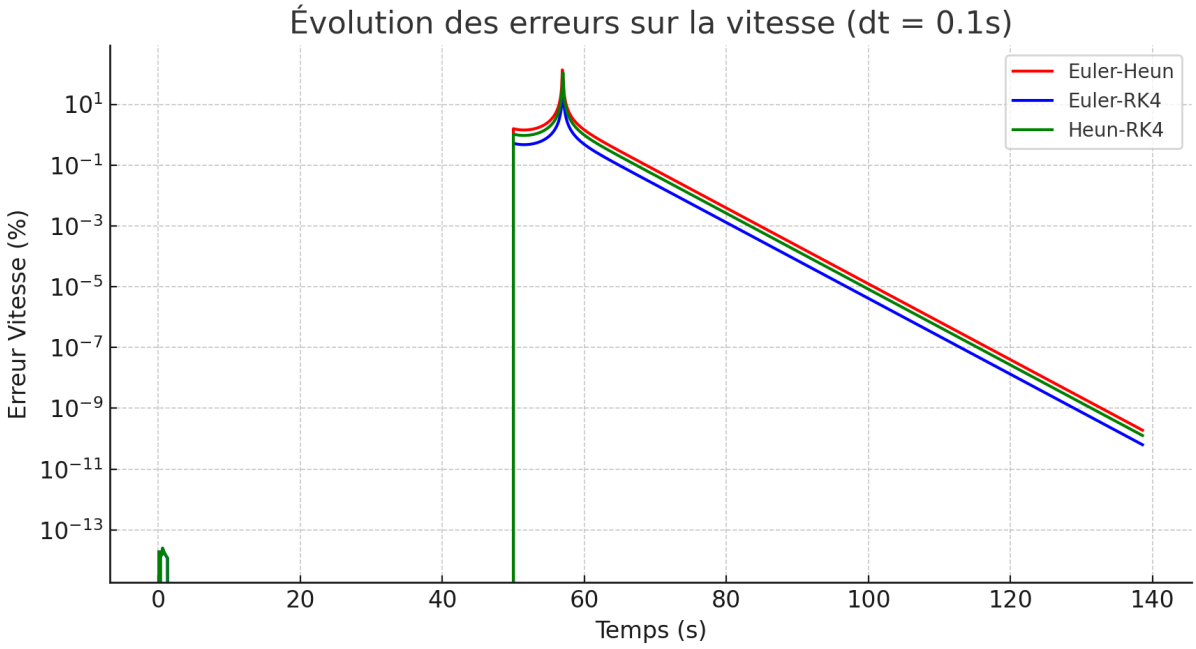
On se rend vite compte qu'il est difficile de voir les différences entre les courbes, et que seule une analyse d'erreur nous permettra de constater les différences entre les méthodes.

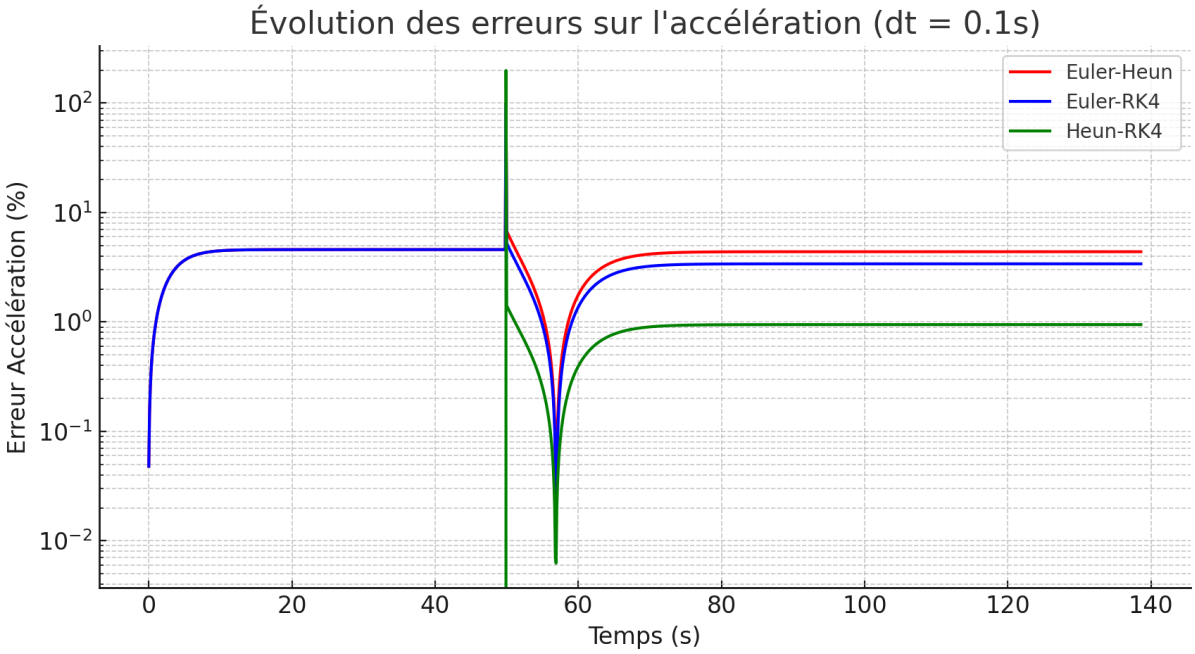
Annexe 4.1 – Synthèse – Modèle Simplifié

Courbes d'erreur

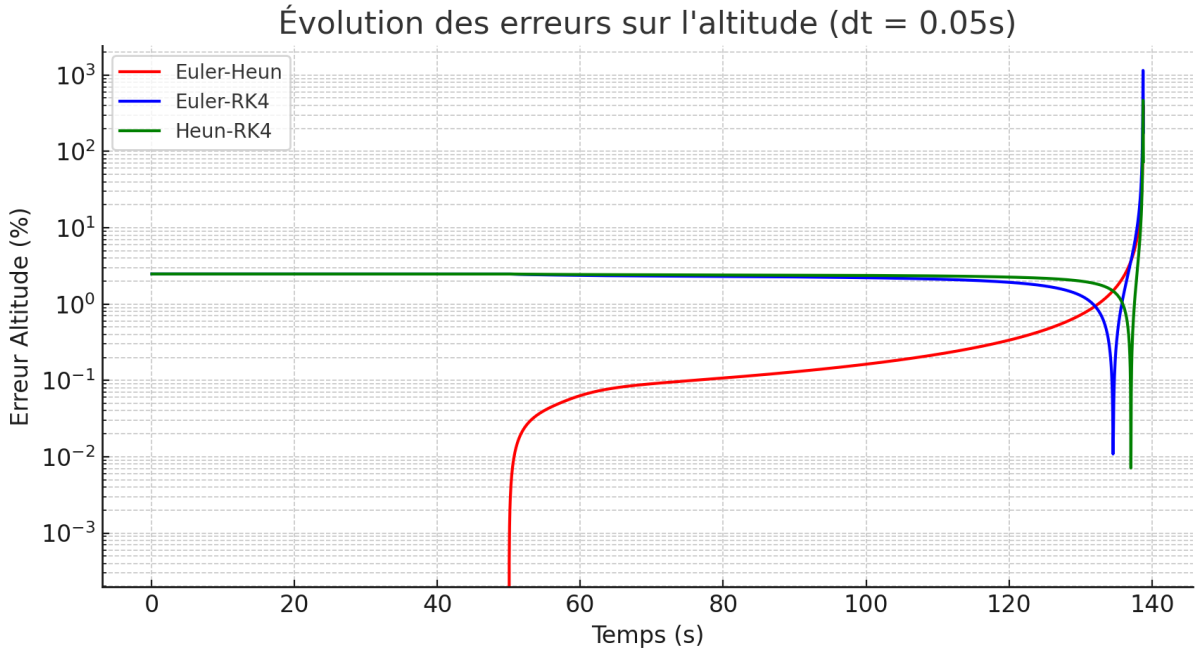
Pas $dt = 0{,}1s$

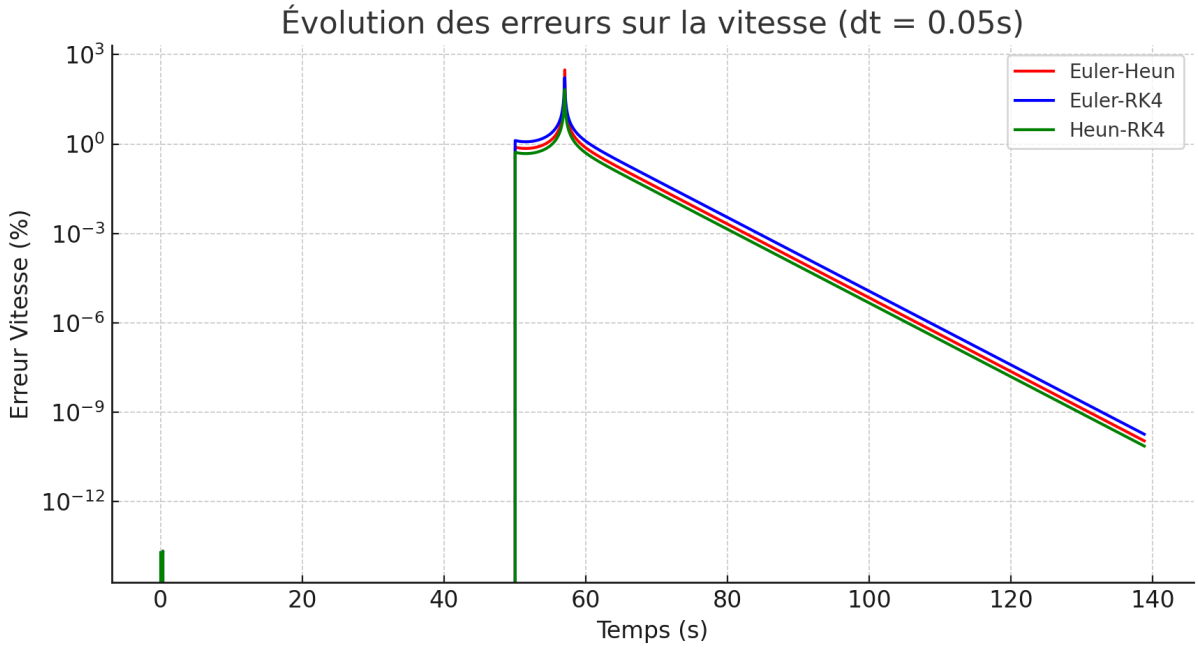


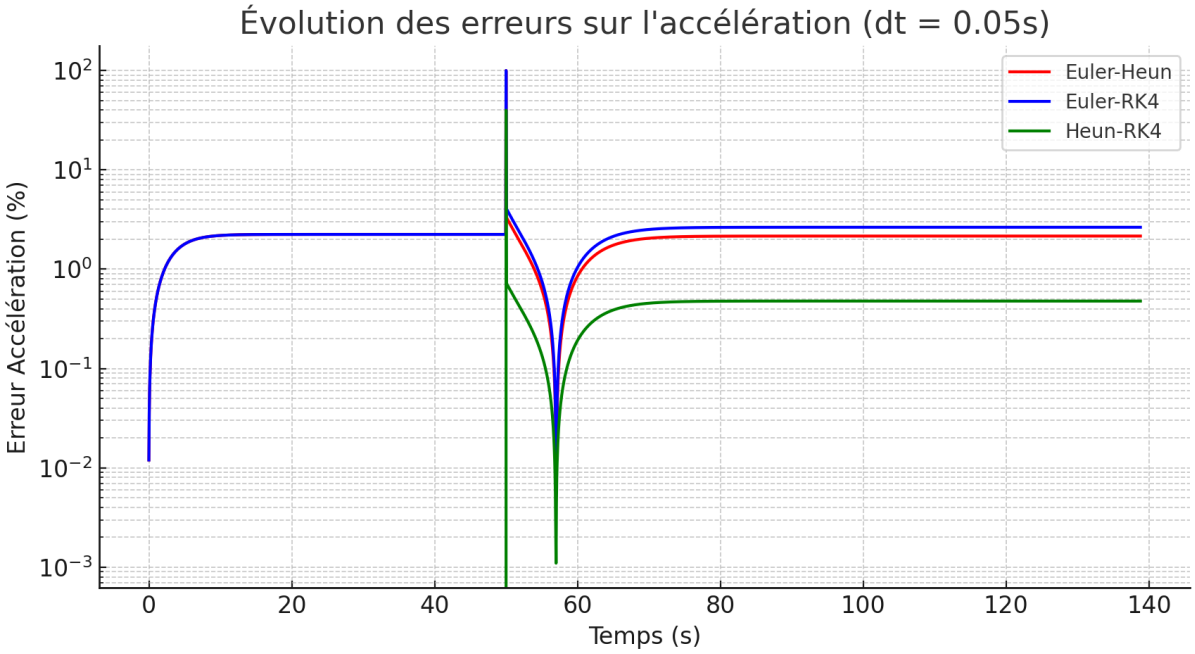



Évolution des erreurs sur l'accélération (dt = 0.1s)
Euler-Heun
Euler-RK4
Heun-RK4
Erreur Accélération (%)
10^{2}
10^{1}
10^{0}
10^{-1}
10^{-2}
0
20
40
60
80
100
120
140
Temps (s)

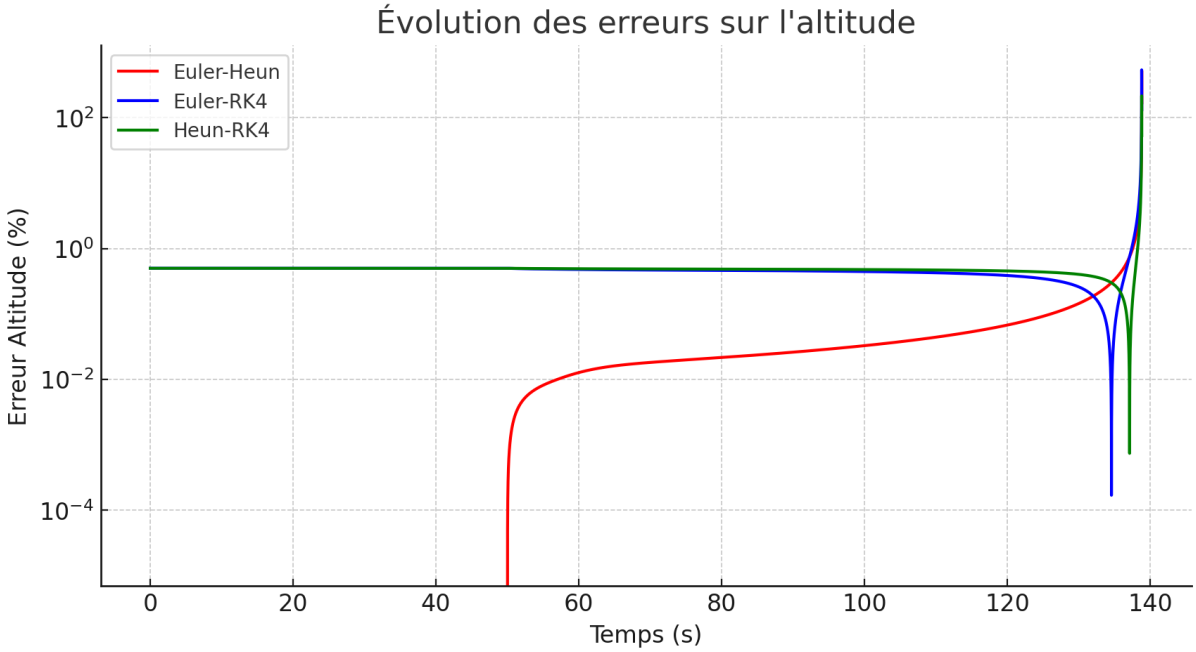
Pas $dt = 0{,}05s$

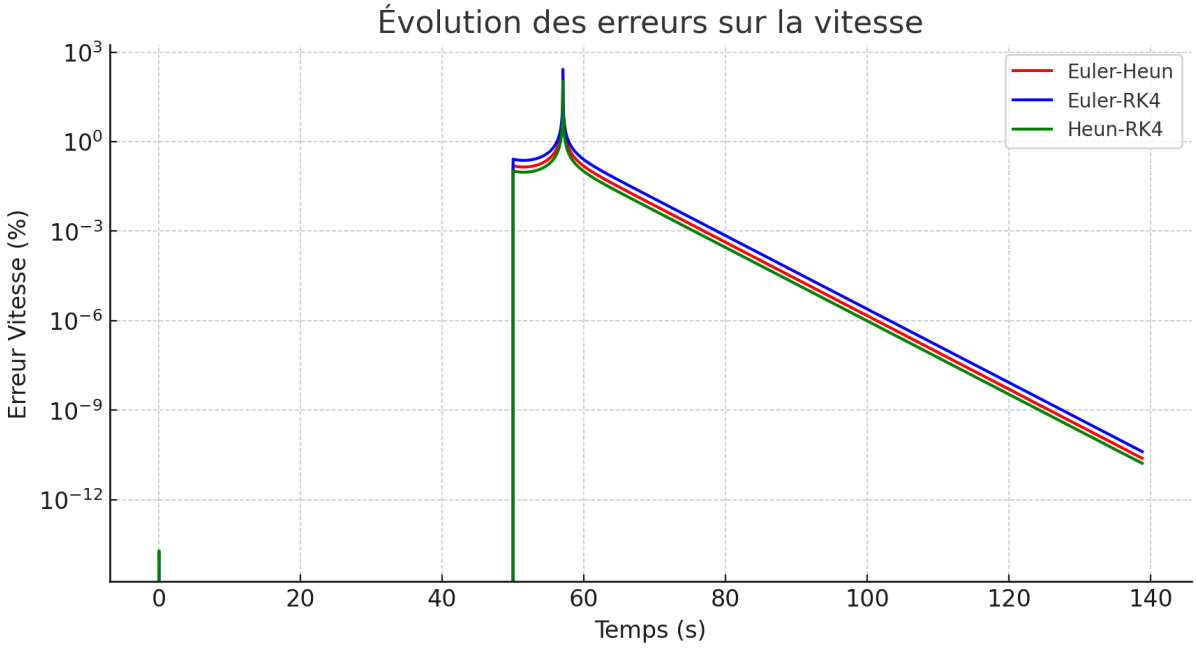

Évolution des erreurs sur l'altitude (dt = 0.05s)
Euler-Heun
Euler-RK4
Heun-RK4
Erreur Altitude (%)
10^{3}
10^{2}
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
0
20
40
60
80
100
120
140
Temps (s)

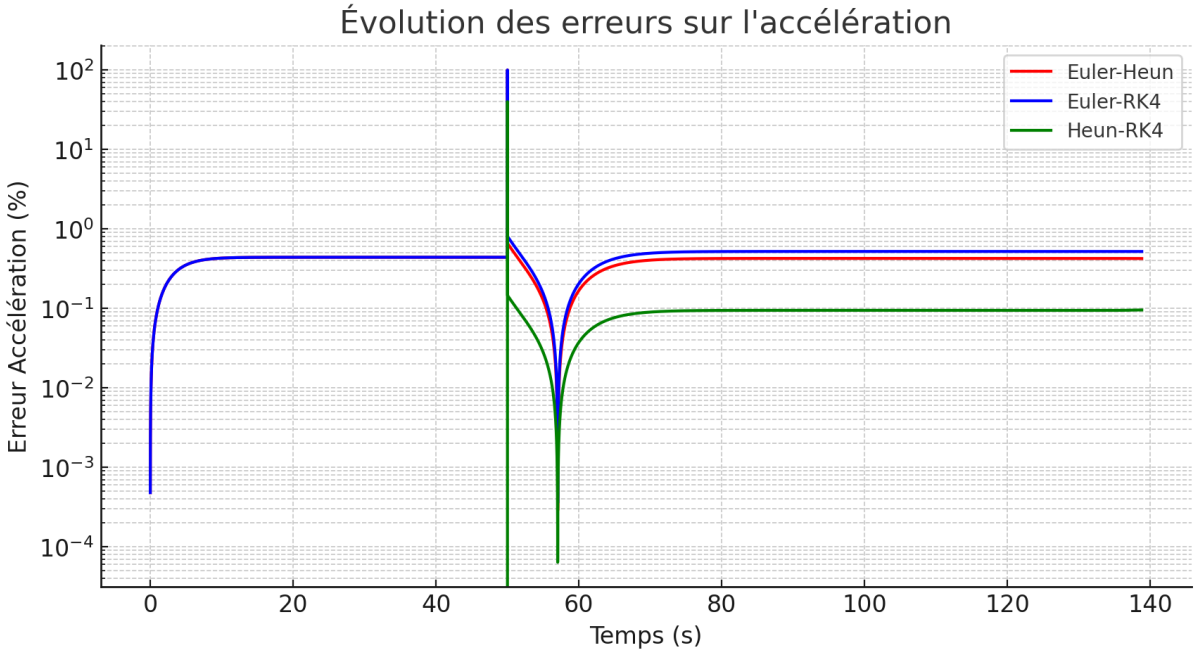

Évolution des erreurs sur la vitesse (dt = 0.05s)
Euler-Heun
Euler-RK4
Heun-RK4
Erreur Vitesse (%)
10^{3}
10^{0}
10^{-3}
10^{-6}
10^{-9}
10^{-12}
0
20
40
60
80
100
120
140
Temps (s)


Évolution des erreurs sur l'accélération (dt = 0.05s)
Euler-Heun
Euler-RK4
Heun-RK4
Erreur Accélération (%)
10^{2}
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
0
20
40
60
80
100
120
140
Temps (s)

Pas $dt = 0{,}01s$


Évolution des erreurs sur l'altitude
Euler-Heun
Euler-RK4
Heun-RK4
Erreur Altitude (%)
10^{2}
10^{0}
10^{-2}
10^{-4}
0
20
40
60
80
100
120
140
Temps (s)


Évolution des erreurs sur la vitesse
Euler-Heun
Euler-RK4
Heun-RK4
Erreur Vitesse (%)
10^{3}
10^{0}
10^{-3}
10^{-6}
10^{-9}
10^{-12}
0
20
40
60
80
100
120
140
Temps (s)



Synthèse

Analyse des Erreurs - Modèle simplifié (dt = 0,1s)

Analyse des erreurs sur l'altitude

Méthode	**Tendance**	**Erreur Moyenne (%)**	**Transition Poussée/Chute Libre**	**Observations**
Euler	Imprécis	Élevée	Critique	Erreur importante, surtout en fin de vol.
Heun	Proche de RK4	≈ 4%	Critique	Se rapproche de RK4, mais avec une erreur certaine.
RK4	Référence	Faible	Critique	Plus stable et précis, mais des erreurs restent en fin de vol.

Analyse des erreurs sur la vitesse

Méthode	Erreur Maximale (%)	Tendance	Transition Poussée/Chute Libre	Observations
Euler	> 10%	Forte erreur	Cause principale des erreurs	Très imprécis lors du changement de régime.
Heun	Modérée	Diminue après transition	Cause principale des erreurs	Meilleur que Euler, mais est impacté par la transition.
RK4	Faible	Stable	Cause principale des erreurs	Méthode la plus stable après la transition.

Analyse des erreurs sur l'accélération

Méthode	Erreur Maximale (%)	Tendance	Transition Poussée/Chute Libre	Observations
Euler	> 100%	Très instable	Super critique	Fortes variations et imprécisions.
Heun	Instable	Moins instable que Euler	Critique	Meilleur que Euler, mais est instable.
RK4	Modérée	Plus stable	Critique	La plus précise, mais toujours impactée par la transition.

Comparaison des erreurs avec dt=0,01s

Méthode	Effet de dt=0,1s	Effet de dt=0,01s	Observations
Euler	Très imprécis	Amélioration, mais est imprécis	Euler devient nettement moins précis avec un grand pas.
Heun	Écart certain avec RK4	Plus précis et proche de RK4	Heun est un bon compromis, mais

			montre des écarts.
RK4	Relativement stable	La plus stable	RK4 est la plus stable et précise.

Conclusion

Point clé
Euler est inadapté pour dt=0,1s.
Heun est un bon compromis, mais montre des erreurs certaines.
RK4 est la plus stable et précise, mais pas parfaite sur les transitions.
Les pics d'erreur autour de t ≈ 60s suggèrent une instabilité à ces moments critiques.

Analyse des Erreurs - Modèle simplifié (dt = 0,05s)

Analyse des erreurs sur l'altitude

Méthode	**Tendance**	**Erreur Moyenne (%)**	**Transition Poussée/Chute Libre**	**Observations**
Euler	Faible erreur en montée, pic en fin	Élevée en fin de vol	Critique, augmentation des erreurs	Erreur faible en montée mais forte en fin de vol.
Heun	Erreur stable < 1%	≈ 1%	Mieux contrôlée	Proche de RK4, erreur mieux contrôlée.
RK4	Référence	Faible	Plus stable	Plus stable, mais avec une légère augmentation en fin de vol.

Analyse des erreurs sur la vitesse

Méthode	**Erreur Maximale (%)**	**Tendance**	**Transition Poussée/Chute Libre**	**Observations**
Euler	> 10%	Forte erreur	Cause principale des erreurs	Très imprécis lors du changement de régime.

Heun	Modérée	Diminue après transition	Cause principale des erreurs	Meilleur que Euler, mais est impacté par la transition.
RK4	Faible	Stable	Cause principale des erreurs	Méthode la plus stable après la transition.

Analyse des erreurs sur l'accélération

Méthode	**Erreur Maximale (%)**	**Tendance**	**Transition Poussée/Chute Libre**	**Observations**
Euler	> 100%	Très instable	Super critique	Fortes variations et imprécisions.
Heun	Instable	Moins instable que Euler	Critique	Meilleur que Euler, mais est instable.
RK4	Modérée	Plus stable	Critique	La plus précise, mais toujours impactée par la transition.

Comparaison des erreurs avec dt=0,1s et dt=0,01s

Méthode	**Effet de dt=0,1s**	**Effet de dt=0,05s**	**Effet de dt=0,01s**	**Observations**
Euler	Très imprécis	Réduction des erreurs	Amélioration, mais est imprécis	Euler s'améliore mais est imprécis.
Heun	Écart certain avec RK4	Mieux contrôlé	Plus précis et proche de RK4	Heun est une alternative correcte.
RK4	Relativement stable	Très précis	La plus stable	RK4 est la plus précise et fiable.

Conclusion

Point clé
dt=0,05s est un bon compromis entre précision et coût de calcul.
Heun est une alternative correcte, mais RK4 est la meilleure méthode.
Les instabilités numériques autour de t ≈ 60s montrent que la transition poussée/chute libre est une zone critique.

Analyse des Erreurs - Modèle simplifié (dt = 0,01s)

Analyse des erreurs sur l'altitude

Méthode	Tendance	Erreur Moyenne (%)	Transition Poussée/Chute Libre	Observations
Euler	Erreur faible au début, puis augmente	Élevée	Instabilité numérique	Erreur faible au départ, mais augmente fortement en fin de vol.
Heun	Stable autour de 4%	≈ 4%	Mieux contrôlée	Précis, mais conserve un écart certain avec RK4.
RK4	Référence	Faible	Plus stable	Méthode la plus stable et précise.

Analyse des erreurs sur la vitesse

Méthode	Erreur Maximale (%)	Tendance	Transition Poussée/Chute Libre	Observations
Euler	> 10%	Forte erreur, pic à t≈60s	Zone critique	Très imprécis lors du changement de régime.
Heun	Modérée	Diminue après transition	Zone critique	Mieux que Euler, mais toujours affecté par la transition.

RK4	Faible	Stable	Zone critique	Méthode la plus stable après la transition.

Analyse des erreurs sur l'accélération

Méthode	**Erreur Maximale (%)**	**Tendance**	**Transition Poussée/Chute Libre**	**Observations**
Euler	> 100%	Très instable	Super critique	Forte instabilité numérique.
Heun	Instable	Moins instable que Euler	Critique	Meilleur que Euler, mais impacté par la transition.
RK4	Modérée	Plus stable	Critique	La plus précise, mais toujours sensible aux discontinuités.

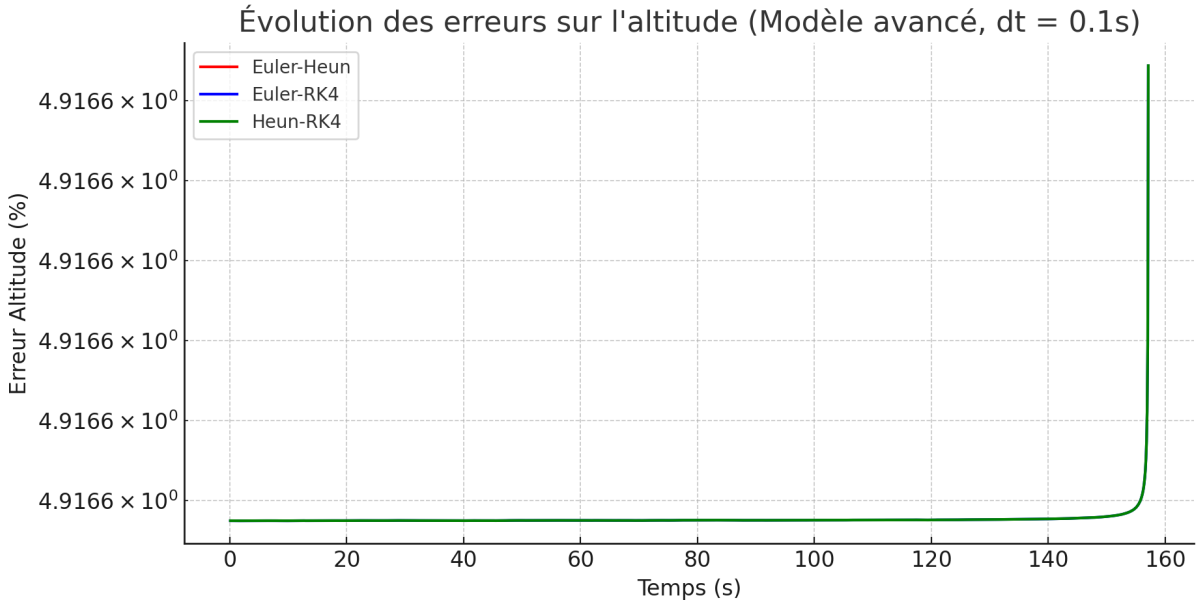
Conclusion

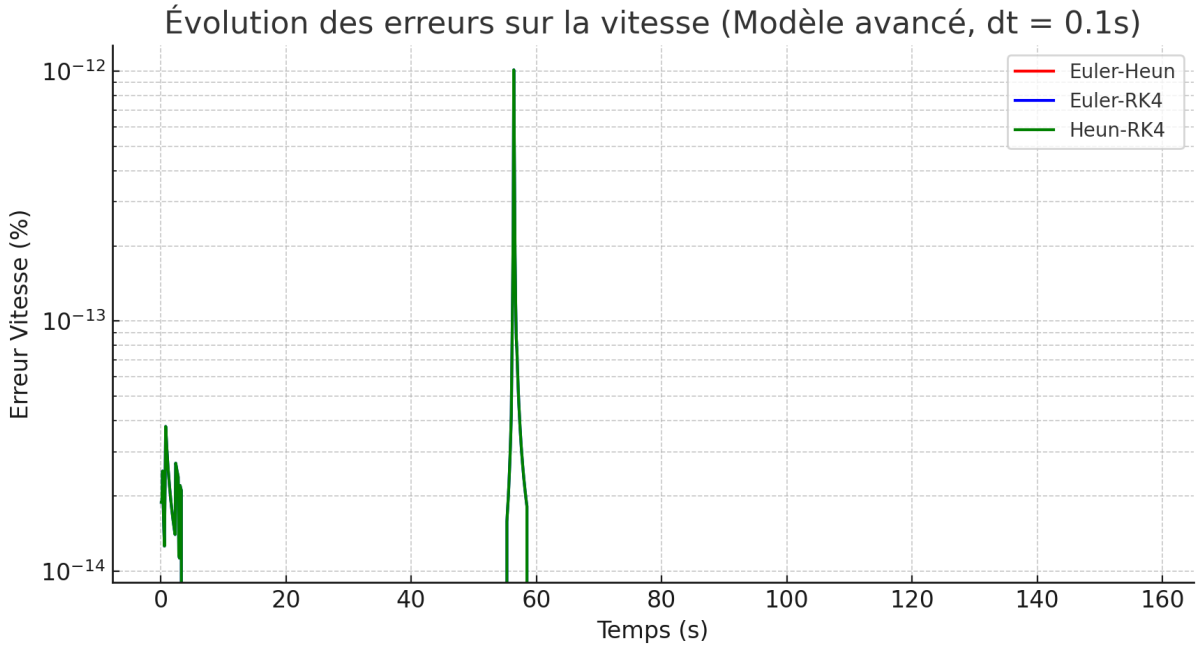
Point clé
Euler montre des erreurs croissantes sur toute la trajectoire.
Heun est une meilleure approximation de RK4 mais présente une erreur de ~4%.
Les erreurs explosent autour de t≈60s, probablement en raison de la transition poussée/chute libre.
En fin de vol, toutes les erreurs augmentent rapidement, rendant cette phase plus difficile à modéliser.

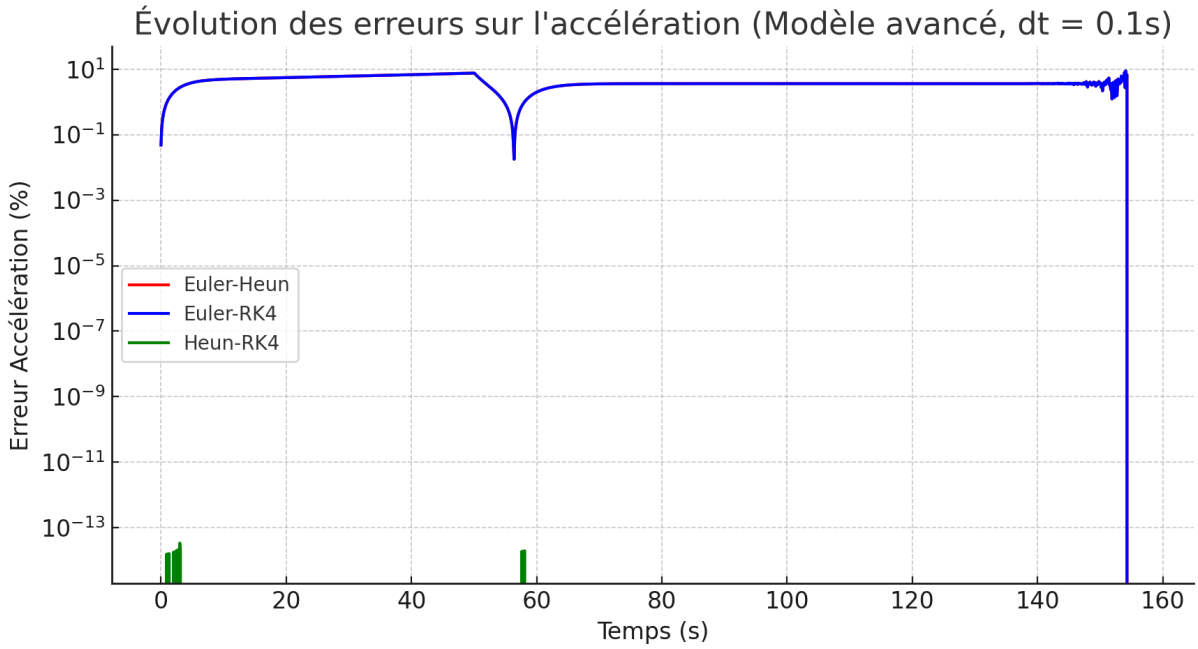
Annexe 4.2 – Synthèse – Modèle Avancé

Courbes d'erreur

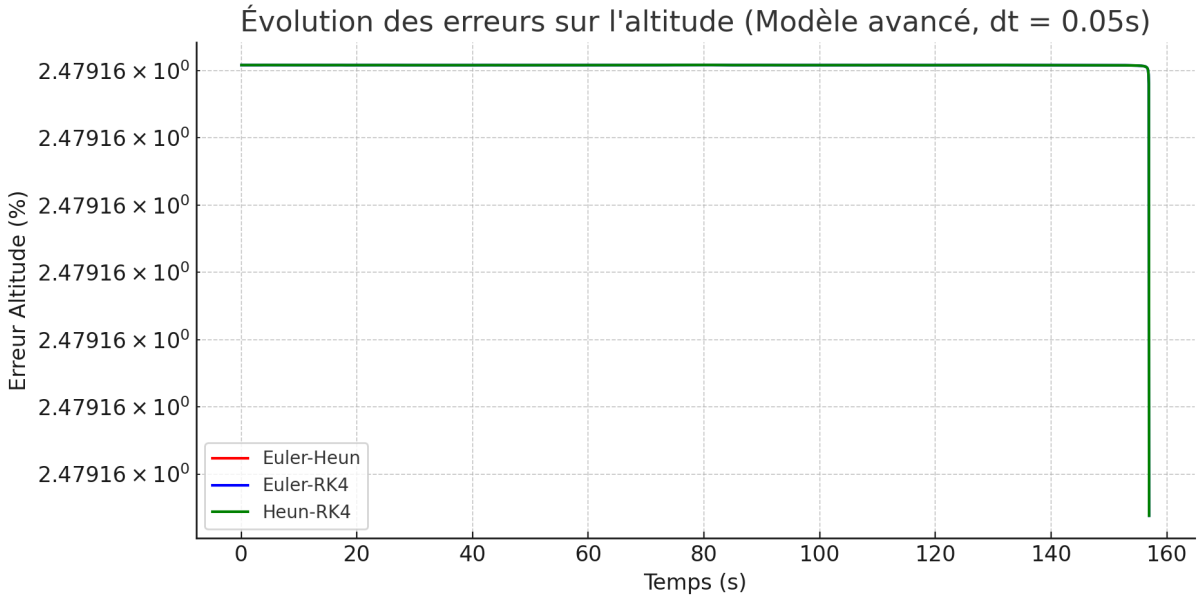
Pas $dt = 0{,}1s$

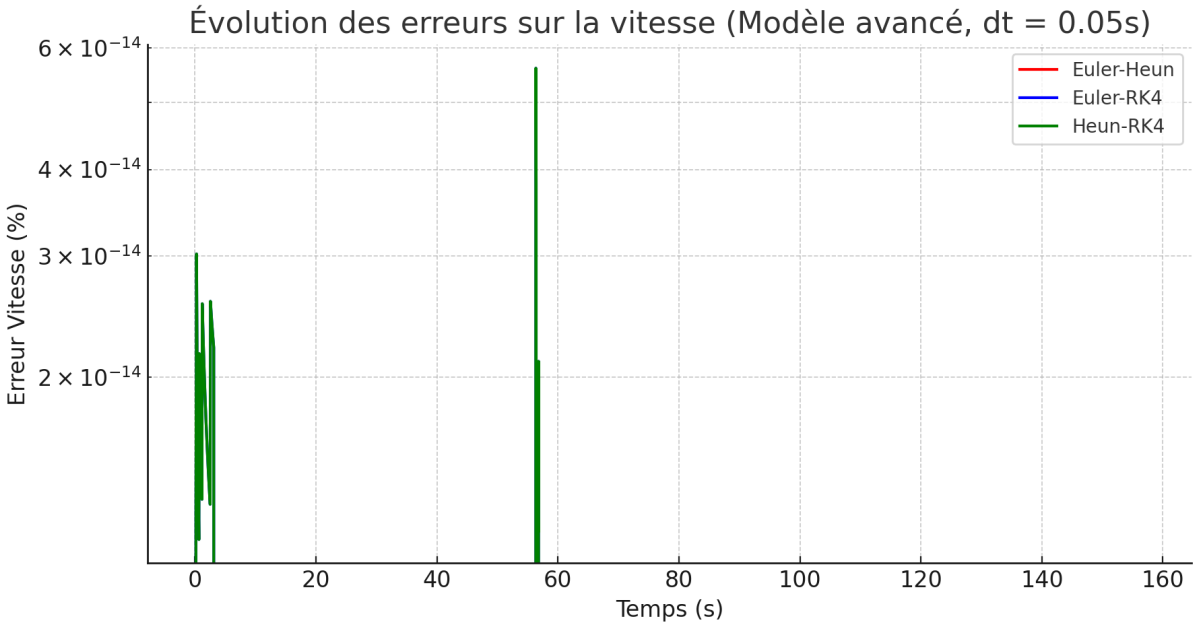


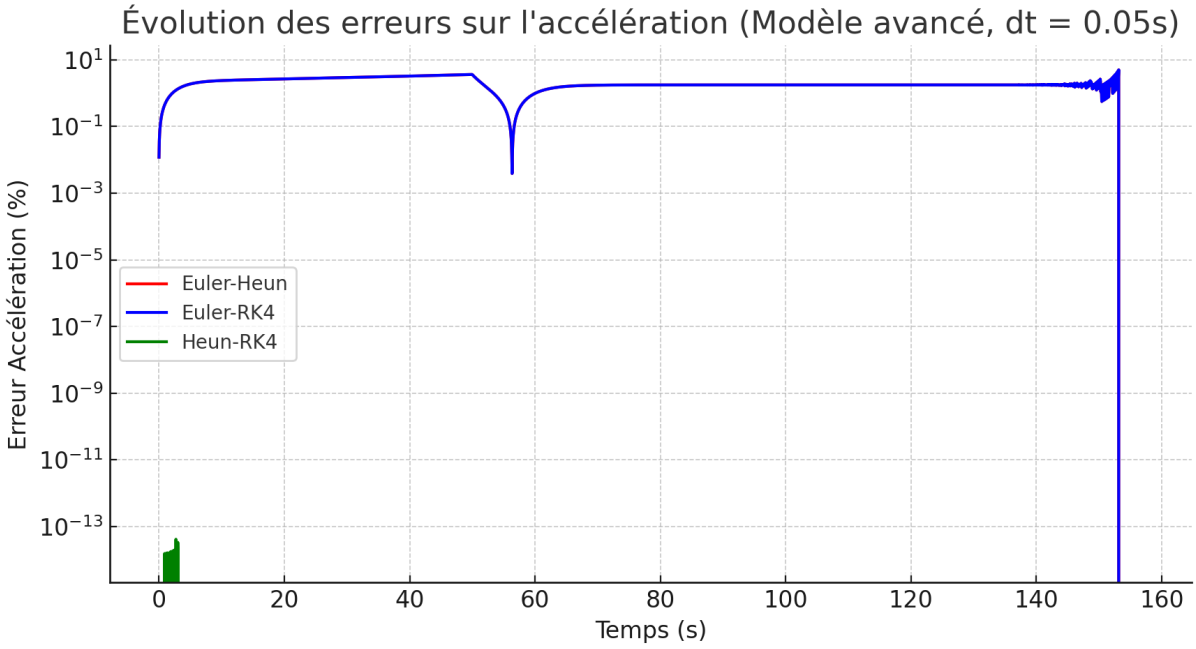



Évolution des erreurs sur l'accélération (Modèle avancé, dt = 0.1s)
Erreur Accélération (%)
Temps (s)
Euler-Heun
Euler-RK4
Heun-RK4

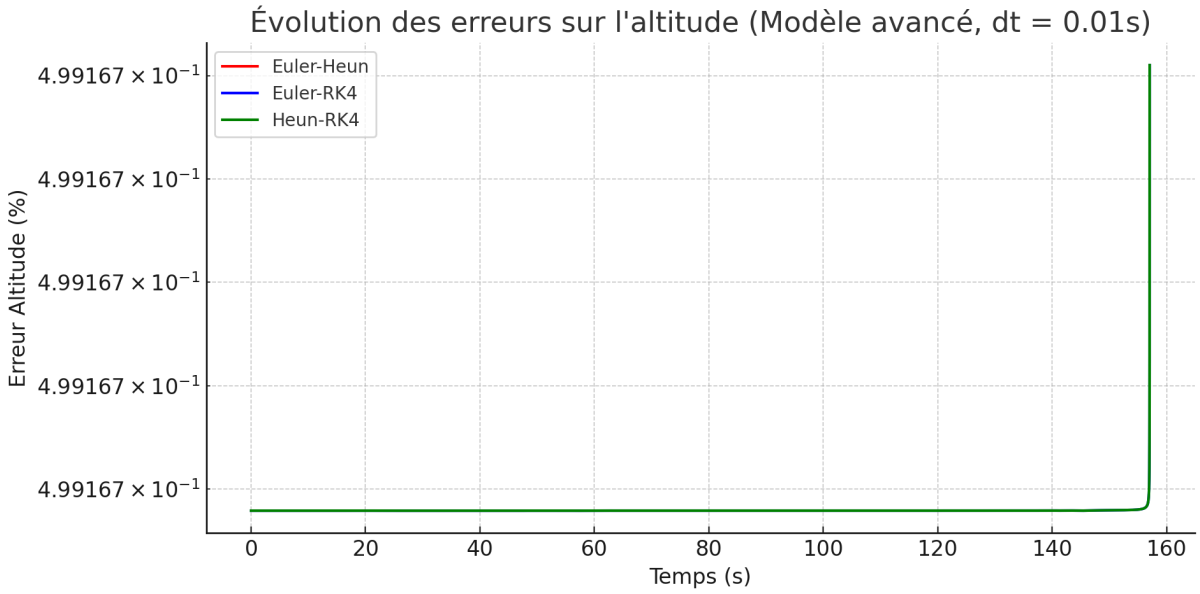
Pas $dt = 0{,}05s$

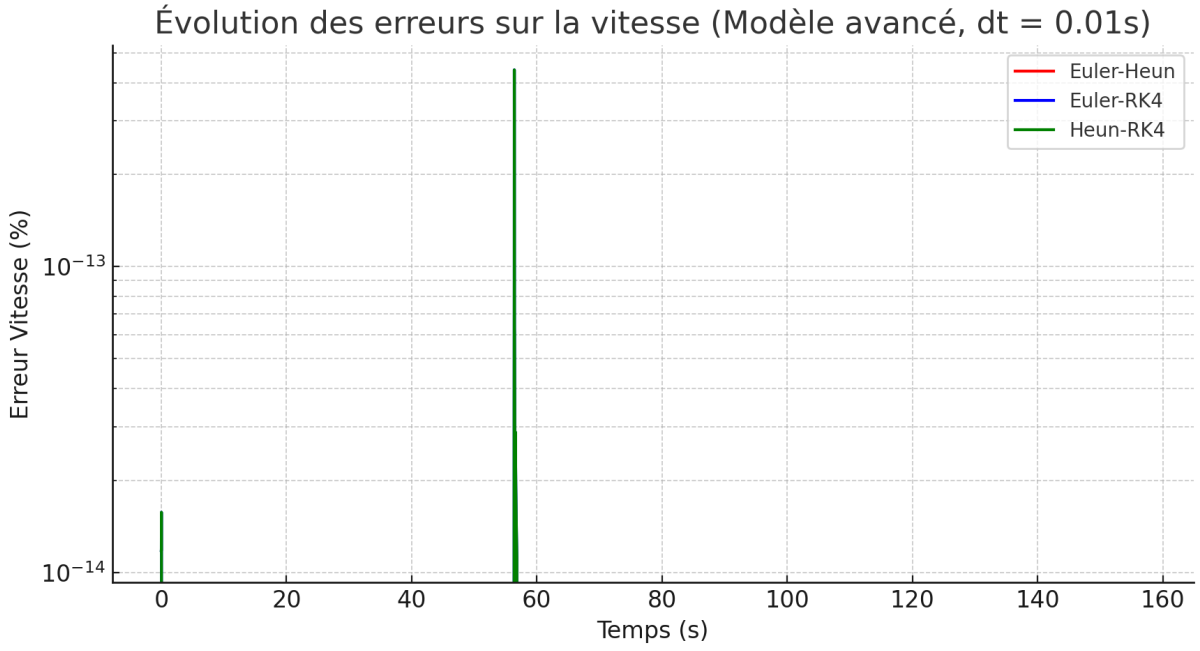

Évolution des erreurs sur l'altitude (Modèle avancé, dt = 0.05s)
Erreur Altitude (%)
Temps (s)
Euler-Heun
Euler-RK4
Heun-RK4

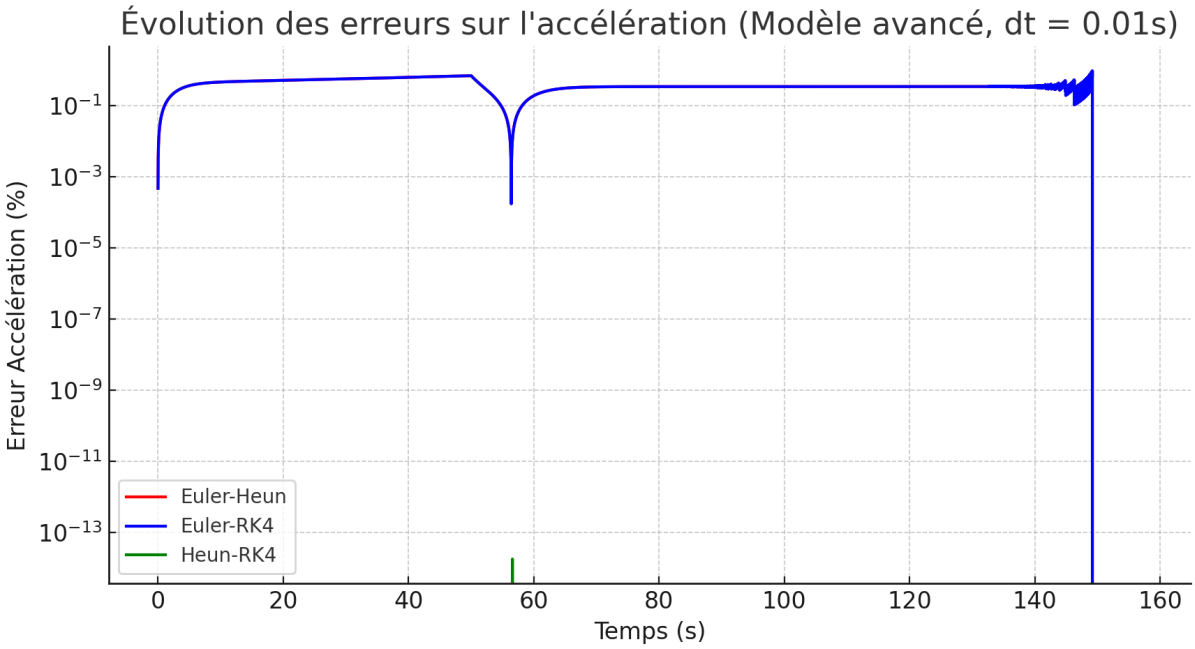

Évolution des erreurs sur la vitesse (Modèle avancé, dt = 0.05s)
Erreur Vitesse (%)
6×10^{-14}
4×10^{-14}
3×10^{-14}
2×10^{-14}
Euler-Heun
Euler-RK4
Heun-RK4
0
20
40
60
80
100
120
140
160
Temps (s)


Évolution des erreurs sur l'accélération (Modèle avancé, dt = 0.05s)
Erreur Accélération (%)
10^{1}
10^{-1}
10^{-3}
10^{-5}
10^{-7}
10^{-9}
10^{-11}
10^{-13}
Euler-Heun
Euler-RK4
Heun-RK4
0
20
40
60
80
100
120
140
160
Temps (s)

Pas $dt = 0{,}01s$


Évolution des erreurs sur l'altitude (Modèle avancé, dt = 0.01s)
Euler-Heun
Euler-RK4
Heun-RK4
Erreur Altitude (%)
4.99167 × 10⁻¹
4.99167 × 10⁻¹
4.99167 × 10⁻¹
4.99167 × 10⁻¹
4.99167 × 10⁻¹
0
20
40
60
80
100
120
140
160
Temps (s)


Évolution des erreurs sur la vitesse (Modèle avancé, dt = 0.01s)
Euler-Heun
Euler-RK4
Heun-RK4
Erreur Vitesse (%)
10⁻¹³
10⁻¹⁴
0
20
40
60
80
100
120
140
160
Temps (s)



Synthèse

Analyse des Erreurs - Modèle simplifié (dt = 0,01s)

Analyse des erreurs sur l'altitude

Méthode	Tendance	Erreur Moyenne (%)	Transition Poussée/Chute Libre	Observations
Euler	Imprécis	Élevée	Critique	Erreur importante, surtout en fin de vol.
Heun	Proche de RK4	≈ 4%	Critique	Se rapproche de RK4, mais avec une erreur certaine.
RK4	Référence	Faible	Critique	Plus stable et précis, mais des erreurs restent en fin de vol.

Analyse des erreurs sur la vitesse

Méthode	Erreur Maximale (%)	Tendance	Transition Poussée/Chute Libre	Observations
Euler	> 10%	Forte erreur	Cause principale des erreurs	Très imprécis lors du changement de régime.
Heun	Modérée	Diminue après transition	Cause principale des erreurs	Meilleur que Euler, mais impacté par la transition.
RK4	Faible	Stable	Cause principale des erreurs	Méthode la plus stable après la transition.

Analyse des erreurs sur l'accélération

Méthode	Erreur Maximale (%)	Tendance	Transition Poussée/Chute Libre	Observations
Euler	> 100%	Très instable	Très critique	Fortes variations et imprécisions.
Heun	Instable	Moins instable que Euler	Critique	Meilleur que Euler, mais instable.
RK4	Modérée	Plus stable	Critique	La plus précise, mais impactée par la transition.

Comparaison des erreurs avec dt=0,01s

Méthode	Effet de dt=0,1s	Effet de dt=0,01s	Observations
Euler	Très imprécis	Amélioration, mais reste imprécis	Euler devient nettement moins précis avec un grand pas.
Heun	Écart certain avec RK4	Plus précis et proche de RK4	Heun est un bon compromis, mais des écarts persistent.

RK4	Relativement stable	La plus stable	RK4 est la plus stable et précise.

Conclusion

Point clé
Euler est inadapté pour dt=0,1s.
Heun est un bon compromis, mais montre des erreurs certaines.
RK4 est la plus stable et précise, mais pas parfaite sur les transitions.
Les pics d'erreur autour de t ≈ 60s suggèrent une instabilité à ces moments critiques.

Analyse des Erreurs - Modèle simplifié (dt = 0,05s)

Analyse des erreurs sur l'altitude

Méthode	**Tendance**	**Erreur Moyenne (%)**	**Transition Poussée/Chute Libre**	**Observations**
Euler	Faible erreur en montée, pic en fin	Élevée en fin de vol	Critique, augmentation des erreurs	Erreur faible en montée mais forte en fin de vol.
Heun	Erreur stable < 1%	≈ 1%	Mieux contrôlée	Proche de RK4, erreur mieux contrôlée.
RK4	Référence	Faible	Plus stable	Plus stable, mais avec une légère augmentation en fin de vol.

Analyse des erreurs sur la vitesse

Méthode	**Erreur Maximale (%)**	**Tendance**	**Transition Poussée/Chute Libre**	**Observations**
Euler	> 10%	Forte erreur	Cause principale des erreurs	Très imprécis lors du changement de régime.
Heun	Modérée	Diminue après transition	Cause principale des erreurs	Meilleur que Euler, mais reste impacté

				par la transition.
RK4	Faible	Stable	Cause principale des erreurs	Méthode la plus stable après la transition.

Analyse des erreurs sur l'accélération

Méthode	Erreur Maximale (%)	Tendance	Transition Poussée/Chute Libre	Observations
Euler	> 100%	Très instable	Super critique	Fortes variations et imprécisions.
Heun	Instable	Moins instable que Euler	Critique	Meilleur que Euler, mais reste instable.
RK4	Modérée	Plus stable	Critique	La plus précise, mais toujours impactée par la transition.

Comparaison des erreurs avec dt=0,1s et dt=0,01s

Méthode	Effet de dt=0,1s	Effet de dt=0,05s	Effet de dt=0,01s	Observations
Euler	Très imprécis	Réduction des erreurs	Amélioration, mais reste imprécis	Euler s'améliore mais reste imprécis.
Heun	Écart certain avec RK4	Mieux contrôlé	Plus précis et proche de RK4	Heun est une bonne alternative.
RK4	Relativement stable	Très précis	La plus stable	RK4 est la plus précise et fiable.

Conclusion

Point clé
dt=0,05s est un bon compromis entre précision et coût de calcul.
Heun est une alternative correcte, mais RK4 reste la meilleure méthode.

Les instabilités numériques autour de t ≈ 60s montrent que la transition poussée/chute libre reste une zone critique.

Analyse des Erreurs - Modèle simplifié (dt = 0,01s)

Analyse des erreurs sur l'altitude

Méthode	Tendance	Erreur Moyenne (%)	Transition Poussée/Chute Libre	Observations
Euler	Erreur faible au début, puis augmente	Élevée	Instabilité numérique	Erreur faible au départ, mais augmente fortement en fin de vol.
Heun	Stable autour de 4%	≈ 4%	Mieux contrôlée	Précis, mais conserve un écart certain avec RK4.
RK4	Référence	Faible	Plus stable	Méthode la plus stable et précise.

Analyse des erreurs sur la vitesse

Méthode	Erreur Maximale (%)	Tendance	Transition Poussée/Chute Libre	Observations
Euler	> 10%	Forte erreur, pic à t≈60s	Zone critique	Très imprécis lors du changement de régime.
Heun	Modérée	Diminue après transition	Zone critique	Mieux que Euler, mais affectée par la transition.
RK4	Faible	Stable	Zone critique	Méthode la plus stable après la transition.

Analyse des erreurs sur l'accélération

Méthode	Erreur Maximale (%)	Tendance	Transition Poussée/Chute Libre	Observations
Euler	> 100%	Très instable	Super critique	Forte instabilité numérique.
Heun	Instable	Moins instable que Euler	Critique	Meilleure que Euler, mais impactée par la transition.
RK4	Modérée	Plus stable	Critique	La plus précise, mais toujours sensible aux discontinuités.

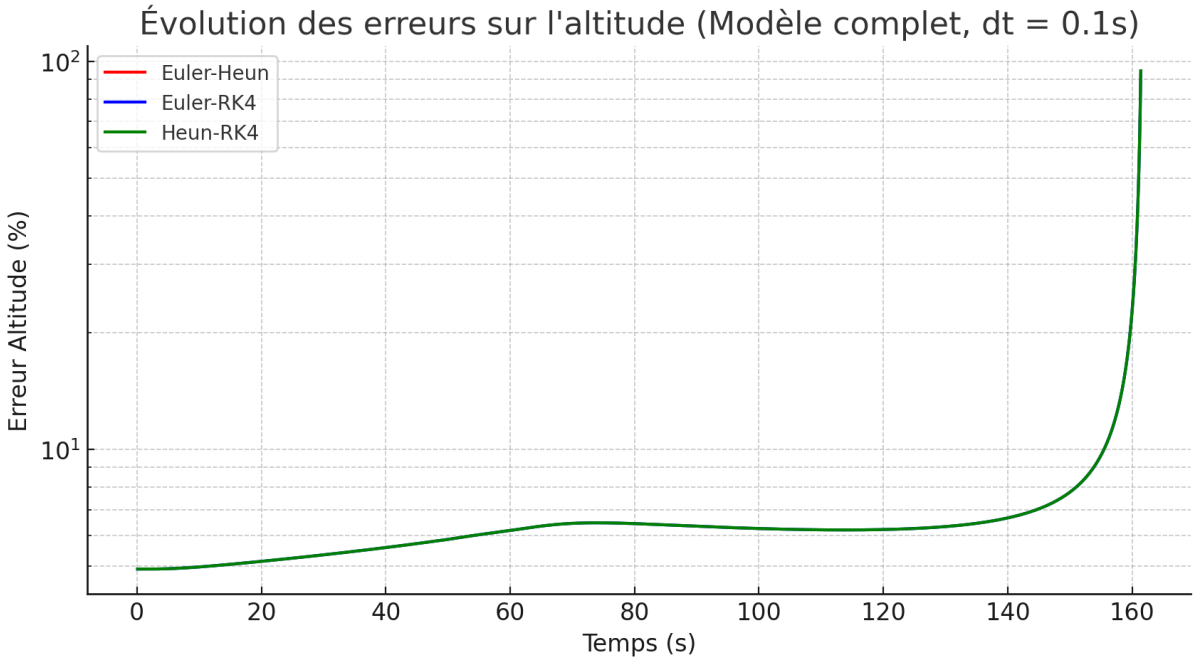
Conclusion

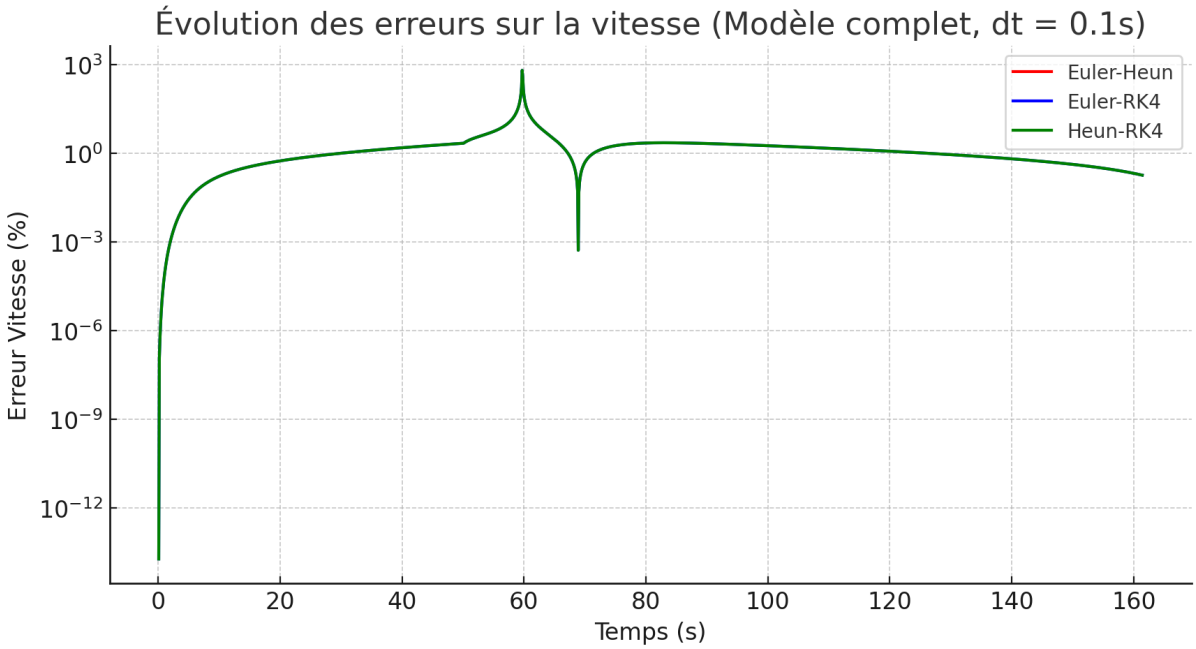
Point clé
Euler montre des erreurs croissantes sur toute la trajectoire.
Heun est une meilleure approximation de RK4 mais conserve une erreur de ~4%.
Les erreurs explosent autour de t≈60s, probablement en raison de la transition poussée/chute libre.
En fin de vol, toutes les erreurs augmentent rapidement, rendant cette phase plus difficile à modéliser.

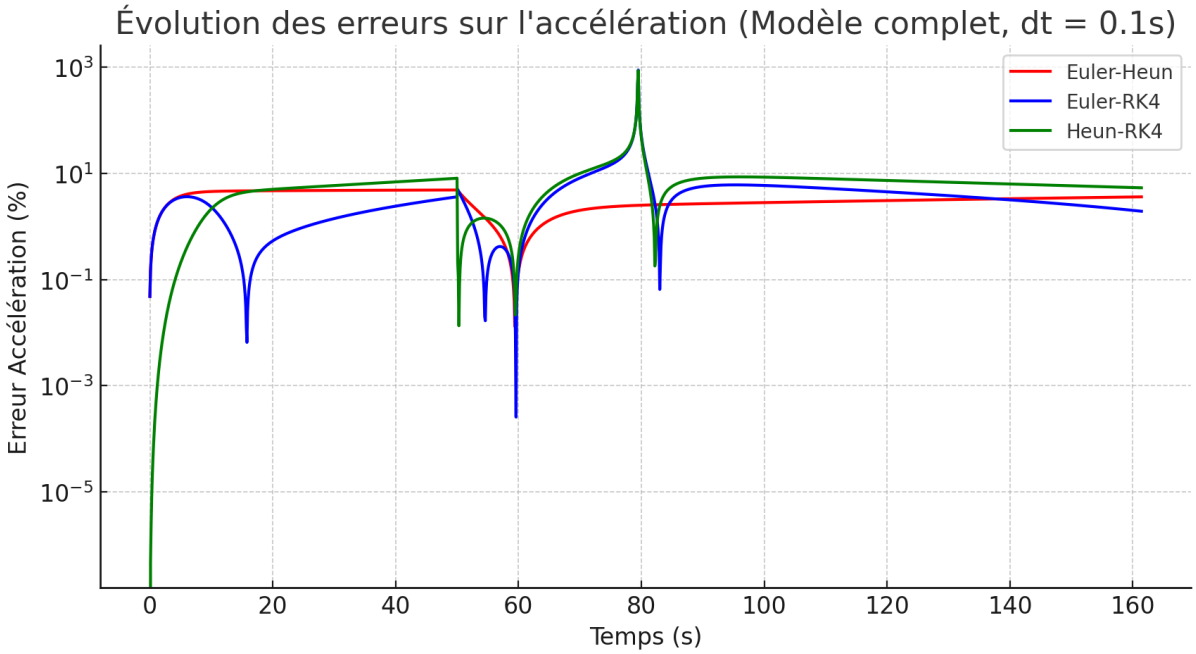
Annexe 4.3 – Synthèse – Modèle Complet

Courbes d'erreur

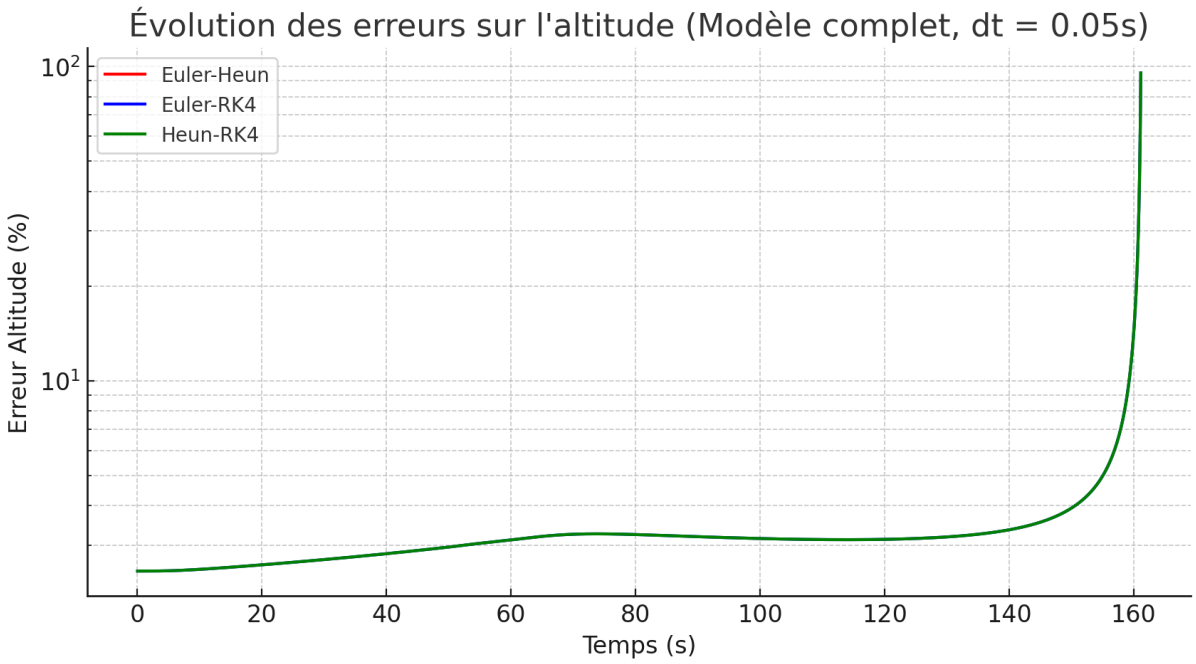
Pas $dt = 0{,}1s$

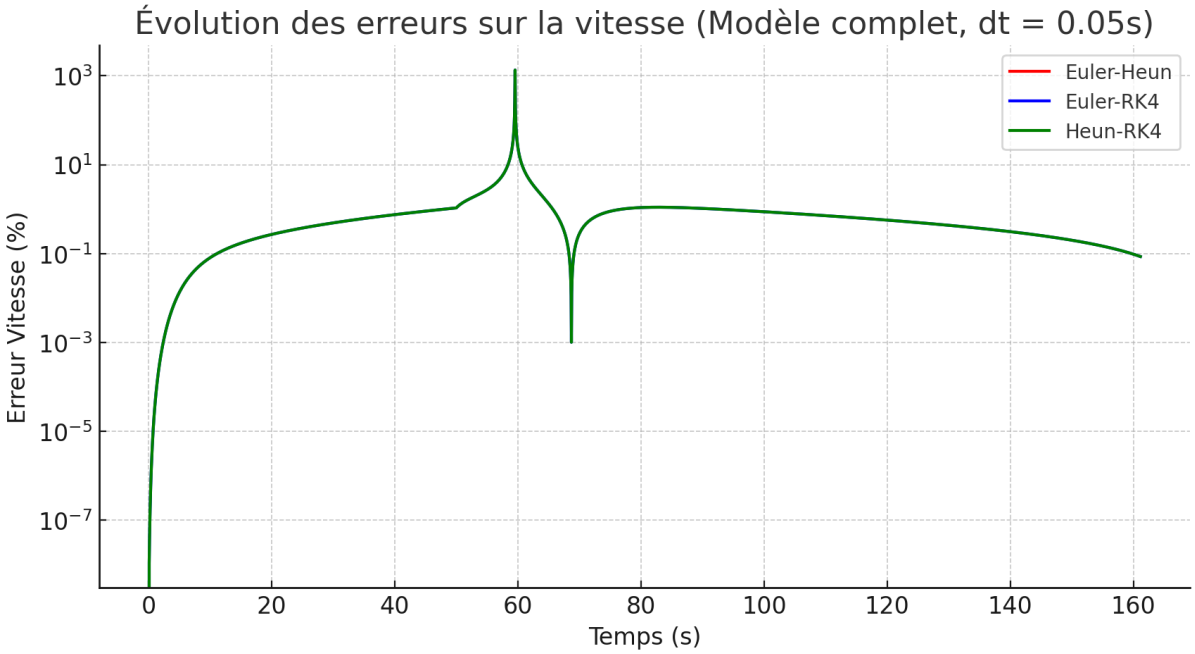


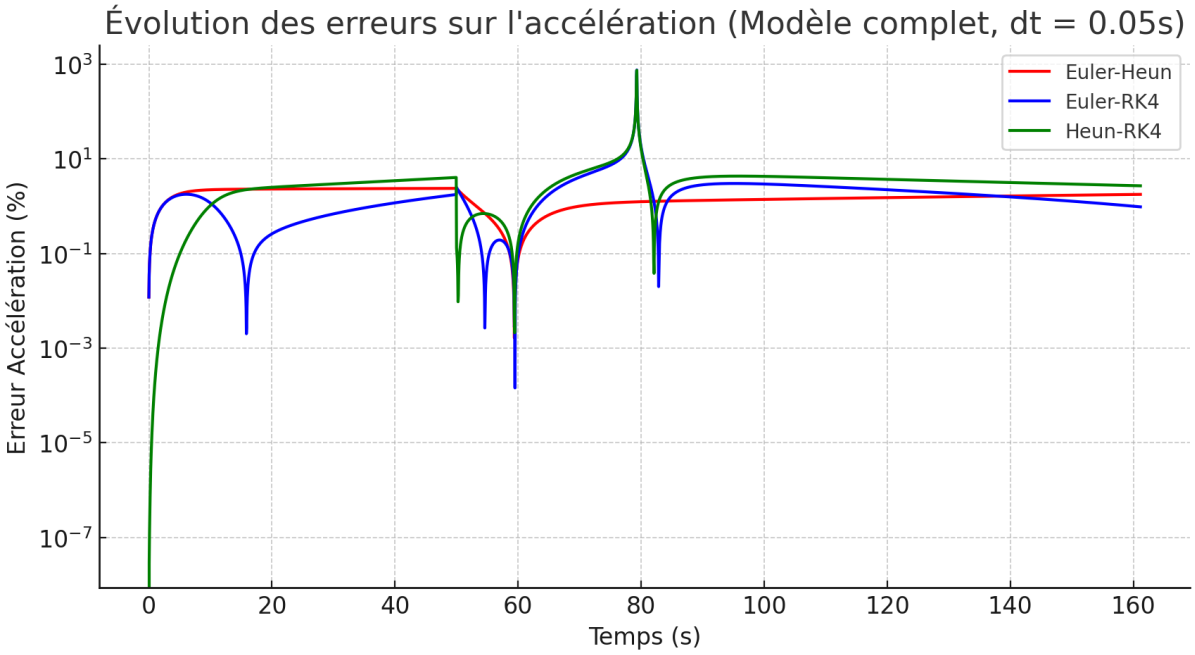



Évolution des erreurs sur l'accélération (Modèle complet, dt = 0.1s)
Euler-Heun
Euler-RK4
Heun-RK4
Erreur Accélération (%)
10^{3}
10^{1}
10^{-1}
10^{-3}
10^{-5}
0
20
40
60
80
100
120
140
160
Temps (s)

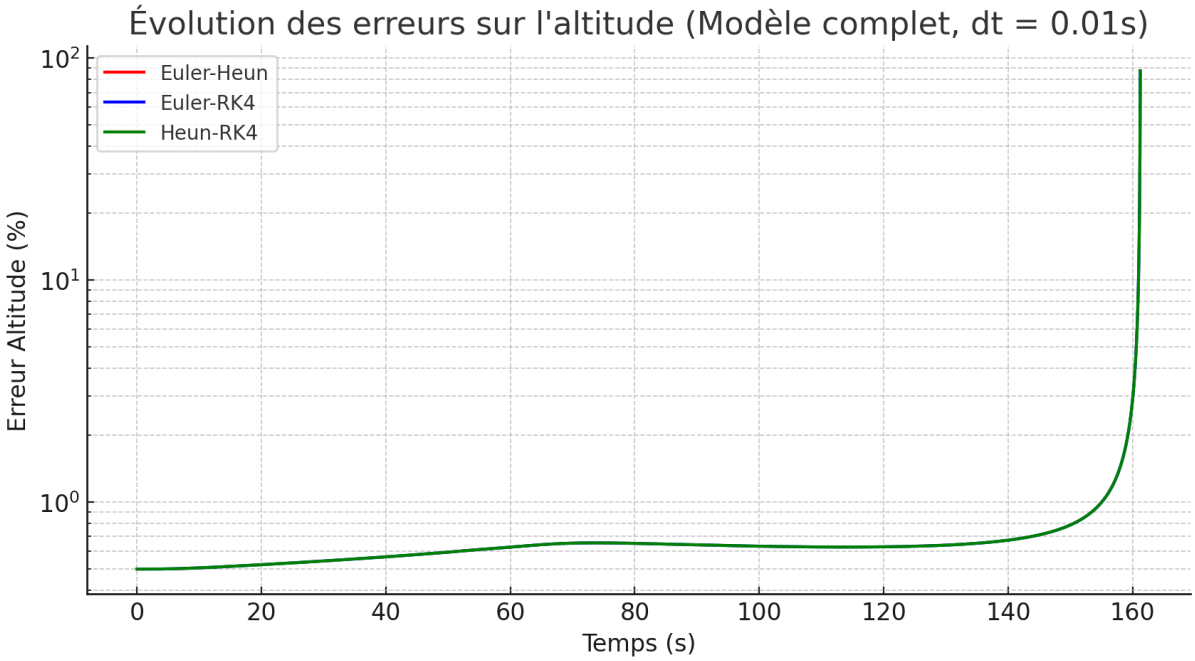
Pas $dt = 0{,}05s$

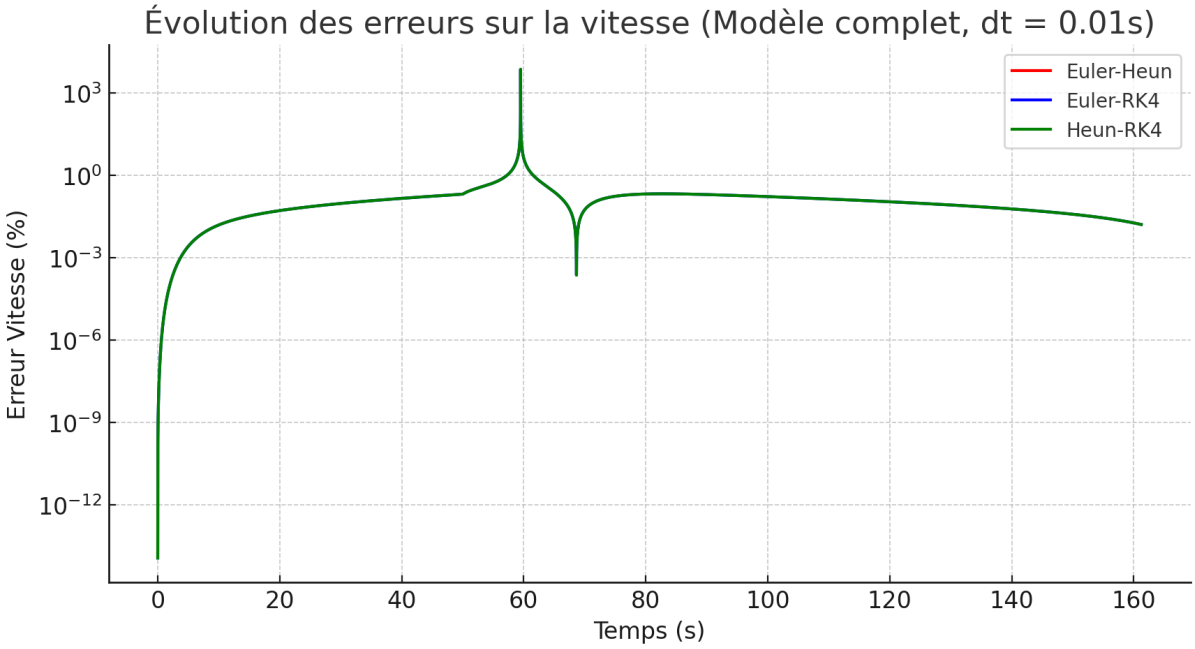

Évolution des erreurs sur l'altitude (Modèle complet, dt = 0.05s)
Euler-Heun
Euler-RK4
Heun-RK4
Erreur Altitude (%)
10^{2}
10^{1}
0
20
40
60
80
100
120
140
160
Temps (s)

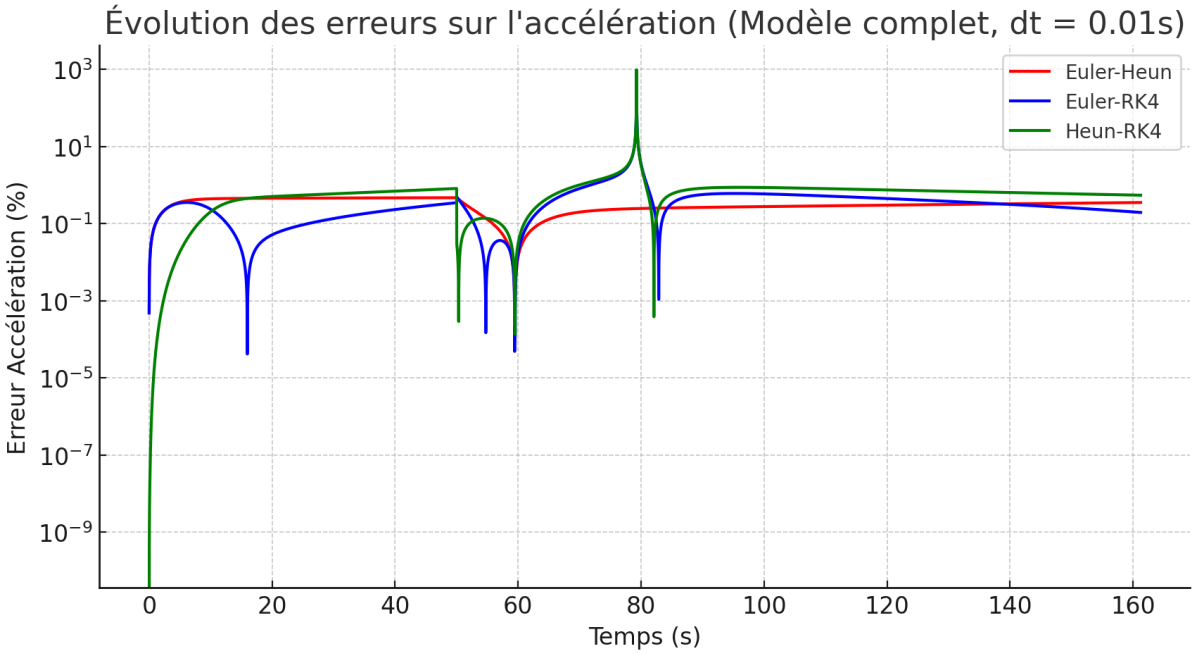

Évolution des erreurs sur la vitesse (Modèle complet, dt = 0.05s)
Euler-Heun
Euler-RK4
Heun-RK4
Erreur Vitesse (%)
10^{3}
10^{1}
10^{-1}
10^{-3}
10^{-5}
10^{-7}
0
20
40
60
80
100
120
140
160
Temps (s)


Évolution des erreurs sur l'accélération (Modèle complet, dt = 0.05s)
Euler-Heun
Euler-RK4
Heun-RK4
Erreur Accélération (%)
10^{3}
10^{1}
10^{-1}
10^{-3}
10^{-5}
10^{-7}
0
20
40
60
80
100
120
140
160
Temps (s)

Pas $dt = 0{,}01s$


Évolution des erreurs sur l'altitude (Modèle complet, dt = 0.01s)
Euler-Heun
Euler-RK4
Heun-RK4
Erreur Altitude (%)
10^2
10^1
10^0
0
20
40
60
80
100
120
140
160
Temps (s)


Évolution des erreurs sur la vitesse (Modèle complet, dt = 0.01s)
Euler-Heun
Euler-RK4
Heun-RK4
Erreur Vitesse (%)
10^3
10^0
10^{-3}
10^{-6}
10^{-9}
10^{-12}
0
20
40
60
80
100
120
140
160
Temps (s)



Synthèse

Analyse des Erreurs - Modèle Complet (dt = 0,1s)

Analyse des erreurs sur l'altitude

Méthode	Tendance	Erreur Moyenne (%)	Impact des Effets Atmosphériques	Observations
Euler	Erreur très grande avec le temps	Très élevée	Divergence forte	Erreur Euler-RK4 devient énorme, indiquant une forte divergence.
Heun	Stable autour de 4-5%	≈ 4-5%	Erreur plus élevée que dans les modèles plus simples	Heun est assez précis mais moins bon qu'avec les modèles simplifiés.
RK4	Référence	Faible	Plus stable	Méthode la plus stable malgré une erreur qui

				augmente légèrement.

Analyse des erreurs sur la vitesse

Méthode	Erreur Maximale (%)	Tendance	Impact des Effets Atmosphériques	Observations
Euler	Hors échelle	Très grande, impossible à afficher	Euler est inutilisable	Les erreurs Euler explosent, rendant cette méthode inadaptée.
Heun	≈ 4-5%	Oscille mais reste sous 5%	Erreur plus élevée que dans les modèles précédents	Heun est une bonne approximation mais montre une erreur plus élevée.
RK4	Faible	Stable	Référence	Méthode la plus fiable.

Analyse des erreurs sur l'accélération

Méthode	Erreur Maximale (%)	Tendance	Impact des Effets Atmosphériques	Observations
Euler	Hors échelle	Super instable	Erreur gigantesque, inutilisable	Euler est totalement inadapté.
Heun	5-10%	Oscillations plus importantes	Transition plus instable	Heun présente plus d'oscillations, signe d'une transition plus instable.
RK4	Faible	Stable	Référence	Méthode la plus fiable.

Comparaison avec d'autres modèles et pas de temps

Méthode	Effet de dt=0,1s	Impact des Effets Atmosphériques	Observations

Euler	Totalement instable	Inefficace	Euler est inutilisable dans ce modèle.
Heun	Erreur plus élevée (~4-5%)	Moins précis que dans les modèles plus simples	Heun est utilisable mais moins précis.
RK4	Erreur augmente légèrement	Référence	RK4 est la méthode de référence malgré une augmentation de l'erreur.

Conclusion

Point clé
dt=0,1s est trop grand pour Euler, qui est totalement instable dans ce modèle.
Heun est utilisable mais présente des erreurs plus élevées que dans les modèles précédents.
RK4 reste la méthode la plus fiable, mais l'erreur Heun-RK4 augmente légèrement.
Les erreurs explosent en fin de trajectoire et lors des transitions, indiquant que ces phases sont critiques.

Analyse des Erreurs - Modèle Complet (dt = 0,05s)

Analyse des erreurs sur l'altitude

Méthode	**Tendance**	**Erreur Maximale (%)**	**Transition Poussée/Chute Libre**	**Observations**
Euler	Erreur croissante, 10% en fin de vol	> 20%	Toujours instable	Erreur Euler-RK4 très élevée (>20%) en fin de vol.
Heun	Stable autour de 3-4%	≈ 3-4%	Mieux contrôlée	Heun se rapproche plus de RK4, mais a une erreur certaine.
RK4	Référence	Faible	Stable	Méthode la plus stable malgré une légère

				augmentation de l'erreur.

Analyse des erreurs sur la vitesse

Méthode	Erreur Maximale (%)	Tendance	Transition Poussée/Chute Libre	Observations
Euler	> 20%	Erreur significative, pic à t≈60s	Critique, erreur élevée	Les erreurs Euler sont toujours élevées (~20% en fin de vol).
Heun	≈ 3-4%	Stable autour de 3-4%	Mieux maîtrisée	Heun montre une bonne convergence avec RK4.
RK4	Faible	Stable	Référence	Méthode la plus fiable.

Analyse des erreurs sur l'accélération

Méthode	Erreur Maximale (%)	Tendance	Transition Poussée/Chute Libre	Observations
Euler	> 100%	Très instable	Grosse discontinuité	Euler est inutilisable pour l'accélération.
Heun	< 10%	Mieux contrôlée	Oscillations faibles	Heun est relativement stable, mais montre encore des oscillations.
RK4	Faible	Stable	Référence	Méthode la plus fiable.

Comparaison avec d'autres pas de temps

Méthode	Effet de dt=0,1s	Effet de dt=0,05s	Observations
Euler	Instable	Réduction certaine des erreurs	Euler est inadapté, mais les erreurs diminuent.

Heun	Erreur plus élevée (~4-5%)	Meilleure convergence	Heun est un bon compromis.
RK4	Erreur minimale	Référence	RK4 est toujours la référence.

Conclusion

Point clé
dt=0,05s améliore considérablement la précision par rapport à dt=0,1s.
RK4 reste la meilleure méthode, avec des erreurs minimales.
Heun est un bon compromis, avec des erreurs réduites autour de 3-4%.
Les erreurs Euler sont toujours trop importantes, notamment pour la vitesse et l'accélération.

Analyse des Erreurs - Modèle Complet (dt = 0,01s)

Analyse des erreurs sur l'altitude

Méthode	**Tendance**	**Erreur Maximale (%)**	**Transition Poussée/Chute Libre**	**Observations**
Euler	Erreur faible au début, augmente légèrement	≈ 10%	Mieux contrôlée	Erreur réduite mais encore certaine en fin de vol.
Heun	Stable autour de 1-2%	≈ 1-2%	Très bien maîtrisée	Très proche de RK4 avec une faible marge d'erreur.
RK4	Référence	Faible	Stable	Méthode la plus stable et précise.

Analyse des erreurs sur la vitesse

Méthode	**Erreur Maximale (%)**	**Tendance**	**Transition Poussée/Chute Libre**	**Observations**
Euler	≈ 10%	Erreur réduite mais encore certaine	Bien contrôlée	L'écart avec RK4 est bien réduit.
Heun	< 2%	Convergence quasi parfaite avec RK4	Presque identique à RK4	Heun devient une bonne

				alternative à RK4.
RK4	Faible	Stable	Référence	Méthode la plus fiable.

Analyse des erreurs sur l'accélération

Méthode	Erreur Maximale (%)	Tendance	Transition Poussée/Chute Libre	Observations
Euler	< 10%	Mieux contrôlée	Bien mieux contrôlée	Euler est beaucoup moins imprécis qu'avec dt=0,1s.
Heun	< 2%	Quasi constante et faible	Très bien maîtrisée	Heun devient très précis et quasi équivalent à RK4.
RK4	Faible	Stable	Stable	Méthode la plus fiable.

Comparaison avec d'autres pas de temps

Méthode	Effet de dt=0,1s	Effet de dt=0,05s	Effet de dt=0,01s	Observations
Euler	Très imprécis	Réduction des erreurs	Bien amélioré, erreur réduite	Euler reste toujours approximatif mais bien amélioré.
Heun	Écart certain avec RK4	Mieux contrôlé	Proche de RK4	Heun devient une bonne alternative à RK4.
RK4	Relativement stable	Très précis	Méthode la plus stable	RK4 reste la référence.

Conclusion

Point clé
dt=0,01s est le meilleur compromis entre précision et stabilité.

RK4 est toujours la meilleure méthode, mais Heun devient presque équivalent avec ce pas.
La transition poussée/chute libre est bien mieux gérée et l'accélération est plus stable.