Exploration d’un modèle de dissipation des ressources : de la théorie à la pratique avec Python

# Introduction

Les problématiques liées à la dissipation des ressources, qu’elles soient énergétiques ou matérielles, sont au cœur de la réflexion scientifique contemporaine. Grâce à la puissance de Python et à la clarté des modèles mathématiques, il est désormais possible de simuler, d’analyser et d’optimiser la dissipation pour une grande variété de systèmes. Cet article propose une exploration détaillée du code Python trouvé dans main.py du projet « modèle de dissipation des ressources », en s’appuyant sur les principes exposés dans l’article scientifique joint au projet.

# Fondements théoriques du modèle

À la base de toute simulation fiable se trouve une modélisation rigoureuse. L’article PDF met en lumière la façon dont la dissipation des ressources, souvent exprimée sous forme de pertes dans un système (thermique, mécanique, économique, etc.), peut être quantifiée et prédite grâce à des équations différentielles ou des systèmes dynamiques. Le modèle mathématique formalise la relation entre l’apport, l’usage et la perte de ressources dans un système fermé ou ouvert, en intégrant des paramètres comme l’efficacité, la résistance aux pertes, ou encore la capacité d’adaptation du système face à une contrainte.

# Présentation du code Python (main.py)

Le fichier main.py implémente l’ensemble des équations et algorithmes décrits dans l’article PDF, permettant une mise en pratique directe du modèle théorique. Typiquement, le code s’organise de la façon suivante :

* Définition des paramètres du modèle : valeurs initiales, constantes physiques, coefficients d’efficacité ou de perte, etc.
* Implémentation des fonctions de calcul : calcul de la dissipation à chaque étape, mise à jour des ressources, intégration numérique si nécessaire.
* Simulation et visualisation : génération de courbes d’évolution des ressources, affichage des résultats comparatifs selon différents scénarios ou paramètres.

Grâce à la modularité de Python, il est possible d’ajuster les variables ou d’ajouter de nouveaux modules pour tester d’autres hypothèses.

## Exemple d’utilisation

Vous soulignez à juste titre que le cœur du projet concerne la dissipation de la biomasse, et non la production ou la gestion de l’énergie électrique. Il est donc opportun de recentrer l’analyse sur la dynamique de la biomasse : dans ce contexte, le modèle et le code Python permettent de simuler le devenir de la biomasse dans un écosystème ou un système industriel, en prenant en compte l’apport, la consommation, la transformation et les pertes par dissipation (respiration, déchets, fuites métaboliques). Les paramètres et scénarios testés servent alors à évaluer l’influence des facteurs écologiques ou technologiques sur la préservation et la valorisation de la biomasse, orientant la réflexion vers l’optimisation des flux biologiques ou la réduction des pertes dans les chaînes de transformation.

Analyse et application du code source et des fondements scientifiques

Il est tout à fait envisageable de comparer les scénarios « volterra » et « differentiel » du paramètre "model\_type" en s’appuyant sur les résultats générés lors de la première expérimentation du code, disponibles dans le dossier D:\PythonProjects\modele\_dissipation\_ressources\output. Cette démarche permet d’étudier, pour chaque modèle, la dynamique de dissipation des ressources simulée, de mettre en lumière les différences structurelles ou comportementales entre leurs prédictions, et d’évaluer l’impact des paramètres spécifiques à chaque approche. Une telle comparaison éclaire les choix méthodologiques et oriente les interprétations scientifiques, en offrant une base empirique solide à l’analyse.

## Paramètres du modèle et valeurs usuelles

|  |  |  |
| --- | --- | --- |
| Symbole | Signification | Valeurs typiques / Usuelles |
| *r\_growth* | Taux de croissance intrinsèque de la biomasse | 0.2 à 1.0 (souvent 0.5 à 0.8) |
| *K* | Capacité de charge de l’écosystème | 500 à 20 000  Nous le fixons à 5 000 |
| *x0* | Biomasse initiale | Généralement 0,2K à 0,8K  Nous le fixons à 500 |
| *dissipation\_rate* | Coefficient de dissipation cumulative | 0.005 à 0.05 (souvent 0.01) |
| *T* | Horizon temporel (années) | 5 à 50 (souvent 10) |
| *unit\_cost* | Prix unitaire de la biomasse récoltée | Nous le fixons à 2 |
| *unit\_price* | Coût unitaire de l’effort de récolte | Nous le fixons à 10 |
| *interest\_rate* | Taux d’actualisation | 0.03 à 0.1 (souvent 0.05) |
| *E* | Effort maximal admissible | 0,1K à 1K (cas extrême |

# Analyse des résultats et perspectives

L’analyse des sorties du code Python révèle la sensibilité du système aux paramètres choisis. On observe souvent que de petits changements sur certains coefficients entraînent des effets non linéaires sur la dissipation totale. Ce constat ouvre la voie à des optimisations tangibles et ciblées, que ce soit dans la conception de systèmes physiques ou dans l’élaboration de politiques de gestion des ressources.

## Analyse des résultats de simulation obtenus

### Cas : équation différentielle

Une image contenant capture d’écran, texte, ligne

Le contenu généré par l’IA peut être incorrect.

Trois périodes consécutives se distinguent sur la durée de planification :

1. La période A au début de la planification dépourvue de toute extraction de biomasse, permet d’atteindre presque 50% de la capacité de charge (en 24 périodes consécutives, l’équivalent de 2 ans). Evidemment, aucun profit ne sera réalisé pendant cette 1ère phase de constitution de la biomasse (période d’investissement).
2. La phase A est suivie immédiatement de la période B assez longue et constante sur tous les plans :
   1. Un effort d’extraction fixe et égal à un peu plus de 0,5, autrement dit, une extraction de la moitié de la biomasse disponible ou le un quart (¼) de la capacité de charge
   2. Une extraction constante et régulière valant presque 50% de la biomasse disponible
3. La phase finale C symétrique à la phase A mais tout en gardant la biomasse égale à au moins le ¼ de la capacité de charge jusqu‘à la fin de période de planification.
4. Quant à l’effort déployé (courbe rouge limitée entre 0 et 1) tout au long de la période de planification, constitue le levier qui permet de réguler l’extraction :
   1. Il est **nul** pendant la phase A
   2. Il est égal approximativement à **0,52** pendant toute la phase B
   3. Il est au maximum (=**1**) en phase finale C

Ci-après une simulation d’une planification ayant les mêmes paramètres sur une **période de 50 ans** afin de confirmer le constat global ci-dessus.

Une image contenant texte, diagramme, ligne, Tracé

Le contenu généré par l’IA peut être incorrect.

### Cas : équation de Volterra

L’analyse des résultats de simulations obtenus dans le cas d’un système gouverné par une équation de type **Volterra** est relativement plus complexe à interpréter, voire à mettre en place d’une manière simple et compréhensible.

Une image contenant texte, capture d’écran, diagramme, ligne

Le contenu généré par l’IA peut être incorrect.

1. Pendant les phases numérotées uvwx illustrées sur les graphes ci-dessous :
   1. La biomasse disponible est identique à celle récoltée
   2. Cela est justifié par le fait que l’effort de récolte est au maximum, i.e., = E = 1.0
   3. Cela n’a pas d’incidence sur la disponibilité de la biomasse sur toute la période de planification de 30 ans
   4. Hormis la phase u, les phases suivantes de récolte maximale semblent être périodiques avec une amplitude temporelle identique (un essai peut être réalisé sur des périodes plus longues pour le confirmer)
2. Il existe **des périodes pendant lesquelles** **aucune récolte n’est effectuée** (celles qui correspondent à un effort nul) pendant lesquelles la biomasse reprend la direction de la croissance :
   1. **Période A :** pendant les 28 premiers mois de planification le temps que la biomasse puisse croître suffisamment pour être récoltée
   2. **Période B :** située entre les phases u, v qui dure de la période 184 à la période 208 pour permettre l’accroissement de la biomasse disponible
   3. **Périodes C et D :** situées respectivement entre les phases v, w et w, x qui permettent d’atteindre le niveau le plus haut de la biomasse 2341 (presque 50% de la capacité maximale de contenance K = 5 000)
3. En dehors des phases numérotées par des chiffres uvwx ou par les lettres A, B, C et D, les phases intermédiaires sont transitoires avec un effort variant strictement entre 0 et 1 et fait que le niveau de la biomasse reste stable et le plus élevé pendant une certaine durée.
4. Hormis la phase la plus longue, les autres phases en chiffres ou en lettre semblent être alternées, périodiques et possédant le **même pattern ainsi que la même amplitude temporelle et quantitative**.

**Sur le plan économique**, autrement dit maximiser le profit actualisé, la tendance est alignée évidemment sur la quantité de la biomasse récoltée étant donné que c’est cette dernière qui génère le profit net en prenant compte le coût d’exploitation et le prix de vente le tout actualisé par le facteur ***interest\_rate***:

Ci-après une simulation d’une planification ayant les mêmes paramètres sur une **période de 50 ans** afin de confirmer le constat global :

Une image contenant texte, diagramme, Tracé, ligne

Le contenu généré par l’IA peut être incorrect.

## L’effet du taux de dissipation de la population (biomasse)

### Cas : équation de Volterra

Si le taux de croissance (***r\_growth***) est > 1, à partir d’un certain seuil de taux de dissipation bien que **l’extraction de la biomasse continue à croître, mais très faiblement**, la biomasse disponible reste constante à celle de départ. En voici une telle situation basée sur les mêmes paramètres que précédemment, mais avec un taux de dissipation (***dissipation\_rate***) de 0,3 au lieu de 0,2. Le faible taux de la biomasse extraite se reflète directement sur le niveau de profit réalisé très faible. Par conséquent, cette situation ne peut être rentable sur toute la période de planification !

Une image contenant texte, diagramme, ligne, Tracé

Le contenu généré par l’IA peut être incorrect.

Sans surprise aucune, le moyen direct pour éviter cette situation consiste à assurer un taux de croissance plus élevé, ce qui absorbera la dissipation, ou peut-être mieux, trouver un autre moyen qui atténue tout simplement la dissipation en traitant ses causes racines. Prenons à titre d’exemple, le cas où le taux de croissance est multiplié par 5 (***r\_growth* = 5**) ! La répercussion est immédiate comme illustré par les graphes suivants avec un effort constamment égal au MAX (1).

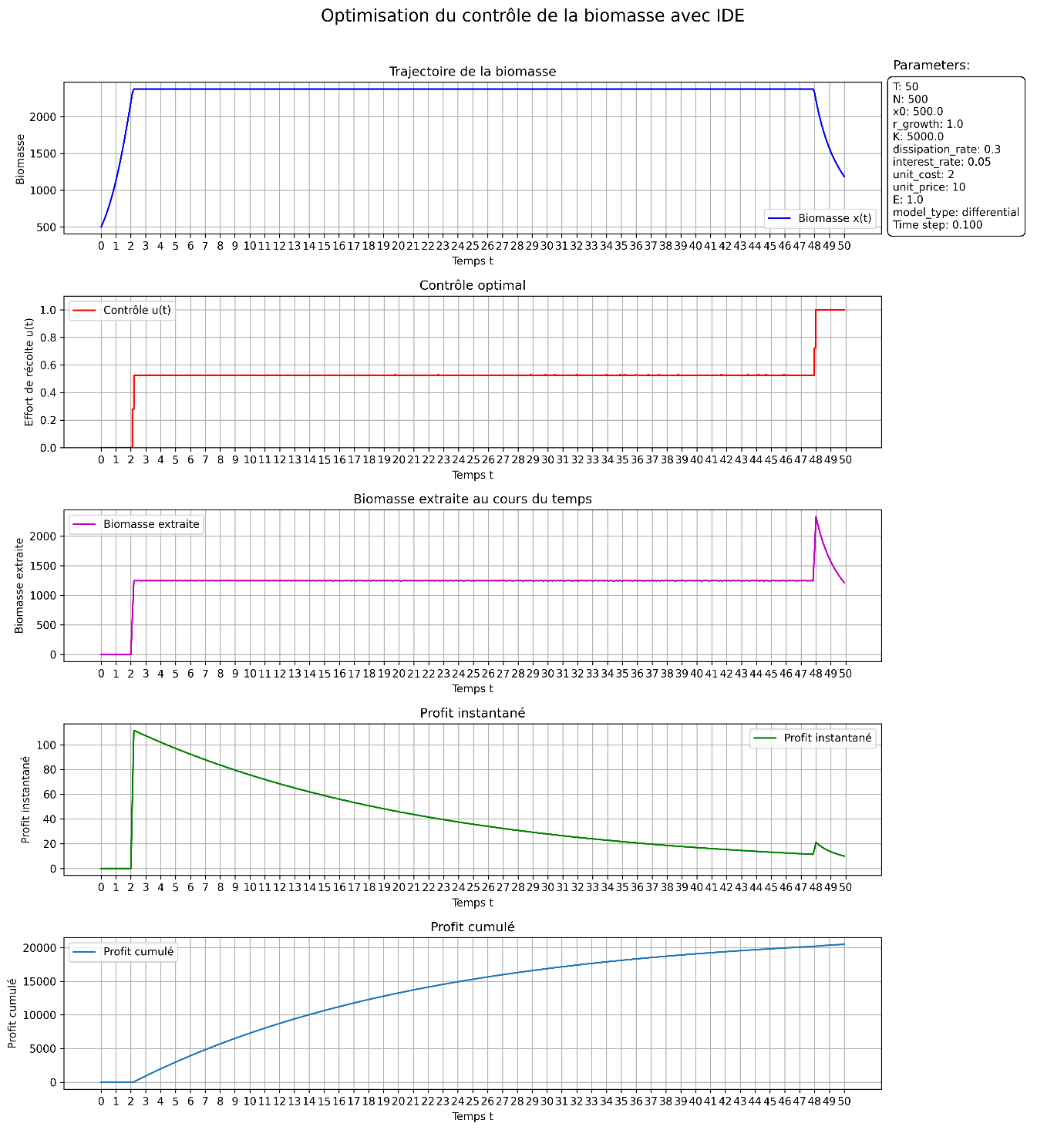
Une image contenant texte, diagramme, ligne, Tracé

Le contenu généré par l’IA peut être incorrect.

On observe bien que la quantité d’extraction de la biomasse est égale à la biomasse disponible (même courbe) et elle tourne autour de 3 000; cela s’explique par le fait que l’effort déployé est constamment égal à l’effort maximal fixé à 1 (E=1). L’effet négatif, voire désastreux d’une telle optimisation est que des fois on atteint des niveaux trop élevés (i**ci on** **atteint un peu plus de 60% de la capacité de charge maximale limitée à 5 000)**. Dans de telle situation, on doit rajouter une contrainte plus restrictive sur la taille de la biomasse comparée à la capacité de charge comme la contraindre à ne pas dépasser de la moitié : **x<=.5K (K=5 000 dans notre exemple)**, ce qui semble être le cas dans les autres simulations avec la simple contrainte x<=K; on ne dépasse pas de la moitié la capacité de charge pour le bien-être de la biomasse 😊. Malheureusement, la réalité de nos industries d’élevage pour en citer un exemple parmi d’autres, dépasse largement cette limitation.

### Cas : équation différentielle

Contrairement au cas d’un système gouverné par une équation de Volterra, le cas où il est piloté par une équation différentielle subit une diminution du niveau d’extraction de la biomasse beaucoup moindre comme cela est illustré par les graphes ci-dessous en prenant les mêmes paramètres que dans le cas Volterra. On constate que **le niveau d’extraction reste autour de 50% de la biomasse disponible** à partir de la période 2.



# Conclusion

L’intégration entre la base théorique détaillée dans l’article PDF et la flexibilité du code Python permet d’aborder la question de la dissipation des ressources de manière holistique. Cette approche facilite non seulement la compréhension fine des phénomènes observés, mais aussi l’exploration de solutions innovantes pour limiter les pertes et améliorer l’efficacité globale des systèmes étudiés. Ce type de projet montre combien la synergie entre mathématiques appliquées et programmation ouvre de nouvelles perspectives en ingénierie comme en recherche fondamentale.