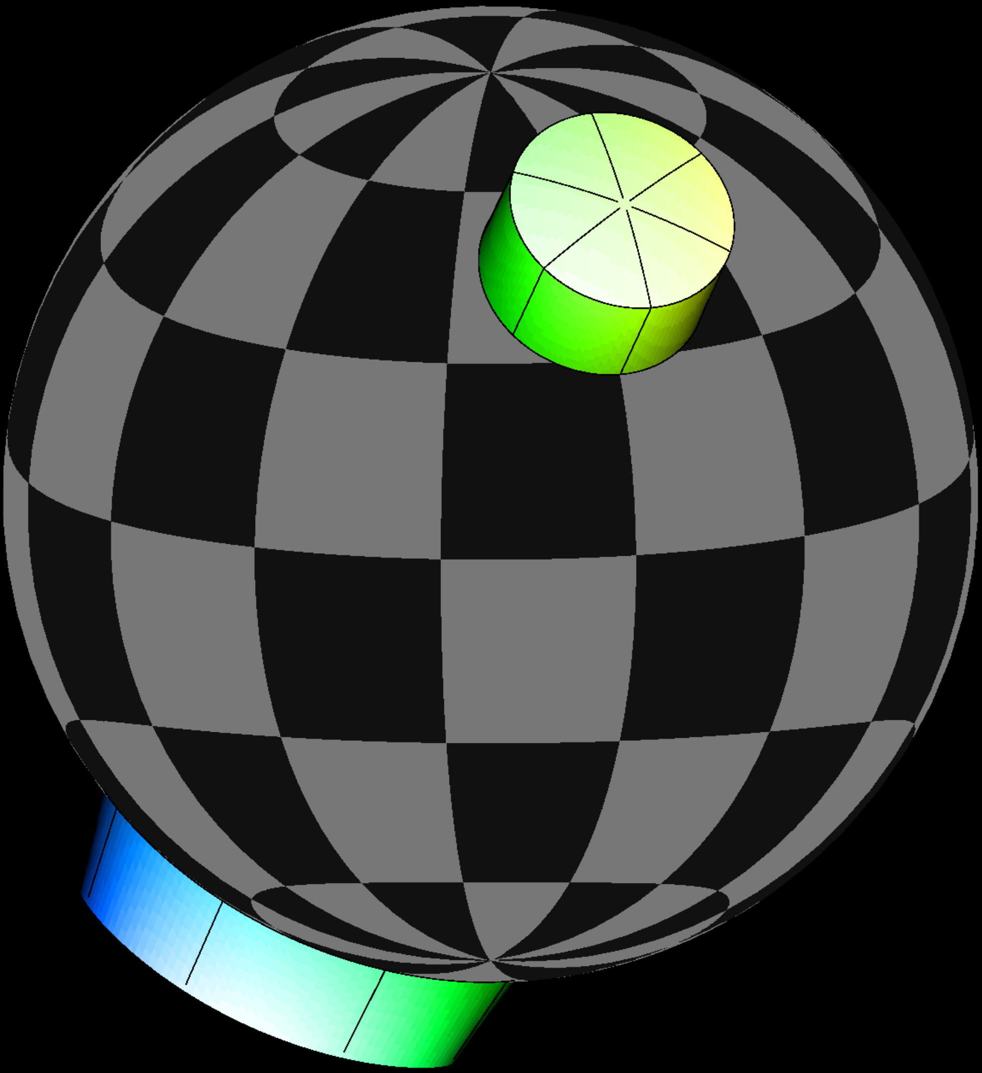
Light Bending around neutron stars

Sebastian Falkner



Light Bending around neutron stars

Masterarbeit aus der Physik
Vorgelegt von

Sebastian Falkner

Bamberg, 28.03.2012

Dr. Karl Remeis Sternwarte Bamberg
Astronomisches Insitut der
Friedrich-Alexander-Universität Erlangen-Nürnberg

Betreuer: Prof. Dr. Jörn Wilms

Image on the Titlepage: *Simulation of a counterclockwise spinning, accreting neutron star as seen by a distant observer. Photons are only emitted from the accretion columns homogeneously and isotropically. The Brightness represents the intensity of the flux and the color the Doppler shift.*

Contents

Deutsche Zusammenfassung

Die vorliegende Masterarbeit beschäftigt sich mit der Entwicklung eines Programms, welches den beobachteten Fluss von Neutronensternen mit Akkretionssäulen oder *Hot Spots* berechnet. Dabei basieren die Berechnungen des Flusses auf beliebig definierbaren Abstrahlungsprofilen und berücksichtigen speziell und allgemein relativistische Effekte.

Berücksichtigt werden dabei die speziell relativistischen Effekte der Doppler Verschiebung und der Aberration, und die, der allgemein relativistischen Rotverschiebung und Lichtkrümmung. Der Fokus liegt auf dem allgemein relativistischen Effekt der Lichtkrümmung und sein Einfluss auf den beobachteten zeitabhängigen Fluss bzw. dem Pulsprofil. Dieses zeigt die periodische Variabilität des Flusses über die Rotationsperiode des Neutronensterns. Allgemein wird diese periodische Variabilität mit der Beschränkung der Strahlung auf bestimmte Regionen (Akkretionssäule oder *Hot Spots*) auf der Oberfläche des Neutronensterns erklärt, die man durch die Rotation unter unterschiedlichen Winkeln betrachtet. Dabei bündelt das starke Magnetfeld das einfallende Plasma, welches seine kinetische Energie dann an den magnetischen Polen auf der Oberfläche des Neutronensterns in Form von Strahlung abgibt. Da Neutronensterne sehr massereich und kompakt sind, hat der Lichtkrümmungseffekt einen beträchtlichen Einfluss, zum Beispiel ist bis zu 85% der Neutronensternoberfläche sichtbar.

Das entwickelte Programm kann jedes beliebige Pulsprofil berechnen, besonders das von langsam rotierenden Neutronensternen mit Rotationsfrequenzen $\lesssim 1$ Hz, zum Beispiel, *High Mass X-ray Binaries*. Es ist möglich zwischen sphärischen Neutronensternen mit zylindrischen Akkretionssäulen oder mit *Hot Spots* zu wählen. Jedoch ist das Programm in der Lage jede beliebig definierte Akkretionssäulengeometrie zu prozessieren. Zusätzlich ist es möglich beliebige Abstrahlungsprofile für jede einzelne Region des Objekts zu definieren. Dies erlaubt es theoretische Modelle unter dem Einfluss relativistischer Effekte, insbesondere der Lichtkrümmung, zu testen. Die Berechnungen der Lichtkrümmung basiert dabei auf der Schwarzschild Metrik. Die entsprechenden Gleichungen können entweder mit numerischen Verfahren oder einer analytischen Näherung (Beloborodov, 2002) für Radien größer als zwei Schwarzschild Radien gelöst werden. Die meisten Neutronensterne besitzen jedoch größere Radien. Darüber hinaus ist eine Erweiterung für kleinere Radien bis hin zum Schwarzschild Radius möglich.

Weiterhin ist es möglich das Programm für höhere Frequenzen zu erweitern. Dafür muss nur zusätzlich die durch die spezielle Relativität gegebene Zeitverzögerung bzw. Phasenversatz der beobachteten Photonen berücksichtigt werden. Dieser Effekt hat besonders für hohe Rotationsfrequenzen einen großen Einfluss. Diese Erweiterung ist besonders für Millisekunden Pulsare interessant, da diese Rotationsfrequenzen bis zu 700 Hz haben können.

In dieser Arbeit werden Pulsprofile von Neutronensternen genauer untersucht, die Rotationsfrequenzen $\lesssim 1$ Hz besitzen und von den Akkretionssäulen abstrahlen. Dies geschieht für verschiedenste Konfigurationen der Akkretionssäulen, d.h. für unterschiedliche Lagen, Größen, Inklinationen des Systems zum Beobachter und unterschiedlichen Abstrahlprofilen. Dabei sind die Akkretionssäulen aus einer zylindrischen Hülle und einer sphärischen Kappe zusammengesetzt und deren Abstrahlung im Einzelnen betrachtet. Diese Pulsprofile zeigen, dass die Lichtkrümmung ihre Form sehr beeinflusst, aber keinerlei Asymmetrien verursacht. Asymmetrien in Pulsprofilen langsam rotierender Neutronensterne können nur mit asymmetrischen Abstrahlprofilen oder asymmetrischer Lage der Akkretionssäulen erklärt werden.

Abstract

The topic of this thesis is the development of a program, which determines the observed flux of a neutron star with accretion columns or hot spots. The calculation of the flux is based on an arbitrary given emission profile and accounts for special and general relativistic effects.

The determination of the flux considers the special relativistic effect of the Doppler shift and aberration, and the general relativistic effect of gravitational redshift and light bending. The main focus is on the general relativistic light bending effect and its impact on the observed time dependent flux, the pulse profile. The pulse profile shows the periodic variability of the flux over the rotation period of the neutron star. This periodic variability is commonly explained by the localized emission regions (accretion columns or hot spots) on the neutron star seen under different angles due to the rotation. Thereby, the strong magnetic field collimates incoming plasma, which releases the obtained kinetic energy on the surface of the neutron star at the magnetic pols in form of radiation. As neutron stars are very massive and compact, the light bending effect has a considerable influence, e.g., one can see about 85% of the surface of a neutron star.

The program developed in this thesis can calculate any arbitrary pulse profile, especially of slow rotating neutron stars with spin frequencies $\lesssim 1$ Hz, e.g., High Mass X-ray Binaries. Therefore, the program provides the possibility to choose spherical neutron stars with cylindrical accretion columns or with hot spots. Furthermore, the program is capable to process arbitrary defined accretion column geometries. Additionally, it is possible to define arbitrary emission profiles for each individual part of the object, which allows to test theoretical models under the influence of relativistic effects, especially the effect of light bending. The determination of the light bending effect are based on the Schwarzschild metric. The program provides the possibility to solve the according equations numerically or with an analytically approximation (Beloborodov, 2002) for radii greater than two Schwarzschild radii. However, most neutron stars exhibit a greater radius. Nevertheless, an extension to even lower radii, down to the Schwarzschild radius, is also possible.

Further, the program was designed in a way that it is possible to extend it also for higher spin frequencies. This extension only requires to additionally account for the special relativistic effect of the time delay causing a phase shift of the observed photons, which has a considerable influence for higher frequencies. Especially for Millisecond Pulsars, which can exhibit spin frequencies up to 700 Hz this extension is of interest.

In this thesis the pulse profile of neutron stars with spin frequencies $\lesssim 1$ Hz are focused on, where the emission comes from the accretion columns. This investigation is done for various settings for the accretion columns, namely their position on the neutron star surface, their size, the inclination of the system to the observer and different emission profiles. Thereby, the accretion columns are composed of a cylindrical column and a spherical cap and the emission from these individual parts is distinguished between and discussed individually. The investigated pulse profiles show that the light bending has a considerable influence on their shape, but it does not introduce any asymmetry. Namely, asymmetries in pulse profiles of slowly rotating neutron stars can only be explained by asymmetric emission profiles or asymmetric accretion column positions on the surface of the neutron star.

Chapter 1

Introduction

With the theory of special relativity (1905), Albert Einstein has revolutionized our understanding of space and time in the sense that both are relative rather then absolute constructs. Meaning that observers in different inertial frames would measure different times and lengths for the same event. Space and time are indeed closely intertwined, which led to the introduction of the four dimensional spacetime, in which space and time are united and invariants can be defined. In 1915, Einstein presented the final version of his theory of general relativity, which also accounts for gravity and describes the interaction of spacetime and matter. "Matter tells spacetime how to curve, and curved space tells matter how to move" is how Wheeler (1990) puts it. Also Einstein's theory of general relativity already could successfully explain the perihelion shift of mercury, which has not been achieved before, it only got commonly accepted after Arthur Eddington experimentally confirmed (Dyson et al., 1920) the value of the deflection of light rays observed from stars close to the sun predicted by Einstein's theory.

1.1. Light Bending

We are used to the idea that the trajectory of massless particles (e.g., photons) is a straight line. In general relativity, however, a more general definition of such trajectories has to be made. Geodesics describe the path of the shortest distance between two points, which are not necessarily straight. For example, if we move from point A to point B on the surface of the earth in a direct way, our path is not a straight line as we are moving on a curved surface. The shortest path we can choose to get from one place on the earth to another is along a great circle connecting these points, whereas a straight line connecting both points would go through the earth. Similar to the curved surface of the earth, in general relativity the spacetime can be curved and therefore the movement of particles is more generally described by geodesics. The shape of the geodesics are determined by the curvature of spacetime (more precisely by the underlying metric), which is determined by the mass distribution. The more mass is concentrated in a particular area, the higher is the curvature of the spacetime in its vicinity. Hence, the trajectory of photons passing by a massive object will show a certain deviation from a straight path, which is called the gravitational light bending effect.

For experimental evidence Einstein suggested to measure the deflection angle of light rays from distant stars closely passing by the sun[1] and calculated the value of this deflection predicted by his theory of general relativity. As seen in Fig. 1.1, the path of the starlight gets bend towards the center of mass, while passing by the sun. Hence, for a distant observer

[1]This measurement is only possible during an eclipse, as otherwise the light of the sun is too bright.

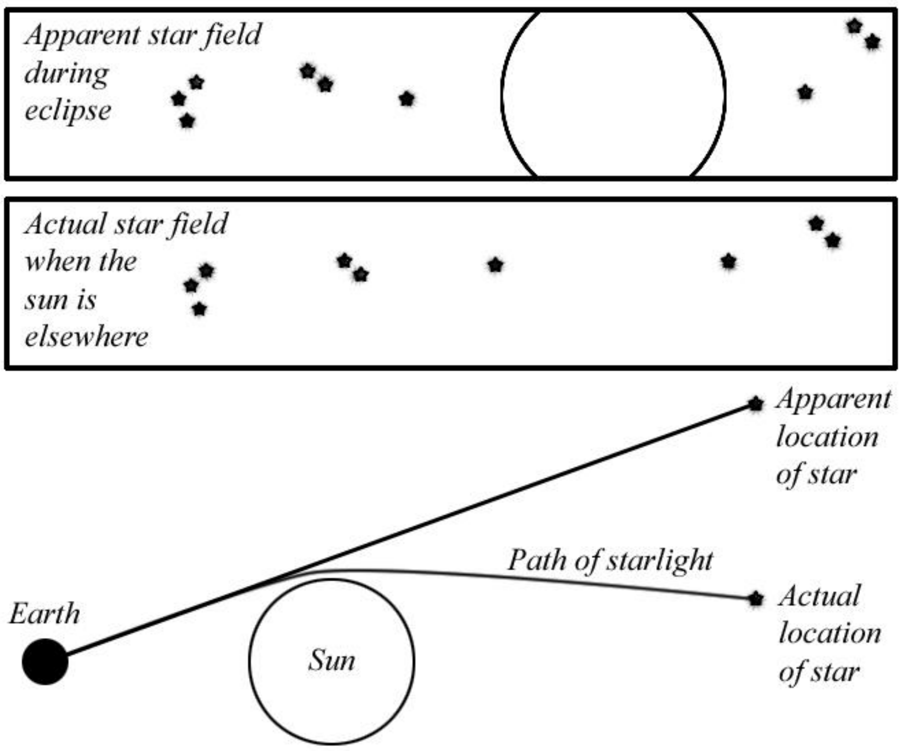

Apparent star field during eclipse
Actual star field when the sun is elsewhere
Apparent location of star
Path of starlight
Actual location of star
Earth
Sun

Fig. 1.1.: *Illustration of the deflection of star light in the vicinity of the sun. Top: Star field around the sun. Middle: Same star field as in the top panel in absence of the sun. Bottom: Comparison of the real position of a star and its apparent position. (Credits:* `http://cosmictimes.gsfc.nasa.gov/`*)*

the apparent position of a star, whose light got bended, seems to be further away from the bending mass than it really is. Comparing the first two panels in Fig. 1.1, in which a star field is shown in the presents of the sun and without, it is noticeable that the difference of the star positions with and without the presence of the sun increases the closer the star is to the center of the sun.

The bending of light not only occurs for photons passing by a massive object, but also for photons emitted from it. Depending on the mass of the object, which determines the strength of the light bending, this effect can have a considerable influence on observations of such an object, like, e.g., neutron stars.

1.2. Neutron Stars

In this section a brief discussion of neutron stars is given with the focus on those properties important for this thesis. For a more detailed discussion see, e.g., Fürst (2011) and references therein, which the following section is mainly based on.

Neutron stars are the remnants of dying stars with masses $M > 8\ \mathrm{M}_\odot$ (Smartt, 2009), where $\mathrm{M}_\odot$ is the mass of the sun. In the final stages the iron core of those dying stars reaches a critical mass. The iron core is the ash of the last possible stage of nuclear fusion burning lighter to heavier elements, which provides the necessary energy, in the form of outwards directed

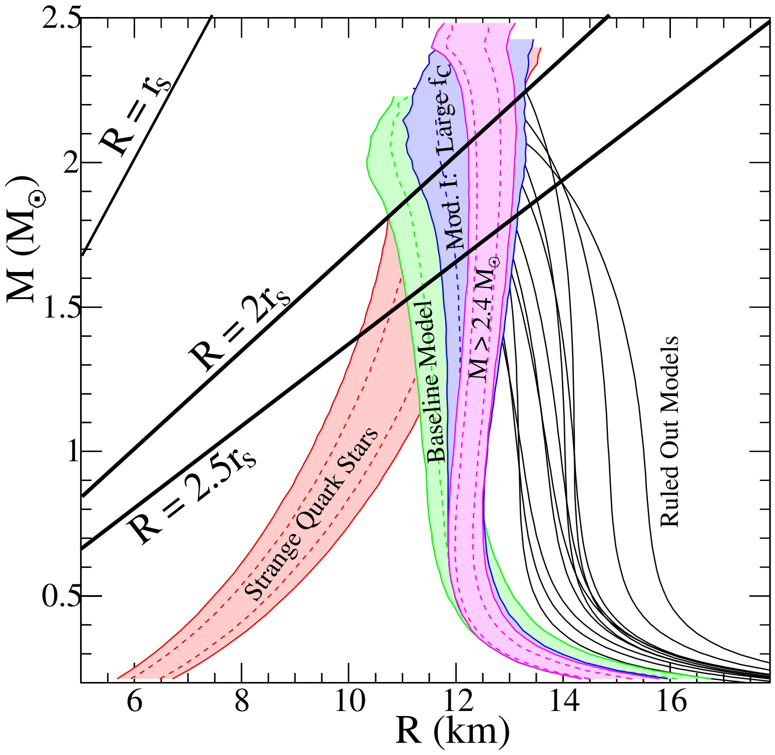


Fig. 1.2: *Possible mass to radius relation for neutron stars.* r_s *relates to the Schwarzschild radius, which depends on the mass* M *only (see Sec. 2.1), where* $r_s \approx 3\ \mathrm{km} \cdot \left(\frac{M}{\mathrm{M}_\odot}\right)$. *Figure adapted from Steiner et al. (2013).*

pressure[2], to balance the gravitational force. As the fusion of iron and heavier elements requires energy instead of releasing it, the only force within the iron core counteracting the gravitation is due to the degeneracy of the electrons. Once the iron core exceeds a critical mass, beyond which the gravitation overcomes the pressure of the electron degeneracy, the collapse of the core sets in. In this process the outer layers of the star get expelled, whereas neutrons are generated via the inverse β-decay in the core, while it is collapsing under its own gravitation. The collapse can be stopped by the degenerate pressure of the created neutron gas resulting in a neutron star, otherwise a black hole is formed. The created neutron stars are very massive, holding a mass of ~1.4 $\mathrm{M}_\odot$ within a radius of only ~10 km, which results in a considerable impact on the curvature of spacetime in its vicinity. The structure of the interior of a neutron star is very complex and not completely understood yet. However, Steiner et al. (2013) calculated constraints on possible mass to radius ratios based on various theoretical models seen in Fig. 1.2. This figure shows that most mass to radius ratios theories predict for neutron stars relate to radii greater than two Schwarzschild radii, where the Schwarzschild radius is given by

$$r_\mathrm{s} \approx 3\ \mathrm{km} \cdot \left(\frac{M}{\mathrm{M}_\odot}\right) \tag{1.1}$$

The knowledge of the mass is necessary to determine the curvature of spacetime (see Sec. 2.1). Furthermore, the radius of the neutron star is required to predict the degree of bending photons experience, as the bending depends also on the radius, at which they are emitted (see Sec. 2.2).

Another interesting property many neutron stars exhibit is a periodic variability in their observed flux over time, where the period of this variability corresponds to the rotation period of the spinning neutron star. Figure 1.3 shows this variability in form of a pulse profile

[2]This pressure can be provided by radiation or gas and depends on the exact conditions within the star.

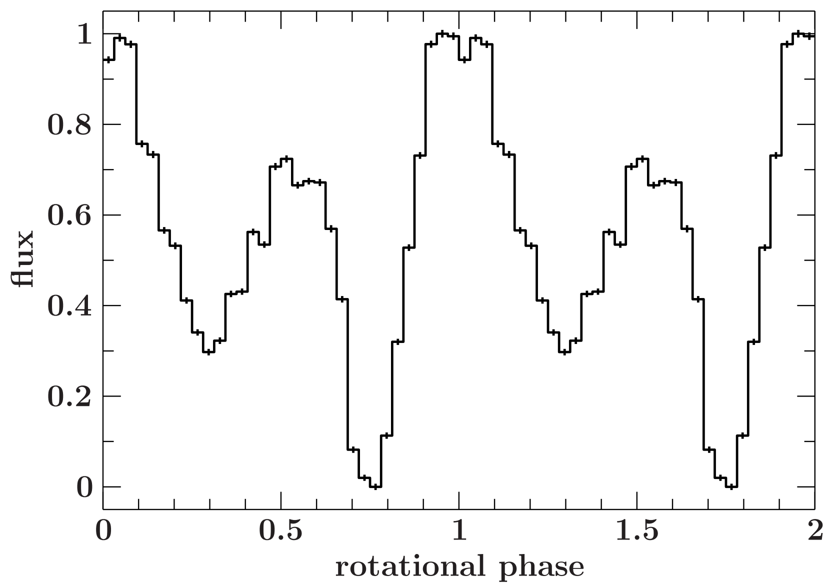


Fig. 1.3: *A measured pulse profile of A0535+262 (priv. com. M. Kühnel). The flux is normalized to its minimal and maximal value.*

exhibiting two characteristic peaks[3] in flux separated by about half a rotation phase. This phenomenon is commonly explained by the extremely strong magnetic fields (Lamb et al., 1973) neutron stars can exhibit, which reach up to the order of 10^{12}-10^{15} G (Haensel et al., 2007) and can be approximated by a dipole. Such a strong magnetic field has a large influence on the surroundings of the neutron star. As shown in Fig. 1.4, plasma (ionized material) attracted by the gravitational force of the neutron star at a certain point, the Alfvén radius, has to follow the magnetic field lines on its way down onto the surface of the neutron stars. The resulting collimation of the incoming matter leads to the formation of accretion columns on the surface of the neutrons star at the magnetic poles.

Furthermore, the kinetic energy the material obtained by transforming the gravitational energy, is released in the bottom parts of those accretion columns, mainly in form of photons in the X-ray regime, e.g., produced by bremsstrahlung as the particles are stopped on the neutron star surface, thermal radiation from the hot spots or cyclotron radiation processes[4]. Note, however, that the bending of light is independent of the photon energy (see Sec. 2.2), but does depend on the location of the emission. Therefore, the geometry of the accretion column is very important. The geometry, however, depends on several conditions, e.g., the coupling of the matter to the magnetic field lines and the accretion rate. The simplest geometry assumable is a cylindrical or cone like filled accretion column (see, e.g., Becker & Wolff, 2007). Nevertheless, also more complex geometries are thinkable, e.g., hollow cylinders or cones (Kraus, 2001). The formation of the accretion columns depends on the accretion rate. The supply of sufficient amounts of matter is only possible in binary systems, in which there is a normal companion star[5] besides the neutron star. In Low Mass X-Ray Binaries (LMXBs) the main accretion channel is the Roche lobe overflow of the companion star, whereas in High Mass X-ray binaries (HMXBs) it is the stellar wind. These systems are defined by the mass of their companion star. In HMXBs the companion stars are very massive compared to those in LMXBs. As the lifespan of stars decrease dramatically with their mass HMXBs

[3]The pulse profile in Fig. 1.3 is just representative, there are also pulse profiles with a more complex shape.
[4]The exact mechanisms of the radiation processes are very complex and beyond the scope of this thesis. For a deeper discussion of this topic see, e.g., Becker & Wolff (2007).
[5]Normal star in the sense of stars, which still perform nuclear fusion.

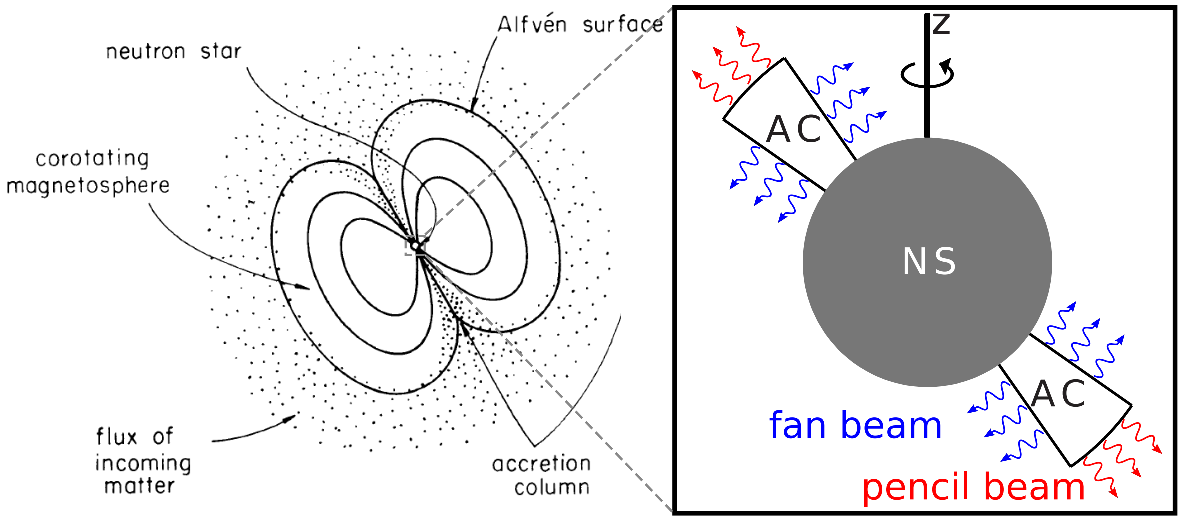


Fig. 1.4.: *Sketch of a neutron star with accretion columns. The left side (Lamb et al., 1973) shows a neutron star with its magnetosphere. The dots represent matter, which is falling onto the neutron star. At the Alfvén surface the ionized matter follows the magnetic field lines to the magnetic pols on the surface of the neutron star. The focused matter stream causes the formation of accretion columns (AC) as seen in the zoom in. Emission from the accretion column walls is called fan beam and that from the cap pencil beam. The rotation axis is denoted by z.*

are very young systems. The lifespan of the massive companion star in HMXBs is short, i.e., the progenitor of the neutron star, which evolved even faster, had do be more massive. The young HMXBs commonly show stronger magnetic fields, which in combination with the high accretion rates leads to formation of accretion columns especially in these systems.

So far we have seen that the strong magnetic fields of neutron stars combined with a sufficient accretion rate leads to localized regions on the neutron star producing a considerable amount of emission. If we additionally consider the rotation of the neutron star, we can explain the observed pulse profile of those neutron stars seen in Fig. 1.3. While the neutron star is spinning around its rotation axis, we see different parts of the accretion columns under different angles, as the axis of the magnetic dipole is not necessarily aligned with the rotation axis, leading to a periodical variability of the observed flux with the rotation period. Figure 1.4 shows the accretion columns on the neutron star, composed of a cap and a wall, where the emission from the caps, called pencil beam, is mostly directed along the magnetic field lines, whereas the emission from the walls, called fan beam, is mostly rectangular to it.

The emission originates from regions close to the surface of the neutron star and therefore the light bending effect has to be considered in order to make quantitative statements about the shape of the pulse profiles.

1.3. The Aim of this Master Thesis

The topic of this thesis is the development of a program, which determines the observed flux of a neutron star with accretion columns, based on an arbitrary given emission profile and accounting for special and general relativistic effects. The main focus is on the effect of light bending within general relativity and its impact on the observed flux. Not only the overall flux is of interest, but also the flux of the individual regions contributing to it, for example, the cap and the wall of the individual accretion columns. Furthermore, the aim is to keep this program as common as possible, i.e., to avoid constraints of any kind, which limit the functionality or its extensibility.

The theoretical equations of the light bending effect discussed in Sec. 2.2 are based on the Schwarzschild metric (Sec. 2.1). In Sec. 2.3 the determination of the observed flux is discussed, which accounts also for other relativistic effects besides the light bending, namely the gravitational redshift, Doppler shift and aberration.

Afterwards, Chap. 3 provides detailed information about the implementation of the theoretical equations and the structure of the program. Additionally, App. A provides an example code to demonstrate the usage, where an detailed list of all code related functions with description of its usage can be found in App. C. In App. B the storing of the results is explained, i.e., the file structure and the file format.

In Chap. 4 the pulse profile of neutron stars with emission from the accretion columns for various configurations are looked at under the aspect of general relativistic light bending. Finally, Chap. 5 provides a summary of the program, the conclusions drawn from the pulse profiles under the influence of light bending and the future prospects.

Chapter 2

Theoretical Background

The mathematical description of General Relativity is given by Einstein's field equation (Einstein, 1916), which includes a metric $g_{\mu\nu}$ containing the information about the curvature of spacetime, or in other words how distances has to be measured. One of the simplest, but nevertheless most important solutions the Einstein's field equation, is the **Schwarzschild metric**. In this chapter this solution will be discussed and its related equations will be motivated and partly deduced. A more detailed discussion can be found in any literature of General Relativity (see, e.g., Misner et al., 1973; Carroll, 2004; Hartle, 2003; Fließbach, 1990). All formula presented in the following are given in units such that the speed of light and the gravitational constant are equal to one ($c \equiv G \equiv 1$).

2.1. The Metric

The Schwarzschild metric is the unique spherically symmetric vacuum solution for Einstein's field equation found by Schwarzschild (1916). In this section this solution is briefly discussed primarily based on the description of Carroll (2004) and Hartle (2003). In spherical coordinates $x^\mu = (t, r, \theta, \phi)$ the according line element (see, e.g., Fließbach, 1990) is given by

$$\mathrm{d}s^2 = -\left(1 - \frac{2M}{r}\right)\mathrm{d}t^2 + \left(1 - \frac{2M}{r}\right)^{-1}\mathrm{d}r^2 + r^2\left(\mathrm{d}\theta^2 + \sin^2\theta\,\mathrm{d}\phi^2\right) \quad , \tag{2.1}$$

where the constant M can be interpreted as the mass of the gravitating object. The line element $\mathrm{d}s$ and the metric $g_{\mu\nu}$ are connected by the relation

$$\mathrm{d}s^2 = g_{\mu\nu}\mathrm{d}x^\mu\mathrm{d}x^\nu \quad . \tag{2.2}$$

Equation (2.1) is the vacuum solution and therefore only describes spacetime properly outside the object. The Schwarzschild metric is static in the sense that it is completely independent of the time coordinate and does not include time-space cross terms ($\mathrm{d}x^i\mathrm{d}t + \mathrm{d}t\mathrm{d}x^i$). The spherical symmetry can be seen by looking at the two-dimensional surface of constant t and constant r, reducing the line element (Eq. 2.1) to

$$\mathrm{d}\Sigma^2 = r^2(\mathrm{d}\theta^2 + \sin^2\theta\,\mathrm{d}\phi^2) \quad , \tag{2.3}$$

which describes a sphere of radius r in flat three-dimensional space. As the properties of the metric originate in the distribution of mass, the spherical symmetry also has to apply to this. In fact, the specific form of the mass distribution is irrelevant and can be assumed point like as long as it fulfills this criterion. The point mass relates to an infinite curvature of spacetime causing the occurrence of a real singularity in the metric at $r = 0$ as in Eq. 2.1 the metric coefficients become infinite. There is also a singularity at the Schwarzschild radius

$$r_\mathrm{s} = 2M \quad . \tag{2.4}$$

But unlike the singularity at $r = 0$, this one originates in the choice of coordinates and can be avoided by a transformation to more appropriate coordinates. In our case, however, the chosen coordinates are sufficient as we are only interested in radii exceeding the Schwarzschild radius.

Another important solution to Einstein's field equation is the Kerr metric (found by Kerr, 1963), which also accounts for effects inferred by rotation of the gravitating mass at the expense of the introduction of a new parameter. In the case of no rotation the Kerr metric converges to the Schwarzschild metric. The drawback of the Kerr metric in contrast to the Schwarzschild metric is the increased mathematical complexity. Additionally, for none point like objects rotation also entails the oblateness of the object. The exact degree of the oblateness requires knowledge about the equation of state of the object, which for neutron stars is barely known. Cadeau et al. (2007) have shown that for fast rotating oblate neutron stars the effects of the oblateness dominate by far those of the choice of metric and that for spherical surfaces the differences between Kerr and Schwarzschild are insignificant. They come to the conclusion that in case of neutron stars there is no advantage of using the Kerr metric in general and therefore the calculations presented in the following are based on the Schwarzschild metric.

2.2. Equations of Motion

Having made the choice which metric to use, the next step is to deduce the equations of motion to describe the propagation of particles and photons parametrized with $x^\mu(\lambda)$. To achieve this, the Geodesic Equation (see, e.g., Fließbach, 1990)

$$\frac{\mathrm{d}^2 x^\kappa}{\mathrm{d}\lambda^2} + \Gamma^\kappa_{\mu\nu} \frac{\mathrm{d}x^\mu}{\mathrm{d}\lambda} \frac{\mathrm{d}x^\nu}{\mathrm{d}\lambda} = 0 \tag{2.5}$$

has to be solved. Thereby the Christoffel symbols $\Gamma^\kappa_{\mu\nu}$ depend on the metric $g_{\mu\nu}$ and their derivations. Analog to the equations of motion in classical mechanics, the Geodesic Equation (Eq. 2.5) can be derived within the Lagrangian formalism by minimizing the relativistic generalized action (Krolik, 1999)

$$S = \int_A^B \mathcal{L}\ \mathrm{d}\lambda = \int_A^B \mathrm{d}\lambda \left[g_{\mu\nu} \frac{\mathrm{d}x^\mu}{\mathrm{d}\lambda} \frac{\mathrm{d}x^\nu}{\mathrm{d}\lambda} \right]^{-\frac{1}{2}} \tag{2.6}$$

along the path $x^\mu(\lambda)$ by means of variation. The combination of the Schwarzschild metric (Eq. 2.1) with the corresponding solutions for the Christoffel symbols (see, e.g., Fließbach, 1990, Chap. 30) yields to the equations of motion for photons $u^\mu = \mathrm{d}x^\mu/\mathrm{d}\lambda$ (Pechenick et al., 1983):

$$u^0 \equiv \frac{\mathrm{d}t}{\mathrm{d}\lambda} = (1 - r_\mathrm{s}/r)^{-1} \tag{2.7}$$

$$u^1 \equiv \frac{\mathrm{d}r}{\mathrm{d}\lambda} = \left[1 - b^2 (1 - r_\mathrm{s}/r)/r^2\right]^{1/2} \tag{2.8}$$

$$u^2 \equiv \frac{\mathrm{d}\theta}{\mathrm{d}\lambda} = 0 \tag{2.9}$$

$$u^3 \equiv \frac{\mathrm{d}\Psi}{\mathrm{d}\lambda} = br^{-2} \tag{2.10}$$

To obtain these equations the different symmetries contained in the Schwarzschild metric were used. The invariance under time translation leads to conservation of energy and the spherical symmetry corresponds to conservation of angular momentum meaning the propagation of particles and photons takes place in a plane. Commonly the equatorial plane $\theta = \pi/2$ is chosen to which the coordinate system can be easily rotated, if the photon is not in this plane. Ψ relates to ϕ in the equatorial plane. The conservation of angular momentum manifests itself in the impact parameter b, which is constant over the whole photon trajectory.

2.2.1. Photon Trajectory

We are now interested in the actual trajectory embedded in the Schwarzschild spacetime photons follow. As the propagation takes place in a plane $R(\Psi)$ or $\Psi(R)$ fully describe the photon trajectory, where Ψ is the angle between the line of sight ($\Psi = 0$, escape direction) and the current position of the photon (see Fig. 2.1). By integrating the quotient of Eq. 2.10 and Eq. 2.8 $\Psi(R)$ can be derived (see, e.g., Beloborodov, 2002):

$$\begin{aligned} \Psi(R) &= \int_{\Psi(R)}^{\Psi(\infty)} \mathrm{d}\Psi = \int_R^\infty \frac{\mathrm{d}\Psi}{\mathrm{d}\lambda}\frac{\mathrm{d}\lambda}{\mathrm{d}r}\mathrm{d}r \\ &= \int_R^\infty \mathrm{d}r \frac{1}{r^2}\left[\frac{1}{b^2} - \frac{1}{r^2}\left(1 - \frac{r_\mathrm{s}}{r}\right)\right]^{-1/2} \end{aligned} \tag{2.11}$$

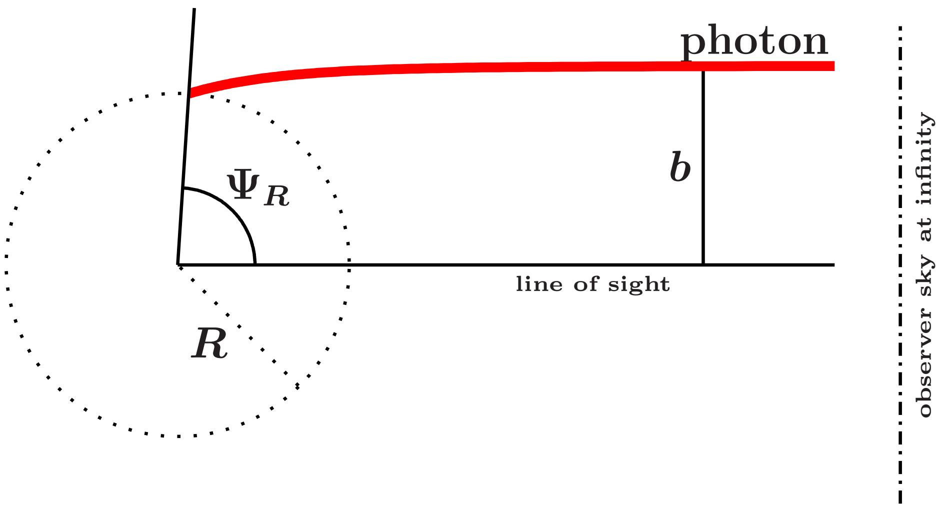


Fig. 2.1.: *Sketch of a photon trajectory starting at radius R, seen by an distant observer ($r \to \infty$, $\Psi = 0$) under the angle Ψ_R with an impact parameter b.*

Where the upper integration limit refers to the observer, who is set to be far away ($r \to \infty$). The lower limit R corresponds to the radius of the current photon position. As shown in Fig. 2.1, for $R \to \infty$ the impact parameter equals the distance between the line of sight and the point the photon hits the observer sky (plane rectangular to the line of sight for $r \to \infty$). Expressing this relation with a formula gives

$$\lim_{R\to\infty} R \sin(\Psi(R)) = b \quad . \tag{2.12}$$

To confirm this relation we look at Eq. 2.11 and simplify it for the case that $R \to \infty$. First we make the substitutions

$$u_r := \frac{r_\mathrm{s}}{r} \qquad \text{and} \qquad u := \frac{r_\mathrm{s}}{R} \quad , \tag{2.13}$$

with which Eq. 2.11 rewrites to

$$\Psi(u) = \int_0^u \mathrm{d}u_r \left[\frac{r_\mathrm{s}^2}{b^2} - u_r^2(1-u_r)\right]^{-1/2} \quad . \tag{2.14}$$

For $u_r \leq u \ll 1$ (i.e., $R \to \infty$) the following approximation can be done:

$$\begin{aligned}\widetilde{\Psi}(u) := \Psi(u)\Big|_{u_r \ll 1} &= \int_0^u \mathrm{d}u_r \left[\frac{r_\mathrm{s}^2}{b^2} - u_r^2 + \mathcal{O}(u_r^3)\right]^{-1/2} \\ &= \arcsin\left(u\frac{b}{r_\mathrm{s}}\right) \qquad (2.15)\\ \widetilde{\Psi}(R) &= \arcsin\left(\frac{b}{R}\right) \qquad (2.16)\end{aligned}$$

Using this simplification it can easily be seen that Eq. 2.12 is fulfilled.

Unfortunately the integral (Eq. 2.11) cannot be solved analytically. Nevertheless, the integrand of Eq. 2.11 reveals some information about the photon trajectory without evaluating the integral itself. Obviously there are cases in which the integrand is undefined, namely the radicand in Eq. 2.11 has to fulfill

$$\frac{r^4}{b^2} - r^2\left(1 - \frac{r_\mathrm{s}}{r}\right) > 0 \quad .$$

Multiplying with b^2/r and looking at the border case leads to the cubic equation

$$C(r) := r^3 - b^2 r + b^2 r_\mathrm{s} = 0 \quad , \tag{2.17}$$

where $r = 0$ has been ruled out as this corresponds to the singularity. The cubic equation (Eq. 2.17) has three potential solutions for the radius r (see, e.g., Press et al., 1992), which can be expressed as

$$r_\kappa(b) = -2\sqrt{\frac{b^2}{3}} \cos\left(\frac{\cos^{-1}\left(\frac{3\sqrt{3}}{2}\frac{r_\mathrm{s}}{b}\right) + \kappa \cdot 2\pi}{3}\right) \quad , \tag{2.18}$$

where $\kappa = -1, 0, 1$. These solutions only apply, if $b > b_\mathrm{c}$ with the critical impact parameter

$$b_\mathrm{c} = \frac{3\sqrt{3}}{2} r_\mathrm{s} = \sqrt{3} r_\mathrm{c} \quad , \tag{2.19}$$

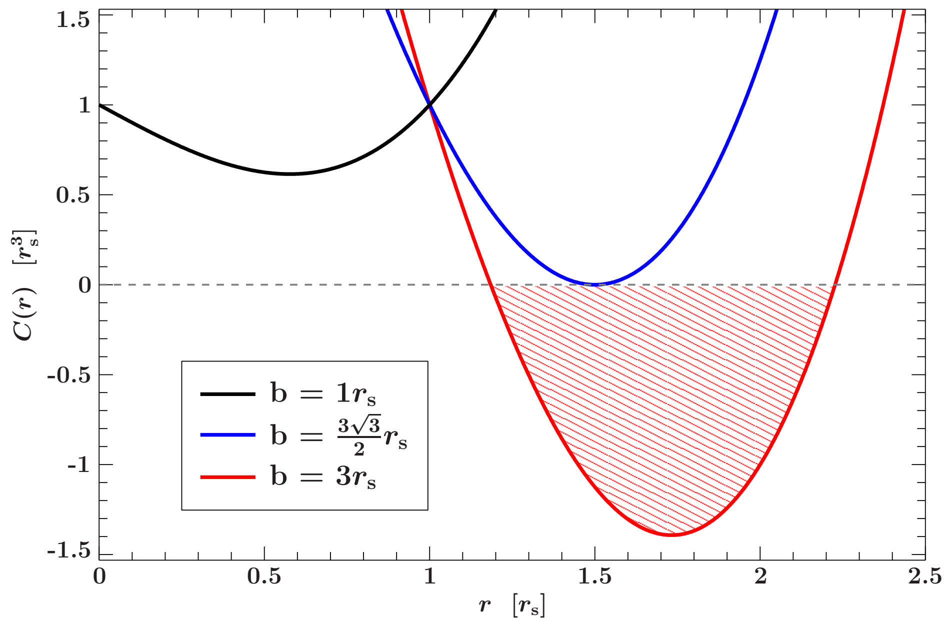


Fig. 2.2.: *Plots of $C(r)$ for three different values of b. $b = \frac{3\sqrt{3}}{2}r_s$ marks the transition between the case $C(r) > 0$ for all $r > 0$ and the case $C(r) < 0$ for $r_{-1} < r < r_1$ (shaded region).*

where $r_c = r_{-1,1}(b_c)$ is the critical radius. Otherwise $C(r) = 0$ only has one real solution for r, which is always smaller than or equal to zero. But we are only interested in $r > 0$ by definition. Accordingly, for $b > b_c$ the solution for $\kappa = 0$ can be neglected as well. This leaves the two roots r_{-1} and r_1, between which $C(r)$ is negative and therefore the integral (Eq. 2.11) undefined. In Fig. 2.2 $C(r)$ is plotted for three different b values. A more detailed interpretation reveals three different kinds of photon orbits. On the on hand trajectories with an impact parameter below the critical value b_c are unbound orbits, on which photons travel from infinity towards the singularity at $r = 0$ or vice versa[1]. For impact parameter greater than the critical value there are two possible kinds of orbits. Firstly, there exists bound orbits, laying within the region $0 \leq r \leq r_c$. On the other hand, there are unbound orbits for $r > r_c$, which are entirely outside the critical radius r_c. A more detailed discussion of the different kinds of possible orbits is given by Chandrasekhar (1983).

The orbits of interest in this thesis are those in which the trajectory reaches the observer ($r \to \infty$) as we want analyze the impact of relativistic effects on the observations. These are the unbound orbits, which have to be treated differently for calculation purposes. Unbound orbits without a periastron are just determined by Eq. 2.11 in contrast to orbits with a periastron

$$r_p(b) := r_1(b) = -2\sqrt{\frac{b^2}{3}}\cos\left(\frac{\cos^{-1}\left(\frac{3\sqrt{3}}{2}\frac{r_s}{b}\right) + 2\pi}{3}\right) \quad , \tag{2.20}$$

where r_p corresponds to the maximum of the solutions of Eq. 2.18. To obtain a correctly described trajectory crossing the periastron the integral to calculate $\Psi(R)$ has to be split in a way that first the path from infinity to the periastron is calculated and then that from the

[1]The orbits themselves are not limited by the Schwarzschild radius. However, the required energy to escape the gravitating objects reaches infinity at the event horizon making it impossible for anything to escape once it crossed the event horizon.

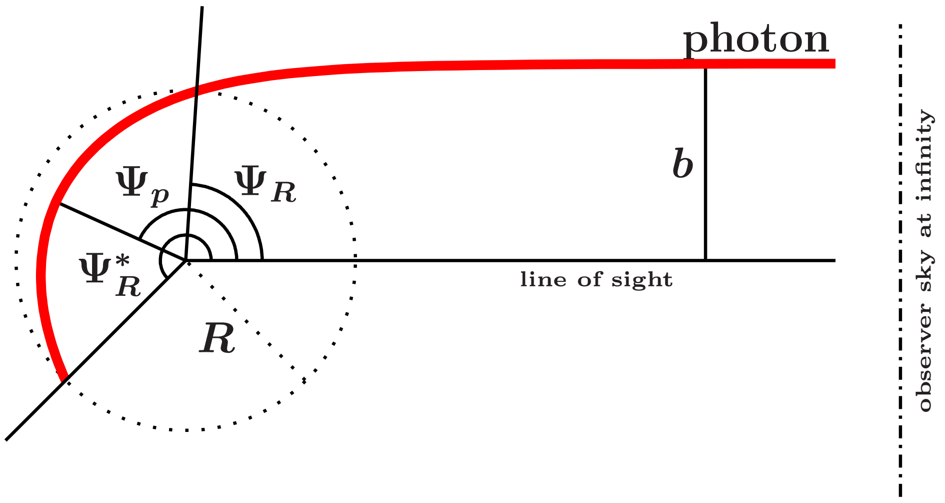


Fig. 2.3.: *Sketch of a photon trajectory starting at radius R approaching the periastron at (r_p, Ψ_p) then traveling outwards and passing again the radius R at (R, Ψ_R), seen by an distant observer ($r \to \infty$, $\Psi = 0$) under the angle Ψ^*_R with an impact parameter b.*

periastron to the radius R, which can be written as

$$\Psi^*(R) = 2\Psi(r_\mathrm{p}) - \Psi(R) \quad , \tag{2.21}$$

where Ψ^* relates to angles greater than $\Psi(r_\mathrm{p})$. Figure 2.3 shows an example for such an trajectory. Note that these photon trajectories are symmetric to their periastron, which can be seen easily by looking at Eq. 2.11.

There is a more suitable variable to describe a photon path, especially if not the entire orbit is involved, e.g., when describing the relativistic photon trajectories of photons emanating from the surface or the vicinity of a compact object. Here the emission angle α is much more convenient, which is defined as the angle between the radial component of the photons four-velocity and its overall direction (see, e.g., Beloborodov, 2002):

$$\begin{aligned}\tan(\alpha) &= \left.\frac{|u^3|}{|u^1|}\right|_{\theta=\frac{\pi}{2}} = \left.\frac{\sqrt{g_{33}u^3u^3}}{\sqrt{g_{11}u^1u^1}}\right|_{\theta=\frac{\pi}{2}} \\ &= \left[\frac{R^2}{b^2}\left(1-\frac{r_\mathrm{s}}{R}\right)^{-1} - 1\right]^{-1/2}\end{aligned} \tag{2.22}$$

leading to

$$\sin(\alpha) = \frac{b}{R}\sqrt{1-\frac{r_\mathrm{s}}{R}} \tag{2.23}$$

The emission angle allows to distinguish more easily whether the photon trajectory exhibits a periastron or not, as the emission angle is unique along a trajectory. Namely, $\alpha < \pi/2$ for those without a periastron and $\alpha > \pi/2$ for those with a periastron. $\alpha = \pi/2$ corresponds to the periastron itself. Note that the symmetry of the photon trajectory regarding the periastron combined with Eq. 2.23 gives the following relation:

$$\alpha^* = \pi - \alpha \tag{2.24}$$

as $\sin\alpha^* = \sin\alpha$ for $\alpha^* \neq \alpha$ (see Fig. 2.4). Furthermore, there is a maximum value for the emission angle for trajectories with periastron based on the critical impact parameter b_c (Ferrigno et al., 2011):

$$\begin{aligned}\alpha_{\max}(u) &:= \pi - \arcsin\left(\frac{b_c}{r_s} u\sqrt{1-u}\right) \\ &= \pi - \arcsin\left(\frac{3\sqrt{3}}{2} u\sqrt{1-u}\right)\end{aligned} \tag{2.25}$$

b_c is the minimal possible impact parameter for trajectories with periastron and therefore corresponds to the maximal emission angle for a given emission radius. The corresponding maximal angle Ψ is then given by

$$\Psi_{\max}(u) := \Psi^*(u)\Big|_{b=b_c} \quad . \tag{2.26}$$

Additionally, we can test the classical limes for flat spacetime, in which Ψ is supposed to be equal to α. By substituting the impact parameter in Eq. 2.16 with the relation given in Eq. 2.23 and calculating the limes

$$\lim_{R\to\infty} \widetilde{\Psi}(R) = \lim_{R\to\infty} \arcsin\left(\sin(\alpha)/\sqrt{1 - r_s/R}\right) = \alpha$$

indeed leads to equality of Ψ and α.

2.2.2. Time Delay

Another value of interest is the time difference in the arrival time of photons due to the different lengths of their trajectories. At first we derive the travel time itself similar to the

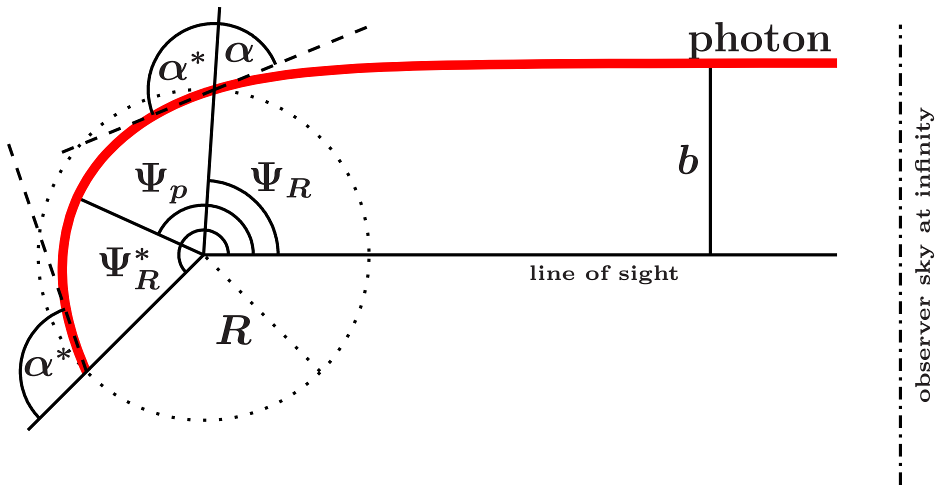


Fig. 2.4.: *Sketch of a photon trajectory starting at radius R with an emission angle α^* approaching the periastron at (p, Ψ_p) then traveling outwards and passing again the radius R at (R, Ψ_R) with $\alpha = \pi - \alpha^*$, seen by an distant observer $(r \to \infty,\ \Psi = 0)$ under the angle Ψ^*_R with an impact parameter b.*

derivation of the photon trajectory (Eq. 2.11), which leads to:

$$\begin{aligned} t(R) &= \int_{t(R)}^{t(\infty)} \mathrm{d}t = \int_R^\infty \frac{\mathrm{d}t}{\mathrm{d}\lambda}\frac{\mathrm{d}\lambda}{\mathrm{d}r}\mathrm{d}r \\ &= \int_R^\infty \mathrm{d}r \left(1-\frac{r_\mathrm{s}}{r}\right)^{-1}\left[1-\frac{b^2}{r^2}\left(1-\frac{r_\mathrm{s}}{r}\right)\right]^{-1/2} \end{aligned} \tag{2.27}$$

Integrating the radius to infinity obviously leads to infinite travel times. Therefore, it is necessary to define the difference of travel times to a certain reference, which is chosen to be the trajectory with $b = 0$ (see, e.g., Poutanen & Beloborodov, 2006), which coincides with the line of sight:

$$\begin{aligned} \Delta t(R) &:= t(R) - t(R)\Big|_{b=0} \\ &= \int_R^\infty \mathrm{d}r \left(1-\frac{r_\mathrm{s}}{r}\right)^{-1}\left\{\left[1-\frac{b^2}{r^2}\left(1-\frac{r_\mathrm{s}}{r}\right)\right]^{-1/2} - 1\right\} \end{aligned} \tag{2.28}$$

Furthermore, we also need to take into account that for trajectories starting at different radii there is an additional contribution to the time delay, which is necessary in order to be able to compare their time delays. Namely, this is the time difference between the reference points as shown in Fig. 2.5, which can be calculated easily:

$$\begin{aligned} \delta t(R_1, R_2) &:= t(R_1)\Big|_{b=0} - t(R_2)\Big|_{b=0} \\ &= \int_{R_1}^{R_2} \mathrm{d}r \left(1-\frac{r_\mathrm{s}}{r}\right)^{-1} \\ &= r_\mathrm{s}\ln\left(\frac{R_2 - r_\mathrm{s}}{R_1 - r_\mathrm{s}}\right) + R_2 - R_1 \end{aligned} \tag{2.29}$$

Now we can write down the overall time delay $\Delta T(R)$ a distant observer measures between simultaneously emitted photons:

$$\begin{aligned} \Delta T(R) &:= \Delta t(R) + \delta t(R_\mathrm{ref}, R) \\ &= \int_R^\infty \mathrm{d}r \left(1-\frac{r_\mathrm{s}}{r}\right)^{-1}\left\{\left[1-\frac{b^2}{r^2}\left(1-\frac{r_\mathrm{s}}{r}\right)\right]^{-1/2} - 1\right\} \\ &+ r_\mathrm{s}\ln\left(\frac{R - r_\mathrm{s}}{R_\mathrm{ref} - r_\mathrm{s}}\right) + R - R_\mathrm{ref} \end{aligned} \tag{2.30}$$

R_ref can be chosen arbitrarily as long as it exceeds r_s. Note that the symmetry of the photon trajectory is also maintained for the time delay. Hence, the same relation as for the photon trajectory with periastron (Eq. 2.21) has to be taken into account, which leads to

$$\Delta T^*(R) = 2\Delta T(p) - \Delta T(R) \quad . \tag{2.31}$$

2.2.3. Analytical Approximations

As already mentioned, the integral determining the photon trajectory (Eq. 2.11) is not analytically solvable. The same applies to the time delay (Eq. 2.28). However, Beloborodov (2002) came up with a very simple yet extremely precise analytical approximation. He considered

$$x := 1 - \cos(\alpha) \tag{2.32}$$

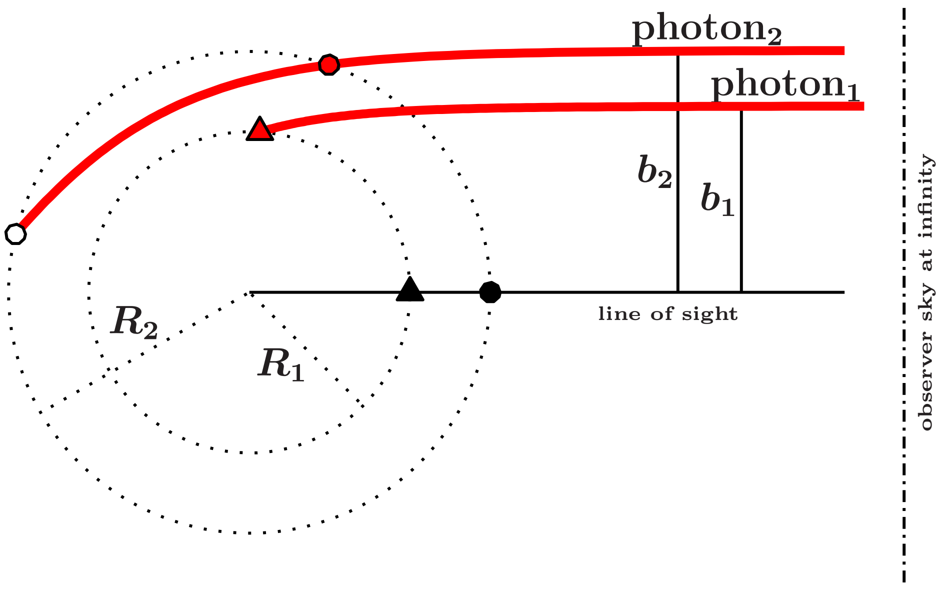


Fig. 2.5.: *Shown are two photon trajectories emitted at different radii R_1 and R_2 with different impact parameter b_1 and b_2. $\Delta t(R)$ (Eq. 2.28) would measure the time difference $\Delta t(R_2)$ between the solid and the red-filled circle and respectively $\Delta t(R_1)$ between the triangles. To compare these, also the time difference $\delta t(R_1, R_2)$ (Eq. 2.29) between points marked with a solid triangle and solid circle has to be calculated. If a trajectory possesses a periastron this has to be taken into account according to Eq. 2.31. For example, the time delay between simultaneous emitted photons at the white-filled circle and the red-filled triangle would be $\Delta T^*(R_2) - \Delta T(R_1)$ with arbitrary R_{ref}.*

to be a small parameter. Following Beloborodov (2002), we substitute the impact parameter in Eq. 2.14 with the relation given by Eq. 2.23, which leads to

$$\Psi = \frac{b}{r_\mathrm{s}} \int_0^u \mathrm{d}u_r \left[1 - x(2-x)\frac{u_r^2(1-u_r)}{u^2(1-u)}\right]^{-1/2} \quad , \tag{2.33}$$

where

$$x(2-x) = u^2(1-u)\sin^2\alpha \quad . \tag{2.34}$$

Equation 2.33 can now be simplified using the binomial approximation:

$$\begin{aligned}\Psi &\approx \frac{b}{r_\mathrm{s}} \int_0^u \mathrm{d}u_r \left[1 + \frac{1}{2}x(2-x)\frac{u_r^2(1-u_r)}{u^2(1-u)}\right] \\ &= \frac{b}{r_\mathrm{s}} u \left[1 + x(2-x)\frac{4-3u}{24(1-u)}\right]\end{aligned} \tag{2.35}$$

This approximation only holds, if the second expression of the radicand in Eq. 2.33 is smaller one, which is fulfilled for $u < 2/3$. However, this is no issue as for the approximation derived in the following the upper limit for u is below that value.

In the next step we insert Eq. 2.35 into the expansion of

$$y := 1 - \cos(\Psi) \quad , \tag{2.36}$$

leading to the following expression (Beloborodov, 2002):

$$\begin{aligned} y &= 1 - 1 + \frac{\Psi^2}{2!} - \frac{\Psi^4}{4!} + \mathcal{O}(x^6) \\ &= \frac{x}{1-u} - u^2 \left[\frac{1}{112} \left(\frac{x}{1-u} \right)^3 + \frac{1}{224} \left(\frac{5}{3} - u \right) \left(\frac{x}{1-u} \right)^4 + \mathcal{O}(x^5) \right] \end{aligned} \tag{2.37}$$

Taking only the linear term in x in Eq. 2.37, it provides the analytical approximation in the more readable form:

$$\Psi = \arccos\left(1 - \frac{1-\cos\alpha}{1-u}\right) = \arccos\left(1 - \frac{1-\cos\alpha}{1-\frac{r_\mathrm{s}}{R}}\right) \tag{2.38}$$

This equation requires that $u \leq 1/2$ (and accordingly $R \geq 2r_\mathrm{s}$) for arbitrary $\alpha \leq \pi/2$. Note that this limitation makes it necessary to redefine the critical impact parameter and the critical radius as $r_\mathrm{c} < 2r_\mathrm{s}$ (Eq. 2.19). Therefore we set $r_\mathrm{c} = 2r_\mathrm{s}$ in case of the approximation and accordingly

$$b_\mathrm{c} = 2\sqrt{2}\, r_\mathrm{s} = \sqrt{2}\, r_\mathrm{c} \quad . \tag{2.39}$$

Keep in mind that this redefinition also changes the equations (Eq. 2.25 & Eq. 2.26) for the maximal emission angle α_max and Ψ_max.

By combining the approximation Eq. 2.38 and Eq. 2.23, we find (see Beloborodov, 2002)

$$r(\Psi) = \left[\frac{r_\mathrm{s}^2 (1-\cos\Psi)^2}{4(1+\cos\Psi)^2} + \frac{b^2}{\sin^2\Psi} \right]^{1/2} - \frac{r_\mathrm{s}(1-\cos\Psi)}{2(1+\cos\Psi)} \quad . \tag{2.40}$$

It is important to keep in mind that Eq. 2.38 and Eq. 2.40 only describe the photon trajectory up to the periastron even though this is not obvious by looking at these equations. To account for trajectories with a periastron Eq. 2.21 has to be considered just like for the exact equation of the photon trajectory (Eq. 2.11).

Also notice that in the expansion of y (Eq. 2.37) the x^2-term vanishes, what explains the high accuracy of this approximation as seen in Fig. 2.6, which shows the contours of the error $\delta\beta/\beta$, where

$$\beta = \Psi - \alpha \tag{2.41}$$

is the bending angle. For example, the error is less than 3% for $R > 3r_\mathrm{s}$ and $\alpha \leq \pi/2$. The highest accuracy is achieved for large radii and small emission angles and as expected the error increases with increasing α and u. Beloborodov (2002) also states that this accuracy is much higher compared to the approximation derived by expanding Eq. 2.33 in terms of small r_s/R. Furthermore, Fig. 2.6 shows the region in the α-u plane a neutron star with $M = 1.4\ \mathrm{M}_\odot$ can occupy, where the maximal range of the mass to radius ratio u is based on Steiner et al. (2013). For such a neutron star, the error is less than 9%.

The equations for the time delay can be approximated with the same approach as above. Using the substitution given in Eq. 2.13 and using the binomial approximation the time delay in Eq. (2.28) rewrites as

$$\begin{aligned} \Delta t &= r_\mathrm{s} \int_0^u \mathrm{d}u_r \frac{1}{u_r^2(1-u_r)} \left\{ \left[1 - \frac{b^2}{r_\mathrm{s}^2} \frac{1}{u_r^2(1-u_r)} \right]^{-1/2} - 1 \right\} \\ &\approx r_\mathrm{s} \int_0^u \mathrm{d}u_r \frac{1}{u_r^2(1-u_r)} \left\{ \left[1 + \frac{1}{2} \frac{b^2}{r_\mathrm{s}^2} \frac{1}{u_r^2(1-u_r)} \right] - 1 \right\} \\ &= x \frac{r_\mathrm{s}}{u(1-u)} - x^2 \frac{r_\mathrm{s}}{2u(1-u)} \quad . \end{aligned} \tag{2.42}$$

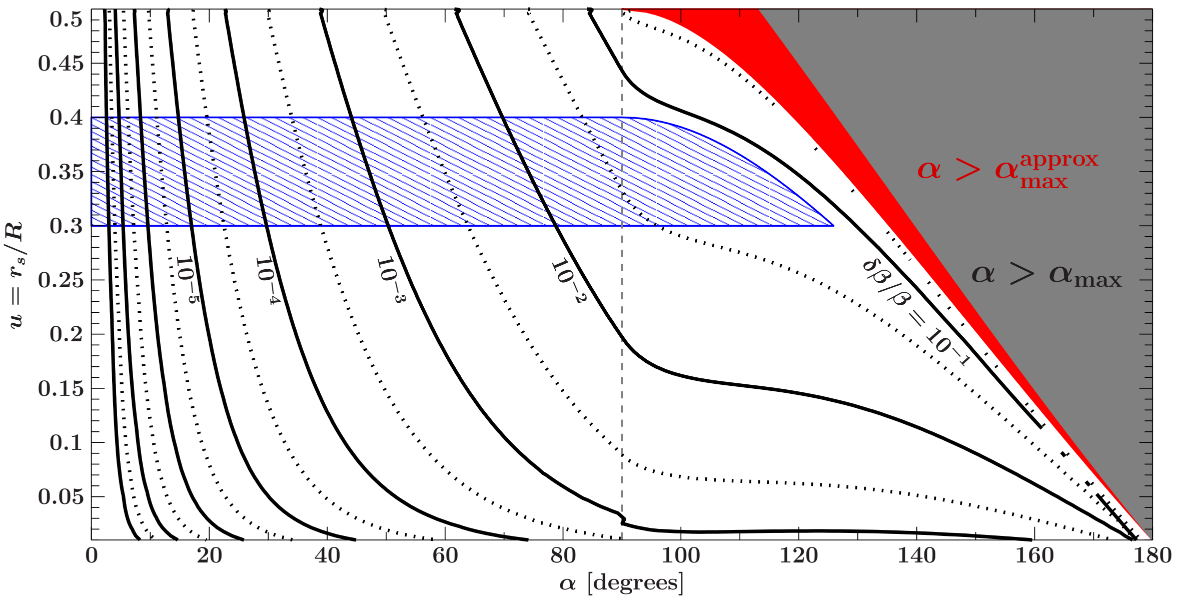


Fig. 2.6.: *This figure (adapted from Beloborodov, 2002,* Fig. 2) *shows the accuracy of the analytical approximation (Eq. 2.38) by giving the contours of the error $\delta\beta/\beta$, with the bending angle $\beta = \Psi - \alpha$, depending on the emission angle α and the inverse emission radius r_g/R. The contours are given in logarithmic steps of* 0.5*. The gray and red areas relate to combinations of α and u, for which there is no solution for the exact photon trajectory in Eq. 2.21 and respectively for the approximation (Eq. 2.38), where α_{max} is given by Eq. 2.25. The blue shaded area confines the region a neutron star with $M = 1.4\ \mathrm{M}_\odot$ can occupy (Steiner et al., 2013). Note that in this figure the y-axis begins at $u = 0.01$. However, we would expect that $\delta\beta = 0$ for $u = 0$ and arbitrary α, as for $u = 0$ the apparent and real emission angle are equal ($\alpha = \Psi$). For numerical reasons the contours of $\delta\beta/\beta$ do not behave well for $u \to 0$, therefore the plot shows only values $u > 0.01$.*

Keeping only the linear term in x we find

$$\Delta t = R(1 - \cos\Psi) \quad , \tag{2.43}$$

which corresponds to the time delay one would expect in flat space time. As Poutanen & Beloborodov (2006) point out, the accuracy of this approximation is not as good as for Eq. 2.38, but is sufficient for most calculations. A higher accuracy would require to consider higher orders already in the expansion of the time delay in Eq. 2.42, which, in combination with Eq. 2.37, leads to (see Poutanen & Beloborodov, 2006)

$$\Delta t = R(1 - \cos\Psi)\left[1 + \frac{u(1-\cos\Psi)}{8} + \frac{u(1-\cos\Psi)^2}{8}\left(\frac{1}{3} - \frac{u}{14}\right)\right] \quad . \tag{2.44}$$

2.3. Flux

In the previous sections (2.1 & 2.2) of this chapter the properties of photon trajectories in a general relativistic framework using the Schwarzschild metric were discussed. Besides gravitational light bending and time delays there are other relativistic effects, which have to be accounted for to describe observations appropriate. These effects will be motivated and partly derived in this section and eventually we obtain an equation for the observed flux of a (fast) rotating compact object a distant observer measures considering all previously discussed relativistic effects.

2.3.1. Gravitational Redshift

The gravitational redshift, which is a consequence of Einsteins Principle of Equivalence, decreases the energy of photons and accordingly their frequency while they propagate out of the gravitational potential of a gravitating mass. The deeper the potential the photon has to overcome, the higher the redshift. Hence this effect is especially relevant for massive and compact objects. In this section the derivation of the gravitational redshift is outlined and mainly based on the description of Carroll (2004).

The frequency ν of a photon, which is traveling along the path $x^{\mu}(\lambda)$ (see Sec. 2.1 & 2.2), measured by an observer moving with the four-velocity U^{μ} is given by

$$\nu = -g_{\mu\nu} U^{\mu} \frac{\mathrm{d}x^{\nu}}{\mathrm{d}\lambda} \quad . \tag{2.45}$$

The four-velocity also fulfills the relation

$$U_{\mu} U^{\mu} = g_{\mu\mu} U^{\mu} U^{\mu} = -\epsilon \quad , \tag{2.46}$$

where ϵ is constant. We now consider a stationary observer in Schwarzschild coordinates, which means that the three spatial components equal zero leaving only the U^0 component, which can be determined to

$$U^0 = \epsilon \left(1 - \frac{r_\mathrm{s}}{r}\right)^{-1/2} \tag{2.47}$$

by using Eq. 2.46 and the according entry of the Schwarzschild metric (Eq. 2.1). This relation combined with Eq. 2.45 leads to

$$\begin{aligned} \nu &= -g_{00} U^0 \frac{\mathrm{d}t}{\mathrm{d}\lambda} \\ &= \epsilon \left(1 - \frac{r_\mathrm{s}}{r}\right)^{-1/2} \end{aligned} \tag{2.48}$$

We are now interested in the ratio of the frequencies of a photon emitted at radius R and observed by a distant observer, which can be obtain from Eq. 2.48:

$$\frac{\nu_\mathrm{obs}}{\nu_0} = \left(1 - \frac{r_\mathrm{s}}{R}\right)^{1/2} \quad , \tag{2.49}$$

where ν_obs is the observed frequency of a photon at $r \to \infty$, which was originally emitted with the frequency ν_0 at radius R. Also interesting is that this relation makes only sense for photons emanated from radii exceeding the Schwarzschild radius, at which the redshift approaches infinity.

2.3.2. Relativistic Doppler Shift & Aberration

Until now only gravitational effects were discussed, but there are also special relativistic effects that have to be accounted for introduced by the relative motion of observer and the source of the photons. In this case we consider the motion caused by the rotation of the object. To transfrom values correctly from the frame of the source and that of the observer, the Lorentz

Transformation has to be applied, which on the one hand causes the photons emitted from the surface of a rotating object to be Doppler shifted by the factor

$$\delta := \frac{\nu'}{\nu} = \frac{\sqrt{1-\beta^2}}{1-\beta\cos\xi} \quad , \tag{2.50}$$

where ν' relates to the photon frequency in the corotating frame of the object and ν to the fixed lab frame of the observer (from now on primed values correspond to the corotating frame). $\beta = v/c$ is the velocity of the spot on the object, where the photon is emitted and ξ is the angle between the direction of the spot velocity and the emission direction of the photon. On the other hand the Lorentz Transformation provides the following relation (see Poutanen & Beloborodov, 2006)

$$\cos\alpha' = \delta\cos\alpha \tag{2.51}$$

between the emission angle in the corotating and the lab frame, which relates to the relativistic beaming/aberration causing the emission being directed along the direction of motion.

2.3.3. Lab Frame

To be able to give the equations of the previous section in a more concrete form we have to choose a coordinate system for our lab frame. We will adopt the coordinate system Poutanen & Beloborodov (2006) used, which is shown in Fig. 2.7. The unit vector $\boldsymbol{n}$ points from the center of the object toward the position of the spot were the photon gets emitted. While the object is rotating around its spin axis, which is chosen to coincide with the z axis, the spot direction of motion is given by $\boldsymbol{\beta}$. The direction of the photon at the point of emission is denoted with the unit vector $\boldsymbol{k}_0$, where the angle between $\boldsymbol{k}_0$ and $\boldsymbol{n}$ is the real emission angle α. While the photons propagate away from the object their direction changes due to the effect of gravitational light bending. Only those photons propagating along the line of sight, given by the unit vector $\boldsymbol{k}$, at large distances arrive the observer. $\boldsymbol{k}$ is chosen to be within the x,z-plane, where i is its inclination angle measured from the spin axis. The angle between the line of sight and $\boldsymbol{n}$, denoted with Ψ, is the apparent emission angle.

Having given the coordinate system we now can rephrase the previous derived equations accordingly, starting with the absolute value of the spot velocity β, which can easily determined to be

$$\beta = 2\pi R \frac{f}{\sqrt{1-u}} \sin\theta \quad , \tag{2.52}$$

where R is the radius at which the photon emanates from and the spin frequency f already was corrected for the redshift (Eq. 2.49). The second value we have to rephrase to determine the Doppler factor (Eq. 2.50) is the angle ξ between the initial direction of the photon propagation and the spot velocity. To obtain this value we first express $\boldsymbol{k}$ as

$$\boldsymbol{k} = \boldsymbol{n}\cos\Psi + \boldsymbol{n}_\perp \sin\Psi \quad , \tag{2.53}$$

where $\boldsymbol{n}_\perp$ is that vector rectangular to $\boldsymbol{n}$, which lies in the plane defined by $\boldsymbol{n}$ and $\boldsymbol{k}$. Now the initial propagation direction can be expressed in terms of $\boldsymbol{k}$ and $\boldsymbol{n}$ as follows:

$$\begin{aligned} \boldsymbol{k_0} &= \boldsymbol{n}\cos\alpha + \boldsymbol{n}_\perp \sin\alpha \\ &= \boldsymbol{n}\cos\alpha + \boldsymbol{n}_\perp \sin\Psi\frac{\sin\alpha}{\sin\Psi} + \boldsymbol{n}\cos\Psi\frac{\sin\alpha}{\sin\Psi} - \boldsymbol{n}\cos\Psi\frac{\sin\alpha}{\sin\Psi} \\ &= \boldsymbol{k}\frac{\sin\alpha}{\sin\Psi} + \boldsymbol{n}\frac{\sin(\Psi-\alpha)}{\sin\Psi} \end{aligned} \tag{2.54}$$

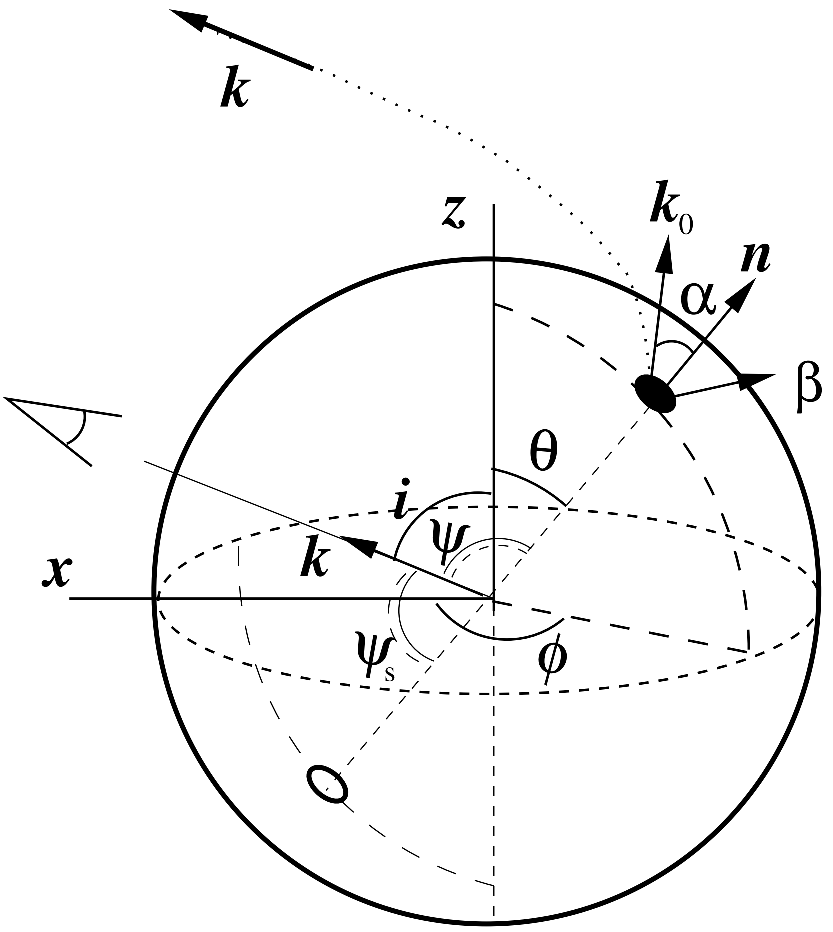


Fig. 2.7: *Scheme of the geometry of light bending and its fundamental parameter. Dotted line relates to the photon trajectory. Adopted from Poutanen & Beloborodov (2006).*

Explicitly writing down

$$\boldsymbol{n} = \begin{pmatrix} \sin\theta\cos\phi \\ \sin\theta\sin\phi \\ \cos\theta \end{pmatrix} \tag{2.55}$$

and

$$\boldsymbol{k} = \begin{pmatrix} \sin i \\ 0 \\ \cos i \end{pmatrix} \tag{2.56}$$

we can determine

$$\begin{aligned} \cos\xi &= \boldsymbol{k}_0 \circ \frac{\boldsymbol{\beta}}{\beta} = \frac{\sin\alpha}{\sin\Psi} \cdot \boldsymbol{k} \circ \frac{\boldsymbol{\beta}}{\beta} \\ &= -\frac{\sin\alpha}{\sin\Psi} \sin\phi \sin i \quad , \end{aligned} \tag{2.57}$$

where the direction of the spot velocity is simply the unit vector in ϕ direction ($\boldsymbol{\beta}/\beta = \boldsymbol{e}_\phi$) and accordingly $\boldsymbol{\beta} \circ \boldsymbol{n} = 0$. Now the Doppler factor is completely described by quantities within the lab frame.

There is also and relation between Ψ and the other system parameters given by

$$\begin{aligned} \cos\Psi &= \boldsymbol{k} \circ \boldsymbol{n} \\ &= \cos i \cos\theta + \sin i \sin\theta \cos\phi \quad , \end{aligned} \tag{2.58}$$

which describes the periodical change of the apparent emission angle of the emitting spot with the rotation of the object.

Furthermore, we also need an additional parameter to determine the impact point of the photon on the observer sky as the impact parameter b only gives the projected distance to the origin of coordinates. Therefore, we define the observer sky to be centered to the line of sight with the two axis A and B, where A coincides with the y axis as $\boldsymbol{e}_y \circ \boldsymbol{k} = 0$. Introducing an azimuthal angle ρ, specifying the rotation around the line of sight, A and B can be written as

$$A = b\cos\rho \qquad \text{and} \qquad B = b\sin\rho \quad , \tag{2.59}$$

where ρ is defined by the following two relations:

$$\begin{aligned}\cos\rho &= \frac{[\boldsymbol{k}\times(\boldsymbol{n}\times\boldsymbol{k})]\circ\boldsymbol{e}_y}{|\boldsymbol{k}\times(\boldsymbol{n}\times\boldsymbol{k})|}\\ &= \frac{\sin\theta\sin\phi}{\sqrt{\sin^2\theta\sin^2\phi+(\sin i\cos\theta-\cos i\sin\theta\cos\phi)^2}}\end{aligned} \tag{2.60}$$

$$\begin{aligned}\sin\rho &= \frac{|[\boldsymbol{k}\times(\boldsymbol{n}\times\boldsymbol{k})]\times\boldsymbol{e}_y|}{|\boldsymbol{k}\times(\boldsymbol{n}\times\boldsymbol{k})|}\\ &= \frac{\sin i\cos\theta-\cos i\sin\theta\sin\phi}{\sqrt{\sin^2\theta\sin^2\phi+(\sin i\cos\theta-\cos i\sin\theta\cos\phi)^2}}\end{aligned} \tag{2.61}$$

2.3.4. Observed Spectral and Bolometric Flux

In the previous sections of this chapter the special and general relativistic effects were discussed, which occur for photons emitted in the vicinity of compact objects. In this section, which mainly follows the description of Poutanen & Beloborodov (2006), we finally obtain the observed flux from a spot on the surface of the object accounting for these effects. The spectral flux at a photon energy E is given by

$$\mathrm{d}F_E = I_E\mathrm{d}\Omega \quad , \tag{2.62}$$

where I_E is the specific radiative intensity and $\mathrm{d}\Omega$ the solid angle, both in the lab frame. The solid angle can be expressed as

$$\mathrm{d}\Omega = \frac{\delta}{1-u}\cos\alpha\frac{\mathrm{d}\cos\alpha}{\mathrm{d}\cos\Psi}\frac{\mathrm{d}S'}{D^2} \quad , \tag{2.63}$$

where $\mathrm{d}S'$ is the area of the spot in the corotating frame and D the distance from the object to the observer. Further, we can determine $\mathrm{d}\cos\alpha/\mathrm{d}\cos\Psi$ to be

$$\frac{\mathrm{d}\cos\alpha}{\mathrm{d}\cos\Psi} = 1-u \quad , \tag{2.64}$$

using Eq. 2.23 and the approximation given in Eq. 2.38. Note that Eq. 2.63 only applies to spots, whose surface vector $\mathbf{d}\boldsymbol{S}$ aligns with the position unit vector $\boldsymbol{n}$ or in other words, to spots on a sphere. In general one can write

$$\mathrm{d}\Omega = \cos\gamma'\frac{\mathrm{d}S'}{D^2} \quad , \tag{2.65}$$

where γ' is the emission offset angle in the corotating frame given by

$$\cos\gamma' = \mathbf{dS'} \circ \boldsymbol{k}'_0 / |\mathbf{dS'}| \tag{2.66}$$

and respectively

$$\cos\gamma = \mathbf{dS} \circ \boldsymbol{k}_0 / |\mathbf{dS}| \tag{2.67}$$

in the lab frame. The relation between $\mathbf{dS}$ and $\mathbf{dS'}$, and between $\boldsymbol{k}_0$ and $\boldsymbol{k}'_0$ is given by a Lorentz transformation (Poutanen & Beloborodov, 2006). For spots on a sphere $\alpha \equiv \gamma$ and Eq. 2.63 equals Eq. 2.65 using Eq. 2.64. Based on the Liouville's theorem for curved spacetime (see, e.g., Misner et al., 1973) the relation between the observed intensity and the local intensity $I'_{E'}(\gamma')$ measured in the corotating frame is:

$$I_E = \left(\frac{E}{E'}\right)^3 I'_{E'}(\gamma') \quad , \tag{2.68}$$

where the ratio of the observed and emitted photon energy is given by

$$\frac{E}{E'} = \delta\sqrt{1-u} \quad , \tag{2.69}$$

a combination of the Doppler shift (Eq. 2.50) and the gravitational redshift (Eq. 2.49). Putting everything together the spectral flux reads as

$$\mathrm{d}F_E = I'_{E'}(\gamma')\ \delta^3 (1-u)^{3/2} \cos\gamma' \frac{\mathrm{d}S'}{D^2} \quad . \tag{2.70}$$

By integrating Eq. 2.68 over the energy we get the relation between the observed and local bolometric intensity:

$$I = \left(\frac{E}{E'}\right)^4 I'(\gamma') \quad , \tag{2.71}$$

with which the bolometric flux can be written as:

$$\mathrm{d}F = I'(\gamma')\ \delta^4 (1-u)^2 \cos\gamma' \frac{\mathrm{d}S'}{D^2} \tag{2.72}$$

Also notice that the equations for the spectral and the bolometric flux (Eq. 2.70 and Eq. 2.72) do not account for time delays caused by different propagation paths of photons (see Sec. 2.2.2) emitted at different phases ϕ. To obtain the flux at the observed phase ϕ_{obs} one has to follow the relation

$$\mathrm{d}\bar{F}(\phi_{\mathrm{obs}}) = \mathrm{d}F(\phi_{\mathrm{obs}} - \Delta\phi) \quad , \tag{2.73}$$

where $\Delta\phi$ is given by

$$\Delta\phi = 2\pi f \Delta T \quad . \tag{2.74}$$

Chapter 3

Implementation

In this Chapter, the implementation of the theoretical formulas discussed in Chap. 2 will be explained. After a short outline of the basic structure of the program, the single modules will be focused on in more detail. The intention of this chapter is to give an overview of the program deriving the observed flux and its modules, and a general understanding how the theoretical equations discussed in Chap. 2 were implemented. The code itself is written in Isis (Houck & Denicola, 2000), a script language using the S-Lang[1] interpreter. In this chapter no actual code will be shown. For a list of functions related to the program and their usage see App. C.

Figure 3.1 shows a sketch of the structure of the program, which is divided into modules containing the the different calculation steps. The boxed numbers correspond to the individual sections in this Chapter, in which the according module is discussed.

`Parameter` (Sec. 3.1):
: At the very beginning all important physical as well as code related `parameters` are stored into a `Struct_Type`, which will be passed to all following modules (see App. C, `lb_par_init`).

`Object` (Sec. 3.2):
: After the parameters have been specified, the geometrical structure of the `object` is built, which is sampled by small surface elements.

`Mapping` (Sec. 3.3):
: The `mapping` module creates a lookup table for the individual values of the photon trajectory. This module is independent of the `object` module.

`Interpolation` (Sec. 3.4):
: To obtain the parameters of the photon trajectory at a specific point, the values within the lookup table have to be interpolated.

`Projection` (Sec. 3.5):
: The `projection` module is the core of the program performing the actual projection of the defined object onto the observer sky for each rotation phase. It projects the locations of the individual surface elements of the `object` by interpolating the trajectory values stored in the lookup table.

`Overlap & oversize` (Sec. 3.6):
: An optional step is the search for `overlaps`, in which for each phase those regions of the object are identified, which would overlap in the observer plane.

[1]See `http://www.jedsoft.org/slang/`

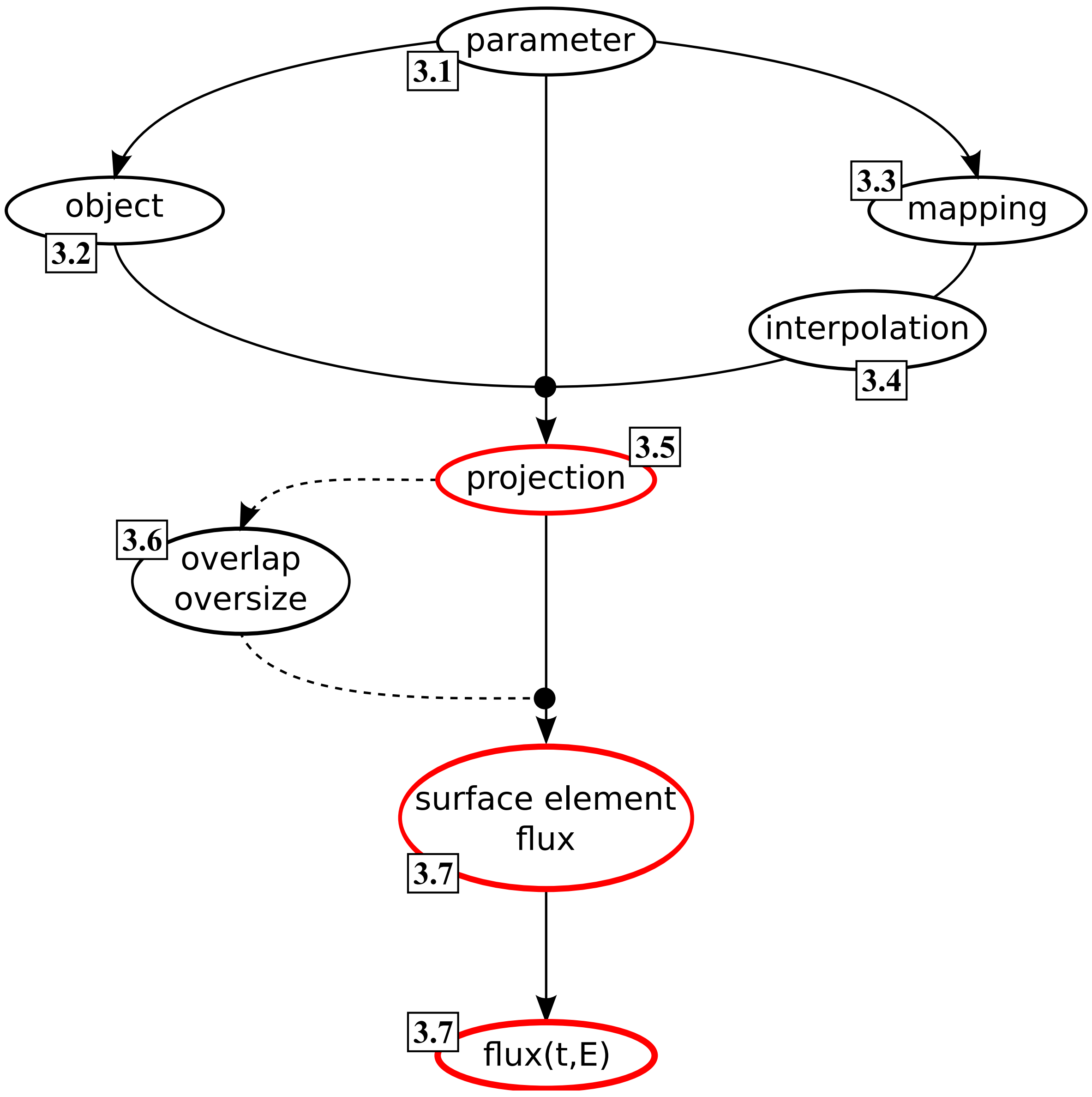


Fig. 3.1.: *Structure of the program, where the* `projection` *is the core module. The solid arrows connecting the individual modules indicate their dependencies, e.g., the* `object` *and the* `mapping` *module are independent. Dashed arrows relate to optional steps. For modules marked with red ellipses, the program provides plotting functions. The boxed numbers correspond to the individual sections in this Chapter, in which the according module is discussed.*

`Surface element flux` (Sec. 3.7):
: The first of two steps to calculated the observed flux, in which the flux of every single surface element is calculated based on a given emission profile.

`Flux(t,E)` (Sec. 3.7):
: In the next step, the flux portions of the surface elements are summed up according to a given time and energy grid, resulting in a time and energy resolved flux matrix. The obtained flux also distinguishes between the different parts of the object.

`Plotting` (App. C):
Additionally, the program provides plotting functions to visualize the `projection`, the `projection` combined with the `surface element flux` and the time resolved `flux`.

`Storing` (Sec. B):
Furthermore, it is possible to save and load the results of each of the individual modules.

3.1. Parameter

In this very first step all important parameter concerning the characteristics of the object are set, which in this case is a neutron star. Included are all general information about the neutron star, like its radius R, mass M and spin frequency f. Additionally, Rac, hac and iac give the radius, height and inclination, with respect to the neutron star spin axis, of the accretion column(s) or hot spot(s). The inclination angle of the observer to the spin axis of the neutron star and the distance are denoted with i and D. Besides these parameters describing the system and the geometry of the object, there are also parameters specifying routine related settings, e.g., which projection method should be used, where one can choose between the exact relativistic photon trajectory (Eq. 2.11), the analytical approximation (Eq. 2.38) or simply the geometrical projection. This list of parameters gets then passed to the other modules, which select the required information. Note that in general all these parameters are fixed throughout the whole routine with only two exceptions, rmin and rmax, defining the minimal and maximal occurring radius, are changed by the function `lb_obj_rrange`. The description of the individual parameters will be given in the respective section. A list of all parameters is given in the description of `lb_par_init` in App. C.

3.2. Object

This program is designed to model a spherical neutron star with accretion columns[2]. In this module the main purpose is to build the three dimensional geometrical structure of the object. However, before going into details the general approach to model the structure will be explained.

The basic idea is that the geometrical structure of the object represents the surface, from which the photons emanate. We assume that this surface is solid, i.e., photons hitting it are absorbed or, in other words, we only consider trajectories directly reaching the observer. The object is fully described by two dimensional surfaces. It is convenient to sample these surfaces in a way that the surface of the object is totally covered without any gaps, which can be achieved by suitably describing the surfaces with triangles. The choice of triangles was made as they are the simplest way to define a plane. Moreover, the surface vectors of triangles can be easily determined, which are needed in several steps later on (e.g., emission profiles). Eventually, the object builds up of surface elements, each containing the vertex vectors of the underlying triangle, its surface vector[3] and the barycentric vector[4]. Each surface element represents a spot $\mathrm{d}S'$ (see Eq. 2.70, Eq. 2.72) in the flux equation (Eq. 2.70, Eq. 2.72). To calculate the flux, we also have to obtain the light bending parameters (see Sec. 2.2), which

[2]Note that in principle the program is capable to process any geometrical structure as long as it is given in the format defined by `lb_obj_init`.

[3]The surface vector is defined by $\mathbf{d}\boldsymbol{S} = \boldsymbol{N}|\mathbf{d}\boldsymbol{S}|$, where $\boldsymbol{N}$ is the normal vector to the surface and $|\mathbf{d}\boldsymbol{S}|$ its area.

[4]The barycentric vector is the vector to the centroid of a given surface.

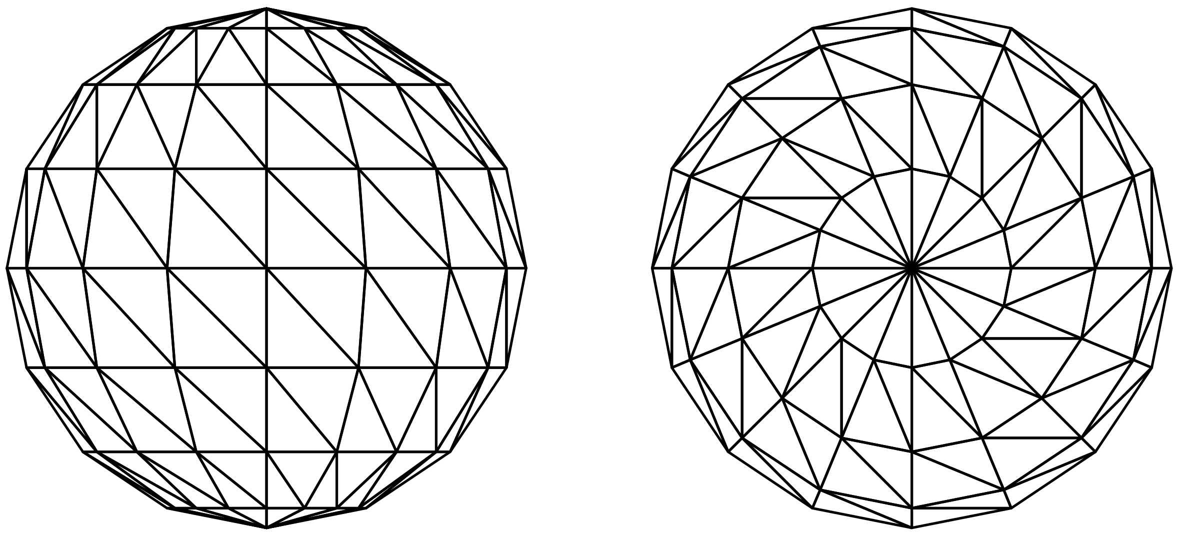


Fig. 3.2.: *Sphere created with* `lb_obj_build_sphere` *seen under the inclination of* 90° *(left) and* 0° *(right).*

will be related to the according barycentric vector of each surface element at the according phase ϕ (see Fig. 2.7). The geometrical structure of the specified `object` is constant. However, the spinning of the `object` around its z-axis can be easily accounted for by rotating the vectors to the according phase. Furthermore, the surface elements include information about their affiliation (e.g., neutron star surface, accretion column wall/cap), which allows to calculate the contribution of each of these individual parts of the object to the flux. In particular, the `object` module provides two main functions to build the different parts of a neutron star. Firstly, the `lb_obj_build_sphere` function, which creates a sphere with radius R, where the number of surface elements this sphere is sampled with can be varied. Figure 3.2 shows an example for such an sphere seen under the inclination of 90° and 0°. Note that the vertices of the surface elements are equidistant in θ and ϕ in spherical coordinates and therefore their area is not constant.

The other function is called `lb_obj_build_column` and creates an accretion column or a hot spot. The accretion column is composed of a cylindrical wall with radius Rac and height hac, and a cap as seen in Fig. 3.3. The cap can be set to be either flat or convex, where the radius of the convexity is $\sqrt{\mathsf{R}^2 - \mathsf{Rac}^2} + \mathsf{hac}$, with the neutron star radius R. It is also possible to create a hot spot with this function by just generating the accretion column cap and set its height to zero.

Additionally, there is a function `lb_obj_build_ns` combining the functionalities of the previous two. `lb_obj_build_ns` builds a neutron star with two accretion columns, whose radii and heights can be set individually. Furthermore, the accretion columns can be inclined with respect to the spin axis of the neutron star according to the iac parameter. Figure 3.4 shows a neutron star with two antipodal accretion columns aligned with the spin axis. Note that the composition of the sphere and the columns leads to hidden regions beneath the column. The effort to create smooth connections is very high, especially if the accretion columns are inclined with different angles. Moreover a smooth connection would not prevent overlaps of projected surface elements on the observer sky. Therefore, a step, in which these overlaps are identified is necessary either way and it is sufficient to just stick the columns onto the neutron star.

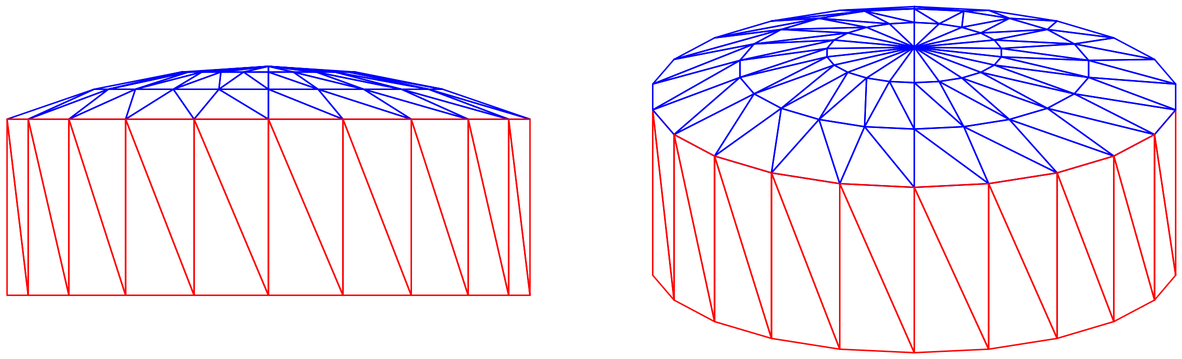


Fig. 3.3.: *Accretion column created with* `lb_obj_build_column` *seen under the inclination of 90° (left) and 20° (right). Red and blue lines relate to surface elements building up the accretion column wall and accretion column cap respectively.*

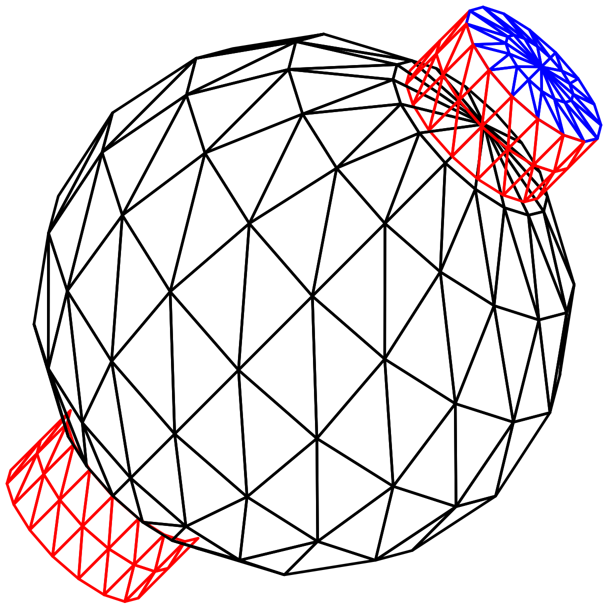


Fig. 3.4: *Composition of a neutron star with antipodal accretion columns provided by* `lb_obj_build_ns`.

3.3. Mapping

In this step a table for the different parameter of the photon trajectory is created. This is a necessary step, as it is not possible to obtain the emission angle α and impact parameter b for a given emission radius R and angle Ψ, if trajectories with periastron are allowed. The reason is that for the given values (R, Ψ) it is not a priori known, whether a periastron exits or not. Therefore, the periastron r_{p} (Eq. 2.20) and the according Ψ_{p} has to be calculated at first. The determination of the periastron, however, requires the impact parameter b, which is unknown by then. But for each spot on the object, specified by (R, Ψ), we want to determine the according emission angle and impact parameter. Hence, a lookup table is needed. Fortunately, this lookup table only has to be two dimensional as the trajectories lie in a plane (see Sec. 2.2). Moreover, a lookup table is even an advantage in the case that the exact equations are used, as a time-consuming numerical integration for every spot at every phase step can be avoided.

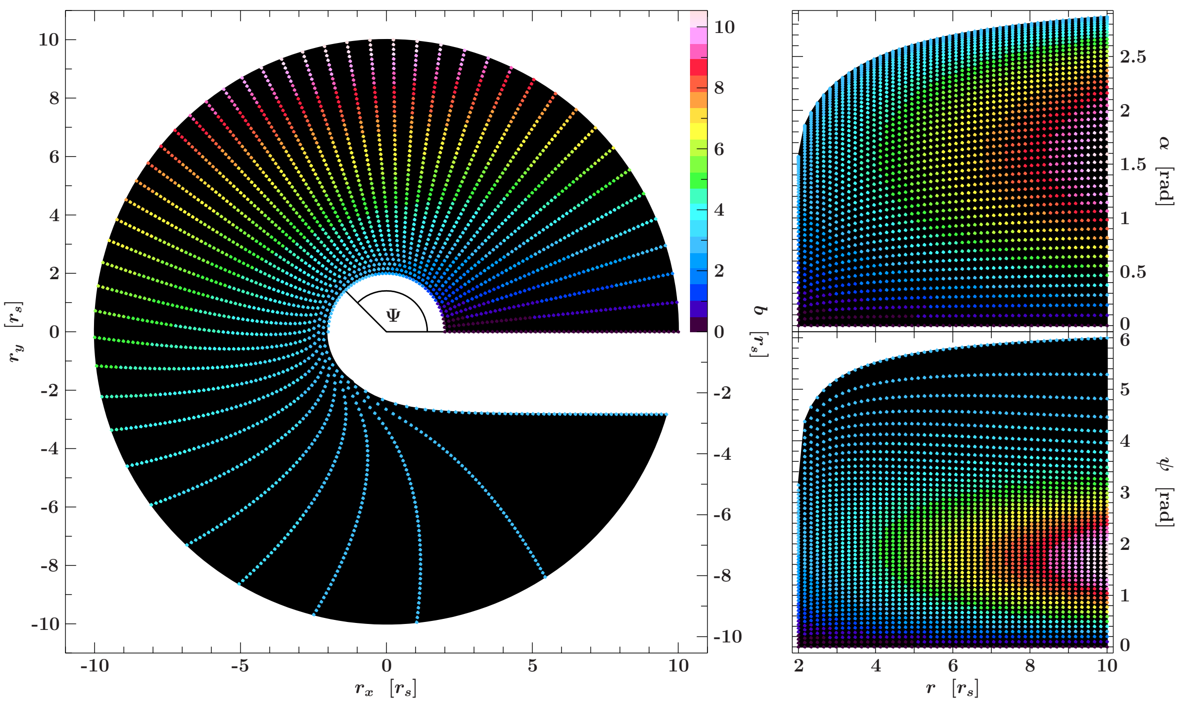


Fig. 3.5.: *Visualization of the table parameter. Right top and bottom panel shows the distribution of the points in the r-α and r-Ψ plane, where each point contains the values (r, α, Ψ, b). The left panel shows the same as the right top panel, but in polar coordinates. The black shaded region mark the boundaries given by* rmin*,* rmax*, $\alpha_{\max}(R)$ (Eq. 2.25) and accordingly $\Psi_{\max}(R)$ (Eq. 2.26). The value of the impact parameter of each point is given by its color. The color scale for the values of b is shown in the left panel, to which the second y-axis corresponds.*

3.3.1. Lookup table

Now we will look more closely into the creation of the lookup table. To give the emission angle and radius, and then calculate the according apparent emission angle with Eq. 2.21 and the impact parameter with Eq. 2.23 is the most convenient approach to obtain all required information about the photon trajectory. To create a lookup table this calculation is done for nmr equidistant radii R within [rmin, rmax] and nma different emission angles within $[0, \alpha_{\max}(R)]$, where $\alpha_{\max}(R)$ depends on the emission radius R (Eq. 2.25). In contrast to the equidistant radii, the step size between each α is set to follow the distribution given by

$$\alpha_n = \alpha_{\max}(R)\left[1 - \left(1 - \frac{n}{\mathsf{nma} - 1}\right)^{\mathsf{ashape}}\right] \quad , \tag{3.1}$$

where $n \in [0, \mathsf{nma} - 1]$, which is supposed to improve the coverage of evaluated points simultaneously in the r-α and r-Ψ plane. Otherwise, choosing equal step sizes for α as well, the points in the r-Ψ plane would be concentrated at low values of Ψ, meaning that the distance of adjacent Ψ's close to the maximum $\Psi_{\max}$ (Eq. 2.26) would be greater than those close to zero by a huge factor. Figure 3.5 visualizes the table with its trajectory parameters calculated based on the approximation (Eq. 2.38), for which ashape = 1.65 turned out to be the best value (see Sec. 3.4). As seen in the left panel of Fig. 3.5 the coverage for large Ψ is noticeably lower than for small Ψ. However, receiving photons from this region is unlikely, if the object is a

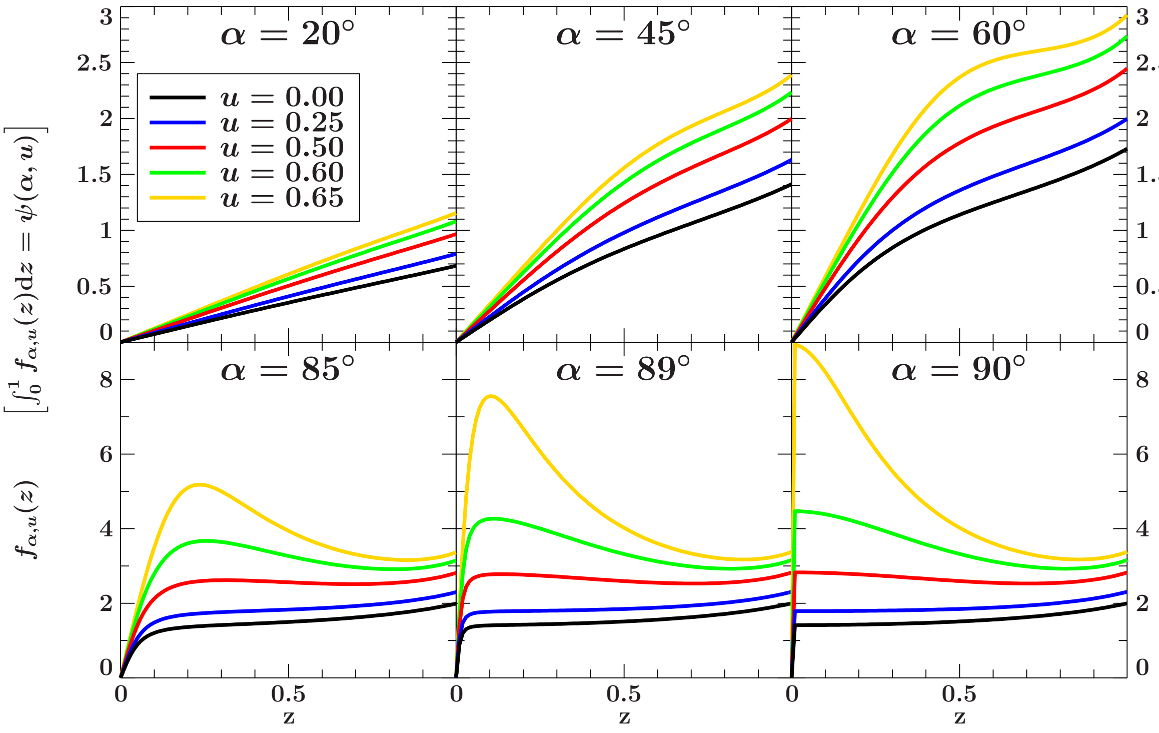


Fig. 3.6.: *Integrand $f_{u,\alpha}(z)$ (Eq. 3.3) for different values of u (color coded) and different emission angles (panels).*

neutron star with reasonable sized accretion columns. Moreover note that this example table underlying Fig. 3.5 has only (nma × nmr) = (50 × 50) points and is supposed for visualization purposes. nma relates to the number of individual emission angles α and nmr to the number of emission radii the table is created with. Besides the ratio of these numbers, which can be chosen arbitrarily, the point coverage of the table depends on the minimal (rmin) and maximal (rmax) radius of the object. The different photon trajectories also can be seen in Fig. 3.5, especially in the left panel, as the impact parameter, which is color coded, is constant over the whole trajectory. Meaning that points of the same color relate to the same trajectory.

3.3.2. Numerical solution to the exact photon trace

Besides the approximated solution (Eq. 2.38) for the photon trajectory, it is also possible to create a table based on the exact photon trajectory (Eq. 2.11). Unfortunately the integrand of Eq. 2.11 diverges for $r \to r_\mathrm{p}$ as shown in Sec. 2.2.3. Hence, Eq. 2.11 is unsuitable for simple numerical integration methods. However, there is a substitution avoiding this divergence:

$$\Psi = \int_0^1 \mathrm{d}z\ f_{u,\alpha}(z) \quad , \tag{3.2}$$

where $z \coloneqq \sqrt{1 - R/r}$ and

$$f_{u,\alpha}(z) = 2z\sin\alpha\left\{\cos^2\alpha(1-u) + z^2\sin^2\alpha\left[(2-z^2)(1-u) - u(1-z^2)^2\right]\right\}^{-1/2} \quad . \tag{3.3}$$

Figure 3.6 shows the integrand $f_{u,\alpha}(z)$ for different inverse emission radii u and for different emission angles α in each panel. As can be seen, $f_{u,\alpha}(z)$ does not diverge at the periastron, but forms a sharp edge for $\alpha = \pi/2$ and $z \to 0$. However, overall $f_{u,\alpha}(z)$ is well behaved, especially for $u < 0.5$. Therefore, it is sufficient to use simple integration methods to solve this integral (Eq. 3.2). The standard method, which is used in the present case, is the Romberg method adapted from Press et al. (1992, Sec. 4.3). The Romberg method solves Eq. 3.2 with high accuracy within appropriate time scales. Only for $\alpha \to \pi/2$ the calculation time goes noticeably up for requested relative accuracies below $\sim 10^{-6}$ due to the steep slope. Nevertheless, this accuracy is sufficient for our interests.

3.4. Interpolation

The `mapping` module, described in the last section, provides a table for all important parameters (r, α, Ψ, b) of the photon trajectory on a certain grid. After creating the object and the lookup table, we want to obtain these parameters for a given point (R, Ψ) on the object, which can be achieved by interpolation. In the following the interpolation method used in the present routine is described.

3.4.1. Bilinear interpolation

Figure 3.7, which is basically a zoom in of the bottom right panel in Fig. 3.5, sketches the different steps of the interpolation procedure to determine the emission angle α and impact parameter b related to a given point (R, Ψ). The basic idea is to perform a bilinear interpolation. However, an additional step is necessary before a bilinear interpolation can be performed. The reason is that the grid in the r-Ψ plane of the table is not completely rectangular, which is a requirement for this kind of interpolation. As can be seen in Fig. 3.7, the points are aligned in columns of constant radius. Based on this condition, two vertices with the same $r_1 < R$ and $\Psi_1 \le \Psi < \Psi_2$ can be found. Afterwards, the according second two vertices with $r_2 > R$ at Ψ_1 and Ψ_2 are obtained by interpolation. These four vertices now compose the rectangular required for the bilinear interpolation, with which eventually α and b are determined. Note that for $\Psi \to \Psi_{\text{max}}$ it can happen that there is no vertex (r_1, Ψ_2) with $\Psi_2 > \Psi$. In that case the two left sided vertices relate to the two last points in that r-column and the second step in the bilinear interpolation is an extrapolation instead (see Fig. 3.7, right panel).

3.4.2. Interpolation accuracy

A simple method to get a feeling for the quality of the interpolation is to obtain the interpolated values α_I and b_I for a given point (R, Ψ), then calculate $b(R, \alpha_\text{I})$ (Eq. 2.23) and $\Psi(R, \alpha_\text{I})$ (Eq. 2.21) and compare these with the given Ψ and b_I. Figure 3.8 shows the result for this testing procedure for various combinations of R and Ψ. Obviously the accuracy of the interpolated Ψ value significantly decreases the closer it is to the maximum value Ψ_{max}. The reason is that $\mathrm{d}\Psi/\mathrm{d}\alpha$ and accordingly the changes between each grid point are highest for $\alpha \to \alpha_{\text{max}}$. Therefore, the accuracy of the interpolation in this region is worse, as linear interpolations are used. In principle, the interpolation to obtain the two missing vertices could be replaced by a polynomial interpolation and additionally using more points, but doing so shows no noticeable improvements in the accuracy. Another way to account for this loss in accuracy is to increase the point coverage in this region in the table, which was already done (see Sec. 3.3) by introducing a parameter `ashape`, which controls the distance between two adjacent points (Eq. 3.1). Note that the values of the accuracy relates to interpolations based on

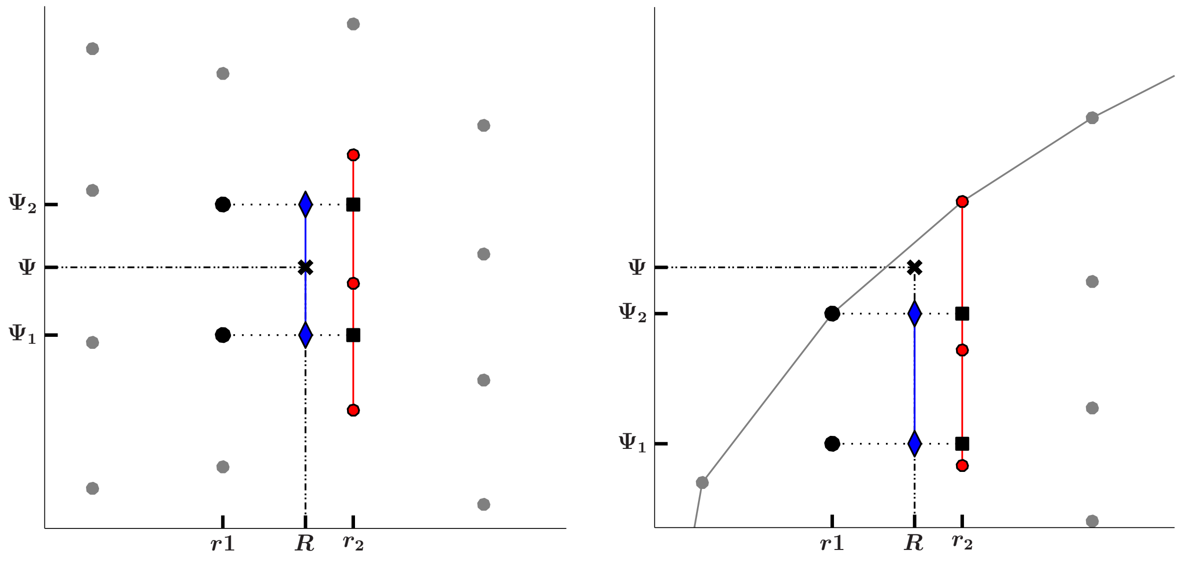


Fig. 3.7.: *Sketch of the individual interpolation steps based on a given table (see Fig. 3.5). For a given point (R, Ψ) (cross), in the beginning the two left sided vertices (r_1, Ψ_1) and (r_1, Ψ_2) (black circles) are identified in the table, where the circles mark the points included in the table and the gray line relates to Ψ_{max}. For a bilinear interpolation also (r_2, Ψ_1) and (r_2, Ψ_1) (squares) are needed. As these are not included in the table, the according values of the emission angle α_{ij} and impact parameter b_{ij}, with $i = 1, 2$ and $j = 2$, are obtained by interpolation of the surrounding points in the r_2 column (red circles). The next steps follow those of a bilinear interpolation. At first α_i and b_i at (R, Ψ_i) (blue diamonds) for $i = 1, 2$ are determined by linear interpolation of the two according upper ($i = 2$) and lower ($i = 1$) vertices (r_1, Ψ_i) and (r_2, Ψ_i). Finally α and b related to (R, Ψ) are determined by interpolating, or if necessary extrapolating (right panel), these two points (blue diamonds).*

the table shown in Fig. 3.5, which has only 50×50 points for a large range of radii (rmin = $2r_s$, rmax = $10r_s$). The accuracy significantly increases with increasing number of table points.

3.5. Projection

The `projection` module presented in this chapter is the core of the routine. This module projects every surface element of the object at different preset phases onto the observer sky using the previously calculated table of trajectory parameters. In addition to the given (R, Ψ) values the according emission angle α, the Doppler factor δ and the emission offset angle γ are provided, where γ is defined as the angle between the initial emission direction $\boldsymbol{k}_0$ and the surface vector $\mathbf{d}\boldsymbol{S}$ of the surface element. Besides these parameters, which are needed to determine the observed flux, also the impact parameters A and B for the centroid and vertices are calculated, which are necessary to deal with projection effects related to the finite spot size. A and B are the x and y components of the impact parameter b on the plane of the observer sky, where $b = \sqrt{A^2 + B^2}$. For a detailed discussion of these parameters see Sec. 2.3.3.

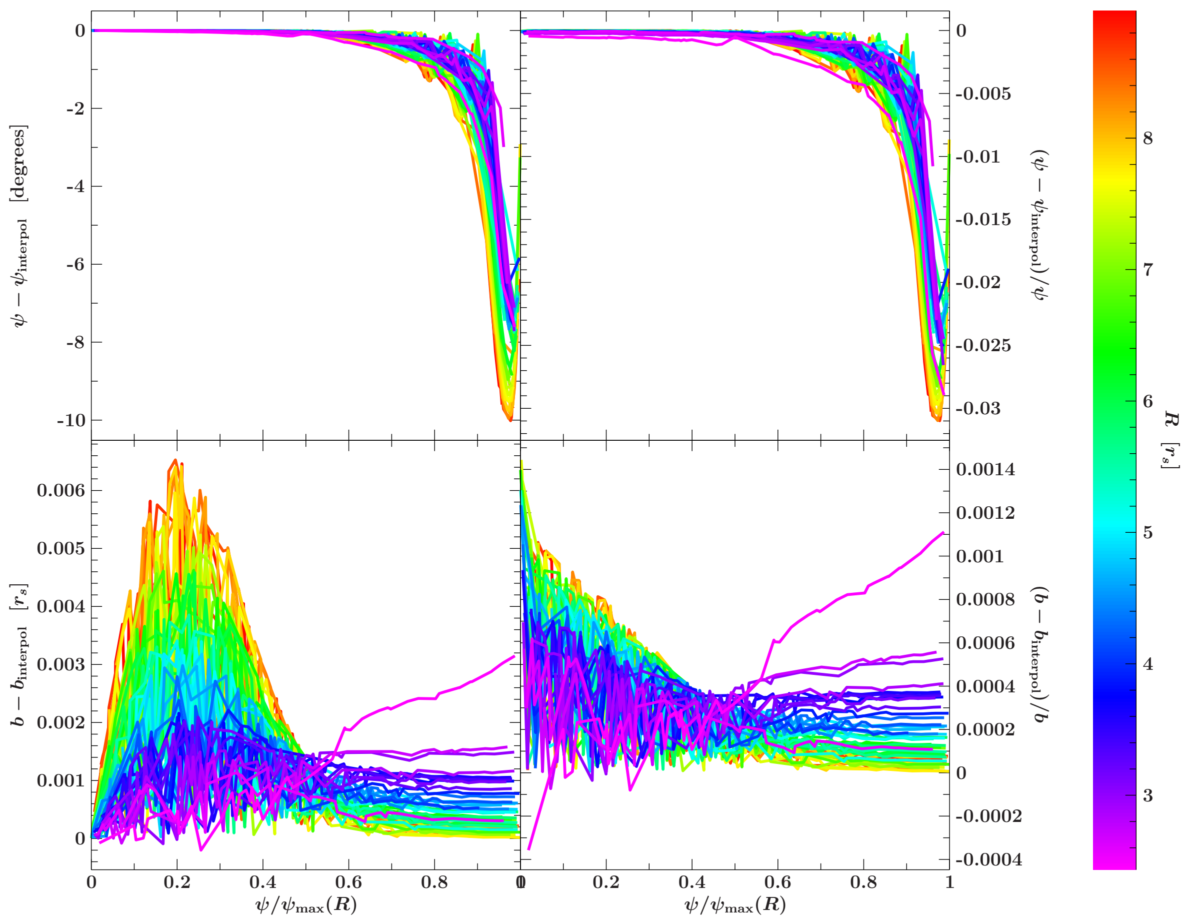


Fig. 3.8.: *Absolute and relative accuracy of the interpolation procedure based on the table shown in Fig. 3.8. The top panels show the comparison between a given Ψ and $\Psi_{interpol}$, which is calculated using Eq. 2.21 and the interpolated value $\alpha_{interpol}$, where the color relates to different emission radii. The bottom panel shows the same for the impact parameter.*

3.5.1. Types of Photon Trajectories

First of all we have a closer look at the entirety of the photon trajectories, which can be classified into different regions. As seen in Fig. 3.9 there are four distinguishable regions. Depending on the radius R of the object a critical photon trajectory can be specified, whose periastron equals the radius of the object and therefore is a limit for trajectories exhibiting a periastron. This limit can be quantified with the according limiting impact parameter

$$\begin{aligned} b_{\mathrm{lim}} &:= b(\alpha = \pi/2, R) \\ &= \frac{R}{\sqrt{1 - r_{\mathrm{s}}/R}} \quad , \end{aligned} \tag{3.4}$$

similar to the critical impact parameter b_{c} (Eq. 2.19). All trajectories with a smaller impact parameter are without a periastron, meaning they end on the object surface.

Otherwise the trajectories exhibit a periastron. These trajectories can be separated again. Each line in Fig. 3.9 corresponds to a photon trajectory. Noticeable is the region behind the object, from the observer point of view, in which the trajectories intersect. A photon emitted

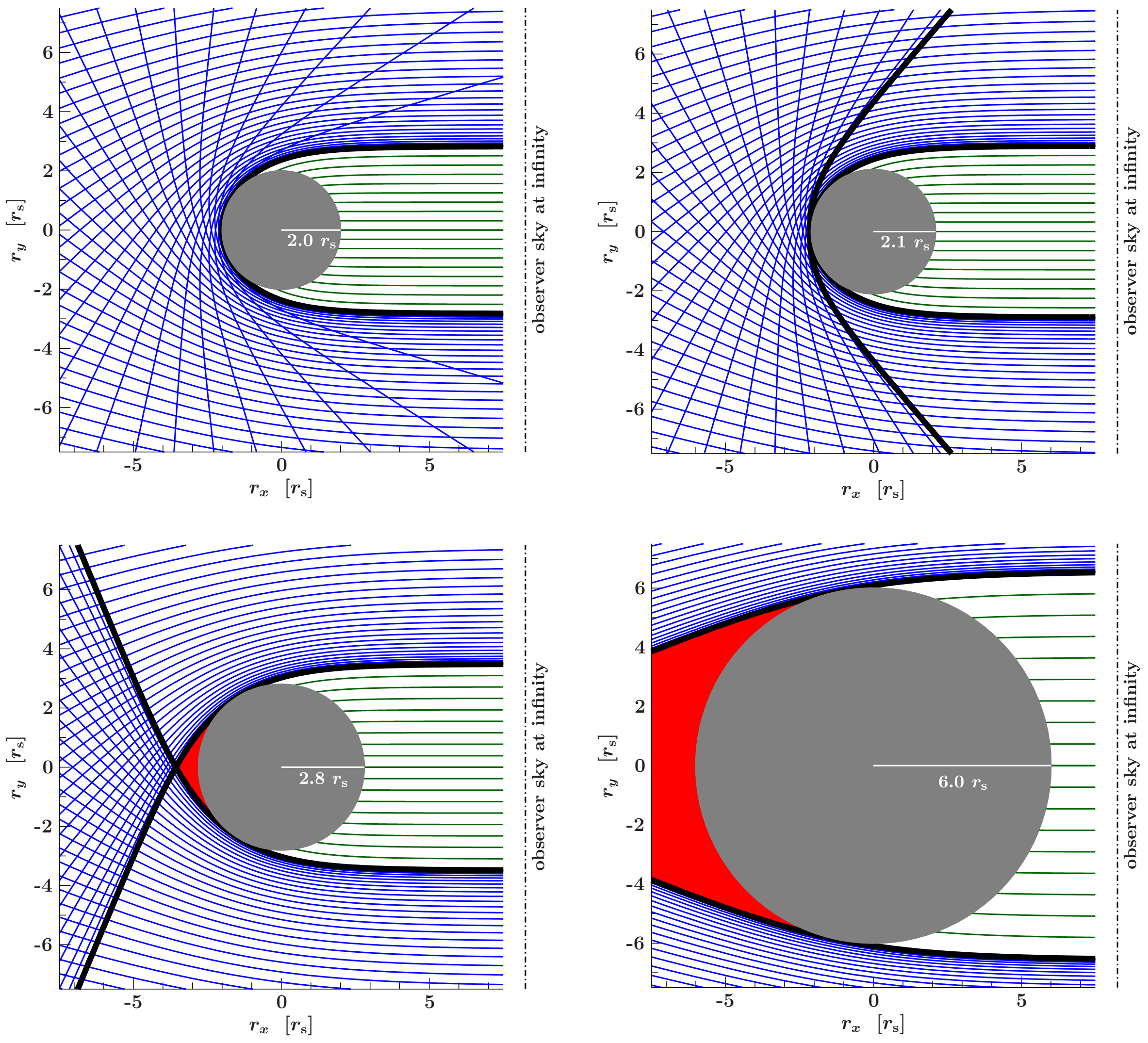


Fig. 3.9.: *The four panels in this figure show various photon trajectories (blue & green lines), which can be classified in different regions separated by a maximally bent trajectory (black line) with $b = b_{\rm lim}$, where the periastron equals the radius of the object. There are trajectories without a periastron (green lines) and those with periastron (blue lines). For the later ones there is also a region, in which photons have two possible trajectories to reach the observer (blue intersecting lines). Photons emitted in the red region can not reach the observer. Note that all trajectories are calculated using the analytical approximation (Eq. 2.38).*

at such an intersection can follow two possible trajectories to reach the observer, where the relation between the two solutions is $\Psi_2 = 2\pi - \Psi_1$ for the same radius. The apparent emission angle at the periastron $\Psi_{\rm p}$ is always less than 180° for $R < 2r_{\rm s}$, and hence, at least one of the two possible trajectories exhibits a periastron. This region enlarges for decreasing radii of the object in expand of the region, in which there is only one possible trajectory.

Additionally, there is also a region directly behind the object with respect to the observer sky, from which no photon can reach the observer. Note that it can not easily be determined, whether or not there are solutions for a given point (R, Ψ). If there exists a solution for a photon emitted at (R, Ψ), the other solution $(R, 2\pi - \Psi)$ cannot be ruled out. Therefore, both

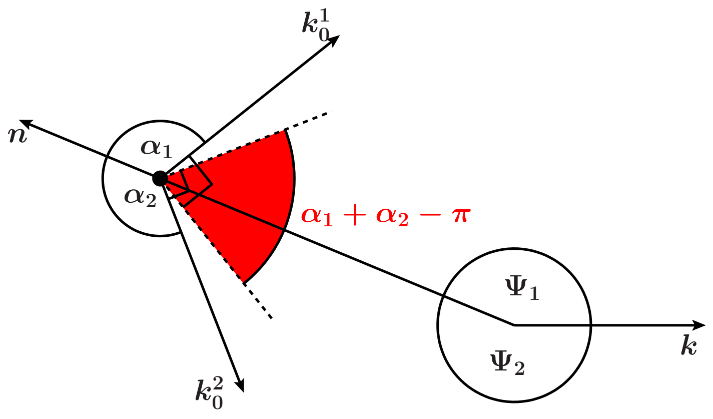


Fig. 3.10: *Scheme showing the allowed directions (red shaded angle) of the surface vector in order to have two possible trajectories a photon emitted with initial emission directions $\boldsymbol{k}_0^1$ and $\boldsymbol{k}_0^2$ at (R, Ψ) (black dot) can follow to reach the observer.*

possibilities have to be considered.

Furthermore, the restriction of $R > 2r_\mathrm{s}$ prevents regions with three or more potential trajectories. The top left panel of Fig. 3.9 shows the limit $R = 2r_\mathrm{s}$, in which a trajectory with the deflection angle 180° exists. Note that this limit relates to the analytical approximation (Eq.2.38). The approximation, however, always overestimates the light bending effect and therefore there are also no regions with more than two solutions for the exact calculations, if $R > 2r_\mathrm{s}$.

3.5.2. Projection Procedure

The basic projection procedure follows the idea of this classification. After rotating the object to the current phase, for the centroid of each surface element the coordinate (R, Ψ) is determined, where Ψ is given by Eq. 2.58. Afterwards, the trajectories regarding (R, Ψ) and $(R, 2\pi - \Psi)$ are calculated and matched with the limiting trajectory defined by b_lim. There is an additional criterion regarding photons in order to be emitted in the first place, namely that the emission offset angle γ' in the corotating frame has to be less than $\pi/2$. Otherwise the photon would be emitted toward the surface. For each possible solution the remaining parameters of the trajectory are calculated with respect to the centroid of the surface element. In the end for each solution of each surface element's centroid, the parameters $(\alpha, \gamma, \delta, A, B)$ are known. In addition also the impact parameters of the vertices of the surface elements are determined, which are needed for further selections (e.g., overlaps, see Sec. 3.6).

3.5.3. Uniqueness of the Trajectory for Neutron Stars with Accretion Columns

Note that it is very unlikely for one surface element to have two actual solutions $\boldsymbol{k}_0^1$ and $\boldsymbol{k}_0^2$, as this case would require that the according emission offset angles, γ_1 as well as γ_2, are smaller than $\pi/2$. This condition is only fulfilled, if the surface vector $\mathbf{d}\boldsymbol{S}$ is directed toward the center of mass of the gravitating object, meaning $\mathbf{d}\boldsymbol{S} \circ \boldsymbol{n}/|\mathbf{d}\boldsymbol{S}| < 0$, where the direction range is limited by $\pi - (\gamma_1 + \gamma_2) = \alpha_1 + \alpha_2 - \pi$ as shown in Fig. 3.10. This range shrinks with decreasing emission radius R and Ψ. The reason for this restriction is that at least one of the two solutions relates to a trajectory with an emission angle $\alpha > \pi/2$ and $\boldsymbol{k}_0^1$ as well as $\boldsymbol{k}_0^2$ lie in different semi-planes of the $\boldsymbol{k}$-$\boldsymbol{n}$ plane divided by $\boldsymbol{n}$. Only a photon source detached from the neutron star surface could fulfill this requirements and provide two trajectories reaching the observer.

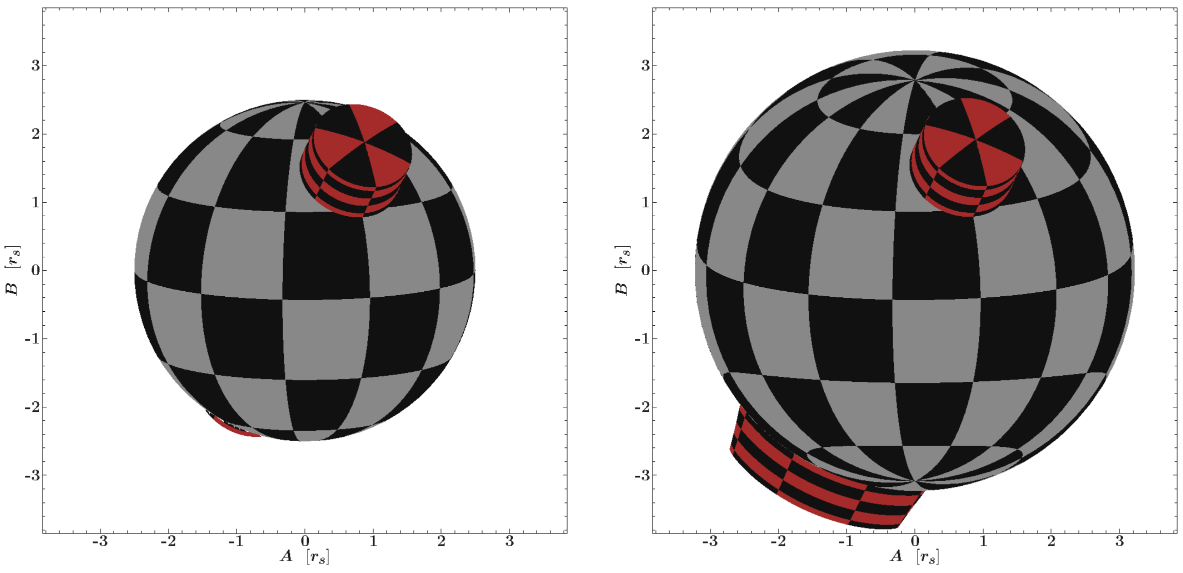


Fig. 3.11.: *Geometrical (left panel) and relativistic (right panel) projected neutron star with $R = 10$ km and $M = 1.35$ M$_\odot$ seen under an inclination of $i = 80°$ at a phase $\phi = 0.06$, where $\phi = 0$ corresponds to the phase, in which the upper accretion column is in the very front. The two antipodal accretion columns have a radius $R_{AC} = 3$ km, a height of $h_{AC} = 3$ km and an inclination of $i_{AC} = 45°$ with respect to the spin axis of the neutron star.*

3.5.4. Relativistic versus Geometrical Projection

Figure 3.11 shows the comparison of the geometrical and the relativistic projection based on the analytical approximation (Eq. 2.38). The most obvious effect of the relativistic light bending is the apparent enlargement of the projected object radius compared to its real radius, where the apparent radius is b_{lim}. The enlargement is connected to the visibility of a higher fraction of the surface, which is given by

$$\begin{aligned} S_{\text{v}}/S &= \frac{1}{2}(1 - \cos\Psi_{\text{p}}) \\ &= \frac{1}{2(1-u)} \quad , \end{aligned} \tag{3.5}$$

for the approximation (Eq. 2.38). For the relativistic example shown in Fig. 3.11 the fraction $S_{\text{v}}/S = 0.83$, which is 33% higher than in the geometrical case. For example, both poles of the spin axis are visible in the relativistic case in contrast to the geometrical one. Moreover, this effect also increases the visibility of the accretion columns. Due to the relativistic light bending the wall of the rear accretion column is completely visible, whereas in the geometrical projection only a small fraction is visible. Also noticeable is that there is a significant stretching in tangential direction of surfaces close to the maximal apparent emission angle.

3.6. Overlaps

In Sec. 3.2 it was already mentioned that there are surface elements, which overlap in the observer sky. For example, the upper accretion column in Fig. 3.11 covers parts of the neutron star surface. To avoid including photons emitted from this covered region into the further calculations, the according surface elements have to be determined. The approach to identify these overlaps implemented in the present routine and related issues are discussed in the following.

To find overlapping regions in the observer sky, we look at the projections of the individual surface elements. We now have to compare each of those projections and spot those, which are overlapping in the A-B plane. It is not necessary to compare every projected surface element to each other, but only those very close. Looking at the individual projection of the surface elements one can a priori exclude those elements from the comparison procedure, between which the distance is greater than the sum of their circumferences. Taking advantage of this fact reduces the number of individual comparisons dramatically. Therefore, the observer sky is divided into cells by overlying it with a grid, where the size of the cells is defined by the maximal occurring circumference of the projected surface elements. Then a given projection only has to be compared to those within the same cell and those of the next neighbor cells. The basic idea of the comparison itself is to check whether or not for a given projection, there exists another projection, whose centroid is enclosed by the projected vertices of the given surface element. The algorithm[5] used to perform this check is based on Orwant et al. (1999, Chap. 10). Two surface elements are defined to overlap, if the centroid of the projection of one of these surface elements lies within the projected area of the other one. The next step is to determine, which of these two overlapping elements is covering the other, which can be achieved by comparing the apparent emission angles accordingly.

Note that this procedure includes approximations, namely the definition of overlap. A more accurate way would be to perform the check for every vertex instead of only for the centroid. However, this would increase the processing time at least by a factor three. Furthermore, in case of an overlap, the contribution of the covered surface element to the flux is completely ignored, instead of only the covered fraction. The calculation of the real covered fraction would be very complex, as the covered fraction in the observer sky does not correspond to the covered fraction in the corotating frame. The reason for this mismatch is related to the deformation of the plane of the surface element introduced by the projection (see below). However, the error introduced by these assumptions can be reduced by downsizing the surface elements. Furthermore, this procedure is not always necessary. For example, in the case of a neutron star with accretion columns, where only emission from the accretion columns is considered. In order to test, whether or not it is necessary to find overlaps, the projection can be plotted (see, e.g., Fig. 3.11) with the provided plotting functions (see App. C).

3.6.1. Deformation

Besides these general effects of the light bending described in the last section, there are also effects related to the finite size of the surface elements, with which the object is sampled. The relativistic projection can cause a deformation of the plane of a surface element, such that the projection of the vertices, defined in the corotating frame, no longer encloses the projected surface. Figure 3.12 shows an example of such a deformation. For simplicity we consider a linear surface element, which is specified by the two vertices (R, Ψ_1) and (R, Ψ_2). In the projection procedure described above, we normally only project these vertices and the according centroid. In Fig. 3.12 for every point of the surface element the according impact parameter is given. Note that there are points in that plane with impact parameters outside the range enclosed by the vertices. Therefore, the projection of the vertices of the

[5]In our case the surface elements are triangular, which simplifies this procedure.

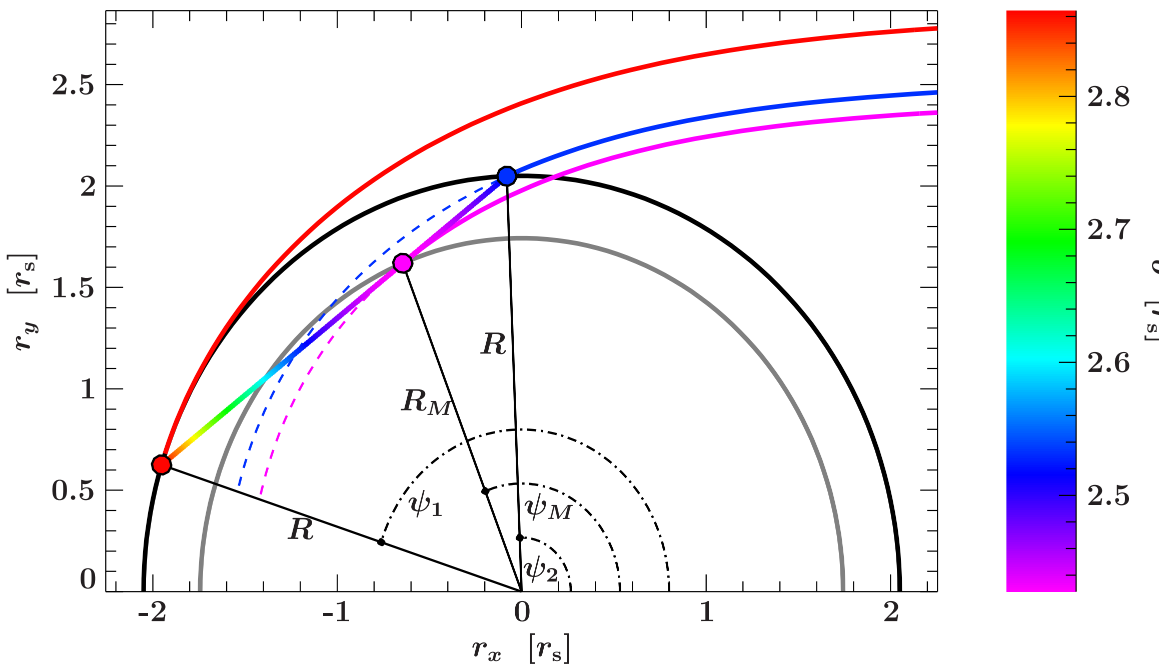


Fig. 3.12.: *Scheme showing the deformation of the plane of a surface element two dimensionally simplified.* (R, Ψ_1) *and* (R, Ψ_2) *relate to the vertices of the surface element and* (R_M, Ψ_M) *to the location with the minimal impact parameter, where the values of the impact parameter are color coded. Additionally the trajectories to the three mentioned points are drawn.*

original surface element do not define the projection properly. This deformation effect can be minimized by reducing the size of the surface elements, as the degree of the deformation depends on the spread of the radius R combined with the spread of the emission angle α over the surface element (see Fig. 3.5).

3.6.2. Magnification

Another effect related to the finite size of the surface elements is the occurrence of excessively magnified projected surface elements, where the diameter of the projected surface can reach $2b_{\mathrm{lim}}$. This magnification originates from large angles between the individual $\boldsymbol{k}$-$\boldsymbol{n}$ planes of the different vertices of a surface element and the fact that the relativistic light bending enlarges distances in radial direction. This effect especially occurs for large surface elements or ones near or along the line of sight on the other side of the gravitational center ($\Psi \to \pi$). The second case, however, can be neglected for neutron stars with accretion columns as this region is not visible or would require surface elements, whose surface vector are directed towards the neutron star (see Fig. 3.9). Therefore, a sufficiently small size of the surface elements is enough to minimize this effect.

It is worth mentioning that the two effects described above do not influence the calculations of the flux, as neither the shape nor the projected size of the surface elements are needed for its calculation (Eq. 2.70, Eq. 2.72). In contrary, the procedure of the overlap identification relies on the projections of the surface elements and therefore these effects have to be taken into account at this point. In this program, it is possible to set a maximum value `par.magnilim` for the relative magnification $\mathrm{d}S/\mathrm{d}S'$ of the size of a surface element and to get a list of those surface elements, which exceed this limit. Elements in that list will be excluded from the overlap procedure. Otherwise these deformed and oversized projected surface elements would cause to find overlaps, which do not exist.

3.7. Flux

The last step is to calculate the observed flux based on the preparations described in the previous sections. The calculation[6] is divided into two steps. At first, the flux of each individual surface element at every phase is determined based on the equation for the spectral (Eq. 2.70) or bolometric (Eq. 2.72) flux. Additionally, the emission profile I' has to be specified, which in general can be chosen arbitrarily. This routine provides the possibility to choose isotropic and homogeneous emission ($I' \equiv 1$), or emission depending on the emission offset angle γ', where $I'(\gamma')$ is either a uniform or a normal distribution with $I'(0) = 1$. Figure 3.13 shows an example for the relativistic projection at various phases of a neutron star with isotropic and homogeneous emission from the two identical and antipodal accretion columns.

In the second step the fluxes of the individual surface elements are combined to obtain an overall time and energy dependent flux information. We are not only interested in the overall observed flux, but also in the contribution of the individual parts of the object. Therefore the routine provides also the contribution of the individual parts to the flux beside the overall flux. This separation allows a detailed analysis and comparison. For example, we can create a pulse profile and compare the contribution of each of the emitting parts. Figure 3.14 shows the pulse profile of the neutron star shown in Fig. 3.13 for every single part and their possible combinations. Distinguishable are the contributions of the surface of the neutron star (NS), the individual accretion column caps (cap) and walls (wall). Additionally, different combinations are provided, namely the sum of the accretion column caps (pencil), the sum of the walls (fan), the individual accretion columns (AC) in total and the overall contribution of both accretion columns. In principle, it is also possible to define arbitrary regions on the object surface.

To get a feeling for the impact of the choice of the number of surface elements (#SE) the object is build up with (see Sec. 3.2) and the size of the mapping table ($\#R \times \#\alpha$), with which the light bending parameters are interpolated (see Sec. 3.3, Sec. 3.4), we vary these parameters for a given object. Figure 3.15 shows the pulse profile of Fig. 3.14 calculated with different amounts of surface elements. In this case #SE relates to the number of surface elements the accretion column is sampled with as the neutron star surface is not emitting in this case. Also notice that the relative difference of the pulse profiles is approximately constant over all phases. Furthermore, lowering the amount of surface elements reduces the mean flux, i.e., the flux is underestimated.

Figure 3.16 shows the pulse profile of Fig. 3.14 based on different mapping tables with varying number of points $\#R \times \#\alpha$. This figure shows that the impact is very small, although the tables are very small. The explanation for this small impact is that the relation between the apparent emission angle Ψ and the emission angle α is approximatively linear for small values of α or respectively Ψ and therefore the bilinear interpolation (see Sec. 3.4) is very accurate, also for only few grid points. Note that during the projection procedure (Sec. 3.5) a

[6]The important parameters regarding the flux calculation are stored in an own `Struct_Type`, where the specification of spectral or bolometric flux is included. In the spectral case also information about the energy grid can be given (see App. C).

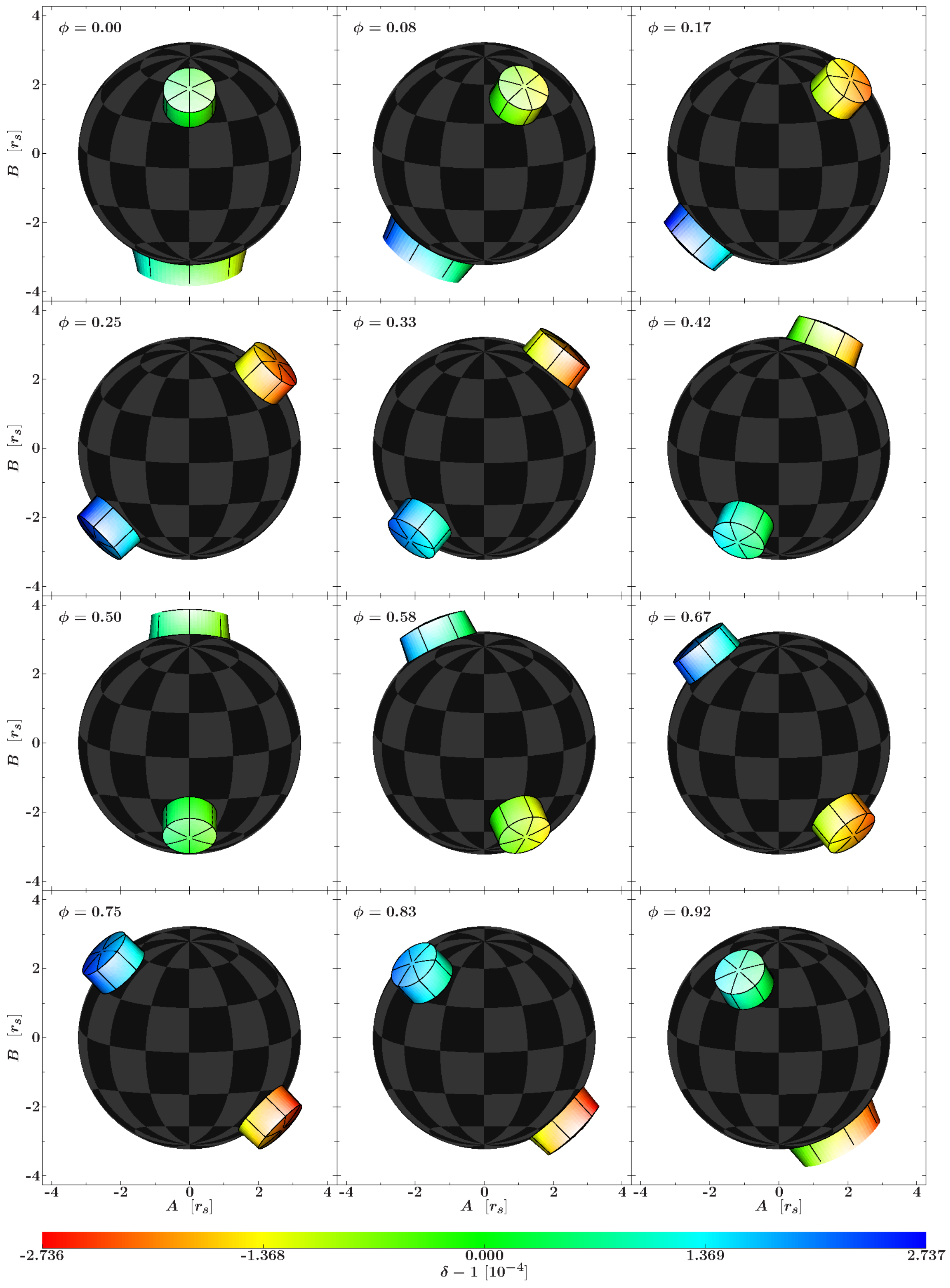


Fig. 3.13.: *Projection of a neutron star (R = 10 km, M = 1.35 $M_\odot$, i = 80°, f = 1 Hz) with isotropic and homogeneous emission ($I' \equiv 1$) from the antipodal accretion columns (R_{AC} = 3 km, h_{AC} = 3 km, i_{AC} = 45°) at different phases ϕ. The color relates to the Doppler factor δ and the brightness to the strength of the bolometric flux.*

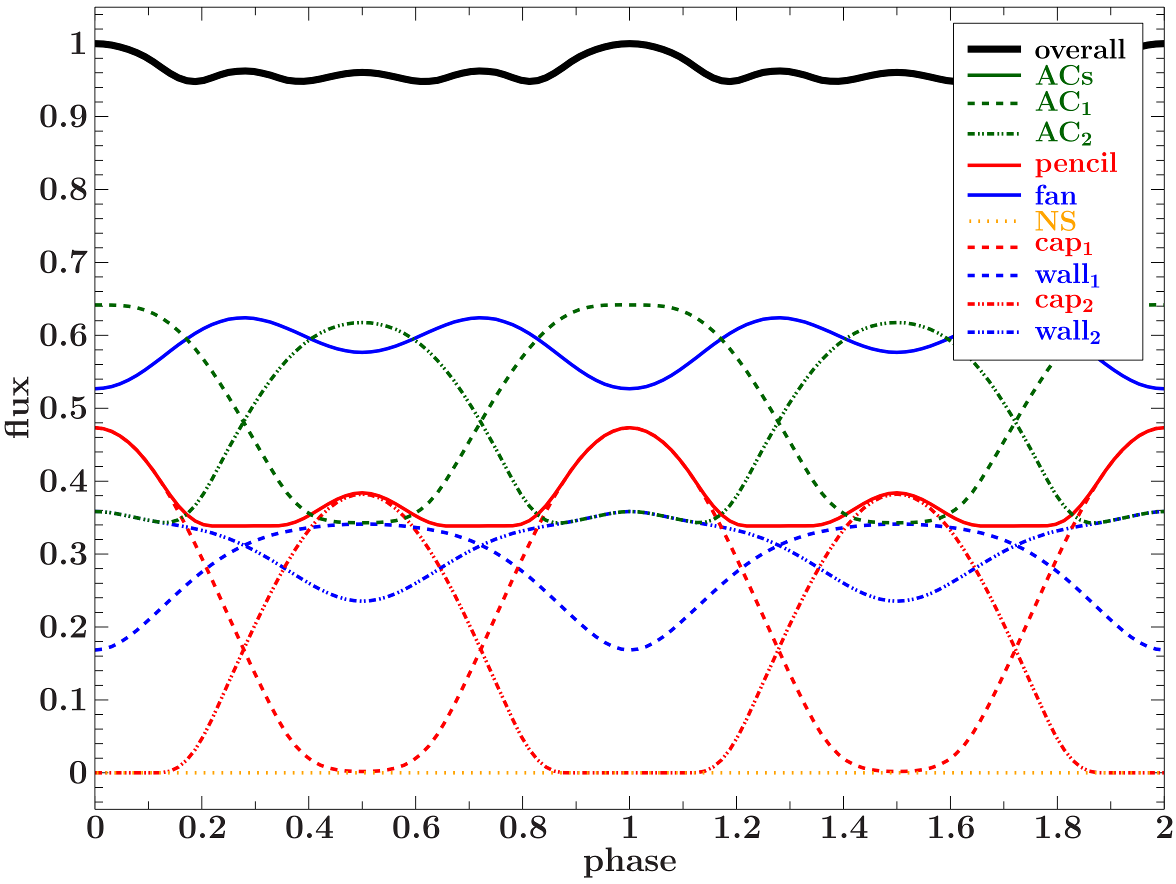


Fig. 3.14.: *Pulse profile of the neutron star shown in Fig. 3.13. The shown flux is considered to be bolometric, where the emission is isotropic and homogeneous ($I' \equiv 1$), and normalized to the maximum flux. Parts with index 1 relate to the upper accretion column and index 2 to the lower one.*

limiting impact parameter b_{lim} (Eq. 3.4) according to the radius of neutron star was defined. This definition also results in a limiting apparent emission angle Ψ_{lim}, which is always smaller than the maximal angle Ψ_{max}. Hence, angles Ψ within $[\Psi_{\text{lim}}, \Psi_{\text{max}}]$ are not allowed. In this case the radius of the neutron star is sufficiently large to ensure that Ψ_{lim} is small enough to avoid angles within the region, in which the accuracy of the interpolation drops as described in Sec. 3.4. However, $\Psi_{\text{lim}} \rightarrow \Psi_{\text{max}}$ for $R \rightarrow 2r_{\text{s}}$ and therefore the impact of the size of the mapping table increases with decreasing radius of the object.

In summary, the developed program presented in this Chapter is suitable to determine the time and energy dependent observed spectral or bolometric flux of a neutron star with accretion columns or hot spots accounting for general relativistic effects, namely gravitational redshift (Sec. 2.3.1) and moreover light bending (Sec. 2.2), and special relativistic effects, Doppler shift and aberration (Sec. 2.3). The program was designed to especially use the analytical approximation (Sec. 2.2.3) for the photon trajectory, which has the restriction $R > 2r_{\text{s}}$. Although the exact solution for the photon trajectory is not a subject to that restriction, the implemented numerical procedure solving the integral (Eq. 2.11) is optimized for the same range of R (Sec. 3.3.2). Therefore, the program in general request that $R > 2r_{\text{s}}$. Nevertheless, this limitation only exclude a small fraction of possible mass to radius ratios of

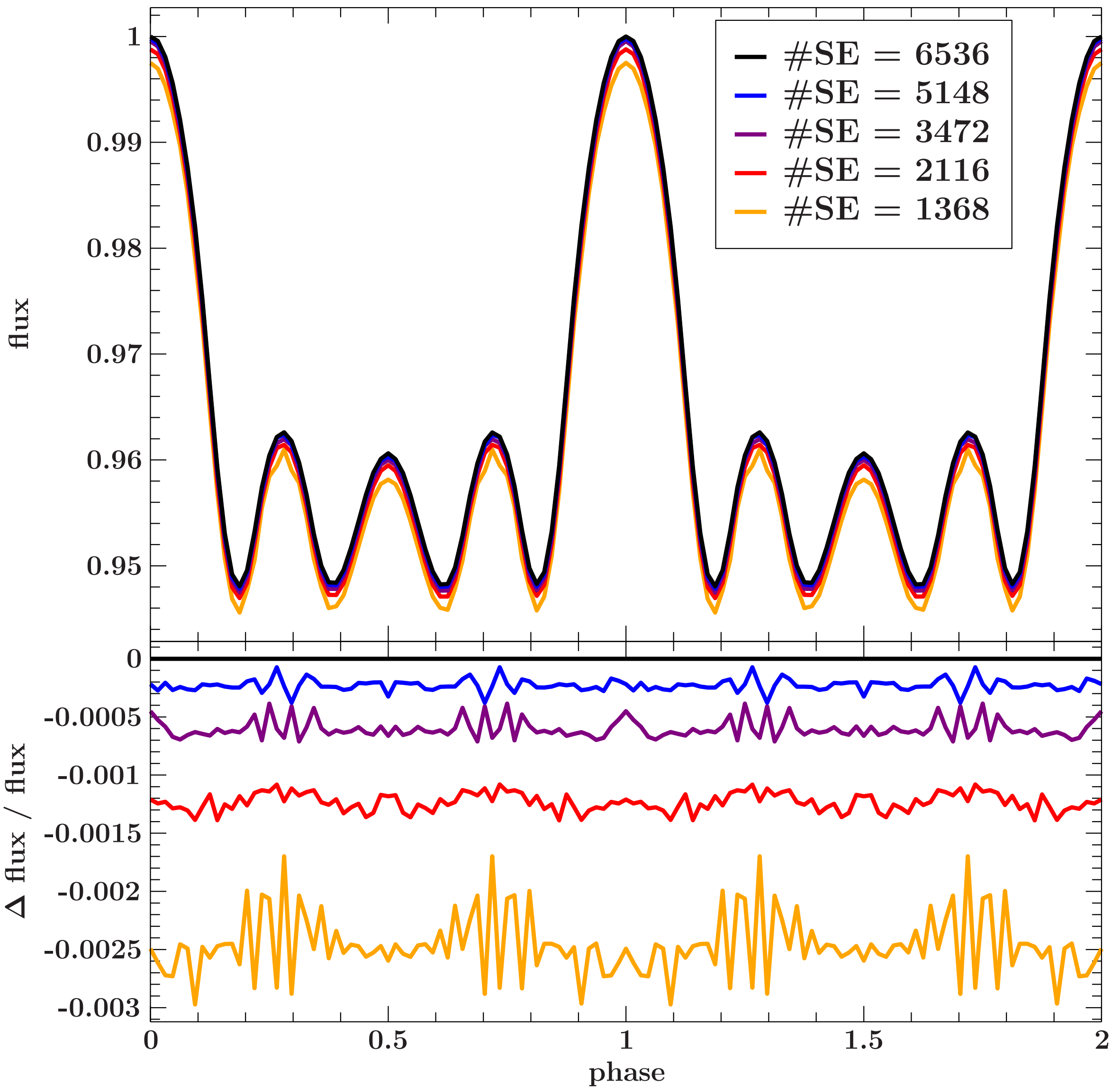


Fig. 3.15.: *Pulse profile for different number of surface elements (#SE). The pulse profiles in the top panel correspond to the overall pulse profile in Fig. 3.14, where the flux is normalized to the maximal flux in the pulse profile with the highest #SE. The bottom panel shows the relative differences of each pulse profile to the one with the highest #SE.*

neutron stars as seen Fig. 1.2. If needed, an extension for radii smaller then two Schwarzschild radii can be easily implemented.

Furthermore, the processing time to obtain the flux observation of a given object is very small. The processing time for the projection procedure at one phase mainly depends on the chosen number of surface elements and the size of the mapping table. This time has to be multiplied with the number of phases. Overall the processing time[7] to obtain the flux of an object over several phases ranges between several seconds and several minutes for adequate chosen settings.

[7]The calculations were performed on a machine with an Intel(R) Core(TM) i5 (3.20 GHz).

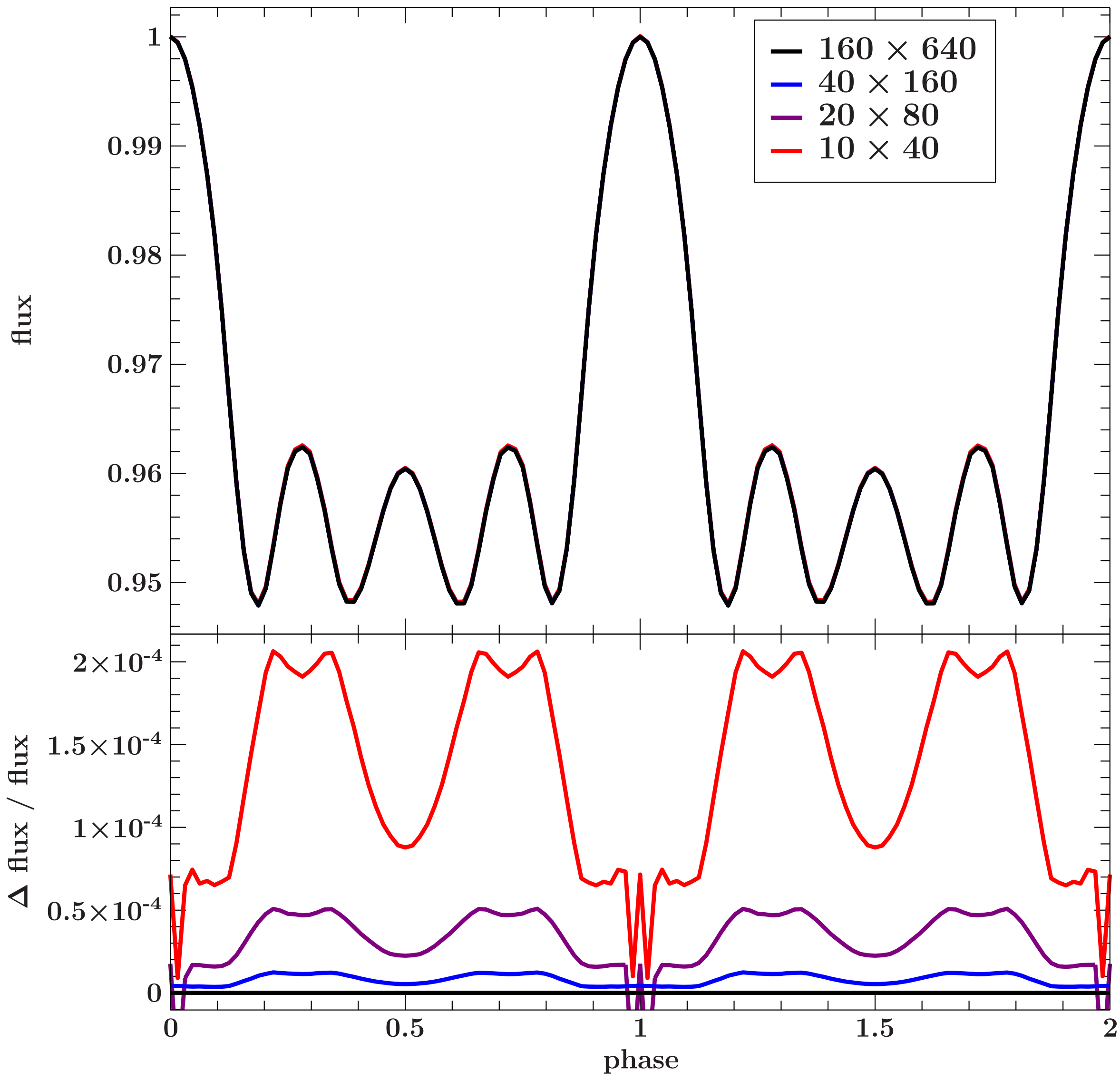


Fig. 3.16.: *Pulse profile for varying sizes of the mapping table ($\#R \times \#\alpha$). The pulse profiles in the top panel correspond to the overall pulse profile in Fig. 3.14, where the flux is normalized to the maximal flux in the pulse profile with the highest $\#R \times \#\alpha$. The bottom panel shows the relative differences of each pulse profile to the one with the highest $\#R \times \#\alpha$.*

Note that in this program the time delay discussed in Sec. 2.2.2 is not accounted for. However, the effect of the time delay, or more precisely the resulting phase shift $\Delta\phi$ (Eq. 2.74), is only important for high spin frequencies f' of the neutron star. This effect causes photons emitted simultaneously from the object to arrive the observer at different times. The higher the frequency the more important this effect gets. The maximal possible observed phase shift for a neutron star of radius R and an intrinsic spin frequency f' based on the approximated time delay (Eq. 2.43) is given by

$$\Delta\phi_{\max} = 4\pi \frac{R}{c\sqrt{1-u}} \cdot f' \tag{3.6}$$

where the expression $(1 - \cos\Psi) = 2$ as this is the maximum value this expression can have.

Considering a radius $R = 16$ km, for the entirety of the neutron star with accretion columns, and a mass $M = 1.4\ \mathrm{M}_\odot$ for the neutron star we get

$$\Delta\phi_{\max} \approx 8 \cdot 10^{-4} \left(\frac{f'}{Hz} \right) \tag{3.7}$$

The maximal observed phase shift is less than 0.013% for such a neutron star with $f' = 1$ Hz. Hence, the calculations performed with the presented program are sufficient for small frequencies, for which the time delay can be neglected.

To also account for the time delay, the flux contributions of the single surface elements have to be phase shifted accordingly and summed up based on a new phase grid. Note that the phase shift introduces an individual phase grid for each surface element.

Chapter 4

Results

In this Chapter we have a closer look at the observed flux of a neutron star determined with the program described in Chap. 3 giving an insight in the possibilities of the program. We are choosing a setup for our neutrons star and vary one specification, where the other parameters are fixed, and look at the individual pulse profiles. The basic setup is a neutron star with a radius $R = 10$ km and a mass $M = 1.35\ \mathrm{M}_{\odot}$, resulting in a Schwarzschild radius $r_{\mathrm{s}} = 4$ km and a mass to radius ratio $r_{\mathrm{s}}/R = 0.4$. The neutron star is spinning with a period of 1 sec ($f' = 1$ Hz), where the inclination of the line of sight to the spin axis is $i = 80°$. Further, the isotropic and homogeneous emission ($I' \equiv 1$) comes from the two antipodal accretion columns, which both have a radius $R_{\mathrm{AC}} = 3$ km and height $h_{\mathrm{AC}} = 3$ km, and are inclined to the spin axis of the neutron star by $i_{\mathrm{AC}} = 45°$. Besides the overall pulse profile of this neutron star, also the pulse profiles of the individual parts (see Sec. 3.7) will be given to show the contribution of each of these parts. Note that in the case of constant intensity ($I' \equiv 1$) the flux per area is constant. Furthermore, the flux scale in all figures in this chapter is relative to the respective one in the top left panel.

4.1. Comparison of the projection methods

At first we look at the pulse profiles based on the different projection methods provided by the program. Figure 4.1 shows the pulse profile using the relativistic projection determined with the exact photon trajectory (Eq. 2.11) and with the approximated photon trajectory (Eq. 2.38). Furthermore, the figure shows the pulse profile obtained with simple geometrical projection, i.e., $\Psi \equiv \alpha$. The relative difference of the two relativistic methods is less than 0.4% and shows the high accuracy of the analytical approximation suggested by Beloborodov (2002), whereas the shape of the geometrical overall pulse profile is totally different compared to the relativistic cases and also the mean flux is much lower. As seen in Fig. 3.11, the light bending causes the accretion columns to be visible over a longer time, which results in a broadening of the peaks seen in the individual pulse profiles, especially for the emission from the walls of the accretion columns. Disregarding the difference of their minimal and maximal flux, the shape of the relativistic as well as the geometrical fan beam contribution looks similar: In both cases there are two peaks at phase $\phi \sim 0.25, 0.75$. Distinguishing the contribution, however, of the individual walls of the accretion columns one sees that in the relativistic case each contribution is single peaked in contrast to the geometrical case. Fig. 3.13 shows that both relativistically projected accretion columns are visible over the whole rotation period. Moreover, the flux of the individual walls peaks when the according accretion column is in the

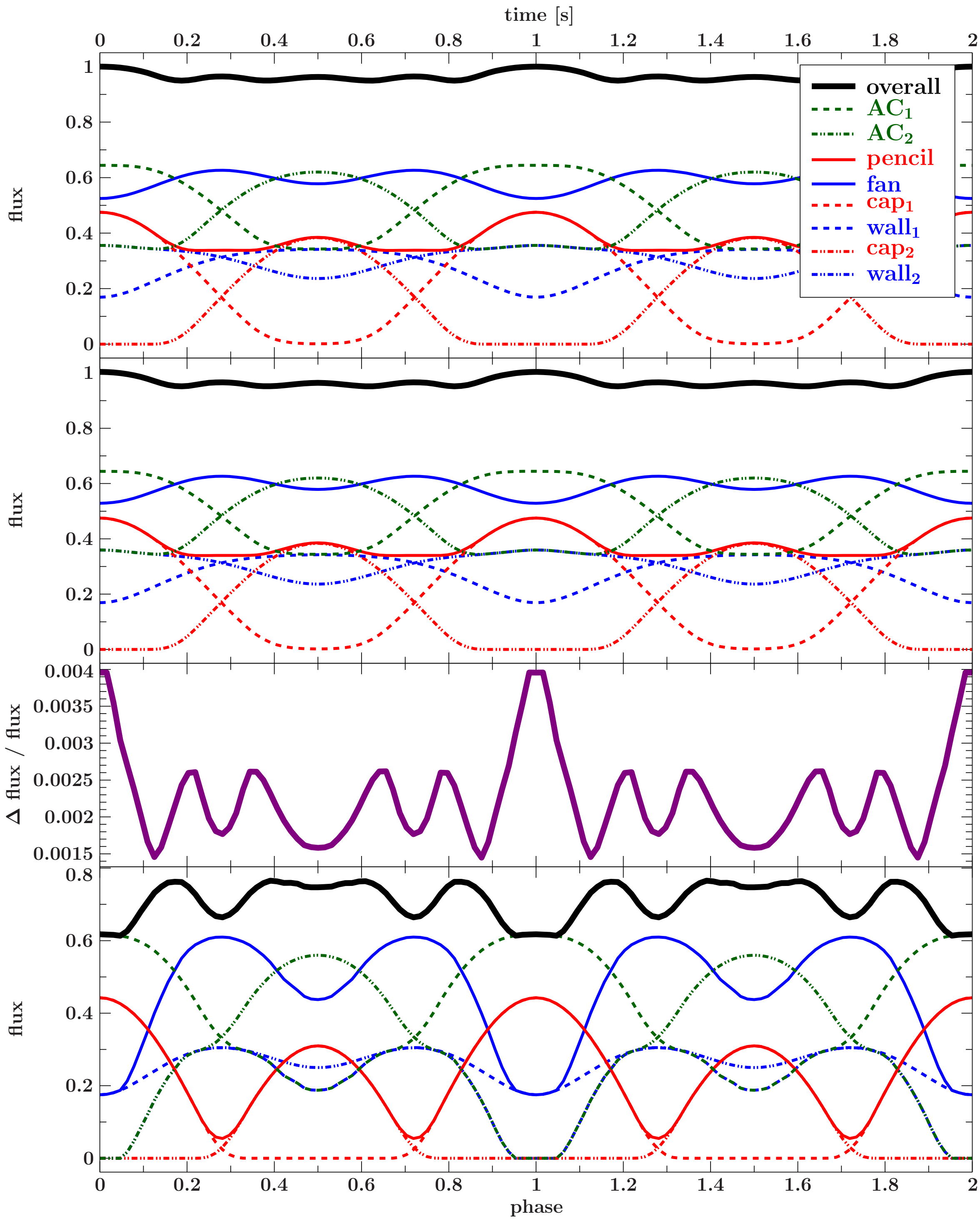


Fig. 4.1.: *Pulse profiles of a neutron star (R = 10 km, M = 1.35 $M_\odot$, i = 80°, f = 1 Hz) with isotropic and homogeneous emission ($I' \equiv 1$) from the antipodal accretion columns (R_{AC} = 3 km, h_{AC} = 3 km, i_{AC} = 45°) based on different projection methods. The top panel shows the pulse profile according to the exact relativistic projection (Eq. 2.11), the second panel the approximation (Eq. 2.38) and the bottom panel the geometrical projection. In the third panel the relative difference of the overall pulse profile of the second to the first panel is given.*

very back. In the geometrical case, the accretion column walls are only partly visible at these phases, i.e., a certain part of the emission is covered by the neutron star reducing the observed flux and causing the double peaked pulse profile. Overall, Fig. 4.1 shows the importance to account for the effect of light bending and the accuracy of the analytical approximation.

4.2. Variation of the Accretion Column Settings

Now we investigate the impact of the size and the location of the accretion columns on the neutron star onto the pulse profile. In Fig. 4.2, the inclination i of the line of sight with respect to the spin axis of the neutron star and the inclination i_{AC} of the accretion columns with respect to the spin axis is varied. For small i the variability of the individual contributions is small and increases with increasing inclination. Furthermore, the cap of the second accretion column gets more and more visible, increasing the contribution of the according accretion column, until the pulse profile of both accretion columns are identical for $i = 90°$. In the latter case the observer looks edge on to the neutron star with its identical and antipodal accretion columns leading to the high symmetry in the pulse profile. On the other hand, the increase of i_{AC} also increases the variability. The ratio of the mean flux of the two accretion columns, however, remains nearly constant. The variability increases due to the higher change of the location of the accretion column on the neutron star. For example, if the magnetic field is aligned to the rotation axis the system would show now changes over a rotation as the accretion column would just rotate around its symmetry axis. The ratios of the contributions remain similar as the accretion columns are identical and antipodal, and the inclination i is close to 90°. In all cases, the mean flux of the overall pulse profile barely changes, similar to the ratio of the fan and pencil beam.

A closer look at the pulse profile with $i = 70°$ (Fig. 4.2, left middle panel) reveals a narrow peak in the profile of the second accretion column wall at $\phi = 1$. At this phase this accretion column is located behind the neutron star. This narrow peak is due to the special configuration, where the inclination of the line of sight is close to the inclination of one of the accretion columns. In the case that both inclinations coincide these peaks are most pronounced. Figure 4.3 shows that for $i = i_{\mathrm{AC}}$ these peaks are always present. The explanation for these extraordinary peaks is that for such configurations the wall of the according accretion column is fully visible, seen as a ring[1] around the neutron star as shown in Fig. 4.4. This ring effect is only possible due to the light bending. While the accretion column is moving to the back its projection is enlarged in tangential direction. The closer the accretion column is to the very back the more of the wall gets visible and the greater the tangential enlargement, until a whole ring is visible. The increase of the visible area combined with the fact that the according surfaces are seen under small emission offset angles γ lead to the peak in the observed flux. Furthermore, the fraction of the phase, in which this ring effect occurs, is rather small, causing the peak to be narrow compared to the other peaks in the pulse profile. The closer the accretion column is to the spin axis, the longer the ring is visible resulting in a broadening of the peak. It is worth mentioning that this effect particularly applies for the accretion column wall. To see this effect for the cap of the accretion column or a hot spot their size has to be unreasonable large, covering over ~20% of the neutron star surface (see Eq. 3.5).

Further, Fig. 4.3 shows the case of unequal inclinations for the two accretion columns, namely the first one is fixed to $i_{\mathrm{AC1}} = 45°$, where the inclination i_{AC2} of the other accretion

[1]Similar to the Einstein ring (Einstein, 1936) observed in gravitational lens systems.

column is varied. Therefore, only the contribution of the the second accretion column changes. The shape of the fan beam is similar to the shapes we already saw in the previous plots, however, the profile of the pencil beam differs. The asymmetry in the inclinations of the accretion columns causes the slopes of the peaks to be unequal. In all previous plots, e.g., in the four bottom panels of Fig. 4.3 the pulse profile of the pencil beam is flat around the phases, where the pulse profiles of the individual caps intersect. This flatness is due to the equal slopes of the peaks. For unequal inclinations of the accretion columns, however, the slopes are different caused by their different rotational velocities. The difference in the slopes results in different shapes of the pencil beam peaks, especially, if the contributions of both caps are of the same order as for $i_{\mathrm{AC2}} = 60°$. Furthermore, this effect is most noticeable for the pencil beam as its emission is more focused in one direction than for the fan beam resulting in a steeper slope of the pencil beam peaks.

In Fig. 4.5, we now change the size of the accretion columns and their radius to height ratio and look at the resulting pulse profiles. First of all, the mean flux drops significantly with the reduction of the size as the emitting area is reduced. Furthermore, we again see that the shape of the individual pulse profiles remains similar. For example, the fan beam in every panel shows two peaks and their ratio remains similar. Furthermore, if one compares the two cases, in which $R_{\mathrm{AC}} = h_{\mathrm{AC}}$, the ratio of the fan and pencil beam show no noticeable differences. Therefore, the accretion column size mainly changes the scaling of the flux, at fixed height to radius ratios. On the contrary, the ratio of the accretion column height to its radius mainly influences the ratio of the contributions of the fan and pencil beam. The greater the radius the higher is the contribution of the pencil beam compared to the one of fan beam. It is also worth mentioning that the maximal fluxes of the two peaks of the fan beam occurring in most pulse profiles described so far are equal, excluding peaks caused by the ring effect. The ratio of the according dips changes. For the pencil beam this situation is vice versa, the minimal flux of the dips are always equal, where the ratio of the peaks varies. This symmetry is due to the fact that the accretion columns are either antipodal or of the same size or both.

4.3. Emission Profile $I' = I'(\gamma')$

Until now, we considered the emission from the accretion columns to be isotropic and homogeneous, i.e., $I' \equiv 1$. We now let the emission depend on the emission offset angle γ' in the corotating frame (see Sec. 2.3.3), where $I'(\gamma')$ is following a normal distribution

$$I'(\gamma') = \exp\left(-\frac{\gamma'^2}{2\sigma'^2}\right) \quad , \tag{4.1}$$

which is normalized such that $I'(0) = 1$. In other words, the emission is focused into the direction aligned with the surface elements normal vector $\mathbf{d}\boldsymbol{S}'$ ($\gamma' = 0$), whereas the intensity decreases for directions close to the surface plane ($\gamma' \to 90°$). Note that the intensity is not constant and therefore the flux per area is neither.

Figure 4.6 shows examples for various values for the distribution width σ'. Starting from the top left to the bottom right the width is reduced from very large to very small. This decrease results in a decreased mean flux, as the flux per area is not constant in this case. Moreover, there are significant changes in the shapes of the individual pulse profiles. The profile of the fan and pencil beam in the panel with largest distribution width looks similar to those shown in Fig. 4.5. The reduction of the width leads to an increase of the relative variability of the individual pulse profiles. The further the distribution width decreases the

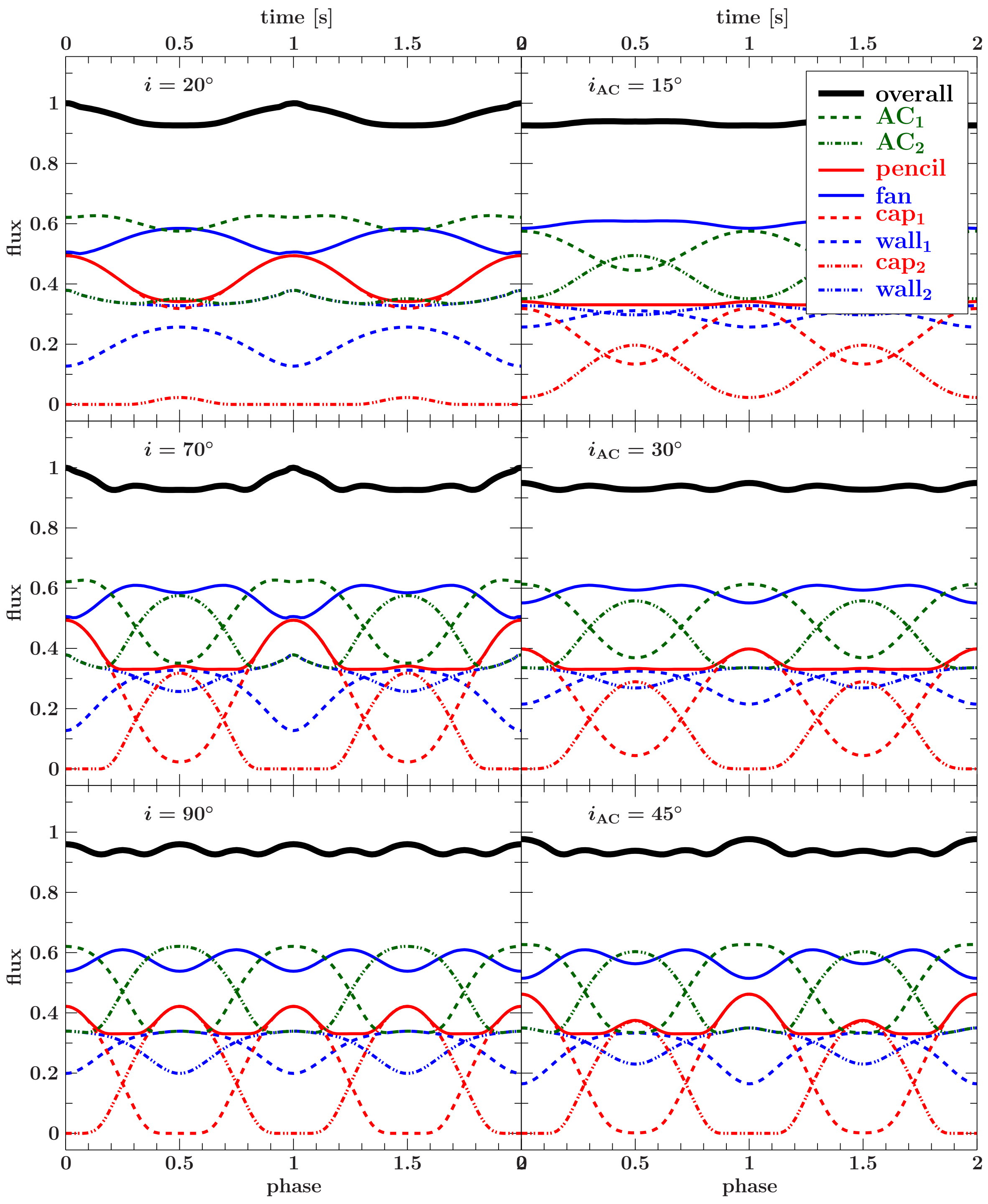


Fig. 4.2.: *Pulse profile of a neutron star (R = 10 km, M = 1.35 $M_\odot$, i = 80°, f = 1 Hz) under different inclinations i (left column) with isotropic and homogeneous emission ($I' \equiv 1$) from the antipodal accretion columns (R_{AC} = 3 km, h_{AC} = 3 km, i_{AC} = 45°), where i_{AC} is varied in the right column.*

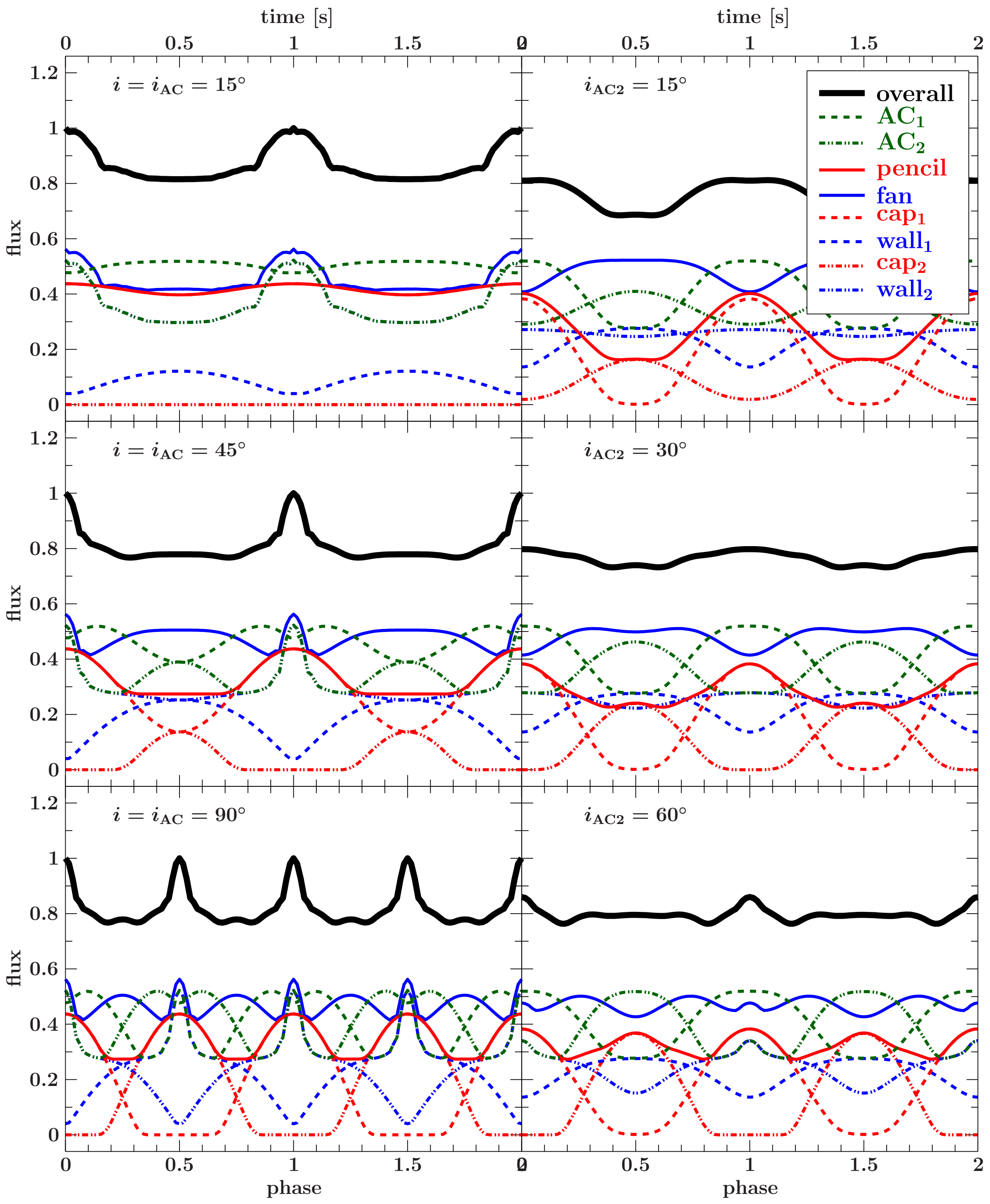


Fig. 4.3.: *Pulse profile of a neutron star (R = 10 km, M = 1.35 $M_{\odot}$, i = 80°, f = 1 Hz) with isotropic and homogeneous emission ($I' \equiv 1$) from the antipodal accretion columns (R_{AC} = 3 km, h_{AC} = 3 km). In the left column the inclinations i and i_{AC} are varied, where $i = i_{AC}$. In the right column the inclination of the first accretion column is fixed at i_{AC1} = 45° and i_{AC2} is varied. The fluxes are normalized to the maximum flux of the first panel.*

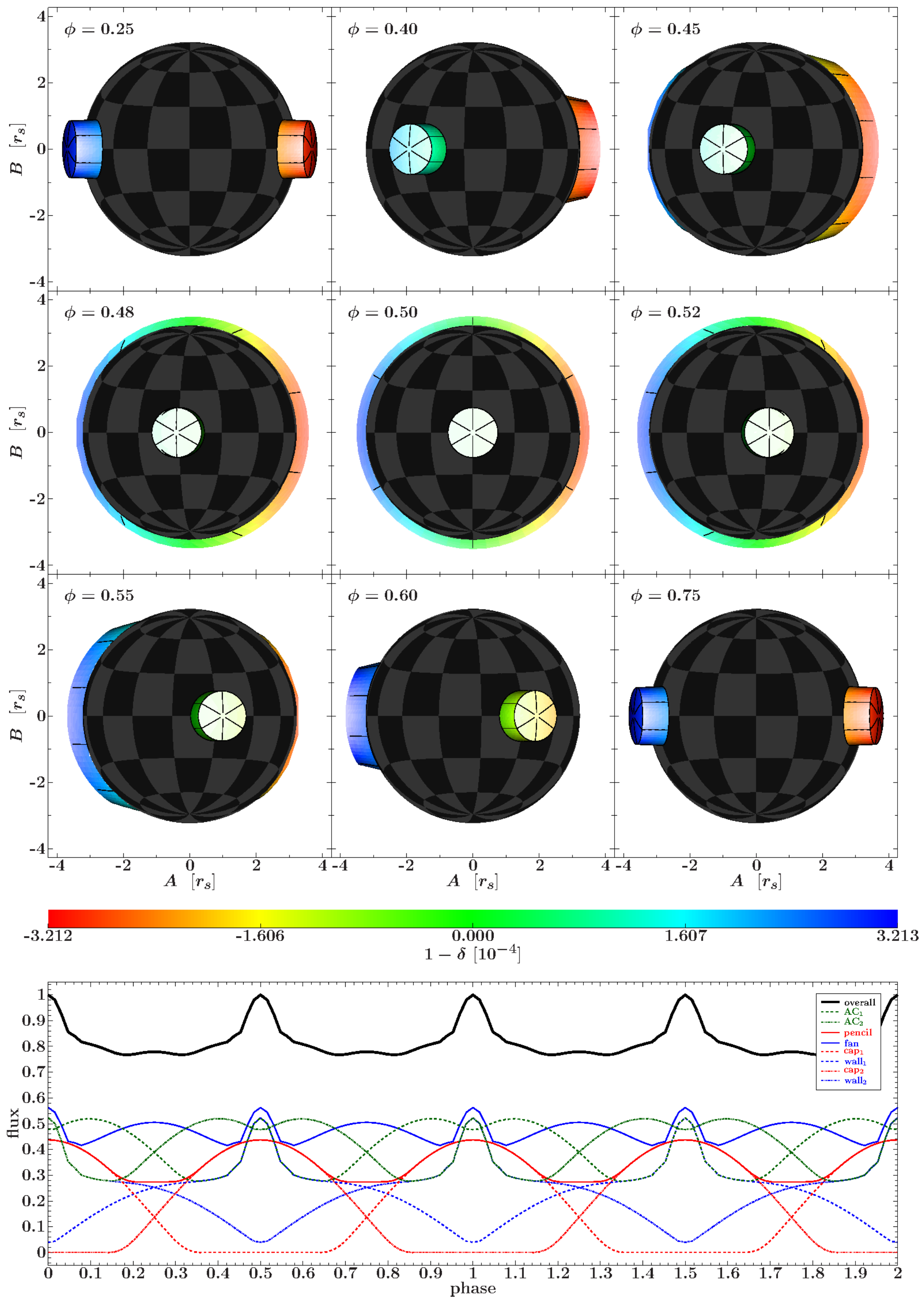


Fig. 4.4.: *Projection of a neutron star (R = 10 km, M = 1.35 $M_\odot$, i = 90°, f = 1 Hz) with isotropic and homogeneous emission ($I' \equiv 1$) from the antipodal accretion columns (R_{AC} = 3 km, h_{AC} = 3 km, i_{AC} = 90°) visualizing the ring effect.*

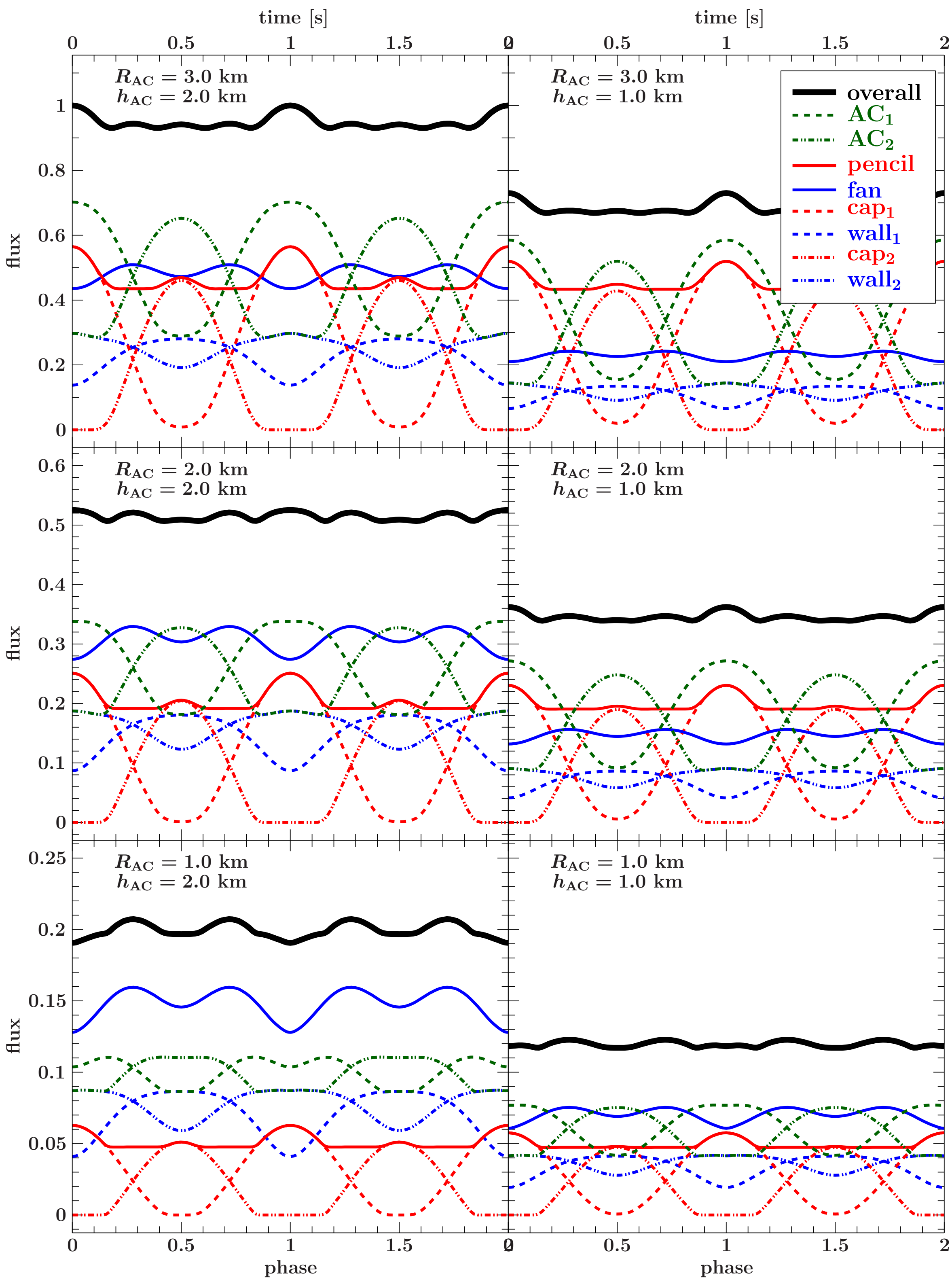


Fig. 4.5.: *Pulse profile of a neutron star (R = 10 km, M = 1.35 $M_\odot$, f = 1 Hz) with isotropic and homogeneous emission ($I' \equiv 1$) from the antipodal (i_{AC} = 45°) accretion columns with different radii R_{AC} and heights h_{AC}.*

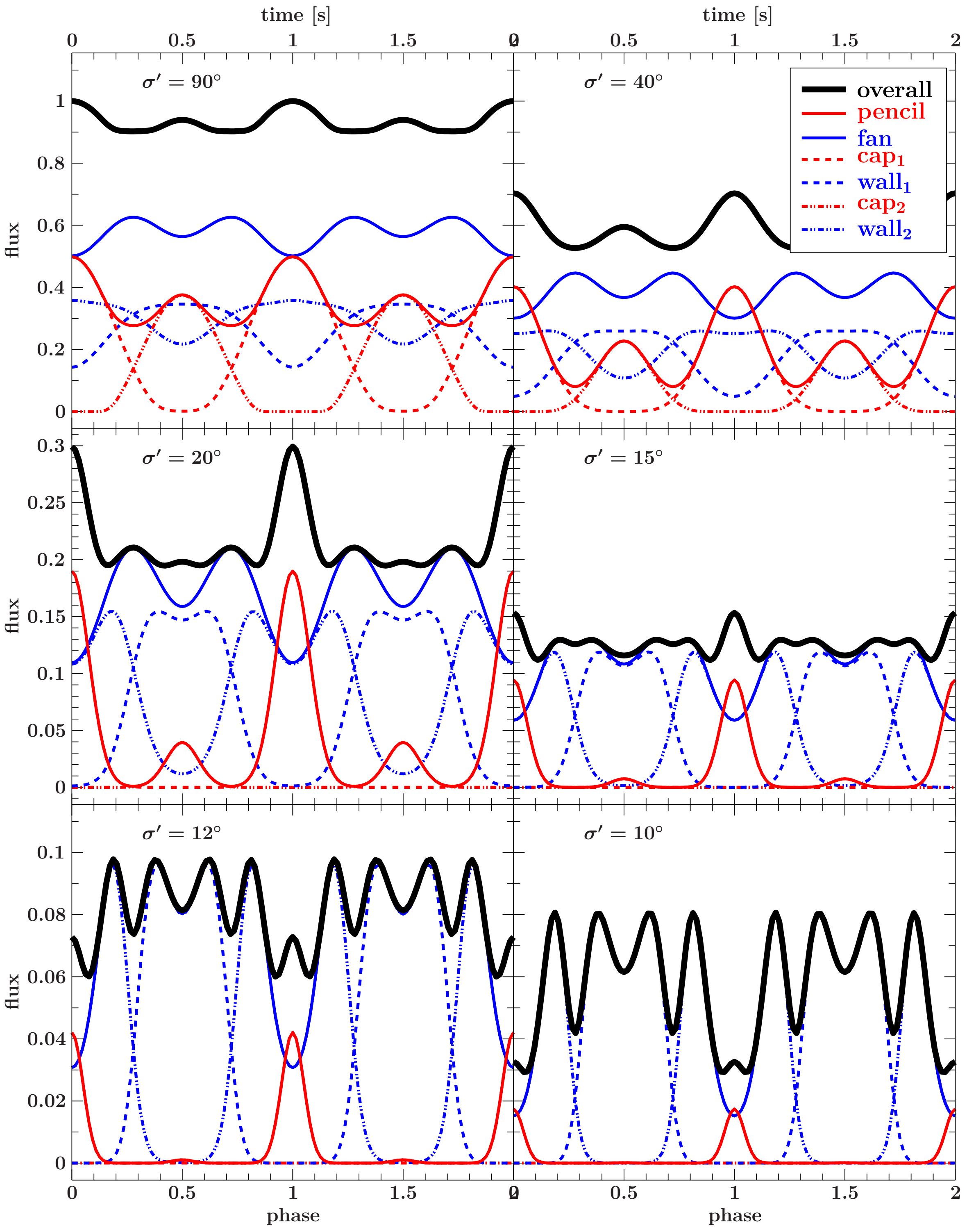


Fig. 4.6.: *Pulse profile of a neutron star (R = 10 km, M = 1.35 $M_\odot$, i = 80°, f = 1 Hz) with emission from the antipodal accretion columns (R_{AC} = 3 km, h_{AC} = 3 km, i_{AC} = 45°), where the emission profile $I' = I'(\gamma')$ depends on the emission offset angle γ' and is following a normal distribution (Eq. 4.1).*

less is the contribution of the second cap. On the contrary, the relative contribution of the peak in the pencil beam at phase $\phi = 1$, which is related to the first cap, first increases for $\sigma' = 90°$ to $\sigma' = 20°$ and then decreases again. Furthermore, the two peaks of the fan beam each split up into two new peaks for $\sigma' \leq 20°$, leading to a four peaked fan beam. At a closer look this splitting already occurs for $\sigma' \leq 40°$ for the profiles of the individual accretion column walls, but only becomes noticeable in the fan beam for lower distribution widths. Overall the number of peaks increases from two over four to five. The relative increase of the contribution of the fan beam compared to the pencil beam is easily explained by the fact that the emission of the pencil beam is focused in one direction, whereas the emission of the fan beam is focused into a plane. The focusing is given by the distribution of the normal vectors of the according surfaces. Therefore, the wall of the accretion column provides a much higher range of emission angles at any given time compared to the cap. Or in other words, the window, in which the emission of the cap can be observed is much smaller. On the other hand, if the cap is in that window the focused pencil beam leads to steep peaks.

It is noticeable that all of the discussed pulse profiles show general symmetries. First of all, the pulse profiles are symmetric to $\phi = 0.5$, i.e., flipping the pulse profile at this phase results in the same profile. This symmetry is simply due to the fact that the location of the two accretion columns on the neutron star are separated in phase by $\phi = 0.5$. And moreover, the shape of each peak is symmetric to its maximum. One way to obtain asymmetric shaped peaks in the composed pulse profiles is a phase separation of the individual accretion columns unequal to half a phase. In such a case, the sum of symmetric peaks can lead to an asymmetric composition. The symmetry of the peaks in the pulse profiles of the individual parts of the object, however, is independent of the phase separation of the accretion column. Meaning that the an asymmetric phase separation only explains the asymmetry of peaks composed of at least two individual peaks. There are two further possibilities to introduce asymmetric shaped peaks also for peaks which are not a composition. On the one hand, one can choose an asymmetric emission profile. On the other hand the spin frequency of the neutron star can be increased to values above 1 Hz (Eq. 3.6), where the influence of special relativistic effects, especially the time delay, has to be considered.

Amongst other effects (e.g., oblateness) the increase of the spin frequency increases the impact of the phase shift (Eq. 2.74) caused by the time delay of the photons (Sec. 2.2.2). For example, the maximal possible phase shift given by Eq. 3.7 is $\sim 13\%$ for an intrinsic spin frequency $f' = 1000$ Hz. As mentioned in Sec. 3.7, the program is not yet accounting for this effect[2]. Nevertheless, it is possible to make some general statements. Additionally, to the increase with the spin frequency, the phase shift increases with increasing apparent emission angle Ψ. Considering an emitting spot, whose apparent emission angle Ψ periodically decreases and increases while the spot is moving toward and away from the observer due to the rotation (Eq. 2.58) results in an increase and decrease of the phase shift. Furthermore, the time a spot needs to rotate from the back to the front and vice versa decreases with increasing spin frequency. Therefore, the observed flux of an emitting spot gets compressed in time, when it is moving towards the observer and stretched in time, when it is moving away. Hence, the phase shift has an impact on the shape of the pulse profile and degrades the symmetry of the peaks. For example, the slope of a pencil beam peak on the ascending side increases and decreases on the descending side.

[2]The focus in this thesis is on HMXBs and LMXBs, in which the neutron star exhibits spin frequencies less than 1 Hz and therefore, the effect of the time delay is not important. The program, however, was designed to be easily extendable for this effect.

Additionally, the Doppler factor (Eq. 2.50) depends on the spin frequency. Moreover, the Doppler factor of a spot over a rotation period is asymmetric, i.e., the maximal and minimal Doppler factor for a given spot on the object over a complete rotation are not symmetric around $\delta = 1$, e.g., as seen in Fig. 4.4.

The phase shift and the Doppler shift are special relativistic effects, which are important for high velocities or respectively high spin frequencies. Hence, we can conclude that the general relativistic effect of the light bending does not introduce any asymmetry to the pulse profiles.

Chapter 5

Conclusions & Future Prospects

5.1. Summary & Conclusions

In summary, the developed program presented in this thesis is suitable to determine the time and energy dependent flux of a neutron star with accretion columns or hot spots. The calculation of the flux is based on an arbitrary given emission profile and accounts for general relativistic effects, namely gravitational redshift and moreover light bending, and special relativistic effects, namely the Doppler shift and aberration.

The program is subdivided in `modules` to separate the individual calculation steps and to make it easier to implement extensions. With the `object` module functions are provided to build a spherical neutron star with cylindrical accretion columns, with either a flat or spherical cap, or circular hot spots. However, it is possible to define arbitrary geometries as long as the object satisfies the code structure described in Sec. 3.2.

The `mapping` module, which is independent of the `object`, creates a lookup table for the special and general relativistic parameters of the photon trajectory. The equations of the general relativistic effects are based on the Schwarzschild metric. The calculations to obtain the lookup table uses an analytical approximation (Beloborodov, 2002), which restricts $R > 2r_\mathrm{s}$, or alternatively numerical methods to solve the exact photon trajectory. Although the exact solution for the photon trajectory is not subject to the restriction of $R > 2r_\mathrm{s}$, the implemented numerical procedure solving the integral (Eq. 2.11) is optimized for the same range of R (Sec. 3.3.2), which is reasonable for neutron stars. Therefore, the program in general requests that $R > 2r_\mathrm{s}$, which excludes only a small fraction of possible mass to radius ratios of neutron stars (see Fig. 1.2). Moreover, if needed it is possible to extend this module with numerical methods to also solve the exact solution for $r_\mathrm{s} < R \leq 2r_\mathrm{s}$.

Further, I will implement an alternative interpolation method, as the bilinear interpolation used in the `interpolation` module requires a particular point distribution of the lookup table. The bilinear interpolation is sufficient in most cases, however, for large deflection angles of the photon trajectories close to the maximal possible value it looses accuracy (see Fig. 3.8). An interpolation method, which does not rely on a particular point distribution could increase the accuracy. Nevertheless, the advantage of the bilinear interpolation is its speed.

In Chap. 4 the pulse profile of neutron stars with emission from the accretion columns for various configurations were looked at. Thereby, the focus was on the effect of light bending to the pulse profile and the contribution of the individual parts of the accretion columns. For slowly rotating neutron stars with rotation periods over 1 second, e.g. HMXBs, especially the effect of light bending is very important as the influence of the special relativistic effects is insignificant for these rotation periods. The investigated pulse profiles show that the light bending has a considerable influence on their shape, but it does not introduce any asymmetry. Namely, asymmetries in pulse profiles of slowly rotating neutron stars can only be explained by asymmetric emission profiles or asymmetric accretion column positions on the surface of the neutron star, i.e., phase separations unequal to half a phase.

5.2. Future Prospects

The `flux` module provides the possibility to choose between isotropic and homogeneous emission or emission dependent on the emission offset angle γ following an uniform or normal distribution. However, the module is designed to include arbitrary emission profiles, which may depend not only on the emission angle, but also on the location on the object, the photon energy and the time, or other parameters. Therefore, it will be possible to include realistic emission profiles, e.g., based on theoretical models of the radiative processes within the accretion columns (see, e.g., Araya & Harding, 1999; Becker & Wolff, 2007; Schönherr et al., 2007). These models describe the theoretical structure of the accretion columns and their emission profiles. By combining and including them into this program it will be possible to provide self-consistent models. The great advantage of the presented program is that it in general does not introduce any constraints for the objects[1].

There are other works on this subject, for example, the decomposition method developed by Kraus et al. (1995), which was successfully used in several analysis of pulse profiles of HMXBs (e.g., Blum & Kraus, 2000; Sasaki et al., 2010; Caballero et al., 2011; Sasaki et al., 2012). The approach of Kraus et al. (1995) to decompose an observed pulse profile into single-pole pulse profiles, however, relies on several assumptions and prerequisites. For example, the assumption that an asymmetric pulse profile is the sum of two symmetric contributions or that the emission regions are equal with axisymmetric beam patterns. For a detailed discussion of this decomposition method read Kraus et al. (1995). In contrast, the geometrical structure of the emission region and moreover the according emission profile can be defined arbitrarily in the program developed in this thesis. Furthermore, it is also possible to define an individual and arbitrary emission profile for each of these regions.

Further, the program was designed in a way that it is possible to extend it also for higher spin frequencies than 1 Hz. This extension only requires to additionally account for the special relativistic effect of the time delay causing a phase shift of the observed photons. The maximal phase shift caused by the time delay is about 0.013% (Eq. 3.6, Eq. 3.7) for a neutron star with an intrinsic spin frequency of 1 Hz. Further, the phase shift is proportional to the frequency (Eq. 2.74) and therefore has a considerable influence for higher frequencies. The increase of the spin frequency also increases the influence of the Doppler factor, as it is dependent on it. Furthermore, high spin frequencies cause the oblateness of the neutron star, i.e., it can no longer assumed to be spherical (see, e.g., Cadeau et al., 2007). All these effects connected to high spin frequencies are the subject of further extensions for this program, which allow to investigate particularly the impact of the special relativistic effects additionally to the relativistic light bending. This extension will allow to study the observed flux and especially the pulse profiles of fast rotating neutron stars, like, e.g., millisecond pulsars, which exhibit spin frequencies up to 700 Hz (see, e.g., Hessels et al., 2006).

[1]Objects of interest are especially neutron stars exhibiting rotations periods $\gtrsim 1$ s, like, e.g., HMXBs, which show the strongest magnetic fields and exhibit a sufficient accretion rate to allow the formation of accretion columns. Furthermore, most neutron stars have radii greater than two Schwarzschild radii.

Appendix A

Example code

In the following an example code is shown. It is very basic only using the necessary functions to determine the bolometric flux of neutron star with two antipodal accretion columns, and to plot the projection for every phase and the according pulse profile for chosen parts.

```
% Including the light bending library
require("lbscripts");

% Initialization of the parameter structure
variable par = lb_par_init(; path = getcwd, ID = "example",
                            lbmeth = "belob",
                            onns = 0, onac = 1,
                            nphi = 64,
                            u = 0.4, R = 10e3, f = 1,
                            i = 80.*PI/180.,
                            Rac = 3e3, hac = 3e3, iac = 45.*PI/180.,
                            nns = 43, nac = 55,
                            nma = 100, nmr = 100 );

% Build the object: Neutron star with accretion columns
variable obj = lb_obj_build_ns ( par );

% Create lookup table for the light bending parameter
variable map = lb_psir2b_map ( par );

% Perform the projection procedure
variable bend, proj;
(bend,proj) = lb_projectobj ( par, obj, map );

% Initialization of the parameter structure for the flux
variable fpar = lb_fluxpar_init( par ;
                                  FID = "fluxmodel",
                                  flux = "bolo" );

% Calculation of the surface element flux. I(gamma) follows normal
% distribution with width sig
```

```
variable seflux = lb_flux_se( par, fpar, obj, bend ;
                              dis = "normal", sig = 45.*PI/180. );

% Caculation of the time (& energy) dependent flux
variable flux = lb_flux( par, fpar, obj, seflux, bend );

% Plotting the projection for every phase
variable xf;
xf = lb_xfig_plot_projection( par, fpar, obj, proj, bend, seflux ;
                              grid );

% Determine the parts, for which the pulse profile will be plotted
variable ufield = ["all","ac1cap","ac1col","ac2cap","ac2col"];

% Plotting the actual pulse profile
variable pp = lb_xfig_plot_pulseprofile( par, fpar, flux ;
                                         usefields = ufield );
```

Appendix B

Storing

The routine provides functions to save and load the results of each module described in Chap. The description of this function can be found in App. C. The files are in the Fits format (see Hanisch et al., 2001). The main `<path>` to the directory, in which the files are being stored is defined with the `par.path` variable in the parameter structure. The parameter structure (`par`) itself is stored in a text file, which is readable by Isis. For each object with an individual identification `par.ID` a separate directory is created automatically. This folder contains the object (`obj`), the table (`map`) and the projection parameter (`bend`, `proj`). Additionally, the parameter (`fpar`) for each flux model (`fpar.FID`) a stored in a individual file, where the according flux of the individual surface elements (`seflux`) and the processed flux (`flux`) in an extra subfolder. If not otherwise specified the plots created with the plotting functions are also stored in this subfolder. The following diagram shows the file structure:

```
[<path>]
  – <ID>_par.sl
  – [<ID>]
     ↳ <ID>_obj.fits
     ↳ <ID>_map.fits
     ↳ <ID>_bend.fits
     ↳ <ID>_proj.fits
     ↳ <ID>_<FID>_fpar.fits
     ↳ [<ID>_<FID>]
        ∟ <ID>_<FID>_seflux.fits
        ∟ <ID>_<FID>_flux.fits
```

Furthermore each Fits-file contains the parameter `par`. When loading this file, the stored parameters are compared to the original parameters in the main directory to ensure that there were no changes.

Appendix C

Function declarations & usage

The code is written in Isis[1] (Interactive Spectral Interpretation System), a script language using the S-Lang[2] interpreter.
In the following a list of all functions within the light bending routine is given. Besides a description for each function the expected arguments and their data type, possible qualifiers with their standard value and data type, the data type of the return value and potential dependencies are given, if they exists.

`function_name( arguments )`

Argument	`arg1 = data type` *optional note*
Qualifier	`qual1` [standard value] `(dt)` *optional note*
Return	`data type`
Dependencies	`function_name`
Description	First the function name and its expected arguments are given. Then the single arguments are listed and which data type they have to have. The squared bracket following the data type give the expected dimension(s): No brackets relates to none array, empty brackets to array of arbitrary length, entries separated by backslashes to possible array lengths and entries separated by comma to a matrix (e.g. `[n,m]` $n \times m$ matrix). For the qualifiers the data type is given in a shortened form within round brackets, where `I` relates to `Integer_Type`, `D` to `Double_Type`, `S` to `String_Type`, `V` to `Vector_Type` and `R` to `Ref_Type`. If (–) is given instead of a real data type, the qualifier just have to exist to enable the functionality given in its description. In the dependencies row the function names of those function, which get called within it, are given. If one of these functions is not defined within the light bending routine, the containing library will be given in brackets. The dependencies are supposed to show, to which functions qualifiers get passed through. That means that these qualifier can be given the superior function, even if it does not appear in its description.

[1]See Houck & Denicola (2000) or `http://space.mit.edu/cxc/isis/`
[2]See `http://www.jedsoft.org/slang/`

parameter

`lb_grid_phase( par )`

Argument	`par = Par_Type`	*See* `lb_par_init`		
Qualifier	`nphi`	[par.nphi]	`(Integer_Type)`	*Number of phase grid points*
	`dphi`	[par.dphi]	`(Double_Type)`	*Distance between two adjacent phase grid points*
Return	`Double_Type[nphi]`			
Description	Returns a phase grid (low bins) with `nphi` equidistant points or a distance of `dphi` within $[0, 2\pi[$.			

`lb_partson( par )`

Argument	`par = Par_Type` *See* `lb_par_init`
Return	`Integer_Type[]`
Description	Gives an integer array of indices corresponding to the components of the object, which are set to be included in flux calculation (Needed for plotting). For technical reasons each component requires 2 indices: Neutron star (NS) [1,2]; cap of first accretion column (AC1cap) [3,4]; wall of first accretion column (AC1col) [5,6]; AC2cap [7,8]; AC2col [9,10].

`lb_part_index( n )`

Argument	`n = Integer_Type`
Return	`Integer_Type[]`
Description	Returns an integer or integer array of unique indices corresponding to the components of the object.

`lb_part_name( index )`

Argument	`index = Integer_Type`
Return	`String_Type[]`
Dependencies	`lb_part_index`
Description	Returns the name(s) of a component of the object according to the given index (see `lb_part_index`).

`lb_par_init( )`

Qualifier

i	$[80\frac{\pi}{180}]$	`(D)`	*Inclination to NS spin axis [radian]*
f	[1]	`(D)`	*NS spin frequency [Hz]*
u	[0.4]	`(D)`	r_s/r
R	$[12\cdot10^3]$	`(D)`	*NS radius [m]*
M	$[\frac{uRc^2}{(2G)}]$	`(D)`	*NS Mass [kg]*
D	[1]	`(D)`	*Distance to Observer*
Rac	$[1\cdot10^3]$	`(D[1/2])`	*AC1/2 radius [m]*
hac	$[1\cdot10^3]$	`(D[1/2])`	*AC1/2 height [m]*
iac	[0]	`(D[1/2])`	*AC1/2 inclination to NS spin axis*
rmin	[2]	`(D)`	*Minimal radius [r_s]*
rmax	[10]	`(D)`	*Maximal radius [r_s]*
nns	[65]	`(I)`	*Degree of NS grid fineness*
nac	[65]	`(I)`	*Degree of AC grid fineness*
nmr	[200]	`(I)`	*# radii in $r\alpha$-map*
nma	[800]	`(I)`	*# α in $r\alpha$-map*
onns	[0]	`(I)`	*Emission from NS [0/1]*
onac	[1]	`(I)`	*Emission from AC [0/1]*
oncap	[1]	`(I)`	*Emission from AC cap [0/1]*
oncol	[1]	`(I)`	*Emission from AC col [0/1]*
ng	[6]	`(I)`	*# plotted grid lines / pattern*
gridns	[onns]	`(I)`	*Add grid lines of NS [0/1]*
gridac	[1]	`(I)`	*Add grid lines of AC [0/1]*
nphi	[1]	`(I)`	*Number of phase grid points*
dphi	[]	`(D)`	*Distance between two adjacent phase grid points*
lbmeth	['belob']	`(S)`	*Projection method ['exact', 'belob', 'geome']*
magnilim	[8]	`(I)`	*Magnification limit in* `oversize`
ID	['unnamed']	`(S)`	*Object identification*
path	[]	`(S)`	*Main saving path*

Return `Par_Type := Struct_Type{ID, path, i, f, u, R, M, rs, D, Rac1, hac1, iac1, Rac2, hac2, iac2, rmin, rmax, nns, nac, nmr, nma, onns, oncap, oncol, ng, gridns, gridac, nphi, dphi, lbmeth, magnilim, c, G}`

Description Initialize the parameter structure containing all important values for the routine. Besides the listed qualifiers, the returned structure also contains the speed of light `c`, the gravitational constant `G` and the Schwarzschild radius `rs`. Most values are expected to be one-dimensional, but the three (`Rac`, `hac`, `iac`) defining the accretion column. If one or more specifications of the individual accretion columns are not equal a two-dimensional array for the according value can be given.

build_object

`lb_obj_init( )`

Qualifier	N	[0]	(I)	*Length of* r, ds *&* pg
	Nv	[0]	(I)	*Length of* v
	sph	[]	(-)	*Vectors in spherical coordinates*

Return `Obj_Type := Struct_Type{v=type[Nv], pg=Integer_Type[N,3], r=type[N], ds=type[N], no=Integer_Type[N], grid=Array_Type[0]}`

Description Initializes an object and returns a `Struct_Type` containing the individual position vectors v of the vertices of the surface elements, where pg stores the according indices; the barycentric vector r and the surface vector ds of the surface elements; a list of integers on, where the entries define the part of the neutron star the according surface element belongs to and a grid array containing lists of indices specifying grid lines for the `plotting` module. If the sph qualifier is given `type=Sph_Type` (`typedef struct{ r, phi, theta } Sph_Type`), otherwise `type=Vector_Type`

`lb_cart2sph( v )`

Argument `v = Vector_Type[]`
Return `Sph_Type[]`
Description Transforms a given vector v from Cartesian into spherical coordinates.

`lb_sph2cart( v )`

Argument `v = Sph_Type[]` *See* `lb_obj_init`
Return `Vector_Type[]`
Description Transforms a given vector v from spherical into Cartesian coordinates.

`lb_obj_cart2sph( obj )`

Argument `obj = Obj_Type` *See* `lb_obj_init`
Return `Obj_Type`
Dependencies `lb_obj_init, lb_cart2sph`
Description Transforms the according entries of the object obj from Cartesian into spherical coordinates.

`lb_obj_sph2cart( obj )`

Argument `obj = Obj_Type` *See* `lb_obj_init`
Return `Obj_type`
Dependencies `lb_obj_init, lb_sph2cart`
Description Transforms the according entries of the object obj from spherical into Cartesian coordinates.

`lb_obj_rotate( obj, angle )`

Argument

obj	=	`Obj_Type`	*See* `lb_obj_init`
angle	=	`Dobule_Type`	*Rotation angle in radian*

Qualifier

n	[(0,0,1)]	`(V)`	*Rotation axis*

Return `Obj_Type`

Dependencies `lb_obj_init`

Description Rotates a given object obj around the axis n with the given angle, where the vectors in obj have to be in Cartesian coordinates (`Vector_Type`).

`lb_obj_dphi( obj, angle )`

Argument

obj	=	`Obj_Type`	*See* `lb_obj_init`
angle	=	`Dobule_Type`	*Rotation angle in radian*

Return `Obj_Type`

Description Rotates a given object obj around the z-axis by adding the given angle to the phi components, where the vectors in obj have to be in spherical coordinates (`Sph_Type`).

`lb_obj_build_column( par, R, h )`

Argument

par	=	`Par_Type`	*See* `lb_par_init`
R	=	`Dobule_Type`	*Radius of Column*
h	=	`Dobule_Type`	*Height of Column*

Qualifier

oncap	[par.oncap]	`(I)`	*Emission from AC cap [0/1]*
oncol	[par.oncol]	`(I)`	*Emission from AC col [0/1]*
ng	[par.ng]	`(I)`	*# plotted grid lines / pattern*
grid	[par.gridac]	`(I)`	*Add grid lines of AC [0/1]*
n	[par.nac]	`(I)`	*Degree of AC grid fineness*
Rns	[par.R]	`(D)`	*NS radius [m]*
eps	[$89\frac{\pi}{180}$]	`(D)`	*Distinguish between AC cap & col*
onlycap	[]	`(-)`	*Only column cap*
flat	[]	`(-)`	*Column cap is flat*

Return `Obj_Type`

Description	In this function a cylindrical accretion column with radius R and height h is build. If the qualifier flat is given the cap is flat, otherwise it is is a spherical segment of a sphere with radius $\sqrt{\mathsf{Rns}^2 - \mathsf{R}^2} + \mathsf{h}$. Furthermore, the column gets an offset to the origin according to the Rns qualifier, meaning that the column is adjusted, so that it would sit on the surface of a sphere with radius Rns. If the qualifier onlycap exists or h = 0,the column only consists of the column cap (e.g. to define a hot spot). The amount of surface elements depends on n. oncap and oncol control whether there is emission from the column cap/wall or not. In case of no emission the according entries in obj.on are negative, where the absolute value corresponds to part of the object (See `lb_partson`). If gridac = 1 grid lines, used in the `plotting` module, are determined, where the amount depends on ng. The grid lines are stored in obj.grid, an array of lists with indices corresponding to the according entries in obj.v. eps is just a variable used to distinguish between column cap and wall to determine the according entries of obj.on.

`lb_obj_build_sphere(` par, R `)`

Argument	par = `Par_Type`			*See* `lb_par_init`
	R = `Dobule_Type`			*Radius of sphere*
Qualifier	on	[par.onns]	`(I)`	*Emission from NS surface [0/1]*
	ng	[par.ng]	`(I)`	*# plotted grid lines / pattern*
	grid	[par.gridns]	`(I)`	*Add grid lines of NS [0/1]*
	n	[par.nac]	`(I)`	*Degree of NS grid fineness*
Return	`obj`			
Description	In this function a sphere/neutron star with radius R is build. The amount of surface elements depends on n. on controls whether there is emission from the surface of the sphere or not. In case of no emission the according entries in obj.on are negative, where the absolute value corresponds to part of the object (See `lb_partson`). If gridns = 1 grid lines, used in the `plotting` module, are determined, where the amount depends on ng. The grid lines are stored in obj.grid, an array of lists with indices corresponding to the according entries in obj.v.			

`lb_obj_merge(` objs `)`

Argument	objs = `Obj_Type[]`	*Array of* `obj`*'s (See* `lb_par_init`*)*
Return	`Obj_Type`	
Description	This function merges several obj's into one obj and adjusts the entries of those struct fields, that contain indices.	

`lb_obj_rrange(` par, obj `)`

Argument	par = `Par_Type`			*See* `lb_par_init`
	obj = `Obj_Type`			*See* `lb_obj_init`
Qualifier	digit	[3]	`(I)`	*# significant digits*

Description	Determines the maximal and minimal occurring radius within the given obj, round these to the number of significant digits given by digit and stores them in par.rmin/par.rmax.

`lb_obj_build_ns( par )`

Argument	`par = Par_Type` *See* `lb_par_init`			
Qualifier	Rac	[[par.Rac1,par.Rac2]]	`(D[2])`	*Radii of AC1 & AC2 [m]*
	hac	[[par.hac1,par.hac2]]	`(D[2])`	*Heights of AC1 & AC2 [m]*
	iac	[[par.iac1,par.iac2]]	`(D[2])`	*Inclination of AC1 & AC2 to NS spin axis*
Return	`Obj_Type`			
Dependencies	`lb_obj_build_sphere, lb_obj_build_column, lb_obj_rotate, lb_obj_merge, lb_obj_rrange`			
Description	Combines the functionality of `lb_obj_build_sphere` and `lb_obj_build_column` and creates a neutron star with accretion columns according to the given parameters. `lb_obj_rrange` is also called in this function.			

mapping

`lb_b_ua( u, alpha )`

Argument	`u = Double_Type` *Inverse emission radius* $u = r_\mathrm{s}/R$
	`alpha = Double_Type` *Emission angle* α
Qualifier	lbmeth [] `(S)` *If 'geome', geometrical projection*
Return	`Double_Type`
Description	Calculates the impact parameter b according Eq. 2.23. If qualifier lbmeth='geome' the geometrical projected counterpart $(\sin(\alpha)/u)$ is returned.

`lb_a_ub( u, b )`

Argument	`u = Double_Type` *Inverse emission radius* $u = r_\mathrm{s}/R$
	`b = Double_Type` *Impact parameter* b
Qualifier	lbmeth [] `(S)` *If 'geome', geometrical projection*
Return	`Double_Type`
Description	Calculates the emission angle $\alpha \leq \pi/2$ according Eq. 2.23. If qualifier lbmeth='geome' the geometrical projected counterpart $(\arcsin(bu))$ is returned.

`lb_p_b( b )`

Argument	`b = Double_Type` *Impact parameter* b
Qualifier	lbmeth [] `(S)` *If 'geome', geometrical projection*
Return	`Double_Type`

Description Calculates the periastron $r_\mathrm{p}$according Eq. 2.20. If qualifier lbmeth='geome' the geometrical projected counterpart (b) is returned.

`lb_alphamax( u )`

Argument	`u = Double_Type[]`	*Inverse emission radius $u = r_\mathrm{s}/R$*		
Qualifier	rcrit	[2]	`(D)`	*Critical radius r_c (Eq. 2.19)*
	bcrit	[lb_b_ua(1/rcrit,π/2)]	`(D)`	*Critical impact parameter b_c (Eq. 2.19)*
Return	`Double_Type`			
Dependencies	`lb_b_ua`			
Description	Calculates the maximal emission angle α_max (Eq. 2.25) based on the specified critical impact parameter.			

`lb_psimax( u )`

Argument	`u = Double_Type[]` *Inverse emission radius $u = r_\mathrm{s}/R$*
Return	`Double_Type[]`
Dependencies	`lb_psi_ua, lb_alphamax`
Description	Calculates the maximal angle Ψ_max (Eq. 2.26) based on the specified critical impact parameter.

`lb_psiper( up )`

Argument	`up = Double_Type[]` *Inverse periastron $u = r_\mathrm{s}/r_\mathrm{p}$*
Qualifier	lbmeth ['belob'] `(S)` *Projection method ['exact', 'belob', 'geome']*
Return	`Double_Type[]`
Dependencies	`lb_psi_ua_exact`
Description	Calculates the apparent emission angle at the periastron Ψ_p. lbmeth='exact': Eq. 2.11; lbmeth='belob': Eq. 2.38; lbmeth='geome': $\Psi_\mathrm{p} \equiv \pi/2$.

`lb_psi_ua_belob_( u, alpha )`

Argument	`u = Double_Type`	*Inverse emission radius $u = r_\mathrm{s}/R$*
	`alpha = Double_Type`	*Emission angle α*
Return	`Double_Type`	
Description	Calculates the angle Ψ according to Eq. 2.38 for $\alpha \leq \pi/2$.	

`lb_psi_ua_belob( u, alpha )`

Argument	`u = Double_Type`	*Inverse emission radius $u = r_\mathrm{s}/R$*
	`alpha = Double_Type`	*Emission angle α*
Qualifier	b [lb_b_ua(u, alpha)] `(D)`	*Impact parameter b*
Return	`Double_Type`	
Dependencies	`lb_b_ua, lb_p_b, lb_psi_ua_belob_`	

Description	Calculates the angle Ψ according to Eq. 2.38 and Eq. 2.21 for $\alpha \leq \alpha_{\max}$.

`lb_psi_ua_exact_( x )`

Argument	`x = Double_Type`			
Qualifier	u	[]	`(D)`	*Inverse emission radius $u = r_s/R$*
	alpha	[]	`(D)`	*Emission angle α*
Return	`Double_Type`			
Description	Evaluates the integrand of the photon trajectory at the point `x`. Note, that the numerical integration functions require only one argument and additional parameters, like u and alpha, to be given as qualifiers.			

`lb_psi_ua_exact( u, alpha )`

Argument	`u = Double_Type`	*Inverse emission radius $u = r_s/R$*		
	`alpha = Double_Type`	*Emission angle α*		
Qualifier	method	[&qromb]	`(R)`	*Integration method*
	fkt	[&lb_psi_ua_exact_]	`(R)`	*Function to be integrated*
	b	[lb_b_ua(u, alpha)]	`(D)`	*Impact parameter b*
Return	`Double_Type`			
Dependencies	`qromb(isisscripts), lb_psi_ua_exact_`			
Description	Calculates the angle Ψ according to Eq. 2.11 and Eq. 2.21 for $\alpha \leq \alpha_{\max}$ with numerical methods. This function will integrate the function assigned to the fkt qualifier, which needs to have exactly one argument (any other variables can be given as qualifier). The integration method can be set with the method qualifier, which requests three arguments: the function, which is to be integrated; the lower and upper integration limit.			

`lb_psi_ua( u, alpha )`

Argument	`u = Double_Type`	*Inverse emission radius $u = r_s/R$*		
	`alpha = Double_Type`	*Emission angle α*		
Qualifier	lbmeth	['belob']	`(S)`	*Projection method ['exact', 'belob', 'geome']*
Return	`Double_Type`			
Dependencies	`lb_psi_ua_exact, lb_psi_ua_belob`			
Description	Calculates the angle Ψ with the method set in the lbmeth qualifier. lbmeth='exact': Eq. 2.11; lbmeth='belob': Eq. 2.38; lbmeth='geome': $\Psi \equiv \alpha$.			

`lb_psir2b_map( par )`

Argument	`par = Par_Type`	*See* `lb_par_init`

Qualifier	nr	[par.nmr]	(I)	*Number of radii*
	na	[par.nma]	(I)	*Number of emission angles*
	rmin	[par.rmin]	(D)	*Minimal radius [r_s]*
	rmax	[par.rmax]	(D)	*Maximal radius [r_s]*
	lbmeth	['belob']	(S)	*Projection method ['exact', 'belob', 'geome']*
	ashape	[1.65]	(D)	*Shape of α grid*

Return `Map_Type := Struct_Type{(r,a,p,b)=Double_Type[nr,na]}`

Dependencies `lb_alphamax, lb_b_ua, lb_psi_ua`

Description Creates a table for the photon trace parameter within the boundaries [rmin, rmax], where nr gives the number of radii this region is sampled with (equidistant). na specifies the number of emission angles, with which the region $[0, \alpha_{\max}]$ is sampled. The step size between each α depends on ashape following Eq. 3.1. For lbmeth='exact', 'belob' or 'geome', ashape = 1.3, 1.65 and 1, respectively. A `Struct_Type` is returned, containing the radius, α, Ψ and b at na × nr different points given in the $r - \alpha$ plane.

`lb_dbb_map( )`

Qualifier	n	[32]	(I)	*Dimension of returned struct fields ($n \times n$)*
	belob	[]	(-)	*If given: $\alpha_{max} = \pi/2$*

Return `Struct_Type`

Dependencies `lb_alphamax, lb_b_ua, lb_psi_ua`

Description Similar to lb_psir2b_map, but calculates the exact as well as approximated photon trajectory parameter to provide the error $\delta\beta/\beta$ (see Fig. 2.6).

interpolation

`lb_interp( map, r, p )`

Argument	map	= `Map_Type`	*See* `lb_psir2b_map`
	r	= `Double_Type`	*Radius* [r_s]
	p	= `Double_Type`	*Apparent emission angle* Ψ

Qualifier	imethod1	[&interpol_polynomial]	(R)	*Interpretation method*
	imethod2	[&interpol_points]	(R)	*Interpretation method*
	next	[0]	(I)	*# of additional interpolation points*
	only	[]	(-)	*Only b is determined*

Return `(a=Double_Type, ) b=Double_Type`

Dependencies `interpol_points, interpol_polynomial(isisscripts)`

Description	Finds and returns the to r and p according emission angle a and impact parameter b by interpolating the given map (trajectory table). The interpolation method to perform the bilinear interpolation is set with the imethod2 qualifier and that to determine the second two required off grid vertices by imethod1, where next sets the additional points taken into account.

`lb_interp_test(  )`

Argument	`par = Par_Type` *See* `lb_par_init`			
Qualifier	inr	[10]	`(I)`	*Number of radii*
	inp	[50]	`(I)`	*Number of emission angles*
	irmin	[2.1]	`(D)`	*Minimal radius [r_S]*
	irmax	[9.9]	`(D)`	*Maximal radius [r_S]*
	map	[lb_psir2b_map(par)]	`(M)`	*Trajectory table*
	rnd	[]	`(-)`	*Random $r \in$ [irmin, irmax] and $\Psi \in [0, \Psi_{max}]$*
Return	`Struct_Type[inr]{(r,p,b,a,rb,rp)=Double_Type[inp]}`			
Dependencies	`lb_psir2b_map, lb_psimax`			
Description	Tests the interpolation performed by `lb_interp` by interpolating inr radii r within [irmin,irmax] and inp Ψ (p) within $[0, \Psi_\mathrm{max}(R)$ to obtain the according impact parameters b and emission angles a. The interpolated emission angle a then is used to calculate the real according impact parameter rb (Eq. 2.23) and the real apparent emission angle rp. If the rnd qualifier is given the initial radii and Ψ's are randomly distributed with their boundaries.			

Projection

`lb_sincos_rho(` st, ct, si, ci, sp, cp `)`

Argument	st = `Double_Type`	$\sin\theta$
	ct = `Double_Type`	$\cos\theta$
	si = `Double_Type`	$\sin i$
	ci = `Double_Type`	$\cos i$
	sp = `Double_Type`	$\sin\phi$
	cp = `Double_Type`	$\cos\phi$
Return	`sinrho=Double_Type, cosrho=Double_Type`	
Description	Calculates and returns $(\sin\rho, \cos\rho)$ (Eq. 2.61 & Eq. 2.60).	

`lb_beta( c, f, rs, r, st )`

Argument	c = `Double_Type`	*Speed of light*
	f = `Double_Type`	*NS spin frequency f*
	rs = `Double_Type`	*NS Schwarzschild radius r_s*
	r = `Double_Type`	*Emission radius R*
	st = `Double_Type`	$\sin\theta$
Return	`Double_Type`	
Description	Calculates and returns the absolute value of the spot velocity β (Eq. 2.52).	

`lb_delta( beta, alpha, psi, si, sp )`

Argument	beta = `Double_Type`	*Absolute value of spot velocity*
	alpha = `Double_Type`	*Emission angle α*
	psi = `Double_Type`	*Apparent emission angle Ψ*
	si = `Double_Type`	$\sin i$
	sp = `Double_Type`	$\sin\phi$
Return	`Double_Type`	
Description	Calculates and returns the Doppler factor δ (Eq. 2.50).	

`lb_psi_pt_( si, ci, st, ct, cp )`

Argument	si = `Double_Type`	$\sin i$
	ci = `Double_Type`	$\cos i$
	st = `Double_Type`	$\sin\theta$
	ct = `Double_Type`	$\cos\theta$
	cp = `Double_Type`	$\cos\phi$
Return	`Double_Type`	
Description	Calculates and returns the apparent emission angle Ψ (Eq. 2.58).	

`lb_k0_dir( r, k, a )`

Argument	r = `Vector_Type`	*Position vector $\boldsymbol{r}$ of spot*
	k = `Vector_Type`	*Line of sight $\boldsymbol{k}$*
	a = `Double_Type`	*Emission angle α*
Return	`Vector_Type`	
Description	Calculates and returns the initial emission vector $\boldsymbol{k}_0$ by rotating $\boldsymbol{r}$ around $\boldsymbol{r} \times \boldsymbol{k}$ with angle α.	

`lb_gamma( r, ds, alpha, k )`

Argument	r = `Vector_Type`	*Position vector $\boldsymbol{r}$ of spot*
	ds = `Double_Type`	*Surface vector $\mathbf{d}\boldsymbol{S}$*
	alpha = `Double_Type`	*Emission angle α*
	k = `Vector_Type`	*Line of sight $\boldsymbol{k}$*
Return	`Double_Type`	

Description Calculates and returns the emission offset angle γ between the initial emission direction $\boldsymbol{k}_0$ and the surface vector $\mathbf{d}\boldsymbol{S}$ of the spot.

`lb_projectobj( par, obj, map )`

Argument	par = `Par_Type`	*See* `lb_par_init`		
	obj = `Obj_Type`	*See* `lb_obj_init`		
	map = `Map_Type`	*See* `lb_psir2b_map`		
Qualifier	phase	[lb_grid_phase(par)]	`(D[])`	*Phase grid*
	rmin	[par.R]	`(D)`	*Radius of spheric NS [m]*
	chatty	[0]	`(I)`	*Information output*

Return
```
Bend_Type := Struct_Type[#phase]{
(psi,alpha,gamma,delta,dp)=Array_Type[#SE](Double_Type[#sol])},
Pro_Type := Struct_Type[#phase]{
(a,b)=Array_Type[#SE](Double_Type[#sol]),
(pga,pgb)=Array_Type[#SE][#sol](Double_Type[3]}
```

Dependencies `lb_grid_phase, lb_b_ua, lb_obj_cart2sph, lb_sincos_rho, lb_psi_pt_, lb_interp, lb_beta, lb_gamma, lb_delta, lb_alphamax, lb_obj_dphi`

Description Performs the projection procedure of the given object obj based on the table map at the individual phases set with phase. The value given with rmin defines the minimal radius a trajectory can exhibit without hitting the surface of the spherical neutron star surface. The function returns two `Struct_Types`, where each field is an array according to the number of phases (#phase). The first structure contains $(\Psi, \alpha, \gamma, \delta, \Delta\phi)$ for each surface element (#SE) and each possible solution (#sol), where $\Delta\phi$ is the phase delay (Eq. 2.74). Note, that the value of $\Delta\phi$ is just a dummy. The second structure contains the impact parameter A and B (Eq. 2.59) of the centroid of each spot and those of the according vertices (pga, pgb).

`lb_se_size_projected( a, b, gamma )`

Argument	a = `Double_Type[3]`	*A of spot vertices*
	b = `Double_Type[3]`	*B of spot vertices*
	gamma = `Double_Type`	*Emission offset angle γ*

Return `Double_Type`

Description Calculates and returns the apparent spot area for one spot in the observer sky.

`lb_se_sizemagni( par, obj, proj )`

Argument	par = `Par_Type`	*See* `lb_par_init`
	obj = `Obj_Type`	*See* `lb_obj_init`
	proj = `Pro_Type`	*See* `lb_projectobj`

Qualifier	magni	[]	`(-)`	*Sizes relative to* $\|\mathbf{d}\boldsymbol{S}\|$
	smin	[]	`(R)`	*If given, stores min. projected spot size*
	smax	[]	`(R)`	*If given, stores max. projected spot size*
	oversize	[]	`(R)`	*If given, stores lists of indices of oversized spots (*par.magnilim*)*

Return `Array_Type[#phase][#SE](Double_Type[#sol])`

Dependencies `lb_se_size_projected`

Description Calculates and returns the apparent spot area for every spot at every phase in the projection proj of the object obj. If the magni qualifier is given, the spot sizes are divided by their according size $|\mathbf{d}\boldsymbol{S}|$ in the corotating frame. Additionally a `Ref_Type` smin or smax can be given, which then stores the minimal and maximal occurring projected spot size. A list of indices related to those spots, which exceed the specified magnification limit set with par.magnilim is stored in oversize, which is `OList_Type := Array_Type[#phase](Integer_Type[*,2])`.

overlap

`point_in_triangle(` px,py,x,y `)`

Argument	px	`= Double_Type`	*x coordinates of point*
	py	`= Double_Type`	*y coordinates of point*
	x	`= Double_Type[3]`	*x coordinates of triangle*
	y	`= Double_Type[3]`	*y coordinates of triangle*

Return `Double_Type`

Description Returns 1, if point (px, py) is within the triangle defined by (x, y) and 0, if not.

`lb_se_maxabsize(` pro `)`

Argument pro `= Pro_Type[1]` *See* `lb_projectobj`

Qualifier oversize [Integer_Type[0,2]] `(I[*,2])` *Indices of oversized surface elements*

Return `(amin,amax,bmin,bmax,maxda,maxdb)=Double_Type`

Dependencies `lb_se_in_overlaplist`

Description Determines the maximal overall expansion (amin, amax) and (bmin, bmax) in A and B coordinates on the observer sky. Additionally the maximal surface element size (maxda, maxdb) is returned. Surface elements in the oversize list are being ignored.

`lb_se_lookuptable(` pro `)`

Argument pro `= Pro_Type[1]` *See* `lb_projectobj`

Qualifier	oversize	[Integer_Type[0,2]]	`(P)`	*Indices of oversized surface elements*
	enlarge	[10^{-5}]	`(D)`	*Area enlargement percentage*
	notable	[]	`(-)`	*Table with only one field*

Return `Array_Type[na,nb](Integer_Type[2])`

Dependencies `lb_se_in_overlaplist`

Description Creates a lookup table for the location of the surface elements. The dimensions (na, nb) of the table are determined by the values obtained from `lb_se_maxabsize`. If na or nb are smaller three, or the notable qualifier is given, na and nb are set to zero.

`lb_neighbours(` I, J, dim `)`

Argument	I =	`Integer_Type`	*Index* I
	J =	`Integer_Type`	*Index* J
	dim =	`Integer_Type[2]`	*Dimensions of* I *and* J

Return `Integer_Type[*,*]`

Description Returns a index list of the neighbors cell of the given cell (I, J) in the lookup table, which has the dimensions dim.

`lb_se_overlap(` par, obj, pro `)`

Argument	par =	`Par_Type`	*See* `lb_par_init`
	obj =	`Obj_Type`	*See* `lb_obj_init`
	pro =	`Pro_Type`	*See* `lb_projectobj`

Qualifier	oversize	[OList_Type]	`(O)`	*Array of indices of oversized surface elements, see* `lb_se_sizemagni`

Return `OList_Type`

Dependencies `lb_se_lookuptable, lb_neighbours, point_in_triangle, array_remove(isisscripts)`

Description Returns for each phase a list of indices of those surface elements, which are covered by another surface element in the observer sky. Surface elements listed in oversize are ignored.

`lb_se_in_overlaplist(` olaplist, i, j `)`

Argument	olaplist =	`OList_Type`	*See* `lb_se_sizemagni`
	i =	`Integer_Type`	*Index of the surface element*
	j =	`Integer_Type`	*Index of the solution*

Return `Integer_Type`

Description Returns 1, if the j-th solution of the i surface element is listed in the given list olaplist.

`flux`

`lb_fluxpar_init( par )`

Argument	`par = Par_Type` *See* `lb_par_init`			
Qualifier	FID	['fluxmodel']	(S)	*Flux model identification*
	fpath	[]	(S)	*Saving (sub) path for flux model*
	nphase	[1]	(D)	*# phases for time grid*
	tend	[1/par.f]	(D)	*Maximal time for time grid*
	Ebin	[1]	(I)	*# energy bins*
	Emin	[1.]	(D)	*Minimal energy*
	Emax	[100.]	(D)	*Maximal energy*
	Escl	['log']	(S)	'log' *or* 'linear' *energy grid*
	flux	['bolo']	(S)	*bolometric or spectral flux*
	flux_eq	[]	(S)	*Stores flux model*
	tdelay	[0]	(I)	*Consider time delay 0/1*
	periodic	[1]	(I)	*Periodical pulse profile 0/1*

Return `FPar_Type := Struct_Type{FID, fpath, flux, flux_eq, nphase, tend, Ebin, Emin, Emax, Escl, tdelay, periodic}`

Description Initializes a structure containing parameter related to the flux calculations. A unique flux model name can be given with FID. fpath specifies a path for saving, which is a sub folder of par.path by default. For the time grid either nphase or tend can be set, which defines the number of phases to included or the end time. The energy grid has Ebin bins between Emin and Emax with a linear or logarithmic scale (Escl='log','linear'). flux='bolo','spec' defines whether the flux is bolometric or spectral, where the according flux model is stored in flux_eq. If tdelay=1 the time delay considered (not implemented yet) and if in addition periodic=1 the pulse profile is periodically continued.

`lb_grid_energy( fpar )`

Argument `fpar = FPar_Type` *See* `lb_fluxpar_init`

Return `Double_Type[fpar.Ebin+1]`

Description Creates a linear or logarithmic energy grid, according to fpar.Escl between fpar.Emin and fpar.Emax with fpar.Ebin bins. The returned array contains the nested bin low/high values.

`lb_grid_time( par, fpar )`

Argument	`par = Par_Type` *See* `lb_par_init`			
	`fpar = FPar_Type` *See* `lb_fluxpar_init`			
Qualifier	nphase	[fpar.nphase]	(D)	*# phases for time grid*
	tend	[fpar.tend]	(D)	*Maximal time for time grid*

Return `lb_grid_phase`

Dependencies `Double_Type[]`

Description Creates a time grid within $[0, \text{tend}]$. The returned array contains the nested bin low/high values.

`lb_int_dist_normal( x, s, mu )`

Argument	x = `Double_Type[]`	
	s = `Double_Type`	*Standard deviation σ*
	mu = `Double_Type`	*Mean μ*
Return	`Double_Type[]`	
Description	Calculates the normal distribution with standard deviation s and mean mu at x.	

`lb_int_dist_uniform( x, lo, hi )`

Argument	x = `Double_Type[]`	
	lo = `Double_Type`	*Lower limit*
	hi = `Double_Type`	*Upper limit*
Return	`Integer_Type[]`	
Description	Returns 1, if x is within [lo, hi], otherwise 0.	

`lb_domega( par, delta, gamma )`

Argument	par = `Par_Type`	*See* `lb_par_init`
	delta = `Double_Type`	*Doppler factor δ*
	gamma = `Double_Type`	*Emission offset angle γ*
Return	`Double_Type`	
Description	Calculates $\mathrm{d}\Omega$ (Eq. 2.65).	

`lb_efrac( u, delta )`

Argument	u = `Double_Type`	*Inverse emission radius u*
	delta = `Double_Type`	*Doppler factor δ*
Return	`Double_Type`	
Description	Calculates the energy ratio E/E' (Eq. 2.69).	

`lb_dflux_define( par, fpar )`

Argument	par = `Par_Type`		*See* `lb_par_init`
	fpar = `FPar_Type`		*See* `lb_fluxpar_init`
Qualifier	sig $[45\frac{\pi}{180}]$	`(D)`	*Standard deviation/width*
	dis ['isho']	`(S)`	*Intensity distribution $I'(\gamma')$*
Description	Defines the `lb_dflux` function, which calculates the flux (Eq. 2.70 or Eq. 2.72), according to the parameter specified in par, fpar and the qualifier. dis defines the intensity distribution: 'isho' for isotropic and homogeneous intensity; 'uniform'/'normal' for a uniform/normal distribution of $I'(\gamma')$ with width/standard deviation sig around the mean $\gamma = 0$. If fpar.flux='spec' the exponent of E/E' is set to the according value for the spectral flux (Eq. 2.68), otherwise the bolometric value (Eq. 2.71) is used. Additionally the definition of `lb_dflux` is stored in fpar.flux_eq as string.		

`lb_dflux(` par, R, delta, gamma `)`

Argument	par = `Par_Type`	*See* `lb_par_init`	
	R = `Double_Type`	*Emission radius R*	
	delta = `Double_Type`	*Doppler factor δ*	
	gamma = `Double_Type`	*Emission offset angle γ*	
Return	`Double_Type`		
Dependencies	`lb_int_dist_normal, lb_int_dist_uniform, lb_dcadcp_belob, lb_dcadcp_exact, lb_domega, lb_efrac`		
Description	This function is defined by `lb_dflux_define` and calculates the flux for a surface element.		

`lb_flux_se(` par, fpar, obj, bend `)`

Argument	par = `Par_Type`	*See* `lb_par_init`		
	fpar = `FPar_Type`	*See* `lb_fluxpar_init`		
	obj = `Obj_Type`	*See* `lb_obj_init`		
	bend = `Bend_Type`	*See* `lb_projectobj`		
Qualifier	egrid	[lb_grid_energy(fpar)]	`(D[])`	*Energy grid*
	tgrid	[lb_grid_time(par,fpar)]	`(D[])`	*Time grid*
Return	`SEFlux_Type := Array_Type[#tbins][#SE][#sol][#Ebins]`			
Dependencies	`lb_dflux_define, ld_dflux, lb_grid_energy, lb_grid_time`			
Description	Calculates the flux for each surface elements according the definition of `lb_dflux` using the specified time and energy grid. Note that the time grid depends on the phase grid used for the projection procedure.			

`lb_flux_add(` flux `)`

Argument	flux = `Flux_Type`	*See* `lb_flux`		
Qualifier	combis	[['all','ac','ac1','ac2','cap','col']]	`(S[])`	*NS Part combinations*
Return	`Flux_Type`			
Description	This functions adds to flux the possible combination out of combis of the fields of the given flux structure.			

`lb_flux(` par, fpar, obj, flux_se, bend `)`

Argument	par = `Par_Type`	*See* `lb_par_init`
	fpar = `FPar_Type`	*See* `lb_fluxpar_init`
	flux_se = `SEFlux_Type`	*See* `lb_flux_se`
	obj = `Obj_Type`	*See* `lb_obj_init`
	bend = `Bend_Type`	*See* `lb_projectobj`

Qualifier	egrid	[lb_grid_energy(fpar)]	(D[])	*Energy grid*
	tgrid	[lb_grid_time(par,fpar)]	(D[])	*Time grid*
	tdelay	[fpar.tdelay]	(I)	*Consider time delay 0/1*
	periodic	[fpar.periodic]	(I)	*Periodical pulse profile 0/1*
	overlap	[OList_Type]	(O)	*Array of indices of oversized/overlapping surface elements, see* `lb_se_sizemagni`

Return `Flux_Type := Struct_Type{ <fields> = Double_Type[#tbins,#ebins]}`

Dependencies `lb_flux_add`

Description Calculates the overall flux based on the given flux of each surface element flux_se. The <fields> of the returned structure relate to the different parts of the neutron star (ns,ac1cap,ac1col,ac2cap,ac2col) and their combinations: all, ac, ac1, ac2, cap or col. The structure only contains those fields, which relate to parts of the neutron star specified to emit photons (par.onns, par.oncap, par.oncol). Each <fields> is a (#tbins × #ebins) matrix, where each entry corresponds to the flux at a certain time and energy.

plotting

`lb_xfig_prepare_grid(` pro, grid `)`

Argument pro = `Pro_Type[1]` *See* `lb_projectobj`
obj = `Obj_Type` *See* `lb_obj_init`

Return `Struct_Type[]{ (a,b) = Integer_Type[]}`

Description For a given projection (pro) at one phase the grid lines (obj.grid) are prepared for plotting. obj.grid is an `Array_Type[#gridlines]`, where each entry contains indices of those surface element vertices defining the grid line. This function selects the according A and B solutions in pro. Each individual continuous line is stored as an entry of the returned `Struct_Type[]`.

`lb_array_trans(` array `)`

Argument array = `Array_Type[](<type>)`

Return `<type>[]`

Description Transforms the given `Array_Type` into a `<type>` array.

`lb_delta_mima(` bend `)`

Argument bend = `Bend_Type` *See* `lb_projectobj`

Return `(dmin, dmax) = Double_Type`

Dependencies `lb_array_trans`

Description	Finds the minimal and maximal value of the Doppler factor (dmin, dmax) in bend.

`lb_seflux_mima(` seflux `)`

Argument	seflux = `SEFlux_Type` *See* `lb_flux_se`
Return	`(fmin, fmax) = Double_Type`
Dependencies	`lb_array_trans`
Description	Finds the minimal and maximal value of the flux of a single surface element (fmin, fmax) in seflux.

`lb_xfig_plot_pulseprofile(` par, fpar, flux `)`

Argument	par = `Par_Type` *See* `lb_par_init`
	fpar = `Fpar_Type` *See* `lb_fluxpar_init`
	flux = `Flux_Type` *See* `lb_flux`

Qualifier

path	[fpar.fpath]	`(S)`	*Saving path*
name	[]	`(S)`	*File name*
type	['.pdf']	`(S)`	*File type*
usefields	[]	`(S[])`	*Fields of flux to plot*
lstyle	[]	`(I[])`	*Line styles*
lcolor	[]	`(S[])`	*Line colors*
lwidth	[]	`(I[])`	*Line widths*
size	[[12,9]]	`(D[2])`	*Plot sizes*
tgrid	[lb_grid_time(par,fpar)]	`(D[])`	*Time grid*
energy	[[*]]	`(I[])`	*Indices of energy to include & sum up*
norm	[]	`(D)`	*Normalization*
pmark	[]	`(D)`	*Phase marker within* $[0, 1]$
fonts	[4]	`(I)`	*Font size within* $[0, 9]$
norender	[]	`(-)`	*Plot will not be rendered*

Return	`Struct_Type`
Dependencies	`lb_grid_time, (slxfig), (isisscripts)`
Description	Creates and renders a xfig plot showing the pulse profile(s) of the specified fields (usefields) of the given flux. By default usefields includes all available fields. The line style(s), line color(s) and line width(s) can be specified by style, lcolor and lwidth, where their lengths must match the numbers of fields to plot. The given time grid (tgrid) specifies the time axis label. The energy qualifier is a list of indices related to the energy grid used to calculate the flux. The according entries in the given flux will be summed up to create the pulse profile. A normalization norm for the flux can be given and corresponds by default the maximal value occurring in the pulse profile. If pmark is set to a value within $[0, 1]$ a marker is plotted at the according phase.

`lb_xfig_plot_projection(` par, fpar, obj, pro, bend, seflux `)`

Argument	par = `Par_Type`	*See* `lb_par_init`	
	fpar = `Fpar_Type`	*See* `lb_fluxpar_init`	
	obj = `Obj_Type`	*See* `lb_obj_init`	
	pro = `Pro_Type`	*See* `lb_projectobj`	
	bend = `Bend_Type`	*See* `lb_projectobj`	
	seflux = `SEFlux_Type`	*See* `lb_flux_se`	

Qualifier	path	[fpar.fpath]	`(S)`	*Saving path*
	name	[]	`(S)`	*File name*
	type	['.png']	`(S)`	*File type*
	size	[2]	`(D)`	*Plot size factor*
	fonts	[4]	`(I)`	*Font size within* $[0, 9]$
	rmax	[]	`(D)`	*Plot dimensions*
	dmin	[]	`(D)`	*Minimal Doppler factor*
	dmax	[]	`(D)`	*Maximal Doppler factor*
	fmin	[]	`(D)`	*Minimal SE flux*
	fmax	[]	`(D)`	*Maximal SE flux*
	color1	['#111111']	`(S)`	*Grid element color*
	color2	['#333333']	`(S)`	*NS grid element color*
	color3	['brown']	`(S)`	*AC grid element color*
	overlap	[OList_Type]	`(O)`	*See* `lb_se_sizemagni`
	background	['black']	`(S)`	*No axis, background color*
	notrickdepth	[]	`(-)`	*No SE depth calculation*
	grid	[]	`(-)`	*Additional grid lines*
	nocolorscale	[]	`(-)`	*No color scale*
	flux	[]	`(F)`	*Additional pulse profile*
	usefields	[['all','ac1','ac2']]	`(S[])`	*Fields of flux to plot*
	makemovie	[]	`(-)`	*Creates movie*
	bitrate	[600]	`(I)`	*Bit rate of movie*
	fps	[8]	`(I)`	*Frame per second of movie*

Return `Struct_Type`

Dependencies `(slxfig); (isisscripts); lb_xfig_add_colorscale, color_create_map(plotting_colormap.sl); lb_delta_mima, lb_seflux_mima, lb_xfig_prepare_grid, lb_xfig_plot_pulseprofile`

Description Renders a plot of the projected object pro for each phase. Plot dimensions in A and B are given with rmax. For surface elements with obj.on > 0 the plotted color corresponds to its Doppler factor and the brightness to its flux. The minimal and maximal values for the Doppler factor and the flux are given by dmin, dmax, fmin and fmax or being automatically calculated. Surface elements in the overlap list are not plotted. If flux=`Flux_Type` is given a pulse profile for each usefields is added. grid activates grid lines. notrickdepth disables the depth calculation of the individual surface elements. If background is given, axis are deactivated and the background color is set accordingly.

storing

`fits_read_keys(` file, keynames `)`

Argument	file = `S` *Path to file*
	keynames = `S[]` *Keys names to read in file*
Return	`Struct_Type{ <keys> }`
Description	Reads keys, whose names are given with keynames, in file

`lb_save_prep(` par `)`

Argument	par = `Par_Type` *See* `lb_par_init`
Qualifier	itlim [10] `(I)` *Maximal new path tries*
	overwr [0] `(I)` *Overwrite existing path 0/1*
Description	Prepares saving path and test for its existence. If path already exists a new path is chosen or is overwritten if overwr qualifier exists.

`lb_save_make_fname(` par (, fpar), fext `)`

Argument	par = `Par_Type` *See* `lb_par_init`
	fpar = `FPar_Type` *See* `lb_fluxpar_init` *(optional argument)*
	fext = `String_Type` *File name extension*
Return	`String_Type`
Description	Generates a file path and file name.

`lb_load_file_exists(` fname `)`

Argument	fname = `String_Type` *File name*
Description	Tests the existence of a file and returns the number of files with name fname.

`struct_compare(` s, r `)`

Argument	s = `Struct_Type`
	r = `Struct_Type`
Qualifier	sigdig [13] `(I)` *Significant digits of doubles to test*
Dependencies	`String_Type[]`
Description	Compares the structure s to the structure r and returns mismatching fields as string array. sigdig specifies the number of digits for `Double_Type`, which are being compared.

`lb_save_parameter(` par `)`

Argument	par = `Par_Type` *See* `lb_par_init`
Qualifier	fname [par.ID+'_par'] `(S)` *File name*
Description	Saves parameter in an ISIS readable file.

`lb_load_parameter( fname )`

Argument	`fname = String_Type` *File name*
Dependencies	`lb_load_file_exists`
Description	Evaluates `fname` (in ISIS), if existent.

`lb_save_map( par, map )`

Argument	`par = Par_Type` *See* `lb_par_init`
	`map = Map_Type` *See* `lb_psir2b_map`
Dependencies	`lb_save_make_fname`

`lb_load_map( par )`

Argument	`par = Par_Type` *See* `lb_par_init`
Return	`Map_Type`
Dependencies	`lb_load_file_exists, fits_read_keys, struct_compare`
Description	Loads `Map_Type` and compares the parameters

`Vector2Double( v )`

Argument	`v = Vector_Type[]`
Return	`Double_Type[*,3]`

`Double2Vector( d )`

Argument	`d = Double_Type[*,3]`
Return	`Vector_Type[]`

`Array2Matrix( a )`

Argument	`a = Array_Type[]`
Return	`<type>[d1, d2 (, d3)]`
Description	Converts an `Array_Type[d1](<type>[d2])` to a `<type>[d1,d2]` Matrix and an `Array_Type[d1][d2](<type>[d3])` to a `<type>[d1,d2,d3]` Matrix.

`Matrix2Array( m )`

Argument	`m = <type>[d1, d2 (, d3)]`
Return	`Array_Type[]`
Description	Converts a `<type>[d1,d2]` Matrix to an `Array_Type[d1](<type>[d2])` and a `<type>[d1,d2,d3]` Matrix to an `Array_Type[d1][d2](<type>[d3])`.

`lb_save_object(` par,obj `)`

Argument	par = `Par_Type` *See* `lb_par_init`
	obj = `Obj_Type` *See* `lb_obj_init`
Dependencies	`lb_save_make_fname, Vector2Double, Array2Matrix`

`lb_load_object(` par `)`

Argument	par = `Par_Type` *See* `lb_par_init`
Return	`Obj_Type`
Dependencies	`lb_save_make_fname, lb_load_file_exists, fits_read_keys, struct_compare, Double2Vector, Matrix2Array`

`lb_merge_phasedfiles(` par `)`

Argument	par = `Par_Type` *See* `lb_par_init`
Dependencies	`fits_read_keys`
Description	Merges files relating to the same <object> (e.g. `Bend_Type`) into one file, which were split up into different phases.

`lb_save_projection(` par, proj `)`

Argument	par = `Par_Type` *See* `lb_par_init`
	proj = `Pro_Type` *See* `lb_projectobj`
Qualifier	noff [0] `(I)` *# of first phase bin*
Dependencies	`lb_save_make_fname, Array2Matrix`
Description	Saves `Pro_Type`. If proj does not include par.nphi phases the name of created file is extended with the phase bin numbers it includes, where the number of the first phase included in proj has to be specified with noff.

`lb_load_projection(` par `)`

Argument	par = `Par_Type` *See* `lb_par_init`
Dependencies	`lb_save_make_fname, lb_load_file_exists, fits_read_keys, struct_compare, Matrix2Array`

`lb_save_bending(` par, bend `)`

Argument	par = `Par_Type` *See* `lb_par_init`
	Bend = `Bend_Type` *See* `lb_projectobj`
Qualifier	noff [0] `(I)` *# of first phase bin*
Dependencies	`lb_save_make_fname, Array2Matrix`

Description	Saves `Bend_Type`. If bend does not include par.nphi phases the name of created file is extended with the phase bin numbers it includes, where the number of the first phase included in bend has to be specified with noff.

`lb_load_bending(` par `)`

Argument	par = `Par_Type` *See* `lb_par_init`
Dependencies	`lb_save_make_fname, lb_load_file_exists, fits_read_keys, struct_compare, Matrix2Array`

`lb_save_arraytype(` par, array, fname, fext `)`

Argument	par =	`Par_Type`	*See* `lb_par_init`
	array =	`Array_Type`	
	fname =	`String_Type`	*File path and name*
	fext =	`String_Type`	*Extension name*
Qualifier	noff [0]	`(I)`	*# of first phase bin*
Dependencies	`Matrix2Array`		
Description	Saves an `Array_Type[d1](<type>[d3])` or `Array_Type[d1][d2](<type>[d3])`. If d1 is not equal to par.nphi the file name is extended with the numbers of phase bin range the array relates to, where the first phase bin must be set with noff. fext is the extension name corresponding to the single d1 entries.		

`lb_load_arraytype(` par (, fpar), fext `)`

Argument	par =	`Par_Type`	*See* `lb_par_init`
	fpar =	`FPar_Type`	*See* `lb_fluxpar_init` *(optional argument)*
	fext =	`String_Type`	*File name extension*
Return	`OList_Type or SEFlux_Type`		
Dependencies	`lb_save_make_fname, lb_load_file_exists, fits_read_keys, struct_compare, Matrix2Array`		

`lb_save_overlap(` par, olap `)`

Argument	par =	`Par_Type`	*See* `lb_par_init`
	olap =	`OList_Type`	*See* `lb_se_sizemagni`
Dependencies	`lb_save_make_fname, lb_save_arraytype`		

`lb_load_overlap(` par `)`

Argument	par = `Par_Type` *See* `lb_par_init`
Return	`OList_Type`

Dependencies	`lb_load_arraytype`

`lb_save_oversize(` par, osize `)`

Argument	par = `Par_Type` *See* `lb_par_init`
	osize = `OList_Type` *See* `lb_se_sizemagni`
Dependencies	`lb_save_make_fname, lb_save_arraytype`

`lb_load_oversize(` par `)`

Argument	par = `Par_Type` *See* `lb_par_init`
Return	`OList_Type`
Dependencies	`lb_load_arraytype`

`lb_save_fluxparameter(` par, fpar `)`

Argument	par = `Par_Type` *See* `lb_par_init`
	fpar = `FPar_Type` *See* `lb_fluxpar_init`

`lb_load_fluxparameter(` par `)`

Argument	par = `Par_Type` *See* `lb_par_init`
Qualifier	FID [”] (S) *Flux identification*
Return	`FPar_Type[]`
Dependencies	`lb_load_file_exists, fits_read_keys, struct_compare`
Description	Reads and returns every `FPar_Type` matching the specifications in par and has the sub string FID in its file name.

`lb_save_seflux(` par, fpar, seflux `)`

Argument	par = `Par_Type` *See* `lb_par_init`
	fpar = `FPar_Type` *See* `lb_fluxpar_init`
	seflux = `SEFlux_Type` *See* `lb_flux_se`
Dependencies	`lb_save_make_fname, lb_save_arraytype`

`lb_load_seflux(` par, fpar `)`

Argument	par = `Par_Type` *See* `lb_par_init`
	fpar = `FPar_Type` *See* `lb_fluxpar_init`
Return	`SEFlux_Type`
Dependencies	`lb_load_arraytype`

`lb_save_flux(` par, fpar, flux)

Argument	par	=	`Par_Type`	*See* `lb_par_init`
	fpar	=	`FPar_Type`	*See* `lb_fluxpar_init`
	flux	=	`Flux_Type`	*See* `lb_flux`
Dependencies	`lb_save_make_fname`			

`lb_load_flux(` par, fpar)

Argument	par	=	`Par_Type`	*See* `lb_par_init`
	fpar	=	`FPar_Type`	*See* `lb_fluxpar_init`
Dependencies	`lb_save_make_fname, lb_load_file_exists, fits_read_keys`			

ACKNOWLEDGMENTS

I would like to thank my supervisors Jörn Wilms for giving me the opportunity to do my master thesis at the Dr. Karl Remeis observatory in Bamberg and his support. Thomas Dauser deserves special thanks, not only for providing the topic of my thesis and introducing me to the general relativistic subject, but beyond for always being patient and helping with all the question I came up with related to the subject of my thesis. He also always shared his knowledge about ISIS, S-LANG and SLXFIG. I also want to thank Maurizio Falanga for is support, helpfulness and enthusiasm.

I am also grateful for all the advices and help I got from Matthias Kühnel and Moritz Böck improving my programming and plotting skills, the helpful discussion about general relativity with Fabian Brod and the expertise of Sebastian Müller and Ingo Kreykenbohm on neutron star. Furthermore, I want to thanks all my colleagues of the Dr. Karl Remeis observatory for the pleasant working atmosphere and helpfulness. Amongst them my fellow students Felicia Kraußand Tobias Beuchert, with whom I had many helpful discussion and Cornelia Müller, who always cheered me up, especially in the final stages of this thesis. Further I am very thankful for the support Mirjam Örtel during the lap course. Furtheremore, I want to thank my current and former workmates in my office, Thomas Dauser, Maria Obst, Felicia Krauß, Christian Schmid, Michael Wille, Fabian Brod, Felix Fürst and Nathalie Hell for all the fun. I also appreciate the administrative work of Ingo Kreykenbohm, Fritz Schwarm, Thomas Dauser and Jörn Wilms, who always ensured, that the Remeis systems are operational. Moreover, I am grateful for the great social cohesion of all members of the observatory and all the great and funny social activities we experienced together, making the Remeis observatory not only a working place, but also a big community.

Finally I want to thank my whole family, my parents and grandparents, who always supported me in any imaginable way and in all my decisions, and made it possible for me to follow my way and my sister for her patience.

List of Figures

References

Araya R.A., Harding A.K., 1999, ApJ 517, 334

Becker P.A., Wolff M.T., 2007, ApJ 654, 435

Beloborodov A.M., 2002, Astrophys. J., Lett. 566, L85

Blum S., Kraus U., 2000, ApJ 529, 968

Caballero I., Kraus U., Santangelo A., et al., 2011 526, A131

Cadeau C., Morsink S.M., Leahy D., Campbell S.S., 2007, ApJ 654, 458

Carroll S.M., 2004, Spacetime and geometry. An introduction to general relativity, Addison Wesley

Chandrasekhar S., 1983, The mathematical theory of black holes

Dyson F.W., Eddington A.S., Davidson C., 1920, Royal Society of London Philosophical Transactions Series A 220, 291

Einstein A., 1916, Annalen der Physik 354, 769

Einstein A., 1936, Science 84, 506

Ferrigno C., Falanga M., Bozzo E., et al., 2011 532, A76

Fließbach T., 1990, Allgemeine Relativitätstheorie., BI Wissenschaftsverl.

Fürst F., 2011, Ph.D. thesis, Friedrich-Alexander-Universität Erlangen-Nürnberg

Haensel P., Potekhin A.Y., Yakovlev D.G., (eds.) 2007, Neutron Stars 1 : Equation of State and Structure, Vol. 326 of *Astrophysics and Space Science Library*, Astrophysics and Space Science Library

Hanisch R.J., Farris A., Greisen E.W., et al., 2001 376, 359

Hartle J.B., 2003, Gravity : an introduction to Einstein's general relativity, Addison Wesley

Hessels J.W.T., Ransom S.M., Stairs I.H., et al., 2006, Science 311, 1901

Houck J.C., Denicola L.A., 2000, In: Manset N., Veillet C., Crabtree D. (eds.) Astronomical Data Analysis Software and Systems IX, Vol. 216. Astronomical Society of the Pacific Conference Series, p. 591

Kerr R.P., 1963, Physical Review Letters 11, 237

Kraus U., 2001, ApJ 563, 289

Kraus U., Nollert H.P., Ruder H., Riffert H., 1995, ApJ 450, 763

Krolik J.H., 1999, Active galactic nuclei : from the central black hole to the galactic environment, Princeton University Press

Lamb F.K., Pethick C.J., Pines D., 1973, ApJ 184, 271

Misner C.W., Thorne K.S., Wheeler J.A., 1973, Gravitation

Orwant J., Hietaniemi J., Macdonald J., 1999, Mastering algorithms with Perl - practical programming through computer science., O'Reilly

Pechenick K.R., Ftaclas C., Cohen J.M., 1983, ApJ 274, 846

Poutanen J., Beloborodov A.M., 2006, MNRAS 373, 836

Press W.H., Teukolsky S.A., Vetterling W.T., Flannery B.P., 1992, Numerical recipes in FORTRAN. The art of scientific computing, Cambridge: University Press

Sasaki M., Klochkov D., Kraus U., et al., 2010 517, A8

Sasaki M., Müller D., Kraus U., et al., 2012 540, A35

Schönherr G., Wilms J., Kretschmar P., et al., 2007 472, 353

Schwarzschild K., 1916, Abh. Konigl. Preuss. Akad. Wissenschaften Jahre 1906,92, Berlin,1907 189–196

Smartt S.J., 2009 47, 63

Steiner A.W., Lattimer J.M., Brown E.F., 2013, Astrophys. J., Lett. 765, L5

Wheeler J.A., 1990, A Journey into Gravity and Spacetime, Scientific American Library

Declaration

Hiermit erkläre ich, dass ich die Arbeit selbstständig angefertigt und keine anderen als die angegebenen Hilfsmittel verwendet habe.

Bamberg, 28.03.2013

Sebastian Falkner