

# Modélisation Épidémiologique de la Fuite d'Information

Application aux stratégies de contre-ingérence

Arnaud Stienon

UCLouvain - EPL

28 février 2026

# Table des matières

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Modélisation Mathématique</b>	<b>1</b>
2.1	Présentation du modèle SIR classique . . . . .	1
2.2	Définition des variables et hypothèses . . . . .	2
2.3	Le Système Dynamique . . . . .	2
<b>3</b>	<b>Analyse Théorique</b>	<b>3</b>
3.1	Calcul du taux de reproduction de base ( $R_0$ ) . . . . .	3
3.2	Points d'équilibre . . . . .	3
3.3	Analyse de stabilité locale . . . . .	3
<b>4</b>	<b>Simulations Numériques</b>	<b>4</b>
4.1	Discussion des paramètres . . . . .	4
4.2	Scénario de base . . . . .	5
4.3	Scénario d'intervention (Contre-mesures) . . . . .	7
4.3.1	Impact sur les coefficients . . . . .	7
4.3.2	Fonction budget et optimisation . . . . .	8
<b>5</b>	<b>Conclusion</b>	<b>10</b>
5.1	Rappel . . . . .	10
5.2	Apport du projet . . . . .	10
5.3	Limites et critiques du modèle . . . . .	10
5.4	Améliorations . . . . .	11
5.4.1	Intégration des 'bots' . . . . .	11
5.4.2	Modélisation en graphe . . . . .	11
5.4.3	Notes de la communauté . . . . .	11
<b>A</b>	<b>Code python résolution système</b>	<b>11</b>
<b>B</b>	<b>Code python optimisation budget</b>	<b>12</b>

# 1 Introduction

Les fausses informations ("fake news") se propagent aujourd'hui de manière massive sur les réseaux sociaux. Leur vitesse de diffusion pose un problème majeur : une rumeur virale circule souvent bien plus vite que la vérité, ce qui peut rapidement déstabiliser l'opinion publique ou menacer la sécurité d'un État. Pour contrer ce phénomène, les autorités et les plateformes disposent de deux leviers principaux : la suppression algorithmique des comptes problématiques (censure) ou la publication massive de démentis (fact-checking). Toute la difficulté est de savoir quand et comment utiliser ces solutions pour étouffer la rumeur le plus efficacement possible. C'est là qu'interviennent les mathématiques. Puisqu'une information virale se transmet d'individu en individu de la même manière qu'une maladie contagieuse, ce projet propose d'utiliser la modélisation épidémiologique. En adaptant le modèle classique SIR (Susceptibles, Infectés, Retirés) aux réseaux sociaux, l'objectif de ce rapport est de simuler la propagation d'une rumeur via un système d'équations différentielles, afin d'évaluer quantitativement l'efficacité des différentes contre-mesures. Ce projet vise à analyser de manière purement mathématique la transmission de fausses informations et d'autres fuites de données. Son objectif n'est en aucun cas de juger moralement quelles contre-mesures utiliser, ni d'aider à faire disparaître des contenus jugés légitimes d'être partagés.

## 2 Modélisation Mathématique

### 2.1 Présentation du modèle SIR classique

Pour modéliser la diffusion d'une fausse information, nous nous appuyons sur le modèle compartimental SIR, développé en 1927 par Kermack et McKendrick. Ce modèle divise une population totale constante en trois compartiments. Pour simplifier l'analyse, nous manipulons ces groupes comme des proportions de la population totale, de sorte qu'à tout instant  $t$ ,  $S(t) + I(t) + R(t) = 1$ , où  $t$  est en jours :

- $S(t)$  (**Susceptibles**) : Individus sains n'ayant pas encore contracté la maladie, mais pouvant être infectés.
- $I(t)$  (**Infectés**) : Individus porteurs de la maladie, capables de la transmettre.
- $R(t)$  (**Rétablis ou Retirés**) : Individus guéris et définitivement immunisés.

La dynamique de passage d'un compartiment à l'autre est régie par le système d'équations différentielles ordinaires (EDO) non linéaires suivant :

$$\frac{dS}{dt} = -\beta SI \tag{1}$$

$$\frac{dI}{dt} = \beta SI - \gamma I \tag{2}$$

$$\frac{dR}{dt} = \gamma I \tag{3}$$

Ce système repose sur deux paramètres fondamentaux qui déterminent la vitesse de l'épidémie :

- $\beta$  (**Le taux de transmission**) : Il définit le nombre de contacts infectieux par unité de temps. Le terme  $\beta SI$  modélise la loi d'action de masse (les nouvelles infections sont proportionnelles aux rencontres entre Susceptibles et Infectés).
- $\gamma$  (**Le taux de guérison**) : Il correspond à la proportion d'Infectés qui guérissent par unité de temps. Son inverse ( $1/\gamma$ ) représente la durée moyenne de la période infectieuse.

C'est l'évolution de la fonction  $I(t)$  au sein de ce système qui permet de prédire si une épidémie va s'éteindre rapidement ou générer une vague d'infections massive.

## 2.2 Définition des variables et hypothèses

Pour adapter le modèle à notre situation, on va séparer la population en 4 groupes distincts :

- $S(t)$  : Les Ignorants (population ciblée n'ayant pas encore vu la fuite).
- $I(t)$  : Les Propagateurs (individus partageant activement l'information).
- $R(t)$  : Les Muets (individus croyant l'information mais ne la propageant pas).
- $V(t)$  : Les Savants (individus connaissant l'information mais n'y croyant pas, ou sachant qu'elle est fausse).

Supposons comme pour le modèle SIR que la population est constante, i.e.  $S(t) + I(t) + R(t) + V(t) = N$ .  $\beta$  représente toujours le taux de transmission en personne par unité de temps.  $\alpha$  sera le taux de personnes atteintes par l'info qui deviennent propagateurs,  $1 - \alpha$  sera donc le taux de personnes atteintes par l'info ne devenant pas propagateurs. Ce paramètre est nécessaire de par l'ajout d'un 4ème état. En effet, ce paramètre n'est pas dans le modèle SIR, car en épidémiologie on ne passe pas de sain à rétabli sans passer par la catégorie propagateur. Enfin, définissons  $\gamma_1$ , le taux de personnes passant de muet à savant de par la publication d'un démenti ou autre, et  $\gamma_2$  le taux de propagateurs devenant muets.

## 2.3 Le Système Dynamique

Le transfert de population entre ces compartiments est régi par le système d'équations différentielles ordinaires suivant :

$$\frac{dS}{dt} = -\beta SI \quad (4)$$

$$\frac{dI}{dt} = \alpha\beta SI - \gamma_2 I \quad (5)$$

$$\frac{dR}{dt} = \beta SI(1 - \alpha) - \gamma_1 R + \gamma_2 I \quad (6)$$

$$\frac{dV}{dt} = \gamma_1 R \quad (7)$$

Le plus compliqué reste à faire : trouver les coefficients. On pourrait imaginer de résoudre ce système une fois par réseau, et selon l'utilisateur type, les coefficients peuvent varier grandement. On pourrait même imaginer de résoudre ce système pour chaque bulle algorithmique pour gagner en précision. Il ne nous resterait ensuite qu'à sommer chaque catégorie de personnes pour trouver les 4 variables en fonction du temps.

### 3 Analyse Théorique

#### 3.1 Calcul du taux de reproduction de base ( $R_0$ )

Le taux  $R_0$  est un facteur qui indique si une épidémie commence, en  $t = 0$ . Si  $R_0 > 1$ , l'épidémie commence et s'étend, si  $R_0 = 1$ , l'épidémie stagne, et si  $R_0 < 1$ , l'épidémie s'épuise. Au début,  $S \approx 1$  car on normalise la population à 1, et au début tout le monde est Ignorant. Pour que l'épidémie s'étende, il faut que  $\frac{dI}{dt} > 0$  :

$$\frac{dI}{dt} = \beta SI\alpha - \gamma_2 I \approx \beta I\alpha - \gamma_2 I > 0 \implies R_0 := \frac{\beta\alpha}{\gamma_2}.$$

#### 3.2 Points d'équilibre

Commençons par calculer l'état d'équilibre asymptotique du système, i.e. :

$$\frac{dS}{dt} = \frac{dI}{dt} = \frac{dR}{dt} = \frac{dV}{dt} = 0.$$

L'épidémie sera finie quand il n'y aura plus de propagateurs  $I^* = 0$ . Cela nous donne après bref calcul que :

$$\frac{dR}{dt} = -\gamma_1 R = 0 \quad \text{et} \quad \frac{dS}{dt} = \frac{dV}{dt} = 0.$$

L'état final est donc  $(S_\infty, 0, 0, V_\infty)$ , où  $S_\infty + V_\infty = N$ . On a donc une part de la population qui ne sera jamais en contact avec l'information. Calculons maintenant la valeur maximum de  $I$ . Le plus facile est de trouver  $\frac{dI}{dS}$  pour intégrer des deux côtés de l'égalité.

$$\frac{dI}{dS} = \frac{\frac{dI}{dt}}{\frac{dS}{dt}} = \frac{\beta SI\alpha - \gamma_2 I}{-\beta SI} = \frac{\gamma_2 - \beta S\alpha}{\beta S}$$

En intégrant des 2 côtés, on trouve  $I(S) = \frac{\gamma_2}{\beta} \ln(S) - \alpha S + C$ .  $I_{max}$  se trouve quand  $\frac{dI}{dS} = 0$ , ce qui donne  $S_{max} = \frac{\gamma_2}{\alpha\beta}$ . On trouve donc en remplaçant dans la formule de  $I(S)$  que

$$I_{max} = \frac{\gamma_2}{\beta} \ln\left(\frac{\gamma_2}{\alpha\beta}\right) - \frac{\gamma_2}{\beta} + C$$

Sachant que  $S(0) \approx 1$ , on a que  $C = \alpha + I_0$ .

#### 3.3 Analyse de stabilité locale

Pour analyser la stabilité des points d'équilibre, nous calculons la matrice Jacobienne du système quand l'épidémie est finie (donc  $I = 0$ ). Puisque la propagation de l'épidémie ne dépend que de  $I$  et  $S$ , nous pouvons simplifier l'analyse en étudiant le sous-système avec uniquement ces variables :

$$\frac{dS}{dt} = -\beta SI \tag{8}$$

$$\frac{dI}{dt} = \alpha\beta SI - \gamma_2 I \tag{9}$$

La matrice Jacobienne  $J(S, I)$  associée à ce sous-système est :

$$J(S, I) = \begin{pmatrix} -\beta I & -\beta S \\ \alpha\beta I & \alpha\beta S - \gamma_2 \end{pmatrix}$$

Évaluons cette matrice à l'équilibre final de l'épidémie, i.e.  $E^* = (S_\infty, 0)$  :

$$J(S_\infty, 0) = \begin{pmatrix} 0 & -\beta S_\infty \\ 0 & \alpha\beta S_\infty - \gamma_2 \end{pmatrix}$$

Les valeurs propres de cette matrice sont les solutions de  $\det(J - \lambda I) = 0$ , ce qui nous donne :

$$\lambda_1 = 0 \quad \text{et} \quad \lambda_2 = \alpha\beta S_\infty - \gamma_2$$

La valeur propre nulle  $\lambda_1$  correspond au fait qu'il existe une infinité d'équilibre (la droite où  $I = 0$ ). La stabilité du point d'équilibre dépend donc du signe de la seconde valeur propre  $\lambda_2$ . Le point d'équilibre est stable si et seulement si  $\lambda_2 < 0$ , c'est-à-dire :

$$\alpha\beta S_\infty - \gamma_2 < 0 \implies S_\infty < \frac{\gamma_2}{\alpha\beta}$$

Cela coïncide avec notre calcul du  $R_0$ . Au début de l'épidémie ( $S \approx 1$ ), pour que le système soit totalement stable (et donc que la fake news n'arrive pas à créer de pic), il faut que  $1 < \frac{\gamma_2}{\alpha\beta}$ , ce qui équivaut à  $R_0 < 1$ .

## 4 Simulations Numériques

### 4.1 Discussion des paramètres

Pour résoudre le système, nous devons d'abord estimer les coefficients  $\alpha, \beta, \gamma_1, \gamma_2$ . Nous allons calibrer ces paramètres pour le réseau X (ex-Twitter), un réseau propice à la transmission rapide de news. Commençons par  $\alpha$ , qui représente la probabilité qu'un individu voyant l'information décide de la relayer. C'est une mesure directe de la clivance de la nouvelle. La grande majorité des utilisateurs étant passifs, ce taux reste faible. Pour une rumeur banale,  $\alpha \approx 0.0005$ . Pour une information très polémique qui pousse à l'indignation, on atteindra une valeur plus haute, de l'ordre de  $\alpha \approx 0.003$  (soit 3 retweets pour 1000 vues, ce qui est déjà énorme à l'échelle du réseau). Le paramètre  $\beta$  représente la "contagion" de chaque propageur (le nombre de personnes qu'il contamine par unité de temps). Il ne faut pas se baser uniquement sur les influenceurs, mais sur la moyenne de tous les utilisateurs qui relaient. Pour une info peu virale, la portée est faible :  $\beta \approx 50$ . Si le post est très clivant, l'algorithme l'amplifie massivement, touchant de nombreux non-abonnés : nous prendrons  $\beta \approx 300$ . Prendre des valeurs supérieures conduirait à des taux de reproduction ( $R_0$ ) irréalistes, dépassant ceux des virus les plus contagieux au monde. Enfin,  $\gamma_1$  et  $\gamma_2$  représentent l'abandon de la dynamique de propagation ou de la croyance.  $\gamma_1$  s'applique aux Muets : il représente le taux naturel de prise de conscience, sans aucune intervention extérieure ou démenti officiel de l'État. En prenant du recul, en lisant les réponses contradictoires ou via les outils participatifs (comme les "Notes de la communauté" sur X), ces individus finissent par réaliser la fausseté de l'information et rejoignent les Savants. Pour une rumeur très virale, sa forte visibilité attire mécaniquement

de nombreuses vérifications spontanées de la part des internautes ( $\gamma_1 \approx 0.12$ ).  $\gamma_2$ , quant à lui, modélise la perte d'intérêt actif : c'est le taux de Propagateurs qui cessent de relayer l'information pour redevenir de simples Muets. Pour une information très polémique, cet essoufflement est lent car l'indignation entretient l'envie de partager ( $\gamma_2 \approx 0.06$ ). À l'inverse, pour une information totalement inintéressante, l'oubli naturel dominera très vite. L'envie de partager s'estompe presque immédiatement, entraînant un passage rapide des Propagateurs vers l'état passif de Muets ( $\gamma_2 \approx 0.2$ ), tandis que la prise de conscience est plus lente car le sujet ne suscite aucun débat poussant à la vérification ( $\gamma_1 \approx 0.05$ ).

## 4.2 Scénario de base

Dans cette section, nous résolvons le système numériquement à l'aide de Python en utilisant les valeurs des coefficients déterminés précédemment. Nous allons utiliser la fonction `odeint` de `scipy.integrate` pour résoudre le système d'équations différentielles. Le code à l'annexe 1 montre l'implémentation de la résolution en python et les graphiques des différentes variables en fonction du temps. **Résultat pour information peu virale :**

Les coefficients sont donc à leurs valeurs minimales :

$$\beta = 50$$

$$\alpha = 0.0005$$

$$\gamma_1 = 0.05$$

$$\gamma_2 = 0.2$$

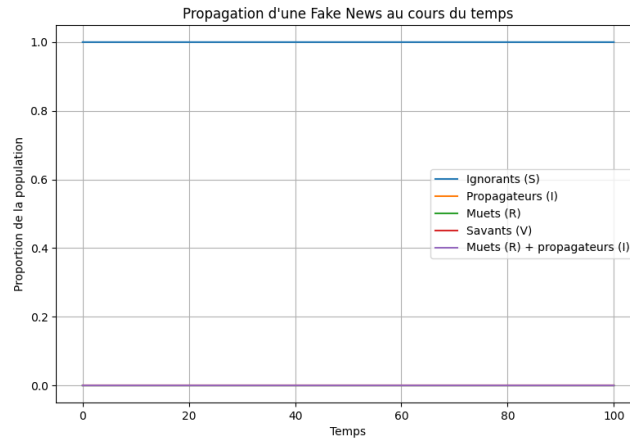


FIGURE 1 – Évolution temporelle des populations avec 1 propagateur initial.

Le graphique ci-dessus est le cas où on n'a que 1 propagateur initial sur toute la communauté francophone de X. On a donc en normalisant par la population totale  $I_0 = \frac{1}{12000000}$ , et  $S_0 = 1 - I_0$ . On voit bien que le nombre de personnes ayant connaissance de l'information tend vers 0, et ce jusqu'à à peu près 1000 propagateurs initiaux. L'information ne se propage donc quasiment pas. Prenons maintenant le cas où il y a 50000 propagateurs initiaux.

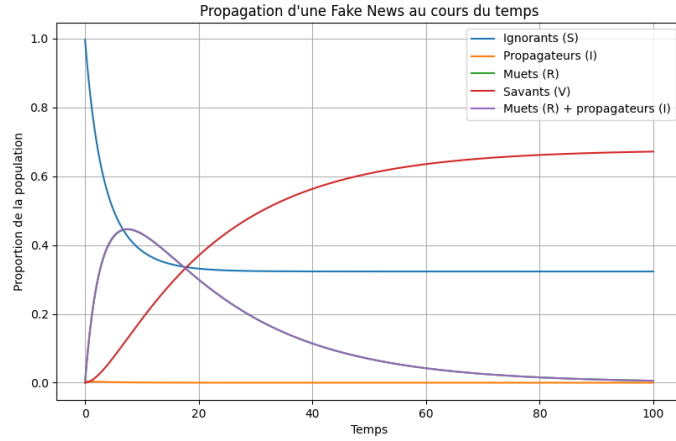


FIGURE 2 – Évolution temporelle des populations avec 50000 propagateurs initiaux.

On voit ici que le pic est atteint en un peu moins de 30 jours. On voit maintenant un vrai pic de personnes ayant vu la news, mais plus de 30% de la population ne vont jamais voir la news. Il est donc quasiment impossible de propager une fausse information peu intéressante et peu virale, car même avec 50000 propagateurs, la news n'est pas propagée très loin. **Résultat pour information très virale :**

Prenons maintenant les valeurs les plus élevées des paramètres, i.e. :

$$\beta = 300$$

$$\alpha = 0.003$$

$$\gamma_1 = 0.12$$

$$\gamma_2 = 0.06$$

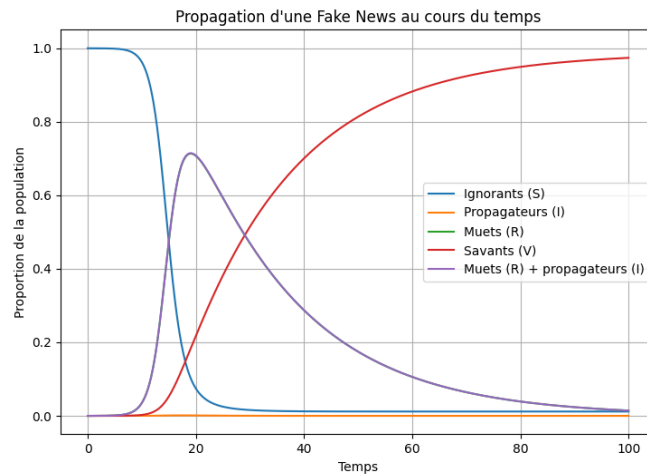


FIGURE 3 – Évolution temporelle des populations avec 1 propagateur initial.



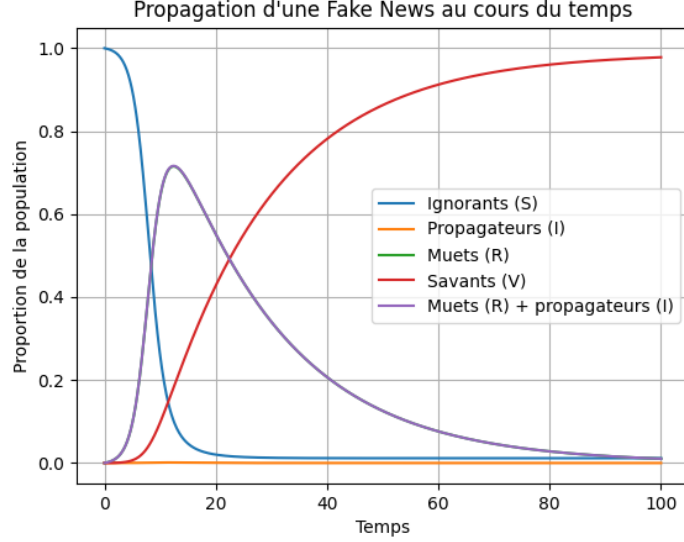


FIGURE 4 – Évolution temporelle des populations avec 100 propagateurs initiaux.

Pour une information extrêmement virale, on voit que le pic est très prononcé. Cependant, on voit quelque chose d'étrange se produire : l'augmentation du nombre de propagateurs initiaux a très peu d'impact sur le pic, en termes de temporalité et d'amplitude ; en passant de 1 à 100 propagateurs, on recule le pic de 6-7 jours, et l'amplitude du pic ne change quasiment pas. Plus on augmente le nombre de propagateurs, plus le pic va tendre vers 1 en  $t = 0$ .

### 4.3 Scénario d'intervention (Contre-mesures)

L'État peut tenter de contrer les fake-news grâce à 2 leviers principaux : la censure et la prévention. Dans cette section, nous allons essayer de trouver des formules décrivant les ressources nécessaires pour chaque technique, et ainsi trouver une combinaison optimale selon l'objectif. On essaiera d'abord de minimiser la hauteur du pic sous contrainte d'un budget fixe, et ensuite de minimiser les coûts d'une campagne sous contrainte d'une certaine valeur du pic violet. Il faut donc aussi trouver comment l'intervention de l'État influence les paramètres de notre système.

#### 4.3.1 Impact sur les coefficients

Pour modéliser la censure, on peut partir de 2 types de censures différentes. Si on utilise le shadowbanning (les fake-news ne se retrouvent plus devant les utilisateurs), on diminue le paramètre de contagion  $\beta$ . D'un autre côté on peut partir de la définition plus naturelle que la censure est la suppression des faux contenus, voire même un bannissement des comptes propagateurs. On va dans ce cas modifier le paramètre  $\gamma_2$ .

Pour modéliser l'impact du shadowbanning sur notre modèle, on va introduire une variable  $u_c$ , tel que  $\beta_{eff} = \beta(1 - u_c)$ . Dans ce cas, le propagateur croit toujours diffuser l'information, mais il ne touche en réalité plus aucun ignorant ( $S$ ). Pour les suppressions, on utilisera un paramètre  $u_2$  traduisant l'effort de modération. On dira tout simplement que  $\gamma_{2,eff} = \gamma_2 + u_2$ .

Ces 2 leviers ont chacun des avantages et des inconvénients. Le simple bannissement est plus simple technologiquement et moins coûteux, mais il a un coût social beaucoup plus important. La suppression arbitraire des comptes est publique, et peut aller dans la direction de certains argumentaires conspirationnistes et provoquer des syndromes de Galilée. Une partie de la population peut donc devenir plus méfiante au regard de l'État, et donc diminuer le paramètre  $k_1$ . Le shadowbanning est moins cher en coût social, mais est plus complexe technologiquement et onéreux, car il nécessite la modification d'algorithmes complexes. Il serait donc intéressant d'essayer de résoudre le problème d'optimisation en prenant cela en compte, mais je laisse cela comme idée au lecteur. Il ne restera plus qu'à lier les paramètres  $u_2$  et  $u_c$  à une campagne de modération réelle. Pour la sensibilisation par la publication de démentis, on va modifier le paramètre  $\gamma_1$ . Comme précédemment on va introduire un paramètre de contrôle  $u_1$ , et une constante d'efficacité  $k_1$ , qui dépend par exemple de la confiance de la population envers le gouvernement. En partant du principe que l'efficacité de la campagne sature après un certain moment, on va la représenter avec une exponentielle décroissante. On partira du principe que  $\gamma_{1,eff} = \gamma_1 + \gamma_{max}(1 - e^{-k_1 u_1})$ . On utilisera pour la censure la suppression plutôt que le shadowbanning, car cette méthode est plus simple pour déterminer le coût. Après ces modifications, on se retrouve avec le système suivant :

$$\frac{dS}{dt} = -\beta(1 - u_2)SI \quad (10)$$

$$\frac{dI}{dt} = \alpha\beta(1 - u_2)SI - (\gamma_2 + u_2)I \quad (11)$$

$$\frac{dR}{dt} = \beta SI(1 - \alpha) - (\gamma_1 + \gamma_{max}(1 - e^{-k_1 u_1}))R + (\gamma_2 + u_2)I \quad (12)$$

$$\frac{dV}{dt} = (\gamma_1 + \gamma_{max}(1 - e^{-k_1 u_1}))R \quad (13)$$

#### 4.3.2 Fonction budget et optimisation

Commençons par définir une fonction budget d'une campagne anti fake-news. Intuitivement, le coût de bannissement des comptes est linéaire car le coût marginal est le même. Pour la sensibilisation, il serait logique que le coût marginal augmente plus la campagne avance, ce qui nous pousse à choisir par exemple une fonction quadratique. Toucher peu de personnes est relativement aisé, et peut se faire avec un simple tweet du gouvernement, mais plus on veut toucher de personnes différentes, plus cela va coûter cher car des personnes se trouvent dans des bulles algorithmiques. On arrive donc à la fonction suivante :

$$B(u_1, u_2) = c_1 u_1^2 + c_2 u_2,$$

avec  $c_1$  et  $c_2$ , des paramètres à déterminer. On peut déterminer ces paramètres en estimant le budget de censure total (on fixe  $B(0, 1)$ , et on se dit que  $B(1, 0)$  est 5x plus cher par exemple). Formulons maintenant le problème de minimisation du pic de croyants et de

propagateurs avec un budget  $B_{max}$  :

$$\min_{u_1, u_2} \left( \max_t (I(t) + R(t)) \right) \quad (14)$$

sous contraintes que :

$$c_1 u_1^2 + c_2 u_2 \leq B_{max} \quad (15)$$

$$0 \leq u_1 \leq 1 \quad (16)$$

$$0 \leq u_2 \leq 1 \quad (17)$$

Pour minimiser le budget pour atteindre un pic en dessous d'une valeur  $\max P_{cible}$ , le problème s'énonce comme suit :

sous contraintes que :

$$\max_t (I(t) + R(t)) \leq P_{cible}$$

$$0 \leq u_1 \leq 1$$

$$0 \leq u_2 \leq 1$$

Le code python en annexe 2 résout le problème au budget maximum, et trace la courbe des muets et des propagateurs, avec et sans intervention de l'État. On ajoutera dans le code une variable  $t_{delay}$ , qui est le temps que met la campagne à démarrer. Ce paramètre est très important car sans lui, la courbe de des croyants resterait à 0. En effet, si on commence avec 1 propagateur initial et que la campagne commence directement, ce propagateur se ferait directement bannir. En utilisant les paramètres dans le code, on obtient le graphique suivant :

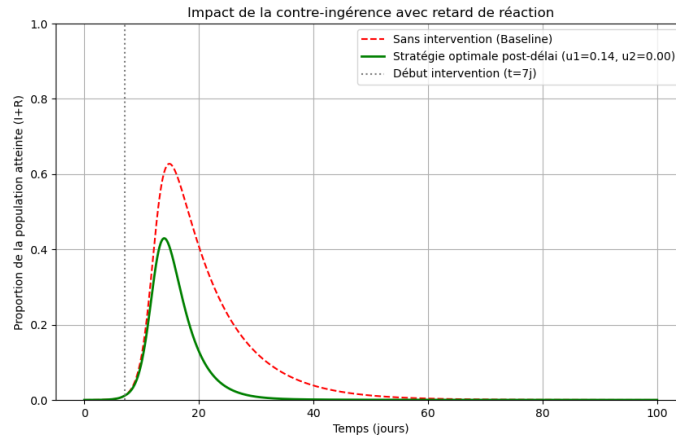


FIGURE 5 – Évolution temporelle des populations avec 1 propagateur initial.

On voit aussi grâce au print que  $u_1 = 0.14$ , et  $u_2 = 0$ . La solution optimale avec ces paramètres nous dit donc que l'État doit mettre tout son budget dans la sensibilisation. Cette situation n'est pas très intéressante car la censure linéaire n'a pas le temps d'être meilleure que la sensibilisation quadratique. En utilisant des paramètres de coûts plus intéressants pour la simulation, i.e.  $c_1 = 800$ , et  $c_2 = 30$ , on obtient le graphique suivant :

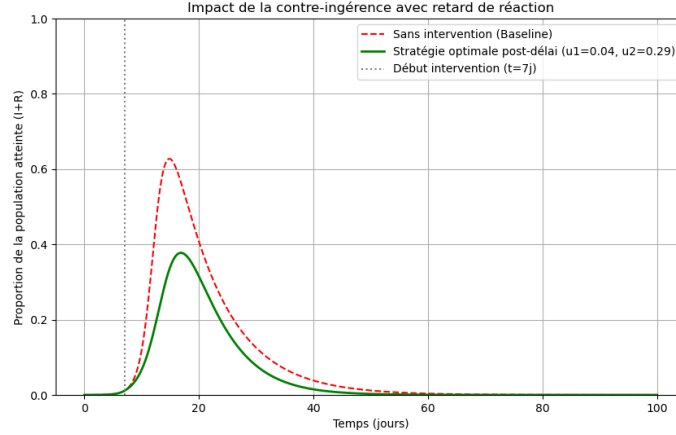


FIGURE 6 – Évolution temporelle des populations avec 1 propagateur initial.

Les quantités optimales sont  $u_1 = 0.0395$  et  $u_2 = 0.2919$ , ce qui est plus modéré. On voit un réel impact sur la courbe grâce à la sensibilisation et la modération, avec une diminution du pic d'approximativement 30%.

## 5 Conclusion

### 5.1 Rappel

En conclusion, nous avons adapté le modèle épidémiologique SIR à la transmission de fake-news sur les réseaux sociaux. Le modèle repose sur des paramètres  $(\alpha, \beta, \gamma_1, \gamma_2)$  qui doivent être déterminés par des données réelles, ce qui peut être fait dans le cadre d'études sociologiques. La section 3.1 détermine le seuil de propagation  $R_0$ . Ce seuil est très important car il donne une information sur l'évolution de la transmission. Si  $R_0 > 1$ , l'épidémie s'étend, si  $R_0 = 1$ , l'épidémie stagne, et si  $R_0 < 1$ , l'épidémie s'éteint. Ensuite, dans la section 4.3, on essaye d'optimiser les campagnes de contre-mesures. On se rend compte grâce au code python pour résoudre le problème d'optimisation que l'optimum dépend du budget et des coûts de chaque type d'intervention.

### 5.2 Apport du projet

Ce modèle peut servir à optimiser les budgets d'intervention contre les fake-news avant de dépenser des ressources réelles. Pour qu'il soit d'une quelconque utilité, il est impératif d'avoir de bonnes approximations des paramètres de la modélisation  $(\alpha, \beta, \gamma_1, \gamma_2)$ , et d'avoir les bons paramètres de coût des contre-mesures  $(B, c1, c2, \gamma_{1,eff}$  et  $\gamma_{2,eff}$ ). Il y a aussi un paramètre très important qu'il faut prendre en compte,  $t_{delay}$ , le temps de réaction entre l'arrivée de la news et la début de la campagne.

### 5.3 Limites et critiques du modèle

Pour notre modèle adapté du SIR, on suppose que la population est homogène et que tout le monde peut interagir avec tout le monde. Dans la réalité, c'est beaucoup plus compliqué car les réseaux sociaux sont composés de "bulles de filtres". Ensuite, chaque paramètre est extrêmement compliqué à déterminer précisément, et chaque paramètre dépend du

temps, alors que notre modèle les suppose constants. Enfin, les bannissements de comptes réduisent la crédibilité de l'État, et vont donc diminuer le paramètre  $k_1$ , ce qui va rendre moins efficace la sensibilisation.

## 5.4 Améliorations

Voici quelques idées d'améliorations ou de sujets intéressants à explorer :

### 5.4.1 Intégration des 'bots'

Pour simuler des 'bots' qui propageraient l'information sans s'arrêter, on pourrait introduire une nouvelle variable  $B$ , le taux de bots dans la population. On aurait donc 1 changement dans le système d'équation :  $\frac{dR}{dt} = \beta SI(1 - \alpha) - \gamma_1 R + \gamma_2(I + B)$ .

### 5.4.2 Modélisation en graphe

Pour prendre en compte les bulles d'algorithme, on pourrait sortir du modèle compartimental (EDO), pour passer à un modèle en graphe et comme ça mieux prendre en compte les structures des réseaux sociaux.

### 5.4.3 Notes de la communauté

Une idée intéressante serait d'analyser les notes de la communauté comme introduit sur X pour comprendre si cela est vraiment plus efficace que la modération classique.

## A Code python résolution système

Listing 1 – Résolution du système d'équations différentielles

```

1 import numpy as np
2 from scipy.integrate import odeint
3 import matplotlib.pyplot as plt
4
5 def modele_fake_news(y, t, beta, alpha, gamma1, gamma2):
6     S, I, R, V = y
7
8     dS_dt = -beta * S * I
9     dI_dt = beta * S * I * alpha - gamma2 * I
10    dR_dt = beta * S * I * (1 - alpha) - gamma1 * R + I * gamma2
11    dV_dt = R * gamma1
12
13    return [dS_dt,
14            dI_dt, dR_dt, dV_dt]
15
16 beta = 50
17 alpha = 0.0005
18 gamma1 = 0.05
19 gamma2 = 0.2
20
21 I0 = 1 / 12000000
22 S0 = 1 - I0

```

```

23 R0 = 0
24 V0 = 0
25 y0 = [S0, I0, R0, V0]
26
27 t = np.linspace(0, 100, 1000)
28
29 solution = odeint(modele_fake_news, y0, t, args=(beta, alpha,
30          gamma1, gamma2))
31
32 S_res = solution[:, 0]
33 I_res = solution[:, 1]
34 R_res = solution[:, 2]
35 V_res = solution[:, 3]
36
37 plt.plot(t, S_res, label='Ignorants(S)')
38 plt.plot(t, I_res, label='Propagateurs(I)')
39 plt.plot(t, R_res, label='Muets(R)')
40 plt.plot(t, V_res, label='Savants(V)')
41 plt.plot(t, I_res + R_res, label='Muets(R) + propagateurs(I)')
42 plt.xlabel('Temps')
43 plt.ylabel('Proportion de la population')
44 plt.title('Propagation d\'une Fake News au cours du temps')
45 plt.legend()
46 plt.grid()
47 plt.show()

```

## B Code python optimisation budget

Listing 2 – Algorithme d'optimisation sous contrainte de budget

```

1 import numpy as np
2 from scipy.integrate import odeint
3 from scipy.optimize import
4     minimize
5 import matplotlib.pyplot as plt
6
7
8 beta, alpha, gamma1, gamma2 = 300, 0.003, 0.12, 0.06
9 gamma_max, k1 = 0.5, 3
10 c1, c2 = 500, 100
11 B_max = 10
12
13 t_delay = 7
14
15 y0 = [1 - 1e-7, 1e-7, 0, 0]
16 t = np.linspace(0, 100, 1000)
17
18
19 def simuler(u):
20     u1, u2 = u
21

```

```

22     def modele(y, t):
23         S, I, R, V = y
24         if t < t_delay:
25             gamma1_actuel = gamma1
26
27         gamma2_actuel = gamma2
28         else:
29             gamma1_actuel = gamma1 + gamma_max * (1 - np.exp(-k1 *
30                 u1))
31             gamma2_actuel = gamma2 + u2
32
33         beta_actuel = beta
34
35         dS_dt = -beta_actuel * S * I
36         dI_dt =
37         alpha * beta_actuel * S * I - gamma2_actuel * I
38         dR_dt = beta_actuel * S * I * (1 - alpha) - gamma1_actuel *
39             R + gamma2_actuel * I
40         dV_dt = gamma1_actuel * R
41
42         return [dS_dt, dI_dt, dR_dt, dV_dt]
43
44     return odeint(modele, y0, t, rtol=1e-10, atol=1e-10)
45
46 def calcul_pic(u):
47     solution = simuler(u)
48     return np.max(solution[:, 1] +
49         solution[:, 2])
50
51 def contrainte_budget(u):
52     u1, u2 = u
53     return B_max - (c1 * u1**2 + c2 * u2)
54
55 bounds = [(0, 1), (0, 1)]
56 constraints = {'type': 'ineq', 'fun': contrainte_budget}
57 u0 = [0.0, 0.0]
58
59 resultat = minimize(calcul_pic, u0, method='SLSQP', bounds=bounds,
60     constraints=constraints, options={'eps': 1e-3,
61         'disp': True})
62
63 u1_opt, u2_opt = resultat.x
64 print("u1_optimal:", u1_opt)
65 print("u2_optimal:", u2_opt)
66
67 sol_base = simuler([0, 0])
68 courbe_base = sol_base[:, 1] + sol_base[:, 2]
69
70 sol_opt = simuler(resultat.x)
71 courbe_opt = sol_opt[:, 1] + sol_opt[:, 2]

```

```

70
71 plt.figure(figsize=(10, 6))
72 plt.plot(t, courbe_base, 'r--', label='Sans intervention (Baseline)')
73 plt.plot(t, courbe_opt,
74          'g', linewidth=2,
75          label=f'Strategie optimale post-delai (u1={u1_opt:.2f}, u2={u2_opt:.2f})')
76
77 plt.axvline(x=t_delay, color='gray', linestyle=':', label=f'Debut intervention (t={t_delay}j)')
78
79 plt.title("Impact de la contre-ingérence avec retard de réaction")
80 plt.xlabel('Temps (jours)')
81 plt.ylabel('Proportion de la population atteinte (I+R)')
82 plt.ylim(0, 1)
83 plt.legend()
84 plt.grid(True)
85 plt.savefig('comparaison_optimisation_delay.png', dpi=300)
86 plt.show()

```