Rapport de stage

Arnaud COSTERMANS et Achile PINSARD

Année universitaire : 2022-2023

Année d'études : Promotion 2024 (L2)
Licence de Science et Technologie
Institut Villebon - *Georges Charpak*

Maître de stage : Cyril DAUPHIN

Enseignante référente : Martine THOMAS

Stage effectué du 15/05 au 13/06

Remerciement

Nous aimerions remercier Cyril DAUPHIN ainsi que l'Institut Villebon - *George Charpak* de nous avoir accueillis pour ce stage.

Résumé

Ce rapport de stage présente le travail que nous avons effectué sur l'étude de transfert radiatif d'un modèle à deux faisceaux lumineux. Actuellement, le modèle à deux faisceaux dans l'espace des réels est bien compris, mais nous allons partir de ce modèle pour établir le portrait de phase de ce modèle. On a démontré que le volume s'y conservait dans le cas isotrope, réduisant ainsi le nombre de calculs numériques à effectuer. Nous avons ensuite établi les équations de ce modèle aux ordres supérieur avant de nous intéresser à des cas concrets comme le comportement de différentes longueurs d'onde dans l'eau et dans l'air et le comportement de la lumière dans un système avec un nuage et un sol réfléchissant.

Table des matières

Table des figures

1 Introduction

Ces dernières années, le domaine du transfert radiatif ne cesse d'évoluer. La nature complexe de ces géométries introduit des difficultés dans la modélisation précise du transfert de rayonnement. De fait, les chercheurs étudient des méthodes de calcul avancées, toutefois les exigences de calcul associées à ces méthodes sont importantes, nécessitant des ressources informatiques considérables.

La lumière peut interagir de multiples manières avec la matière, en effet, quand un photon rentre en collision avec des atomes alors le photon est absorbé par cet atome ce qui a pour conséquence de l'élever à un état excité, cela s'appelle **l'absorption**. Suite au phénomène précédent, l'énergie des photons emmagasinée lors de l'absorption peut être ré-émise dans une direction quelconque, c'est ce qu'on appelle **l'émission**. Comme l'émission peut se de-exciter en passant par différents niveaux d'énergie que lors de l'absorption, le photon émis ne sera pas forcément de la même longueur d'onde (par exemple du niveau $1 \to 3$ pour l'absorption, de $3 \to 2$ et $2 \to 1$ pour l'émission ; respectivement en bleu, rouge et vert sur la figure 1) [2]. Enfin, nous discernons une troisième loi : lorsqu'une absorption puis une émission d'un photon à la même longueur d'onde ont lieu dans un temps très court, cela s'appelle de **la diffusion**, le photon est alors ré-émis avec une nouvelle trajectoire et un nouvel angle aléatoire.

Le transfert radiatif est le processus par lequel l'énergie est transportée sous forme de photons à travers un milieu, le but de ce stage est d'étudier le portrait de phase du modèle de transfert radiatif à 2 faisceaux. Pour cela, nous étudierons dans un premier temps un modèle à doubles faisceaux. Après avoir généralisé à un nombre de faisceaux supérieurs, nous explorerons la diversité des interactions en fonction de la fréquence des flux et du milieu concerné, ouvrant ainsi de nouvelles perspectives dans notre compréhension du phénomène.

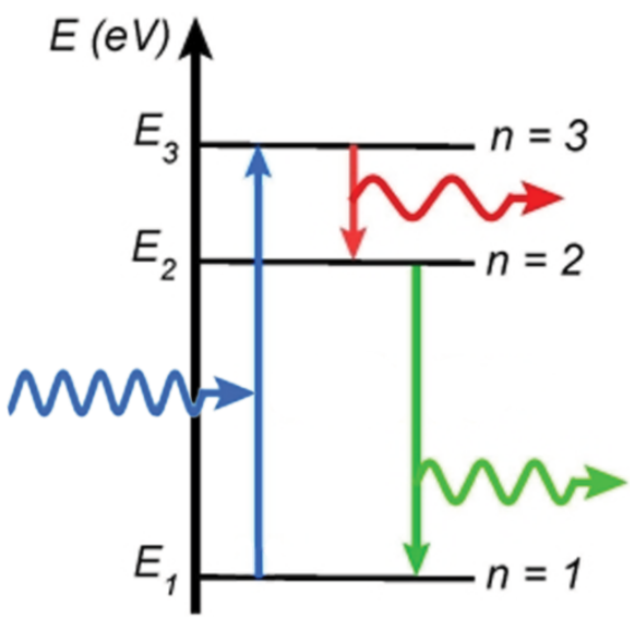


FIGURE 1 – Schéma des niveaux d'énergie d'un atome par des photons

2 Mise en équation du modèle à deux faisceaux

Nous nous sommes intéressés au comportement des flux lumineux dans le modèle du transfert radiatif à deux faisceaux comme il a été décrit dans « Fundamentals of atmosphere radiation » [1]. Ce modèle composé d'un milieu exposé à deux flux de même orientation, mais de sens opposé, c'est-à-dire qu'il n'y a pas d'émission et de diffusion vers les côtés (fig :3, seulement vers le haut ou le bas (selon une direction verticale seulement). Les deux flux s'influencent l'un et l'autre par diffusion et sont atténués par absorption. En effet, nous ne prendrons pas en compte l'émission spontanée [1]. Une direction verticale signifie qu'il y a deux sens pour l'émission, elle n'est pas systématiquement dans un sens ou l'autre, en effet elle a une certaine probabilité de l'être [2].

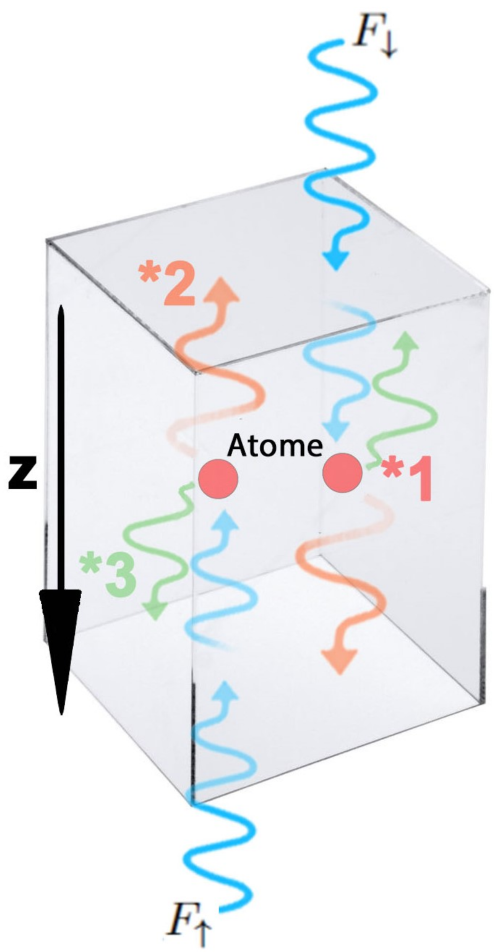


FIGURE 2 – Schéma de l'irradiance de la partie supérieure et inférieure d'un milieu par les flux respectifs $F_\downarrow$ et $F_\uparrow$, selon l'axe Oz orienté vers le bas.

Les interactions considérées dans ce modèle sont donc :
— L'**absorption** du photon par l'atome (Fig 2.*1)
— La **diffusion** du photon vers l'avant (Fig 2.*2)
— La **diffusion** du photon vers l'arrière (Fig 2.*3)

1. l'émission spontanée est une émission qui a lieu un certain temps après l'absorption correspondante.
2. La probabilité qu'un photon suivant une trajectoire vers le haut soit diffusé par l'atome puis renvoyé vers le bas ou réciproquement sont respectivement nommé : $p_{\uparrow\downarrow}$ et $p_{\downarrow\uparrow}$.

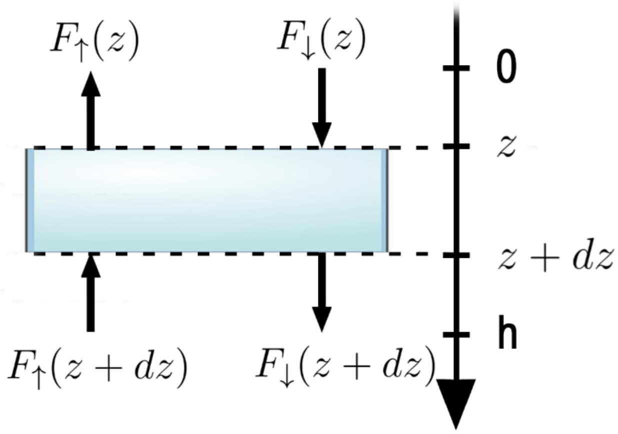


FIGURE 3 – Schéma inspiré de l'article de Bohren[1].

Par conservation de l'énergie, nous pouvons exprimer les flux finaux $F_\downarrow(z+dz)$ et $F_\uparrow(z+dz)$, en fonction de : κ le coefficient d'absorption, β le coefficient de diffusion en m^{-1}, $F_\downarrow$ le flux vers le bas et $p_{\downarrow\uparrow}$ la probabilité qu'un photon orienté vers le bas aille vers le haut. En effet pour exprimer le flux descendant sortant en $z+dz$ il faut lister ce qui l'exprime :

— L'expression du flux descendant doit alimenter l'équation : $F_\downarrow$
— Il faut soustraire la partie absorbée de ce flux sur l'infime partie de z qui s'exprime avec le coefficient d'absorption, car ils ne font plus partie du flux descendant, soit : $\kappa dz F_\downarrow$.
— Soustraire également (pour la même raison) l'expression des photons de ce flux diffusé vers le haut : $\beta dz p_{\downarrow\uparrow} F_\downarrow(z+dz)$
— Et enfin y ajouter la partie du flux montant qui alimente le flux descendant par sa diffusion : $\beta dz p_{\uparrow\downarrow} F(z+dz)$

Ce qui nous donne l'équation générale du flux $F_\downarrow(z+dz)$:

$$F_\downarrow(z+dz) = F_\downarrow(z) - \kappa dz F_\downarrow(z) - \beta dz p_{\downarrow\uparrow} F_\downarrow(z+dz) + \beta dz p_{\uparrow\downarrow} F_\uparrow(z+dz) \tag{1}$$

Après avoir posé les principes de ce modèle, nous pouvons établir des équations, confirmées par Bohren ; dont la caractérisation du comportement des flux descendants :

$$\frac{dF_\downarrow}{dz} = -\kappa F_\downarrow - \beta p_{\downarrow\uparrow} F_\downarrow + \beta p_{\uparrow\downarrow} F_\uparrow \tag{2}$$

En réitérant le même raisonnement avec le flux montant ainsi que le sens de $-\widehat{u_z}$:

$$\frac{dF_\uparrow}{dz} = \kappa F_\uparrow + \beta p_{\uparrow\downarrow} F_\uparrow - \beta p_{\downarrow\uparrow} F_\downarrow \tag{3}$$

Pour s'affranchir des probabilités, nous allons définir g, le paramètre d'asymétrie, qui sera compris entre -1 (diffusion totale vers l'arrière) et 1 (diffusion totale vers l'avant) tel que dans le cas isotrope[3] :

$$p_{\uparrow\downarrow} = p_{\downarrow\uparrow} = \frac{1-g}{2} \tag{4}$$

3. Un milieu isotrope est un milieu dont les propriétés physiques de celui-ci sont invariantes selon la direction, cela se traduit ici comme $p_{\uparrow\downarrow} = p_{\downarrow\uparrow}$.

$$p_{\downarrow\downarrow} = p_{\uparrow\uparrow} = \frac{1+g}{2} \tag{5}$$

On définit également l'albédo de simple diffusion [4] par :

$$w = \frac{\beta}{\beta+\kappa} \tag{6}$$

On remarque que, lorsque w est nul, il n'y a pas de diffusion, tandis que, lorsque w vaut 1, il n'y a pas d'absorption. Nous allons également appeler τ **l'épaisseur optique** tels que :

$$d\tau = (\beta+\kappa)dz \tag{7}$$

L'épaisseur optique correspond à la distance parcourue dans le milieu par un flux, dans une unité particulière qui est le **libre parcours moyen** [5], c'est la distance moyenne parcourue par un photon pour que celui-ci est subi une interaction avec un atome.

En utilisant les équations 4, 6 et 7 avec 2 ou 3 on obtient :

$$\frac{dF_\downarrow}{d\tau} = (w-1)F_\downarrow - w\frac{1-g}{2}F_\downarrow + w\frac{1-g}{2}F_\uparrow \tag{8}$$

$$\frac{dF_\uparrow}{d\tau} = (1-w)F_\uparrow + w\frac{1-g}{2}F_\uparrow - w\frac{1-g}{2}F_\downarrow \tag{9}$$

3 Analyse du système dynamique

Maintenant que nous avons établi ce système d'équations différentielles couplées, nous allons utiliser les méthodes utilisées dans l'étude des systèmes dynamiques [3] pour obtenir le portrait de phase du système. Ces méthodes sont utilisées lorsqu'on dérive par rapport au temps, mais on peut également s'en servir lorsque nous dérivons par l'épaisseur optique. La première étape est de passer nos équations sous forme matricielle. Les équations 8 et 9 donnent :

$$\begin{pmatrix} \dot{F_\downarrow} \\ \dot{F_\uparrow} \end{pmatrix} = \begin{pmatrix} -1+w\frac{1+g}{2} & w\frac{1-g}{2} \\ -w\frac{1-g}{2} & 1-w\frac{1+g}{2} \end{pmatrix} \begin{pmatrix} F_\downarrow \\ F_\uparrow \end{pmatrix} \tag{10}$$

On va ensuite déterminer les vecteurs propres et valeurs propres de cette matrice. Les valeurs propres nous permettront de déterminer l'allure du diagramme de phase et les vecteurs propres nous permettront d'en déterminer les axes directeurs. Pour simplifier les équations, on pose :

$$\begin{cases} a = 1 - w\left(\frac{1+g}{2}\right) \\ b = w\left(\frac{1-g}{2}\right) \end{cases} \tag{11}$$

On obtient les valeurs propres et vecteurs propres suivant :

$$\begin{cases} \lambda_1 = \sqrt{a^2-b^2} & \overrightarrow{v_1} = \begin{pmatrix} b \\ a+\lambda_1 \end{pmatrix} \\ \lambda_2 = -\sqrt{a^2-b^2} & \overrightarrow{v_2} = \begin{pmatrix} a+\lambda_1 \\ b \end{pmatrix} \end{cases} \tag{12}$$

4. L'albédo est une valeur comprise entre 0 et 1 qui s'obtient par le rapport entre le flux entrant et le flux réfléchi vers l'arrière

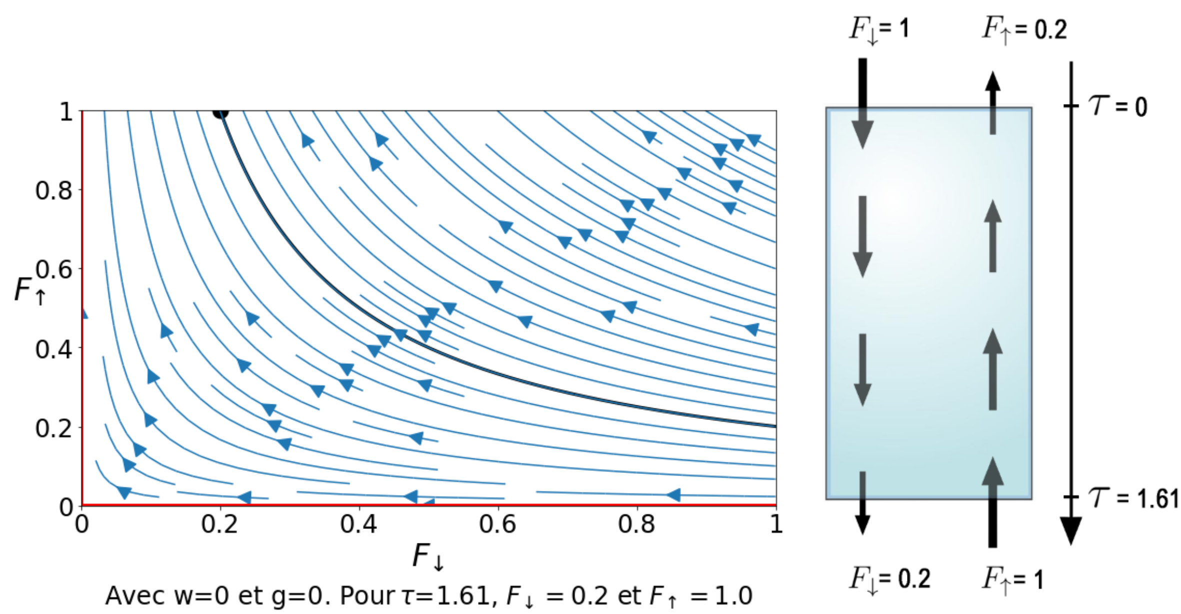
5. C'est pourquoi deux faisceaux dont le coefficient β et κ auront parcouru la même distance à la sorti d'un milieu, mais pas forcément la même épaisseur optique

Nous avons vérifié que nos calculs étaient bons avec le site "wolframe alpha". On remarque que $a - b > 0$ et donc que λ_1 et λ_2 sont des réels de signe opposé et donc que notre diagramme de phase aura un point selle en (0,0).

4 Traçage des graphes

Afin de comprendre comment les flux évoluent entre eux, nous avons tracé des graphes de phase pour plusieurs valeurs de w et g.

Pour cela, nous avons utilisé la fonction "streamplot" de la bibliothèque "mathplotlib". Cette fonction permet habituellement de tracer des vitesses en fonction des positions, pourtant comme le temps peut être considéré comme analogue à l'épaisseur optique, nous pouvons exploiter cette fonction. Cette fonction permet de tracer l'ensemble des solutions pour ce système dynamique, mais pas une solution particulière. Pour faire cela, nous nous sommes tournés vers la fonction `"solve_ivp"` de la bibliothèque "scipy.integrate". Cette fonction prend en argument une fonction qui contient notre système dynamique, une valeur initiale et finale de τ et les valeurs initiales de $F_\downarrow$ et $F_\uparrow$ puis renvoie deux listes contenant les valeurs de $F_\downarrow$ et $F_\uparrow$ au cours de τ que l'on trace avec "plot" de "mathplotlib".

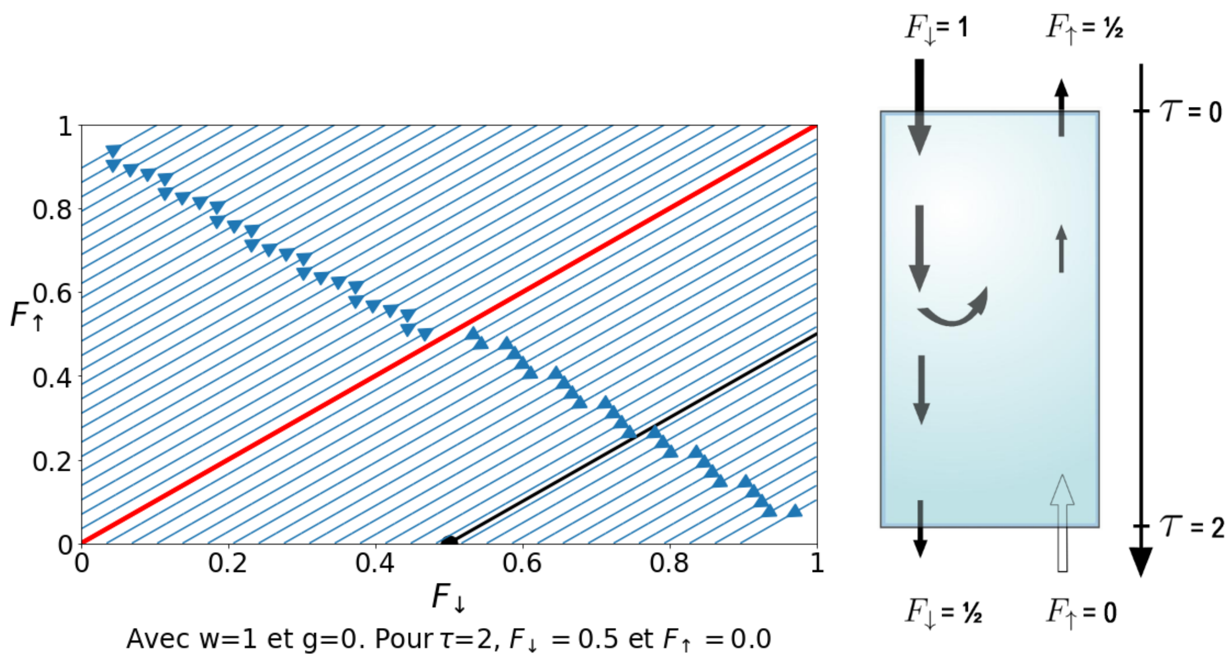


(a) (b)

FIGURE 4 – Comportement des flux lorsque $w = 0$.
(a) Graphe de $F_\uparrow$ en fonction de $F_\downarrow$. Le schéma ne dépend pas de g quand $w = 0$. Les vecteurs propres sont représentés en rouge, ils sont confondus avec les axes.
(b) Schéma du comportement des flux lumineux de la courbe noire, la taille des flèches est proportionnelle à l'intensité du flux lumineux.

Nous avons maintenant des graphiques qui représentent l'évolution des flux montants en fonction des flux descendants. Ces graphiques sont des champs vectoriels dont une courbe représente tous les couples de flux montant et descendant en fonction de l'épaisseur optique pour une condition initiale. Les flèches représentent la direction du vecteur quand τ augmente. La courbe noire représente la solution particulière du système dynamique qui est représenté sur le schéma à droite (b) et le point noir l'état du système pour un τ final fixé.

Dans le cas où w=0 (Fig : 4), cela signifie que l'albédo est nul, c'est-à-dire qu'il n'y a pas de diffusion, nous retrouvons alors la loi de Beer-Lambert. La totalité du flux rentrant est donc égale au flux sortant additionné à celui absorbé dans le milieu pour chaque faisceau. Prenons l'exemple de la courbe noire : initialement avec un flux vers le bas et vers le haut respectivement de 1 et 0.2, en augmentant l'épaisseur optique jusqu'à 1.61, nous constatons que le flux est absorbé de manière exponentielle jusqu'à atteindre un flux montant de 1 et descendant [6] de 0.2. Le cas souvent observé à cause de notre soleil est avec initialement $F_{\downarrow} = 1$ et $F_{\uparrow} = 0$, nous constatons que lorsque τ augmente le flux montant est inchangé mais le flux descendant tend vers 0.

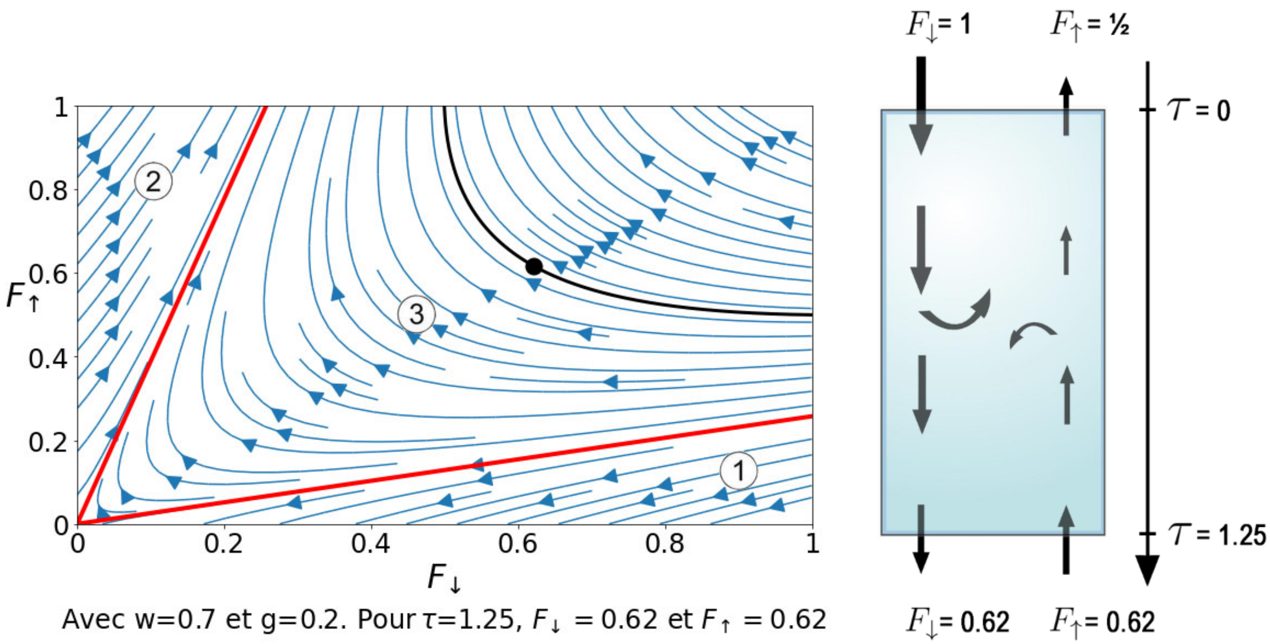


(a) (b)

FIGURE 5 – Comportement des flux lorsque $w = 1$.
(a) Graphe de $F_{\uparrow}$ en fonction de $F_{\downarrow}$. Le schéma ne dépend pas de g lorsque $w = 1$. Les vecteurs propres sont représentés en rouge, ils sont confondus entre eux.
(b) : Schéma du comportement des flux lumineux de la courbe noire, la taille des flèches est proportionnelle à l'intensité du flux lumineux.

Dans le cas où $w = 1$, β a la même valeur partout mais seul les conditions initiales changent. L'absorption est nulle et le portrait de phase est montré sur la figure 5. Plus la droite se rapproche de l'axe des vecteurs propres, plus l'effet de diffusion s'intensifie jusqu'à atteindre une diffusion complète. Concernant la nouvelle courbe noire, nous pouvons constater que les flux initiaux sont de 0.5 en ordonnée et 1 en abscisse. La seule manière d'obtenir ce cas de figure avec un flux montant final nul est que la moitié du flux descendant soit ré-émis vers l'arrière pour devenir du flux montant et que l'autre moitié poursuit sa trajectoire, car il n'y a pas d'absorption (Fig : 5b).

6. flux montant signifie flux vers le haut, et réciproquement pour le flux descendant



(a) (b)

FIGURE 6 – Comportement des flux lorsque $w = 0.7$ et $g = 0.2$.
(a) Graphe de $F_\uparrow$ en fonction de $F_\downarrow$. Le choix de $w = 0.7$ et $g = 0.2$ est arbitraire pour représenter un cas quelconque. Les vecteurs propres sont représentés en rouge.
(b)Comportement des flux lorsque $w = 0.7$ et $g = 0.2$ pour la courbe noire, la taille des flèches est proportionnelle à l'intensité du flux lumineux.

Lorsque l'on prend un cas quelconque, l'interprétation est plus complexe, en effet ici tous les phénomènes vus précédemment sont présents avec les proportions variables, par exemple comme g=0.7 dans ce cas $p_{\uparrow\downarrow}$ et $p_{\downarrow\uparrow}$ sont de $\frac{1-0.7}{2} = 0.15$.

Nous distinguons trois zones sur ce graphique (Fig : 6a). Dans la zone comprise entre le vecteur propre et l'axe des abscisses (zone 1), nous constatons que tous les flux montants tendent vers 0, cela est dû au fait que pour un flux $F_\downarrow$ de 1, si $F_\uparrow$ est inférieur ou égal à 0.25 alors il tendra vers 0 en augmentant l'épaisseur optique. La composante de $F_\downarrow$ qui est diffusée vers l'arrière ne permet pas d'alimenter le flux montant à cause de l'absorbance du milieu, elle ne peut donc pas nourrir le flux opposé s'il est inférieur à 0.25. L'interprétation de la zone 2 suit le même raisonnement, mais avec τ inversé.

On peut prendre comme exemple la courbe noire pour expliquer la zone 3. Elle représente la solution particulière avec un flux initial montant de 0.5 et descendant de 1. Lorsque $\tau = 1.25$, $F_\downarrow$ et $F_\uparrow$ ont tout deux comme valeur 0.62.

4.1 Calcul de l'angle entre les vecteurs propres

Pour calculer l'angle entre les vecteurs propres, nous allons exploiter les formules du produit scalaire. On prend $V_1 = \begin{pmatrix} x_1 \\ y_1 \end{pmatrix}$ et $V_2 = \begin{pmatrix} x_2 \\ y_2 \end{pmatrix}$

$$\overrightarrow{V_1}.\overrightarrow{V_2} = ||\overrightarrow{V_1}|| * ||\overrightarrow{V_2}|| * cos(\theta) = x_1 * x_2 + y_1 * y_2 \tag{13}$$

$$\sqrt{x_1^2 + y_1^2} * \sqrt{x_2^2 + y_2^2} * cos(\theta) = x_1 * x_2 + y_1 * y_2 \tag{14}$$

$$\theta = cos^{-1}\left(\frac{x_1 * x_2 * \sqrt{x_1^2 + y_1^2} * \sqrt{x_2^2 + y_2^2} * + y_1 * y_2 * \sqrt{x_1^2 + y_1^2} * \sqrt{x_2^2 + y_2^2}}{x_1^2 x_2^2 + x_1^2 y_2^2 + y_1^2 x_2^2 + y_1^2 y_2^2}\right) \tag{15}$$

Comme on se place dans le cas isotrope où $x_1 = y_2 = b$ et $y_1 = x_2 = a + \lambda_1$ on obtient

$$\theta = cos^{-1}\left(\frac{2 * x_1 * y_1}{x_1^2 + y_1^2}\right) = cos^{-1}\left(\frac{b}{a}\right) = cos^{-1}\left(\frac{w\left(\frac{1-g}{2}\right)}{1 - w\left(\frac{1+g}{2}\right)}\right) \tag{16}$$

C'est pourquoi dans les cas extrêmes on à :

- Lorsque $w = 0$
  $\theta = cos^{-1}(0) = \frac{\pi}{2}$ ce qui suis bien le graphique 4a.
- lorsque $w = 1$
  $\theta = cos^{-1}\left(\frac{\frac{1-g}{2}}{1-\left(\frac{1+g}{2}\right)}\right) = 0$ ce qui suis bien le graphique 5a

En résolvant avec python l'équation 16 l'angle est de $\theta = 1.10\,\text{rad} = 63.55°$ pour $w = 0.7$ et $g = 0.2$

4.2 Milieu non isotrope

Le milieu étudié précédemment a jusque là été considéré comme isotrope, les propriétés du milieu étant désormais mis en évidence nous allons étudier un milieu qui ne répond pas à l'égalité de l'isotropie. En reprenant les calcule, on trouve que

$$\begin{pmatrix} \dot{F_\downarrow} \\ \dot{F_\uparrow} \end{pmatrix} = \begin{pmatrix} -1 + wp_{\downarrow\downarrow} & wp_{\uparrow\downarrow} \\ -wp_{\downarrow\uparrow} & 1 - wp_{\uparrow\uparrow} \end{pmatrix} \begin{pmatrix} F_\downarrow \\ F_\uparrow \end{pmatrix} \tag{17}$$

Sur la figure 7 nous pouvons constater que l'allure globale des courbes suivent le même modèle que dans le modèle isotrope avec également $w = 0.7$ (fig : 6). La différence drastique est le coefficient des vecteurs propres.

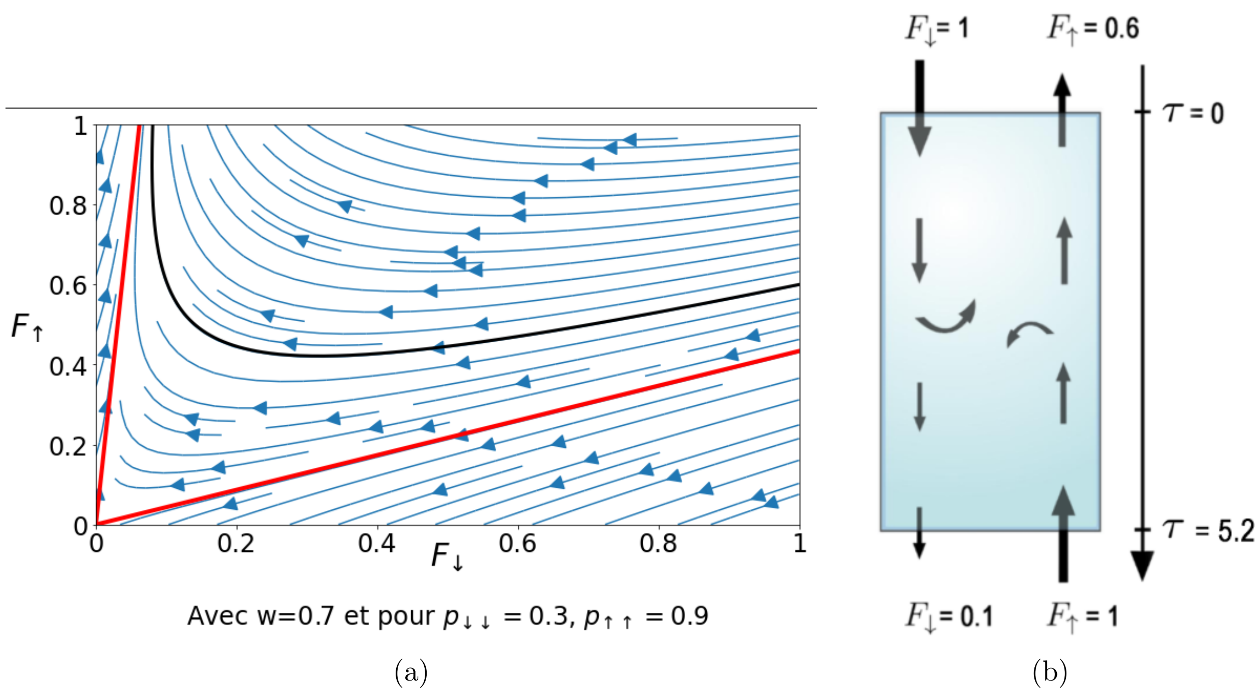


FIGURE 7 – Comportement des flux dans un milieu non isotrope pour $P_{\uparrow\uparrow} = 0.9$ et $P_{\downarrow\downarrow} = 0.3$.
(a) Graphe de $F_\uparrow$ en fonction de $F_\downarrow$ dans un milieu non isotrope . Les vecteurs propres sont représentés en rouge.
(b)Comportement des flux dans ce même milieu pour la courbe noire, la taille des flèches est proportionnelle à l'intensité du flux lumineux.

5 Conservation du volume

On sait que dans un champ de vitesse, si la divergence d'un champ vectoriel est nulle, cela indique que le volume à l'intérieur d'une région donnée reste constant. La divergence peut s'exprimer comme :

$$div\,\overrightarrow{v} = \frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} \tag{18}$$

Or τ est analogue au temps et l'on peut donc dire que $\begin{pmatrix} \dot{F_\downarrow} \\ \dot{F_\uparrow} \end{pmatrix}$ s'exprime comme un vecteur vitesse.

5.1 Cas Isotope

En utilisant les équations 10 et 11 on obtient :

$$div \begin{pmatrix} \dot{F_\downarrow} \\ \dot{F_\uparrow} \end{pmatrix} = -a + a = 0 \tag{19}$$

La divergence de $\begin{pmatrix} \dot{F_\downarrow} \\ \dot{F_\uparrow} \end{pmatrix}$ est nulle donc le volume se conserve dans l'espace des phases. On remarque que la divergence est égale à la trace de la matrice de l'équation 10. La conservation du volume implique qu'un point dans le volume initial sera contenu dans le volume final, ce qui simplifie les calculs.

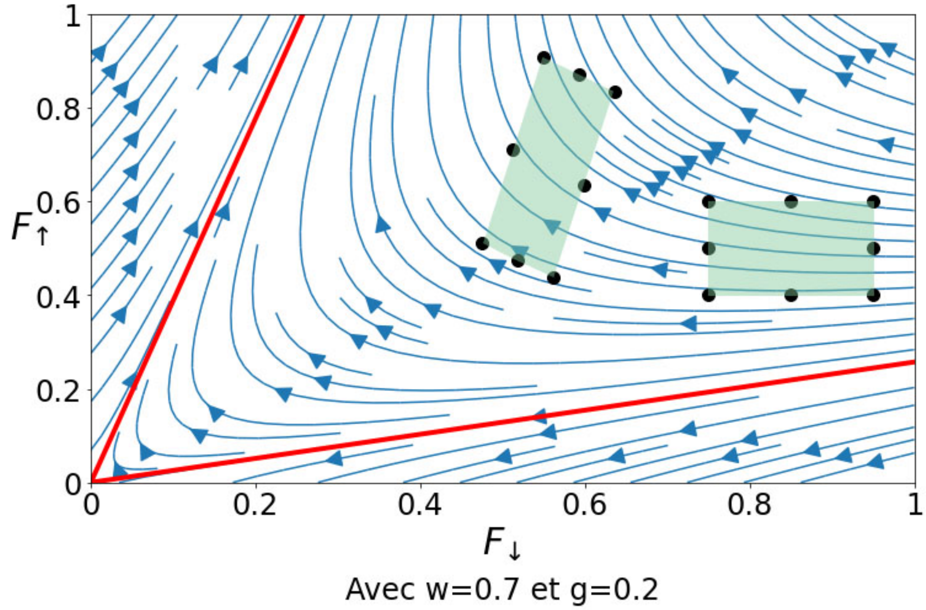


FIGURE 8 – Graphique de $F_\uparrow$ en fonction de $F_\downarrow$ représentant la conservation de la surface (les points sont des valeurs calculées pour deux valeurs de τ espacer de 1.25).

Comme le montre le graphique 8, les surfaces des deux carrés verts n'ont pas la même forme, mais possèdent le même volume. Un point contenu dans la surface d'une des formes pour un certain τ sera contenu dans l'autre surface pour le τ correspondant, il restera forcément dans cette même surface après la transformation par la matrice de l'équation 10.

5.2 Cas Non-Isotrope

Dans le cas non isotrope, on trouve à partir de 17 que :

$$div \begin{pmatrix} \dot{F}_\downarrow \\ \dot{F}_\uparrow \end{pmatrix} = -1 + wp_{\downarrow\downarrow} + 1 - wp_{\uparrow\uparrow} = w(p_{\downarrow\downarrow} - p_{\uparrow\uparrow}) \tag{20}$$

On remarque que $div \begin{pmatrix} \dot{F}_\downarrow \\ \dot{F}_\uparrow \end{pmatrix} = 0$ si et seulement si $p_{\downarrow\downarrow} = p_{\uparrow\uparrow}$ ce qui reviendrait à se placer dans le cas isotrope (ce qui est possible de voir visuellement à partir de la fig : 9), de plus le volume augmente si $p_{\downarrow\downarrow} > p_{\uparrow\uparrow}$ et diminue si $p_{\downarrow\downarrow} < p_{\uparrow\uparrow}$.

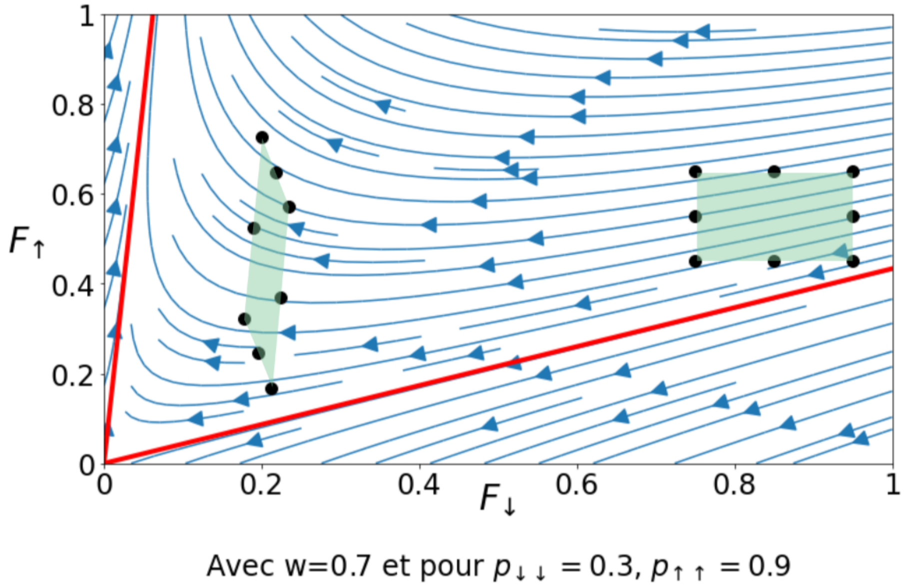


FIGURE 9 – Graphique de $F_\uparrow$ en fonction de $F_\downarrow$ représentant la non-conservation de la surface (les points sont des valeurs calculées pour deux valeurs de τ espacer de 2).

6 Modèle aux ordres supérieurs

On va essayer de généraliser le modèle a 2 faisceaux en rajoutant deux faisceaux latéraux

$$\frac{dF_\downarrow}{dz} = -\kappa F_\downarrow - \beta(p_{\downarrow\uparrow} + p_{\downarrow\leftarrow} + p_{\downarrow\rightarrow})F_\downarrow + \beta(p_{\uparrow\downarrow}F_\uparrow + p_{\leftarrow\downarrow}F_\leftarrow + p_{\rightarrow\downarrow}F_\rightarrow) \tag{21}$$

car $p_{\downarrow\uparrow} + p_{\downarrow\leftarrow} + p_{\downarrow\rightarrow} + p_{\downarrow\downarrow} = 1$, on peut donc écrire :

$$\frac{dF_\downarrow}{dz} = (-\kappa - \beta + \beta p_{\downarrow\downarrow})F_\downarrow + \beta(p_{\uparrow\downarrow}F_\uparrow + p_{\leftarrow\downarrow}F_\leftarrow + p_{\rightarrow\downarrow}F_\rightarrow) \tag{22}$$

On peut étendre cela aux trois autres équations en faisant attention au signe et à l'axe selon lequel on dérive.

$$\frac{dF_\uparrow}{dz} = (\kappa + \beta - \beta p_{\uparrow\uparrow})F_\uparrow - \beta(p_{\downarrow\uparrow}F_\downarrow + p_{\rightarrow\uparrow}F_\rightarrow + p_{\leftarrow\uparrow}F_\leftarrow) \tag{23}$$

$$\frac{dF_\leftarrow}{dy} = (-\kappa - \beta + \beta p_{\leftarrow\leftarrow})F_\leftarrow + \beta(p_{\rightarrow\leftarrow}F_\rightarrow + p_{\uparrow\leftarrow}F_\uparrow + p_{\downarrow\leftarrow}F_\downarrow) \tag{24}$$

$$\frac{dF_\rightarrow}{dy} = (\kappa + \beta - \beta p_{\rightarrow\rightarrow})F_\rightarrow - \beta(p_{\leftarrow\rightarrow}F_\leftarrow + p_{\downarrow\rightarrow}F_\downarrow + p_{\uparrow\rightarrow}F_\uparrow) \tag{25}$$

En posant ces équations sous forme matricielle $\dot{X} = AX$ et $h = -\kappa - \beta$, on obtient :

$$\begin{pmatrix} \frac{dF_\downarrow}{dz} \\ \frac{dF_\uparrow}{dz} \\ \frac{dF_\leftarrow}{dy} \\ \frac{dF_\rightarrow}{dy} \end{pmatrix} = \begin{pmatrix} h + \beta p_{\downarrow\downarrow} & \beta p_{\uparrow\downarrow} & \beta p_{\leftarrow\downarrow} & \beta p_{\rightarrow\downarrow} \\ -\beta p_{\downarrow\uparrow} & -h - \beta p_{\uparrow\uparrow} & -\beta p_{\leftarrow\uparrow} & -\beta p_{\rightarrow\uparrow} \\ \beta p_{\downarrow\leftarrow} & \beta p_{\uparrow\leftarrow} & h + \beta p_{\leftarrow\leftarrow} & \beta p_{\rightarrow\leftarrow} \\ -\beta p_{\downarrow\rightarrow} & -\beta p_{\uparrow\rightarrow} & -\beta p_{\leftarrow\rightarrow} & -h - \beta p_{\rightarrow\rightarrow} \end{pmatrix} \begin{pmatrix} F_\downarrow \\ F_\uparrow \\ F_\leftarrow \\ F_\rightarrow \end{pmatrix} \tag{26}$$

On remarque que $div(\dot{X}) = tr(A)$ et que $tr(A) = 0$ si et seulement si $p_{\rightarrow\rightarrow} = p_{\leftarrow\leftarrow}$ et $p_{\downarrow\uparrow} = p_{\uparrow\downarrow}$.

On en conclut donc que le volume se conserve seulement dans le cas isotrope, ce qui est en accord avec les résultats précédents.

On généralise au cas avec n dimensions avec $h = -\kappa - \beta$.

$$\begin{pmatrix} \frac{dF_1}{dx_1} \\ \frac{dF_2}{dx_1} \\ \vdots \\ \frac{dF_{2n-1}}{dx_n} \\ \frac{dF_{2n}}{dx_n} \end{pmatrix} = \begin{pmatrix} h + \beta p_{1,1} & \beta p_{2,1} & \dots & \beta p_{2n-1,1} & \beta p_{2n,1} \\ -\beta p_{1,2} & -h - \beta p_{2,2} & \dots & -\beta p_{2n-1,2} & -\beta p_{2n,2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \beta p_{1,2n-1} & \beta p_{2,2n-1} & \dots & h + \beta p_{2n-1,2n-1} & \beta p_{2n,2n-1} \\ -\beta p_{1,2n} & -\beta p_{2,2n} & \dots & -\beta p_{2n-1,2n} & -h - \beta p_{2n,2n} \end{pmatrix} \begin{pmatrix} F_1 \\ F_2 \\ \vdots \\ F_{2n-1} \\ F_{2n} \end{pmatrix} \quad (27)$$

On a toujours $div(\dot{X}) = tr(A)$ et que $tr(A) = 0$ si et seulement si $\forall n$, $p_{2n-1,2n-1} = p_{2n,2n}$. La trace étant nulle, la divergence l'est aussi.

Nous pouvons généraliser qu'à l'ordre N, le volume se conserve si et seulement si le milieu est isotrope.

7 Variation du portrait de phase avec la fréquence

Nous avons vu précédemment que les interactions photons-milieu suivent des lois, elles-mêmes régies par des coefficients d'absorption et de diffusion. Ces coefficients dépendent de la nature du milieu, mais aussi la longueur d'onde du flux lumineux étudié, c'est pourquoi après avoir étudié les cas théoriques, nous allons étudier la variation des flux $F_\downarrow$ et $F_\uparrow$ dans l'eau et dans l'air en fonction de deux longueurs d'ondes : 450nm (bleu) et 675nm (rouge).

7.1 Variation dans l'eau

On remarque que dans l'eau les différentes longueurs d'onde n'auront pas la même intensité pour une profondeur identique (fig : 10a). Cela s'explique par le fait que chaque longueur d'onde possède un libre parcours moyen différent dans l'eau, alors elles ont donc des épaisseurs optiques différentes pour une même distance parcourue. En comparant la figure 10b et 10c on constate que l'allure de la courbe de τ_{final} est très similaire à celle de la courbe de κ. Cela est dû au fait que $\tau = (\beta + \kappa)z$ et comme $\beta << \kappa$ on obtient alors une allure de courbe quasiment identique.

L'interprétation de la figure 10a peut se faire de plusieurs manières à l'aide de la figure 10d. Si l'on regarde simplement $F_\downarrow$ alors lorsque $z = 50$, la longueur d'onde majoritaire est le bleu tandis qu'il reste bien moins de rouge et de vert car leur valeur de $\bar{F}_\downarrow$ est plus petite. Le plongeur représenté sur la figure 10d observera donc bien plus de bleu que d'autres couleurs.

On peut également comprendre de quelle couleur devrait être le flux montant à 50m pour obtenir du blanc à la surface de l'eau $F_\uparrow$. En effet, pour avoir de la lumière blanche en sortie il faut que quand $\tau = 0$, toutes les longueurs d'onde soient à la même intensité. Pour cela il faudra partir avec beaucoup plus de rouge que de bleu (représenté en bas à droite sur la fig : 10d). Ces valeurs se lisent sur l'axe $F_\uparrow$ au point où les courbes se finissent.

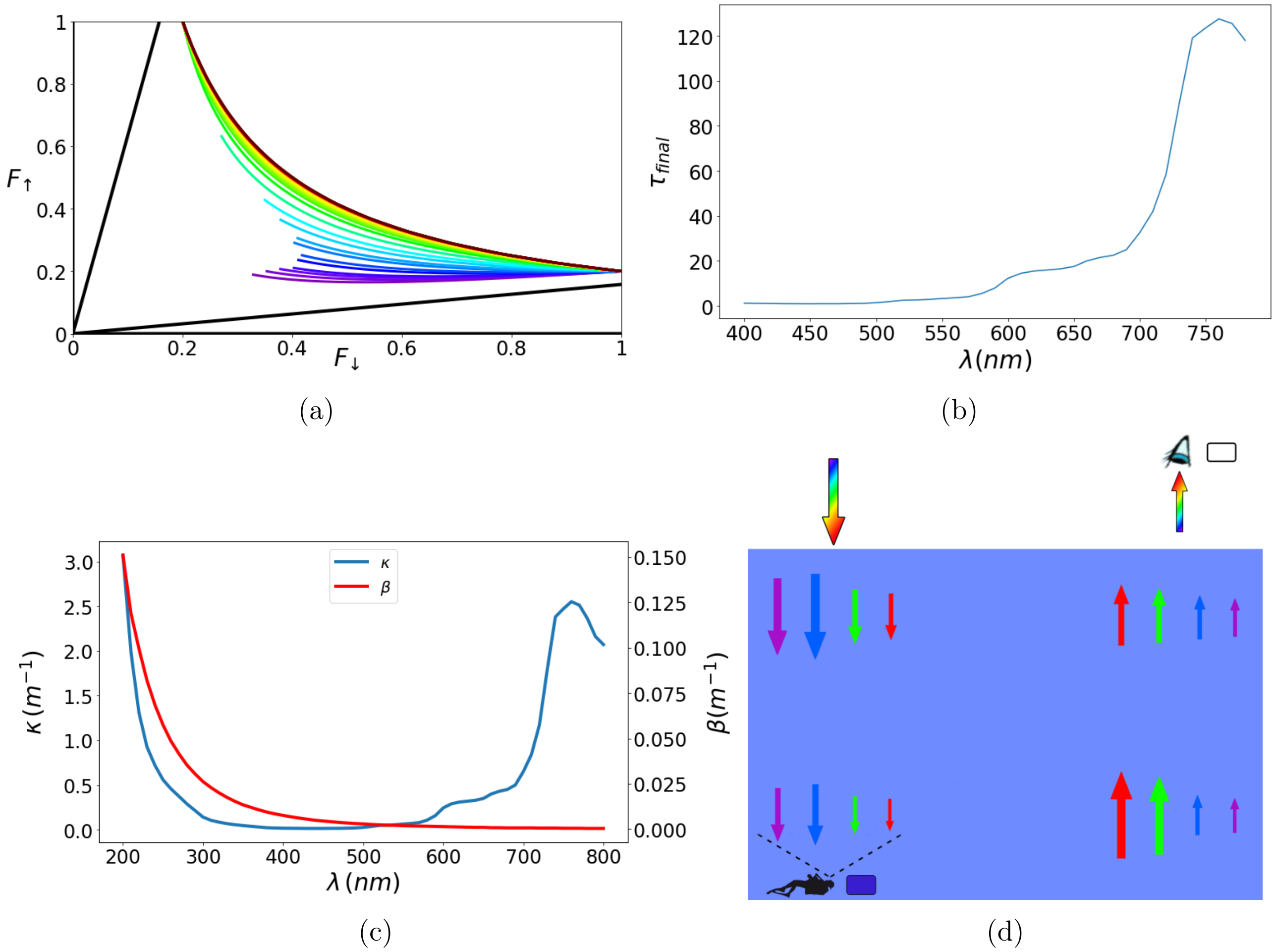


FIGURE 10 – Comportement des différentes longueurs d'onde dans l'eau

Les coefficients d'absorption et de diffusion ont été extraits de l'article Smith, 1981 [4].

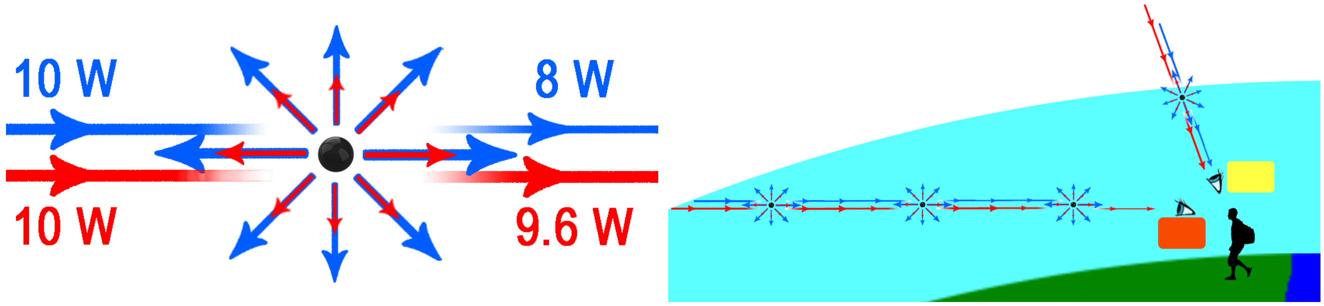
(a) Graphe $F_\uparrow$ en fonction de $F_\downarrow$. Toutes les courbes ont (1 ; 0.2) comme point de départ et sont tracées sur une profondeur de 50m. Les vecteurs propres associés au w de la longueur d'onde 400nm (min) est visible. Les vecteurs propres de la longueur d'onde 780nm (max) sont quasiment confondus avec les axes. Les couleurs des courbes sont issues du calcul des valeurs RGB associées aux longueurs d'onde.

(b) Graphe de la longueur d'onde en fonction du τ final de la figure a.

(c) Variation de β (en rouge) et κ (en bleu) en fonction de la longueur d'onde.

(d) Schéma représentant les fréquences et leurs proportions dans l'eau avec les valeurs de la figure 10a, la taille des flèches est proportionnelle à l'intensité des flux

7.2 Variation dans l'air



(a) (b)

FIGURE 11 – La diffusion de Rayleigh
(a) Schéma explicatif du modèle de diffusion de Rayleigh de flux bleu et rouge, la taille des flèches est proportionnelle à l'intensité des flux.
(b) Schéma de l'impact de la diffusion de Rayleigh dans la couleur perçue du soleil, la taille des flèches est proportionnelle à l'intensité des flux.

Dans l'air c'est la diffusion de Rayleigh [5] qui est le mécanisme prépondérant dans le visible. Le bleu est plus diffusé que le rouge dans toutes les directions de l'espace. Le faisceau bleu perd donc plus de flux que le faisceau rouge dans la direction incidente (voir fig : 11). Cela se traduit par un coefficient d'absorption plus fort pour le bleu que pour le rouge dans le modèle à deux faisceaux. Plus précisément on a :

$$\kappa_{bleu} = \left(\frac{\lambda_{rouge}}{\lambda_{bleu}}\right)^4 * \kappa_{rouge} \quad et \quad \beta_{bleu} = \left(\frac{\lambda_{rouge}}{\lambda_{bleu}}\right)^4 * \beta_{rouge} \quad \Rightarrow \quad w_{rouge} = w_{bleu} \tag{28}$$

Les portraits de phase du bleu et du rouge sont donc les mêmes dans l'air (voire fig : 12).

Par contre l'épaisseur optique du faisceau bleu est plus grande que celle du faisceau rouge. $\bar{\tau}_{bleu} = \left(\frac{\lambda_{rouge}}{\lambda_{bleu}}\right)^4 * \bar{\tau}_{rouge}$. C'est ce que nous observons dans la figure 12. La cas Ⓐ de la figure 12 montre que le flux $F_{\downarrow}$ rouge est supérieur au flux $F_{\downarrow}$ bleu. Au début $F_{\downarrow 0\, bleu} = F_{\downarrow 0\, rouge}$. On retrouve ainsi le fait que le coucher du soleil apparaisse rouge (figure 11b).

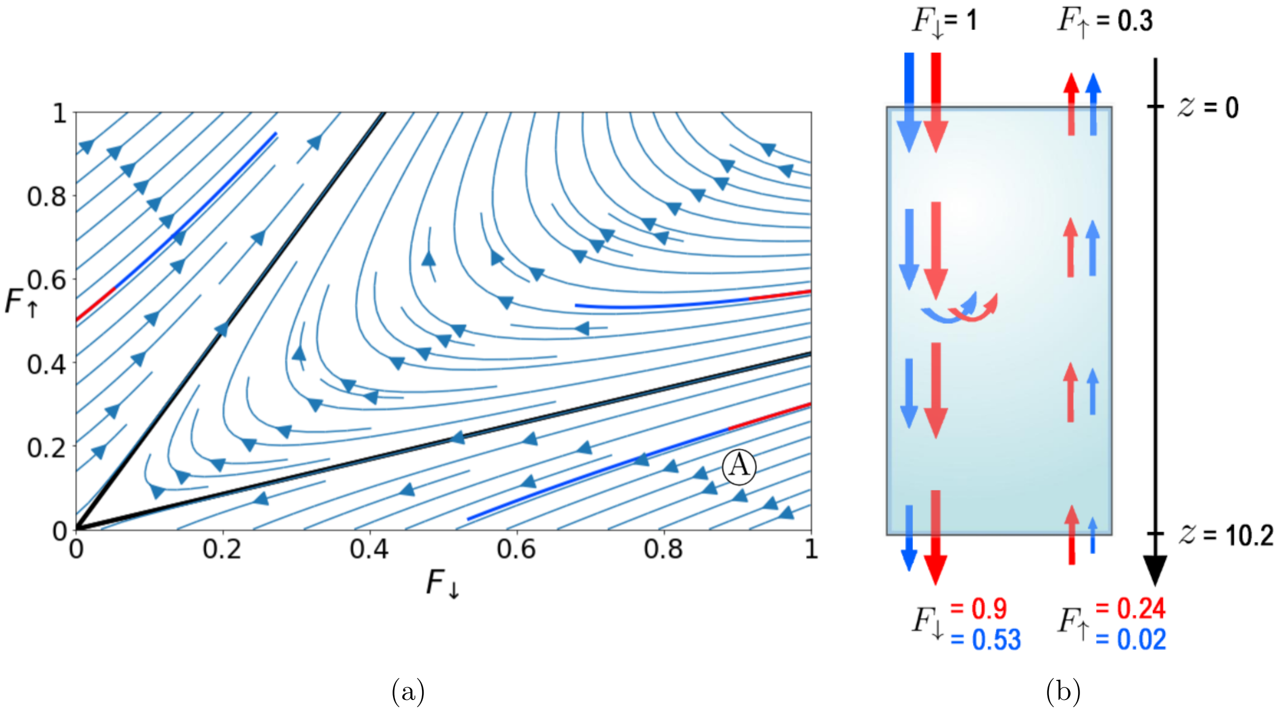
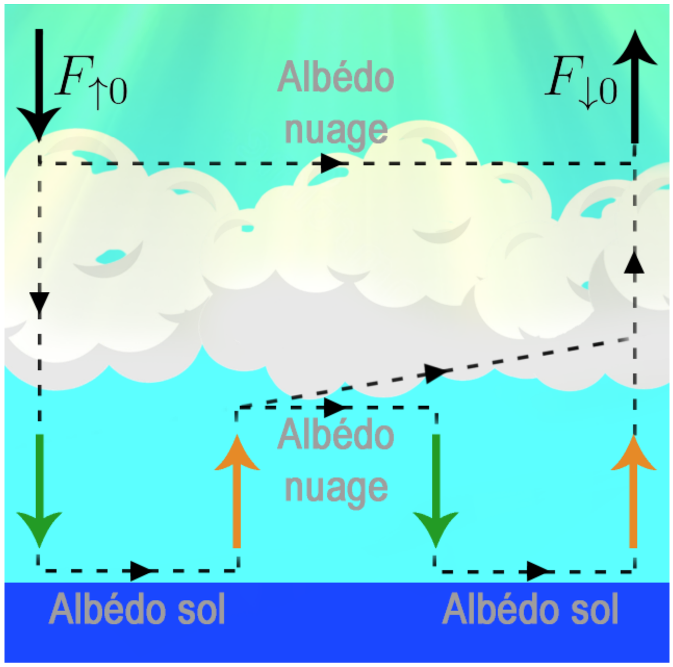


FIGURE 12 – Comportement des faisceaux lumineux dans l'air dans les longueurs d'onde bleus et rouges lorsque $w = 0.83$.
(a) Graphe de $F_\uparrow$ en fonction de $F_\downarrow$ avec $g = 0$ et $w_{rouge} = w_{bleu}$. Nous constatons que τ_{bleu} est 5 fois supérieur à τ_{rouge}. Les vecteurs propres sont confondus et représentés en noir.
(b) Comportement des faisceaux lumineux pour les courbes rouges et bleues nommées Ⓐ, la taille des flèches est proportionnelle à l'intensité du flux lumineux.

À l'aide de la diffusion de Rayleigh, nous pouvons expliquer un phénomène du quotidien : la couleur du soleil en fonction de l'orientation du soleil. La couleur du soleil couchant ou levant paraîtra donc plus rouge car les rayons perçus auront parcouru une plus grande distance dans l'atmosphère car le flux bleu sera plus diffusé (fig : 11b).

8 Étude du transfert radiatif à travers un nuage

Nous allons étudier dans cette partie le transfert radiatif à travers un nuage à l'aide du portrait de phase du système à 2 faisceaux.



$F_{\uparrow \bar{\tau}} = \sum \uparrow \qquad F_{\downarrow \bar{\tau}} = \sum \downarrow$

FIGURE 13 – schéma représentant le système complexe : nuage / surface terrestre

À l'aide de la figure 13 des phénomènes nouveaux sont mis en évidence :

— $F_{\downarrow 0}$ alimente directement le flux montant $F_{\uparrow 0}$ par l'interaction avec la surface du nuage, régie par son albédo.

— Le flux descendant $F_{\downarrow \bar{\tau}}$ nourrit le flux montant $F_{\uparrow \bar{\tau}}$ suite à la réflexion sur le sol, ce flux sera donc proportionnel à l'albédo du sol.

— Une réflexion en chaîne a lieu entre le sol et le nuage (représenté une seule fois sur la figure) ce qui engendre une somme de flux alimentant $F_{\uparrow 0}$.

On peut calculer que

$$F_{\downarrow \bar{\tau}} = F_{\downarrow \bar{\tau}} + A_{sol}A_{nuage}F_{\downarrow \bar{\tau}} + (A_{sol}A_{nuage})^2 F_{\downarrow \bar{\tau}} + \cdots + (A_{sol}A_{nuage})^n F_{\downarrow \bar{\tau}} \tag{29}$$

c'est une suite géométrique que l'on peut réécrire à l'aide de la formule de la somme des termes d'une suite géométrique comme

$$F_{\downarrow \bar{\tau}} = \frac{F_{\downarrow \bar{\tau}}}{1 - A_{sol}A_{nuage}} \tag{30}$$

car $A_{sol}A_{nuage} < 1$ alors quand n tend vers l'infini $(A_{sol}A_{nuage})^n$ tend vers 0. De plus comme $F_{\uparrow \bar{\tau}}$ est obtenu après la réflexion de $F_{\downarrow \bar{\tau}}$ sur le sol :

$$F_{\uparrow \bar{\tau}} = \frac{A_{sol}F_{\downarrow \bar{\tau}}}{1 - A_{sol}A_{nuage}} \tag{31}$$

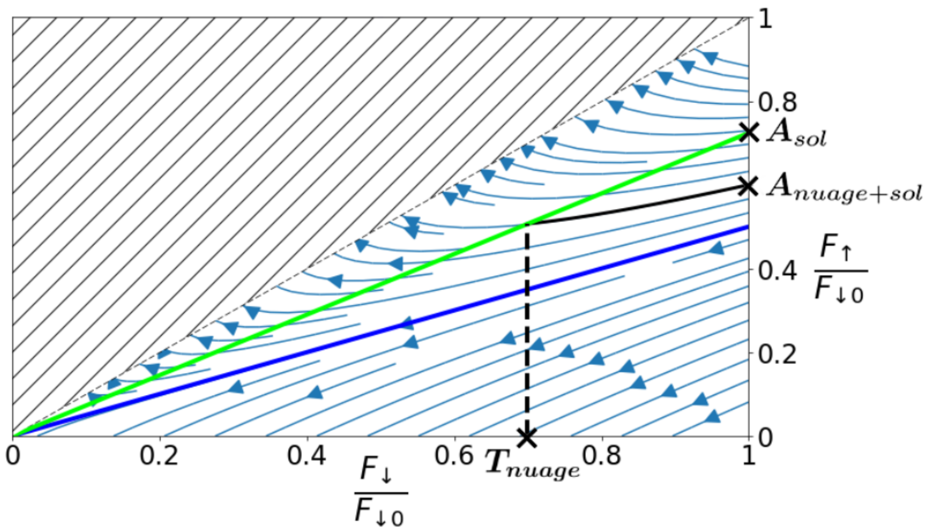


FIGURE 14 – Graphe représentant $\frac{F_\uparrow}{F_{\downarrow 0}}$ en fonction de $\frac{F_\downarrow}{F_{\downarrow 0}}$
Ce Graphe a été obtenu avec $w = 0.9$ et $g = 0.2$. La courbe bleue représente le vecteur propre associé à cet espace de phase. La courbe noire correspond à une solution particulière de ce système. La courbe verte est une droite qui fait l'intersection entre l'origine et la fin de la courbe noire.

Nous pouvons identifier plusieurs, valeurs important sur la figure 14.

— L'intersection entre l'axe des ordonnées et de la courbe noire est égale à l'albédo du nuage et du sol, cette valeur est définie comme :

$$A_{(nuage+sol)} = \frac{F_{\uparrow 0}}{F_{\downarrow 0}} \tag{32}$$

— La transmittance du nuage T_{nuage} peut se lire l'axe des abscisses au point final de la courbe. La transmittance est définie comme l'intensité lumineuse sortant divisée par celle sortant. Elle est donc définie comme

$$T_{nuage} = \frac{F_{\downarrow \bar{\tau}}}{F_{\downarrow 0}} \tag{33}$$

— La trajectoire du faisceau doit se terminer sur la droite d'équation $\bar{F}_{\uparrow \bar{\tau}} = A_{sol} \bar{F}_{\downarrow \bar{\tau}}$ où A_{sol} est l'abedo du sol.A_{sol} se lit donc sur l'axe des ordonnées. C'est l'intersection de la droite verte avec l'axe des ordonnées.

Aucune courbe ne peut se trouver dans la zone hachurée, en effet cela impliquerait que le milieu possède un albédo plus grand que 1, ce qui est physiquement impossible.

9 Conclusion

En conclusion, nous avons pu voir de nombreuses propriétés du modèle à deux faisceaux dans l'espace des phases. Nous avons d'abord vu comment le modèle à deux flux a été établi dans la littérature, puis nous avons calculé les vecteurs et valeurs propres à associer à ce système. À partir de ces valeurs, nous avons pu tracer le portrait de phase des cas simples (simplement avec de la diffusion ou de l'absorption) puis le cas général dans un milieu quelconque. Nous avons ensuite pu démontrer la conservation du volume dans le cas isotrope et que au contraire dans le cas non isotrope, le volume ne se conserve pas. Dans un second temps nous avons déduit les équations du modèle de transfert radiatif aux ordres supérieurs inspiré du modèle à deux faisceaux, avant de nous intéresser aux comportements de différentes longueurs d'onde dans l'eau et dans l'air. Enfin, nous nous sommes intéressés au comportement de la lumière quand un nuage se trouve au-dessus d'un sol réfléchissant.

Nous avons donc une meilleure compréhension des propriétés du portrait de phase dans plusieurs scénarios, mais il en reste plusieurs à explorer. Par exemple l'étude du comportement de la lumière en eau peu profonde où l'on considère que le fond marin (roche, sable, algue) réfléchit la lumière et engendre un flux montant. Cela permettrait d'expliquer les fonds turquoises des eaux caribéennes. Pousser plus loin notre analyse du modèle avec le nuage clarifierait des modèles comme un sol qui réfléchit sélectivement certaines longueurs d'onde comme une prairie, une forêt ou du goudron.

Ce stage a également été une occasion pour nous de découvrir encore plus le monde de la recherche ainsi que de nous familiariser avec les outils de rédaction utilisés en science. En effet, ce rapport a été rédigé sur LaTeX et nous a permis d'avoir une meilleure gestion des figures, des équations et de la numérotation.

Références

[1] C. F. Bohren, Fundamentals of atmospheric radiation. Wiley VCH, 1986.

[2] A. Mitofsky, Direct Energy Conversion. A.T. Still University, 2018. [Online]. Available : https://www.trine.edu/books/documents/de_text1.0.0.pdf

[3] M. Hirsch, S. Smale, and R. Devaney, Differential Equations, Dynamical Systems, and an Introduction to Chaos, 3rd ed. Academic Press, 2013. [Online]. Available : https://www.sciencedirect.com/science/article/pii/B9780123820105000270

[4] R. C. Smith and K. S. Baker, "Optical properties of the clearest natural waters (200–800 nm)," Appl. Opt., vol. 20, no. 2, pp. 177–184, Jan 1981. [Online]. Available : https://opg.optica.org/ao/abstract.cfm?URI=ao-20-2-177

[5] C. R. R. Nave, "Rayleigh scattering," last accessed 7 June 2023. [Online]. Available : http://hyperphysics.phy-astr.gsu.edu/hbase/atmos/blusky.html

A Bilan Personnel

Ce stage nous a permis de débloquer de nouvelles compétences, mais également d'en affiner. En effet, les tâches demandées nous ont obligés à développé et comprendre de nouveaux outils, en l'occurrence en mathématique, physique et informatique avec la découverte dans un premier temps des systèmes dynamiques ainsi que des propriétés matricielles. Dans un second temps, nous avons pu approfondir notre compréhension des interactions entre la lumière et un milieu, et enfin nous avons lu la documentation de fonction informatique que nous avons ensuite détournée pour notre étude.

Concernant la documentation, l'exploitation de cours et d'articles on permit d'en affiné notre utilisation et l'extraction d'information.

La particularité de ce stage a été l'autonomie ainsi que la pédagogie qu'il a fallu développer. Effectivement, notre maître de stage étant de passage de manière quotidienne nous avons développé une autonomie et une organisation permettant une bonne productivité, de plus notre sujet n'étant pas déjà effectué notre grand enjeu a été de donner des explications claires et de difficulté croissante, d'où la présence de nombreux schéma et graphique, pour le rendre le plus accessible possible.

D'un point de vue relationnel, nous avons continué de mieux nous connaître l'un l'autre. En effet, nous avions déjà travaillé ensemble pendant de nombreux mois durant l'APP de L1 et nous avions appris à jouer sur nos forces. Arnaud ayant plus d'aisance avec l'informatique, a rédigé et organisé le programme Python. À l'inverse, Achile étant plus expérimenté avec Photoshop et des logiciels de montage a confectionné les schémas et les ajustements visuels de ce rapport. En revanche, nous débutions tous deux dans l'utilisation de LaTeX, nous avons alors travaillé ensemble pour comprendre le fonctionnement de toutes les commandes, notamment celle des figures et des équations. Lorsque l'on s'approchait d'une impasse, notre superviseur, Cyril DAUPHIN, venait nous guider tout en nous laissant une grande autonomie pour atteindre nos objectifs.

B Code Python

Le Code python utiliser pour produire toutes ces figures peut être retrouvé ci-dessous ou sur le lien ici : `https://github.com/Danura30082/Code-stage-L2`

```python
# -*- coding: utf-8 -*-
"""
Created on Mon May 22 10:43:22 2023
@author: Arnaud Costermans
"""
# Import libraries
import numpy as np
import matplotlib.pyplot as plt
#import matplotlib.patches as ptch
from scipy.integrate import solve_ivp
from wavelen2rgb import wavelen2rgb #importer de
↪ http://www.johnny-lin.com/py_refs/wavelen2rgb.html

#donnée issue de https://doi.org/10.1364/AO.20.000177 (Raymond C. Smith and Karen S. Baker
↪ (1981))
```

```python

Liste_longeur_onde=[200, 210, 220, 230, 240, 250, 260, 270, 280, 290, 300, 310, 320, 330, 340, 350, 360, 370, 380, 390, 400, 410, 420, 430, 440, 450, 460, 470, 480, 490, 500, 510, 520, 530, 540, 550, 560, 570, 580, 590, 600, 610, 620, 630, 640, 650, 660, 670, 680, 690, 700, 710, 720, 730, 740, 750, 760, 770, 780, 790, 800]
Liste_kappa=[3.07, 1.99, 1.31, 0.927, 0.72, 0.559, 0.457, 0.373, 0.288, 0.215, 0.141, 0.105, 0.0844, 0.0678, 0.0561, 0.0463, 0.0379, 0.03, 0.022, 0.0191, 0.0171, 0.0162, 0.0153, 0.0144, 0.0145, 0.0145, 0.0156, 0.0156, 0.0176, 0.0196, 0.0257, 0.0357, 0.0477, 0.0507, 0.0558, 0.0638, 0.0708, 0.0799, 0.108, 0.157, 0.244, 0.289, 0.309, 0.319, 0.329, 0.349, 0.4, 0.43, 0.45, 0.5, 0.65, 0.839, 1.169, 1.799, 2.38, 2.47, 2.55, 2.51, 2.36, 2.16, 2.07]
Liste_beta=[0.151, 0.119, 0.0995, 0.082, 0.0685, 0.0575, 0.0485, 0.0415, 0.0353, 0.0305, 0.0262, 0.0229, 0.02, 0.0175, 0.0153, 0.0134, 0.012, 0.0106, 0.0094, 0.0084, 0.0076, 0.0068, 0.0061, 0.0055, 0.0049, 0.0045, 0.0041, 0.0037, 0.0034, 0.0031, 0.0029, 0.0026, 0.0024, 0.0022, 0.0021, 0.0019, 0.0018, 0.0017, 0.0016, 0.0015, 0.0014, 0.0013, 0.0012, 0.0011, 0.001, 0.001, 0.0008, 0.0008, 0.0007, 0.0007, 0.0007, 0.0007, 0.0006, 0.0006, 0.0006, 0.0005, 0.0005, 0.0005, 0.0004, 0.0004, 0.0004]

def calcul_a_b_Isotrope(w, g):
    """
    Calcule a et b, les coeficients de la matrice dans le cas isotrope

    Parameters
    ----------
    w : float
        valeur compris entre 0 et 1 qui represente l'albedo simple.
    g : float
        valeur compris entre -1 et 1 qui represente le parametre d'asymetrie.

    Returns
    -------
    a : float
        valeur compris entre 0 et 1 qui corespond a des coefficients de la matice.
    b : float
        valeur compris entre 0 et 1 qui corespond a des coefficients de la matice.

    """

    a=(1-w*((1+g)/2))
    b=(w*((1-g)/2))
    return a, b

def calcul_a_b_c_d_non_Isotrope(w,g):
    """
    Calcule a,b,c et d, les coeficient de la matrice dans le cas non-isotrope

    Parameters
    ----------
    w : float
```

```
        valeur compris entre 0 et 1 qui represente l'albedo simple.
    g : float
        valeur compris entre -1 et 1 qui represente le parametre d'asymetrie.


    Returns
    -------
    a : float
        valeur compris entre 0 et 1 qui corespond a des coefficients de la matice.
    b : float
        valeur compris entre 0 et 1 qui corespond a des coefficients de la matice.
    c : float
        valeur compris entre 0 et 1 qui corespond a des coefficients de la matice.
    d : float
        valeur compris entre 0 et 1 qui corespond a des coefficients de la matice.

    """
    e=g[0]
    f=g[1]
    a=(w*e-1)
    b=(w*(1-f))
    c=-w*(1-e)
    d=(-w*f+1)
    return a,b,c,d

def modelflux(tau, Fbas_Fhaut,w,g):
    """
    Permet de calculer Fbas' et Fhaut'. Cette fonction est uniqument appelle par solve_ivp de
    ↪ la librairie scipy.integrate

    Parameters
    ----------
    tau : float
        l'épaiseur optique.
    Fbas_Fhaut : tuple [float,float]
        Vecteur composer de Fbas et de Fhaut.
    w : float
        valeur compris entre 0 et 1 qui represente l'albedo simple.
    g : float or (float,float)
        Soit valeur compris entre -1 et 1 qui represente le parametre d'asymetrie
        ou un Tuple contenant p_bas,bas et p_haut,haut compris entre 0 et 1.

    Returns
    -------
    list : tuple [float,float]
        Vecteur compose de Fbas' et de Fhaut'.

    """
```

```
    Fbas, Fhaut= Fbas_Fhaut
    if type(g)==tuple:
        a,b,c,d=calcul_a_b_c_d_non_Isotrope(w, g)
        return [ a*Fbas+b*Fhaut, c*Fbas+d*Fhaut]
    else:
        a,b = calcul_a_b_Isotrope(w, g)
        return [ -a*Fbas+b*Fhaut, -b*Fbas+a*Fhaut]


def graph_des_phase(w, g, color="#1f77b4"):
    """
    Génère le portrait de phase

    Parameters
    ----------
    w : float
        valeur compris entre 0 et 1 qui represent l'albedo simple.
    g : float or (float,float)
        Soit valeur compris entre -1 et 1 qui represent le parametre d'asymetrie
        ou un Tuple contenant p_bas,bas et p_haut,haut compris entre 0 et 1.
    color: string or (float,float,float)
        un argument couleur de mathplotlib ou une valeur RVB. The default is "black".
    Returns
    -------
    None.

    """
    #creation des parametres de streamplot
    Y, X = np.mgrid[0:1:200j, 0:1:200j]
    if type(g)==tuple:
        a,b,c,d=calcul_a_b_c_d_non_Isotrope(w, g)
        U = a*X+b*Y
        V = c*X+d*Y
    else:
        a,b = calcul_a_b_Isotrope(w, g)
        U = -a*X+b*Y
        V = -b*X+a*Y

    #tracage du digramme des phases
    plt.streamplot(X, Y, U, V, density = 1,arrowsize=3,color=color)

def solution_particuliere (Fbas_init, Fhaut_init, w, g, color="black", resolution=1000,
    tau_min=0, tau_max=20):
    """
    Permet de tracer une solution particuliaire

    Parameters
    ----------
```

```
    Fbas_init : float
        La valeur initiale de Fbas. Compris entre 0 et 1.
    Fhaut_init : float
        La valeur initiale de Fhaut. Compris entre 0 et 1.
    w : float
        valeur compris entre 0 et 1 qui represente l'albedo simple.
    g : float or (float,float)
        Soit valeur compris entre -1 et 1 qui represente le parametre d'asymetrie
        ou un Tuple contenant p_bas,bas et p_haut,haut compris entre 0 et 1.
    color: string or (float,float,float)
        un argument couleur de mathplotlib ou une valeur RVB. The default is "black".
    resolution : int, optional
        Le nombre de point qui seront tracer pour la solution particuliaire. The default is 1000.
    tau_min : int, optional
        La valuer initiale pour la derivation par tau, l'epaiseur optique. The default is 0.
    tau_max : int, optional
        La valuer final pour la derivation par tau, l'epaiseur optique. The default is 20.

    Returns
    -------
    None.

    """



    #calcule de la solution particuliaire
    solution=solve_ivp(modelflux, [tau_min,tau_max], [Fbas_init,Fhaut_init], t_eval=np.linspace(tau_min,tau_max,resolution), args=(w,g))

    #tracage de la solution particuliere
    plt.plot(solution.y[0], solution.y[1], '-',color=color, lw=3)


def point_particulier(Fbas_init, Fhaut_init, w, g, tau=0, resolution=1000, tau_min=0, tau_max=20):
    """

    Permet de tracer un point sur une solution particuliaire
    Parameters
    ----------
    Fbas_init : float
        La valeur initiale de Fbas. Compris entre 0 et 1.
    Fhaut_init : float
        La valeur initiale de Fhaut. Compris entre 0 et 1.
    w : float
        valeur compris entre 0 et 1 qui represente l'albedo simple.
    g : float or (float,float)
```

```
        Soit valeur compris entre -1 et 1 qui represente le parametre d'asymetrie
        ou un Tuple contenant p_bas,bas et p_haut,haut compris entre 0 et 1.
    tau : int, optional
        La valeur de tau, l'epaisseur optique, a laquelle on s'arrete. The default is 0.
    resolution : int, optional
        Le nombre de point qui seront tracer pour la solution particuliaire. The default is 1000.
    tau_min : int, optional
        La valuer initiale pour la derivation par tau, l'epaiseur optique. The default is 0.
    tau_max : int, optional
        La valuer final pour la derivation par tau, l'epaiseur optique. The default is 20.

    Returns
    -------
    solution: OdeResult object of scipy.integrate._ivp.ivp
        Une solution particuliere continue



    """

    #calcule de la solution particuliaire
    solution=solve_ivp(modelflux, [tau_min,tau_max], [Fbas_init,Fhaut_init], dense_output=True, t_eval=np.linspace(tau_min,tau_max,resolution), args=(w,g))
    plt.plot(solution.sol.__call__(tau)[0], solution.sol.__call__(tau)[1], 'k.', ms=20)
    return solution
def vecteur_propre(w,g,typ='r-'):
    """
    Permet de dessiner les vecteur propre sur le graphe

    Parameters
    ----------
    w : float
        valeur compris entre 0 et 1 qui represente l'albedo simple.
    g : float or (float,float)
        Soit valeur compris entre -1 et 1 qui represente le parametre d'asymetrie
        ou un Tuple contenant p_bas,bas et p_haut,haut compris entre 0 et 1.
    typ: string
        Un format String conforme a la documentation Mathplotlib. The default is 'r-'.

    Returns
    -------
    None.

    """
    if type(g)==tuple:
        a,b,c,d=calcul_a_b_c_d_non_Isotrope(w, g)
        e=g[0]
        f=g[1]
```

```
    else:
        a,b = calcul_a_b_Isotrope(w, g)
        e=(g+1)/2
        f=e
    if w!=0:# empeche la division pas 0
        x_v1 = [0, -(2 - e*w - f*w + np.sqrt(4 - 4*e*w - 4*f*w - 4*(w**2) + 4*e*(w**2) + (e**2)*(w**2) + 4*f*(w**2) - 2*e*f*(w**2) + (f**2)*(w**2)))/(2*(-1 + e)*w)]
        y_v1 = [0, 1]
        x_v2 = [0,-(2 - e*w - f*w - np.sqrt(4 - 4*e*w - 4*f*w - 4*(w**2) + 4*e*(w**2) + (e**2)*(w**2) + 4*f*(w**2) - 2*e*f*(w**2) + (f**2)*(w**2)))/(2*(-1 + e)*w)]
        y_v2 = [0, 1]
    elif type(g)!=tuple:
        alpha = np.sqrt(a**2 - b**2)
        x_v1 = [0, b*3]
        y_v1 = [0, (a+alpha)*3]
        y_v2=x_v1
        x_v2=y_v1
    else:
        print("ERROR: w=0 et cas non isotrope pas supporté")
        return
    plt.plot(x_v1,y_v1,typ,lw=4)
    plt.plot(x_v2,y_v2,typ,lw=4)


def mise_en_forme():
    """
    Permet de mettre en forme le graphique

    Returns
    -------
    None.

    """

    #parametrage des axes
    label=["0","0.2","0.4","0.6","0.8","1"]
    plt.tick_params(labelsize=24)
    plt.xlim(0,1)
    plt.ylim(0,1)
    plt.ylabel(r'$F_{\!\uparrow}$', fontsize=30,  rotation='horizontal',labelpad=20.0 ,y=0.45)
    plt.xlabel(r'$F_{\!\downarrow}$', fontsize=30,labelpad=-15.0)
    plt.xticks(np.linspace(0,1,6),label)
    plt.yticks(np.linspace(0,1,6),label)

def arc_en_ceil(Fbas_init, Fhaut_init,g,longeur_onde,tau_max=20):
    """
    Permet de tracer une solution particulier pour un w corespondant a une certain longeru d'onde.
```

```

    Parameters
    ----------
    Fbas_init : float
        La valeur initiale de Fbas. Compris entre 0 et 1.
    Fhaut_init : float
        La valeur initiale de Fhaut. Compris entre 0 et 1.
    g : float or (float,float)
        Soit valeur compris entre -1 et 1 qui represent le parametre d'asymetrie
        ou un Tuple contenant p_bas,bas et p_haut,haut compris entre 0 et 1.
    longeur_onde : int
        La longeur d'onde a tracer avec son w respectif.

    Returns
    -------
    None.

    """

    if not(longeur_onde<380 or 780<longeur_onde):
        lamda=Liste_longeur_onde.index(longeur_onde)
        w=Liste_beta[lamda]/(Liste_beta[lamda]+Liste_kappa[lamda])
        solution_particuliere(Fbas_init, Fhaut_init, w, g, color=tuple([float(x/100) for x in
        ↪  wavelen2rgb(Liste_longeur_onde[lamda])]),tau_max=tau_max)

def point_final (Fbas_init, Fhaut_init, w, g, seuil_max=1, seuil_min=0, resolution=1000,
↪  tau_min=0, tau_max=20):
    """
    Cette fonction permet de trouver le point sur une solution particuliere qui dépasse un
↪  certain seuil de Fhaut

    Parameters
    ----------
    Fbas_init : float
        La valeur initiale de Fbas. Compris entre 0 et 1.
    Fhaut_init : float
        La valeur initiale de Fhaut. Compris entre 0 et 1.
    w : float
        valeur compris entre 0 et 1 qui represente l'albedo simple.
    g : float or (float,float)
        Soit valeur compris entre -1 et 1 qui represente le parametre d'asymetrie
        ou un Tuple contenant p_bas,bas et p_haut,haut compris entre 0 et 1.
    seuil_max : float, optional
        Le seuil supérieur au-delà duquel on arrête la solution. The default is 1.
    seuil_min : float, optional
        Le seuil inférieur en dessous duquel on arrête la solution. The default is 0.
    resolution : int, optional
```

```python
        Le nombre de point qui seront tracer pour la solution particuliaire. The default is 1000.
    tau_min : int, optional
        La valuer initiale pour la derivation par tau, l'epaiseur optique. The default is 0.
    tau_max : int, optional
        La valuer final pour la derivation par tau, l'epaiseur optique. The default is 20.

    Returns
    -------
    tau : float
        La valeur de tau au seuil où on arrête la solution.
    Fbas : float
        La valeur de Fbas au seuil où on arrête la solution.
    Fhaut : float
        La valeur de Fhaut au seuil où on arrête la solution.

    """


    #calcule de la solution particuliaire
    solution=solve_ivp(modelflux, [tau_min,tau_max], [Fbas_init,Fhaut_init], dense_output=True, t_eval=np.linspace(tau_min,tau_max,resolution), args=(w,g))
    for loop in range (len(solution.t)):
        if solution.y[1][loop]>seuil_max:
            if abs(seuil_max-solution.y[1][loop])<abs(seuil_max-solution.y[1][loop-1]):
                endpoint=loop
                break
            else:
                endpoint=loop-1
                break
        elif solution.y[1][loop]<seuil_min:
            if abs(seuil_min-solution.y[1][loop])<abs(seuil_min-solution.y[1][loop-1]):
                endpoint=loop
                break
            else:
                endpoint=loop-1
                break
        endpoint=loop
    Fbas=solution.y[0][endpoint]
    Fhaut=solution.y[1][endpoint]
    tau=solution.t[endpoint]
    return tau, Fbas , Fhaut

# %% w=0 Figure 4

w, g, Fbas_init, Fhaut_init=0, 0, 1, 0.2 #definition des parametres de cette figure
fig = plt.figure(figsize = (12, 7))  #creation du fond de la figure
mise_en_forme()   #mise en forme des axes et des legendes
```

```python
graph_des_phase(w, g)    #Crée le portrait de phase
solution_particuliere(Fbas_init, Fhaut_init, w, g)  #permet de tracer une solution particuliere qui part de Fbas_init, Fhaut_init
vecteur_propre(w, g) #permet de tracer les vecteurs propre de la matrice
tau_final, Fbas_final, Fhaut_final = point_final(Fbas_init, Fhaut_init, w, g) #on recupére les cordonnés du point final
point_particulier(Fbas_init, Fhaut_init, w, g, tau_final) #on trace le point final

#on crée la legende du graphe
legende="Avec w=" + str(w) + " et g=" + str(g) +". Pour " + r'$\tau=$' + str(round(tau_final,2)) + ", " + r'$F_{\!\downarrow}=$' + str(round(Fbas_final,1)) + " et " + r'$F_{\!\uparrow}=$'+str(round(Fhaut_final,1))
plt.text(.5, -0.25, legende, fontsize=24, color='black', ha='center')
plt.show()

# %% w=1 Figure 5

w, g, Fbas_init, Fhaut_init=1, 0, 1, 0.5  #definition des parametres de cette figure
fig = plt.figure(figsize = (12, 7)) #creation du fond de la figure
mise_en_forme()   #mise en forme des axe et des legende
graph_des_phase(w, g)    #Crée le portrait de phase
solution_particuliere(Fbas_init, Fhaut_init, w, g) #permet de tracer une solution particulier qui part de Fbas_init, Fhaut_init
vecteur_propre(w, g) #permet de tracer les vecteur propre de la matrice
tau_final, Fbas_final, Fhaut_final = point_final(Fbas_init, Fhaut_init, w, g) #on recupére les cordonné du point final
point_particulier(Fbas_init, Fhaut_init, w, g, tau_final) #on trace le point final

#on crée la legende du graphe
legende="Avec w=" + str(w) + " et g=" + str(g) +". Pour " + r'$\tau=$' + str(round(tau_final,2)) + ", " + r'$F_{\!\downarrow}=$' + str(round(Fbas_final,1)) + " et " + r'$F_{\!\uparrow}=$'+str(round(Fhaut_final,1))
plt.text(.5, -0.25, legende, fontsize=24, color='black', ha='center')
plt.show()

# %% w=0.7 Figure 6

w, g, Fbas_init, Fhaut_init=0.7, 0.2, 1, 0.5  #definition des parametres de cette figure
fig = plt.figure(figsize = (12, 7)) #creation du fond de la figure
mise_en_forme()   #mise en forme des axes et des legendes
graph_des_phase(w, g)    #Crée le portrait de phase
solution_particuliere(Fbas_init, Fhaut_init, w, g) #permet de tracer une solution particuliere qui part de Fbas_init, Fhaut_init
vecteur_propre(w, g) #permet de tracer les vecteurs propre de la matrice
tau, Fbas_final, Fhaut_final = point_final(Fbas_init, Fhaut_init, w, g, tau_max=1.25 ) #on recupére les cordonnés du point qui correspond au schema
point_particulier(Fbas_init, Fhaut_init, w, g, tau) #on trace le point qui correspond au schema

```

```python
#on crée la legende du graphe
legende="Avec w=" + str(w) + " et g=" + str(g) +". Pour " + r'$\tau=$' + str(round(tau_final,2)) + ", " + r'$F_{\!\downarrow}=$' + str(round(Fbas_final,2)) + " et " + r'$F_{\!\uparrow}=$'+str(round(Fhaut_final,2))
plt.text(.5, -0.25, legende, fontsize=24, color='black', ha='center')
plt.show()

# %% non-iso Figure 7

w,g=0.7,(0.3,0.9)  #definition des parametres de cette figure
fig = plt.figure(figsize = (12, 7)) #creation du fond de la figure
mise_en_forme()   #mise en forme des axes et des legendes
graph_des_phase(w, g)    #Crée le portrait de phase
solution_particuliere(1, 0.6, w, g) #permet de tracer une solution particuliere qui part de Fbas_init, Fhaut_init
vecteur_propre(w, g) #permet de tracer les vecteur propre de la matrice

#on crée la legende du graphe
legende="Avec w=" + str(w) + " et pour " + r'$p_{\!\downarrow\!\downarrow}=$' + str(g[0]) +", "+r'$p_{\!\uparrow\!\uparrow}=$'+str(g[1])
plt.text(.5, -0.25, legende, fontsize=24, color='black', ha='center')
plt.show()

# %% conservation du volume Figure 8

w, g, Fbas_init, Fhaut_init=0.7, 0.2, 0.75, 0.4  #definition des parametres de cette figure
fig = plt.figure(figsize = (12, 7)) #creation du fond de la figure
mise_en_forme()   #mise en forme des axes et des legendes
graph_des_phase(w, g)    #Crée le portrait de phase
vecteur_propre(w, g) #permet de tracer les vecteur propre de la matrice

#on trace les 8 points a tau=0
for loop in range (3):
    for loop1 in range (3):
        if loop!=1 or loop1!=1: #on empeche le tracage du point central
            point_particulier(Fbas_init+loop1/10, Fhaut_init+loop/10, w, g)

#on trace les 8 point a tau=1.25
for loop in range (3):
    for loop1 in range (3):
        if loop!=1 or loop1!=1: #on empeche le tracage du point central
            point_particulier(Fbas_init+loop1/10, Fhaut_init+loop/10, w, g,1.25)

#on crée la legende du graphe
legende="Avec w=" + str(w) + " et g=" + str(g)
plt.text(.5, -0.25, legende, fontsize=24, color='black', ha='center')
plt.show()
```

```python
# %% non-conservation du volume Figure 9

w, g, Fbas_init, Fhaut_init=0.7, (0.3,0.9), 0.75, 0.45  #definition des parametres de cette figure
fig = plt.figure(figsize = (12, 7)) #creation du fond de la figure
mise_en_forme()   #mise en forme des axes et des legendes
graph_des_phase(w, g)    #Crée le portrait de phase
vecteur_propre(w, g) #permet de tracer les vecteurs propre de la matrice

#on trace les 8 point a tau=0
for loop in range (3):
    for loop1 in range (3):
        if loop!=1 or loop1!=1:  #on empeche le tracage du point central
            point_particulier(Fbas_init+loop1/10, Fhaut_init+loop/10, w, g)


#on trace les 8 point a tau=2
for loop in range (3):
    for loop1 in range (3):
        if loop!=1 or loop1!=1:  #on empeche le tracage du point central
            point_particulier(Fbas_init+loop1/10, Fhaut_init+loop/10, w, g,2)

#on crée la legende du graphe
legende="Avec w=" + str(w) + " et pour " + r'$p_{\!\downarrow\!\downarrow}=$' + str(g[0]) +", "+r'$p_{\!\uparrow\!\uparrow}=$'+str(g[1])
plt.text(.5, -0.25, legende, fontsize=24, color='black', ha='center')
plt.show()


# %% longeur d'onde aire figure 11
w,g,kappa_450,beta_450,tau=0,0 ,0.02,0.1,0.25  #definition des parametre de cette figure
kappa_675=kappa_450/5
beta_675=beta_450/5
w_450=beta_450/(beta_450+kappa_450)
w_675=beta_675/(beta_675+kappa_675)


fig = plt.figure(figsize = (12, 7)) #creation du fond de la figure
mise_en_forme()   #mise en forme des axes et des legendes

vecteur_propre(w_450, g,typ="black") #permet de tracer les vecteurs propre de la matrice
graph_des_phase(w_450, g, "#1f77b4")

#on trace les 6 solutiosn particuliere avec la bonne couleur
solution_particuliere(1, 0.30, w_450, g, color=tuple([float(x/100) for x in wavelen2rgb(Liste_longeur_onde[Liste_longeur_onde.index(450)])]),tau_max=5*tau)
solution_particuliere(1, 0.30, w_675, g, color=tuple([float(x/100) for x in wavelen2rgb(Liste_longeur_onde[Liste_longeur_onde.index(670)])]),tau_max=tau)
```

```python
solution_particuliere(1, 0.57, w_450, g, color=tuple([float(x/100) for x in wavelen2rgb(Liste_longeur_onde[Liste_longeur_onde.index(450)])]),tau_max=5*tau)
solution_particuliere(1, 0.57, w_675, g, color=tuple([float(x/100) for x in wavelen2rgb(Liste_longeur_onde[Liste_longeur_onde.index(670)])]),tau_max=tau)
solution_particuliere(0, 0.50, w_450, g, color=tuple([float(x/100) for x in wavelen2rgb(Liste_longeur_onde[Liste_longeur_onde.index(450)])]),tau_max=5*tau)
solution_particuliere(0, 0.50, w_675, g, color=tuple([float(x/100) for x in wavelen2rgb(Liste_longeur_onde[Liste_longeur_onde.index(670)])]),tau_max=tau)


# %% nuage Figure 14

w, g ,tau,Fbas_init, Fhaut_init= 0.8,-1, 0.75,1,0.6  #definition des parametres de cette figure
fig = plt.figure(figsize = (12, 7)) #creation du fond de la figure
graph_des_phase(w, g, "#1f77b4")
mise_en_forme()   #mise en forme des axes et des legendes

#on hachure sur la partie du graphe interdite
plt.fill_between([0,1], [0,1],[1,1],facecolor="w", hatch="/", edgecolor="k", linewidth=0.0,zorder=100)
plt.plot([0,1],[0,1],'k--')
plt.plot([0,0,1],[0,1,1],'k-',zorder=101)


vecteur_propre(w, g,'b') #permet de tracer les vecteurs propre de la matrice en noir
solution_particuliere(Fbas_init, Fhaut_init, w, g,tau_max=tau) #on trace notre solution particuliere avec
tau,albedoX,albedoY=point_final(Fbas_init, Fhaut_init, w, g,tau_max=tau) #on recupere les cordones du point final
plt.plot([0,albedoX*3],[0,albedoY*3],color=(0,1,0),lw=4) #on trace la droite qui passe par l'originer et le point final
plt.plot([albedoX,albedoX],[0,albedoY],'k--',lw=3) #on trace la ligne pointilleer qui relie le point final a l'absice

#on change la mise en forme des axe
plt.ylabel(r'$\frac{F_{\!\uparrow}}{F_{\!\downarrow\!0}}$', fontsize=40, rotation='horizontal',labelpad=20.0 ,y=0.6)
plt.xlabel(r'$\frac{F_{\!\downarrow}}{F_{\!\downarrow\!0}}$', fontsize=40,labelpad=-15.0)
ax=plt.gca()
ax.yaxis.tick_right()
ax.yaxis.set_label_position("right")

#plt.text()

# montrer la figure
plt.show()

# %% Beta Kappa Figure 10
```

```python

fig = plt.figure(figsize = (12, 7)) #creation du fond de la figure

#on met en forme les axes
plt.tick_params(labelsize=24)
plt.ylabel(r'$\kappa\,(m^{-1})$', fontsize=30,  labelpad=20.0)
plt.xlabel(r'$\lambda\,(nm)$', fontsize=30)

ax1=plt.gca()
plt.tick_params(labelsize=24)
ax2 = ax1.twinx() #on crée le deuxime axe des ordonnées
plt.ylabel(r'$\beta (m^{-1})$', fontsize=30,  labelpad=20.0)
plt.tick_params(labelsize=24)

#on crée les courbes de beta et de kappa
kappa,=ax1.plot(Liste_longeur_onde,Liste_kappa,lw=4)
beta,=ax2.plot(Liste_longeur_onde,Liste_beta,'r',lw=4)

#on cree la legende
ax1.legend([kappa,beta],[r'$\kappa$',r'$\beta $'],fontsize="20",loc=9)
plt.show()

# %% arc en ciel eau z fixé Figure 10

w, g, z= 0,-1, 50  #definition des parametres de cette figure
fig = plt.figure(figsize = (12, 7)) #creation du fond de la figure
mise_en_forme()   #mise en forme des axes et des legendes
#in initialiser les listes qui permettront de tracer les tau finaux
tau_final=[]
Longeur_onde_trace=[]
# on iter a traver la liste en tracant que les longeurs d'onde qui on une couleur corsspondant
for loop in range (len(Liste_longeur_onde)):
    if not(Liste_longeur_onde[loop]<400 or 780<Liste_longeur_onde[loop]):
        lamda=Liste_longeur_onde.index(Liste_longeur_onde[loop])
        tau=z*(Liste_beta[lamda]+Liste_kappa[lamda]) #on calcule le tau associe a la longeur z
        ↪ pour cette longeur d'onde
        arc_en_ceil(1, 0.2, g, Liste_longeur_onde[loop],tau)
        tau_final.append(tau)
        Longeur_onde_trace.append(Liste_longeur_onde[loop])

#on trace le vecteur propre associe a la plus petit longeur d'onde
vecteur_propre(Liste_beta[Liste_longeur_onde.index(400)]/(Liste_beta[Liste_longeur_onde.index(400)]
↪ + Liste_kappa[Liste_longeur_onde.index(400)]), g,'k-')

# creation de la figure des tau finaux dans l'eau
fig = plt.figure(figsize = (12, 7)) #creation du fond de la figure
plt.plot(Longeur_onde_trace,tau_final)
plt.ylabel(r'$\tau_{final}}$', fontsize=30)
```

```python
plt.xlabel(r'$\lambda (nm)$', fontsize=30)
plt.tick_params(labelsize=24)
```

LaTeX