SURFACE DEFORMATIONS IN VAIDYA SPACETIMES

HUAKUN DING

Your file Votre référence
ISBN: 978-0-494-57477-5
Our file Notre référence
ISBN: 978-0-494-57477-5

Surface Deformations in Vaidya Spacetimes

by

© Huakun Ding
B.Sc.

A thesis submitted to the
School of Graduate Studies
in partial fulfillment of the
requirements for the degree of
Master of Science
in
Mathematics.

Department of Mathematics
Memorial University of Newfoundland

July, 2007

ST. JOHN'S NEWFOUNDLAND

Contents

Abstract

The geometry and dynamics of black hole horizons can be easily studied by considering the allowed deformations of their foliating two-surfaces. This article presents a new method of calculating the geometry and deformations of two-surfaces embedded in spacetimes. The calculations are tested by comparing the results obtained with this method with some known results about these objects. By this method, the deformations of future outer trapping surfaces in Vaidya spacetimes are studied.

Acknowledgements

I would like to thank Dr. Ivan Booth, my supervisor, for many useful discussions and guidance along the way. Financial support was provided by the Natural Sciences and Research Council of Canada and the Memorial University of Newfoundland. Submitted in partial fulfillment of the requirements of the degree of Master of Science, the Memorial University of Newfoundland, St. John's, Newfoundland.

Chapter 1

Introduction and Overview

Black holes play an important role in general relativity and astrophysics. The characteristic feature that defines a black hole is its horizon. Traditionally, the black hole region of an asymptotically-flat spacetime is defined as the set of events from which no null curves can reach future null infinity [1]. The boundary of the black hole region is the *event horizon.*

With the help of conformal mappings [1], one can define precise notions of "infinity" for Minkowski space (see Figure 1.1) and can depict a black hole in a spacetime (the spacetime is asymptotically identical to Minkowski space). Let us use I^+ to represent future null infinity, then we can say a null curve has "escaped to infinity" if its counterpart in the conformally mapped spacetime terminates on I^+. It follows that spacetimes are said to contain a black hole if they have regions from which no null curve reach I^+ — thus no causal curve can escape those regions. Equivalently, as shown in Figure 1.1, a spacetime contains a black hole if the complement of the causal past of I^+ is nonempty. Then that complement is the black hole and the boundary of that region is the event horizon.

From the definition above we can see that the event horizon has two important

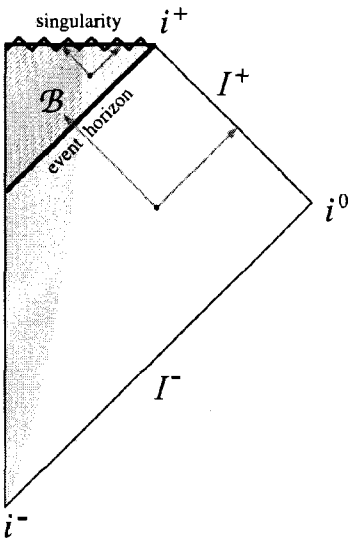


Figure 1.1: A Penrose-Carter diagram showing a spacetime in which a matter distribution collapses to form a black hole [2]. $i^{\pm}$ is future and (or) past timelike infinity, i^0 is the corresponding spacelike infinity, and $I^{\pm}$ is future and (or) past null infinity.

features [2]. First, it is based on the global causal structure of the spacetime. It is a highly nonlocal object as its existence depends on the structure of infinity. Second, it is teleological as we have to wait until the end of time to identify a black hole region. It can't be identified by local measurement. While much analytic work on black holes centers around event horizons, their dual nonlocalities make them not directly useful in numerical evolutions of black hole spacetime.

For such reasons, in numerical evolutions, it is more common to use *apparent horizons*, which is inspired from the important concept "trapped surface" proposed by Penrose R. [3], to characterize a black hole. A closed two-surface S is said to be trapped if $\theta_{(l)} < 0$ and $\theta_{(n)} < 0$, here $\theta_{(l)} = q^{ab}\nabla_a l_b$ and $\theta_{(n)} = q^{ab}\nabla_a n_b$ are respectively the expansions of the outward (l) and inward (n) forward-in-time pointing

null normal vectors to the two-surface and $q_{ab} = g_{ab} + l_a n_b + l_b n_a$ is the transverse two-metric on that surface. Given a Cauchy surface Σ_t, a point $p \in \Sigma_t$ is said to be trapped if it lies on some trapped two-surface in Σ_t. The apparent horizon in Σ_t is the two-dimensional boundary of the union of all of the trapped points and given certain smoothness assumption, it can be shown that it is a surface on which $\theta_{(l)} = 0$ (*marginally outer trapped*) [1]. Correspondingly, if a region of spacetime is foliated by Cauchy surfaces, then the apparent horizon on each slice can be found and so a time-evolved three-dimensional version of the apparent horizon can be defined. Often this is also referred to as the apparent horizon. Generally, the apparent horizon can be discontinuous, its behavior under time evolution can be quite irregular. Furthermore, apparent horizons in spacetime are not unique. A different foliation of the spacetime into spacelike surfaces will usually result in a different location of the apparent horizon through the spacetime [4].

Apparent horizons are more practical than event horizons, and they are well suited to numerical simulations that evolve data from one spatial slice to another. However, they are tied rigidly to the choice of a space-like three-surface so that they are still not global notions. In addition, apparent horizons are not helpful for deriving the laws of black hole mechanics. To solve these problems Hayward introduced the quasi-local notion of *trapping horizons* [5]. A trapping horizon is a hypersurface in a four-dimensional spacetime that is foliated by two-surfaces such that $\theta_{(l)} = 0$, $\theta_{(n)} \neq 0$ and $\delta_n \theta_{(l)} \neq 0$, where δ_n is the normal variation [6]. A trapping horizon can be thought of as the boundary of a black hole if it satisfies $\delta_n \theta_{(l)} < 0$, and $\theta_{(n)} < 0$, as in this case the surfaces "just inside" the horizon are fully trapped [2]. We will call such horizons future outer trapping horizons (FOTHs). FOTHs provide a promising framework to discuss black holes in general spacetimes, not just those which are asymptotically flat. Apart from trapping horizons some closely related programmes have been developed

recently such as isolated horizons which identify and study equilibrium states, and dynamical horizons which correspond to non-trivially evolving horizons[14].

FOTHs come equipped with a preferred foliation into two-surfaces. This foliation is directly relevant to the dynamics of FOTHs. In this paper we will focus on the spacelike two-surfaces that foliate FOTHs. We will call such surfaces future outer trapping surfaces (FOTS). The existence, evolutions and deformations of the full three-dimensional horizons can be well understood by studying the geometry and variations of these two-surfaces [6].

The boundary of the trapped region of a black hole spacetime has been studied both analytically and numerically recently. Eardley [7] suggests that the boundary should correspond to the event horizon, while arguments by Hayward suggest that the boundary should be a trapping horizon [5]. By numerically studying non-symmetric trapped surfaces in Vaidya spacetimes, Schnetter and Krishnan [8] support that the event horizon is the most likely candidate for the boundary of the trapped region. Lately Ben-Dov [9] has proved that in the Vaidya spacetime, there are outer trapped surfaces extending arbitrarily close to the event horizon in any region of the spacetime, thus the event horizon is the boundary of the region containing outer trapped surfaces (closed spacelike two-surfaces for which the outgoing null expansions $\theta_{(l)} < 0$). Furthermore, he shows that there exists a portion of the flat region of a Vaidya spacetime that trapped surfaces can not enter. As a result, the boundary of the region containing trapped surfaces, is not, in general, the event horiozn.

The Vaidya spacetime is the simplest example of a dynamical black hole spacetime [4]. It models the spherically symmetric collapse of null dust and is described by the metric

$$dS^2 = -(1 - \frac{2m(v)}{r})dv^2 + 2dvdr + r^2d\theta^2 + r^2sin^2\theta d\phi^2,$$

where v is an advanced time coordinate, and $m(v)$ is the mass function. We will see in section 4 that the surfaces $r = 2m(v)$ in the Vaidya spacetime are all FOTS. Thus the Vaidya spacetime provides a convenient framework to study the problem about the boundary of the trapped region of a black hole spacetime. In this paper, to gain some insight into this problem, we consider the geometry and deformations of FOTS embedded in Vaidya spacetimes. In particular, we study the finite extensions of FOTS where we try to "push" them towards the event horizon.

This paper is organized as follows. In section 2, to establish our notation, we begin by reviewing the basic material from the theory of embedding of a two-surface in a three-dimensional manifold. Then we introduce a new method for calculating the intrinsic and extrinsic geometry of the embedded two-surfaces. In section 3 we further develop the method to describe the geometry of spacelike two-surfaces embedded in Vaidya spacetimes and study how that geometry changes if the surfaces are deformed. As a test for our algorithm, we compare our results with exact results computed with the GRTensorII computer algebra package. In section 4 we specialize to FOTS in Vaidya spacetimes and study the finite extensions of FOTS to see how much we can distort FOTS. Conclusions are presented in section 5.

Chapter 2

Two-surfaces in Three-dimensional Euclidean Space

In this section, we study two-surfaces embedded in three-dimensional Euclidean space $\mathbb{R}^3$. We first review the characterization of a single embedded two-surface in an Euclidean space, in terms of its intrinsic and extrinsic geometry. Then we introduce a new method for calculating the geometric quantities of the two-surface.

A. Two-surface Geometry

To begin, we recall basic definitions and review the basic material from the theory of embedding of a two-surface in a three-dimensional Euclidean space.

Definition 1: A *covariant derivative* is a connection on the tangent bundle and other tensor bundles. Thus it has a certain behavior on functions, on vector fields, on the duals of vector fields (i.e., covector fields), and most generally of all, on arbitrary tensor fields:

(1) Functions

Given a function f, the covariant derivative $\nabla_v f$ coincides with the normal dif-

ferentiation of a real function in the direction of the vector v.

(2) Vector fields

A covariant derivative ∇ of a vector field u in the direction of the vector v denoted $\nabla_v u$ is defined by the following properties for any vector v, vector fields u, w and scalar functions f and g:

a. $\nabla_v u$ is algebraically linear in v so $\nabla_{fv+gw}u = f\nabla_v u + g\nabla_w u$

b. $\nabla_v u$ is additive in u so $\nabla_v(u+w) = \nabla_v u + \nabla_v w$

c. $\nabla_v u$ obeys the product rule, i.e. $\nabla_v fu = f\nabla_v u + u\nabla_v f$.

(3) Covector fields

Given a field of covectors (or one-form) α, its covariant derivative $\nabla_v \alpha$ can be defined using the following identity which is satisfied for all vector fields u

$$\nabla_v(\alpha(u)) = (\nabla_v \alpha)(u) + \alpha(\nabla_v u).$$

(4) Tensor fields

Once the covariant derivative is defined for fields of vectors and covectors it can be defined for arbitrary tensor fields using the following identities where φ and ψ are any two tensors:

$$\nabla_v(\varphi \otimes \psi) = (\nabla_v \varphi) \otimes \psi + \varphi \otimes (\nabla_v \psi),$$

and if φ and ψ are tensor fields of the same tensor bundle then

$$\nabla_v(\varphi + \psi) = \nabla_v \varphi + \nabla_v \psi.$$

Given coordinate functions x^i, $i = 1, 2, 3, \ldots$ and its basis $e_i = \frac{\partial}{\partial x^i}$, from the definition of covariant derivative, we find that for general vector fields $v = v^i e_i$ and $u = u^i e_i$ we get

$$\nabla_v u = (v^i u^j \Gamma^k_{ij} + v^i \frac{\partial u^k}{\partial x^i}) e_k,$$

where the coefficients Γ^k_{ij} are called Christoffel symbols [4]. In particular,

$$\nabla_{e_j} u = \nabla_j u = (\frac{\partial u^i}{\partial x^j} + u^k \Gamma^i{}_{jk}) e_i.$$

Definition 2: *Lie derivative*, named after Sophus Lie, is a derivation on the algebra of tensor fields over a manifold M. Generally, for a differentiable tensor field T of rank (p, q) and a differentiable vector field Y (i.e. a differentiable section of the tangent bundle TM), then we can define the Lie derivative of T along Y. Let $\varphi : M \times R \longrightarrow M$ be the one-parameter semigroup of local diffeomorphisms of M induced by the vector flow of Y and denote $\varphi_t(p) := \varphi(p, t)$. For each sufficiently small t, φ_t is a diffeomorphism from an neighborhood in M to another neighborhood in M, and φ_0 is the identity diffeomorphism. The Lie derivative of T is defined at a point p by

$$(\pounds_Y T)_p = \frac{d}{dt}\Big|_{t=0} ((\varphi_t)_* T_{\varphi_t(p)}),$$

where $(\varphi_t)_*$ is the pushforward along the diffeomorphism. In other words, if we have a tensor field T and an infinitesimal generator of a diffeomorphism given by a vector field Y, then $\pounds_Y T$ is nothing other than the infinitesimal change in T under the infinitesimal diffeomorphism.

Let S be a two-surface that is smoothly embedded in a three-dimensional Euclidean space $\mathbb{R}^3$ which is equipped with a metric $g_{ab} = diag[1, 1, 1]$. ∇_a denotes the covariant derivative compatible with g_{ab}.

The embedding of S in $\mathbb{R}^3$ can be defined in parametric form with,

$$x^a = X^a(\xi^\alpha), \tag{2.1}$$

$(\alpha, \beta = 1, 2)$, where x^a are coordinates on $\mathbb{R}^3$, ξ^α are coordinates on S, and X^a are the embedding functions. The two vectors $e_\alpha := X^a_{,\alpha} \partial_a$ form a basis of tangent vectors to S at each point of S. The metric induced on S is then given by,

$$q_{\alpha\beta} = X^a_{,\alpha} X^b_{,\beta} g_{ab} = g(e_\alpha, e_\beta). \tag{2.2}$$

It is also called the first fundamental form of S. Tangential indices are lowered and raised with $q_{\alpha\beta}$ and its inverse $q^{\alpha\beta}$, respectively.

The induced metric $q_{\alpha\beta}$ defines the unique covariant derivative d_α on S that is torsionless and satisfies

$$d_\alpha q_{\alpha\beta} = 0. \tag{2.3}$$

The Riemann tensor associated with d_α represents what can be called the intrinsic curvature of S. We shall denote its components by the letter R, as ${R^\alpha}_{\beta\mu\nu}$. ${R^\alpha}_{\beta\mu\nu}$ can be expressed as [10]

$${R^\alpha}_{\beta\mu\nu} = {\Gamma^\alpha}_{\beta\mu,\nu} - {\Gamma^\alpha}_{\beta\nu,\mu} + {\Gamma^\rho}_{\beta\mu}{\Gamma^\alpha}_{\rho\nu} - {\Gamma^\rho}_{\beta\nu}{\Gamma^\alpha}_{\rho\mu}, \tag{2.4}$$

where the Christoffel symbols are defined by

$$\Gamma_{\alpha\beta,\mu} = \frac{1}{2}(-q_{\alpha\beta,\mu} + q_{\alpha\mu,\beta} + q_{\beta\mu,\alpha}), \quad {\Gamma^\rho}_{\alpha\beta} = \sum_\mu \Gamma_{\alpha\beta,\mu} q^{\mu\rho}. \tag{2.5}$$

The corresponding Ricci tensor is denoted: $R_{\alpha\beta} = {R^\mu}_{\alpha\mu\beta}$ and the Ricci scalar (scalar curvature) is denoted: $R = q^{\alpha\beta}R_{\alpha\beta}$. From these we can see that the induced metric $q_{\alpha\beta}$ completely determines the intrinsic geometry of S.

The extrinsic geometry describes how S is embedded in $\mathbb{R}^3$. It corresponds to the change of direction of the normal vector as one moves on S. To describe the extrinsic geometry of S, we consider the unit normal vector n to the surface. The vector can be defined by the requirement that

$$g(n,n) = 1, \quad g(e_\alpha, n) = 0. \tag{2.6}$$

The induced metric can be expressed in terms of n [10]: $q_{ab} = g_{ab} - n_a n_b$ (here q_{ab} is actually an "extended" three-dimensional induced metric [8]).

The extrinsic curvature (the second fundamental form) of the surface is:

$$K_{\alpha\beta} \equiv g(e^a_\alpha \nabla_a e_\beta, n) = K_{\beta\alpha}. \tag{2.7}$$

We shall denote by K the trace of the extrinsic curvature $K_{\alpha\beta}$ with respect to the metric $q_{\alpha\beta}$

$$K \equiv q^{\alpha\beta} K_{\alpha\beta}. \tag{2.8}$$

From the knowledge of Weingarten map [10], K is equal to 2 times the mean curvature of S. We also note the relation

$$K = q^{ab} \nabla_a n^b, \tag{2.9}$$

which shows that K is equal to the expansion of a congruence of geodesics that intersect S orthogonally [11], thus it can be proven that K is equal to the rate of change of the area element (three-volume form on $\mathbb{R}^3$) of S. Here we outline the proof. We shall denote the area element of S by $\sqrt{q}$, where q is the determinant of q_{ab}. From Jacobi's Formula for the derivative of a determinant,

$$\begin{aligned} \pounds_n \sqrt{q} &= \frac{1}{2\sqrt{q}} \pounds_n q \\ &= \frac{1}{2\sqrt{q}} (q q^{ab} \pounds_n q_{ab}) \\ &= \frac{\sqrt{q}}{2} q^{ab} \pounds_n q_{ab}. \end{aligned}$$

From the definition of Lie derivatives, we rewrite this as

$$\frac{\sqrt{q}}{2} q^{ab} \pounds_n q_{ab} = \frac{\sqrt{q}}{2} q^{ab} (n^c \nabla_c q_{ab} + q_{ac} \nabla_b n^c + q_{cb} \nabla_a n^c).$$

Since

$$q^{ab} \nabla_c q_{ab} = q^{ab} \nabla_c (g_{ab} - n_a n_b) = 0,$$

then we have

$$\begin{aligned} \frac{\sqrt{q}}{2} q^{ab} \pounds_n q_{ab} &= \frac{\sqrt{q}}{2} (q^{ab} \nabla_b n_a + q^{ab} \nabla_a n_b) \\ &= \sqrt{q} q^{ab} \nabla_a n_b \\ &= \sqrt{q} K, \end{aligned}$$

from which it follows that

$$K = \frac{\pounds_n \sqrt{q}}{\sqrt{q}}.$$

We note that in this case, $\pounds_n \sqrt{q}$ is equal to the normal variation $\delta_n \sqrt{q}$ [5], thus we can rewrite K as

$$K = \frac{\delta_n \sqrt{q}}{\sqrt{q}}. \tag{2.10}$$

We have described the characterization of the surface S, in terms of its key intrinsic and extrinsic geometry. Next, we introduce a new method to calculate these geometric quantities. Let us consider an embedded surface S having parameterization: $S = [X(u,v),\ Y(u,v),\ Z(u,v)]$. Our method is based on discretizing S first and then computing its geometry at each point. After discretizing S, the tangent vectors and unit normal vectors to S at each point can be calculated. Then the induced metric, the area element and the Ricci scalar of S at each point can be calculated from these vectors. To calculate some variations of these geometric quantities, we consider the neighboring surface of S described by a deformation in the direction of the normal vector n:

$$S' = [X(u,v) + \varepsilon n^x,\ Y(u,v) + \varepsilon n^y,\ Z(u,v) + \varepsilon n^z], \tag{2.11}$$

where ε is a small parameter. We construct geometric quantities on S', calculate the Lie derivatives of the quantities and then pull-back the results onto S.

B. Numerical Implementation

Let us now use one concrete example to demonstrate our method. We consider a torus in the Euclidean space $\mathbb{R}^3$. The torus has parameterization

$$S = [(R + r\cos(u))\cos(v),\ (R + r\cos(u))\sin(v),\ r\sin(u)], \tag{2.12}$$

where $0 \leq u,\ v < 2\pi$, and $R = 5,\ r = 2$ are its two radii. The bulk of the following

part is the output from a Maple worksheet.

```
> with(LinearAlgebra):
> with(plots):
```

The first step is discretizing the surface S. We discretize S into $M \times N$ points and set $M = 60$, $N = 80$.

```
> u:=(i)-> i*2*Pi/M;
```

$$u := i \mapsto 2/M\, i\pi$$

```
> v:=(j)-> j*2*Pi/N;
```

$$v := j \mapsto 2/N\, j\pi$$

```
> M:= 60; N:= 80;
```

$$M := 60$$

$$N := 80$$

Then the embedding equations defined in parametric form at each point are given:

```
> x:=(i,j)->(5+2*cos(u(i)))*cos(v(j));
```

$$x := (i,j) \mapsto (5 + 2\cos(1/30\, i\pi))\cos(1/40\, j\pi)$$

```
> y:=(i,j)->(5+2*cos(u(i)))*sin(v(j));
```

$$y := (i,j) \mapsto (5 + 2\cos(1/30\, i\pi))\sin(1/40\, j\pi)$$

```
> z := (i,j)->2*sin(u(i));
```

$$z := (i,j) \mapsto 2\sin(1/30\, i\pi)$$

Input the three-dimensional metric of the Euclidean space $\mathbb{R}^3$ and the parameterization of S:

```
> G := Matrix(3, 3, [[1, 0, 0], [0,1,0], [0, 0, 1]]);
```

$$G := \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

```
> X:= [seq([seq([evalf(x(i, j)), evalf(y(i,j)), evalf(z(i, j))], j =
1 .. N+2)], i = 1 .. M+2)]:
```

Next, calculate the two tangent vectors, $\frac{\partial}{\partial u} = [\frac{\partial x}{\partial u}, \frac{\partial y}{\partial u}, \frac{\partial z}{\partial u}]$ and $\frac{\partial}{\partial v} = [\frac{\partial x}{\partial v}, \frac{\partial y}{\partial v}, \frac{\partial z}{\partial v}]$, at each point of S. We use the forward difference formula, $f'(x_0) \approx \frac{f(x_0+h)-f(x_0)}{h}$, to approximate the derivative of $f(x_0)$. Forward difference formulas are discussed detailed in Appendix A.

```
> V1 := [evalf(seq([seq(1/2*(X[i+1, j]-X[i,j])*M/Pi, j = 1 .. N+1)],
i = 1 .. M+1))]:
> V2 := [evalf(seq([seq(1/2*(X[i, j+1]-X[i,j])*N/Pi, j = 1 .. N+1)],
i = 1 .. M+1))]:
```

Given the tangent vectors, the unit normal vector to S at each point of S is $n[i,j] = \frac{V1[i,j]\times V2[i,j]}{\|V1[i,j]\times V2[i,j]\|}$. In the code we set the data type of the vectors $X, V1, V2, n, ...$ as lists of elements instead of vectors, which greatly reduces the CPU execution time and lets the worksheet run much faster although it needs us to do some calculation by ourselves and makes the expressions look complicated.

```
> n :=[seq([seq([V1[i,j,2]*V2[i,j,3]-V1[i,j,3]*V2[i, j, 2], -V1[i, j,
1]*V2[i, j,3]+V1[i, j, 3]*V2[i, j,1],V1[i, j, 1]*V2[i, j, 2]-V1[i, j,
2]*V2[i,j, 1]]/((V1[i, j,2]*V2[i,j, 3]-V1[i, j, 3]*V2[i, j,2])^2+(-V1
[i, j,1]*V2[i, j,3]+V1[i, j, 3]*V2[i, j,1])^2+(V1[i, j, 1]*V2[i,j,2]-
V1[i, j, 2]*V2[i, j,1])^2)^(1/2), j = 1..N+1)], i = 1 ..  M+1)]:
```

We now calculate the components of the two-dimensional induced metric of S from the tangent vectors, $q_{\alpha\beta} = X^a_{,\alpha} X^b_{,\beta} g_{ab}$.

```
> q11 := [evalf(seq([seq(V1[i, j,1]^2*G[1, 1]+V1[i, j, 2]^2*G[2, 2]+V1[i,
j,3]^2*G[3, 3], j = 1 ..  N+1)], i = 1 ..  M+1))]:

> q12 := [evalf(seq([seq(V1[i, j, 1]*V2[i,j, 1]*G[1, 1]+V1[i, j, 2]*V2[i,
j, 2]*G[2, 2]+V1[i, j, 3]*V2[i, j,3]*G[3, 3], j = 1 ..  N+1)], i = 1
..  M+1))]:

> q22 := [evalf(seq([seq(V2[i, j,1]^2*G[1, 1]+V2[i, j, 2]^2*G[2, 2]+V2[i,
j,3]^2*G[3, 3], j = 1 ..  N+1)], i = 1 ..  M+1))]:
```

Then the area element of S can be calculated, $A1 = \sqrt{q}$, where q is the determinant of the induced metric $q_{\alpha\beta}$.

```
> A1 := [seq([seq((q11[i, j]*q22[i,j]-q12[i, j]^2)^(1/2), j = 1 ..  N+1)],
i = 1 ..M+1)]:
```

From the Formula (2.5) we can deduce the Ricci Scalar of S from its two-dimensional metric $q_{\alpha\beta}$,

$$R = \frac{1}{2}(\mathrm{q11}(u,\,v)\,(\tfrac{\partial}{\partial u}\,\mathrm{q22}(u,\,v))^2 - 2\,(\tfrac{\partial}{\partial u}\,\mathrm{q11}(u,\,v))\,\mathrm{q22}(u,\,v)$$
$$- 2\,\mathrm{q11}(u,\,v)\,(\tfrac{\partial}{\partial u}\,\mathrm{q12}(u,\,v))\,(\tfrac{\partial}{\partial v}\,\mathrm{q22}(u,\,v)) - 2\,\mathrm{q11}(u,\,v)\,(\tfrac{\partial^2}{\partial v^2}\,\mathrm{q11}(u,\,v))\,\mathrm{q22}(u,\,v)$$
$$+ (\tfrac{\partial}{\partial v}\,\mathrm{q11}(u,\,v))^2\,\mathrm{q22}(u,\,v) + 4\,\mathrm{q11}(u,\,v)\,(\tfrac{\partial^2}{\partial v\,\partial u}\,\mathrm{q12}(u,\,v))\,\mathrm{q22}(u,\,v)$$
$$+ 2\,\mathrm{q12}(u,\,v)\,(\tfrac{\partial}{\partial v}\,\mathrm{q12}(u,\,v)) + \mathrm{q12}(u,\,v)\,(\tfrac{\partial}{\partial u}\,\mathrm{q11}(u,\,v))\,(\tfrac{\partial}{\partial v}\,\mathrm{q22}(u,\,v))$$
$$+ (\tfrac{\partial}{\partial u}\,\mathrm{q11}(u,\,v))\,(\tfrac{\partial}{\partial u}\,\mathrm{q22}(u,\,v))\,\mathrm{q22}(u,\,v) - 2\,\mathrm{q11}(u,\,v)\,(\tfrac{\partial^2}{\partial u^2}\,\mathrm{q22}(u,\,v))\,\mathrm{q22}(u,\,v)$$
$$+ \mathrm{q11}(u,\,v)\,(\tfrac{\partial}{\partial v}\,\mathrm{q11}(u,\,v))\,(\tfrac{\partial}{\partial v}\,\mathrm{q22}(u,\,v)) - \mathrm{q12}(u,\,v)\,(\tfrac{\partial}{\partial v}\,\mathrm{q11}(u,\,v))\,(\tfrac{\partial}{\partial u}\,\mathrm{q22}(u,\,v))$$
$$- 2\,\mathrm{q12}(u,\,v)\,(\tfrac{\partial}{\partial u}\,\mathrm{q12}(u,\,v)))\,\Big/\,(\mathrm{q11}(u,\,v)\,\mathrm{q22}(u,\,v) - \mathrm{q12}(u,\,v)^2)^2$$

We now use the formula above to calculate the Ricci Scalar at each point of S. Let $U11$, $U12$, $U22$, $V11$, $V12$, $V22$, $VV11$, $VU12$, $UU22$ denote $\frac{\partial}{\partial u}q11$, $\frac{\partial}{\partial u}q12$, $\frac{\partial}{\partial u}q22$, $\frac{\partial}{\partial v}q11$, $\frac{\partial}{\partial v}q12$, $\frac{\partial}{\partial v}q22$, $\frac{\partial^2}{\partial v^2}q11$, $\frac{\partial^2}{\partial v\partial u}q12$, $\frac{\partial^2}{\partial u^2}q22$ respectively. First, calculate $U11$, $U12$, $U22$, $V11$, $V12$, $V22$, $VV11$, $VU12$, $UU22$ using the forward difference formula.

```
> U11 := [evalf(seq([seq(1/2*(q11[i+1,j]-q11[i, j])*M/Pi, j = 1 ..N)],
i = 1 .. M))]:
> U12 := [evalf(seq([seq(1/2*(q12[i+1,j]-q12[i, j])*M/Pi, j = 1 ..N)],
i = 1 .. M))]:
> U22 := [evalf(seq([seq(1/2*(q22[i+1,j]-q22[i, j])*M/Pi, j = 1 ..N)],
i = 1 .. M))]:
> V11 := [evalf(seq([seq(1/2*(q11[i,j+1]-q11[i, j])*N/Pi, j = 1 ..N)],
i = 1 .. M))]:
> V12 := [evalf(seq([seq(1/2*(q12[i,j+1]-q12[i, j])*N/Pi, j = 1 ..N)],
i = 1 .. M))]:
> V22 := [evalf(seq([seq(1/2*(q22[i,j+1]-q22[i, j])*N/Pi, j = 1 ..N)],
i = 1 .. M))]:
> VV11 := [evalf(seq([seq(1/2*(V11[i,j+1]-V11[i, j])*N/Pi, j = 1 ..N-1)],
i = 1 .. M-1))]:
> VU12 := [evalf(seq([seq(1/2*(U12[i,j+1]-U12[i, j])*N/Pi, j = 1 ..N-1)],
i = 1 .. M-1))]:
> UU22 := [evalf(seq([seq(1/2*(U22[i+1,j]-U22[i, j])*M/Pi, j = 1 ..N-1)],
i = 1 .. M-1))]:
```

Then substitute $U11$, $U12$, $U22$, $V11$, $V12$, $V22$, $VV11$, $VU12$, $UU22$ into the formula above to calculate the Ricci scalar at each point of S and draw its figure.

```
> R := [evalf(seq([seq(1/2*(q11[i, j]*U22[i,j]^2-2*V12[i, j]*U11[i,
j]*q22[i, j]-2*q11[i, j]*U12[i,j]*V22[i, j]+2*VV11[i, j]*q12[i, j]^2
-2*q11[i, j]*VV11[i,j]*q22[i, j]+V11[i, j]^2*q22[i, j]-4*VU12[i, j]*
q12[i,j]^2+4*q11[i, j]*VU12[i, j]*q22[i, j]+4*U12[i, j]*q12[i,j]*V12[i,
j]+q12[i, j]*U11[i, j]*V22[i, j]+U11[i, j]*U22[i,j]*q22[i, j]-2*V11[i,
j]*q12[i, j]*V12[i, j]+2*UU22[i, j]*q12[i,j]^2-2*q11[i, j]*UU22[i, j]*
q22[i, j]+q11[i, j]*V11[i,j]*V22[i, j]-V11[i, j]*q12[i, j]*U22[i, j]-2
*U22[i, j]*q12[i,j]*U12[i, j])/(q11[i, j]*q22[i, j]-q12[i,j]^2)^2, j=
1 .. N-1)], i = 1 .. M-1))]:
> listplot3d([seq([seq(R[i, j], j = 1..N-1)], i = 1 .. M-1)],labels=[i,
j, 'R'] axes= boxed, orientation= [43, 70]);
```

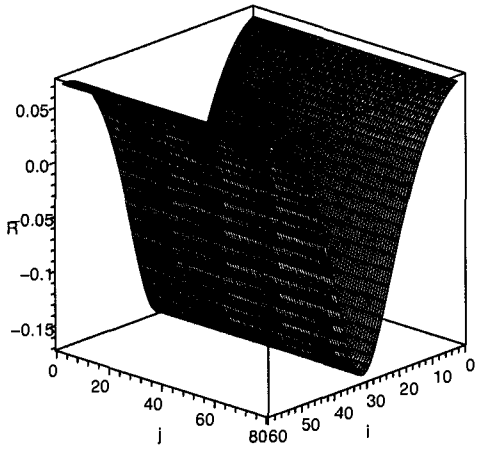


Figure 2.1: The Ricci Scalar, R, calculated by the numerical method.

We now consider the extrinsic geometry of S. In particular, we are interested

in K, the trace of the extrinsic curvature. To calculate K, we first construct a neighboring surface of S described by a normal vector of V. Let ε be a small parameter, p is a point of S. Displace the point p by the normal vector εV to the point p'. Do the same for each point of S, keeping the value of ε fixed. Then, all the points p' define a new surface:

$$S' = S + \varepsilon V,$$

which is depicted schematically in Figure 2.2. Finally, from Formula (2.10) K can be calculated as the fractional rate of change of the area element,

$$K = \lim_{\varepsilon \to 0} \frac{1}{A} \frac{A' - A}{\varepsilon},$$

where A and A' are the area elements of S and S' respectively.

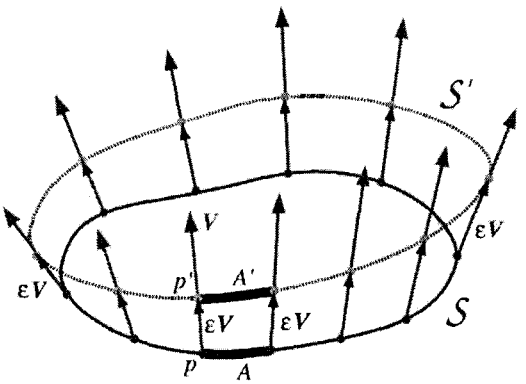


Figure 2.2: A schematic of the surface S and its neighboring surface S'.

Using the method above, we first construct the surface S' and calculate the tangent vectors to S':

```
> eps:= 0.01;
```

$$eps := 0.01$$

```
> X2 := [seq([seq([X[i, j, 1]+eps*n[i, j,1], X[i, j, 2]+eps*n[i, j,
2], X[i, j, 3]+eps*n[i, j, 3]], j = 1 ..N+1)], i = 1 .. M+1)]:
> D1 := [evalf(seq([seq(1/2*(X2[i+1,j]-X2[i, j])*M/Pi, j = 1 .. N)],
i = 1 .. M))]:
> D2 := [evalf(seq([seq(1/2*(X2[i,j+1]-X2[i, j])*N/Pi, j = 1 .. N)],
i = 1 .. M))]:
```

Next, calculate the two-dimensional induced metric and the area element of S' using the same method as on the initial surface S:

```
> d11 := [evalf(seq([seq(D1[i, j,1]^2*G[1, 1]+D1[i, j, 2]^2*G[2, 2]+D1[i,
j,3]^2*G[3, 3], j = 1 .. N)], i = 1 .. M))]:
> d12 := [evalf(seq([seq(D1[i, j, 1]*D2[i,j, 1]*G[1, 1]+D1[i, j, 2]*D2[i,
j, 2]*G[2, 2]+D1[i, j, 3]*D2[i, j,3]*G[3, 3], j = 1 .. N)], i = 1 ..
M))]:
> d22 := [evalf(seq([seq(D2[i, j,1]^2*G[1, 1]+D2[i, j, 2]^2*G[2, 2]+D2[i,
j,3]^2*G[3, 3], j = 1 .. N)], i = 1 .. M))]:
> A2 := [seq([seq(sqrt(d11[i, j]*d22[i,j]-d12[i, j]^2), j = 1 .. N)],
i = 1 .. M)]:
```

Finally, calculate K and depict it in Figure 2.3:

```
> K := [seq([seq((A2[i, j]-A1[i,j])/(eps*A1[i, j]), j = 1 .. N)], i=
1 .. M)]:
> listplot3d([seq([seq(K[i, j], j = 1 ..N)], i = 1 .. M)], axes = boxed,
orientation = [43, 70]);
```

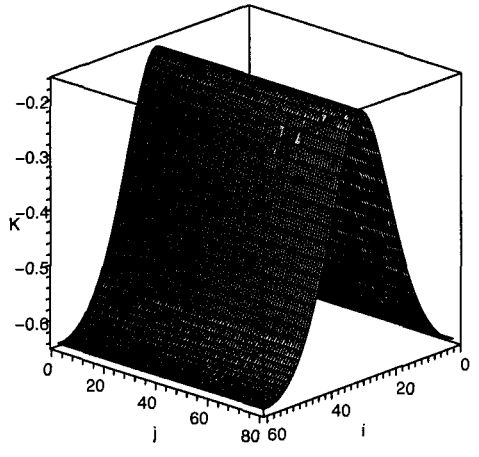


Figure 2.3: The trace of the extrinsic curvature, K, calculated by the numerical method.

As a test for our algorithm we now calculate R and K by another method. From the knowledge of differential geometry, we know that $R = \frac{\tilde{K}}{q}$ [11], where $\tilde{K}$ is the determinant of the second fundamental form $K_{\alpha\beta}$ and q is the determinant of the two-dimensional metric $q_{\alpha\beta}$. In this case for the surface $S = [(R + r\cos(u))\cos(v),\ (R + r\cos(u))\sin(v),\ r\sin(u)]$,

$$q_{\alpha\beta} = \begin{bmatrix} 4 & 0 \\ 0 & (5 + 2\cos(u))^2 \end{bmatrix},$$

$$K_{\alpha\beta} = \begin{bmatrix} 2 & 0 \\ 0 & (5 + 2\cos(u))\cos(u) \end{bmatrix}.$$

So

$$\frac{\tilde{K}}{q} = 0.5\cos(u)(5 + 2\cos(u))/(5 + 2\cos(u))^2. \tag{2.13}$$

Then we can directly calculate that for the surface S the trace of the Weingarten

Map $q^{\alpha\beta}K_{\alpha\beta}$, i.e. K, is

$$-1/2 - \cos(u)/(5 + 2\cos(u)). \tag{2.14}$$

The results of the expressions (2.13) and (2.14) are plotted in Figures 2.4 and 2.5 respectively. The two results of R and K are plotted in the same figure (Figure 2.6), respectively, for comparison. Furthermore, the differences between the two results of R and K are plotted respectively in Figure 2.7. Figures 2.6 and 2.7 show that the two results of R and K both match well. Thus our code recovers the correct value of R and K.

```
> plot3d(1/2*csgn(5+2*cos(u))^2*(5+2*cos(u))*cos(u)/(5+2*cos(u))^2,
> u = 0 .. 2*Pi, v = 0 .. 2*Pi, axes = boxed, orientation = [43,
> 70]);
```

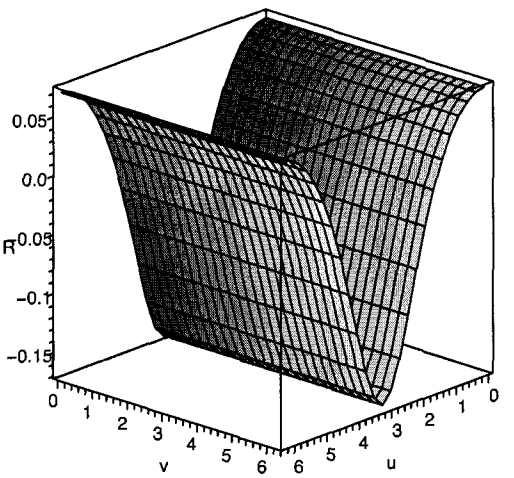


Figure 2.4: The Ricci scalar, R, calculated by algebra

```
> plot3d(-1/2*csgn(5+2*cos(u))-csgn(5+2*cos(u))*cos(u)/(5+2*cos(u)),
u = 0 .. 2*Pi, v = 0 .. 2*Pi, axes = boxed, orientation =[43,70]);
```

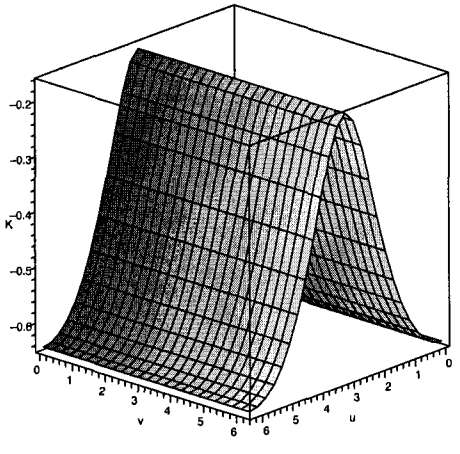
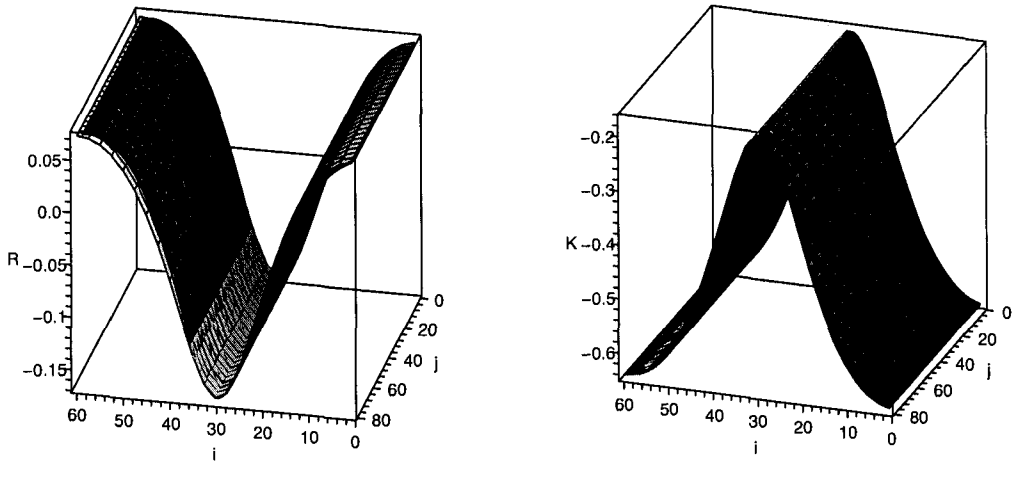


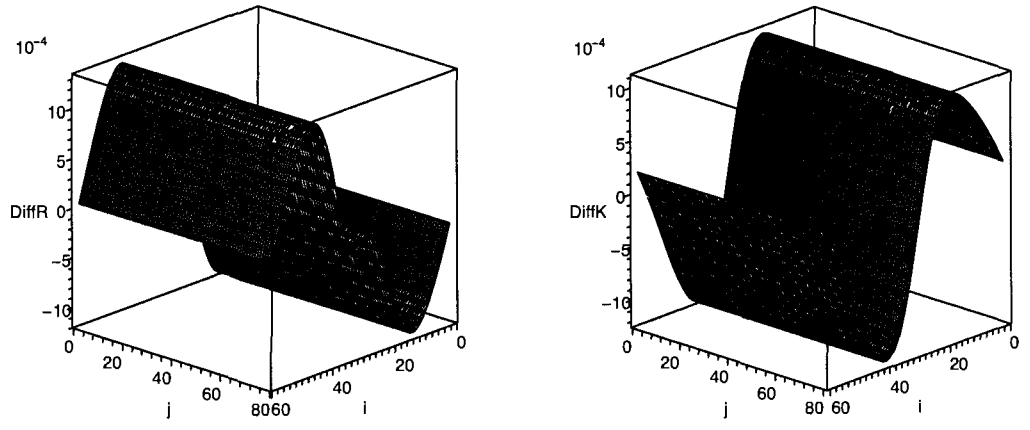
Figure 2.5: The trace of the extrinsic geometry, K, calculated by algebra



(a) The two results of R

(b) The two results of K

Figure 2.6: The two results of R and K are plotted together for comparison



(a) The difference between the two results of R (b) The difference between the two results of K

Figure 2.7: The differences between the two results of R and K.

Chapter 3

Two-surfaces in Vaidya Spacetimes

We have introduced a method for calculating the geometry of a two-surface embedded in a three-dimensional manifold, which allows the analysis to be carried out in a direct manner. In this section we further develop the method to study the geometry and deformations of spacelike two-surfaces embedded in four-dimensional spacetimes. In particular we apply the method to calculate the geometry of spacelike two-surfaces embedded in Vaidya spacetime and their variations. We test our algorithm by comparing our results with exact results obtained with the GRTensorII computer algebra package.

A. Two-surface Geometry

Let S be a closed (i.e. compact without boundary), orientable spacelike 2-surface that is smoothly embedded a 4-dimensional spacetime (M, g_{ab}). ∇_a denotes the covariant derivative compatible with g_{ab}. The embedding of S in M can be defined

in parametric form with,

$$x^a = X^a(\xi^\alpha), \tag{3.1}$$

$(\alpha, \beta = 1, 2)$, where x^a are coordinates on M, ξ^a are coordinates on S, and X^a are the embedding functions. The two vectors $e_\alpha := X^a_{,\alpha}\partial_a$ form a basis of tangent vectors to S at each point of S. The metric induced by the spacetime metric g_{ab} onto S is then given by,

$$q_{\alpha\beta} = X^a_{,\alpha}X^b_{,\beta}g_{ab} = g(e_\alpha, e_\beta). \tag{3.2}$$

This metric defines the compatible derivative operator d_a, the area element and the Ricci scalar on the two-surface.

The set of vectors orthogonal to S at any point form a 2-dimensional Minkowskian vector space. Then there are two linearly independent, future-pointing null vectors normal to S, l^a and n^a. We usually construct a future directed unit timelike normal T^a and the unit outward pointing spacelike normal R^a to S first, then we can define the outgoing and ingoing null vectors

$$l^a := \frac{1}{\sqrt{2}}(T^a + R^a), \tag{3.3}$$

$$n^a := \frac{1}{\sqrt{2}}(T^a - R^a). \tag{3.4}$$

If we further require that $g_{ab}l^a n^b = -1$ then the null vectors are specified only up to one degree of rescaling freedom

$$l^a \longrightarrow f l^a, \quad n^a \longrightarrow f^{-1}n^a, \tag{3.5}$$

where f is a, positive definite (to preserve the future orientation), smooth function on S. The induced metric can be expressed in terms of the null vectors (here it is actually an "extended" four-dimensional induced metric [12]):

$$q_{ab} = g_{ab} + l_a n_b + l_b n_a. \tag{3.6}$$

The extrinsic curvatures of the surface are [11]:

$$K^{(n)}_{ab} \equiv q^c_a q^d_b \nabla_c n_d, \quad K^{(l)}_{ab} \equiv q^c_a q^d_b \nabla_c l_d, \tag{3.7}$$

which describe the way in which S is embedded in M. The null expansions are:

$$\theta_{(n)} = q^{ab}\nabla_a n_b, \quad \theta_{(l)} = q^{ab}\nabla_a l_b. \tag{3.8}$$

These expansions tell us how the area element of S changes as it is deformed along l^a and n^a respectively. They are very important quantities, like the K in section 2; we will encounter them often in the remaining sections of this paper. From (2.9) we know that the expansions are the rates of change of the area element of S, thus we can rewrite them as

$$\theta_{(n)} = \frac{\pounds_n \sqrt{q}}{\sqrt{q}}, \quad \theta_{(l)} = \frac{\pounds_l \sqrt{q}}{\sqrt{q}}, \tag{3.9}$$

where q is the determinant of the metric q_{ab}. In our code, for convenience, we will use the formula (3.9) to calculate the expansions.

We now consider the computation of the Lie derivatives $\pounds_l \sqrt{q}$ and $\pounds_n \sqrt{q}$. Our numerical approach to this problem is based on deforming the surface S, which is also critical for the computation of the deformations of the geometry of S that we will consider later. We have used this method to calculate K in the three-dimensional case in Section 2. Here we demonstrate the method in detail. Let us begin by recalling the definition of deformations [6].

A *deformation (variation)* of a two-surface S_0 is a smooth, one-to-one function $\Phi(s, \lambda) : S_0 \times [-\lambda_0, \lambda_0] \to M$ (with λ_0 some real number) such that $\Phi(S_0, 0) = S_0$. Thus, Φ generates a finite three-surface $\mathcal{T}_\Phi$ and that surface is foliated by images $S_\lambda = \Phi(S_0, \lambda)$ of S_0 as depicted in Figure 3.1. The *deformation vector* field $X^a = (\frac{\partial}{\partial \lambda})^a$ is tangent to the curves of constant $s \in S$. The flow generated by this vector field maps leaves of constant λ into each other. For sufficiently small ϵ, we can intuitively

write the neighboring surface of S_0 as

$$S_\epsilon = S_0 + \epsilon X. \tag{3.10}$$

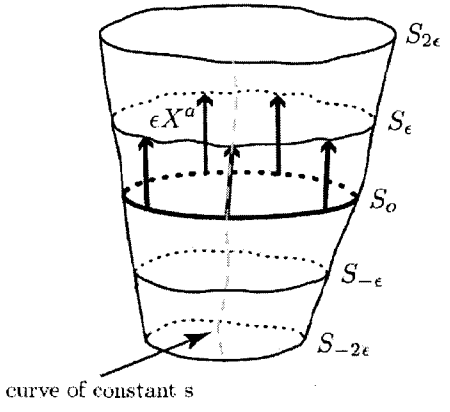


Figure 3.1: A schematic of a section of $\mathcal{T}_\Phi$ around S_0 [6].

We will focus on *normal deformations* where X^a is everywhere perpendicular to the S_λ and so can be written:

$$X^a = Al^a - Bn^a, \tag{3.11}$$

for some functions A and B.

Now we can calculate the Lie derivatives $\pounds_l\sqrt{q}$ and $\pounds_n\sqrt{q}$ and then the expansions. Let us take $\theta_{(l)}$ as an example ($\theta_{(n)}$ can be calculated in the same way). For any value of A and B, the map Φ deforms S_0 into successive surfaces S_λ. First, using Φ we deform the surface S so that $S \to S + \epsilon l$. Then we contruct the area element $\sqrt{q}_\epsilon$ on the new surface S_ϵ, and we have

$$\pounds_l\sqrt{q} = \lim_{\epsilon\to 0}\frac{\sqrt{q}_\epsilon - \sqrt{q}}{\epsilon}, \tag{3.12}$$

Finally, from (3.9) we have

$$\theta_{(l)} = \lim_{\epsilon \to 0} \frac{\frac{\sqrt{q_\epsilon} - \sqrt{q}}{\epsilon}}{\sqrt{q}}. \tag{3.13}$$

For accurateness, in our code we do the same thing for $S \to S - \epsilon l$, and then we average these two results to approximate the rate of change. This method is similar to the "centered difference formula" which is used to calculate the derivative of a function (see Appendix A). We will call such method as "three-point estimate", and we will use it frequently in our code.

We now move on to calculate the deformations for some geometric quantities of S. In particular we are interested in the deformations of the expansions which are important for the dynamics of two-surfaces embedded in spacetimes. The method for calculating the deformations of the expansions is similar to the one we used above to calculate the expansions themselves. Taking the null expansion $\theta_{(l)}$ as an example again, we write its deformation as $\delta_X \theta_{(l)}$ and note that calculating this quantity amounts to:

1. using Φ to deform the initial surface S so that $S \to S + \epsilon X$.
2. constructing the $\theta_{(l)}$ on the new surface S_ϵ.
3. calculating the Lie derivative $\pounds_X \theta_{(l)}$ and pulling-back the results onto S.

In this paper we are mainly interested in spacelike two-surfaces embedded in the Vaidya spacetime. The Vaidya spacetime describes spherically symmetric collapse of null dust [13]. In ingoing Eddington-Finkelstein coordinates (v, r, θ, ϕ), the metric is

$$dS^2 = -(1 - \frac{2m(v)}{r})dv^2 + 2dvdr + r^2 d\theta^2 + r^2 sin^2\theta d\phi^2, \tag{3.14}$$

where the mass function $m(v)$ can be specified as a non-negative, non-decreasing, smooth function of the null coordinate v. The stress energy is determined by the derivative of $m(v)$:

$$T_{ab} = \frac{m'(v)}{4\pi r^2} \partial_a v \partial_b v. \tag{3.15}$$

For constant $m(v)$, the metric is just the standard Schwarzschild metric in ingoing Eddington-Finkelstein coordinates.

We now calculate the intrinsic and extrinsic geometry and their deformations for a spacelike two-surface in the Vaidya spacetime, using the method we discussed previously. Consider a two-surface S in the Vaidya spacetime:

$$S = \{v = V(\lambda),\ r = R(\lambda),\ \theta = \Theta(\lambda),\ \phi = \phi;\ \ 0 \leq \lambda \leq 1,\ -\pi \leq \phi \leq \pi\}. \qquad (3.16)$$

Generally, our approach to this problem is similar to the one used in the three-dimensional case. First, we discretize S and calculate the tangent vectors and null normal vectors at each point. We note that constructing the two future-pointing null normals to S is much more difficult than calculating the unit normal vectors in the three-dimensional case. Then we can calculate the induced metric, area element and Ricci scalar of S at each point in the same way as in Section 2. The expansions of S and their variations are calculated by constructing the neighboring surfaces described by the normal deformations, which have been discussed previously.

B. Numerical Implementation

As an example, let us now illustrate our method with the Maple worksheet below which calculates the various geometric quantities for a given two-surface in the Vaidya spacetime. This Maple worksheet mainly consists of some procedures which make the code concise and efficient.

To begin, we define a procedure to calculate the tangent vectors to any surface which is input as X. When we calculate the derivative of a function, to improve the accuracy, we use the fourth-order accurate five-point difference formula (see Appendix A for details), instead of the former first-order accurate formula. In addition, for the

points at center, we use the centered five-point formula:

$$f'(x_0) \approx \frac{f(x_0-2h) - 8f(x_0-h) + 8f(x_0+h) - f(x_0+2h)}{12h}; \tag{3.17}$$

for the points at the left edge, we use the forward five-point formula:

$$f'(x_0) \approx \frac{-25f(x_0) + 48f(x_0+h) - 36f(x_0+2h) + 16f(x_0+3h) - 3f(x_0+4h)}{12h}; \tag{3.18}$$

and for the points at the right edge, we use the backward five-point formula:

$$f'(x_0) \approx \frac{-25f(x_0) + 48f(x_0-h) - 36f(x_0-2h) + 16f(x_0-3h) - 3f(x_0-4h)}{-12h}. \tag{3.19}$$

```
> with(plots):
> FivePtDer:=proc(X,N)
> local h,DX1,DX2,DXMain;
> h:=1/N;
> DX1:=[evalf(seq((-25*X[i]+48*X[i+1]-36*X[i+2]+16*X[i+3]-3*X[i+4])/(12*h),
> i = 1 .. 2))];
> DXMain := [evalf(seq(((X[i-2]-8*X[i-1])+(8*X[i+1]-X[i+2]))/(12*h),
> i = 3 .. N-1))];
> DX2:=[evalf(seq((-25*X[i]+48*X[i-1]-36*X[i-2]+16*X[i-3]-3*X[i-4])/(-12*h)
> i = N .. N+1))];
> return([op(DX1),op(DXMain),op(DX2)]);
> end proc:
```

Next, calculate the diagonal components of the two-dimensional induced metric from the tangent vectors and then the area element. A quick calculation shows that the components $q12$ and $q21$ are zero, so they are neglected in the code.

```
> AreaEl:=proc(X,Xp,m,N)
> local q11, q22, dA;
> q11:=[seq(-(1-2*m[i]/X[i,2])*Xp[i,1]^2+2*Xp[i,1]*Xp[i,2]+X[i,2]^2*Xp[i
> ,3]^2,i=1..N+1)];
> q22:=[seq(X[i,2]^2*sin(X[i,3])^2,i=1..N+1)];
> dA:=[seq(evalf(sqrt(q11[i]*q22[i])),i=1..N+1)];
> return(dA);
> end proc:
```

Then calculate the Ricci scalar. From (3.16) the metric of the Vaidya spacetime can be rewritten as

$$dS^2 = \{-(1 - \frac{2m(v)}{R})V'^2 + 2V'R' + R^2\Theta'^2\}d\lambda^2 + \{R^2 \sin^2\Theta\}d\phi^2. \tag{3.20}$$

Denote $-(1-\frac{2m(v)}{R})V'^2+2V'R'+R^2\Theta'^2$ and $R^2\sin^2\Theta$ by F and G, respectively. Then from (2.5) the Ricci scalar can be expressed as

$$R = \frac{-1}{\sqrt{FG}}\frac{d}{d\lambda}(\frac{G'}{\sqrt{FG}}). \tag{3.21}$$

```
> Ricci:=proc(X,Xp,m,N)
> local F,G,Gp,PDerTerm,DerTerm,dA,R;
> F:=[seq(-(1-2*m[i]/X[i,2])*Xp[i,1]^2+2*Xp[i,1]*Xp[i,2]+X[i,2]^2*Xp[i,3
> ]^2,i=1..N+1)];
> G:=[seq(X[i,2]^2*sin(X[i,3])^2,i=1..N+1)];
> Gp:=FivePtDer(G,N);
> dA:=[seq(evalf(sqrt(F[i]*G[i])),i=1..N+1)];
```

```
> PDerTerm:=[seq(Gp[i]/dA[i],i=1..N+1)];
> DerTerm:=FivePtDer(PDerTerm,N);
> R:=[seq(-DerTerm[i]/dA[i],i=1..N+1)];
> return(R);
> end proc:
```

The following part calculates the null normal vectors to S. This works by first calculating a canonical pair of timelike and spacelike unit vectors and then constructing null vectors from them.

First, from the two tangent vectors of the 2-surface S, $\frac{\partial}{\partial\lambda} = V'\frac{\partial}{\partial v} + R'\frac{\partial}{\partial r} + \Theta'\frac{\partial}{\partial\theta}$ and $\frac{\partial}{\partial\phi} = \frac{\partial}{\partial\phi}$, we can obtain the normal one-forms:

$$n_1 = \Theta' dr - R' d\theta, \quad n_2 = -\Theta' dv + V' d\theta. \tag{3.22}$$

Then, with the inverse metric of the Vaidya spacetime we convert the normal one-forms to normal vectors:

$$\vec{n}_1 = \Theta'\frac{\partial}{\partial v} + \Theta'(1 - \frac{2m(v)}{R})\frac{\partial}{\partial r} - \frac{R'}{R^2}\frac{\partial}{\partial\theta}, \quad \vec{n}_2 = -\Theta'\frac{\partial}{\partial r} + \frac{V'}{R^2}\frac{\partial}{\partial\theta}. \tag{3.23}$$

It follows that any linear combination, $\vec{n}_1 - \alpha\vec{n}_2$, where α is a parameter function, is normal to the surface S.

Now we can calculate the spacelike unit normal. We know that the vectors $\vec{n}_1 - \alpha\vec{n}_2$ will be spacelike if $(\vec{n}_1 - \alpha\vec{n}_2)\cdot(\vec{n}_1 - \alpha\vec{n}_2)$ is positive. Since

$$(\vec{n}_1 - \alpha\vec{n}_2)\cdot(\vec{n}_1 - \alpha\vec{n}_2) = \Theta'^2(2\alpha + 1 - \frac{2m}{R}) + \frac{(R' + \alpha V')^2}{R^2}, \tag{3.24}$$

we note that if we choose $\alpha = \frac{2m}{R}$, the directions of these vectors are not affected, and the calculations will become simple. In this case,

$$(\vec{n}_1 - \alpha\vec{n}_2)\cdot(\vec{n}_1 - \alpha\vec{n}_2) = \Theta'^2(1 + \frac{2m}{R}) + \frac{(R'R + 2V'm)^2}{R^4} > 0, \tag{3.25}$$

so the vectors $\vec{n}_1 - \alpha\vec{n}_2$ will be spacelike. Then the outward pointing spacelike unit normal can be defined as (we would usually expect $\Theta' > 0$ for the surface S)

$$\vec{R} \equiv \frac{\vec{n}_1 - \alpha\vec{n}_2}{\| \vec{n}_1 - \alpha\vec{n}_2 \|} = \frac{1}{\sqrt{(1+\frac{2m}{R})\Theta'^2 + \frac{(RR'+2mV')^2}{R^4}}}\{\Theta'\frac{\partial}{\partial v} + \Theta'\frac{\partial}{\partial r} - \frac{RR'+2mV'}{R^3}\frac{\partial}{\partial\theta}\}. \tag{3.26}$$

Next we need to construct a timelike normal that is perpendicular to $\vec{R}$. Let us consider $An_1 + Bn_2$, where A and B are two functions we need to solve. We need to solve the equation

$$An_1(\vec{R}) + Bn_2(\vec{R}) = 0. \tag{3.27}$$

It is easy to check that

$$A = -n_2(\vec{R}), \quad B = n_1(\vec{R}) \tag{3.28}$$

will do this. Thus we obtain the required normal vector:

$$\vec{t} = -n_2(\vec{R})\vec{n}_1 + n_1(\vec{R})\vec{n}_2. \tag{3.29}$$

Then with $\alpha = \frac{2m}{R}$ we can define the unit timelike normal:

$$\vec{T} = \frac{\vec{t}}{\sqrt{-\vec{t}\cdot\vec{t}}}. \tag{3.30}$$

Finally, we define the two null normal vectors:

$$\vec{l} = \frac{1}{\sqrt{2}}(\vec{T} + \vec{R}), \tag{3.31}$$

$$\vec{n} = \frac{1}{\sqrt{2}}(\vec{T} - \vec{R}). \tag{3.32}$$

Code for these procedures follows:

```
> NullVec:=proc(X,Xp,m,N)
> local Rden, Tden, vR, n1, n2, n1vR, n2vR, vn1, vn2, prevT, prevT2,
> vT, vL, vN;
> Rden:=[seq(sqrt((1+2*m[i]/X[i,2])*Xp[i,3]^2+(X[i,2]*Xp[i,2]+2*m[i]*Xp[
> i,1])^2/X[i,2]^4),i=1..N+1)];
> vR:=[seq([Xp[i,3]/Rden[i],Xp[i,3]/Rden[i],-(X[i,2]*Xp[i,2]+2*m[i]*Xp[i
> ,1])/X[i,2]^3/Rden[i]],i=1..N+1)];
> Tden:=[seq(Rden[i]*sqrt(Xp[i,3]^2+2*Xp[i,1]*Xp[i,2]/X[i,2]^2-(1-2*m[i]
> /X[i,2])*Xp[i,1]^2/X[i,2]^2),i=1..N+1)];
> n1:=[seq([0,Xp[i,3],-Xp[i,2]],i=1..N+1)];
> n2:=[seq([-Xp[i,3],0,Xp[i,1]],i=1..N+1)];
> vn1:=[seq([Xp[i,3],(1-2*m[i]/X[i,2])*Xp[i,3],-Xp[i,2]/X[i,2]^2],i=1..N
> +1)];
> vn2:=[seq([0,-Xp[i,3],Xp[i,1]/X[i,2]^2],i=1..N+1)];
> n1vR:=[seq(n1[i,1]*vR[i,1]+n1[i,2]*vR[i,2]+n1[i,3]*vR[i,3],i=1..N+1)];
> n2vR:=[seq(n2[i,1]*vR[i,1]+n2[i,2]*vR[i,2]+n2[i,3]*vR[i,3],i=1..N+1)];
> prevT:=[seq(-n2vR[i]*vn1[i]+n1vR[i]*vn2[i],i=1..N+1)];
> prevT2:=[seq(-(1-2*m[i]/X[i,2])*prevT[i,1]*prevT[i,1]+prevT[i,1]*prevT
> [i,2]+prevT[i,1]*prevT[i,2]+X[i,2]^2*prevT[i,3]*prev
> T[i,3],i=1..N+1)];
> vT:=[seq(prevT[i]/sqrt(-prevT2[i]),i=1..N+1)];
> vL:=[seq(1/sqrt(2.)*(vT[i]+vR[i]),i=1..N+1)];
> vN:=[seq(1/sqrt(2.)*(vT[i]-vR[i]),i=1..N+1)];
> return(vL,vN);
> end proc:
```

The following procedure calculates the expansion associated with the normal vector field V. This is pretty straightforward, deform the surface so that $X \to X + \epsilon V$ and calculate the area element on the new surface first, and then do the same thing for $X \to X - \epsilon V$. Finally we average these two results and do

three-point estimate on the rate of change.

```
> Expansion:=proc(X,Xp,V,eps,m,N)
> local dA,XplusV,XplusVp,dAplusV,XminusV,XminusVp,dAminusV,ExpV;
> dA:=AreaEl(X,Xp,m,N);
> XplusV:=X+eps*V;
> XplusVp:=FivePtDer(XplusV,N);
> dAplusV:=AreaEl(XplusV,XplusVp,m,N):
> XminusV:=X-eps*V;
> XminusVp:=FivePtDer(XminusV,N);
> dAminusV:=AreaEl(XminusV,XminusVp,m,N):
> ExpV:=[seq((dAplusV[i]-dAminusV[i])/2/dA[i]/eps,i=1..N+1)];
> return(ExpV);
> end proc:
```

Calculate just the outward expansion $\theta_{(l)}$ using the procedure above.

```
> Expansion_L:=proc(X,eps,m,N)
> local Xp,dA,vL,vN,tL_Num;
> Xp:=FivePtDer(X,N);
> dA:=AreaEl(X,Xp,m,N);
> vL,vN:=NullVec(X,Xp,m,N):
> tL_Num:=Expansion(X,Xp,vL,eps,m,N);
> return(tL_Num);
> end proc:
```

Finally, the procedure below is the core one. It makes use of all the procedures defined previously to calculate both expansions, the Ricci scalar, and the variations $\delta_n\theta_{(l)}$ and $\delta_l\theta_{(l)}$.

```
> AllQuants:=proc(X,eps,m,N)
> local Xp,dA,vL,vN,tL_Num,tN_Num,XplusN,XminusN,tL_plusN,tL_minusN,dNtL,
> R,XplusL,XminusL,tL_plusL,tL_minusL,dLtL;
> Xp:=FivePtDer(X,N);
> dA:=AreaEl(X,Xp,m,N);
> vL,vN:=NullVec(X,Xp,m,N):
> tL_Num:=Expansion(X,Xp,vL,eps,m,N);
> tN_Num:=Expansion(X,Xp,vN,eps,m,N);
> XplusN:=X+eps*vN;   XminusN:=X-eps*vN;
> tL_plusN:=Expansion_L(XplusN,eps,m,N);
> tL_minusN:=Expansion_L(XminusN,eps,m,N);
> dNtL:=[seq((tL_plusN[i]-tL_minusN[i])/2/eps,i=1..N+1)];
> R:=Ricci(X,Xp,m,N);
> XplusL:=X+eps*vL;   XminusL:=X-eps*vL;
> tL_plusL:=Expansion_L(XplusL,eps,m,N);
> tL_minusL:=Expansion_L(XminusL,eps,m,N);
> dLtL:=[seq((tL_plusL[i]-tL_minusL[i])/2/eps,i=1..N+1)];
> return(tL_Num,tN_Num,dNtL,R,dLtL);
> end proc:
```

C. A Sample Calculation

```
> N:=200; h:=1/N; eps:=0.02; a:=2;
```

$$N := 200$$

$$h := \frac{1}{200}$$

$$eps := 0.02$$

$$a := 2$$

As a simple example, we choose $m = 1$, and the parameterization of the initial surface S is $v = 2\cos(\theta)$, $r = 2m = 2$, and $\theta \in [0, \pi]$. Note that in this case, the

four-metric is just the standard Schwarzschild metric, S is a two-sphere (Figure 3.2) and it is a slice of the event/apparent horizon. Thus we expect that on this surface $\theta_{(l)} = 0$, $\theta_{(n)} < 0$ and $\delta_n\theta_{(l)} < 0$. These will be confirmed by the following results.

```
> m:=[seq(1.,i=1..N+1)]:
> X:=[seq([evalf(a*cos(Pi*(i-1)*h)),evalf(2),evalf(Pi*(i-1)*h)],i=1..N+
> 1)]:
> Xp:=FivePtDer(X,N):
```

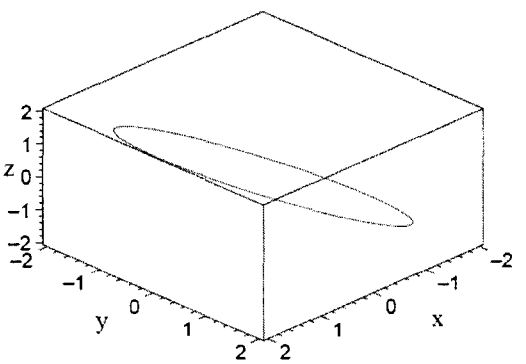


Figure 3.2: The initial two-surface S

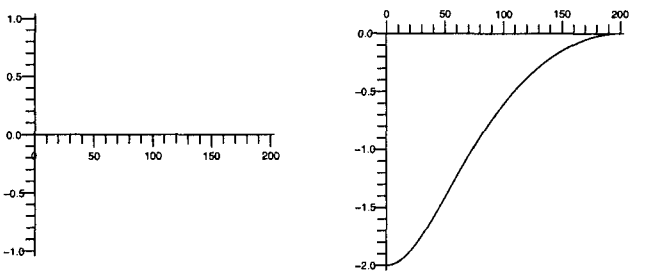
The results of both expansions, the Ricci scalar, and the variations $\delta_n\theta_{(l)}$ and $\delta_l\theta_{(l)}$ are calculated by executing the core procedure - AllQuants.

```
> tL_Num,tN_Num,dNtL_Num,Ric_Num,dLtL_Num:=AllQuants(X,eps,m,N):
```

Finally, figures of the geometric quantities we have calculated are graphed. Note that for the points at the ends, the area element $A = \sqrt{q11q22 - q12^2} = 0$, which makes some relevant results inaccurate to some extent. Thus we neglect these points when we output the results.

```
> listplot(tL_Num[2..N]);
```

```
> listplot(tN_Num[2..N]);
```



(a) The outward expansion $\theta_{(l)}$ (b) The inward expansion $\theta_{(n)}$

Figure 3.3: The expansions

```
> listplot(dNtL_Num[2..N]);
```

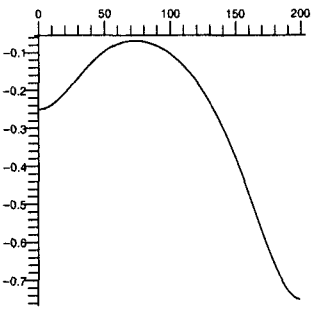


Figure 3.4: The variation $\delta_n\theta_{(l)}$

```
> listplot(dLtL_Num[2..N]);
```

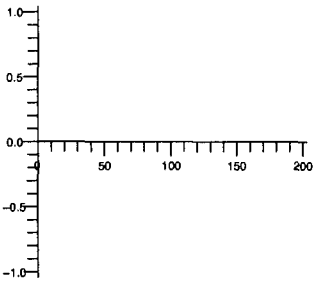


Figure 3.5: The variation $\delta_l \theta_{(l)}$

```
> listplot(Ric_Num[5..N-3]);
```

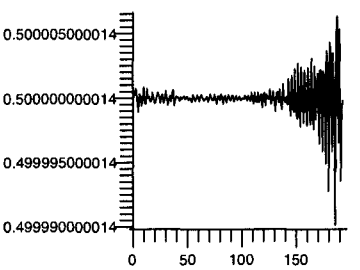


Figure 3.6: The Ricci Scalar

D. Test

To verify our algorithm we calculate the various geometric quantities of S by another method: using the GRTensorII computer algebra package. The detailed GRTensorII session is contained in the Appendix B. Here we directly use the results

and their figures to compare them with the ones calculated by our method.

The following is part of the Maple worksheet in the Appendix B. To compare the two results of expansion θ_n which are calculated by two methods, we graph them in one figure. Also, the difference value of them is plotted:

```
> tN_Ex:=[seq(evalf(subs(theta=(Pi*(i-1)/N),1/8*(sqrt(2)*(-16-8*a*cos(t
> heta)+a^2*(sin(theta))^2))/(sqrt(8+a^2*(sin(theta))^2)))),i=1..N+1)]:
> dNtL_Ex :=
> [seq(evalf(subs(theta=(Pi*(i-1)/N),-1/64*(1024+192*a^2*cos(theta)^2-51
> 2*a*cos(theta)+64*a^2-64*a^3*cos(theta)-16*a^4-a^6+3*a^6*cos(theta)^2-
> 3*cos(theta)^4*a^6+64*cos(theta)^3*a^3+16*a^4*cos(theta)^2+cos(theta)^
> 6*a^6)/(64+16*a^2+a^4-2*a^4*cos(theta)^2-16*a^2*cos(theta)^2+cos(theta
> )^4*a^4))),i=1..N+1)]:
> display(listplot(tN_Num[2..N]),listplot(tN_Ex[2..N]));
> tN_fracDiff:=[seq((tN_Num[i]-tN_Ex[i])/tN_Ex[i],i=2..N-1)]:
> listplot(tN_fracDiff);
```

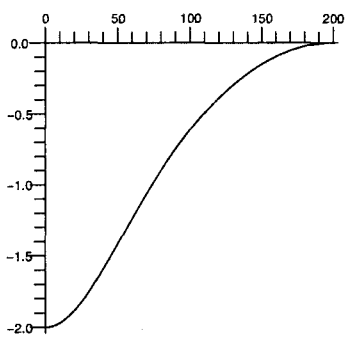


Figure 3.7: The two inward expansions $\theta_{(n)}$

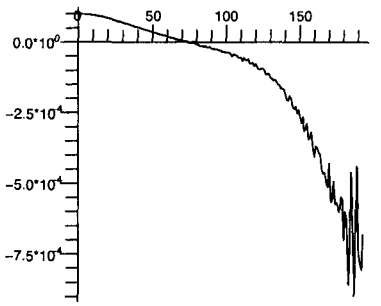


Figure 3.8: The difference between the two results of $\theta_{(n)}$

The two results of the variation $\delta_n\theta_{(l)}$ are graphed in one figure, and their difference is also plotted:

```
> display(listplot(dNtL_Num[2..N]),listplot(dNtL_Ex[2..N]));
> dNtL_fracDiff:=[seq((dNtL_Num[i]-dNtL_Ex[i])/dNtL_Ex[i],i=2..N)]:
> listplot(dNtL_fracDiff);
```

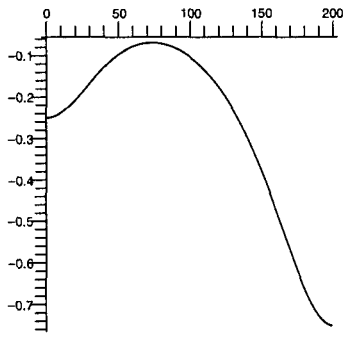


Figure 3.9: The two variations $\delta_n\theta_{(l)}$

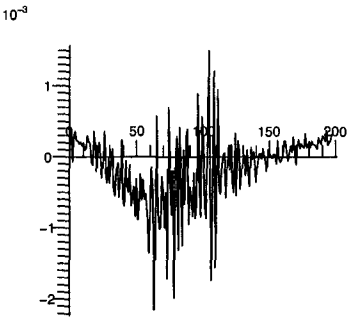


Figure 3.10: The difference between the two results of $\delta_n \theta_{(l)}$

From the result of the Appendix B, the Ricci scalar $R = \frac{1}{2m^2}$, which is invariant on slices of the horizon. So when $m = 1$, $R = 0.5$. The difference between the two results of the Ricci scalar are graphed below:

```
> Ric_fracDiff:=[seq((Ric_Num[i]-0.5)/0.5,i=4..N-2)]:
> listplot(Ric_fracDiff);
```

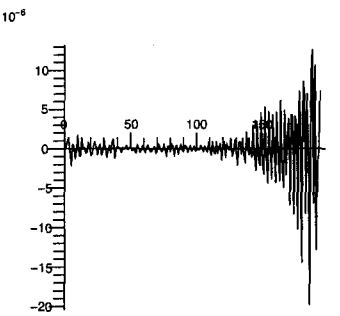


Figure 3.11: The difference between the two results of Ricci Scalar

From Figures (3.8), (3.10) and (3.11) we can see that our numerical calculations match the results computed by the GRTensorII computer algebra package. On the route to these calculations we also see that our program can solve the problem in a very direct and efficient manner. In the next section, we will see this more clearly when we calculate the deformations of two-surfaces in the Vaidya spacetime.

Chapter 4

Deformations of FOTS in Vaidya Spacetimes

A. The Vaidya Spacetime

To begin, we review some features of the Vaidya spacetime (3.15). The Vaidya spacetime is a spherically symmetric spacetime which describes the collapse of null dust in forming a black hole. Here we are mainly interested in the cases shown in Figure 4.1: for $v \leq 0$, $m(v) = 0$; for $v > 0$, $m(v)$ increases monotonically and reaches an asymptotic value M_0 as v tends to infinity.

Let us focus our attention on the two-spheres given by v=constant, r=constant. Scaling the outgoing and ingoing null normals to these two-spheres as

$$l^a = [1, 1 - \frac{2m(v)}{r}, 0, 0], \qquad n^a = [0, -1, 0, 0]. \tag{4.1}$$

The expansions of the outgoing null normal l^a is given by

$$\theta_{(l)} = \frac{1}{r}(1 - \frac{2m(v)}{r}). \tag{4.2}$$

Thus the only spherically symmetric marginally trapped surfaces are the two-spheres $r = 2m(v)$ for a specified v. These will be the apparent horizons on spherically

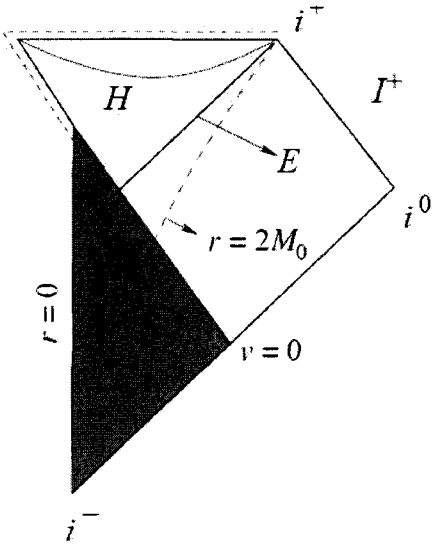


Figure 4.1: Penrose diagrams of the Vaidya spacetime [14]. This diagram is valid for a strictly increasing mass function $m(v)$ which vanishes for $v \leq 0$, and asymptotes to a finite value M_0 for $v \to \infty$. The event horizon is denoted by E. The shaded region of the spacetime is flat.

symmetric Cauchy surfaces which intersect the $r = 2m(v)$ surface [8]. Let us denote the $r = 2m(v)$ surface by H. Unlike in the Schwarzschild spacetime where the $r = 2m$ surface is null and coincides with the event horizon, in this case H is spacelike if $m'(v) > 0$, and it lies strictly inside the event horizon. At late times, H asymptotes to the event horizon.

We now consider a special kind of two-surfaces embedded in the Vaidya spacetime — future outer trapping surfaces (FOTS) that foliate future outer trapping horizons. FOTS are spacelike two-surfaces on which Einstein equations hold and for which $\theta_{(l)} = 0$, $\theta_{(n)} < 0$ and there is a scaling of the null vectors such that $\delta_n \theta_{(l)} < 0$ [6].

First, we note that on each of the two-sphere $r = 2m(v)$ (v=constant) in the Vaidya spacetime, given the scaling (4.1), $\theta_{(l)} = \frac{1}{r}(1 - \frac{2m(v)}{r}) = 0$, $\theta_{(n)} = -\frac{2}{r} < 0$, and $\delta_n\theta_{(l)} = -\frac{2}{r^2} < 0$. Thus these two-surfaces are FOTS, and the 3-surface H given by $r = 2m(v)$ for all v is a FOTH. As shown in Figure (4.1), in the Vaidya spacetime the null event horizon E, the space-like FOTH H, and the time-like surface $r = 2M_0$ all meet tangentially at the future timelike infinity i^+.

In this section we distort FOTS embedded in the Vaidya spacetime, preserving their defining characteristics $\theta_{(l)} = 0$, $\theta_{(n)} < 0$ and $\delta_n\theta_{(l)} < 0$, and study the properties of these deformations. In particular, we are interested if all FOTHs asymptote to the event horizon.

From [6] we know that for any spacelike 2-surface on which $\theta_{(l)} = 0$, it can be shown that

$$\delta_X\theta_{(l)} = -d^2B + 2\tilde{w}^a d_a B - B\delta_n\theta_{(l)} + A\delta_l\theta_{(l)}, \tag{4.3}$$

where $\tilde{w}_a = -q_a^b n_c \nabla_b l^c$ is the connection on the normal bundle $T^\perp S$. Let us begin with this formula to study the deformations of a FOTS whilst preserving its defining characteristics. Note that under sufficiently small variations, we always have $\theta_{(n)} < 0$ and $\delta_n\theta_{(l)} < 0$, thus to understand these deformations we need to find normal vector fields X^a such that $\delta_X\theta_{(l)} = 0$. By solving $\delta_X\theta_{(l)} = 0$ we can get both the evolution and the possible deformation of FOTS. In general, there will be an infinite number of X^a that will solve $\delta_X\theta_{(l)} = 0$ and so an equally infinite number of FOTS-preserving deformations. Here we focus on the case $\delta_l\theta_{(l)} \neq 0$ anywhere on S. In this situation, for any $B \in C^2(S)$ we can always solve $\delta_X\theta_{(l)} = 0$ to find a corresponding A. In our calculations, for convenience, we choose B as a positive constant, then from (4.1) we have

$$A = \frac{B\delta_n\theta_{(l)}}{\delta_l\theta_{(l)}}. \tag{4.4}$$

Under the null energy condition $\delta_l\theta_{(l)} < 0$ [6], thus $A > 0$. In this case, we obtain a l-oriented and spacelike normal variation $X^a = Al^a - Bn^a$. Then we can deform the initial FOTS S into its neighboring FOTS, $S_\epsilon = S + \epsilon X$. Using this method, we can deform the initial FOTS S into successive FOTS.

B. Symmetric Evolutions

Let us now illustrate the discussion above with the deformations of some given FOTS embedded in the Vaidya spacetime:

$$S = \{v = V(\lambda),\ r = R(\lambda),\ \theta = \Theta(\lambda),\ \phi = \phi;\ \ 0 \leq \lambda \leq 1,\ -\pi \leq \phi \leq \pi\}. \qquad (4.5)$$

The mass function is defined as $m(v) = \frac{3}{2} + \frac{1}{2}\mathrm{erf}(v)$ (see Figure 4.2), where $\mathrm{erf}(v)$ is the usual error function. For simplicity, we begin with the surface $v = 0$, $r = 2m(v) = 3$ and $\theta \in [0, \pi]$. We use our code to evolve the FOTS symmetrically towards the event horizon and then compare these numerical evolutions with exact results.

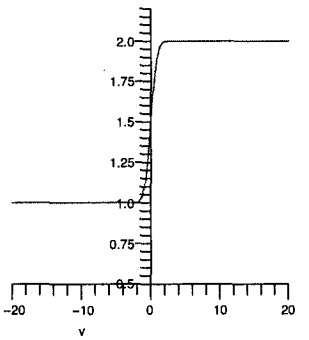


Figure 4.2: $m(v) = \frac{3}{2} + \frac{1}{2}\mathrm{erf}(v)$

Before deforming the FOTS, we first locate the event horizon and give its evolution to gain a better understanding of this situation. In principle the event horizon can be

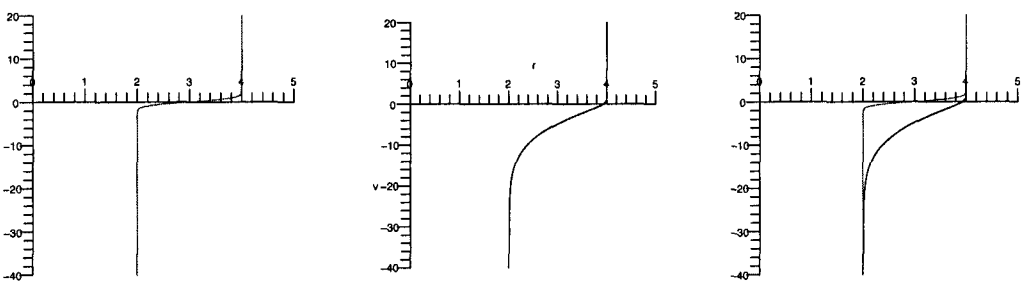
found by tracing the path of null rays through time. Furthermore, in a global sense in time, the future-pointing outgoing null geodesics that begin near the event horizon will converge on to the horizon if integrated backward in time. Building on this idea, the method of "integrating null surfaces backwards in time" [15] was developed to locate the event horizon. In this case, we note that for the null surfaces, $t \cdot t = 0$, thus we have

$$-(1 - \frac{2m(v)}{r})dv^2 + 2dvdr = 0, \tag{4.6}$$

which gives the evolution equation for the event horizon,

$$\frac{dr}{dv} = \frac{1}{2}(1 - \frac{2m(v)}{r}). \tag{4.7}$$

In a global sense, we can begin the "backward integrating" with the final stage, here we choose $v = 20$ and $r(v) = 4$ as the starting condition for the backward integration of Equation 4.7. By solving the equation, we obtain the evolution of the event horizon (Figure 4.3). Figure 4.3 shows that the event horizon coincides with the apparent horizon only at late times after the black hole settling down to a stationary state (v is large enough). In the past, the apparent horizon lies within the event horizon.



(a) Apparent Horizon $2m(v)$ (b) Evolution of event horizon (c) EH vs AH

Figure 4.3: EH and AH

Next, we use our program to simulate the process of the deformations of the FOTS. The first part of the code is similar to the one in Section 3, some procedures are defined to calculate various geometry quantities for S. For conciseness, we omit the details of the procedures that are exactly the same as the ones in Section 3.

```
> restart; with(plots):

> FivePtDer:=proc(X,N)

> AreaEl:=proc(X,Xp,m,N)

> Ricci:=proc(X,Xp,m,N)

> NullVec:=proc(X,Xp,m,N)
```

The following procedure calculates the expansion associated with the normal vector field V. It is a little different from the one in Section 3: First, $m(v)$ is no longer a constant, in this case $m(v) = \frac{3}{2} + \frac{1}{2}\text{erf}(v)$; Second, to improve the accuracy of the value of $A = \frac{B\delta_n\theta_{(l)}}{\delta_l\theta_{(l)}}$ which will be calculated later, we make a little change here - omitting the bottom part of the expansion, $\sqrt{q}$ (the area element of the surface S). From Section 3 the expansions of S are given by $\theta_{(n)} = \frac{\delta_n\sqrt{q}}{\sqrt{q}}$ and $\theta_{(l)} = \frac{\delta_l\sqrt{q}}{\sqrt{q}}$, which leads to the next relations

$$\delta_n(\delta_l\sqrt{q}) = \delta_n(\sqrt{q}\theta_{(l)}) = \sqrt{q}(\theta_{(n)}\theta_{(l)}+\delta_n\theta_{(l)}), \quad \delta_l(\delta_l\sqrt{q}) = \delta_l(\sqrt{q}\theta_{(l)}) = \sqrt{q}(\theta^2_{(l)}+\delta_l\theta_{(l)}). \tag{4.8}$$

So when $\theta_{(l)} = 0$, (4.4) reveals the relation

$$\frac{\delta_n(\delta_l\sqrt{q})}{\delta_l(\delta_l\sqrt{q})} = \frac{\delta_n\theta_{(l)}}{\delta_l\theta_{(l)}}. \tag{4.9}$$

When we approximate the value of A later, for accurateness, we use the formula (4.5) instead of $\frac{\delta_n\theta_{(l)}}{\delta_l\theta_{(l)}} = \frac{\delta_n(\frac{\delta_l\sqrt{q}}{\sqrt{q}})}{\delta_l(\frac{\delta_l\sqrt{q}}{\sqrt{q}})}$. Thus the following procedure actually calculates $\delta_n\sqrt{q}$ and $\delta_l\sqrt{q}$ instead of the expansions $\theta_{(n)}$ and $\theta_{(l)}$.

```
> Expansion:=proc(X,Xp,V,eps,m,N)
> local XplusV,XplusVp,dAplusV,XminusV,XminusVp,dAminusV,ExpV,m1,m2;
> XplusV:=X+eps*V;
> XplusVp:=FivePtDer(XplusV,N);
> m1:=[seq(evalf(3/2+erf(XplusV[i,1])/2),i=1..N-1)];
> dAplusV:=AreaEl(XplusV,XplusVp,m1,N):
> XminusV:=X-eps*V;
> XminusVp:=FivePtDer(XminusV,N);
> m2:=[seq(evalf(3/2+erf(XminusV[i,1])/2),i=1..N-1)];
> dAminusV:=AreaEl(XminusV,XminusVp,m2,N):
> ExpV:=[seq((dAplusV[i]-dAminusV[i])/2/eps,i=1..N-1)];
> return(ExpV);
> end proc:
```

Then calculate just the outward expansion $\theta_{(l)}$.

```
> Expansion_L:=proc(X,eps,m,N)
> local Xp,dA,vL,vN,tL_Num;
> Xp:=FivePtDer(X,N);
> dA:=AreaEl(X,Xp,m,N);
> vL,vN:=NullVec(X,Xp,m,N):
> tL_Num:=Expansion(X,Xp,vL,eps,m,N);
> return(tL_Num);
> end proc:
```

The procedure below invokes all the procedures defined above to calculate both expansions, the Ricci scalar, and the variations $\delta_n\theta_{(l)}$, $\delta_l\theta_{(l)}$, $\delta_n(\delta_l\sqrt{q})$ and $\delta_l(\delta_l\sqrt{q})$.

```
> AllQuants:=proc(X,eps,m,N)
> local Xp,dA,vL,vN,tL_Num,tN_Num,XplusN,XminusN,tL_plusN,tL_minusN,
> dNtL,R,XplusL,XminusL,tL_plusL,tL_minusL,dLtL,m1,m2,m3,m4,tL,tN,
> AdNtL,AdLtL;
> Xp:=FivePtDer(X,N);
> dA:=AreaEl(X,Xp,m,N);
> vL,vN:=NullVec(X,Xp,m,N):
> tL:=Expansion(X,Xp,vL,eps,m,N);
> tN:=Expansion(X,Xp,vN,eps,m,N);
> tL_Num:=[seq(evalf(tL[i]/dA[i]),i=1..N-1)];
> tN_Num:=[seq(evalf(tN[i]/dA[i]),i=1..N-1)];
> XplusN:=X+eps*vN;
> m1:=[seq(evalf(3/2+erf(XplusN[i,1])/2),i=1..N-1)];
> XminusN:=X-eps*vN;
> m2:=[seq(evalf(3/2+erf(XminusN[i,1])/2),i=1..N-1)];
> tL_plusN:=Expansion_L(XplusN,eps,m1,N);
> tL_minusN:=Expansion_L(XminusN,eps,m2,N);
> dNtL:=[seq((tL_plusN[i]-tL_minusN[i])/2/eps/dA[i],i=1..N-1)];
> AdNtL:=[seq((tL_plusN[i]-tL_minusN[i])/2/eps,i=1..N-1)];
> XplusL:=X+eps*vL;
> m3:=[seq(evalf(3/2+erf(XplusL[i,1])/2),i=1..N-1)];
> XminusL:=X-eps*vL;
> m4:=[seq(evalf(3/2+erf(XminusL[i,1])/2),i=1..N-1)];
> tL_plusL:=Expansion_L(XplusL,eps,m3,N);
> tL_minusL:=Expansion_L(XminusL,eps,m4,N);
> dLtL:=[seq((tL_plusL[i]-tL_minusL[i])/2/eps/dA[i],i=1..N-1)];
> AdLtL:=[seq((tL_plusL[i]-tL_minusL[i])/2/eps,i=1..N-1)];
> R:=Ricci(X,Xp,m,N);
```

```
> return(tL_Num,tN_Num,dNtL,R,dLtL,AdNtL,AdLtL);
> end proc:
```

The following procedure is the core one in this section. It deforms a FOTS inputted as X into its neighboring FOTS. For simplicity, we set $B = 1$, then A can be easily calculated as $A = \frac{\delta_n(\delta_l\sqrt{q})}{\delta_l(\delta_l\sqrt{q})}$. The normal variation dX can be written as $dX = Al^a - n^a$, and then the new surface is given as $X1 = X + dX$. In this procedure we also give the parametric equations of $X1$ in the Cartesian coordinate system, which will make us plot the figure of the surface conveniently.

```
> Loop:=proc(X,m,N)
> local Xp,tL_Num,tN_Num,dNtL_Num,Ric_Num,dLtL_Num,tL,tN,B,A,dX,X1,
> m1,C,D,E,F,G,AdLtL,AdNtL,S1;
> Xp:=FivePtDer(X,N):
> tL,tN := NullVec(X, Xp, m, N):
> tL_Num,tN_Num,dNtL_Num,Ric_Num,dLtL_Num,AdNtL,AdLtL:=AllQuants(X,
> eps,m,N):
> B := 1:
> A := [seq(B*AdNtL[i]/AdLtL[i], i = 1..N-1)]:
> dX := [seq(eps*[A[i]*tL[i, 1]-B*tN[i, 1], A[i]*tL[i,2]-B*tN[i, 2],
> A[i]*tL[i, 3]-B*tN[i, 3]], i = 1..N-1)]:
> X1 := X+dX:
> S1:=[seq([evalf(X1[i,2]*cos(X1[i,3])),evalf(X1[i,2]*sin(X1[i,3])),
> evalf(X1[i,1])],i=1..N-1)]:
> m1:=[seq(evalf(3/2+erf(X1[i,1])/2),i=1..N-1)]:
> Xp:=FivePtDer(X1,N):
> tL_Num,tN_Num,dNtL_Num,Ric_Num,dLtL_Num,AdNtL,AdLtL:=AllQuants(X1,
> eps,m1,N):
> return(tL_Num,tN_Num,dNtL_Num,dLtL_Num,Ric_Num,X1,m1,S1);
> end proc:
```

The initial FOTS is inputted below: $v = 0$, $r = 2m = 3$ and $\theta \in [0, \pi]$; in this case the mass function $m(v) = \frac{3}{2} + \frac{1}{2}\mathrm{erf}(v) = \frac{3}{2}$.

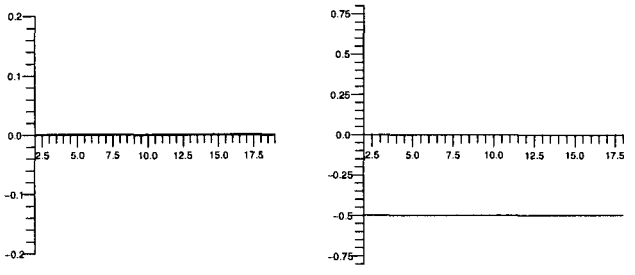
```
> N:=20:  h:=1/N:  eps:=0.02:
> a:=2:
> m:=[seq(evalf(3/2+erf(0)/2),i=1..N-1)]:
> X:=[seq([evalf(0),evalf(2*(3/2+erf(0)/2)),evalf(Pi*i*h)],i=1..N-1)]:
```

Finally, we repeatedly invoke the procedure "Loop" to deform the initial FOTS S into successive FOTS S_1, S_2, ... , S_{T-1}. We obtain the various geometric quantities of the final surface S_{T-1} and the figures of the deformed surfaces at each step. Here we repeatedly deform the initial FOTS for 50 times ($T = 51$) and see how much we can extend the FOTS.

```
> T:=51:
> for t from 1 by 1 while t < T do
>       tL,tN,dNtL,dLtL,Ric,X,m,S:=Loop(X,m,N):
>       Surf[t]:= spacecurve(S, axes=box,  color = red):
> end do:
```

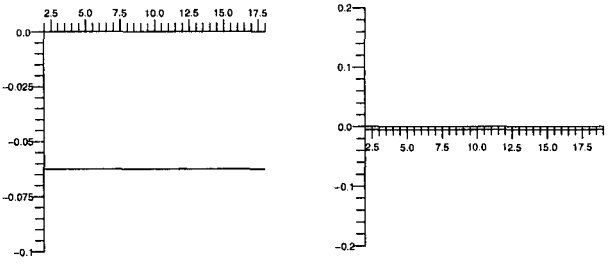
The figures of the expansions, the variations $\delta_n\theta_{(l)}$, $\delta_l\theta_{(l)}$ and the Ricci scalar of the final surface S_{T-1} are plotted below. We can see that on this surface $\theta_{(l)} = 0$, $\theta_{(n)} < 0$ and $\delta_n\theta_{(l)} < 0$, so S_{T-1} is still a FOTS.

```
> listplot(tL, view=[1..N-1,-0.2..0.2]);
> listplot(tN, view=[1..N-1,-0.8..0.8]);
> listplot(dNtL, view=[1..N-1,-0.1..0]);
> listplot(dLtL, view=[1..N-1,-0.2..0.2]);
> listplot(Ric, view=[2..N-2,-0.3..0.3]);
```



(a) The outward expansion $\theta_{(l)}$ (b) The inward expansion $\theta_{(n)}$

Figure 4.4: The expansions



(a) The variation $\delta_n\theta_{(l)}$ (b) The variation $\delta_l\theta_{(l)}$

Figure 4.5: The variations of the expansions

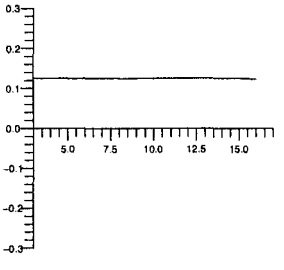


Figure 4.6: The Ricci Scalar

The three-dimensional figures of the 50 deformed surfaces in the Cartesian coordinate system are plotted below. From this figure we can easily see how the FOTS are deformed gradually.

```
> SS:=[seq(Surf[t],t=1..T-1)]:
> display(SS,orientation=[30,70]);
```

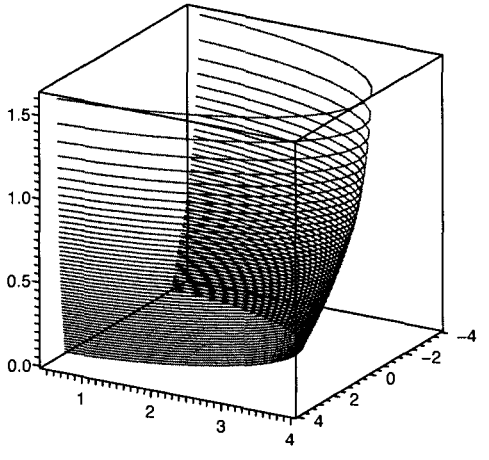
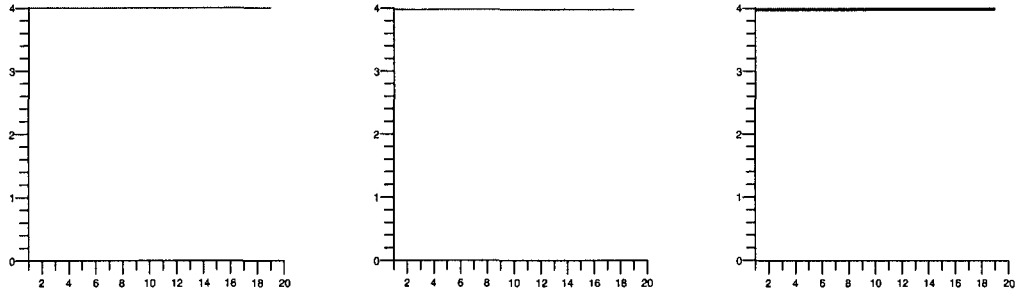


Figure 4.7: The deformed FOTS

To find where we have extended the FOTS, we now plot the figures of r and $2m(v)$ for the final surface S_{T-1}.

```
> r:=[seq(X[i,2],i=1..N-1)]:
> listplot(r,color = red, view=[1..20,0..4.01]);
> listplot(2*m, view=[1..20,0..4.01]);
> display(listplot(r,color = red), listplot(2*m,color =
> black), view=[1..20,0..4.01]);
```



(a) r (b) $2m$ (c) r vs $2m$

Figure 4.8: r and $2m$

From Figure 4.3 we know that in the Vaidya spacetime where $m(v) = \frac{3}{2} + \frac{1}{2}\text{erf}(v)$, at late times $r = 2m(v)$ asymptotes to the event horizon $r = 4$. Combining this with Figures (4.7) and (4.8) we can see that at late times, having been successively deformed for 50 times, the FOTS asymptote to the event horizon.

C. Non-symmetric Evolutions

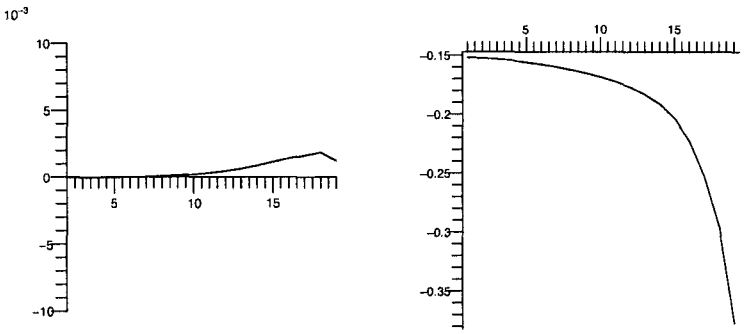
What we have discussed so far are all about symmetric evolutions. Next we turn to study some non-symmetric cases. As noted in [6], the rescaling of the null normal vectors is often important when we study FOTS. When we deform the FOTS, different scaling of the null vectors may generate different deformations of the surfaces. We now make use of scalings as a tool to study some non-symmetric FOTS.

We rescale the null vectors we used previously to become

$$l_1^a = f l^a, \quad n_1^a = \frac{1}{f} n^a, \tag{4.10}$$

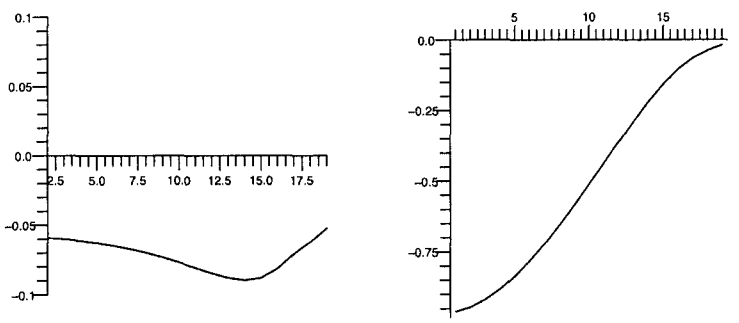
where $f = 3 + 0.5\cos(\theta)$. Then we deform the initial FOTS (4.3) for 100 times with this scaling, using the same method as before. By doing this, the initial symmetric FOTS is deformed into successive non-symmetric FOTS S_1, S_2, ... , S_{100}. At last, we

plot the figures of various geometric quantities for the final surface S_{100} below.



(a) The outward expansion $\theta_{(l)}$ (b) The inward expansion $\theta_{(n)}$

Figure 4.9: The expansions



(a) The variation $\delta_n\theta_{(l)}$ (b) The variation $\delta_l\theta_{(l)}$

Figure 4.10: The variations of the expansions

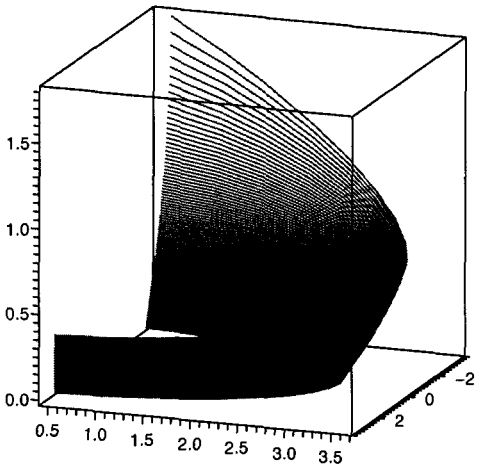
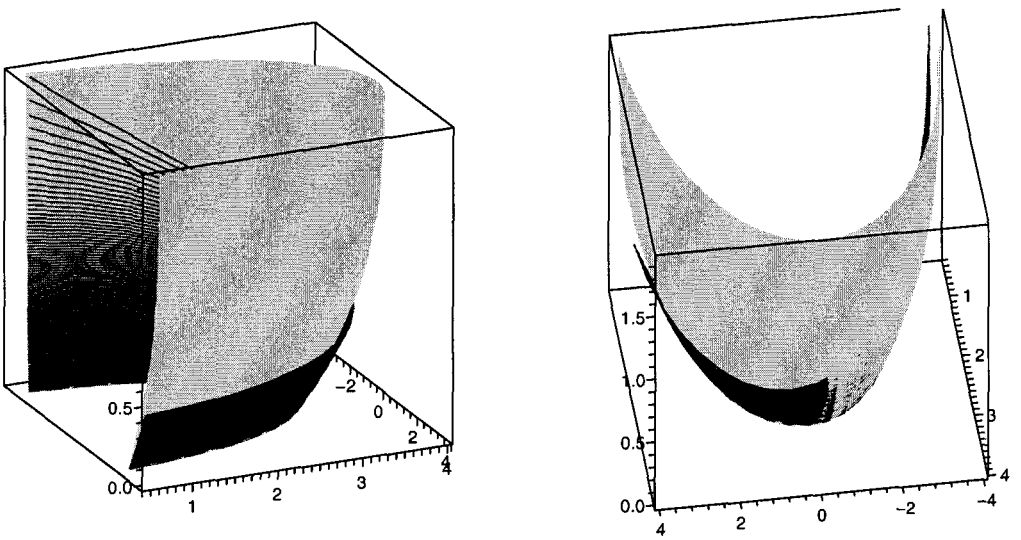


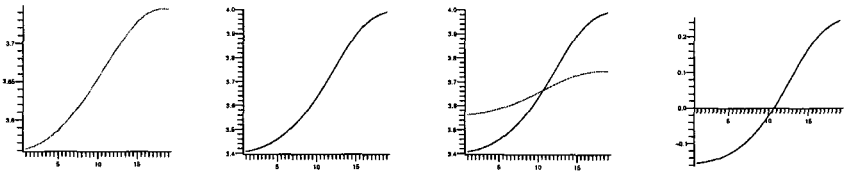
Figure 4.11: The deformed FOTS



(a) Viewed in profile

(b) Viewed from behind

Figure 4.12: The deformed FOTS and Apparent horizon



(a) r (b) $2m$ (c) r vs $2m$ (d) $2m - r$

Figures (4.9) and (4.10) show that the surface S_{100} is still a FOTS. Figure (4.12) graphs the deformations of the non-symmetric deformed FOTS and the apparent horizon at late stage. Figure 4.12 (b) shows that having been deformed for 100 times, the FOTS still do not approach the apparent horizon. In addition, it seems likely that the FOTS diverge away from the apparent horizon as the deformations go on. This is further confirmed in Figure (4.13) which shows that on the FOTS S_{100}, the difference between $2m$ and r, $2m - r$, increases gradually. As a result, it suggests that in this case, the non-symmetric FOTS may not extend to the event horizon.

While our method can calculate the deformations of FOTS in the Vaidya spacetime conveniently, we meet some difficulties at the late stage of the deformations. As shown in our calculations (Figure 4.5 and Figure 4.10), after we repeat the deformations for sufficiently many times, the variation $\delta_l\theta_{(l)}$ becomes infinitesimal, then the value of A (from equation 4.4 $A = \frac{B\delta_n\theta_{(l)}}{\delta_l\theta_{(l)}}$) becomes inaccurate to some extent. This will affect the deformations subsequent and makes the result not accurate enough. For this reason, with the present implementation of our method, we are limited to repeat the deformations for no more than 100 times, instead of continually repeating the deformations throughout the numerical evolution, as would be our final goal. In future implementations of the method, we anticipate developing a more sophisticated method to improve the robustness and accuracy of the code.

Chapter 5

Conclusions

In this paper we have presented a numerical method for computing the geometry and deformations of spacelike two-surfaces embedded in Vaidya spacetimes. We discretize a spacelike two-surface and define normal vectors $X^a = Al^a - Bn^a$ to each point of the surface. Using these normal vectors we deform the surface into its neighboring surfaces and calculate the intrinsic and extrinsic geometry of the surface and their variations. We have implemented these ideas numerically and shown several examples. In these examples, our method has been tested and shown to correctly reproduce known results.

We have also used our method to investigate the deformations of FOTS which foliate FOTH in the Vaidya spacetime. Solutions of $\delta_X \theta_{(l)} = 0$ generate the possible deformations of the FOTH. We have studied the allowed deformations of some FOTS in the Vaidya spacetime and tried to distort them towards the event horizon. The result suggests that in the Vaidya spacetime, some FOTS may not extend to the event horizon after finite extensions.

The geometry and dynamics of black hole horizons can be studied by considering the allowed deformations of their foliating two-surfaces. Our method provides a con-

venient tool to calculate the deformations of two-surfaces in Vaidya spacetimes. The method is being further developed by Dr. Ivan Booth and his students. In the future, the robustness and accuracy of the method will be improved, and more interesting findings are expected.

Appendix A

Numerical Differentiation

In this appendix we briefly review the problem of computing the derivative of a given function $f(x)$. We focus on three forms which are commonly considered: forward, backward and centered differences. The methods discussed here are used repeatedly in our codes.

The derivative of a function f at a point x_0, denoted $f'(x_0)$, is defined by the limit

$$f'(x_0) = \lim_{h \to 0} \frac{f(x_0 + h) - f(x_0)}{h}.$$

So when h is a small positive constant, the forward difference divided by h approximates the derivative $f'(x_0)$,

$$f'(x_0) \approx \frac{f(x_0 + h) - f(x_0)}{h}.$$

This approximation is called the *two-point forward difference formula* [16].

The error in this approximation can be derived from Taylor's theorem. Assuming that $f''(x)$ exists on $[x_0, x_0 + h]$, then, by Taylor's Theorem, $f(x_0 + h) = f(x_o) + f'(x_0)h + f''(\xi)h^2/2$, where $\xi \in [x_0, x_0 + h]$. Solving for $f'(x_0)$, we obtain

$$f'(x_0) = \frac{f(x_0 + h) - f(x_0)}{h} + \frac{f''(\xi)h}{2},$$

so the error in the forward difference formula is $O(h)$. This formula is *first-order accurate.*

The *backward difference formula* can be obtained by replacing h by $-h$ in the forward difference formula, where h is still positive,

$$f'(x_0) \approx \frac{f(x_0) - f(x_0 - h)}{h},$$

The backward difference formula is also first-order accurate.

However, the central difference yields a more accurate approximation. The *three-point centered difference formula* can be obtained by averaging the forward and backward difference approximation,

$$f'(x_0) \approx \frac{f(x_0 + h) - f(x_0 - h)}{2h}.$$

Its error is proportional to square of the spacing. Assuming that $f'''(x) \in C^3$ on $[x_0 - h, x_0 + h]$, then, by Taylor's Theorem again,

$$f(x_0 + h) = f(x_o) + f'(x_0)h + \frac{f''(x_0)h}{2} + \frac{f'''(\xi_+)h^3}{6},$$

$$f(x_0 - h) = f(x_o) - f'(x_0)h + \frac{f''(x_0)h}{2} - \frac{f'''(\xi_-)h^3}{6},$$

where $\xi_+ \in [x_0, x_0 + h]$ and $\xi_- \in [x_0 - h, x_0]$. Solving for $f'(x_0)$, we obtain

$$f'(x_0) = \frac{f(x_0 + h) - f(x_0 - h)}{2h} - \frac{f'''(\xi_+) + f'''(\xi_-)}{12}h^2.$$

By the Intermediate Value Theorem [16], f''' must assume every value between $f'''(\xi_-)$ and $f'''(\xi_+)$ on the interval (ξ_-, ξ_+), including the average of these two values. Therefore, we can rewrite this equation

$$f'(x_0) = \frac{f(x_0 + h) - f(x_0 - h)}{2h} - \frac{f'''(\xi)}{6}h^2,$$

where $\xi \in [x_0 - h, x_0 + h]$. Note that the term involving $f''(x_0)$ is cancelled during the calculation. The center difference formula is *second-order accurate.*

While we can use Taylor's Theorem to derive difference formulas with higher-order accuracy simply by evaluating $f(x)$ at more points, this process can be tedious. An alternative approach is to use the Richardson extrapolation method [16]: first construct the Lagrange interpolating polynomial through some neighboring points, then differentiate the Lagrange polynomial, and finally evaluate the derivative at the desired point. For example, suppose we want to compute the derivative at a point x_0 using the points $\{x_{-m}, ..., x_{-1}, x_0, x_1, ..., x_n\}$, where m and n are known nonnegative integers, and $x_{-m} < x_{-m+1} < ... < x_{n-1} < x_n$. Then a finite difference formula for $f'(x_0)$ can be obtained by analytically computing the derivatives of the Lagrange polynomials $\mathcal{L}_{j,i}(x)_{i=-m}^{n}$ for these points, where $j = m + n + 1$.

Using this method we can derive the *fourth-order accurate five-point centered formula* which is one of the best known finite difference formulas,

$$f'(x_0) = \frac{f(x_0 - 2h) - 8f(x_0 - h) + 8f(x_0 + h) - f(x_0 + 2h)}{12h} + \frac{f^5(\xi)h^4}{30},$$

where $\xi \in [x_0 - 2h, x_0 + 2h]$.

If there is no information available about $f(x)$ for $x < x_0$, then we can use the following forward five-point formula,

$$f'(x_0) = \frac{-25f(x_0) + 48f(x_0 + h) - 36f(x_0 + 2h) + 16f(x_0 + 3h) - 3f(x_0 + 4h)}{12h} + \frac{f^5(\xi)h^4}{5},$$

where $\xi \in [x_0, x_0 + 4h]$. As before, if we have not any information about $f(x)$ for $x > x_0$ we can replace h by $-h$ to obtain a backward formula that approximates $f'(x_0)$ using the values of $f(x)$ at points $\{x_0,\ x_0 - h,\ x_0 - 2h,\ x_0 - 3h,\ x_0 - 4h\}$.

Appendix B

Calculations with GRTensorII

The bulk of this appendix is the output from a Maple worksheet which was constructed by Ivan Booth. In the worksheet various geometric quantities, namely the induced metric $q_{\alpha\beta}$, the Ricci scalar R, the expansions $\theta_{(n)}$ and $\theta_{(l)}$ and the variation $\delta_n\theta_{(l)}$, for the two-sphere in the Vaidya spacetime, $S = \{a\cos(\theta),\ 2,\ \theta\}$, where $\theta \in [0,\ \pi]$ and $a = 2$, are calculated with the GRTensorII computer algebra package. These results are used in Section 3 to test our algorithm.

To begin the Maple session we load the GRTensorII libraries and the Vaidya metric.

```
> restart: readlib(grii): with(plots): grtw();
```

GRTensorII Version 1.79 (R4)

6 February 2001

Developed by Peter Musgrave, Denis Pollney and Kayll Lake

Copyright 1994 – 2001 by the authors.

Latest version available from : http : //grtensor.phy.queensu.ca/

```
> qload(Vaidya);
```

Default spacetime = Vaidya

For the Vaidya spacetime :

Coordinates

x(*up*)

$$x^{a} = [v,\ r,\ \theta,\ \phi]$$

Line element

$$ds^2 = (-1 + \frac{2m}{r})\ dv^2\ + 2\ dv\ dr\ + r^2\ d\theta^2\ + r^2 \sin(\theta)^2\ d\phi^2$$

```
> grcalc(g(up,up)); grdisplay(g(up,up));

Calculated detg for Vaidya (0.000000 sec.)

Calculated g(up,up) for Vaidya (0.016000 sec.)
```

CPU Time = 0.063

For the Vaidya spacetime :

Contravariant metric tensor

g(*up*, *up*)

$$g^{a\ b} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & \frac{r-2m}{r} & 0 & 0 \\ 0 & 0 & \frac{1}{r^2} & 0 \\ 0 & 0 & 0 & \frac{1}{r^2 \sin(\theta)^2} \end{bmatrix}$$

Our analysis begins with the calculation of the two future-pointing null normals to the surface S. The method is similar to the one we discussed in Section 3. First

consider two-surfaces in the Vaidya spacetime, $\{v = V(\lambda),\ r = R(\lambda),\ \theta = \lambda,\ \phi = \phi;\ \ 0 \leq \lambda \leq 1,\ -\pi \leq \phi \leq \pi\}$. We can find two normal one-forms to the surfaces:

$$n_1 = dr - R'd\theta, \quad n_2 = -dv + V'd\theta.$$

With the Vaidya metric we convert them to normal vectors:

$$\vec{n}_1 = [1,\ \frac{r-2m}{r},\ \frac{-R'}{r^2},\ 0], \quad \vec{n}_2 = [0,\ -1,\ \frac{V'}{r^2},\ 0].$$

Then we can define the outward pointing spacelike unit normal:

$$\vec{R} \equiv \frac{\vec{n}_1 - \alpha\vec{n}_2}{\parallel \vec{n}_1 - \alpha\vec{n}_2 \parallel},$$

where α is a parameter function. Here we choose $\alpha = \frac{2m}{r}$, which can simplify the calculation. Next, we construct a timelike normal that is perpendicular to $\vec{R}$:

$$\vec{t} = -n_2(\vec{R})\vec{n}_1 + n_1(\vec{R})\vec{n}_2.$$

Then we can define the unit timelike normal:

$$\vec{T} = \frac{\vec{t}}{\sqrt{-\vec{t}\cdot\vec{t}}}.$$

Finally, the two null normal vectors are defined as:

$$\vec{l} = \frac{1}{\sqrt{2}}(\vec{T} + \vec{R}),$$

$$\vec{n} = \frac{1}{\sqrt{2}}(\vec{T} - \vec{R}).$$

Using this method the null normals to the surface S can be calculated. Note that for S, $V' = -2\sin(\theta)$ and $R' = 0$.

```
> grdef('n2{a}:=[-1,0,Vp(theta),0]');
> grcalc(n2(up));
> grdisplay(n2(up));

This object is already defined. The new definition has been ignored.
```

$$CPU\ Time\ = 0.$$

For the Vaidya spacetime :

$$n2(up)$$

$$\mathrm{n2}(up)$$

$$n2^a = \left[0,\ -1,\ \frac{\mathrm{Vp}(\theta)}{r^2},\ 0\right]$$

```
> Rp(theta):=0;
```

$$\mathrm{Rp}(\theta) := 0$$

```
> grdef('n1{a}:=[0,1,-Rp(theta),0]');
> grcalc(n1(up));
> grdisplay(n1(up));

Components assigned for metric: Vaidya

Created definition for n1(dn)

Created definition for n1(up)

Calculated n1(up) for Vaidya (0.000000 sec.)
```

$$CPU\ Time\ = 0.$$

For the Vaidya spacetime :

$$n1(up)$$

$$\mathrm{n1}(up)$$

$$n1^a = \left[1,\ \frac{r-2\,m}{r},\ 0,\ 0\right]$$

```
> grdef('pR{a}:=n1{a}-2*m/r*n2{a}');
> grcalc(pR(up));
> grdisplay(pR(up));

Created definition for pR(dn)
```

```
Created definition for pR(up)

Calculated pR(dn) for Vaidya (0.000000 sec.)

Calculated pR(up) for Vaidya (0.000000 sec.)
```

$$CPU\ Time\ = 0.$$

For the Vaidya spacetime :

$$pR(up)$$

$$\mathrm{pR}(up)$$

$$pR^a = \left[1,\ 1,\ -\frac{2\,m\,\mathrm{Vp}(\theta)}{r^3},\ 0\right]$$

```
> grdef('pR2:=pR{a}*pR{^a}'); grcalc(pR2); grdisplay(pR2);

Created definition for pR2

Calculated pR2 for Vaidya (0.000000 sec.)
```

$$CPU\ Time\ = 0.$$

For the Vaidya spacetime :

$$pR2$$

$$pR2 = \frac{2\,m\,r^3 + r^4 + 4\,m^2\,\mathrm{Vp}(\theta)^2}{r^4}$$

```
> subs(r=2*m,Vp(theta)=0,Rp=0,grcomponent(pR2));

                                  2
> grdef('R{a}:=pR{a}/sqrt(pR2)');
> grcalc(R(dn)); grdisplay(R(dn));
> grcalc(R(up)); grdisplay(R(up));

Created definition for R(dn)

Calculated R(dn) for Vaidya (0.015000 sec.)
```

$$CPU\ Time\ = 0.015$$

For the Vaidya spacetime :

$$R(dn)$$

$$\mathrm{R}(dn)$$

$$R_a = \left[\frac{2m}{r\sqrt{\%1}}, \frac{1}{\sqrt{\%1}}, -\frac{2m\,\mathrm{Vp}(\theta)}{r\sqrt{\%1}}, 0\right]$$

$$\%1 := \frac{2m\,r^3 + r^4 + 4m^2\,\mathrm{Vp}(\theta)^2}{r^4}$$

```
Created definition for R(up)
```

```
Calculated R(up) for Vaidya (0.000000 sec.)
```

$$CPU\ Time\ = 0.015$$

For the Vaidya spacetime :

$$R(up)$$

$$\mathrm{R}(up)$$

$$R^a = \left[\frac{1}{\sqrt{\%1}}, \frac{1}{\sqrt{\%1}}, -\frac{2m\,\mathrm{Vp}(\theta)}{r^3\sqrt{\%1}}, 0\right]$$

$$\%1 := \frac{2m\,r^3 + r^4 + 4m^2\,\mathrm{Vp}(\theta)^2}{r^4}$$

```
> radsimp(subs(r=2*m,grcomponent(R(up),[v])));
> radsimp(subs(r=2*m,grcomponent(R(up),[r])));
```

$$\frac{2m}{\sqrt{8m^2 + \mathrm{Vp}(\theta)^2}}$$

$$\frac{2m}{\sqrt{8m^2 + \mathrm{Vp}(\theta)^2}}$$

```
> grdef('R2:=R{a}*R{^a}');
> grcalcalter(R2,radsimp); grdisplay(R2);
```

```
Created definition for R2
```

```
Simplification will be applied during calculation.
```

```
Applying routine radsimp to object R2
```

```
Calculated R2 for Vaidya (0.000000 sec.)
```

$$CPU\ Time\ = 0.$$

For the Vaidya spacetime :

$$R2$$

$$R2 = 1$$

```
> grdef('n1R:=n1{a}*R{^a}'); grcalc(n1R);
> gralter(n1R,radsimp);
> grdisplay(n1R);
```

```
Created definition for n1R
```

```
Calculated n1R for Vaidya (0.000000 sec.)
```

$$CPU\ Time\ = 0.$$

```
Component simplification of a GRTensorII object:
```

```
Applying routine radsimp to object n1R
```

$$CPU\ Time\ = 0.$$

For the Vaidya spacetime :

$$n1R$$

$$n1R = \frac{r^2}{\sqrt{2\,m\,r^3 + r^4 + 4\,m^2\,\mathrm{Vp}(\theta)^2}}$$

```
> grdef('n2R:=n2{a}*R{^a}'); grcalc(n2R);
> gralter(n2R,radsimp);
> grdisplay(n2R);
```

```
Created definition for n2R

Calculated n2R for Vaidya (0.000000 sec.)
```

$$CPU\ Time\ = 0.$$

```
Component simplification of a GRTensorII object:


Applying routine radsimp to object n2R
```

$$CPU\ Time\ = 0.$$

For the Vaidya spacetime :

$$n2R = \frac{n2R}{r\sqrt{2\,m\,r^3 + r^4 + 4\,m^2\,\mathrm{Vp}(\theta)^2}}$$

```
> grdef('pT{a}:=-n2R*n1{a}+n1R*n2{a}');
> grcalcalter(pT(up),radsimp); grdisplay(pT(up));

Created definition for pT(dn)

Created definition for pT(up)

Simplification will be applied during calculation.


Applying routine radsimp to object pT(dn)

Calculated pT(dn) for Vaidya (0.016000 sec.)

Applying routine radsimp to object pT(up)

Calculated pT(up) for Vaidya (0.015000 sec.)
```

$$CPU\ Time\ = 0.031$$

For the Vaidya spacetime :

$$pT(up)$$

$$\mathrm{pT}(up)$$

$$pT^a = \left[\frac{r^3 + 2\,\mathrm{Vp}(\theta)^2\,m}{r\,\sqrt{\%1}}, -\frac{2\,m\,(-r\,\mathrm{Vp}(\theta)^2 + r^3 + 2\,\mathrm{Vp}(\theta)^2\,m)}{r^2\,\sqrt{\%1}}, \frac{\mathrm{Vp}(\theta)}{\sqrt{\%1}}, 0\right]$$

$$\%1 := 2\,m\,r^3 + r^4 + 4\,m^2\,\mathrm{Vp}(\theta)^2$$

```
> grdef('pT2:=pT{a}*pT{^a}');
> grcalcalter(pT2,simplify);
> grdisplay(pT2);

Created definition for pT2

Simplification will be applied during calculation.


Applying routine simplify to object pT2

Calculated pT2 for Vaidya (0.032000 sec.)
```

$$CPU\ Time\ = 0.032$$

For the Vaidya spacetime :

$$pT2$$

$$pT2 = -\frac{-r\,\mathrm{Vp}(\theta)^2 + r^3 + 2\,\mathrm{Vp}(\theta)^2\,m}{r^3}$$

```
> grdef('T{a}:=pT{a}/sqrt(-pT2)');
> grcalc(T(up));
> gralter(T(up),radsimp): grdisplay(T(up));

Created definition for T(dn)

Created definition for T(up)
```

```
Calculated T(dn) for Vaidya (0.016000 sec.)

Calculated T(up) for Vaidya (0.000000 sec.)
```

$$CPU\ Time\ = 0.016$$

```
Component simplification of a GRTensorII object:


Applying routine radsimp to object T(up)
```

$$CPU\ Time\ = 0.031$$

For the Vaidya spacetime :

$$T(up)$$

$$\mathrm{T}(up)$$

$$T^{v} = \frac{(r^3 + 2\,\mathrm{Vp}(\theta)^2\,m)\,r}{\sqrt{2\,m\,r^3 + r^4 + 4\,m^2\,\mathrm{Vp}(\theta)^2}\,\sqrt{(-r\,\mathrm{Vp}(\theta)^2 + r^3 + 2\,\mathrm{Vp}(\theta)^2\,m)\,r}}$$

$$T^{r} = -\frac{2\,m\,(-r\,\mathrm{Vp}(\theta)^2 + r^3 + 2\,\mathrm{Vp}(\theta)^2\,m)}{\sqrt{2\,m\,r^3 + r^4 + 4\,m^2\,\mathrm{Vp}(\theta)^2}\,\sqrt{(-r\,\mathrm{Vp}(\theta)^2 + r^3 + 2\,\mathrm{Vp}(\theta)^2\,m)\,r}}$$

$$T^{\theta} = \frac{\mathrm{Vp}(\theta)\,r^2}{\sqrt{2\,m\,r^3 + r^4 + 4\,m^2\,\mathrm{Vp}(\theta)^2}\,\sqrt{(-r\,\mathrm{Vp}(\theta)^2 + r^3 + 2\,\mathrm{Vp}(\theta)^2\,m)\,r}}$$

```
> radsimp(subs(r=2*m,grcomponent(T(up),[v])));
> radsimp(subs(r=2*m,grcomponent(T(up),[r])));
> radsimp(subs(r=2*m,grcomponent(T(up),[theta])));
```

$$\frac{1}{2}\,\frac{4\,m^2 + \mathrm{Vp}(\theta)^2}{m\,\sqrt{8\,m^2 + \mathrm{Vp}(\theta)^2}}$$

$$-\frac{2\,m}{\sqrt{8\,m^2 + \mathrm{Vp}(\theta)^2}}$$

$$\frac{1}{2}\,\frac{\mathrm{Vp}(\theta)}{m\,\sqrt{8\,m^2 + \mathrm{Vp}(\theta)^2}}$$

```
> grdef('T2:=T{a}*T{^a}');
> grcalcalter(T2,simplify);
> grdisplay(T2);
```

```
Created definition for T2

Simplification will be applied during calculation.


Applying routine simplify to object T2

Calculated T2 for Vaidya (0.031000 sec.)
```

$$CPU\ Time\ = 0.031$$

For the Vaidya spacetime :

T2

$$T2 = -\frac{\%1}{\sqrt{\%1\, r}\sqrt{\frac{\%1}{r^3}\, r}}$$

$$\%1 := -r\,\mathrm{Vp}(\theta)^2 + r^3 + 2\,\mathrm{Vp}(\theta)^2\, m$$

```
> grdef('TR:=T{^a}*R{a}');
> grcalcalter(TR,simplify);
> grdisplay(TR);

Created definition for TR

Simplification will be applied during calculation.


Applying routine simplify to object TR

Calculated TR for Vaidya (0.016000 sec.)
```

$$CPU\ Time\ = 0.016$$

For the Vaidya spacetime :

TR

$$TR = All\ components\ are\ zero$$

```
> grdef('L{^a}:=(1/sqrt(2)*(T{^a}+R{^a}))');
> grcalcalter(L(up),simplify); gralter(L(up),radsimp);
> grdisplay(L(up)); grdef('L2:=L{a}*L{^a}');
> grcalcalter(L2,simplify):
> grdisplay(L2);

Created definition for L(up)

Simplification will be applied during calculation.


Applying routine simplify to object L(up)

Calculated L(up) for Vaidya (0.172000 sec.)
```

$$CPU\ \ Time\ \ = 0.172$$

```
Component simplification of a GRTensorII object:


Applying routine radsimp to object L(up)
```

$$CPU\ \ Time\ \ = 0.015$$

$$For\ the\ Vaidya\ spacetime:$$

$$L(up)$$

$$\mathrm{L}(up)$$

$$L^v = \frac{1}{2}\,\frac{\sqrt{2}\,(r^3 + 2\,\mathrm{Vp}(\theta)^2\,m + \sqrt{(-r\,\mathrm{Vp}(\theta)^2 + r^3 + 2\,\mathrm{Vp}(\theta)^2\,m)\,r}\,r)\,r}{\sqrt{2\,m\,r^3 + r^4 + 4\,m^2\,\mathrm{Vp}(\theta)^2}\,\sqrt{(-r\,\mathrm{Vp}(\theta)^2 + r^3 + 2\,\mathrm{Vp}(\theta)^2\,m)\,r}}$$

$$L^r = -\frac{1}{2}\,\frac{\sqrt{2}\,(-2\,r\,\mathrm{Vp}(\theta)^2\,m + 2\,m\,r^3 + 4\,m^2\,\mathrm{Vp}(\theta)^2 - \sqrt{(-r\,\mathrm{Vp}(\theta)^2 + r^3 + 2\,\mathrm{Vp}(\theta)^2\,m)\,r}\,r^2)}{\sqrt{2\,m\,r^3 + r^4 + 4\,m^2\,\mathrm{Vp}(\theta)^2}\,\sqrt{(-r\,\mathrm{Vp}(\theta)^2 + r^3 + 2\,\mathrm{Vp}(\theta)^2\,m)\,r}}$$

$$L^\theta = \frac{1}{2}\,\frac{\sqrt{2}\,\mathrm{Vp}(\theta)\,(r^3 - 2\,m\,\sqrt{(-r\,\mathrm{Vp}(\theta)^2 + r^3 + 2\,\mathrm{Vp}(\theta)^2\,m)\,r})}{\sqrt{2\,m\,r^3 + r^4 + 4\,m^2\,\mathrm{Vp}(\theta)^2}\,r\,\sqrt{(-r\,\mathrm{Vp}(\theta)^2 + r^3 + 2\,\mathrm{Vp}(\theta)^2\,m)\,r}}$$

```
Created definition for L(dn)

Created definition for L2

Simplification will be applied during calculation.


Applying routine simplify to object L(dn)

Calculated L(dn) for Vaidya (0.032000 sec.)

Applying routine simplify to object L2

Calculated L2 for Vaidya (0.000000 sec.)
```

$$CPU\ Time\ = 0.032$$

For the Vaidya spacetime :

$$L2$$

$$L2 = All\ components\ are\ zero$$

```
> radsimp(subs(r=2*m,grcomponent(L(up),[v])));
```

$$\frac{1}{4}\,\frac{\sqrt{2}\,\sqrt{8\,m^2+\mathrm{Vp}(\theta)^2}}{m}$$

```
> radsimp(subs(r=2*m,grcomponent(L(up),[r])));
```

$$0$$

```
> radsimp(subs(r=2*m,grcomponent(L(up),[theta])));
```

$$0$$

```
> radsimp(subs(r=2*m,Vp(theta)=0,grcomponent(L(up),[v])));
```

$$1$$

```
> grdef(`N{a}:=1/sqrt(2)*(T{a}-R{a})`);
> grcalc(N(up)); grdisplay(N(up)); grdef(`N2:=N{a}*N{^a}`);
> grcalcalter(N2,radsimp); grdisplay(N2);
```

```
Created definition for N(dn)

Created definition for N(up)

Calculated N(dn) for Vaidya (0.000000 sec.)

Calculated N(up) for Vaidya (0.015000 sec.)
```

$$CPU\ Time\ = 0.047$$

For the Vaidya spacetime :

$$N(up)$$

$$\mathrm{N}(up)$$

$$N^v = \frac{1}{2}\frac{\sqrt{2}\,(r^3\sqrt{\frac{\%1}{r^4}} + 2\sqrt{\frac{\%1}{r^4}}\,\mathrm{Vp}(\theta)^2\,m - r\sqrt{\%1}\sqrt{\frac{-r\,\mathrm{Vp}(\theta)^2 + r^3 + 2\,\mathrm{Vp}(\theta)^2\,m}{r^3}})}{\sqrt{\%1}\sqrt{\frac{-r\,\mathrm{Vp}(\theta)^2 + r^3 + 2\,\mathrm{Vp}(\theta)^2\,m}{r^3}}\,r\sqrt{\frac{\%1}{r^4}}}$$

$$\%1 := 2\,m\,r^3 + r^4 + 4\,m^2\,\mathrm{Vp}(\theta)^2$$

$$N^r = -\frac{1}{2}\sqrt{2}(-2\,r\sqrt{\frac{\%1}{r^4}}\,\mathrm{Vp}(\theta)^2\,m + r^2\sqrt{\%1}\sqrt{\frac{-r\,\mathrm{Vp}(\theta)^2 + r^3 + 2\,\mathrm{Vp}(\theta)^2\,m}{r^3}}$$
$$+ 2\,m\,r^3\sqrt{\frac{\%1}{r^4}} + 4\sqrt{\frac{\%1}{r^4}}\,\mathrm{Vp}(\theta)^2\,m^2)\Big/(r^2\sqrt{\%1}$$
$$\sqrt{\frac{-r\,\mathrm{Vp}(\theta)^2 + r^3 + 2\,\mathrm{Vp}(\theta)^2\,m}{r^3}}\sqrt{\frac{\%1}{r^4}})$$

$$\%1 := 2\,m\,r^3 + r^4 + 4\,m^2\,\mathrm{Vp}(\theta)^2$$

$$N^\theta = \frac{1}{2}\frac{\sqrt{2}\,\mathrm{Vp}(\theta)\,(r^3\sqrt{\frac{\%1}{r^4}} + 2\,m\sqrt{\%1}\sqrt{\frac{-r\,\mathrm{Vp}(\theta)^2 + r^3 + 2\,\mathrm{Vp}(\theta)^2\,m}{r^3}})}{r^3\sqrt{\%1}\sqrt{\frac{-r\,\mathrm{Vp}(\theta)^2 + r^3 + 2\,\mathrm{Vp}(\theta)^2\,m}{r^3}}\sqrt{\frac{\%1}{r^4}}}$$

$$\%1 := 2\,m\,r^3 + r^4 + 4\,m^2\,\mathrm{Vp}(\theta)^2$$

```
Created definition for N2
```

```
Simplification will be applied during calculation.
```

```
Applying routine radsimp to object N2
```

```
Calculated N2 for Vaidya (0.015000 sec.)
```

$$CPU\ Time\ = 0.015$$

For the Vaidya spacetime :

N2

N2 = All components are zero

```
> radsimp(subs(r=2*m,grcomponent(N(up),[v])));
```

```
> radsimp(subs(r=2*m,grcomponent(N(up),[r])));
```

```
> radsimp(subs(r=2*m,grcomponent(N(up),[theta])));
```

$$\frac{1}{2}\,\frac{\mathrm{Vp}(\theta)^2}{\sqrt{16\,m^2+2\,\mathrm{Vp}(\theta)^2}\,m}$$

$$-\frac{4\,m}{\sqrt{16\,m^2+2\,\mathrm{Vp}(\theta)^2}}$$

$$\frac{\mathrm{Vp}(\theta)}{m\,\sqrt{16\,m^2+2\,\mathrm{Vp}(\theta)^2}}$$

```
> grdef('LN:=L{a}*N{^a}'); grcalc(LN); radsimp(grcomponent(LN));
```

```
Created definition for LN
```

```
Calculated LN for Vaidya (0.000000 sec.)
```

$$CPU\ Time\ = 0.$$

$$-1$$

With the null normals we can now calculate the induced metric $q_{ab} = g_{ab} + l_a n_b + l_b n_a$, and the two null expansions $\theta_{(n)} = q^{ab}\nabla_a n_b$ and $\theta_{(l)} = q^{ab}\nabla_a l_b$.

```
> grdef('tq{a b}:=g{ab}+L{a}*N{b}+L{b}*N{a}');
> grcalc(tq(dn,dn));

Created definition for tq(dn,dn)

Calculated tq(dn,dn) for Vaidya (0.015000 sec.)
```

$$CPU\ Time\ = 0.015$$

```
> grcalc(tq(dn,dn)):
> radsimp(subs(r=2*m,grcomponent(tq(dn,dn),[theta,theta])));
> radsimp(subs(r=2*m,grcomponent(tq(dn,dn),[theta,phi])));
> radsimp(subs(r=2*m,grcomponent(tq(dn,dn),[phi,phi])));
```

$$CPU\ Time\ = 0.$$

$$4\,m^2$$

$$0$$

$$4\,m^2\sin(\theta)^2$$

```
> grdef('tL:=tq{^a ^b}*L{a;b}'); grcalc(tL): grdisplay(tL);

Created a definition for L(dn,cdn)

Created definition for tq(up,up)

Created definition for tL

Calculated g(dn,dn,pdn) for Vaidya (0.015000 sec.)

Calculated Chr(dn,dn,dn) for Vaidya (0.000000 sec.)

Calculated Chr(dn,dn,up) for Vaidya (0.000000 sec.)

Calculated L(dn,cdn) for Vaidya (0.032000 sec.)
```

```
Calculated tq(up,up) for Vaidya (0.015000 sec.)

Calculated tL for Vaidya (0.063000 sec.)
```

$$CPU\ Time\ = 0.125$$

$$For\ the\ Vaidya\ spacetime:$$

$$tL$$

$$tL = \ 89510\ words.\ Exceeds\ grOptionDisplayLimit$$

```
> radsimp(expand(subs(r=2*m,grcomponent(tL))));
```

$$0$$

```
> grdef('tN:=tq{^a ^b}*N{a;b}'); grcalc(tN): grdisplay(tN);

Created a definition for N(dn,cdn)

Created definition for tN

Calculated N(dn,cdn) for Vaidya (0.016000 sec.)

Calculated tN for Vaidya (0.125000 sec.)
```

$$CPU\ Time\ = 0.141$$

$$For\ the\ Vaidya\ spacetime:$$

$$tN$$

$$tN = \ 74308\ words.\ Exceeds\ grOptionDisplayLimit$$

```
> tNH:=radsimp(expand(subs(r=2*m,grcomponent(tN))));
```

$$tNH := -\frac{1}{4}\,\frac{16\sin(\theta)\,m^2 - 4\,m\,(\frac{d}{d\theta}\,\mathrm{Vp}(\theta))\sin(\theta) - 4\,m\,\mathrm{Vp}(\theta)\cos(\theta) - \sin(\theta)\,\mathrm{Vp}(\theta)^2}{\sin(\theta)\,\sqrt{16\,m^2 + 2\,\mathrm{Vp}(\theta)^2\,m^2}}$$

```
> tNH_cos:=radsimp(expand(subs(Vp(theta)=-a*m*sin(theta),tNH)));
```

$$tNH\_cos := \frac{1}{8}\,\frac{\sqrt{2}\,(-16 - 8\,a\cos(\theta) + a^2\sin(\theta)^2)}{m\,\sqrt{8 + a^2\sin(\theta)^2}}$$

```
> plot(subs(m=1,a=2,tNH_cos),theta=0..Pi);
```

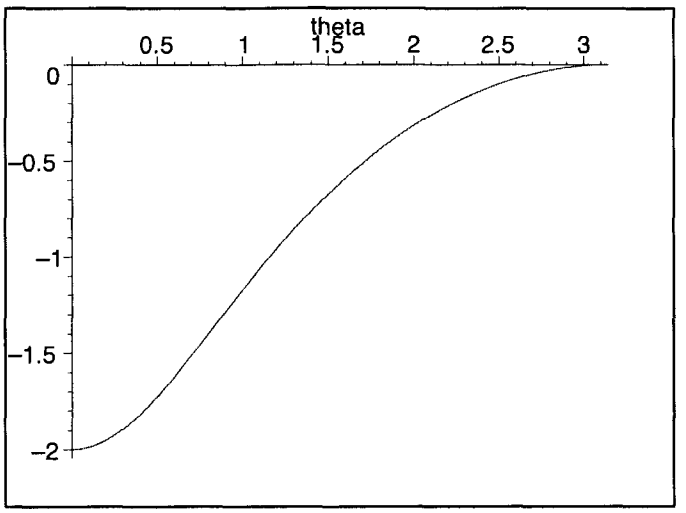


Figure B.1: The inward expansion $\theta_{(n)}$

Next, we calculate the variation $\delta_n\theta_{(l)}$. Here we just outline the calculations but more details are contained in [6]. From [6] we have

$$\delta_X\theta_{(l)} = \mathcal{K}_X\theta_{(l)} - d^2B + 2\tilde{w}^a d_a B - B[\|\tilde{w}\|^2 - d_a\tilde{w}^a - R/2 + G_{ab}l^a n^b - \theta_{(l)}\theta_{(n)}] - A[\|\sigma^{(l)}\|^2 + G_{ab}l^a l^b + (1/2)\theta_{(l)}^2],$$

where $\mathcal{K}_X = -X^a n_b \nabla_a l^b$, $\tilde{w}_a = -q_a^b n_c \nabla_b l^c$ is the normal bundle connection, $\|\tilde{w}\|^2 = \tilde{w}_a\tilde{w}^a$, $\sigma_{ab}^{(l)} = (q_a^c q_b^d - \frac{1}{2}q_{ab}q^{cd})\nabla_c l_d$ is the shear, $\|\sigma^{(l)}\|^2 = \sigma_{ab}^{(l)}\sigma^{(l)ab}$, G_{ab} is the usual Einstein tensor and R is the Ricci scalar. Set $A = 0$, $B = -1$ and $\theta_{(l)} = 0$ in this equation, we have

$$\delta_n\theta_{(l)} = -R/2 + \|\tilde{w}\|^2 - d_a\tilde{w}^a + (8\pi G)T_{ab}l^a n^b,$$

where $T_{ab} = \frac{G_{ab}}{8\pi}$, which tells us that in the absence of matter fields

$$\delta_n\theta_{(l)} = -R/2 + \|\tilde{w}\|^2 - d_a\tilde{w}^a.$$

The following code use this formula to calculate $\delta_n \theta_{(l)}$.

```
> grdef('tom{a}:=-tq{a ^b}*N{c}*L{^c;b}'); grcalc(tom(dn));

Created definition for tq(dn,up)

Created a definition for L(up,cdn)

Created definition for tom(dn)

Calculated tq(dn,up) for Vaidya (0.031000 sec.)

Calculated L(up,cdn) for Vaidya (0.016000 sec.)

Calculated tom(dn) for Vaidya (0.328000 sec.)
```

$$CPU\ \ Time\ \ = 0.375$$

```
> radsimp(subs(r=2*m,grcomponent(tom(dn),[v])));
> radsimp(subs(r=2*m,grcomponent(tom(dn),[r])));
> radsimp(subs(r=2*m,grcomponent(tom(dn),[theta])));
> radsimp(subs(r=2*m,grcomponent(tom(dn),[phi])));
```

$$\begin{array}{c} 0 \\ \frac{1}{16}\,\frac{(8\,m^2+4\,(\frac{d}{d\theta}\,\mathrm{Vp}(\theta))\,m+\mathrm{Vp}(\theta)^2)\,\mathrm{Vp}(\theta)^2}{m^3\,(8\,m^2+\mathrm{Vp}(\theta)^2)} \\ \frac{1}{4}\,\frac{\mathrm{Vp}(\theta)\,(8\,m^2+4\,(\frac{d}{d\theta}\,\mathrm{Vp}(\theta))\,m+\mathrm{Vp}(\theta)^2)}{m\,(8\,m^2+\mathrm{Vp}(\theta)^2)} \\ 0 \end{array}$$

```
> alpha:=2*m/r;
```

$$\alpha := \frac{2\,m}{r}$$

```
> grdef('tomH{a}:=[0,1/16*(8*m^2+4*alpha*(diff(Vp(theta),
> theta))*m+alpha*Vp(theta)^2)*Vp(theta)^2/(m^3*(8*m^2+alpha*Vp(theta)^2
> )),1/4*Vp(theta)*(8*m^2+4*alpha*(diff(Vp(theta),
> theta))*m+alpha*Vp(theta)^2)/(m*(8*m^2+alpha*Vp(theta)^2)),0]');
> grcalc(tomH(dn)); grdisplay(tomH(dn));

Components assigned for metric: Vaidya

Created definition for tomH(dn)
```

$$CPU\ Time\ = 0.$$

For the Vaidya spacetime :

$$tomH(dn)$$

$$\mathrm{tomH}(dn)$$

$$tomH_a = \left[0,\ \frac{1}{16}\,\frac{\left(8\,m^2+\frac{8\,m^2\,(\frac{d}{d\theta}\,\mathrm{Vp}(\theta))}{r}+\frac{2\,m\,\mathrm{Vp}(\theta)^2}{r}\right)\mathrm{Vp}(\theta)^2}{m^3\,(8\,m^2+\frac{2\,m\,\mathrm{Vp}(\theta)^2}{r})},\right.$$
$$\left.\frac{1}{4}\,\frac{\mathrm{Vp}(\theta)\,\left(8\,m^2+\frac{8\,m^2\,(\frac{d}{d\theta}\,\mathrm{Vp}(\theta))}{r}+\frac{2\,m\,\mathrm{Vp}(\theta)^2}{r}\right)}{m\,(8\,m^2+\frac{2\,m\,\mathrm{Vp}(\theta)^2}{r})},\ 0\right]$$

```
> grdef('tomH2:=tq{^a^b}*tomH{a}*tomH{b}');
> grcalc(tomH2);
> om2:=radsimp(subs(r=2*m,grcomponent(tomH2)));

Created definition for tomH2

Calculated tomH2 for Vaidya (0.016000 sec.)
```

$$CPU\ Time\ = 0.016$$

$$om2 := \frac{1}{64}\,\frac{(8\,m^2+4\,(\frac{d}{d\theta}\,\mathrm{Vp}(\theta))\,m+\mathrm{Vp}(\theta)^2)^2\,\mathrm{Vp}(\theta)^2}{m^4\,(8\,m^2+\mathrm{Vp}(\theta)^2)^2}$$

```
> grdef('dtomH:=tq{^a ^b}*tomH{a;b}');
> grcalc(dtomH);
> dom:=factor(radsimp(subs(r=2*m,grcomponent(dtomH))));
```

```
This object is already defined. The new definition has been ignored.
```

$$CPU\ Time\ = 0.$$

$$dom := \frac{1}{16}(64\,m^4\,\mathrm{Vp}(\theta)\cos(\theta) + 64\,m^4\sin(\theta)\,\%1 + 32\,m^3\sin(\theta)\,\%1^2 + 32\,m^3\,\mathrm{Vp}(\theta)\,\%1\cos(\theta) + 32\,m^3\sin(\theta)\,\mathrm{Vp}(\theta)\,(\tfrac{d^2}{d\theta^2}\,\mathrm{Vp}(\theta)) + 16\,m^2\,\mathrm{Vp}(\theta)^3\cos(\theta) + 16\,m^2\,\mathrm{Vp}(\theta)^2\sin(\theta)\,\%1 + 4\,m\sin(\theta)\,\mathrm{Vp}(\theta)^3\,(\tfrac{d^2}{d\theta^2}\,\mathrm{Vp}(\theta)) + 4\,m\,\mathrm{Vp}(\theta)^3\,\%1\cos(\theta) - 4\,m\,\mathrm{Vp}(\theta)^2\sin(\theta)\,\%1^2 + \mathrm{Vp}(\theta)^5\cos(\theta) + \mathrm{Vp}(\theta)^4\sin(\theta)\,\%1) \Big/ (m^3\,(8\,m^2 + \mathrm{Vp}(\theta)^2)^2\sin(\theta))$$

$$\%1 := \tfrac{d}{d\theta}\,\mathrm{Vp}(\theta)$$

```
> Ric:=1/2/m^2;
```

$$Ric := \frac{1}{2\,m^2}$$

```
> dNtL:=simplify(-Ric/2+om2-dom);
```

$$dNtL := -\frac{1}{64}(1024\,m^6\sin(\theta) + 192\,m^4\sin(\theta)\,\mathrm{Vp}(\theta)^2 - 32\,\mathrm{Vp}(\theta)^2\sin(\theta)\,\%1^2\,m^2 - 4\,\mathrm{Vp}(\theta)^4\sin(\theta)\,\%1\,m - \mathrm{Vp}(\theta)^6\sin(\theta) + 256\,m^5\,\mathrm{Vp}(\theta)\cos(\theta) + 256\,m^5\sin(\theta)\,\%1 + 128\,m^4\sin(\theta)\,\mathrm{Vp}(\theta)\,(\tfrac{d^2}{d\theta^2}\,\mathrm{Vp}(\theta)) + 128\,m^4\,\mathrm{Vp}(\theta)\,\%1\cos(\theta) + 128\,m^4\sin(\theta)\,\%1^2 + 64\,m^3\,\mathrm{Vp}(\theta)^3\cos(\theta) + 16\,m^2\,\mathrm{Vp}(\theta)^3\,\%1\cos(\theta) + 16\,m^2\sin(\theta)\,\mathrm{Vp}(\theta)^3\,(\tfrac{d^2}{d\theta^2}\,\mathrm{Vp}(\theta)) + 4\,m\,\mathrm{Vp}(\theta)^5\cos(\theta)) \Big/ (m^4\,(8\,m^2 + \mathrm{Vp}(\theta)^2)^2\sin(\theta))$$

$$\%1 := \tfrac{d}{d\theta}\,\mathrm{Vp}(\theta)$$

```
> expand(subs(Vp(theta)=0,dNtL));
```

$$-\frac{1}{4\,m^2}$$

```
> dNtL_cos:=simplify(expand(subs(Vp(theta)=-a*m*sin(theta),dNtL)));
```

$$dNtL\_cos := -\frac{1}{64}(1024 + 192\,a^2\cos(\theta)^2 - 512\,a\cos(\theta) + 64\,a^2 - 64\,a^3\cos(\theta) - 16\,a^4 - a^6 + 3\,a^6\cos(\theta)^2 - 3\cos(\theta)^4\,a^6 + 64\cos(\theta)^3\,a^3 + 16\,a^4\cos(\theta)^2 + \cos(\theta)^6\,a^6) \Big/ ((64 + 16\,a^2 + a^4 - 2\,a^4\cos(\theta)^2 - 16\,a^2\cos(\theta)^2 + \cos(\theta)^4\,a^4)\,m^2)$$

```
> dNtL_Num:=simplify(subs(m=1,dNtL_cos));
```

$$dNtL\_Num := -\frac{1}{64}(1024 + 192\,a^2\cos(\theta)^2 - 512\,a\cos(\theta) + 64\,a^2 - 64\,a^3\cos(\theta) - 16\,a^4 - a^6 + 3\,a^6\cos(\theta)^2 - 3\cos(\theta)^4\,a^6 + 64\cos(\theta)^3\,a^3 + 16\,a^4\cos(\theta)^2 + \cos(\theta)^6\,a^6) \Big/ (64 + 16\,a^2 + a^4 - 2\,a^4\cos(\theta)^2 - 16\,a^2\cos(\theta)^2 + \cos(\theta)^4\,a^4)$$

```
> plot(subs(a=2,dNtL_Num),theta=0..Pi);
```

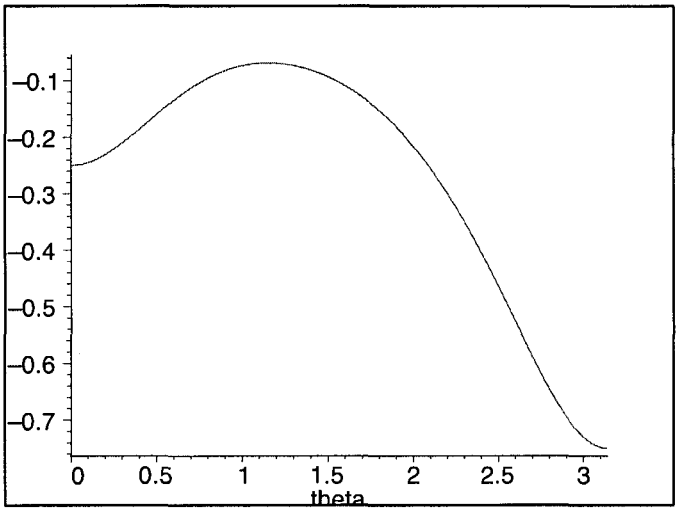


Figure B.2: The variation $\delta_n\theta_{(l)}$

Bibliography

[1] S.W. Hawking and G.F.R. Ellis, *The large scale structure of space-time.* Cambridge University Press, Cambridge. (1973).

[2] I. Booth, *Black hole boundaries.* Can. J. Phys. V83, 1073 (2005).

[3] R. Penrose, Phys. Rev. Lett. 14 57 (1965).

[4] R.M. Wald, *General relativity.* University of Chicago Press, Chicago. (1984).

[5] S. A. Hayward, Phys. Rev. D 49 6467 (1994).

[6] I. Booth and S. Fairhurst, *Isolated, slowly evolving and dy namical trapping horizons:geometry and mechanics from surface deformations.* Phys. Rev. D 75, 084019 (2007).

[7] D. M. Eardley, Phys. Rev. D 57, 2299 (1998).

[8] E. Schnetter and B. Krishnan, Phys. Rev. D 73, 021502(R) (2006).

[9] I. Ben-Dov, *Outer trpped surfaces in Vaidya spacetimes.* Phys. Rev. D 75, 064007 (2007).

[10] E. Gourgoulhon, *3+1 Formalism and Bases of Numerical Relativity.* Lectures at Institut Henri Poincare (Paris, Sept.-Dec. 2006), gr-qc/0703035 (2007).

[11] E. Poisson, *A relativist's toolkit: The mathematics of black-hole mechanics.* Cambridge University Press, Cambridge. (2004).

[12] E. Gourgoulhon, Phys. Rev. D 72, 104007 (2005).

[13] P. C. Vaidya, Proc. Ind. Acad. Sci. A33, 264 (1951).

[14] A. Ashtekar and B. Krishnan, *Isolated and dynamical horizons and their applications.* Living Rev. Rel. 7 (2004).

[15] P. Anninos et al, *Dynamics of Apparent and Event Horizons.* Phys. Rev. Lett. 74, 630 - 633 (1995).

[16] R. L. Burden and J. D. Faires, *Numerical Analysis.* Brooks Cole. (2000).