

Projet de Modélisation Numérique

# Simulation de l'apparence d'un disque d'accrétion autour d'un trou noir

Guillaume Blanc

Thierry Girard

Sous la responsabilité de **Didier Pelat**

DEA Astrophysique et Techniques Spatiales - 1997-1998



# Table des matières

|          |                                                       |           |
|----------|-------------------------------------------------------|-----------|
| <b>1</b> | <b>Les géodésiques</b>                                | <b>7</b>  |
| 1.1      | L'équation des géodésiques . . . . .                  | 7         |
| 1.2      | Paramètre d'impact critique . . . . .                 | 7         |
| 1.3      | Calcul des géodésiques . . . . .                      | 8         |
| 1.3.1    | Programmation de la trajectoire d'un photon . . . . . | 9         |
| 1.3.2    | Les géodésiques obtenues . . . . .                    | 10        |
| 1.3.3    | L'angle de déflexion . . . . .                        | 14        |
| <b>2</b> | <b>Image d'un trou noir "habillé"</b>                 | <b>15</b> |
| 2.1      | Tracé des iso-rayons . . . . .                        | 15        |
| 2.1.1    | Méthode . . . . .                                     | 15        |
| 2.1.2    | Résultats obtenus . . . . .                           | 17        |
| 2.2      | Calcul du redshift . . . . .                          | 18        |
| 2.2.1    | Un peu de physique . . . . .                          | 18        |
| 2.2.2    | Courbes iso-redshift obtenues . . . . .               | 19        |
| <b>3</b> | <b>Image bolométrique</b>                             | <b>21</b> |
| 3.1      | Aspects qualitatifs . . . . .                         | 21        |
| 3.2      | Tracé de l'image correspondante . . . . .             | 23        |
| <b>4</b> | <b>La réponse spectrale du disque</b>                 | <b>25</b> |
| 4.1      | Comment ça marche? . . . . .                          | 25        |
| 4.2      | Calcul du spectre associé . . . . .                   | 25        |
| 4.3      | Le spectre . . . . .                                  | 26        |
| <b>A</b> | <b>Equation des géodésiques</b>                       | <b>33</b> |
| <b>B</b> | <b>Listing du programme</b>                           | <b>35</b> |
| B.1      | Module runge_kutta . . . . .                          | 35        |
| B.2      | Module geodesique . . . . .                           | 37        |
| B.3      | Module photo . . . . .                                | 39        |
| B.4      | Module spectre . . . . .                              | 48        |
| B.5      | Modules constantes et precision . . . . .             | 54        |

|     |                            |    |
|-----|----------------------------|----|
| B.6 | Module graphique . . . . . | 55 |
| B.7 | Module ppal . . . . .      | 56 |

# Introduction

Notre projet informatique de méthodes numériques a consisté à calculer l'apparence d'un disque d'accrétion autour d'un trou noir, tel que le verrait un observateur situé à l'infini.

Pour ce faire, nous avons, dans un premier temps, calculé la trajectoire d'un photon autour du trou noir (métrique de Schwarzschild) en fonction du paramètre d'impact du photon (chapitre 1).

Ensuite, nous avons construit la représentation d'un disque d'accrétion autour du trou noir en calculant pour chaque pixel de l'image (associé à un paramètre d'impact) la trajectoire du photon et son intersection avec le disque d'accrétion (chapitre 2).

Enfin, nous avons reconstruit le spectre d'émission global du disque d'accrétion, en se basant sur l'hypothèse que le disque émet dans une raie infiniment fine de fréquence  $\nu_0$  identique en tout point (chapitre 4).



# Chapitre 1

## Les géodésiques

### 1.1 L'équation des géodésiques

Nous allons étudier l'environnement extérieur d'un trou noir sphérique, statique, sans rotation, sans charge électrique. Pour ce faire, nous utiliserons la métrique de Schwarzschild, qui décrit la géométrie de l'espace-temps extérieur.

En coordonnées sphériques  $(r, \theta, \phi)$ , elle s'écrit :

$$ds^2 = -\left(1 - \frac{2M}{r}\right) dt^2 + \left(1 - \frac{2M}{r}\right)^{-1} dr^2 + r^2 (d\theta^2 + \sin^2\theta d\phi^2) \quad (1.1)$$

avec  $G = c = 1$ .  $M$  est la masse relativiste du trou noir (elle a la dimension d'une longueur) : l'horizon est situé au rayon de Schwarzschild :  $r_s = 2M$ .

On trouvera dans l'annexe A le passage des composantes du tenseur métrique à l'équation de la trajectoire d'un photon dans cette métrique, en coordonnées polaires dans le plan équatorial du trou noir :

$$\left(\frac{1}{r^2} \frac{dr}{d\phi}\right)^2 + \frac{1}{r^2} \left(1 - \frac{2M}{r}\right) = \left(\frac{cte_1}{cte_2}\right)^2 = \frac{1}{b^2} \quad (1.2)$$

dans laquelle  $b$  est le paramètre d'impact.

### 1.2 Paramètre d'impact critique

En fait, le deuxième terme du membre de gauche de l'équation 1.2 définit une sorte de potentiel effectif  $V(r)$  pour le photon (voir [2] p.673) :

$$V^{-2}(r) = \frac{1}{r^2} \left(1 - \frac{2M}{r}\right) \quad (1.3)$$

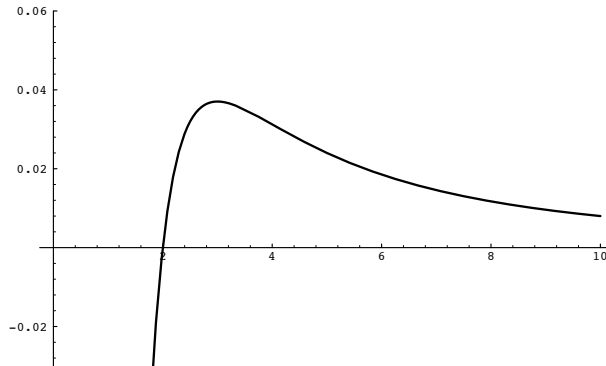


FIG. 1.1: *Potentiel effectif du photon,  $V^{-2}$  en fonction de  $r/M$  ; son maximum apparaît pour  $r = 3M$ , il vaut  $1/(27M^2)$ .*

Pour qu'un rayon avec un paramètre d'impact  $b$  atteigne un point à la distance  $r$ , il faut que  $dr/d\phi$  soit réel, donc  $V^{-2}(r) < b^{-2}$ .

Or la fonction  $V^{-2}(r)$  a son maximum pour  $r_c = 3M$ , où sa valeur est  $V_c^{-2} = 1/(27M^2)$  (voir figure 1.1). Il existe donc un paramètre d'impact critique,  $b_c = 3\sqrt{3}M$  tel que pour  $b < b_c$  les rayons sont capturés par le trou noir, et pour  $b > b_c$ , ils sont défléchis mais pas capturés.

### 1.3 Calcul des géodésiques

On simplifie un peu l'équation des géodésiques 1.2 en posant  $u = 1/r$ , ce qui donne :

$$\left(\frac{du}{d\phi}\right)^2 = 2Mu^3 - u^2 + \frac{1}{b^2} \equiv G(u) \quad (1.4)$$

### 1.3.1 Programmation de la trajectoire d'un photon

La figure 1.2 montre les notations et le repère utilisés.

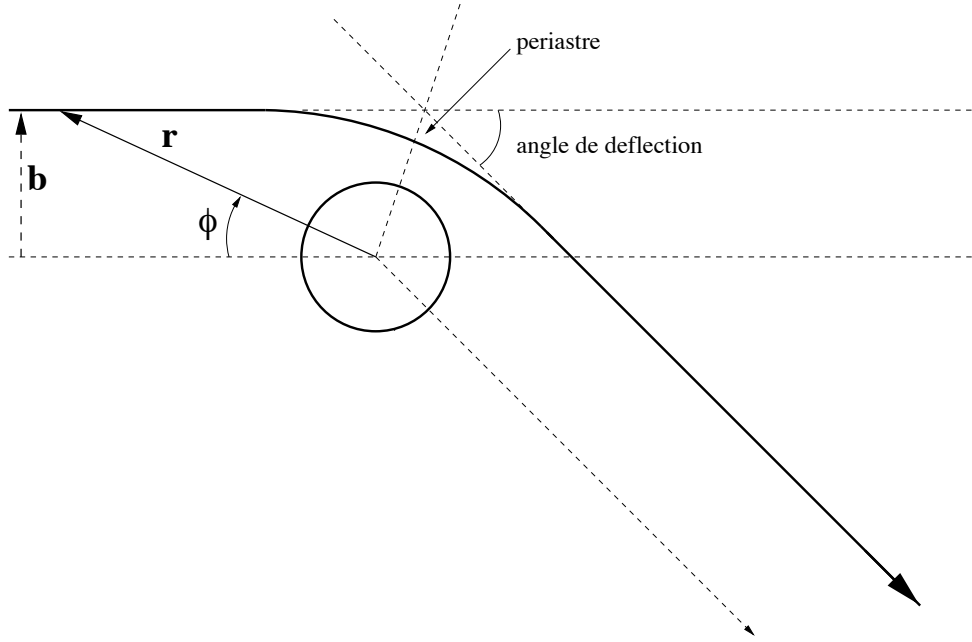


FIG. 1.2: *Trajectoire typique d'un faisceau lumineux.*

Nous avons programmé la résolution de l'équation différentielle par la méthode de Runge-Kutta, à l'ordre 4, à pas constant. Pour cela nous avons utilisé la subroutine correspondante fournie par [3], légèrement adaptée en fortran 90 et en double précision.

Cependant, pour le cas qui nous intéresse, c'est-à-dire  $b > 3\sqrt{3}M$  (par la suite, on ne prendra en compte que les photons qui tombent sur le disque d'accrétion, et qui n'ont donc pas été préalablement avalés par le trou noir),  $du/d\phi$  change de signe quand on passe au périastre. Même si la trajectoire est symétrique par rapport au périastre, il devient difficile de localiser celui-ci numériquement. Il est plus astucieux de dériver 1.4 afin de s'affranchir du carré gênant. On obtient ainsi l'équation du second ordre suivante :

$$\frac{d^2u}{d\phi^2} = 3Mu^2 - u \quad (1.5)$$

que l'on peut transformer en un système de deux équations du premier ordre :

$$\begin{aligned} \frac{dv}{d\phi} &= 3Mu^2 - u \\ \frac{du}{d\phi} &= v \end{aligned} \quad (1.6)$$

système aisément intégrable par la subroutine Runge-Kutta, et qui ne présente pas de singularité.

### 1.3.2 Les géodésiques obtenues

Les figures 1.3 à 1.6 montrent les trajectoires obtenues, pour diverses valeurs du paramètre d'impact  $b$ , avec en ordonnée  $u = 1/r$  et  $r$  respectivement.

La figure 1.3 montre la trajectoire d'un photon dont le paramètre d'impact est inférieur à  $b_c$  : le malheureux va se perdre sur le trou noir. En revanche, sur la figure 1.4 on le voit faire plusieurs tours ( $\sim 3$ ) avant de s'abîmer... Quand  $b > b_c$ , le photon va se perdre dans l'espace.

Le dernier cas (figure 1.6) est celui où le photon ne voit pas le trou noir : il voyage en géométrie euclidienne (en ligne droite).

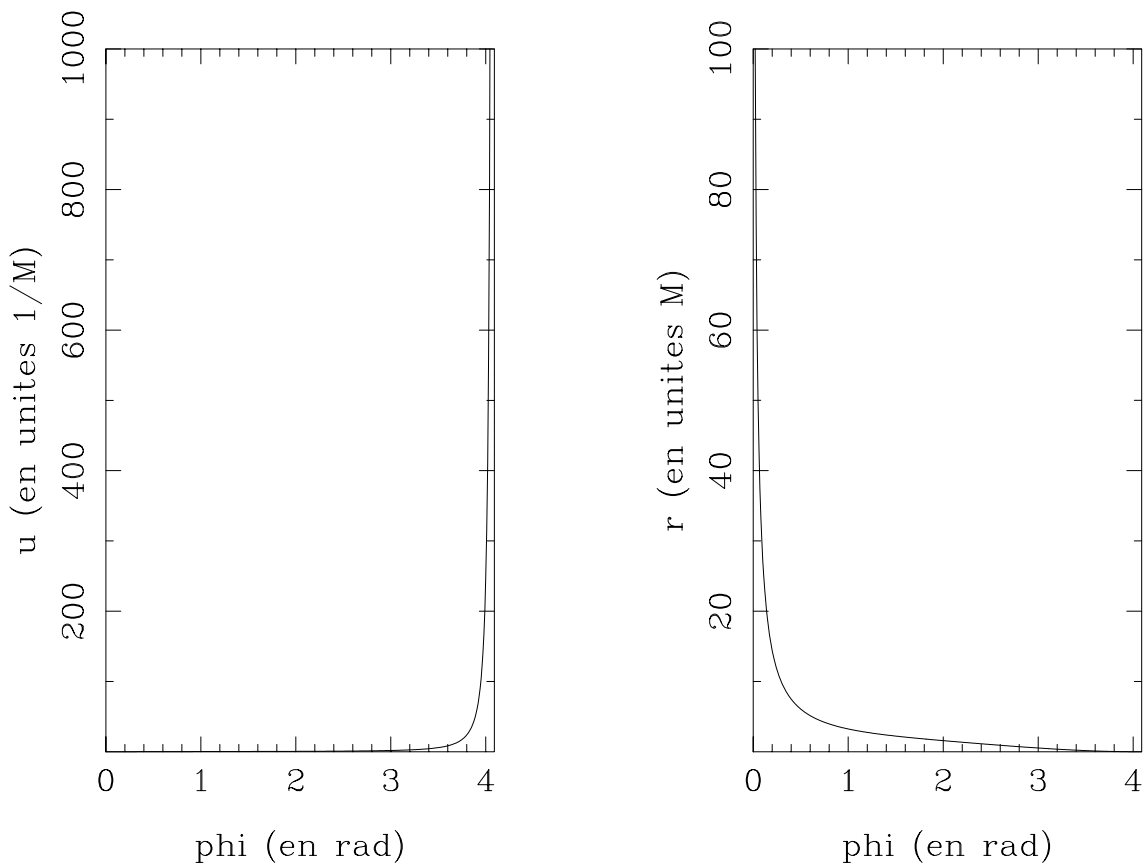


FIG. 1.3: Trajectoire d'un photon (géodésique) pour  $b = 3M$ .

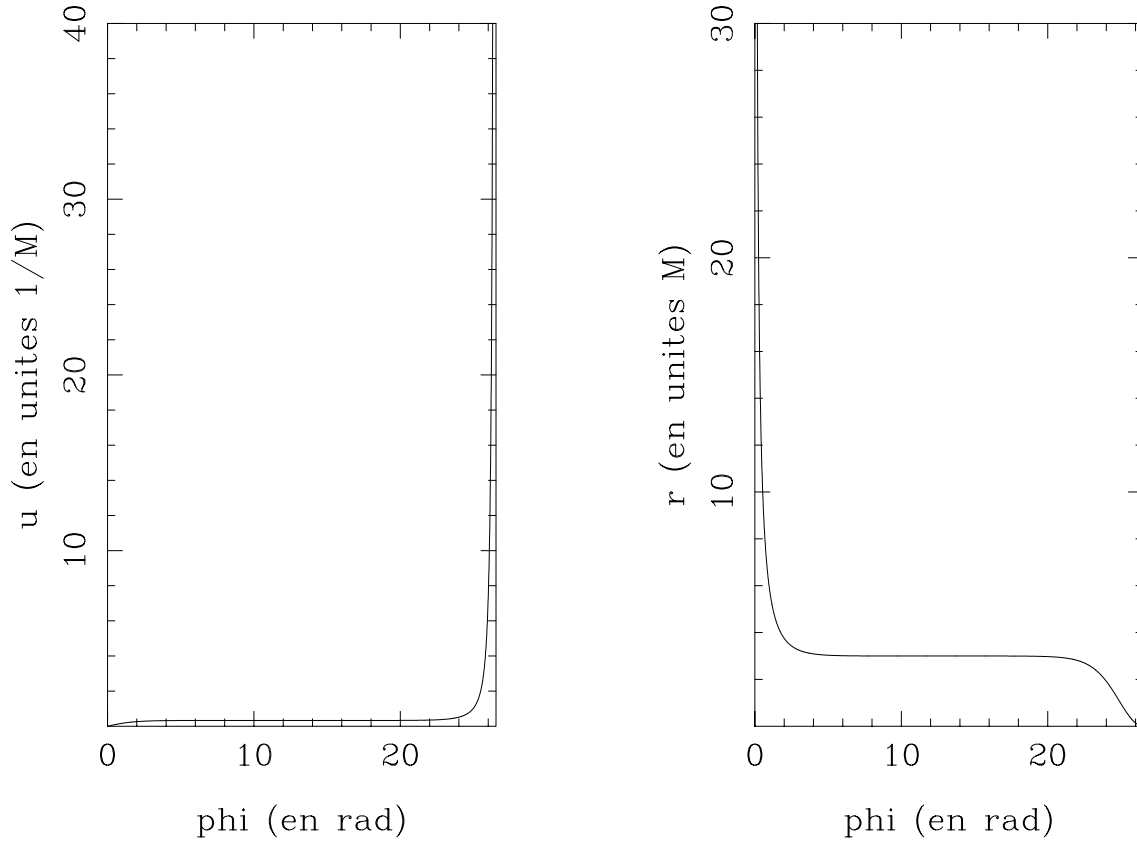


FIG. 1.4: Trajectoire d'un photon pour un paramètre d'impact  $b$  légèrement inférieur à la valeur critique de  $\sqrt{27}M$ .

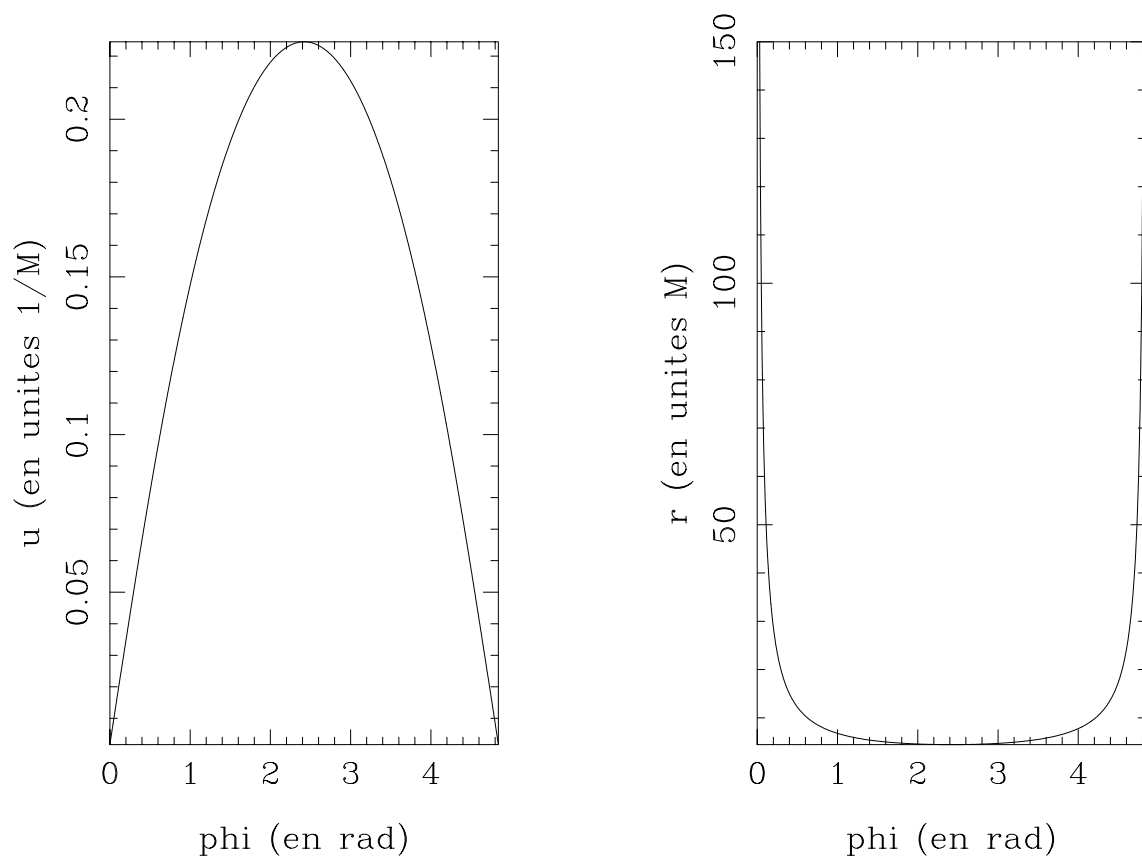


FIG. 1.5: *Trajectoire d'un photon avec  $b = 6M$ .*

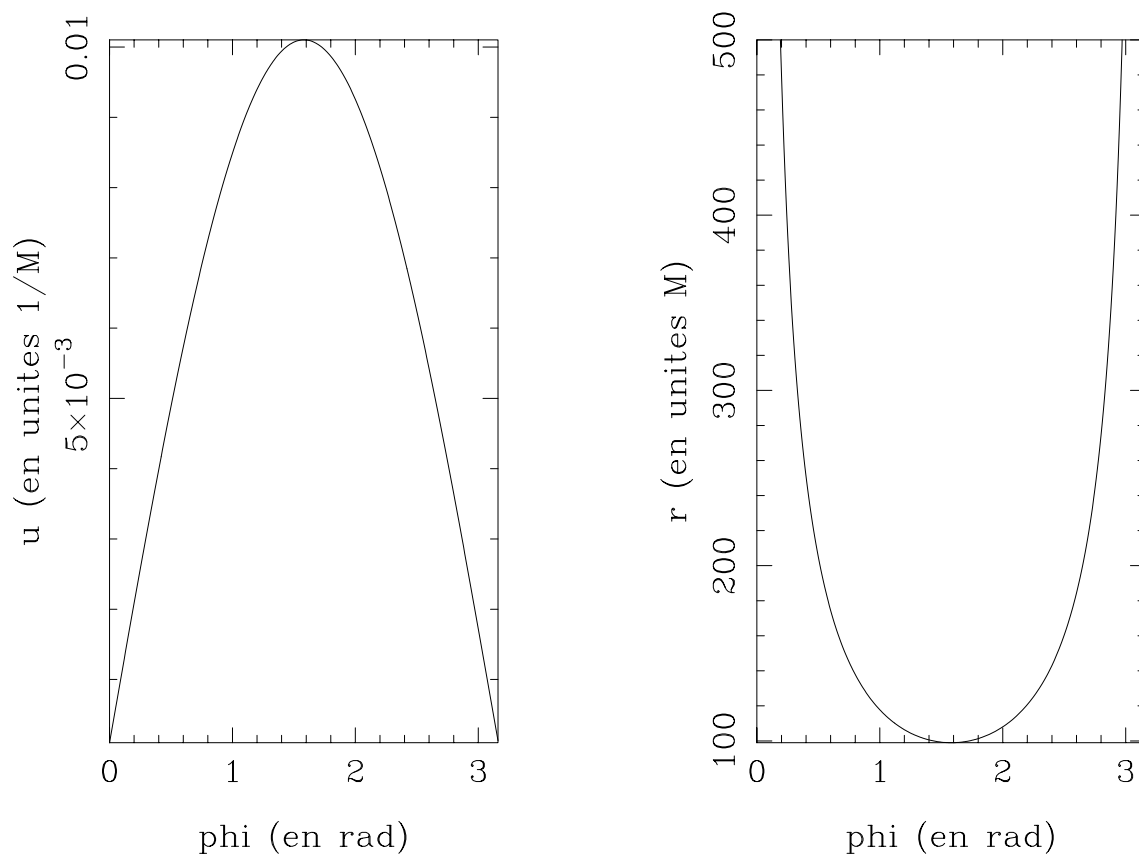


FIG. 1.6: Trajectoire d'un photon loin du trou noir ( $b = 100M$ ).

### 1.3.3 L'angle de déflexion

Comme on le voit sur la figure 1.2, l'angle de déflexion est égal à  $\phi - \pi$ .

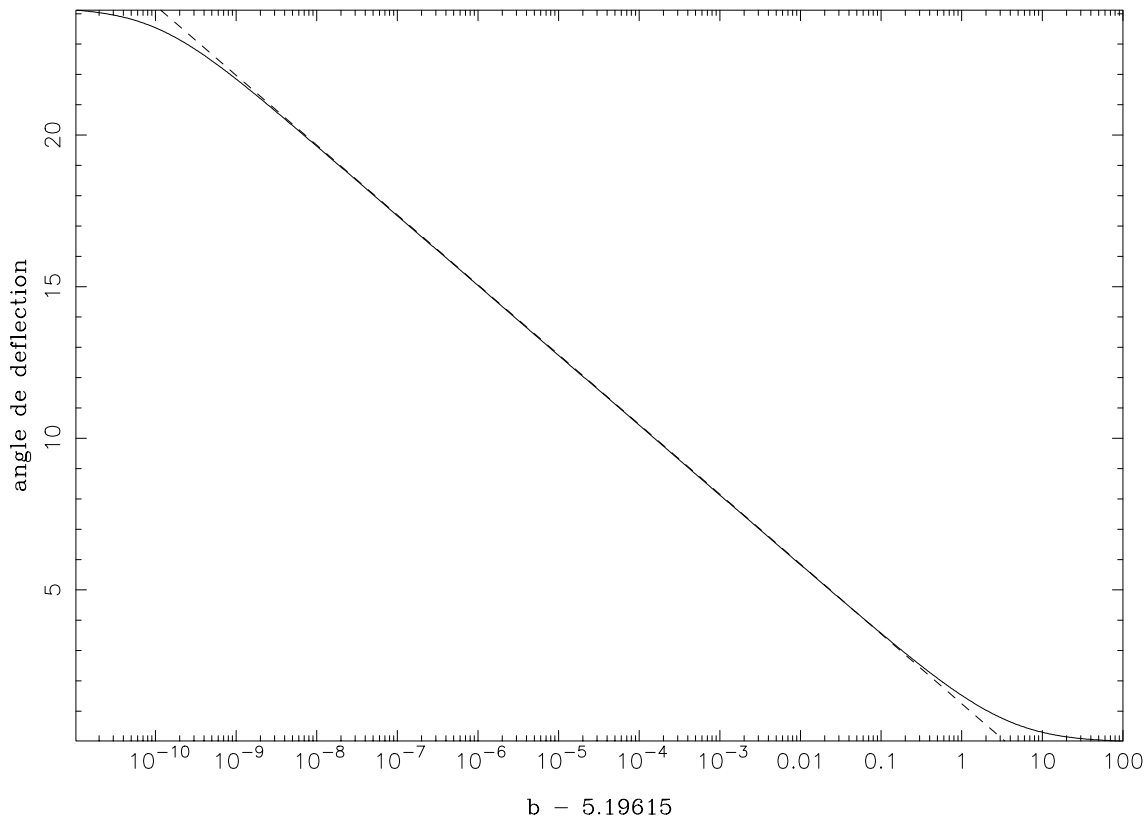


FIG. 1.7: *L'angle de déflexion en fonction du paramètre d'impact. En trait plein, notre résultat ; pour comparaison, en pointillés, la fonction asymptotique :  $b/M \sim 3^{3/2} + 3.48e^{-\theta}$  ( $M = 1$ ).*

Sur la figure 1.7, on a représenté l'angle de déflexion en fonction de  $b - b_c$  (trait plein). En tirets, on a tracé la fonction asymptotique valable quand  $b \rightarrow b_c$  (voir [2], p.678). L'accord est très bon jusqu'à  $b - b_c < 10^{-10}$  ; en-deçà il faut faire plus de tours autour du trou noir... Mais jusque là, la précision obtenue est satisfaisante !

# Chapitre 2

## Image d'un trou noir “habillé”

On suppose maintenant que le trou noir est habillé d'un disque d'accrétion orbitant dans le plan équatorial du trou noir.

Le disque est supposé infiniment mince.

### 2.1 Tracé des iso-rayons

#### 2.1.1 Méthode

On considère une trame située à l'infini du trou noir (dans le plan focal d'une lentille), que l'on repère par sa colatitude  $\theta$  (voir [1]).

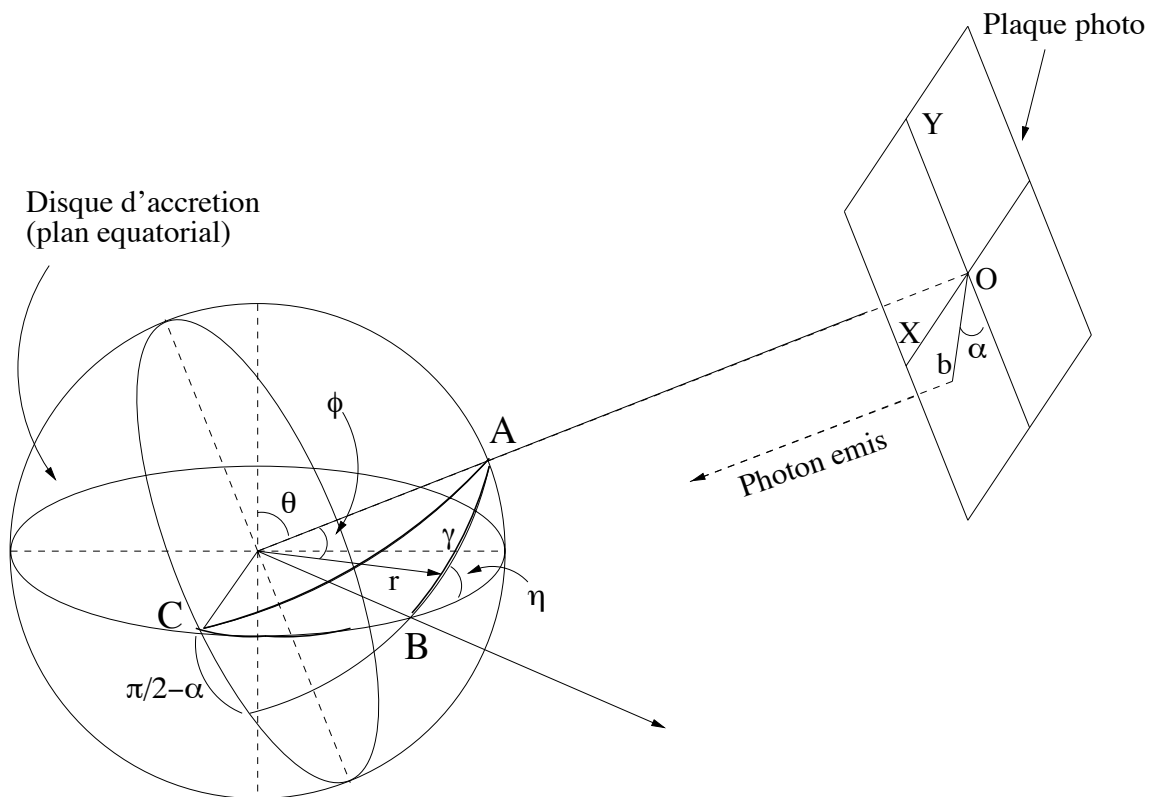
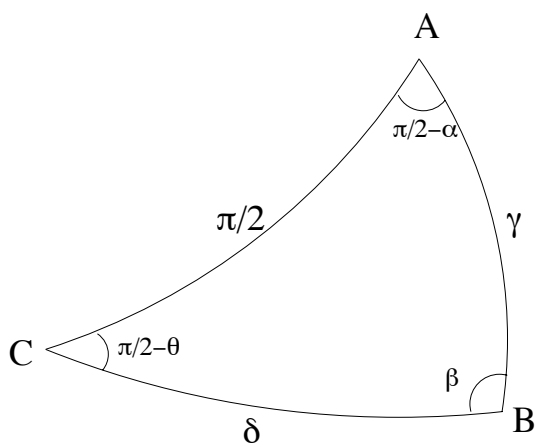
Pour chaque pixel de cette trame, on va envoyer un photon à l'aide de notre calcul de géodésiques. Puis il suffit de noter dans un tableau le rayon obtenu au point d'impact sur le disque ; on fera deux “images”, la première, appelée “directe”, correspondant à la première traversée du plan équatorial, l'autre, dite “fantôme”, correspondra au disque vu d'en dessous (deuxième traversée du plan équatorial).

La figure 2.1 montre les divers paramètres géométriques du problème. En fait, on intègre l'équation des géodésiques de  $\phi = 0$  à  $\phi = \gamma$ ,  $\gamma$  étant l'angle sphérique définissant le plan équatorial.

A l'aide d'une pincée de trigonométrie sphérique, on démontre la relation suivante :

$$\cos \gamma = \cos \alpha (\cos^2 \alpha + \cot^2 \theta)^{-1/2} \quad (2.1)$$

dans le triangle sphérique  $ABC$ , compte tenu des différents angles reportés sur la figure 2.2.

FIG. 2.1: *Géométrie de la chose*FIG. 2.2: *Voici le triangle sphérique que l'on considère*

### 2.1.2 Résultats obtenus

La figure 2.3 montre différentes courbes iso-rayons, pour une colatitude de  $80^\circ$ .

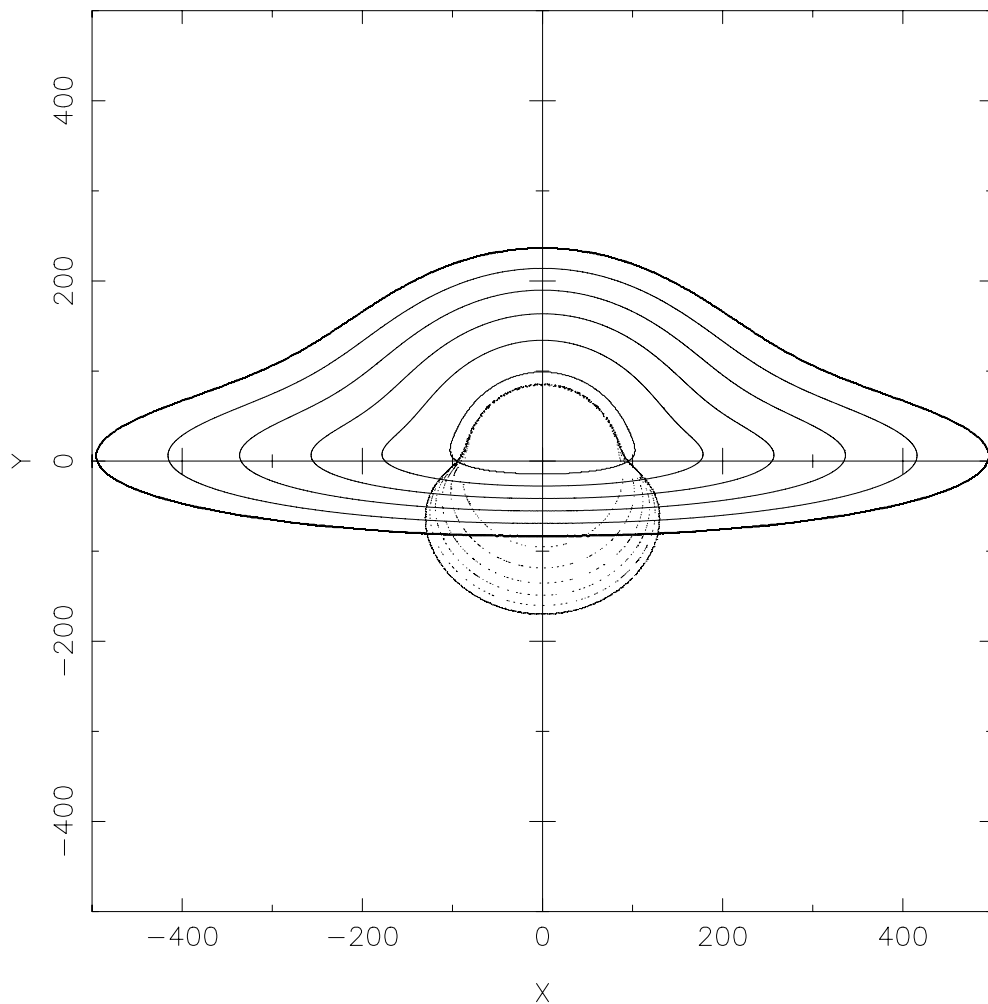


FIG. 2.3: *Courbes iso-rayon pour  $\theta = 80^\circ$ , en trait plein, le dessus du disque, et en pointillé le dessous. Le rayon intérieur du disque est  $5M$ , son rayon extérieur est  $30M$ .*

## 2.2 Calcul du redshift

### 2.2.1 Un peu de physique

Nous allons maintenant calculer le redshift  $z$  en chaque point de l'image. Le facteur  $(1+z)$ , en négligeant le redshift cosmologique à deux origines : l'une gravitationnelle, l'autre venant de l'effet Doppler.

Dans le repère de la particule émettrice le photon a une énergie qui est la projection de la 4-impulsion du photon sur la 4-vitesse de la particule émettrice :

$$E_{em} = p_t u^t + p_\phi u^\phi = p_t u^t \left( 1 + \Omega \frac{p_\phi}{p_t} \right) \quad (2.2)$$

où  $\Omega$  est la vitesse angulaire de la particule autour du trou noir. Les composantes  $p_t$  et  $p_\phi$  de la 4-impulsion du photon sont conservées le long de sa trajectoire ; loin du trou noir,  $p_t$  est l'énergie observée  $E_{obs}$  du photon, tandis que  $p_\phi$  est la projection de son moment cinétique le long de l'axe polaire. Le rapport  $p_\phi/p_t$  est le paramètre d'impact du photon projeté sur l'axe polaire :

$$\frac{p_\phi}{p_t} = b \sin \alpha \sin \theta \quad (2.3)$$

Le redshift du photon est donné par :

$$\begin{aligned} 1+z &= \frac{E_{em}}{E_{obs}} = u^t (1 + \Omega b \sin \alpha \sin \theta) \\ &= \frac{1 + \Omega b \sin \alpha \sin \theta}{\sqrt{-g_{tt} - 2g_{t\phi} - 2g_{\phi\phi}}} \end{aligned} \quad (2.4)$$

Dans le cas de la métrique de Schwarzschild,  $\Omega$  a la valeur keplerienne  $\Omega = (M/r^3)^{1/2}$  et les composantes du tenseur métrique sont données par A.2. Donc, à partir de 2.4, on obtient :

$$1+z = \frac{1 + \sqrt{\frac{M}{r^3}} b \sin \alpha \sin \theta}{\sqrt{1 - \frac{3M}{r}}} \quad (2.5)$$

Comme, on connaît  $r, b, \alpha$  et  $\theta$  pour chaque point de l'image, on détermine aisément le décalage spectral pour chacun d'eux.

### 2.2.2 Courbes iso-redshift obtenues

La figure 2.4 montrent quelques valeurs de  $z$ , pour une colatitude de  $\theta = 80^\circ$ . A gauche du trou noir les photons sont décalés vers le bleu, tandis qu'à droite, ils sont décalés vers le rouge.

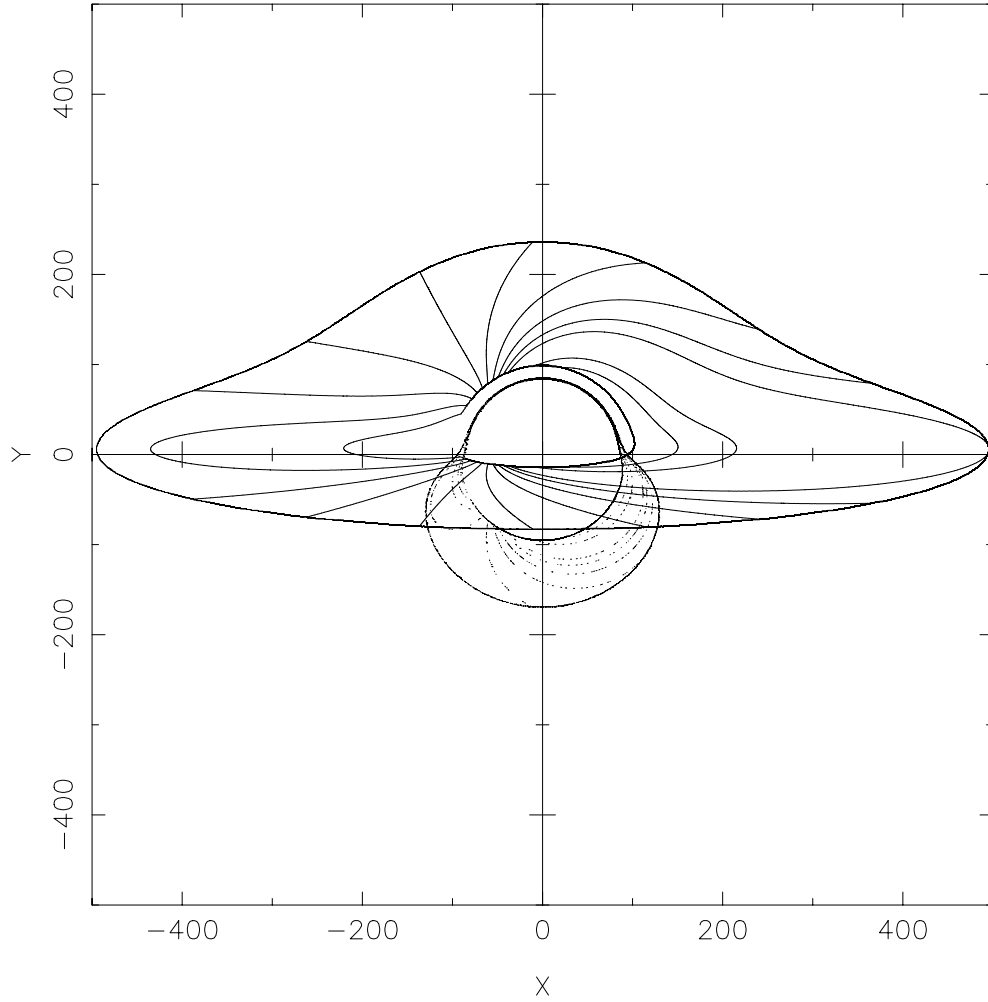


FIG. 2.4: *Courbes iso-redshift, pour le dessus du disque (en traits pleins) et pour le dessous du disque (en pointillés), pour  $\theta = 80^\circ$ . Ceci pour un disque dont le rayon intérieur a été pris égal à  $3M$ , et le rayon extérieur à  $30M$ .*



# Chapitre 3

## Image bolométrique

### 3.1 Aspects qualitatifs

On s'intéresse au flux total qui est issu du disque d'accrétion. Le flux  $F_\nu$  s'écrit de manière générale, pour une fréquence  $\nu$  donnée :

$$F_\nu = \int I_\nu \cos \theta d\Omega \quad (3.1)$$

où  $\theta$  est l'angle par rapport à la normale de la surface émissive, et  $I_\nu$  est l'intensité du rayonnement émis :

$$I_\nu = \frac{dE}{dS dt d\Omega d\nu} \quad (3.2)$$

(énergie émise par unité de surface  $dS$ , de temps  $dt$ , de fréquence  $d\nu$  et d'angle solide  $d\Omega$ ).

Cependant ces quantités ne sont pas conservées par changement de référentiel relativiste.

Nous allons voir ce que l'on obtient en relativité générale, à l'aide d'un petit raisonnement intuitif, qui ne tient lieu en aucun cas de démonstration.

Par définition du décalage spectral, on a :

$$1 + z = \frac{\nu_{source}}{\nu_{obs}} = \frac{dE_{source}}{dE_{obs}}$$

donc :

$$1 + z = \frac{dt_{obs}}{dt_{source}}$$

Par ailleurs, l'élément de surface considéré étant orthogonal aux rayons lumineux,

$$dS_{obs} = dS_{source}$$

mais un élément de longueur  $dl$  le long de la trajectoire devient :

$$dl_{obs} = (1 + z)dl_{source}$$

On en déduit pour l'angle solide :

$$d\Omega_{obs} = (1 + z)^2 d\Omega_{source}$$

soit finalement :

$$\begin{aligned} I_{\nu,source} &= \frac{dE_{source}}{dS_{source} dt_{source} d\Omega_{source} d\nu_{source}} \\ &= \frac{1}{(1 + z)^3} \frac{dE_{obs}}{dS_{obs} dt_{obs} d\Omega_{obs} d\nu_{obs}} \\ &= (1 + z)^{-3} I_{\nu,obs} \end{aligned}$$

Donc, on obtient l'invariant relativiste suivant :

$$\left( \frac{I_{\nu,obs}}{\nu_{obs}^3} \right) = \left( \frac{I_{\nu,source}}{\nu_{source}^3} \right) \quad (3.3)$$

La quantité  $I_{\nu}/\nu^3$  représente en fait la densité de photons dans l'espace des phases. En effet, si on note  $q$  la coordonnée généralisée d'espace, et  $p$ , l'impulsion, on a :

$$d^3q = dt dS \quad (3.4)$$

$$d^3p = k^2 dk d\Omega \propto \nu^2 d\nu d\Omega \quad (3.5)$$

où  $k$  est le vecteur d'onde du photon.

Donc :

$$\frac{I_{\nu}}{\nu^3} = \frac{dE}{\nu(\nu^2 d\nu d\Omega dt dS)} = \frac{\text{Nb photons}}{d^3q d^3p} \quad (3.6)$$

Toujours est-il que compte tenu de 3.3, on obtient la relation suivante sur le flux monochromatique :

$$F_{\nu,obs} = (1+z)^{-5} F_{\nu,source} \quad (3.7)$$

Mais si on intègre la relation 3.7 sur toutes les fréquences, on obtient le flux bolométrique  $F_{bol,obs}$  :

$$F_{bol,obs} = (1+z)^{-4} F_{bol,source} \quad (3.8)$$

## 3.2 Tracé de l'image correspondante

Comme on a calculé le décalage spectral  $z$  pour toutes les points de notre image, on en déduit aisément l'image bolométrique, compte tenu de la relation 3.8 : il suffit de tracer  $F_{bol,obs}/F_{bol,source} = (1+z)^{-4}$  pour chaque pixel de l'image. On obtient la figure 3.1.

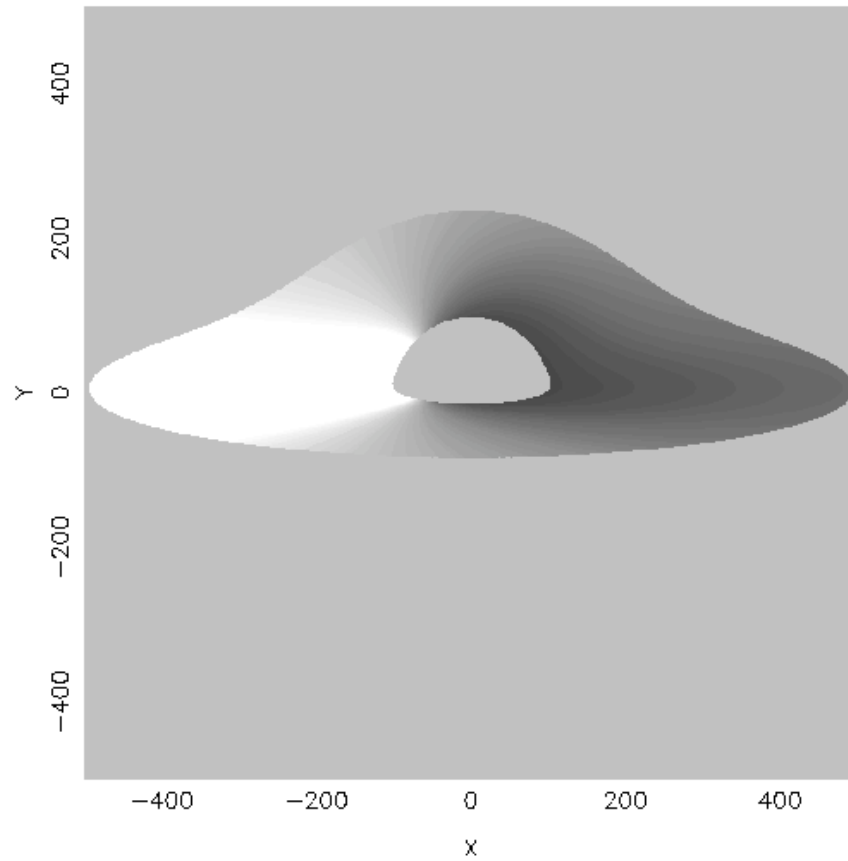


FIG. 3.1: *Image bolométrique,  $\theta = 80^\circ$ . Les rayons intérieur et extérieurs du disque ont été choisis à  $5M$  et  $30M$  respectivement.*

# Chapitre 4

## La réponse spectrale du disque

### 4.1 Comment ça marche?

On suppose que chaque point du disque émet une raie parfaitement monochromatique, de la forme :

$$I_{\nu,source} = r^q \delta(\nu_{source} - \nu_o) \quad (4.1)$$

où  $q$  est un indice spectral ; on prendra  $q = -2$ .

En injectant cette relation dans 3.1, et en prenant  $\theta = 0$  dans cette même relation, on obtient :

$$F_{\nu,obs} = \int \frac{I_{\nu,source}}{\nu_{source}^3} \nu_{obs}^3 d\Omega \quad (4.2)$$

$$= \int r^q (1+z)^{-3} \delta(\nu_{source} - \nu_o) d\Omega \quad (4.3)$$

$$= \int r^q (1+z)^{-3} \delta((1+z)\nu_{obs} - \nu_o) d\Omega \quad (4.4)$$

$$= \frac{1}{(1+z)^4} \int r^q \delta\left(\nu_{obs} - \frac{\nu_o}{1+z}\right) d\Omega \quad (4.5)$$

### 4.2 Calcul du spectre associé

En fait nous allons calculer la grandeur  $\nu_o F_{\nu,obs}$  telle que :

$$\nu_o F_{\nu,obs} = \frac{1}{(1+z)^4} \int r^q \delta\left(\frac{\nu_{obs}}{\nu_o} - \frac{1}{1+z}\right) d\Omega \quad (4.6)$$

Pour calculer numériquement cela, c'est-à-dire intégrer sur l'image, on se donne un tableau "spec" de NB\_NU cases. Dans la case correspondant à  $(1+z)^{-1}$ , on va ajouter

la valeur  $r^q/(1+z)^4$  adéquate.

Cependant, si calcule le spectre de cette manière, avec les rayons et les redshifts précédemment calculés, on obtient un spectre particulièrement bruité.

On pourrait augmenter le temps de calcul, mais il y a une manière plus subtile de mener celui-ci. En effet, on peut remarquer le calcul du trajet d'un photon se fait à paramètre d'impact constant : il suffit de se donner un  $b$ , puis de faire varier l'angle  $\alpha$  (voir figure 2.1) entre 0 et  $\pi$ , en prenant soin auparavant de calculer l'angle  $\gamma$  associé.

En clair, l'algorithme utilisé est le suivant :

- On se donne un pas en angle  $\alpha$ , sur l'image ;
- Pour chaque  $\alpha$ , on calcule l'angle  $\gamma$  associé ;
- On se donne un paramètre d'impact  $b$  ;
- Pour chaque  $b$ , on intègre une seule fois, en relevant le rayon de l'impact sur le disque pour chacun des  $\gamma$  rencontrés<sup>1</sup>.

On fait ainsi beaucoup moins de calculs, ce qui limite le bruit numérique. Pour plus de détails, on pourra se reporter au listing, dans le paragraphe B.4.

### 4.3 Le spectre

La figure 4.1 montre le spectre qu'un tel objet nous donnerait. Sa forme ressemble au spectre d'une galaxie à 21 cm : c'est le même principe, on a un disque en rotation, que l'on observe avec une certaine inclinaison (ici  $i = \theta = 70^\circ$ ). Dans notre cas, la raie est asymétrique, déformée par les effets relativistes.

---

1. En effet, comme  $\gamma$  est une fonction croissante de  $\alpha$ , pour  $\alpha$  variant entre 0 et  $\pi$ , les différents  $\gamma$  sont forcément rangés dans l'ordre croissant, pour  $\alpha \in [0, \pi]$

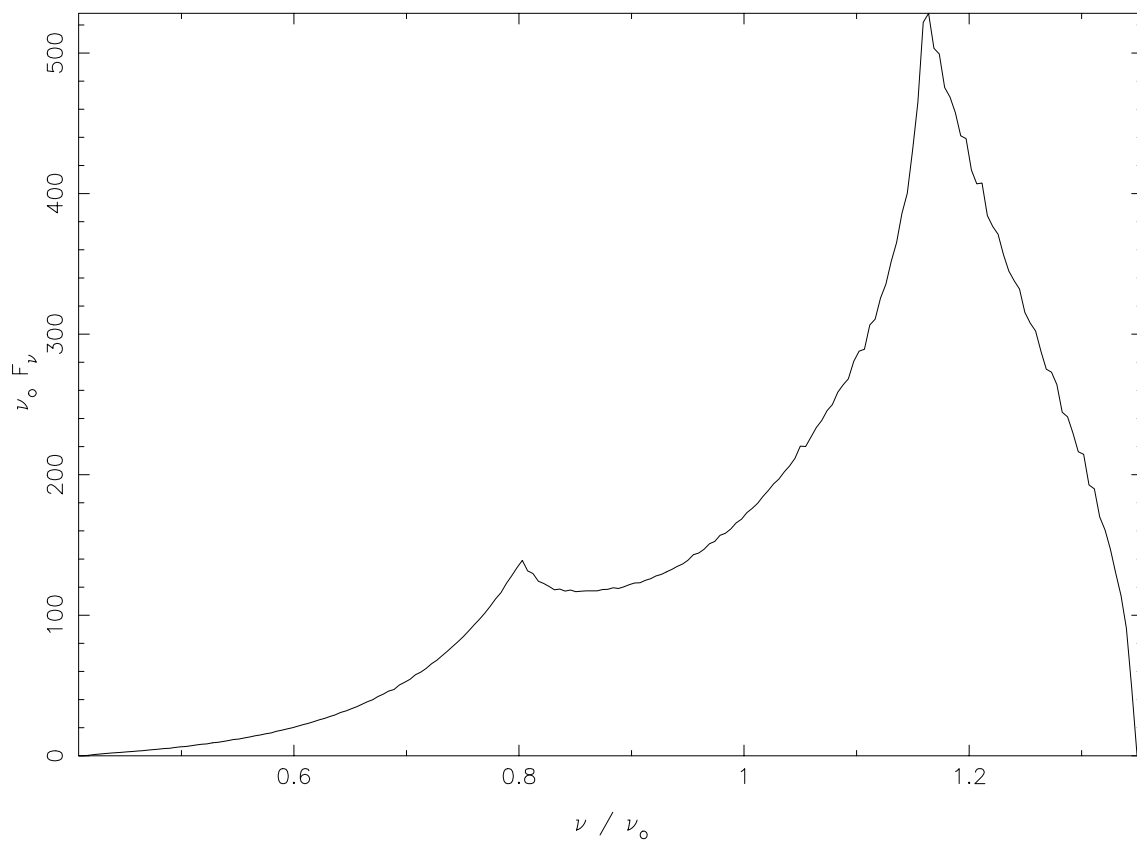


FIG. 4.1: Spectre obtenu pour une colatitude  $\theta = 70^\circ$ , et un disque s'étendant de  $5M$  à  $30M$  autour du trou noir.



# Conclusion

Il reste beaucoup de choses à faire pour mener ce projet à son terme : tout d'abord calculer le spectre, plus réaliste dans l'hypothèse où l'émission du disque est celle d'un corps noir.

Il aurait fallu également éprouver notre programme dans des cas tordus, avec des disques de tailles variées, ce que nous n'avons pas eu le temps de faire...

Mais surtout optimiser les calculs, qui prennent finalement trop de temps.

Quant au programme lui-même, il y a de nombreuses choses à améliorer, à modifier, pour tenir compte de tous les cas possibles...

Néanmoins, les résultats que nous avons obtenu sont plutôt jolis (?), et intéressants, ne serait-ce que pour nous : après avoir rêver sur l'image que Jean-Pierre Luminet montre dans son livre *Les Trous Noirs*, ce fut tout à fait passionnant de s'amuser à la démystifier...



# Bibliographie

- [1] Luminet, J.-P.: *Image of a spherical black hole with thin accretion disk*, A. & A., 75, 228-235, 1979.
- [2] Misner, C.W., Thorne, K.S., Wheeler, A.J.: 1973, *Gravitation*, Freeman, San Francisco.
- [3] *Numerical Recipes*, Integration of Ordinary Differential Equations, Runge Kutta Method, Chapter 16.



## Annexe A

### Equation des géodésiques

La métrique de Schwarzschild en coordonnées sphériques  $(r, \theta, \phi)$  est donnée par :

$$ds^2 = -\left(1 - \frac{2M}{r}\right) dt^2 + \left(1 - \frac{2M}{r}\right)^{-1} dr^2 + r^2 (d\theta^2 + \sin^2\theta d\phi^2) \quad (\text{A.1})$$

avec  $G = c = 1$ ,  $M$  étant la masse gravitationnelle.

Les composantes du tenseur métrique sont donc les suivantes :

$$\begin{aligned} g_{tt} &= -\left(1 - \frac{2M}{r}\right) \\ g_{rr} &= \left(1 - \frac{2M}{r}\right)^{-1} \\ g_{\theta\theta} &= r^2 \\ g_{\phi\phi} &= r^2 \sin^2\theta \end{aligned} \quad (\text{A.2})$$

Par ailleurs, l'équation générale des géodésiques s'écrit :

$$\frac{d}{dp} \left( g_{\lambda\mu} \frac{dx^\mu}{dp} \right) = \frac{1}{2} \partial_\lambda g_{\mu\nu} \frac{dx^\mu}{dp} \frac{dx^\nu}{dp} \quad (\text{A.3})$$

où  $p$  est un paramètre affine le long de la trajectoire du photon.

A partir de A.3 et de A.2, on obtient :

– une équation pour  $\theta$  :

$$\frac{d}{dp} \left( r^2 \frac{\theta}{dp} \right) = r^2 \sin\theta \cos\theta \left( \frac{\phi}{dp} \right)^2 \quad (\text{A.4})$$

qui a pour solution  $\theta = \frac{\pi}{2}$ : la trajectoire est plane, on la choisit dans le plan équatorial.

– une équation pour  $t$ , qui est ignorable, ce qui donne :

$$-\left(1 - \frac{2M}{r}\right) \frac{dt}{dp} = ct e_1 \quad (\text{A.5})$$

– une équation pour  $\phi$ , qui est aussi ignorable :

$$r^2 \sin^2 \theta \frac{d\phi}{dp} = ct e_2 \quad (\text{A.6})$$

– l'intégrale première de l'énergie :

$$\left(1 - \frac{2M}{r}\right)^{-1} dr^2 + r^2 (d\theta^2 + \sin^2 \theta d\phi^2) = -\left(1 - \frac{2M}{r}\right) dt^2 \quad (\text{A.7})$$

Comme  $\theta = \frac{\pi}{2}$ , l'équation A.7 devient :

$$\left(1 - \frac{2M}{r}\right)^{-1} dr^2 + r^2 d\phi^2 = -\left(1 - \frac{2M}{r}\right) dt^2 \quad (\text{A.8})$$

A partir de A.6 et A.5, on élimine les variables  $t$  et  $p$  de A.8 pour obtenir l'équation des géodésiques :

$$\left(\frac{1}{r^2} \frac{dr}{d\phi}\right)^2 + \frac{1}{r^2} \left(1 - \frac{2M}{r}\right) = \left(\frac{ct e_1}{ct e_2}\right)^2 = \frac{1}{b^2} \quad (\text{A.9})$$

où  $b$  est le paramètre d'impact. On obtient ainsi l'équation de la trajectoire d'un photon dans la métrique de Schwarzschild.

# Annexe B

## Listing du programme

Nous avons programmé en Fortran 90, en segmentant notre programme en différents modules :

**module runge\_kutta :** contient la subroutine d'intégration utilisée, ainsi que la fonction retournant le système différentiel.

**module geodesique :** contient les fonctions et subroutines propres au calcul de la trajectoire d'un photon ; il contient en outre le calcul de l'angle de déflexion.

**module photo :** contient tout ce qui permet de faire les images iso-rayon et iso-redshift, ainsi que le calcul du spectre (bruité).

**module spectre :** pour le calcul du spectre en coordonnées polaires.

**module constantes :** tous les paramètres globaux du programme.

**module precision :** pour switcher entre simple et double précision.

**module graphique :** toutes les subroutines d'affichages ; les différents graphes ont été réalisées avec PGPLOT.

**module ppal :** le programme principal, pour basculer d'un calcul à l'autre.

### B.1 Module runge\_kutta

Ce module contient les procédures suivantes :

- subroutine **derivs**
- subroutine **rk\_4**

```

module runge_kutta

use precision, only : P => DP

contains
  !
  ! Subroutine retournant le systeme :
  !   dv/dphi = f(u,phi)
  !   du/dphi = g(v,phi)
  !
  !   dydx(1) = dv/dphi   -   y(1) = v
  !   dydx(2) = du/dphi   -   y(2) = u
  !
  subroutine derivs(x, y, dydx)
    real(P), intent(in) :: x
    real(P), dimension(2), intent(in) :: y
    real(P), dimension(2), intent(out) :: dydx
    real(P) :: aux

    dydx(1) = 3 * y(2)**2 - y(2)
    dydx(2) = y(1)

  end subroutine derivs
  !
  ! Subroutine d'integration numerique, par la methode de
  !   Runge-Kutta a pas constant, a l'ordre 4
  !
  subroutine rk_4 (y, dydx, n, x, h, yout)
    real(P), intent(in) :: h, x
    integer, intent(in) :: n
    real(P), dimension(n), intent(in) :: dydx, y
    real(P), dimension(n), intent(out) :: yout

    real(P) :: h6, hh, xh
    real(P), dimension(n) :: dym, dyt, yt
    integer :: i

    hh = h * 0.5_P
    h6 = h/6._P
    xh = x + hh

    yt = y + hh * dydx

```

```

    call derivs(xh, yt, dyt)

    yt = y + hh * dyt
    call derivs(xh, yt, dym)

    yt = y + h * dym
    dym = dyt + dym
    call derivs(x + h, yt, dyt)

    yout = y + h6 * (dydx + dyt + 2._P * dym)
end subroutine rk_4

end module runge_kutta

```

## B.2 Module geodesique

Ce module contient les procédures :

- subroutine **integ\_geo**
- fonction **defl\_angle**

```

module geodesique

use precision, only : P => DP
use constantes
use runge_kutta

contains
!
! Cette subroutine prend en entree le parametre d'impact 'b' ;
! elle donne en sortie deux tableaux de meme dimension 'dim',
! 'angle' et 'inv_ray' qui sont respectivement l'angle
! phi et l'inverse du rayon, pour chacun des pas d'integration
!
subroutine integ_geo (b, angle, inv_ray, dim)
    implicit none

    real(P), intent(in) :: b
    integer, intent(out) :: dim

    real(P), dimension(N), intent(out) :: angle, inv_ray

```

```

real(P) :: h, phi, M, htry, eps, hdid, hnext
real(P), dimension(2) :: u, diff, yscal
real(P), dimension(2,N) :: yout
real(P), dimension(N) :: xout

integer :: i
!
! Initialisation des variables
!
u(2) = 0._P
u(1) = 1./b
phi = 0._P
h = 0.01_P
M = 1._P

do i = 1, N
  call derivs(phi, u, diff)
  call rk_4(u, diff, 2, phi, h, yout(:,i))

  u = yout(:,i)
  xout(i) = phi
  phi = phi + h
  !
  ! yout(2,i) = u
  ! yout(1,i) = v
  !
  ! Cas ou on atteint l'angle max de deviation
  !
  if (yout(2,i) <= 0.) exit
  !
  ! Cas ou le photon "tombe" sur le trou noir
  !
  if (1./yout(2,i) <= tolerance) exit

enddo

dim = i-2
angle = 0.
inv_ray = 0.
!
angle(1:dim) = xout(1:dim)

```

```

        inv_ray(1:dim) = yout(2,1:dim)
        !
    end subroutine integ_geo
    !
    ! Cette fonction retourne l'angle de deflection pour
    !         le parametre d'impact 'b' donne
    !
    real(P) function defl_angle(b)
        implicit none

        real(P), intent(in) :: b

        real(P), dimension(N) :: xxaux, yyaux
        real(P), dimension(:), allocatable :: xx, yy
        integer :: dim

        call integ_geo (b, xxaux, yyaux, dim)
        allocate (xx(dim), yy(dim))
        xx = xxaux(1:dim)
        yy = yyaux(1:dim)

        if (yy(dim) < 1._P/R_S) defl_angle = xx(dim) - pi
        if (1./yy(dim) < tolerance) write(*,*) "Le photon tombe sur le TN"
        if (dim == N) write(*,*) "On tourne..."

    end function defl_angle
    !
end module geodesique

```

## B.3 Module photo

Ce module contient les procédures suivantes :

- fonction **facteur\_rel**
- fonction **par\_imp**
- fonction **gamma**
- subroutine **integration**
- subroutine **impact**

- subroutine **iso\_z**
- subroutine **spectre\_b**

module photo

```

use precision, only : P => DP
use constantes
use runge_kutta

```

contains

```

!
! Mise au point pour avoir une image cadree sur le trou noir
! Cette fonction retourne le facteur relativiste de
! grossissement du bord externe du disque
!
real(P) function facteur_relat(b)
  implicit none

  real(P), intent(in) :: b
  real(P) :: h, phi, gamma
  real(P), dimension(2) :: u, diff
  real(P), dimension(2,N) :: yout
  real(P), dimension(N) :: xout
  integer :: i

  u(2) = 0._P
  u(1) = 1./b
  phi = 0._P
  h = 0.01_P
  i = 1

  gamma = pi/2      ! Cas ou x = R_MAX, y = 0 : Correction horizontale

```

```

A:  do
      call derivs(phi, u, diff)
      call rk_4(u, diff, 2, phi, h, yout(:,i))
      u = yout(:,i)
      xout(i) = phi
      phi = phi + h

      ! Cas ou on atteint le plan du disque d'accretion
      if (phi >= gamma) then

```

```

        exit A
    endif

    i = i + 1
enddo A

write (*,*) "periastre ", 1./yout(2,i)
write (*,*) "grossissement", b * yout(2,i)

! grossissement = parametre d'impact( = b)
!                / rayon du periastre ( = 1./yout() )

facteur_relat = b * yout(2,i)

end function facteur_relat
!
! Fonction qui retourne le parametre d'impact, compte tenu de la position
!   sur l'image (en x et y) et du facteur de grossissement
!
real(P) function par_imp(x, y, fac_rel)
    implicit none
    real(P), intent(in) :: x, y, fac_rel
    ! parametre d'impact = module du point de l'ecran * facteur d'adaptation
    ! au rayon du disque * facteur de securite * grossissement
    ! relativiste
    par_imp = sqrt(x**2 + y**2) * R_MAX / DIM_IM * MARGE * fac_rel
end function par_imp
!
! Fonction qui retourne l'angle Gamma correspondant au disque
!   d'accretion
!
real(P) function gamma(x, y, theta)
    implicit none
    real(P), intent(in) :: x, y, theta
    real(P) :: alpha, cos_gamma

    if (theta == 0.) then
        gamma = pi/2
    else
        alpha = atan2(x, -y)
        cos_gamma = cos(alpha) / &
            sqrt(cos(alpha)**2 + (cos(theta) / sin(theta))**2)
    end if
end function gamma

```

```

        gamma = acos(cos_gamma)
    endif
end function gamma
!
! Cette subroutine integre jusqu'a ce que Phi = Gamma
!       et retourne le rayon atteint (rimpact) sur le disque
!       Si le photon se perd dans les meandres de l'espace
!       on retourne 0.
!
subroutine integration (b,gamma,rimpact)
    implicit none

    real(P), intent(in) :: b,gamma
    real(P), intent(out) :: rimpact

    real(P) :: h, phi
    real(P), dimension(2) :: u, diff
    real(P), dimension(2,N) :: yout
    real(P), dimension(N) :: xout

    integer :: i

    !
    ! Initialisation des variables
    !
    rimpact = 0.

    u(2) = 0._P
    u(1) = 1./b
    phi = 0._P
    h = 0.001_P

    i = 1
A: do
    call derivs(phi, u, diff)
    call rk_4(u, diff, 2, phi, h, yout(:,i))
    u = yout(:,i)
    xout(i) = phi
    phi = phi + h

    if (phi >= gamma) then
        if ((1._P/yout(2,i) >= R_MIN) .and. &

```

```

        (1._P/yout(2,i) <= R_MAX)) then
            rimpact = 1._P/yout(2,i-1)
        else
            rimpact = 0.
        endif

        exit A
    endif
    !
    !   Cas ou le photon "tombe" sur le trou noir
    !   en-deca de 2 fois le rayon de Schwarzschild
    !
    if (1._P/yout(2,i) < 3._P) then
        rimpact = 0.
        exit A
    endif

    i = i + 1
enddo A

end subroutine integration
!
!   Cette subroutine remplit deux tableaux bidimensionnels
!   contenant l'image :
!       direct : iso_rayon pour la premiere traversee
!               du disque
!       ghost  : iso_rayon pour la deuxieme traversee
!               du disque
!
subroutine impact(theta, direct, ghost, fac_rel)
    implicit none

    real(P), intent(in) :: theta, fac_rel
    real(P), dimension(-DIM_IM:DIM_IM,-DIM_IM:DIM_IM), &
        intent(out) :: direct, ghost

    real(P) :: x, y, b, gamm
    integer :: ix, iy
    real(P) :: rimpact

    direct = 0.
    ghost = 0.

```

```

write(*,*) "Calcul de l'image iso_rayon"

!
! On remplit le tableau "direct"
!
write(*,*) "    Remplissage de l'image directe"
do iy = -DIM_IM, DIM_IM
do ix = 0, DIM_IM

    if((ix == 0) .and. (iy == 0)) then
        if (theta == PI/2._P) then
            rimpact = R_MAX
        else
            rimpact = 0.
        endif
    else
        b = par_imp(real(ix, P), real(iy, P), fac_rel)
        gamm = gamma(real(ix, P), real(iy, P), theta)
        call integration(b,gamm,rimpact)
        ! rimpact = rayon atteint par le photon
        !    sur le disque d'accretion
    endif

    direct(ix,iy) = rimpact
enddo
enddo

!
! On remplit le tableau "ghost"
!
write(*,*) "    Remplissage de l'image fantome"
do iy = -DIM_IM, DIM_IM
do ix = 0, DIM_IM

    if((ix == 0) .and. (iy == 0)) then
        if (theta == PI/2._P) then
            rimpact = R_MAX
        else
            rimpact = 0.
        endif
    endif

```

```

    else
        b = par_imp(real(ix, P), real(iy, P), fac_rel)
        gamm = PI + gamma(real(ix, P), real(iy, P), theta)
        call integration(b, gamm, rimpact)
        ! rimpact = rayon atteint par le photon
        !      sur le disque d'accretion
    endif

    ghost(ix, iy) = rimpact
enddo
enddo

!
! L'image est symetrique par rapport a l'axe Y
!
do iy = -DIM_IM, DIM_IM
do ix = -DIM_IM, -1
    direct(ix, iy) = direct(-ix, iy)
    ghost(ix, iy) = ghost(-ix, iy)
enddo
enddo

!      direct(-DIM_IM:-1,:) = direct(1:DIM_IM,:)
!      ghost(-DIM_IM:-1,:) = ghost(1:DIM_IM,:)

end subroutine impact

!
! Subroutine qui remplit les tableaux "z_direct" et "z_ghost"
!      correspondant aux redshifts
!
subroutine iso_z(fac_rel, theta, direct, ghost, z_direct, z_ghost)
    implicit none

    real(P), intent(in) :: theta, fac_rel
    real(P), dimension(-DIM_IM:DIM_IM, -DIM_IM:DIM_IM), &
        intent(in) :: direct, ghost
    real(P), dimension(-DIM_IM:DIM_IM, -DIM_IM:DIM_IM), &
        intent(out) :: z_direct, z_ghost
    real(P) :: alpha, b
    integer :: ix, iy

```

```

write(*,*) "Calcul du redshift"

do iy = -DIM_IM, DIM_IM
do ix = -DIM_IM, DIM_IM
  if((ix == 0) .and. (iy == 0)) then
    z_direct(ix, iy) = -2.
    z_ghost(ix, iy) = -2.
  else
    b = par_imp(real(ix, P), real(iy, P), fac_rel)
    alpha = atan2(real(ix, P), -real(iy, P))

    if ((direct(ix,iy) /= 0.) .and. (direct(ix,iy) /= 3._P)) then
      z_direct(ix, iy) = (1._P + 1._P / sqrt(direct(ix,iy)**3) * &
        b * sin(theta) * sin(alpha)) / &
        sqrt(1._P - 3._P / direct(ix,iy)) - 1._P
    else
      z_direct(ix, iy) = -2.0
    endif

    if ((ghost(ix,iy) /= 0.) .and. (ghost(ix,iy) /= 3._P)) then
      z_ghost(ix, iy) = (1._P + 1._P / sqrt(ghost(ix,iy)**3) * &
        b * sin(theta) * sin(alpha)) / &
        sqrt(1._P - 3._P / ghost(ix,iy)) - 1._P
    else
      z_ghost(ix, iy) = -2.0
    endif
  endif
endif

enddo
enddo

end subroutine iso_z
!
! Cette subroutine determine le spectre issu du disque
!   emettant dans une raie monochromatique
!
subroutine spectre_b(r, z_aux, spec)
  implicit none

  real(P), dimension(-DIM_IM:DIM_IM,-DIM_IM:DIM_IM), &

```

```

        intent(in) :: r, z_aux
real(P), dimension(-DIM_IM:DIM_IM,-DIM_IM:DIM_IM) :: z
real(P), dimension(NB_NU), intent(out) :: spec
real(P) :: pas, nu_min, nu_max
integer :: case, ix, iy

write(*,*) "Calcul du spectre (raie monochromatique)"

z = z_aux + 1._P

nu_min = 1._P / maxval(z, mask = z > 0.) - 0.01_P
nu_max = 1._P / minval(z, mask = z > 0.) + 0.01_P

write(*,*) "nu min = ", nu_min, "nu max = ", nu_max

spec = 0.
pas = (nu_max - nu_min) / real(NB_NU, P)

do iy = -DIM_IM, DIM_IM
do ix = -DIM_IM, DIM_IM
    !
    ! Le cas 1. + z = -2. correspond au cas ou le photon n'atteint
    !      pas le disque
    !
    if (z(ix,iy) /= -1._P) then
        !
        ! On determine la case du spectre qui va etre incrementee
        !      (celle qui annule la dirac dans l'integrale donnant
        !      le flux)
        !
        case = floor((1._P/(z(ix,iy)) - nu_min) /pas) + 1
        !
        ! En fait on calcule le flux multiplie par une frequence
        !      caracteristique : on obtient le spectre
        !
        spec(case) = spec(case) + r(ix,iy)**Q / (z(ix,iy))**4
    endif
enddo
enddo

end subroutine spectre_b

```

```
end module photo
```

## B.4 Module spectre

Ce module contient les procédures suivantes :

- subroutine **alpha\_gamma**
- subroutine **integ\_angl**
- subroutine **rayon\_angl**
- subroutine **redshift\_angl**
- subroutine **spectre\_angl**

```
module spectre
```

```
  use precision, only : P => DP
  use constantes
  use runge_kutta
  use photo
```

```
contains
```

```
  !
  ! Cette subroutine prend l'angle ''theta'' en entree
  !       et retourne un tableau de ''alpha'' et de ''gamma''
  !
  subroutine alpha_gamma(theta, alpha, gamma)
    implicit none

    real(P), intent(in) :: theta
    real(P), dimension(NB_ANGL), intent(out) :: alpha
    real(P), dimension(NB_ANGL), intent(out) :: gamma

    real(P) :: alph, cosgam
    integer :: i

    alpha = 0.
    gamma = 0.
    !
    ! On remplit le tableau d'alpha de facon a ce qu'il y ait
    !   de plus en plus de pas au fur et a mesure que b
```

```

!    augmente (NB_B = NB_ANGL => nombre de case en
!    alpha egale a l'indice de b)
!
do i=1, NB_ANGL
  alph = PI * real(i-1, P) / real(NB_ANGL, P)
  cosgam = cos(alph) / sqrt(cos(alph)**2 + 1._P / &
    tan(theta)**2)
  alpha(i) = alph
  gamma(i) = acos(cosgam)
enddo

end subroutine alpha_gamma
!
! Cette subroutine prend un parametre d'impact 'b', ainsi
! qu'un tableau de 'gamma', et retourne le tableau des rayons
! d'impact sur le disque 'rimpact'
!
subroutine integ_angl(b, gamma, rimpact)
  implicit none

  real(P), intent(in) :: b
  real(P), dimension(NB_ANGL), intent(in) :: gamma

  real(P), dimension(NB_ANGL), intent(out) :: rimpact

  real(P) :: h, h1, phi
  real(P), dimension(2) :: u, diff
  real(P), dimension(2,N) :: yout
  real(P), dimension(N) :: xout

  integer :: i, j, k
  logical :: adapt_pas

  !
  ! Initialisation des variables
  !
  rimpact = 0.
  adapt_pas = .true.

  u(2) = 0._P
  u(1) = 1._P / b
  phi = 0._P

```

```

h = 0.01_P
h1 = 0.00001_P
h = h1
!
! L'indice "i" est pour incrementer l'angle phi ;
! l'indice "j" est pour se balader dans le tableau "alga"
!
i = 1
j = 1
do
  call derivs(phi, u, diff)
  call rk_4(u, diff, 2, phi, h, yout(:,i))
  u = yout(:,i)
  xout(i) = phi

  phi = phi + h

  if (phi > gamma(j+1)) stop "pas h trop grand"

  if ((phi >= gamma(j)) .and. (phi < gamma(j+1))) then
    if ((1._P / yout(2,i) >= R_MIN) .and. &
        (1._P / yout(2,i) <= R_MAX)) then
      rimpact(j) = 1._P / yout(2,i-1)
    else
      rimpact(j) = 0.
    endif

    j = j + 1
  endif
  !
  ! Cas ou le photon "tombe" sur le trou noir
  !
  if (1._P / yout(2,i) < 3._P) then
    !
    ! Le photon tombe irremediablement...
    !
    rimpact(j:Nb_Angl) = 0.
    exit
  endif
  if (rimpact(j) < 0.) write(*,*) "rayon inside", rimpact(j)
  if (j == Nb_Angl) exit
  i = i + 1

```

```

        enddo

end subroutine integ_angl
!
! Cette subroutine retourne un tableau de ''rayon'' correspondant
! a toute l'image en coordonnees polaires
!
subroutine rayon_angl(theta, gamma, rayon, parim)
    implicit none

    real(P), intent(in) :: theta
    real(P), dimension(NB_B,-NB_ANGL:NB_ANGL), intent(out) :: rayon
    real(P), dimension(NB_ANGL), intent(in):: gamma
    real(P), dimension(NB_B), intent(out) :: parim
    real(P), dimension(NB_B,NB_ANGL) :: rayon_aux

    integer :: ib, i

    rayon = 0.
    rayon_aux = 0.

    do ib = 1, NB_B
        parim(ib) = real(R_MAX, P) / real(NB_B, P) * real(ib, P)
        call integ_angl(parim(ib), gamma, rayon(ib,1:NB_ANGL))
            ! rayon = rayon atteint par le photon
            !          sur le disque d'accretion
    enddo
    !
    ! L'image est symetrique par rapport a l'axe Y
    !          = axe alpha = 0
    !
    do i=1, NB_ANGL
        rayon(:, -i) = rayon(:, i)
    enddo
    !
end subroutine rayon_angl
!
! Retourne le decalage spectral
!
subroutine redshift_angl(theta, rayon, parim, alpha, gamma, redshift)
    implicit none

```

```

real(P), intent(in) :: theta

real(P), dimension(NB_B,-NB_ANGL:NB_ANGL), intent(in) :: rayon
real(P), dimension(NB_ANGL), intent(in) :: alpha
real(P), dimension(NB_ANGL), intent(in) :: gamma
real(P), dimension(NB_B), intent(in) :: parim
real(P), dimension(NB_B,-NB_ANGL:NB_ANGL), intent(out) :: redshift

real(P) :: sinal, aux1, aux2, aux3
integer :: ib, ial

write(*,*) "    Calcul du redshift"

redshift = 0.

do ib = 1, NB_B
do ial = -NB_ANGL, NB_ANGL
    !
    ! La case ial = 0 est ignoree
    !
    if (ial == 0) cycle
    if ((rayon(ib,ial) /= 0.) .and. (rayon(ib,ial) /= 3._P)) then
        sinal = sign(1,ial) * sin(alpha(abs(ial)))
        aux1 = sin(theta) * sinal / sqrt(rayon(ib,ial)**3)
        aux2 = sqrt(1._P - 3._P / rayon(ib,ial))
        aux3 = 1._P + aux1 * parim(ib)
        !
        ! redshift contient en fait 1+z
        !
        redshift(ib, ial) = aux3 / aux2
    endif
enddo
enddo

end subroutine redshift_angl
!
! Cette subroutine determine le spectre issu du disque
!   emettant dans une raie monochromatique
!
subroutine spectre_angl(theta, spec, nu_min, nu_max)
    implicit none

```

```

real(P), intent(in) :: theta
real(P), dimension(NB_B,-NB_ANGL:NB_ANGL) :: r, z
real(P), dimension(NB_NU), intent(out) :: spec
real(P), intent(out) :: nu_min, nu_max
real(P), dimension(NB_B) :: parim
real(P), dimension(NB_ANGL) :: alpha
real(P), dimension(NB_ANGL) :: gamma

real(P) :: pas, jacobien
integer :: case, ib, ial

write(*,*) "Subroutine 'alpha_gamma'"
call alpha_gamma(theta, alpha, gamma)
write(*,*) "Subroutine 'rayon_angl'"
call rayon_angl(theta, gamma, r, parim)
write(*,*) "Subroutine 'redshift_angl'"
call redshift_angl(theta, r, parim, alpha, gamma, z)
write(*,*) "Calcul du spectre (raie monochromatique)"

spec = 0.

nu_min = 1._P / maxval(z, mask = z > 0.) - 0.01_P
nu_max = 1._P / minval(z, mask = z > 0.) + 0.01_P

write(*,*) "nu min = ", nu_min, "nu max = ", nu_max

pas = (nu_max - nu_min) / real(NB_NU, P)

do ib = 1, NB_B
do ial = -NB_ANGL, NB_ANGL
!
! Le cas ial == 0 est ignoree
!
if (ial == 0) cycle
!
! Le cas z = 0. correspond au cas ou le photon n'atteint
!     pas le disque
!
if (z(ib,ial) /= 0. .and. r(ib,ial) /= 0.) then
!
! On determine la case du spectre qui va etre incrementee
!     (celle qui annule la dirac dans l'integrale donnant

```

```

!      le flux)
!
case = floor((1._P / z(ib,ial) - nu_min) / pas) + 1
if (case < 1 .or. case > NB_NU) then
  write(*,*) case, "Pb dans le calcul des cases du spectre"
else
  jacobien = parim(ib)

!
! En fait on calcule le flux multiplie par une frequence
!   caracteristique : on obtient le spectre
!
spec(case) = spec(case) + r(ib,ial)**Q * jacobien / &
              z(ib,ial)**4

endif
endif
enddo
enddo
end subroutine spectre_angl

end module spectre

```

## B.5 Modules constantes et precision

```

module PRECISION
  integer, parameter :: SP = kind(1.), DP = kind(1.D0)
end module PRECISION

module constantes
  use precision
  !
  real(DP), parameter :: tolerance = 16._DP * EPSILON(1._DP)
  real(DP), parameter :: PI = 3.1415926535897932384_DP
  !
  ! Nombre de pas d'integration pour le calcul des geodesiques
  !
  integer, parameter :: N = 40000
  !
  ! Nombre de pixels sur le quart de l'image

```

```

!      (On aura en tout (2 * DIM_IM + 1)^2 pixels
!
integer, parameter :: DIM_IM = 500
!
! Valeur du parametre d'impact critique
!
real(DP), parameter :: R_S = 5.1961524227_DP    ! SQRT(27)
!
! Rayons internes et externes du disque
!
real(DP), parameter :: R_MIN = 5._DP, R_MAX = 30._DP
real(DP), parameter :: MARGE = 1.01_DP
!
! Parametres pour le trace du spectre
!
integer, parameter :: NB_NU = 200
integer, parameter :: Q = -2
!
! Calcul du spectre :
!      Nombre de points entre
!      alpha = 0. et alpha = PI
!
integer, parameter :: NB_ANGL = 600
!
!      Nombre de points pour le
!      rayon (parametre d'impact)
!
integer, parameter :: NB_B = 1000
!
end module constantes

```

## B.6 Module graphique

Ce module contient les procédures suivantes :

- subroutine **trace\_geo**
- subroutine **trace\_defl**
- subroutine **image\_iso\_r**
- subroutine **trace\_spectre**

Il ne nous semble pas utile de mettre le listing (assez long) de ce module ici ; en fait il n'a servi qu'à produire les figures qui illustrent ce rapport.

## B.7 Module ppal

```
program ppal

use precision, only : P => DP, SP
use constantes
use geodesique
use graphique
use spectre

    implicit none

    real(P) :: theta
    integer :: i

!   call trace_geo ()
!   call trace_defl ()
!
!   Saisie de l'angle de prise de vue (colatitude)
!
write(*,*) "Donner l'angle de vue THETA (en degre) = "
read(*,*) theta

theta = theta * PI / 180._P

!   call image_iso_r(theta)

    call trace_spectre(theta)

end program ppal
```