

Modèle de dissipation des ressources : de la théorie à la pratique avec Python

Par : Abdel YEZZA, Ph.D

Date : juil. 2025

Sommaire

Introduction	2
Le code	3
Biomass Optimization Model.....	3
Description	3
🎯 Key Features	3
🔧 Installation.....	3
📦 Configuration	4
Model parameters and usual values.....	4
Symbol Meaning Typical / Common Values.....	4
🚀 Usage	5
📋 Output	5
Visualizations	5
Files Generated	5
Examples of output files	5
Mathematical Model	6
Objective Function	6
Constraints	6
Dynamics.....	6
📁 Project Structure	6
Key Files Description	7
📜 License	7
🤝 Contributing.....	7
Analyse des résultats des simulations obtenus	8
Analyse des résultats de simulation obtenus.....	8
Cas : équation différentielle	8
Cas : équation de Volterra	10
L'effet du taux de dissipation de la population (biomasse)	13

Cas : équation de Volterra	13
Cas : équation de Volterra	15

Introduction

Les problématiques liées à la dissipation des ressources, qu'elles soient énergétiques, matérielles ou vivantes, sont au cœur de la réflexion scientifique depuis des siècles si on s'appuie sur les écrits, voire plus. En 1990, dans le cadre de ma thèse de doctorat, j'ai eu l'opportunité de travailler sous la direction de [Francis Clarke](#) (pionnier de l'analyse proximale ou non lisse) sur une application des résultats théoriques obtenus au modèle d'optimisation des ressources renouvelables. Ce modèle était déjà connu dans le cas des équations différentielles, mais moins exploré pour les équations de Volterra. J'ai toujours souhaité reprendre le travail théorique réalisé et le compléter par des simulations numériques renforçant les conclusions obtenues. Le développement de nouvelles compétences en programmation et les avancées informatiques récentes offrent aujourd'hui la possibilité de reprendre ces travaux avec une approche numérique, c'est cette compétence qui me manquait dans les années 80/90 !

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 biomasses afin d'optimiser leur extraction et surtout éviter leur extinction. Cet article présente une exploration du code Python accessible au grand public via GITHUB : [ayezza/modele_dissipation_ressources](#) en s'appuyant sur les principes exposés dans l'article scientifique joint au projet et extrait de la thèse de doctorat.

Dans cet article, je laisse du côté la théorie aux spécialistes et je mets l'accent sur les aspects pratiques et concrets illustrés par du code PYTHON !

Les deux modèles seront étudiés du point de vue numérique et programmatique :

1. **Le modèle gouverné par des équations différentielles**, il s'agit du modèle le plus connu et répandu
2. **Le modèle gouverné par des équations de Volterra** qui prend en compte le passé et la mémoire de la biomasse

Le code

Biomass Optimization Model

This project implements a numerical optimization model for biomass management, considering both Volterra integral equations (with memory effects) and differential equations approaches.

The Volterra integral equation case is formulated based on my doctorate thesis in 1991, fourth application (pp. 159-169): [Thèse de doctorat Yezza Abdel](#).

Description

The model optimizes the harvesting effort to maximize profit while maintaining sustainable biomass levels. It supports two different modeling approaches:

- Volterra integral equation (with memory/dissipation effects)
- Standard differential equation (logistic growth with harvesting)

Key Features

- Configurable parameters via JSON file
- Biomass dynamics simulation using either Volterra or differential equations
- Profit optimization with harvesting constraints
- Detailed visualization of results (5 different plots)
- Export of results to CSV and Excel files

Installation

1. Clone this repository
2. Install required dependencies:

```
pip install numpy pandas scipy matplotlib openpyxl uuid json
```

Configuration

Create a **params.json** file with the following parameters that can be modified from which the main program will extract the parameters:

```
{
    "T": 20,                                // Time horizon generally in years
    "N": 240,                               // Number of time steps for simulation
    "x0": 500.0,                            // Initial biomass
    "r_growth": 5.0,                         // Growth rate
    "K": 5000.0,                            // Carrying capacity
    "dissipation_rate": 0.05,                // Memory effect parameter (applies to Volterra model only)
    "interest_rate": 0.05,                  // Economic discount rate
    "unit_cost": 2.0,                        // Cost per unit effort
    "unit_price": 10.0,                      // Price per biomass unit
    "E": 1.0,                               // Maximum harvesting effort
    "model_type": "volterra"                // Model type: "volterra" or "differential"
}
```

Model parameters and usual values

All parameters are fixed in the JSON file: **params.json** which should be in the same folder as the main program **main.py**

Symbol Meaning Typical / Common Values

Symbol	Signification	Valeurs typiques / Usuelles
r_growth	Biomass intrinsic growth rate	0.2 to 1.0 (often 0.5 to 0.8)
K	Ecosystem load capacity	500 to 20 000 We set it at 5 000
x0	Initial Biomasse value	Typically 0.2K to 0.8K We set it at 500
dissipation_rate	Cumulative dissipation coefficient	0.005 to 0.05 (often 0.01)
T	Time horizon (years)	5 to 50 (often 10)
unit_cost	Unit price of harvested biomass	We set it at 2
unit_price	Unit cost of harvesting effort	We set it at 10
interest_rate	Discount rate	0.02 to 0.1 (often 0.05)
E	Maximum permissible effort	0.1 to 1 (extreme case)

Usage

Run the model:

```
python main.py
```

The program will:

1. Load parameters from **params.json**
2. Optimize the harvesting strategy depending of the model type (**Volterra** or **Differential**)
3. Generate visualizations and save them as png images in the **output** directory
4. Save results to CSV and Excel files in the **output** directory

Output

Visualizations

These graphs are generated in the same figure:

1. Biomass trajectory
2. Control effort
3. Extracted biomass
4. Instantaneous profit
5. Cumulative profit

Files Generated

All output files are prefixed by a unique GUID:

- **biomass_optimization_results_GUID.png**: Combined visualization of all results
- **biomass_optimization_timeseries_GUID.csv**: Time series data
- **detailed_results_GUID.xlsx**: Detailed results with multiple sheets including:
 - **Detailed_Results**: Time series of all variables
 - **Parameters**: Model configuration

Examples of output files

1. Graphs (Volterra Model or Differential Model depending on parameter **model_type** value)
2. CSV
3. Excel

Mathematical Model

Objective Function

Maximize the discounted profit:

$$J(x, u) = \int_0^T e^{\delta t} (c - px(t)) u(t) dt$$

c = Cost

p = Price

Constraints

- $0 \leq u(t) \leq E$ (Harvesting effort bounds)
- $x(t) \geq 0$ (Non-negative biomass)

Dynamics

Volterra Model:

$$x(t) = x_0 + \int_0^t \left[rx(s) \left(1 - \frac{x(s)}{K} \right) e^{-\rho(t-s)} - u(s)x(s) \right] ds$$

Differential Model:

$$\frac{dx}{dt} = rx(t) \left(1 - \frac{x(t)}{K} \right) - u(t)x(t)$$

Project Structure

```
modele_dissipation_ressources/
├── main.py          # Main application file with BiomassModel class
├── params.json      # Configuration parameters
├── README.md        # Project documentation
├── requirements.txt # Python dependencies
└── output/          # Generated output directory
    ├── biomass_optimization_results_GUID.png  # Visualization plots
    ├── biomass_optimization_timeseries_GUID.csv # Time series data
    └── detailed_results_GUID.xlsx            # Detailed analysis results
```

Key Files Description

- **main.py:** Contains the core BiomassModel class implementing:
 - Biomass dynamics simulation (Volterra/Differential)
 - Optimization algorithms
 - Visualization methods
 - Results generation
- **params.json:** Configuration file containing:
 - Model parameters
 - Simulation settings
 - Economic parameters
- **output/:** Directory containing generated results:
 - PNG files for visualizations
 - CSV files for time series data
 - Excel files with detailed analysis

License

This project is open source and available under the MIT License.

Contributing

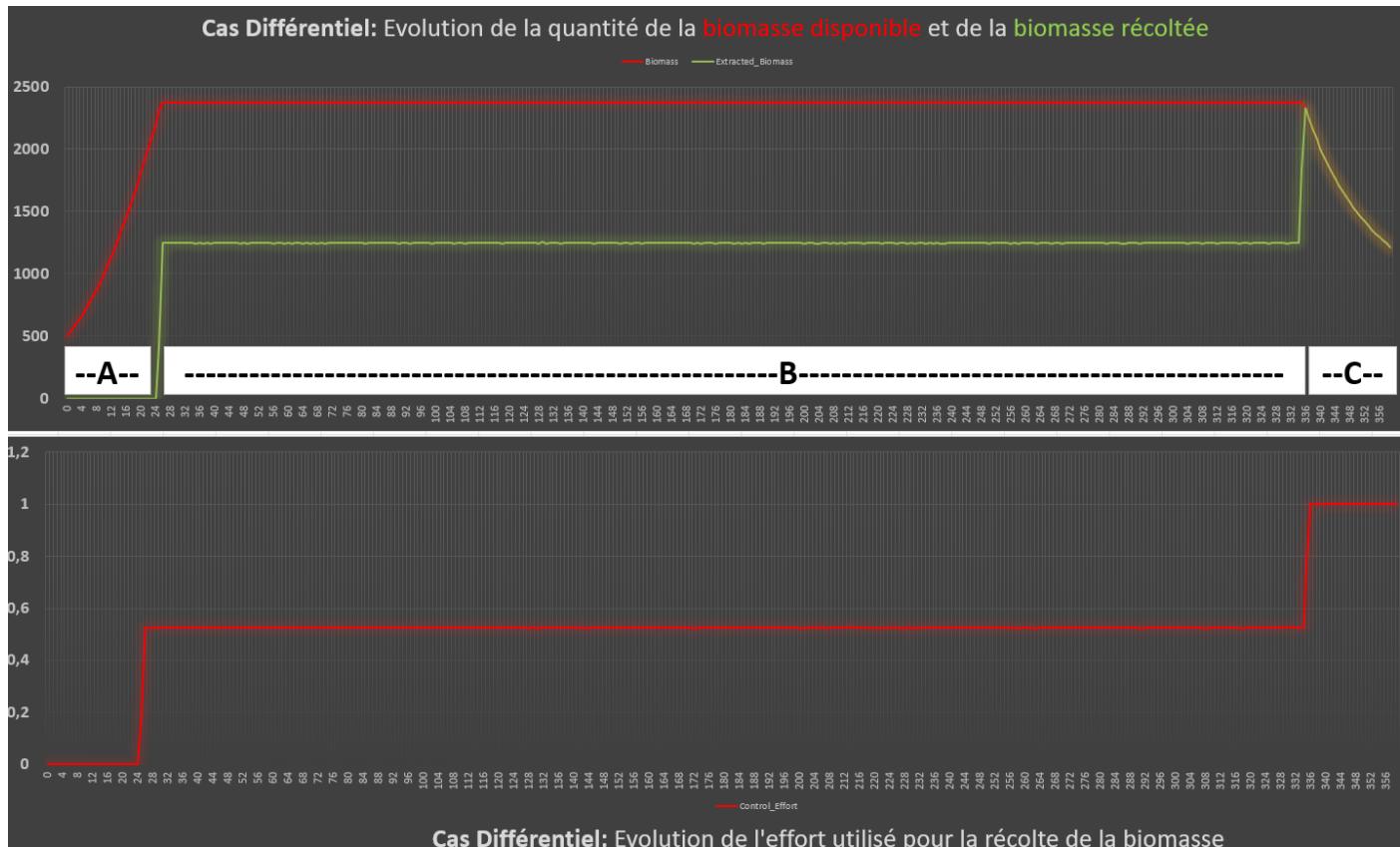
Contributions are welcome! Please feel free to submit a Pull Request.

Analyse des résultats des simulations obtenus

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

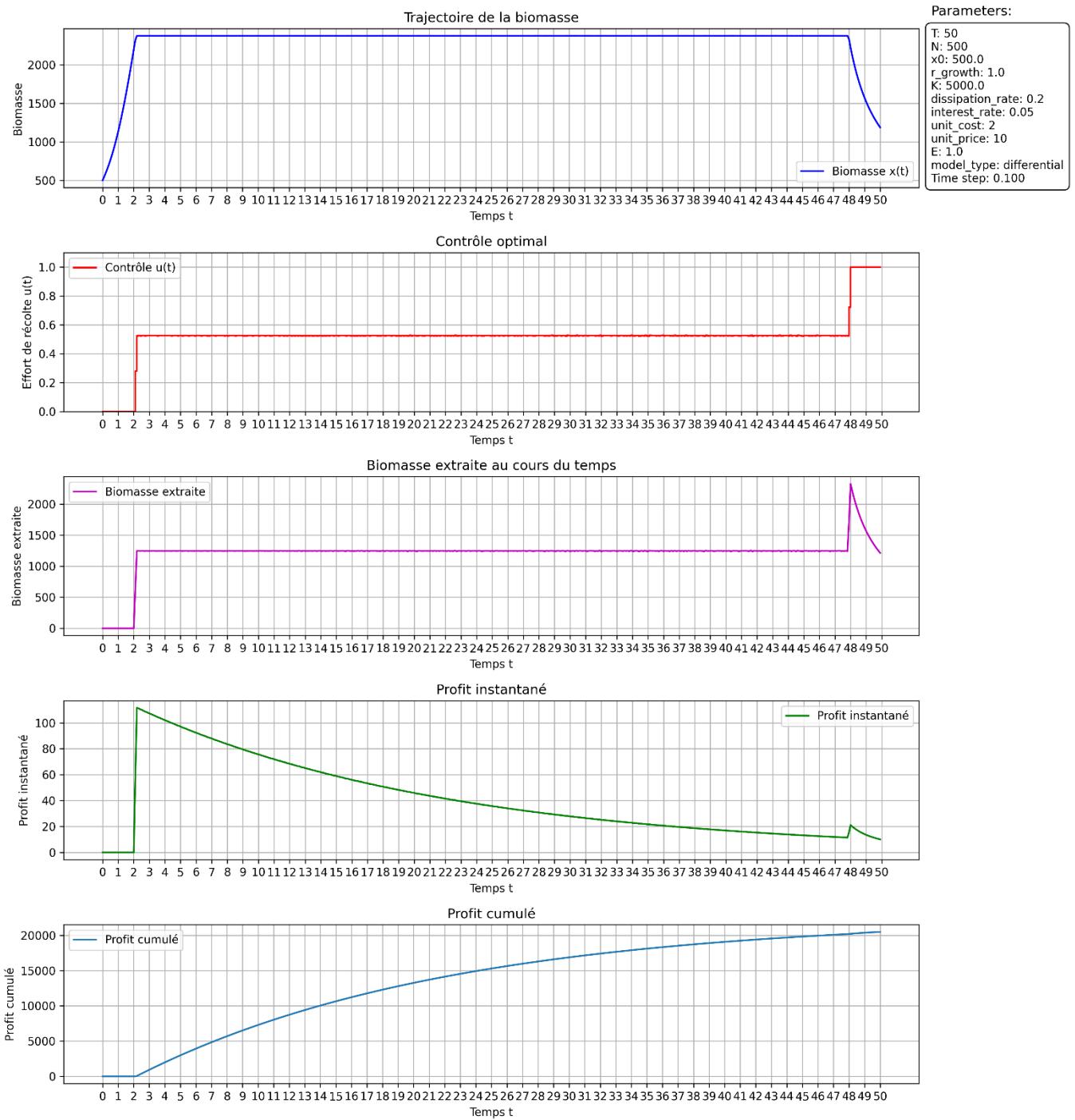


Trois périodes consécutives se distinguent sur la durée de planification :

- 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 1^{ère} phase de constitution de la biomasse (période d'investissement).
- La phase **A** est suivie immédiatement de la période **B** assez longue et constante sur tous les plans :
 - 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 ($\frac{1}{4}$) de la capacité de charge
 - Une extraction constante et régulière valant presque 50% de la biomasse disponible
- La phase finale **C** symétrique à la phase **A** mais tout en gardant la biomasse égale à au moins le $\frac{1}{4}$ de la capacité de charge jusqu'à la fin de période de planification.
- 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 :
 - Il est **nul** pendant la phase **A**
 - Il est égal approximativement à **0,52** pendant toute la phase **B**
 - Il est au maximum (=1) en phase finale **C**

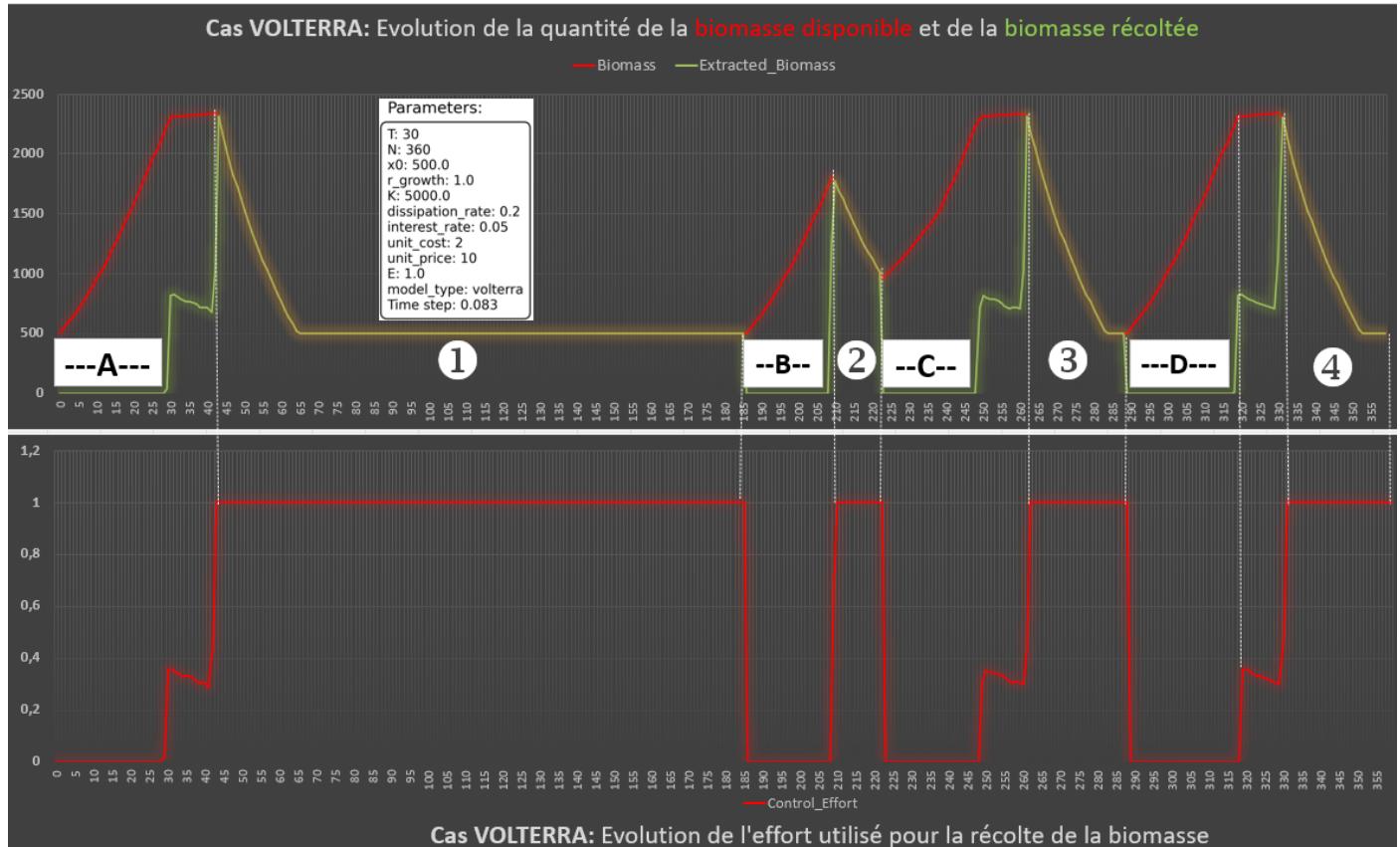
Ci-après une simulation d'une planification ayant les mêmes paramètres sur une **période de 50 ans** afin de confirmer le constat global ci-dessus.

Optimisation du contrôle de la biomasse avec IDE



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.



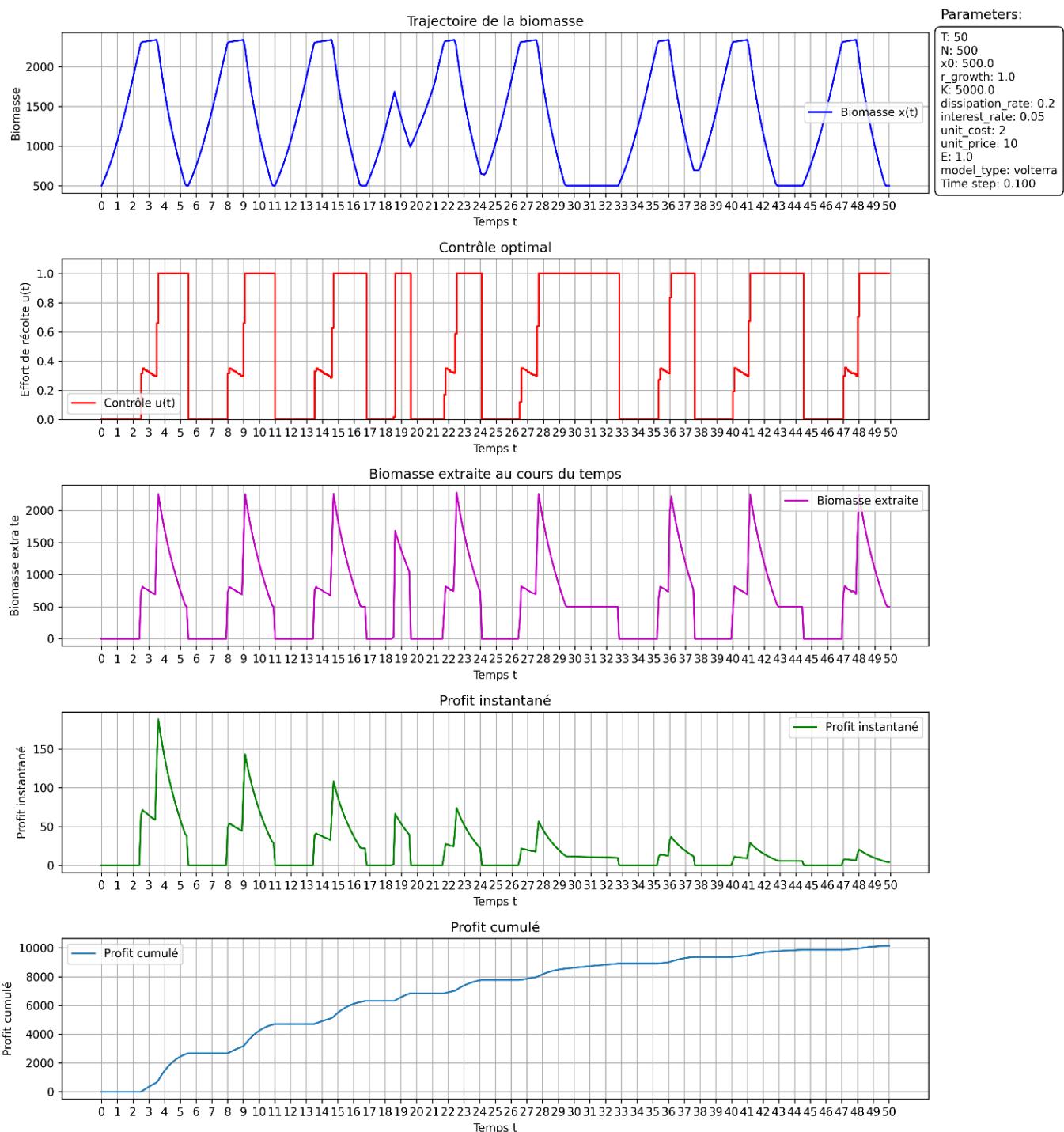
1. Pendant les phases numérotées ① ② ③ ④ illustrées sur les graphes ci-dessous :
 - a. La biomasse disponible est identique à celle récoltée
 - b. Cela est justifié par le fait que l'effort de récolte est au maximum, i.e., = E = 1.0
 - c. Cela n'a pas d'incidence sur la disponibilité de la biomasse sur toute la période de planification de 30 ans
 - d. Hormis la phase ①, 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 :
 - a. **Période A :** pendant les 28 premiers mois de planification le temps que la biomasse puisse croître suffisamment pour être récoltée
 - b. **Période B :** située entre les phases ①, ② qui dure de la période 184 à la période 208 pour permettre l'accroissement de la biomasse disponible
 - c. **Périodes C et D :** situées respectivement entre les phases ②, ③ et ③, ④ 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 **1 2 3 4** 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 :

Optimisation du contrôle de la biomasse avec IDE

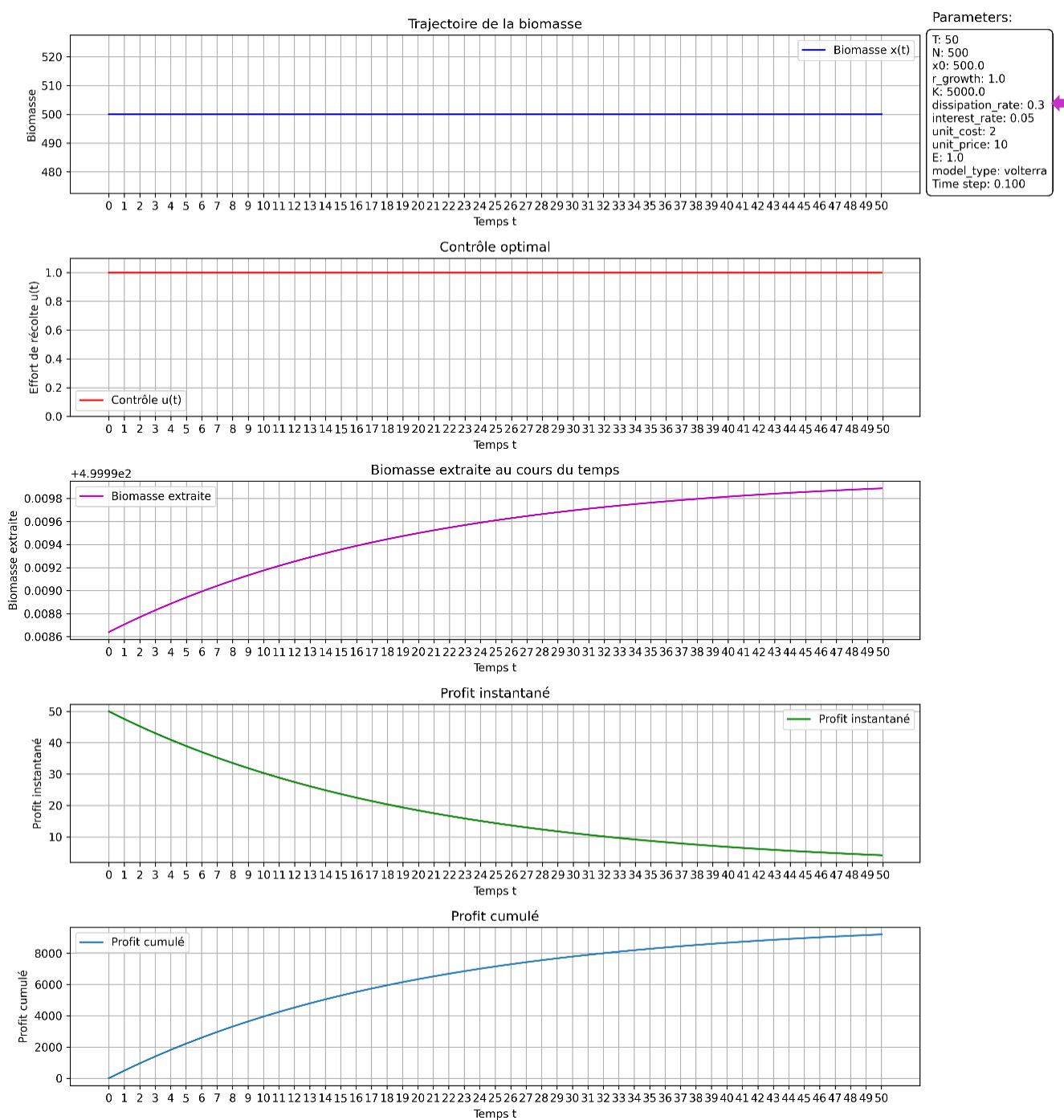


L'effet du taux de dissipation de la population (biomasse)

Cas : équation de Volterra

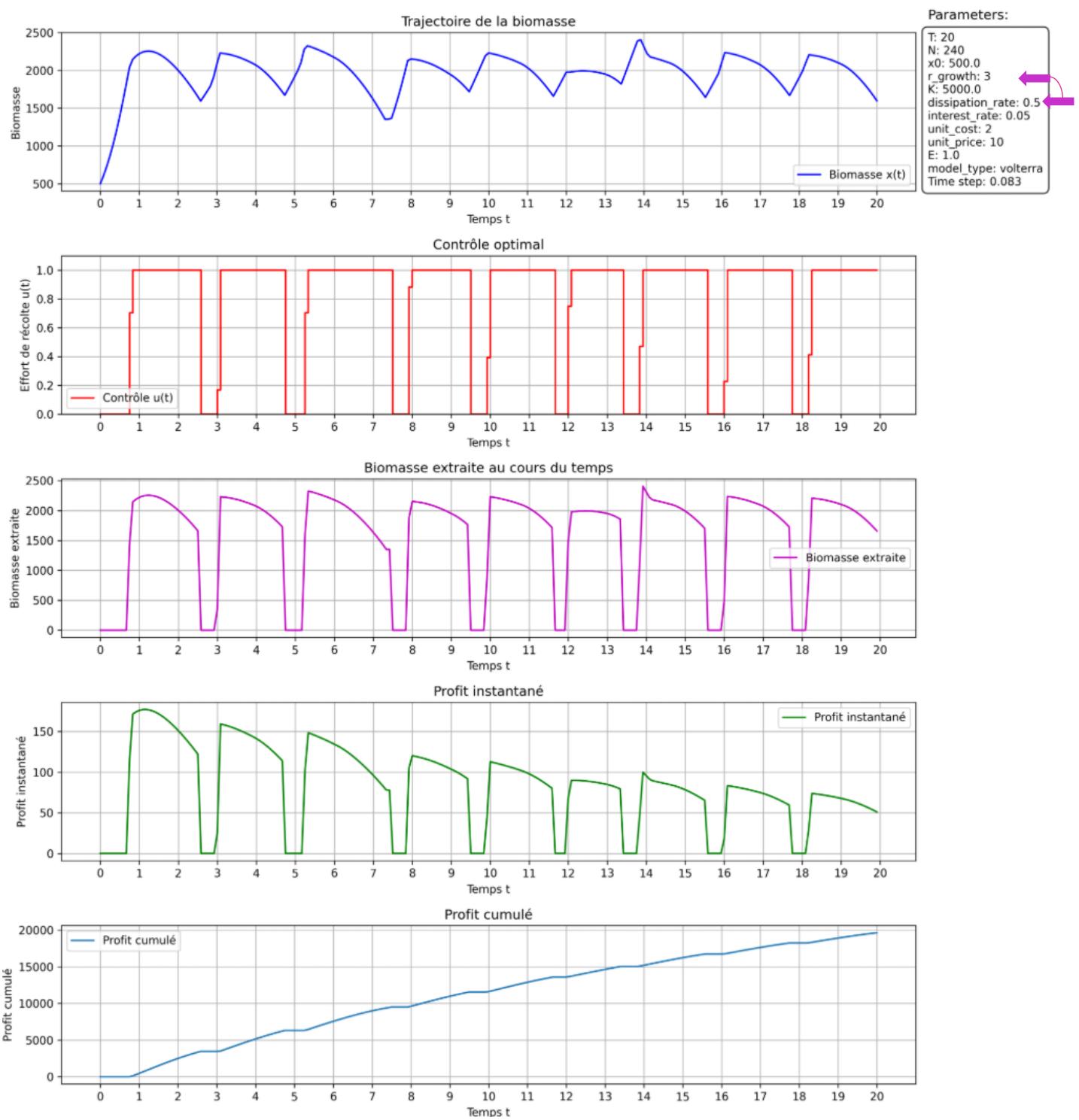
Si le taux de croissance (*r_growth*) est > 1 , à partir d'un certain seuil de taux de dissipation bien que l'**extraction de la biomasse continue à croître, mais très faiblement**, la biomasse disponible reste constante à celle de départ. En voici une telle situation basée sur les mêmes paramètres que 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 !

Optimisation du contrôle de la biomasse avec IDE



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 3 (**r_growth = 3**) ! La répercussion est immédiate comme illustré par les graphes suivants avec un effort presque périodiquement oscillant entre 0 et 1, mais plus souvent fixé à 1 !

Optimisation du contrôle de la biomasse avec IDE



On observe bien que la quantité d'extraction de la biomasse est presque égale à la biomasse disponible (même pattern avec quelques périodes de repos) avec une moyenne de 1 959 individus de la biomasse et de 1943

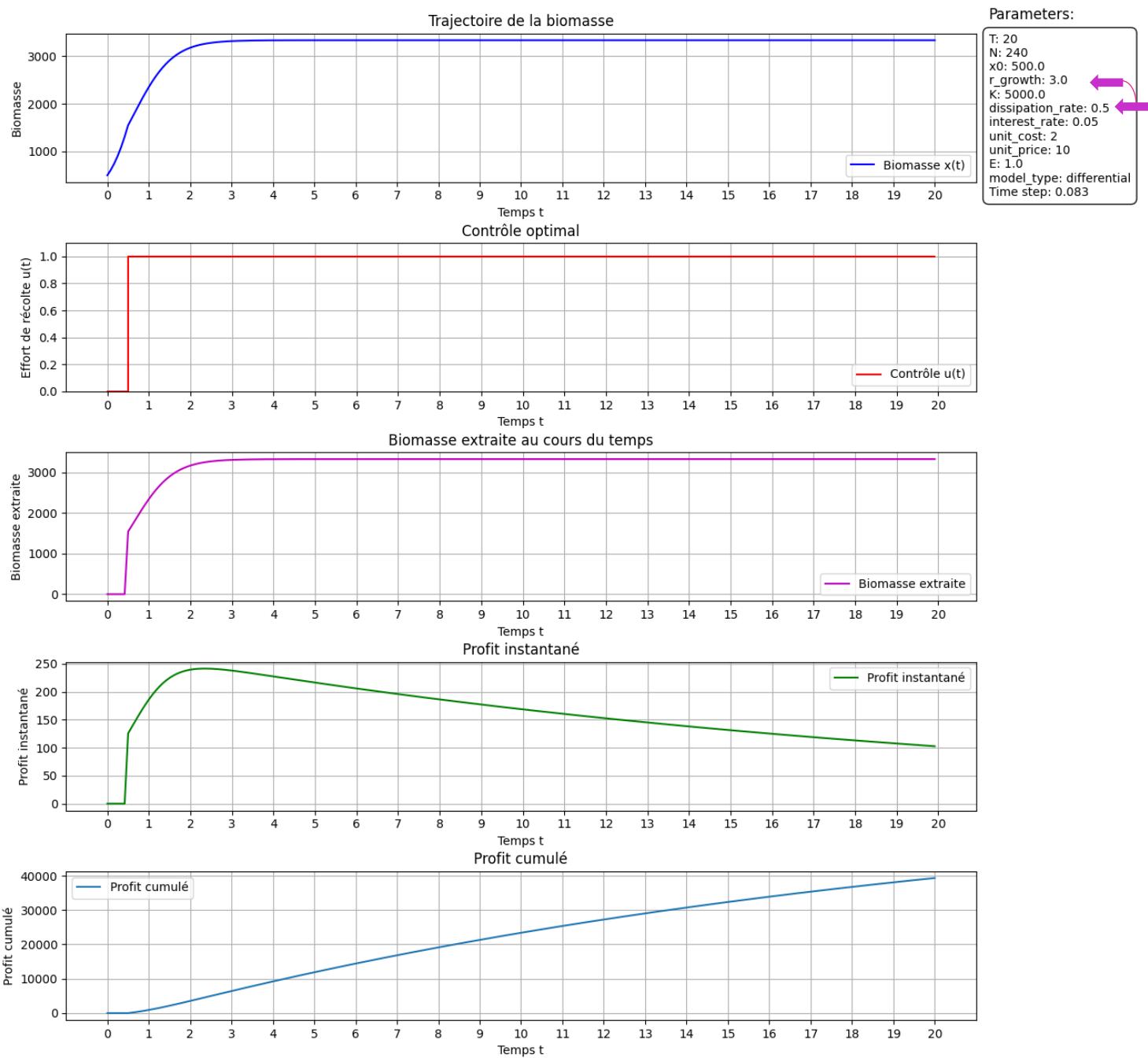
individus de la biomasse extraite hors périodes nulles; et un maximum de 2406. Cela s'explique par le fait que l'**effort déployé est constamment égal à l'effort maximal fixé à 1 ($E=1$)**. L'effet négatif, voire désastreux d'une telle optimisation est que des fois on atteint des niveaux trop élevés (**ici on atteint un peu plus de 48% de la capacité de charge maximale limitée à 5 000, ce qui est acceptable**). 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 : **$x \leq .5K$ ($K=5\,000$ dans notre exemple)**, ce qui semble être le cas dans les autres simulations avec la simple contrainte $x \leq K$; 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 qu'un exemple parmi d'autres, dépasse largement cette limitation.

Cas : équation de Volterra

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 moins comme cela est illustré par les graphes ci-dessous en prenant les mêmes paramètres que dans le cas Volterra. On constate que globalement en moyenne aussi bien **le niveau de la biomasse disponible que celui de l'extraction sont autour de 64% de la capacité de charge**. En outre, à partir de la période 6 (mois), les deux évoluent d'une manière égale et convergent vers une valeur commune constante de 3 333 à partir du 59^{ème} mois (~5^{ème} année) comme la synthèse statistique fournie par PYTHON le confirme ci-dessous ainsi que le graphe qui suit :

Summary of results:													
	T	N	x0	r_growth	K	dissipation_rate	interest_rate	unit_cost	...	period	Time	Biomass	Extracted_Biomass
count	240.0	240.0	240.0	240.0	240.0	240.0	2.400000e+02	240.0	...	240.00000	240.000000	240.000000	240.000000
mean	20.0	240.0	500.0	3.0	5000.0	0.5	5.000000e-02	2.0	...	119.50000	9.958333	3205.763698	3184.261545
std	0.0	0.0	0.0	0.0	0.0	0.0	6.953395e-18	0.0	...	69.42622	5.785518	457.774038	571.882604
min	20.0	240.0	500.0	3.0	5000.0	0.5	5.000000e-02	2.0	...	0.00000	0.000000	500.00000	0.000000
25%	20.0	240.0	500.0	3.0	5000.0	0.5	5.000000e-02	2.0	...	59.75000	4.979167	3333.084254	3333.084254
50%	20.0	240.0	500.0	3.0	5000.0	0.5	5.000000e-02	2.0	...	119.50000	9.958333	3333.333329	3333.333329
75%	20.0	240.0	500.0	3.0	5000.0	0.5	5.000000e-02	2.0	...	179.25000	14.937500	3333.333333	3333.333333
max	20.0	240.0	500.0	3.0	5000.0	0.5	5.000000e-02	2.0	...	239.00000	19.916667	3333.333333	3333.333333

Optimisation du contrôle de la biomasse avec IDE



Je vous laisse conclure quant à la comparaison entre les deux modèles et examiner dans quelle mesure nous sommes en capacité d'appliquer l'un ou l'autre dans nos activités et interactions avec la nature, les environnements et toutes les biomasses que notre terre nous offre depuis déjà des millions d'années 😊