Implementación del Paquete xPand de Mathematica en el Desarrollo de una Rutina de Cálculo de Perturbaciones Lineales en Espacio-Tiempos de Bianchi I

Autor
Efren Steven Esquea Potes

Trabajo de grado realizado como requisito
parcial para optar al título de físico

Director
Prof. Cesar Alonso Valenzuela Toledo
Codirector
MSc. John Bayron Orjuela Quintana

Universidad del Valle
Departamento de Física
Programa de Física
Cali, Colombia
Febrero de 2023

Dedicado a mis padres, Alice y Efrén, porque sin importar qué tan difícil haya sido el camino o cuán oscuro o complicado se haya tornado el panorama, siempre han estado allí para apoyarme.

Universidad del Valle, Facultad de Ciencias, Departamento de Física
Calle 13 No 100 - 00, Edificio E20, Cali - Colombia
e-mail: steven.esquea@correounivalle.edu.co

Agradecimientos

Infinitas palabras seguirían siendo pocas para intentar transmitir y expresar el magno agradecimiento que siento hacia todas aquellas personas que han brindado su apoyo para que este trabajo haya podido realizarse. Agradezco al profesor Cesar Alonso Valenzuela por la confianza que siempre ha depositado en mí y por haberme permitido realizar este trabajo bajo su dirección; me siento honrado por ello. A John Bayron Orjuela, por la buena disposición que siempre tuvo para ayudarme y para resolver mis dudas. Debo decir que el plan de trabajo que me brindó fue clave para poder avanzar de manera fluida en las actividades correspondientes al desarrollo del trabajo. Les agradezco mucho y espero que este trabajo pueda aportar a otros estudiantes aunque sea una mínima parte de lo mucho que hacerlo me aportó a mí.

Agradezco también al profesor Javier Madroñero, porque su buena disposición como docente y como director de plan han contribuido a que el camino a la finalización de mi pregrado sea fructífero.

Me siento muy agradecido también con el profesor Heber Mesa, del Departamento de Matemáticas, por su bonita iniciativa de publicar de forma gratuita un curso completo de Geometría Diferencial en video. Sin duda alguna fue una herramienta crucial para entender los conceptos matemáticos con los que me estaba enfrentando. También me siento igualmente agradecido con el profesor Hernán Ocampo, por haberme permitido asistir a su curso de Métodos Geométricos de la Física y por haberme brindado valiosas sugerencias y aclaraciones fundamentales para poder realizar este trabajo. Su conocimiento, experiencia y forma de conectar las Matemáticas con la Física hacen de cada una de sus charlas una experiencia no sólo de gran valor, sino inspiradora.

Me gustaría también ofrecer un agradecimiento especial a los profesores Otto Vergara, Jesús María Calero, al profesor jubilado Rubén Vargas y nuevamente a los profesores Cesar Valenzuela y Javier Madroñero, del departamento de Física, por sus consejos, comprensión y buena disposición a escuchar y ayudarme a buscar soluciones cuando las dificultades de salud afectaron mi rendimiento académico. También al profesor José Kattán de la Facultad de Artes y al profesor Eduardo Marlés Sáenz, de la Escuela de Ingeniería Eléctrica y Electrónica, porque ofrecieron también una grata voz de aliento cuando menos lo esperaba.

Agradezco por supuesto a mi alma mater, la Universidad del Valle, por toda la formación que me brindó y todas las experiencias que como estudiante de pregrado me permitió vivir: momentos, sonrisas, lágrimas, personas, luchas, crecimiento... Por todo ello, ¡muchas gracias!

Finalmente, quiero agradecer a las personas más importantes en mi vida, mi familia. Quiero agradecer a mis padres, Alice y Efrén, por el apoyo que me han brindado

siempre, desde que tengo uso de razón, porque sin importar qué tan difícil haya sido el camino o cuán oscuro o complicado se haya tornado el panorama, ellos siempre han estado allí para apoyarme, escucharme, aconsejarme o simplemente para acompañarme, asegurándose de que no me sienta solo. Debo agradecer también a mi pareja, Angi, porque me ha ayudado a despertar una mejor versión de mí, algo que ha sido fundamental para avanzar en mis estudios y en mi vida. También por su apoyo como profesional, ayudándome a estructurar ideas para este trabajo. Y por haber unido fuerzas con mis padres para facilitarme el tiempo y las herramientas tecnológicas necesarias en el momento más oportuno. Agradezco también a mi hermana, Vicky, porque durante el «sube y baja» de salud mental que experimenté durante mis estudios, se atrevió a preguntar «¿Cómo estás?», en lugar de juzgar. Y al resto de mis hermanos y hermanas, porque siempre buscaron la manera de apoyarme en el momento preciso.

Por último, pero no menos importante, a Chispas, a mis sobrinos y a todas aquellas personas que a menudo han podido ver en mí mucho más de lo que yo mismo he sido capaz de ver. A todas ellas, ¡muchísimas gracias!

Abstract

In this work, the linear perturbations of the equations of motion and Einstein field equations in a Bianchi I background are derived for the case of the canonical scalar field. The calculation is performed from the instructions provided by the Wolfram Mathematica packages xPand, xPert, and xTensor. To do this, a brief review is made about geometric concepts associated with space-type hyper-surfaces embedded in a 4-dimensional Lorentzian manifold with signature $(-,+,+,+)$ and its connection with the basic cosmological perturbation theory in an anisotropic spacetime of the Bianchi I. Then the relationship between the previous concepts and the instructions provided by the software is established, from which a code implementation that allows finding the equations of the perturbations is made, detailing the most important aspects of the algorithm used. Finally, some observations are made about the results obtained, the algorithm and the methodology used, in addition to providing an appendix with all the code used during this project.

Keywords

Cosmological perturbations, scalar field, anisotropic universe, computational algebra, Differential Geometry.

Resumen

En este trabajo se derivan las perturbaciones lineales de las ecuaciones de movimiento y de campo de Einstein en una métrica de fondo Bianchi I para el caso del campo escalar canónico. El cálculo se realiza a partir de las instrucciones proporcionadas por los paquetes xPand, xPert y xTensor de Wolfram Mathematica. Para ello se hace una breve revisión de conceptos geométricos asociados a hipersuperficies tipo espacio encajadas en una variedad lorentziana de 4 dimensiones con signatura $(-,+,+,+)$ y su conexión con la teoría básica detrás de las perturbaciones cosmológicas en un espaciotiempo anisotrópico Bianchi del tipo I. Luego se establece la relación entre los conceptos previos y las instrucciones proporcionadas por el software, a partir de lo cual se hace una implementación en código que permita encontrar las ecuaciones de las perturbaciones, detallando los aspectos más importantes del algoritmo utilizado. Finalmente se plantean algunas observaciones sobre los resultados obtenidos, el algoritmo y la metodología trabajada, además de proporcionar un apéndice con todo el código utilizado durante este proyecto.

Palabras clave

Perturbaciones cosmológicas, campo escalar, universo anisotrópico, álgebra computacional, Geometría Diferencial.

Notación

- En este trabajo se trabaja con la signatura $(-,+,+,+)$.
- Salvo en las partes en donde se especifica algo diferente, los índices griegos pueden tomar los valores $0, 1, 2$ y 3, mientras que los índices latinos pueden tomar valores $1, 2$ y 3.
- Siempre que se usen índices griegos en cantidades puramente espaciales, se entenderá que hacen referencia a sus extensiones a cuatro dimensiones.
- ∇_μ denota la derivada covariante asociada a la métrica del espaciotiempo.
- D_i denota la derivada covariante asociada a la métrica espacial, mientras que D_α denota su versión en cuatro dimensiones.
- Las letras en negrillas normalmente se usarán para representar tensores abstractos (ej: $\boldsymbol{n}, \boldsymbol{g}, \boldsymbol{K}$), mientras que las mismas letras con índices griegos o latinos serán usados para representar sus componentes (ej: $n^\mu, g_{\mu\nu}, K_{ij}$).
- Se asume el convenio de suma de Einstein, tanto para tensores espaciotemporales, como para los puramente espaciales de tres dimensiones.
- Dado un vector $\boldsymbol{v}$ definido sobre un espacio tangente, se denota por $\underline{\boldsymbol{v}}$ a su covector asociado.
- Dado un covector $\boldsymbol{\omega}$ definido sobre un espacio cotangente, se denota por $\vec{\boldsymbol{\omega}}$ a su vector asociado, por lo que $\vec{\boldsymbol{\nabla}} t_s$ es el gradiente de la 1-forma gradiente $\mathbf{d}t_s$.
- dados dos vectores $\boldsymbol{u}$, $\boldsymbol{v}$ y un covector $\boldsymbol{\omega}$ de un espacio tangente, se tiene que

$$\begin{aligned} \boldsymbol{u} \cdot \boldsymbol{v} &\equiv \boldsymbol{g}(\boldsymbol{u}, \boldsymbol{v}) = g_{\mu\nu} u^\mu u^\nu, \\ \langle \boldsymbol{\omega}, \boldsymbol{u} \rangle &\equiv \boldsymbol{g}(\vec{\boldsymbol{\omega}}, \boldsymbol{v}) = \omega_\mu u^\mu. \end{aligned}$$

- En la implementación se asume que se trabaja con el miembro no nulo de ecuaciones escritas de la forma:

$$A^{\mu_1 \cdots \mu_n}{}_{\nu_1 \ldots \nu_n}(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n) = 0.$$

  Esto quiere decir que, a menos que se indique otra cosa, toda manipulación que se haga a la ecuación se hace en realidad sobre $A^{\mu_1 \cdots \mu_n}{}_{\nu_1 \ldots \nu_n}(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n)$, por lo que los resultados finales deben ser siempre igualados a cero para recuperar el carácter de ecuación.

Índice

Introducción

Una de las consideraciones que a menudo se hace en los cursos de formación en Física es aquella en la que se expresa que el universo es homogéneo e isótropo. Esta consideración, sin embargo, puede ser no del todo válida dependiendo de la escala en la que se trabaje y del observable o del fenómeno mismo que se esté estudiando. Por ejemplo, al hablar del comportamiento de la materia por efecto de la gravedad a gran escala, ya no se puede afirmar que el universo sea completamente homogéneo, pues contiene galaxias, cúmulos de galaxias, etc. A esto se le conoce como *estructura*. A gran escala ($> 100Mpc$), un modelo homogéneo e isótropo como el FLRW[1] ofrece una buena descripción del universo. No obstante, a medida que se disminuye la escala, empiezan a aparecer inhomogeneidades. En etapas tempranas, sin embargo, antes de la formación de tal estructura, por encima de los 10 Mpc, estas inhomogeneidades pueden entenderse como pequeñas desviaciones de un fondo homogéneo, es decir, como perturbaciones lineales de una métrica de fondo homogénea, por lo que resulta de utilidad explorar las perturbaciones de los modelos cosmológicos, incluso si es a orden lineal [1]-[3].

Por otra parte, el tratar de entender esa estructura lleva también a preguntarse qué generó las perturbaciones iniciales o *primordiales* que dieron origen a ella. Es aquí donde entra en juego la Inflación, un mecanismo que, entre otras cosas, provee un escenario para que se puedan dar tales perturbaciones [4]. Este mecanismo es ahora una parte importante del modelo estándar de la cosmología, por lo que entenderlo es una tarea de gran relevancia. El estudio de las perturbaciones ofrece una vía no sólo para modelar las inhomogeneidades del universo a gran escala, sino para entender el mecanismo que da origen ellas [3]. Ahora, si bien la inflación se ha venido estudiando asumiendo que ha durado lo suficiente como para que el espaciotiempo de Friedmann-Lemaître sea válido, no está mal partir de un modelo más general y luego ver bajo qué condiciones se recupera el FLRW durante la inflación [4]. Esto, unido al hecho de que desde mediados de la década de 1990 se hayan venido realizando mediciones de radiación cósmica de fondo que ponen en duda la isotropía del espacio, ha hecho que algunos cosmólogos se interesen en los modelos tipo Bianchi, llamados así en honor a Luigi Bianchi, quien los clasificó en 1987 basándose en la estructura de simetría que estos presentan. Estos modelos son homogéneos, mas no isotrópicos. El modelo más simple de ellos es el Bianchi I, que considera un factor de escala diferente para cada una de las tres direcciones ortogonales del espacio. Dicho esto, es interesante entonces ahora no sólo pensar en el estudio de las perturbaciones en una métrica como la FLRW, sino en una más general como la Bianchi I.

Yendo un poco más allá, sin embargo, es bien sabido que las perturbaciones han sido durante mucho tiempo una herramienta de gran importancia para la Física, puesto que permite encontrar soluciones aproximadas a partir de otras ya bien conocidas. Si bien esto no es algo nuevo y se suele trabajar bastante incluso en cursos introductorios de Electromagnetismo, Mecánica Cuántica o Mecánica Estadística, su aplicación en el

[1] Iniciales de Friedmann-Lemaître-Robertson-Walker.

ámbito de la relatividad general puede llegar a ser una tarea un poco más complicada, debido a la interrelación existente entre el espaciotiempo y la materia que este contiene. Esta interrelación hace que cualquier perturbación que se haga sobre la materia tenga un efecto sobre la geometría del espaciotiempo. Matemáticamente hablando, dado de que la Relatividad General se desarrolla en el lenguaje de la Geometría Diferencial, tenemos entonces que estudiar las perturbaciones implica perturbar la métrica. El problema adicional que surge aquí es que trabajar con una métrica perturbada y comparar con una «ideal» o de fondo implica aceptar implícitamente que hay una relación entre dos variedades que, en principio, son distintas, por lo que se hace necesario conocer tal relación, que además no es única, para poder trabajar. Este tipo de cosas pueden hacer que trabajar las perturbaciones aquí pueda llegar a ser algo complicado desde el punto de vista operativo, sobre todo cuando se desea estudiar la evolución tardía del universo, donde los efectos gravitatorios dejan de ser lineales y la descripción a primer orden ya no es suficiente, por lo que es necesario recurrir a órdenes superiores [2], [3], [5]. Vemos entonces que perturbar en Relatividad General puede transformarse en una tarea algo compleja.

Afortunadamente, con el avance del álgebra computacional, a día de hoy se han desarrollado herramientas tales como xPand [5], un paquete de Wolfram Mathematica especializado en el cálculo de perturbaciones cosmológicas a orden arbitrario. Este paquete funciona sobre la base de xAct, una suite libre dedicada al cálculo tensorial con un enfoque abstracto, es decir, sin elegir un sistema de coordenadas en particular [6], lo que la hace altamente versátil y expandible.

Ahora bien, en el grupo de investigación en Geometría, Gravitación y Cosmología del Departamento de Física de la Universidad del Valle se han venido explorando, ya desde hace algún tiempo, modelos con anisotropía, por lo que el empleo de herramientas computacionales especializadas es útil para la exploración de las perturbaciones de dichos modelos. Sin embargo, si bien el uso de xPand, xAct u otras herramientas de software similares puede resultar bastante simple o intuitivo para aquellas personas con algo de experiencia en Relatividad General o Geometría Diferencial, no todos los estudiantes de pregrado que se inician en estas áreas están familiarizados con ellas. Incluso los tutoriales que se encuentran sobre el tema, a menudo parten de la suposición de que el lector ya domina la base teórica necesaria, lo que hace que su lectura pueda resultar algo complicada, sobre todo para los recién iniciados. Es por eso que en este proyecto se busca brindar un acercamiento a la herramienta desde un punto de vista práctico y pedagógico al aplicarla al cálculo de las perturbaciones lineales de las ecuaciones de movimiento y de campo de Einstein para el caso del campo escalar canónico en el espaciotiempo de Bianchi I.

Para llevar a cabo esto, en la primera parte de este documento, en la sección 1, se presentan algunos elementos geométricos cuya comprensión permite aprovechar las instrucciones proporcionadas por xAct y xPand. En la sección 2, por su parte, se presentan los conceptos de Cosmología física con los que se va a trabajar en este caso. Estas dos secciones constituyen, por tanto, la base teórica sobre la cual se va a trabajar. En la sección 3 se da una revisión general de los paquetes de software que se utilizarán,

algunas de sus particularidades y las instrucciones básicas necesarias que se utilizan en la implementación. No se describen aquí todas las instrucciones con las que cuentan los paquetes, pues tal revisión podría constituir un proyecto completo en sí mismo, dada la complejidad de algunos de sus algoritmos y de la cantidad de instrucciones que presentan, lo cual va más allá del objetivo de este trabajo. En la sección 4, por su parte, se presenta la implementación como tal, en código, del cálculo en cuestión, detallando sus partes más importantes y mostrando los resultados obtenidos. El código completo se presenta en el apéndice C y como un archivo de Mathematica de acceso libre que puede ser descargado a través del enlace proporcionado en la referencia [7]. Finalmente, en la sección 5 se presentan algunas reflexiones sobre los resultados obtenidos y sobre la metodología utilizada durante el proyecto, mientras que en la sección 6 se presentan algunas conclusiones e ideas en las que es posible trabajar a partir de lo realizado en este trabajo.

1. Elementos de Geometría

En esta sección se dará un repaso de algunos conceptos matemáticos que permitirán establecer una conexión natural e intuitiva entre la geometría y las instrucciones trabajadas con xPand.

1.1. Superficies diferenciables

Podría decirse que una superficie diferenciable puede entenderse como una porción de $\mathbb{R}^2$ *inmersa* en $\mathbb{R}^3$ [2]. Ahora, de manera más formal, un subconjunto $S \subseteq \mathbb{R}^3$ es una superficie regular o diferenciable si y sólo si para todo punto p en S existe una vecindad U de p y un difeomorfismo ϕ, tal que

$$\phi : \Omega \to U,$$

donde $\Omega \subseteq \mathbb{R}^2$ [9]. A la pareja (U, ϕ) se le conoce como parametrización de S en p y básicamente describe la manera en que la porción de $\mathbb{R}^2$ está incrustada en $\mathbb{R}^3$, es decir, si forma un plano, un trozo de él o una esfera, por dar algunos ejemplos sencillos. Por otra parte, a la pareja (U, ψ), con $\psi = \phi^{-1}$ se le conoce como carta de S en p y es de utilidad a la hora de generalizar la noción de superficies a dimensiones superiores. Al conjunto de cartas $\{(U_i, \psi_i)\}$ se le conoce como atlas de S si la superficie está completamente recubierta por los elementos del conjunto, esto es, si $S = \cup_i U_i$.

De lo anterior se tiene que una curva α trazada en Ω se verá entonces reflejada, gracias a ϕ, en una curva β sobre la superficie S. No sólo eso, sino que como ϕ es un difeomorfismo, la velocidad de la curva α se verá también reflejada sobre la superficie en la velocidad de β, ahora por cuenta de la diferencial de ϕ. Esto permite entonces trazar curvas coordenadas en Ω que tengan sus correspondientes imágenes sobre S, lo cual facilita el trabajo sobre la superficie en una determinada región, ya que puede hacerse como si se tratase simplemente de una porción de $\mathbb{R}^2$.

Geometría de las superficies diferenciables

La geometría de las superficies diferenciales en un punto p se estudia básicamente a partir del espacio tangente a la superficie en dicho punto y de su vector normal. Veamos ahora cómo obtenemos una base de ese espacio tangente.

Sea $\alpha = (x^1, x^2)$ una curva en $\mathbb{R}^2$ parametrizada por unidad de longitud con origen en q. Sea también (U, ϕ) una parametrización de la superficie S en p, de tal manera que $p = \phi(q)$. La curva α tendrá como imagen sobre la superficie a una curva $\beta = \phi(x^1, x^2)$, cuya velocidad tendrá componentes $\boldsymbol{\phi_{x^1}}(q)$ y $\boldsymbol{\phi_{x^2}}(q)$, no necesariamente ortogonales, las cuales vienen dadas por

$$\boldsymbol{\phi_{x^1}}(q) = \left.\frac{\partial \phi}{\partial x^1}\right|_q \qquad \text{y} \qquad \boldsymbol{\phi_{x^2}}(q) = \left.\frac{\partial \phi}{\partial x^2}\right|_q .$$

[2] Cuando además no tiene intersecciones, se dice que está *encajada* [8].

Estos dos vectores forman una base del espacio tangente a S en p, denotado por $\mathcal{T}_pS$. Esta base no necesariamente es ortogonal y tampoco tiene por qué estar normalizada, por lo que cualquier distancia en las vecindades de p que desee medirse, tendrá esto reflejado.

Ahora, si se desea calcular distancias sobre la superficie, hay que asegurarse de que los puntos involucrados pertenezcan a U. Cuanto más preciso y fiel a la superficie se desee ser, más pequeño deberá ser U. Esto lleva a que los mejores elementos para calcular distancias sobre S sean los diferenciales. Si se tiene entonces una curva l sobre la superficie, su elemento diferencial vendrá dado por:

$$\mathrm{d}l = \frac{\partial l}{\partial x^1}\mathrm{d}x^1 + \frac{\partial l}{\partial x^2}\mathrm{d}x^2.$$

Con esto, el elemento de línea vendrá dado por:

$$\begin{aligned}\mathrm{d}l \cdot \mathrm{d}l = &\left(\frac{\partial l}{\partial x^1}\cdot\frac{\partial l}{\partial x^1}\right)\mathrm{d}x^1\mathrm{d}x^1 + \left(\frac{\partial l}{\partial x^1}\cdot\frac{\partial l}{\partial x^2}\right)\mathrm{d}x^1\mathrm{d}x^2 \\ &+ \left(\frac{\partial l}{\partial x^2}\cdot\frac{\partial l}{\partial x^1}\right)\mathrm{d}x^2\mathrm{d}x^1 + \left(\frac{\partial l}{\partial x^2}\cdot\frac{\partial l}{\partial x^2}\right)\mathrm{d}x^2\mathrm{d}x^2.\end{aligned}$$

Lo anterior se puede escribir de una forma más compacta introduciendo una forma bilineal simétrica definida positiva dada por

$$\begin{aligned}\boldsymbol{g} : \mathcal{T}_pS \times \mathcal{T}_pS &\to \mathbb{R} \\ (\boldsymbol{u}, \boldsymbol{v}) &\to \boldsymbol{u}\cdot\boldsymbol{v}.\end{aligned}$$

A esta forma bilineal se le conoce como **primera forma fundamental** de S y permite medir ángulos entre vectores, mientras que su forma cuadrática asociada, es decir, aquella en que $\boldsymbol{u} = \boldsymbol{v}$, permite medir normas. Con esto, el elemento de línea puede reescribirse como

$$\mathrm{d}l \cdot \mathrm{d}l = \sum_{i,j=1}^{2} g_{ij}\mathrm{d}x^i\mathrm{d}x^j,$$

donde:

$$g_{ij} = \boldsymbol{g}\left(\frac{\partial l}{\partial x^i}, \frac{\partial l}{\partial x^j}\right) = \frac{\partial l}{\partial x^i}\cdot\frac{\partial l}{\partial x^j}.$$

Vale la pena resaltar que g_{ij} brinda toda la información sobre la base del espacio tangente, ya que g_{11} y g_{22} dan cuenta de cómo es la magnitud los vectores $\frac{\partial l}{\partial x^1}$ y $\frac{\partial l}{\partial x^2}$, respectivamente; mientras que g_{12} y g_{21} llevan información del ángulo que hay entre ellos. Por esto se le conoce también como **métrica** de la superficie S. Adicionalmente, el hecho de que las velocidades de las curvas coordenadas formen una base del espacio tangente al punto, hace que la métrica permita calcular el producto punto de dos

vectores de dicho espacio. Sea $\boldsymbol{u}$ y $\boldsymbol{v}$ dos vectores del espacio tangente, se tiene que:

$$\begin{aligned}\boldsymbol{u}\cdot\boldsymbol{v} &= \left(u^1\boldsymbol{\phi}_{\boldsymbol{x^1}} + u^2\boldsymbol{\phi}_{\boldsymbol{x^2}}\right)\cdot\left(v^1\boldsymbol{\phi}_{\boldsymbol{x^1}} + v^2\boldsymbol{\phi}_{\boldsymbol{x^2}}\right)\\ &= \sum_{i,j=1}^{2} g_{ij}u^i v^j,\end{aligned} \tag{1.1}$$

donde ahora

$$g_{ij} = \boldsymbol{g}\left(\boldsymbol{\phi}_{\boldsymbol{x^i}}, \boldsymbol{\phi}_{\boldsymbol{x^j}}\right) = \boldsymbol{\phi}_{\boldsymbol{x^i}}\cdot\boldsymbol{\phi}_{\boldsymbol{x^j}}.$$

Ya con una noción de cómo se realizan las mediciones de ángulos (forma bilineal g_{ij}) y normas (forma cuadrática asociada a g_{ij}) sobre en una superficie, lo siguiente es recordar cómo se puede entender la geometría misma de ella. La noción más intuitiva es quizás la extrínseca. Ya anteriormente se mencionó que toda superficie es localmente orientable, por lo que es posible definir un vector normal a ella. La aplicación de Gauss asociada a cada punto debe coincidir entonces con el producto cruz normalizado de los vectores base del espacio tangente a S en p:

$$N(p) = \pm\left.\frac{\boldsymbol{\phi}_{\boldsymbol{x^1}}\times\boldsymbol{\phi}_{\boldsymbol{x^2}}}{\left|\boldsymbol{\phi}_{\boldsymbol{x^1}}\times\boldsymbol{\phi}_{\boldsymbol{x^2}}\right|}\right|_{\phi^{-1}(p)}.$$

El signo $\pm$ viene de que la superficie sea localmente orientable. Esto permite que se pueda elegir el vector normal en un sentido o en otro.

Ahora, de manera análoga a como la curvatura de una curva se estudia a partir de qué tanto cambia el vector normal asociado a ella, la curvatura de las superficies se estudia analizando qué tanto cambia el $N(p)$ de un punto a otro, por lo que el interés se traslada a analizar la diferencial de la aplicación de Gauss, que salvo por un signo menos, es lo que se conoce como aplicación de Weingarten u operador de forma:

$$\begin{aligned}\boldsymbol{A}_p : \mathcal{T}_pS &\longrightarrow \quad \mathcal{T}_pS\\ \boldsymbol{u} &\longrightarrow -\mathrm{d}\boldsymbol{N}_p(\boldsymbol{u}).\end{aligned}$$

Dado que este operador de forma tiene como punto de partida y de llegada el espacio tangente, puede ser expresado entonces en términos de la base de dicho espacio. Tenemos entonces que

$$\begin{aligned}\mathrm{d}\boldsymbol{N}_p(\boldsymbol{\phi}_{\boldsymbol{x^i}}) &= \left(\boldsymbol{\phi}_{\boldsymbol{x^1}}\cdot\mathrm{d}\boldsymbol{N}_p(\boldsymbol{\phi}_{\boldsymbol{x^i}})\right)\boldsymbol{\phi}_{\boldsymbol{x^1}} + \left(\boldsymbol{\phi}_{\boldsymbol{x^2}}\cdot\mathrm{d}\boldsymbol{N}_p(\boldsymbol{\phi}_{\boldsymbol{x^i}})\right)\boldsymbol{\phi}_{\boldsymbol{x^2}}\\ &= \left(\boldsymbol{\phi}_{\boldsymbol{x^1}}\cdot -\boldsymbol{A}_p(\boldsymbol{\phi}_{\boldsymbol{x^i}})\right)\boldsymbol{\phi}_{\boldsymbol{x^1}} + \left(\boldsymbol{\phi}_{\boldsymbol{x^2}}\cdot -\boldsymbol{A}_p(\boldsymbol{\phi}_{\boldsymbol{x^i}})\right)\boldsymbol{\phi}_{\boldsymbol{x^2}},\end{aligned}$$

con $i = 1, 2$. Esto se puede escribir también de manera más compacta introduciendo otra forma bilineal simétrica, $\boldsymbol{K}$, no necesariamente positiva definida, que se conoce como **segunda forma fundamental** [3] y que contiene información de cómo se curva

[3] Algunos textos reservan este nombre para la forma cuadrática asociada a esta forma bilineal. Esto no debe suponer un problema, debido a la correspondencia biunívoca entre ambas.

la superficie:

$$\begin{aligned} \boldsymbol{K} : \mathcal{T}_p(\mathcal{S}) \times \mathcal{T}_p(\mathcal{S}) &\longrightarrow \mathbb{R} \\ (\boldsymbol{u}, \boldsymbol{v}) &\longrightarrow \boldsymbol{u} \cdot \boldsymbol{A}_p(\boldsymbol{v}). \end{aligned} \tag{1.2}$$

Con esto se tiene que

$$\mathrm{d}\boldsymbol{N}_p(\boldsymbol{\phi}_{\boldsymbol{x}^i}) = \sum_{j=1}^{2} -K_{ij}\boldsymbol{\phi}_{\boldsymbol{x}^j},$$

donde

$$K_{ij} = \boldsymbol{K}(\boldsymbol{\phi}_{\boldsymbol{x}^i}, \boldsymbol{\phi}_{\boldsymbol{x}^j}) = \boldsymbol{g}(\boldsymbol{\phi}_{\boldsymbol{x}^i}, \boldsymbol{A}_p(\boldsymbol{\phi}_{\boldsymbol{x}^j})).$$

Vemos entonces que el cambio del vector normal está directamente relacionado con la manera en cómo se curva la superficie, estando esta información condensada dentro de la segunda forma fundamental.

A partir del determinante de las formas matriciales asociadas a la primera y segunda forma fundamental, puede definirse una cantidad conocida como curvatura gaussiana, k:

$$k = \frac{\det \boldsymbol{K}}{\det \boldsymbol{g}}.$$

Puede pensarse en esta cantidad como algo que condensa la información de las curvaturas asociadas a las curvas coordenadas incluidas en la segunda forma fundamental. La curvatura gaussiana da también entonces una medida de cómo se curva la superficie. Ahora, puede demostrarse que la curvatura gaussiana puede, de hecho, reescribirse en términos de únicamente la primera forma fundamental y sus derivadas [9]. Esto último implica que dicha curvatura, en un principio definida a partir de información basada en el ambiente como lo es su vector normal, es en realidad una cantidad intrínseca, por lo que no depende de cómo la superficie está inmersa en el ambiente. ¡Claro!, la curvatura se estudia a partir del cambio en el vector normal, pero dicho cambio está sobre el espacio tangente. Por tanto, tiene sentido que se pueda hacer geometría sobre la superficie usando elementos propios de ella. Es a esto precisamente a lo que se le conoce como Geometría intrínseca.

Como resumen, se tienen entonces tres conclusiones importantes:

1. La métrica o primera forma fundamental da la información de cómo se mide ángulos y distancias sobre la superficie.

2. La segunda forma fundamental lleva la información de cómo se curva la superficie.

3. Es posible hacer geometría sobre la superficie de manera intrínseca, es decir, a partir de elementos propios de ella.

1.2. Hipersuperficies encajadas en una variedad

Habiendo ya explorado algunas ideas intuitivas de lo que es una superficie y cómo podemos entenderla como un subconjunto de $\mathbb{R}^2$ inmerso en $\mathbb{R}^3$, se puede pasar a extender estas ideas a un formalismo un poco más abstracto, como lo es el de las variedades diferenciables. Una variedad diferenciable de n dimensiones es un conjunto de puntos localmente homeomorfo a $\mathbb{R}^n$ tal que para toda intersección de abiertos, existe una función diferenciable de cambio de coordenadas entre las coordenadas asociadas a ellos [10]. En adelante se trabajará en el caso de una variedad lorentziana, $\mathcal{M}$, de 4 dimensiones con signatura $(-,+,+,+)$.

Pushforward

De manera similar a como una superficie en $\mathbb{R}^3$ puede ser entendida como la imagen de un subconjunto de $\mathbb{R}^2$ dada por un mapeo que la «sumerja» en $\mathbb{R}^3$, en este caso una hipersuperficie Σ de $\mathcal{M}$ puede ser entendida como la imagen de una 3-variedad $\hat{\Sigma}$ que ha sido inmersa en $\mathcal{M}$ por la acción de un mapeo Φ, siendo este un homeomorfismo, para garantizar que Σ no se interseque consigo misma y que quede así **encajada** en $\mathcal{M}$ [4]. En este caso se establecerá además la condición de que Φ sea un difeomorfismo, para garantizar adicionalmente la diferenciabilidad. Tenemos entonces que $\Sigma = \Phi(\hat{\Sigma})$, donde $\Phi : \hat{\Sigma} \rightarrow \mathcal{M}$. Como se está trabajando con variedades, es posible definir de manera local un sistema de coordenadas $(\{x^i\})$ en $\hat{\Sigma}$ y uno $(\{x^{\mu\prime}\})$ en $\mathcal{M}$. Con esto, el mapeo Φ, en dichas coordenadas locales, puede ser escrito como

$$\begin{aligned}&\Phi : U \subset \hat{\Sigma} \rightarrow V \subset \mathcal{M}\\&(x^1, x^2, x^3) \rightarrow (x^{0\prime}, x^{1\prime}, x^{2\prime}, x^{3\prime}),\end{aligned}$$

donde U es un abierto de $\hat{\Sigma}$ y V es un abierto de $\mathcal{M}$. Este mapeo lleva entonces puntos de $\hat{\Sigma}$ a puntos de la variedad $\mathcal{M}$ y con ello, curvas a curvas. De la geometría diferencial de superficies en $\mathbb{R}^3$ se sabe que cuando se tiene una función f definida entre superficies S_1 y S_2, la diferencial de dicha función, queda definida entre los espacios tangentes de dichas superficies. Algo análogo ocurre con las variedades (ver figura 1). Dada una función definida entre variedades, en este caso Φ, esta induce una segunda función, Φ_*, conocida como **pushforward** o **aplicación progrediente** y que relaciona vectores definidos en el espacio tangente a un punto sobre la primera variedad con elementos del espacio tangente a un punto sobre la segunda:

$$\begin{aligned}&\Phi_* : \mathcal{T}_p(\hat{\Sigma}) \rightarrow \mathcal{T}_{\Phi(p)}(\mathcal{M})\\&(v^1, v^2, v^3) \rightarrow (v^{0\prime}, v^{1\prime}, v^{2\prime}, v^{3\prime}).\end{aligned}$$

[4] Esto es útil para para evitar que a un mismo punto se puedan asociar diferentes vectores normales no paralelos.

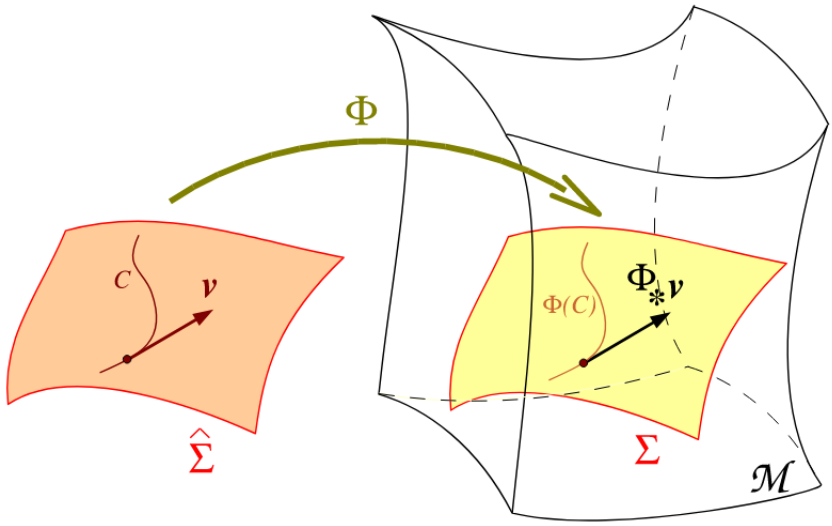

Figura 1: Una hipersuperficie Σ de $\mathcal{M}$ puede ser entendida como la imagen de una 3-variedad $\hat{\Sigma}$ que ha sido inmersa en $\mathcal{M}$ por la acción de un mapeo Φ. Imagen tomada de la referencia [11].

Pullback

Se sabe que los espacios tangentes a un punto en una variedad tienen un espacio dual asociado, el espacio cotangente. Si la aplicación progrediente relaciona los espacios tangentes, es natural preguntarse si esta induce algún tipo de aplicación que relacione los espacios cotangentes. En efecto, así como Φ induce un mapeo Φ_* que va de $\mathcal{T}_p(\hat{\Sigma})$ a $\mathcal{T}_{\Phi(p)}(\mathcal{M})$, este último también induce uno Φ^*, denominado **pullback** (por su nombre en Inglés), **retroacción** o **aplicación regrediente** y que va del espacio cotangente de $\mathcal{T}^*_{\Phi(p)}(\mathcal{M})$ al espacio cotangente $\mathcal{T}^*_p(\hat{\Sigma})$, llevando así covectores de $\mathcal{M}$ a covectores de $\hat{\Sigma}$:

$$\begin{aligned}\Phi^* : \mathcal{T}^*_{\Phi(p)}(\mathcal{M}) &\to \mathcal{T}^*_p(\hat{\Sigma}) \\ (\omega'_0, \omega'_1, \omega'_2, \omega'_3) &\to (\omega_1, \omega_2, \omega_3).\end{aligned}$$

En el caso de formas multilineales se tiene que dada una forma multilineal, $\boldsymbol{W}$, sobre $\mathcal{T}_p(\mathcal{M})$, se puede construir otra forma $\Phi^*\boldsymbol{W}$ sobre $\hat{\Sigma}$ como:

$$\Phi^*\boldsymbol{W}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n) = \boldsymbol{W}(\Phi_*\boldsymbol{v}_1, \ldots, \Phi_*\boldsymbol{v}_n),$$

siendo esto es válido para $(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n)$ en $\mathcal{T}_p(\mathcal{M})^n$.

De lo anterior se tiene que, dado un tensor métrico $\boldsymbol{g}$ sobre $\mathcal{M}$, es posible construir una métrica $\boldsymbol{\gamma}$ sobre la hipersuperficie $\hat{\Sigma}$ y, por tanto, sobre Σ, simplemente teniendo en cuenta que

$$\boldsymbol{\gamma} = \Phi^*\boldsymbol{g}. \tag{1.3}$$

Y dado que esta métrica depende de cómo es $\boldsymbol{g}$ y de la forma en que la hipersuperficie está embebida en $\mathcal{M}$, a $\boldsymbol{\gamma}$ se le conoce como **métrica inducida** por $\boldsymbol{g}$ sobre Σ o como primera forma fundamental de la hipersuperficie Σ [5].

1.2.1. Vector normal de una hipersuperficie tipo espacio

Un caso de particular interés es aquél en el que $\mathcal{M}$ es una variedad de Lorentz con signatura $(-,+,+,+)$ y la hipersuperficie posee una métrica inducida con signatura $(+,+,+)$. En tal caso se dice que Σ es una hipersuperficie tipo espacio.

En adelante, se asumirá que Σ es una hipersuperficie orientable[6] tipo espacio localmente descrita como un conjunto de nivel de un cierto campo escalar $t_s : \mathcal{M} \to \mathbb{R}$ [7], es decir:

$$\Sigma = \{p \in \mathcal{M} \mid t_s(p) = constante\}$$

De esta manera, se pueden elegir un sistemas de coordenadas locales en los que la aplicación progrediente se pueda escribir de la siguiente forma:

$$\begin{aligned} \Phi : \hat{\Sigma} &\to \mathcal{M} \\ (x^1, x^2, x^3) &\to (t_s(p), x^1, x^2, x^3). \end{aligned} \tag{1.4}$$

Consideremos ahora una foliación de hipersuperficies vinculadas a los distintos valores de t_s, de tal manera que Σ sea un miembro de ella. Dado que Σ es una superficie de nivel del campo escalar t_s, tenemos entonces que su gradiente, $\vec{\boldsymbol{\nabla}} t_s$, define una dirección normal ella. Y como no se ha impuesto ninguna restricción sobre su norma, esta puede ser distinta de la unidad, así que podemos definir un vector normal unitario, $\boldsymbol{n}_\pm$ dado por la normalización del gradiente de t_s, es decir,

$$\boldsymbol{n}_\pm \equiv \pm(-\vec{\boldsymbol{\nabla}} t_s \cdot \vec{\boldsymbol{\nabla}} t_s)^{-\frac{1}{2}} \vec{\boldsymbol{\nabla}} t_s.$$

El signo negativo dentro del paréntesis viene de que $\mathcal{M}$ es una variedad lorentziana y Σ es tipo espacio, por lo que el campo escalar asociado a la superficie de nivel no nula satisface la siguiente desigualdad:

$$\vec{\boldsymbol{\nabla}} t_s \cdot \vec{\boldsymbol{\nabla}} t_s < 0.$$

El signo $\pm$, por su parte, viene del hecho de que la hipersuperficie es orientable, lo cual da la libertad de elegir el sentido del vector normal.

[5] Este nombre se clarifica si se tiene en cuenta que este caso es conceptualmente análogo a construir la primera forma fundamental de una superficie abstracta haciendo un pullback de la métrica euclidiana de $\mathbb{R}^3$

[6] Esto es para que al aplicarlo al espaciotiempo haya una noción de dirección temporal válida de manera global.

[7] La s de t_s simplemente es un recordatorio de que se trata de un campo escalar y que no debe ser confundido con el tiempo físico.

Por otra parte, dado que el interés último será aplicar estos desarrollos al espacio-tiempo, conviene identificar t_s como un campo escalar que actúa a manera de coordenada temporal que incrementa su valor hacia el futuro. Resulta apropiado entonces definir un vector normal, $\boldsymbol{n}$, de tal manera que apunte en tal dirección. Tenemos entonces que:

$$\boldsymbol{n} \equiv -N\vec{\boldsymbol{\nabla}}t_s,$$

donde a $N \equiv (-\vec{\boldsymbol{\nabla}}t_s \cdot \vec{\boldsymbol{\nabla}}t_s)^{-\frac{1}{2}}$ y se le conoce como **función lapso**. Aplicando estos conceptos a la Física, N establece la relación entre el tiempo coordenado, t_s, y el tiempo físico, τ medido por un observador con cuadrivelocidad $\boldsymbol{n}$, es decir, un observador espacialmente inmóvil [11]. Es decir,

$$\delta\tau = N\delta t_s.$$

El valor de N dependerá entonces de si se trabaja con el tiempo cósmico, $\tau = t$, en cuyo caso $N = 1$; o el tiempo conforme, $\tau = \eta$, en donde ahora $N = a$, siendo a el factor de escala que gobierna la expansión del universo.

1.2.2. Descomposición de vectores del espacio tangente de $\mathcal{M}$

En el estudio de las superficies en $\mathbb{R}^3$, un vector tridimensional puede ser expresado localmente en términos de los vectores base del espacio tangente a una superficie y su vector normal. Esto, con algo de cuidado, puede trasladarse a las superficies abstractas [8] e incluso a las variedades de 4 dimensiones sobre las que estamos trabajando [12], pero para ello se necesita definir una nueva aplicación.

Proyector ortogonal

Se intuye que un vector en $\mathcal{T}_p\mathcal{M}$ puede ser descompuesto en una parte tangente a Σ en p y una parte paralela al vector normal. Sin embargo, no se cuenta hasta ahora con una aplicación directa que permita hacer tal descomposición. Veamos, tenemos hasta ahora dos formas de movernos entre Σ y $\mathcal{M}$, la aplicación progrediente y la aplicación regrediente. La primera lleva vectores de Σ a vectores de $\mathcal{M}$ y la segunda llevar covectores de $\mathcal{M}$ a covectores de Σ. Hasta hora no tenemos ninguna que permita llevar vectores de $\mathcal{M}$ a Σ, por lo que habrá que construirla. Para ello basta con hacer la retroacción de una 1-forma $\underline{\boldsymbol{v}}$ desde $\mathcal{M}$ hasta Σ y luego tomar el vector $\boldsymbol{v}$ asociado a ella. Es decir, hacemos primero:

$$\Phi^*(\underline{\boldsymbol{v}}) = \underline{\boldsymbol{v}} + \langle \mathbf{d}t_s, \boldsymbol{v}\rangle \mathbf{d}t_s.$$

Si se toma ahora el vector asociado a ambos lados, se llega a

$$\vec{\boldsymbol{\gamma}}(\boldsymbol{v}) = \boldsymbol{v} + \langle \mathbf{d}t_s, \boldsymbol{v}\rangle \vec{\boldsymbol{\nabla}}t_s,$$

donde $\vec{\boldsymbol{\gamma}}(\boldsymbol{v})$ es el dual asociado a $\Phi^*(\underline{\boldsymbol{v}})$. Expresando todo en términos del vector normal, se obtiene finalmente que:

$$\vec{\boldsymbol{\gamma}}(\boldsymbol{v}) = \boldsymbol{v} + \langle \underline{\boldsymbol{n}}, \boldsymbol{v} \rangle \boldsymbol{n}, \tag{1.5}$$

o en componentes:

$$\gamma^{\alpha}{}_{\beta} = \delta^{\alpha}{}_{\beta} + n^{\alpha} n_{\beta}.$$

A $\vec{\boldsymbol{\gamma}}$ se le conoce como **proyector ortogonal** o simplemente **aplicación de proyección** y tal como su nombre sugiere, proyecta un vector definido en el espacio tangente de $\mathcal{M}$ en el espacio tangente de la hipersuperficie Σ, es decir, $\vec{\boldsymbol{\gamma}} : \mathcal{T}_p(\mathcal{M}) \to \mathcal{T}_p(\Sigma)$, lo que a su vez permite definir la derivada covariante asociada a la métrica inducida como:

$$D_{\rho} T_{\mu_1 \ldots \mu_n} = \gamma^{\alpha}{}_{\rho} \gamma^{\beta_1}{}_{\mu_1} \ldots \gamma^{\beta_n}{}_{\mu_n} \nabla_{\alpha} T_{\beta_1 \ldots \beta_n},$$

que junto con la métrica inducida en sí misma, permiten hacer geometría sobre la hipersuperficie.

1.2.3. Descomposición del tensor métrico

Con lo visto hasta ahora, se tienen las herramientas necesarias para descomponer una métrica definida en $\mathcal{M}$ en términos de la métrica inducida sobre la hipersuperficie y del vector normal a ella, lo que también se conoce como descomposición $3+1$.

Para empezar, recordemos que el tensor métrico es una forma bilineal, por lo que dados $\boldsymbol{u}, \boldsymbol{v} \in \mathcal{T}_p\mathcal{M}$, debe cumplirse que:

$$\begin{aligned}
\boldsymbol{g}(\boldsymbol{u}, \boldsymbol{v}) &= \boldsymbol{g}(\vec{\boldsymbol{\gamma}}(\boldsymbol{u}) - (\boldsymbol{n} \cdot \boldsymbol{u})\boldsymbol{n}, \vec{\boldsymbol{\gamma}}(\boldsymbol{v}) - (\boldsymbol{n} \cdot \boldsymbol{v})\boldsymbol{n}) \\
&= \boldsymbol{g}(\vec{\boldsymbol{\gamma}}(\boldsymbol{u}), \vec{\boldsymbol{\gamma}}(\boldsymbol{v})) - (\boldsymbol{n} \cdot \boldsymbol{v}) \underbrace{\boldsymbol{g}(\vec{\boldsymbol{\gamma}}(\boldsymbol{u}), \boldsymbol{n})}_{0} - (\boldsymbol{n} \cdot \boldsymbol{u}) \underbrace{\boldsymbol{g}(\boldsymbol{n}, \vec{\gamma}(\boldsymbol{v}))}_{0} + (\boldsymbol{n} \cdot \boldsymbol{u})(\boldsymbol{n} \cdot \boldsymbol{v}) \underbrace{\boldsymbol{g}(\boldsymbol{n}, \boldsymbol{n})}_{-1} \\
&= \boldsymbol{g}(\vec{\boldsymbol{\gamma}}(\boldsymbol{u}), \vec{\boldsymbol{\gamma}}(\boldsymbol{v})) - (\boldsymbol{n} \cdot \boldsymbol{u})(\boldsymbol{n} \cdot \boldsymbol{v}),
\end{aligned}$$

donde se ha usado la ecuación 1.5 para descomponer $\boldsymbol{u}$ y $\boldsymbol{v}$ en términos de la proyección en Σ y del vector normal. Ahora, dada una base $(\{\boldsymbol{\partial_\mu}\})$ de $\mathcal{T}_p\mathcal{M}$, asociada a un sistema de coordenadas $(\{x^{\mu}\})$, si hacemos $\boldsymbol{u} = \boldsymbol{\partial_\mu}$ y $\boldsymbol{v} = \boldsymbol{\partial_\nu}$ en la ecuación anterior:

$$\boldsymbol{g}(\boldsymbol{\partial_\mu}, \boldsymbol{\partial_\nu}) = \boldsymbol{g}(\vec{\boldsymbol{\gamma}}(\boldsymbol{\partial_\mu}), \vec{\boldsymbol{\gamma}}(\boldsymbol{\partial_\nu})) - (\boldsymbol{n} \cdot \boldsymbol{\partial_\mu})(\boldsymbol{n} \cdot \boldsymbol{\partial_\nu}).$$

Si tenemos en cuenta que $\boldsymbol{g}(\boldsymbol{\partial_\mu}, \boldsymbol{\partial_\nu}) = g_{\mu\nu}$ y $(\boldsymbol{n} \cdot \boldsymbol{\partial_\mu})(\boldsymbol{n} \cdot \boldsymbol{\partial_\nu}) = n_\mu n_\nu$, entonces la métrica definida en $\mathcal{M}$ puede descomponerse como:

$$g_{\mu\nu} = \boldsymbol{g}(\vec{\boldsymbol{\gamma}}(\boldsymbol{\partial_\mu}), \vec{\boldsymbol{\gamma}}(\boldsymbol{\partial_\nu})) - n_\mu n_\nu, \tag{1.6}$$

Pero observemos que $\boldsymbol{g}(\vec{\boldsymbol{\gamma}}(\boldsymbol{\partial_\mu}), \vec{\boldsymbol{\gamma}}(\boldsymbol{\partial_\nu}))$ es una forma bilineal definida a partir de las proyecciones sobre Σ de los vectores $\boldsymbol{\partial_\mu}$ y $\boldsymbol{\partial_\nu}$, por lo que en realidad es una extensión a 4 dimensiones de otra forma bilineal definida en 3 dimensiones. Más aún, dado que $\boldsymbol{\partial_\mu}$ y $\boldsymbol{\partial_\nu}$ son vectores base, tenemos que la forma bilineal 3-dimensional a la que se hace

referencia es en realidad la métrica inducida, $\boldsymbol{\gamma}$, que fue introducida en la ecuación 1.3, por lo que $\boldsymbol{g}(\vec{\boldsymbol{\gamma}}(\boldsymbol{\partial_\mu}), \vec{\boldsymbol{\gamma}}(\boldsymbol{\partial_\nu}))$ no es otra cosa que la versión extendida de ella. Es decir:

$$\gamma_{4_{\mu\nu}} = \boldsymbol{\gamma_4}(\boldsymbol{\partial_\mu}, \boldsymbol{\partial_\nu}) \equiv \boldsymbol{g}(\vec{\boldsymbol{\gamma}}(\boldsymbol{\partial_\mu}), \vec{\boldsymbol{\gamma}}(\boldsymbol{\partial_\nu})). \tag{1.7}$$

Pero como en términos operativos la parte tiempo-tiempo de $\boldsymbol{\gamma_4}$ se anula y la distinción se puede hacer a partir del barrido de sus índices (griegos yendo de 0 a 3 o latinos yendo de 1 a 3), se optará por denotarlas igual, $\boldsymbol{\gamma}$, sin riesgo a confusión.

Llevando esto a la ecuación 1.6 tenemos que el tensor métrico puede expresarse en términos de la métrica inducida y el vector normal como:

$$g_{\mu\nu} = \gamma_{\mu\nu} - n_\mu n_\nu \qquad \text{ó} \qquad \boldsymbol{g} = \boldsymbol{\gamma} - \boldsymbol{n} \otimes \boldsymbol{n}. \tag{1.8}$$

Por último, observemos que la identificación de la versión extendida de la métrica inducida es un procedimiento que puede ser llevado a otras n-formas lineales y que puede entenderse como la acción de un mapeo $\vec{\boldsymbol{\gamma}}^*$ que al ser aplicado sobre una n-forma (3-dimensional) $\boldsymbol{W}$ definida en $\mathcal{T}_p^*(\hat{\Sigma})^n$, la lleva a una n-forma $\boldsymbol{W}_4 \equiv \vec{\boldsymbol{\gamma}}^* \boldsymbol{W}$ (4-dimensional) y que está definida en $\mathcal{T}_p^*(\mathcal{M})^n$. Esto es válido gracias a la identificación $\hat{\Sigma}/\Sigma$ y al puente establecido por la ecuación 1.5, razón por la cual se dice que este mapeo es inducido por el proyector ortogonal [11]. En adelante nos referiremos a él como el **extensor** o **aplicación de extensión** y se asumirá que las cantidades que estén definidas originalmente sobre Σ, pero con índices corriendo de 0 a 3 hacen referencia a su versión extendida.

Componentes del tensor métrico

Veamos ahora la forma de las componentes del tensor métrico $\boldsymbol{g}$ en la base $(\{\boldsymbol{\partial_\mu}\})$ mediante un elemento de línea sobre $\mathcal{M}$, el cual puede escribirse como

$$\begin{aligned} \mathrm{d}s^2 &= g_{\mu\nu}\ \mathrm{d}x^\mu\ \mathrm{d}x^\nu \\ &= g_{00}\ \mathrm{d}x^0\ \mathrm{d}x^0 + 2g_{0i}\ \mathrm{d}x^0\ \mathrm{d}x^i + g_{ij}\ \mathrm{d}x^i\ \mathrm{d}x^j, \end{aligned}$$

donde $g_{\mu\nu} = \boldsymbol{\partial_\mu} \cdot \boldsymbol{\partial_\nu} = \boldsymbol{g}\left(\boldsymbol{\partial_\mu}, \boldsymbol{\partial_\nu}\right)$. Si se identifica a t_s como x^0, en virtud de lo expresado en 1.4, su velocidad $\boldsymbol{\partial_{t_s}}$ identificará curvas de $\mathcal{M}$ espacialmente constantes, por lo que también está bien hacer $\boldsymbol{\partial_0} = \boldsymbol{\partial_{t_s}}$. Con esto se tiene entonces que $(\{\boldsymbol{\partial_\mu}\}) = (\boldsymbol{\partial_{t_s}}, \{\boldsymbol{\partial_i}\})$ y el elemento de línea $\mathrm{d}s^2$ puede reescribirse como:

$$ds^2 = (\boldsymbol{\partial_{t_s}} \cdot \boldsymbol{\partial_{t_s}})\ \mathrm{d}t_s^2 + 2(\boldsymbol{\partial_{t_s}} \cdot \boldsymbol{\partial_i})\ \mathrm{d}t_s\ \mathrm{d}x^i + (\boldsymbol{\partial_i} \cdot \boldsymbol{\partial_j})\ \mathrm{d}x^i\ \mathrm{d}x^j.$$

Para expresar todo en términos de la métrica inducida sobre la hipersuperficie y su vector normal es necesario descomponer también los vectores de la base $(\boldsymbol{\partial_{t_s}}, \{\boldsymbol{\partial_i}\})$. Sin embargo, por la forma en que esta ha sido construida, su parte espacial coincide con la de la hipersuperficie. Teniendo en cuenta esto, de las ecuaciones 1.3 y 1.7 se tiene que $(\boldsymbol{\partial_i} \cdot \boldsymbol{\partial_j}) = \gamma_{ij}$, por lo que el elemento de línea toma la siguiente forma:

$$ds^2 = (\boldsymbol{\partial_{t_s}} \cdot \boldsymbol{\partial_{t_s}})\ \mathrm{d}t_s^2 + 2(\boldsymbol{\partial_{t_s}} \cdot \boldsymbol{\partial_i})\ \mathrm{d}t_s\ \mathrm{d}x^i + \gamma_{ij}\ \mathrm{d}x^i\ \mathrm{d}x^j.$$

De aquí falta sólo reescribir el vector $\boldsymbol{\partial_{t_s}}$ en términos de sus componentes normal y tangente a la hipersuperficie, ya que en ningún momento se ha exigido que $\boldsymbol{\partial_{t_s}}$ sea ortogonal a Σ, así que podría no serlo (ver figura 2).

De la ecuación 1.5 se sabe que cualquier vector $\boldsymbol{v}$ de $\mathcal{T}_p\mathcal{M}$ puede ser expresado como

$$\boldsymbol{v} = -\langle \underline{\boldsymbol{n}}, \boldsymbol{v}\rangle \boldsymbol{n} + \vec{\gamma}(\boldsymbol{v}).$$

Aplicando esta descomposición a $\boldsymbol{\partial_{t_s}}$,

$$\begin{aligned}\boldsymbol{\partial_{t_s}} &= -\langle \underline{\boldsymbol{n}}, \boldsymbol{\partial_{t_s}}\rangle \boldsymbol{n} + \vec{\gamma}(\boldsymbol{\partial_{t_s}}) \\ &= -\langle -N\mathbf{d}t_s, \boldsymbol{\partial_{t_s}}\rangle \boldsymbol{n} + \langle \mathbf{d}x^\mu, \vec{\gamma}(\boldsymbol{\partial_{t_s}})\rangle \boldsymbol{\partial_i} \\ &= N\langle \mathbf{d}t_s, \boldsymbol{\partial_{t_s}}\rangle \boldsymbol{n} + \beta^i \boldsymbol{\partial_i} \\ &= N\boldsymbol{n} + \boldsymbol{\beta},\end{aligned}$$

donde $\beta^i = \langle \mathbf{d}x^\mu, \vec{\gamma}(\boldsymbol{\partial_{t_s}})\rangle$. Al vector $\boldsymbol{\beta} = \beta^i \boldsymbol{\partial_i} \equiv \vec{\gamma}(\boldsymbol{\partial_{t_s}})$, tangencial a Σ, se le conoce como **vector desplazamiento** o **shift vector**, por su nombre en Inglés, y vemos que básicamente determina qué tanto difiere el vector asociado a la parte temporal de la base coordenada que se ha elegido, respecto a N veces el vector normal a la hipersuperficie. O visto de otro modo, partiendo de un punto p y haciendo un pequeño cambio temporal δt_s, el vector desplazamiento dice qué diferencia hay entre arrastrar la hipersuperficie en dirección al vector normal y hacerlo a lo largo de la curva coordenada t_s asociada a $\boldsymbol{\partial_{t_s}}$.

Ya con lo anterior. podemos pasar a calcular los productos internos. Para la parte temporal se tiene entonces:

$$\begin{aligned}\boldsymbol{\partial_{t_s}} \cdot \boldsymbol{\partial_{t_s}} &= \langle N\underline{\boldsymbol{n}} + \beta_i \mathbf{d}x^i, N\boldsymbol{n} + \beta^j \boldsymbol{\partial_j}\rangle \\ &= -N^2 + \beta_i\beta^i.\end{aligned}$$

Ahora para los términos cruzados:

$$\boldsymbol{\partial_{t_s}} \cdot \boldsymbol{\partial_i} = \langle N\underline{\boldsymbol{n}} + \beta_j \mathbf{d}x^j, \boldsymbol{\partial_i}\rangle = \beta_i.$$

Con esto, se llega finalmente a que la descomposición del elemento de línea en términos de la métrica inducida, la función lapso y el vector de desplazamiento, puede expresarse como:

$$\mathrm{d}s^2 = (-N^2 + \beta_i\beta^i)\ \mathrm{d}t_s^2 + 2\beta_i\ \mathrm{d}t_s\ \mathrm{d}x^i + \gamma_{ij}\ \mathrm{d}x^i\ \mathrm{d}x^j. \tag{1.9}$$

1.2.4. Curvatura Extrínseca

Se sabe hasta ahora que la hipersuperficie Σ es constante en el tiempo coordenado t_s. Es claro que otro instante $t_s + \delta t_s$ tendrá otra hipersuperficie asociada $\Sigma_{t_s+\delta t_s}$ que no necesariamente tiene que coincidir con la primera. Es justo preguntarse cómo evoluciona entonces Σ o, más precisamente, $\boldsymbol{\gamma}$, ante un cambio infinitesimal de t_s. Sin embargo, de

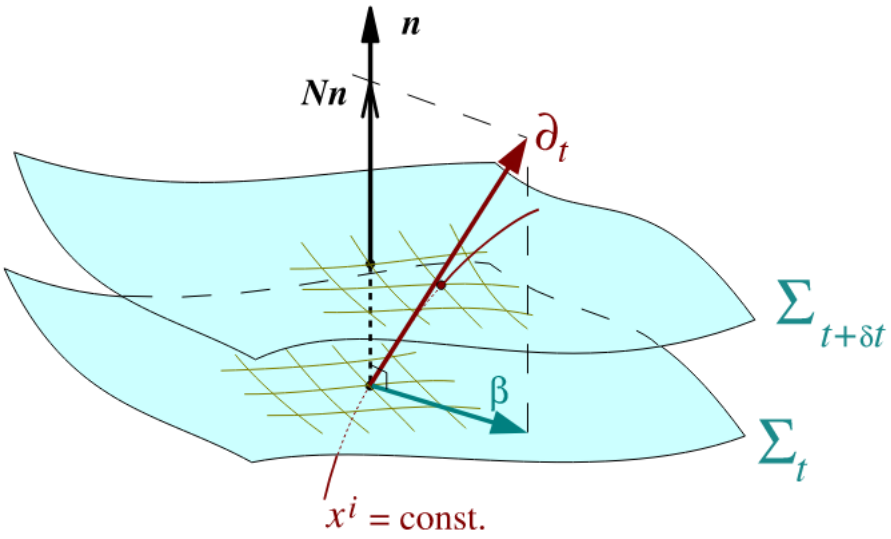

Figura 2: La curva coordenada asociada a $\boldsymbol{\partial_{t_s}}$ no necesariamente es paralela al vector normal. Por álgebra lineal, la única exigencia que hay sobre el vector $\boldsymbol{\partial_{t_s}}$ es que no sea generado por la base de la hipersuperficie. Cuando la curva coordenada es ortogonal a la hipersuperficie $\boldsymbol{\partial_{t_s}}$ es paralelo a $N\boldsymbol{n}$ y, por tanto, $\boldsymbol{\beta} = 0$. Imagen tomada de la referencia [11].

la sección 1.2.1 sabemos que los cambios en t_s están vinculados al vector normal. Preguntarnos entonces cómo son los cambios de $\boldsymbol{\gamma}$ respecto a t_s se traduce en preguntarnos cómo son sus cambios cuando nos movemos a lo largo del vector normal. En últimas, teniendo en cuenta que estos cambios son infinitesimales, estamos interesados en cómo es la derivada de Lie de γ a lo largo del vector normal $\boldsymbol{n}$, es decir, $\boldsymbol{\mathcal{L}_n\gamma}$.

Volviendo ahora al paralelismo establecido con una superficie en $\mathbb{R}^3$, recordamos que los cambios del vector normal a una superficie están vinculados a una forma bilineal que lleva la información de cómo es la curvatura extrínseca asociada a la superficie. Podría esperarse entonces que estudiar los cambios de $\boldsymbol{\gamma}$ respecto al vector normal esté asociado de alguna manera a una forma más abstracta de curvatura extrínseca de la hipersuperficie. Y, de hecho, así es. Puede demostrarse, como se ilustra de forma detallada en la referencia [11] que

$$\boldsymbol{K} = \frac{1}{2}\boldsymbol{\mathcal{L}_n\gamma},$$

donde a $\boldsymbol{K}$ se le conoce como **tensor de curvatura extrínseca** o **segunda forma fundamental** de Σ y se define como una forma bilineal de manera análoga a la de la ecuación 1.2, pero ahora para la hipersuperficie abstracta Σ, es decir,

$$\begin{aligned} \boldsymbol{K} : \mathcal{T}_p(\Sigma) \times \mathcal{T}_p(\Sigma) &\longrightarrow \mathbb{R} \\ (\boldsymbol{u}, \boldsymbol{v}) &\longrightarrow -\boldsymbol{\gamma}(\boldsymbol{u}, \nabla_{\boldsymbol{v}}\boldsymbol{n}), \end{aligned} \tag{1.10}$$

donde $\boldsymbol{\nabla}_{\boldsymbol{v}}$ es la derivada covariante asociada a la métrica $\boldsymbol{g}$ de $\mathcal{M}$ a lo largo del campo vectorial $\boldsymbol{v}$. Con esto se tiene entonces que el tensor de curvatura extrínseca caracteriza la manera en la que la hipersuperficie Σ está incrustada en la variedad $\mathcal{M}$.

Como se mencionó en la sección 1.2.3, la extensión de $\boldsymbol{K}$ a $\boldsymbol{K}_4$, es decir, a su versión en las 4 dimensiones de $\mathcal{M}$, se obtiene de manera análoga a como se obtuvo 1.7. En adelante, se omitirá el índice y se entenderá que si se está trabajando sobre $\mathcal{M}$, se hace referencia a la versión extendida de tal cantidad.

2. Elementos de Cosmología

La Cosmología Física estudia la evolución del universo a partir de dos elementos clave: las ecuaciones de Einstein y la ecuación de Boltzmann. Las primeras dan detalles sobre cómo es la relación entre el espaciotiempo y la materia, mientras que la segunda ofrece información de la evolución estadística de la materia en sí misma. Dados el propósito y el alcance de este proyecto, en este reporte sólo se abordan las ecuaciones de Einstein. Detalles sobre la ecuación de Boltzmann pueden encontrarse en la referencia [3].

2.1. Ecuaciones de Campo de Einstein

La relación entre el espaciotiempo y la materia viene dada por una acción S de la forma:

$$S = S_G + S_{mat}$$

siendo S_G y S_{mat} acciones correspondientes a los sectores de gravedad y materia, respectivamente. Estas pueden escribirse como:

$$S_G = \int \mathcal{L}_G(g_{\mu\nu}, \partial_\alpha g_{\mu\nu})\sqrt{-g}\ \mathrm{d}^4x,$$
$$S_{mat} = \int \mathcal{L}_{mat}(g_{\mu\nu}, \partial_\alpha g_{\mu\nu}, \{materia\})\sqrt{-g}\ \mathrm{d}^4x,$$

donde g es el determinante de la forma matricial del tensor métrico y donde $\{materia\}$ hace referencia a los campos de materia de los que pueda depender $\mathcal{L}_{mat}$, junto con sus derivadas. La forma más simple de describir la relación entre espaciotiempo y materia viene dada por la acción de Hilbert-Einstein:

$$S = \frac{1}{2\kappa}\int (R - \Lambda)\sqrt{-g}\ \mathrm{d}^4x + \int \mathcal{L}_{mat}\sqrt{-g}\ \mathrm{d}^4x,$$

donde κ es una constante [8], R es el escalar de Ricci y Λ es la constante cosmológica. Esta acción relaciona pues la geometría del espaciotiempo con la materia que este contiene.

Puede verificarse que aplicar el principio de acción estacionaria a la variación de S debida a los campos de materia reproduce las ecuaciones de movimiento de estos, mientras que al hacerlo sobre la variación debida al tensor métrico se llega a lo que se conoce como las **ecuaciones de campo de Einstein** [1]:

$$G_{\mu\nu} + \Lambda g_{\mu\nu} = \kappa\ T_{\mu\nu},$$

donde $G_{\mu\nu}$ es el tensor de Einstein y $T_{\mu\nu}$ es el tensor de energía-momento, ambos

[8] $\kappa = 8\pi G$, siendo G la constante de Newton.

definidos como:

$$T_{\mu\nu} \equiv -\frac{2}{\sqrt{-g}}\frac{\delta(\mathcal{L}_{mat}\sqrt{-g})}{\delta g^{\mu\nu}} = g_{\mu\nu}\mathcal{L} - 2\frac{\delta\mathcal{L}_{mat}}{\delta g^{\mu\nu}}$$
$$G_{\mu\nu} \equiv R_{\mu\nu} - \frac{1}{2}g_{\mu\nu}R,$$

siendo $R_{\mu\nu}$ el tensor de Ricci.

Dado que la constante cosmológica puede absorberse en el tensor de energía-momento, es común escribir las ecuaciones de Einstein simplemente como:

$$G_{\mu\nu} = \kappa\, T_{\mu\nu}, \tag{2.1}$$

Ahora, vale la pena resaltar que el lado izquierdo de esta expresión, vinculado al tensor de Ricci, puramente geométrico, describe la curvatura de la variedad espaciotemporal, mientras que el lado derecho contiene el tensor de energía-momento, que lleva consigo la información de la materia. De esta manera, las ecuaciones de Einstein establecen una relación clara y directa entre la curvatura del espaciotiempo y la materia que hay en él.

Fluido Perfecto

La forma más general de expresar el tensor de energía-momento en términos de la 4-velocidad u^μ del fluido viene dada por:

$$T_{\mu\nu} = \rho\, u_\mu u_\nu + \mathcal{P}\gamma_{\mu\nu} + q_\mu u_\nu + q_\nu u_\mu + \pi_{\mu\nu},$$

donde ρ corresponde a la densidad, $\mathcal{P}$ es la presión, q^μ es flujo de energía relativo a u^μ y $\pi_{\mu\nu}$ es la **presión anisotrópica**[9], que da cuenta de su viscosidad [1].

Ahora, la cuadrivelocidad del fluido cumple con que $u_\mu u^\mu = -1$ y satisface además la siguiente relación:

$$\nabla_\mu u_\nu = \frac{1}{3}\Theta\gamma_{\mu\nu} + \sigma_{\mu\nu} - u_\mu\dot{u}_\nu + \omega_{\mu\nu},$$

donde $\dot{u}_\nu$ es la derivada respecto al tiempo cósmico de u_ν; Θ es la traza de $\nabla_\mu u_\nu$ y describe cómo es la distorsión isotrópica, por lo que se le conoce como **expansión**; $\sigma_{\mu\nu}$ es el tensor de corte o cizalladura, que no tiene traza y es ortogonal a la cuadrivelocidad; y $\omega_{\mu\nu}$ corresponde a la vorticidad y describe la rotación del flujo de materia.

Ahora, un fluido perfecto es aquél en el cual $\pi_{\mu\nu} = 0$ y $q^\mu = 0$. Si el fluido no es sometido a fuerzas no gravitacionales, las líneas de flujo son en realidad geodésicas [2], por lo que $\dot{u}_\nu = 0$. Con esto se tiene entonces que para un fluido ideal que sólo está siendo afectado por la gravedad, el tensor de energía-momento toma la forma:

$$T_{\mu\nu} = \rho\, u_\mu u_\nu + P\gamma_{\mu\nu} = \begin{pmatrix} -\rho & 0 & 0 & 0 \\ 0 & \mathcal{P} & 0 & 0 \\ 0 & 0 & \mathcal{P} & 0 \\ 0 & 0 & 0 & \mathcal{P} \end{pmatrix}, \tag{2.2}$$

[9]También conocida como **estrés anisotrópico** o **tensión anisotrópica**.

mientras que la derivada covariante de u_ν queda como:

$$\nabla_\mu u_\nu = \frac{1}{3}\Theta\gamma_{\mu\nu} + \sigma_{\mu\nu} + \omega_{\mu\nu}.$$

Con esto vemos entonces que si la vorticidad es cero en cualquier instante, la cuadrivelocidad define hipersuperficies ortogonales a ella, por lo $\nabla_\mu u_\nu$ puede ser identificada con la curvatura extrínseca introducida en la sección 1.2.4, con lo que se obtiene finalmente que:

$$K_{\mu\nu} = \nabla_\mu u_\nu = \frac{1}{3}\Theta\gamma_{\mu\nu} + \sigma_{\mu\nu}, \tag{2.3}$$

donde queda establecido que Θ corresponde a la traza de $K_{\mu\nu}$, mientras que $\sigma_{\mu\nu}$ corresponde a su parte sin traza. Para poder aprovechar esta relación entre curvatura extrínseca y corte, en adelante se asumirá que se está trabajando con un fluido perfecto con vorticidad cero que sólo está siendo afectado por la gravedad.

2.2. Espaciotiempo FLRW

La expansión del universo es un hecho que ha sido corroborado experimentalmente en numerosas ocasiones y puede ser modelado mediante la métrica de Friedmann-Lemaître-Robertson-Walker (en adelante FLRW), que en su forma euclidiana en coordenadas comóviles puede ser escrita como:

$$(g^{FLRW})_{\mu\nu} = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & a^2(t) & 0 & 0 \\ 0 & 0 & a^2(t) & 0 \\ 0 & 0 & 0 & a^2(t) \end{pmatrix}, \tag{2.4}$$

donde t es el tiempo cosmológico y $a(t)$ es el factor de factor de escala que caracteriza la expansión y cuya evolución se estudia a partir de las ecuaciones de Einstein y la composición de los constituyentes del universo [3]. Nótese que esta métrica puede descomponerse usando la ecuación 1.9 para un caso particular en donde $N = 1$, $\beta_i = 0$ y donde $\boldsymbol{\gamma}$ es una métrica riemanianna euclidiana que podemos denotar por $\boldsymbol{\gamma}^{FLRW}$, como recordatorio de que es inducida por la de FLRW. Se tiene entonces que

$$\begin{aligned}(ds^2)^{FLWR} &= -dt^2 + (\gamma^{FLRW})_{ij}\mathrm{d}x^i\mathrm{d}x^j \\ &= -dt^2 + a^2(t)(g^{MK})_{ij}\mathrm{d}x^i\mathrm{d}x^j,\end{aligned}$$

siendo $\boldsymbol{g}^{MK}$ la métrica de Minkowski.

La forma explícita de las ecuaciones de Einstein en la métrica FLRW plana para el caso de un fluido perfecto (ecuación 2.2) se conoce como ecuaciones de Friedmann y son dos:

$$\begin{aligned}H^2 &= \frac{8\pi G}{3}\rho, \\ \frac{\ddot{a}}{a} &= -\frac{4\pi G}{3}\left(3\mathcal{P} + \rho\right),\end{aligned}$$

donde $H \equiv \frac{\dot{a}}{a}$ es el parámetro de Hubble y $\dot{a} = \frac{\mathrm{d}a}{\mathrm{d}t}$ y donde la primera ecuación corresponde a la componente tiempo-tiempo, mientras que la segunda surge de la combinación de esta con la traza de la componente espacio-espacio.

Para el caso del campo escalar canónico, dado por la densidad lagrangiana

$$\mathcal{L}_{mat} = -\frac{1}{2}\nabla_\mu\phi\nabla^\mu\phi + V(\phi),$$

las ecuaciones de Friedmann toman la forma:

$$H^2 = \frac{\kappa}{3}\left[\frac{1}{2}\dot{\phi}^2 + V(\phi)\right],$$
$$\dot{H} = -\frac{\kappa}{2}\dot{\phi}^2,$$

mientras que la ecuación de movimiento puede escribirse como:

$$\ddot{\varphi} + 3H\dot{\varphi} + a^2 V_\varphi(\varphi) = 0,$$

que por tratarse de campo escalar, nos referiremos a ella como ecuación de Klein-Gordon.

2.3. Espaciotiempo Bianchi I

Inspeccionando un poco la métrica FLRW a partir de las componentes de su forma matricial, se puede notar no sólo que su base es ortogonal, sino que describe un universo isotrópico, puesto que la norma de sus vectores espaciales es la misma, $a(t)$. La forma más simple de describir un universo no isotrópico es utilizando curvas coordenadas cuyas velocidades sigan siendo ortogonales pero de norma distinta. Esta forma de describir el universo es lo que se conoce como modelo espaciotemporal Bianchi I. El elemento de línea que describe este modelo viene dado por:

$$(\mathrm{d}s^2)^{AI} = -\mathrm{d}t^2 + \sum_{i=1}^{3} a_i^2(t)(\mathrm{d}x^i)^2,$$

expresión que puede ser reescrita como:

$$(\mathrm{d}s^2)^{AI} = -\mathrm{d}t^2 + \sum_{i,n,k=1}^{3} \frac{a^6(t)}{2(a_n)^2(a_k)^2}|\epsilon_{ink}|(\mathrm{d}x^i)^2,$$

donde $a(t) \equiv \left(\prod_{i=1}^{3} a_i(t)\right)^{1/3}$ y ϵ_{ink} es el símbolo de Levi-Civita. De aquí se observa que la métrica FLRW plana constituye un caso particular de la métrica de Bianchi I para cuando los factores de escala en cada dirección son iguales. Si se denota por $(\gamma^{AI})_{ij}(t)$ a la métrica inducida por Bianchi I sobre el espacio, se puede escribir:

$$(\mathrm{d}s^2)^{AI} = -\mathrm{d}t^2 + (\gamma^{AI})_{ij}(t)\mathrm{d}x^i\mathrm{d}x^j.$$

Siguiendo lo realizado en el caso de FLRW, se puede pensar la métrica espacial como una transformación conforme de otra que llamaremos γ_{ij}, de tal forma que:

$$(\gamma^{AI})_{ij}(t) = a^2(t)\gamma_{ij}(t).$$

Por tanto:

$$(\mathrm{d}s^2)^{AI} = -\mathrm{d}t^2 + a^2(t)\gamma_{ij}(t)\mathrm{d}x^i\mathrm{d}x^j.$$

Y si además se introduce el tiempo conforme η de manera tal que $\mathrm{d}t \equiv a\mathrm{d}\eta$ llegamos a que

$$(\mathrm{d}s^2)^{AI} = a^2(\eta)[-\mathrm{d}\eta^2 + \gamma_{ij}(\eta)\mathrm{d}x^i\mathrm{d}x^j].$$

Pero aquí se observa que el segundo factor del lado derecho no es más que otro caso particular de la ecuación 1.9 para un sistema de coordenadas distinto. Así las cosas, bien puede decirse simplemente que el tensor métrico, $(g^{AI})_{\mu\nu}$, correspondiente al modelo Bianchi I, viene dado por una transformación conforme de la forma

$$(g^{AI})_{\mu\nu} = a^2(\eta)\bar{g}_{\mu\nu}(\eta), \tag{2.5}$$

donde $\bar{g}_{\mu\nu}(\eta)$ es un tensor métrico al que llamaremos métrica conforme. Conocer esto es importante a la hora de trabajar con xPand, como se explica en la sección 3.3.

Algo que vale la pena destacar de esta forma de ver la métrica Bianchi I como una transformación conforme de $\bar{g}_{\mu\nu}(\eta)$ es que la expansión está completamente contenida en el factor $a^2(\eta)$, por lo que la hipersuperficie con métrica inducida γ_{ij}, asociada a la descomposición $3+1$ de $\bar{g}_{\mu\nu}(\eta)$ no tiene expansión isotrópica, por lo que $\Theta = 0$, que al reemplazarlo en la ecuación 2.3, permite concluir que $K_{ij} = \sigma_{ij}$.

Como detalle final, puede notarse que con la introducción del tiempo conforme, las ecuaciones de Friedmann en el modelo FLRW plano, por ser un caso particular del Bianchi I, pueden ser reescritas como:

$$\mathcal{H}^2 = \frac{\kappa}{3}\left[\frac{1}{2}\phi'^2 + V(\phi)a^2(\eta)\right], \tag{2.6}$$

$$\mathcal{H}' = -\frac{\kappa}{3}\left[\phi'^2 - V(\phi)a^2(\eta)\right], \tag{2.7}$$

mientras que la ecuación de Klein-Gordon se expresaría como:

$$\varphi'' + 2\mathcal{H}\varphi' + a^2 V_\varphi(\varphi) = 0, \tag{2.8}$$

donde $\mathcal{H} \equiv \frac{a'}{a}$ y los símbolos primados indican derivada de Lie a lo largo del vector normal a la hipersuperficie asociada a la descomposición $3+1$ de $\bar{g}_{\mu\nu}(\eta)$.

2.4. Perturbaciones Cosmológicas

En la sección 2.1 se habló de cómo la acción de Hilbert-Einstein establece una relación entre la geometría del espaciotiempo y la materia que este contiene. Esto se ve reflejado en las ecuaciones de Einstein. Es claro entonces que una perturbación en los campos de materia causará una perturbación en la geometría de la variedad, lo que se verá reflejado en forma de perturbaciones en la métrica. Con esto en mente, una métrica perturbada a primer orden puede escribirse como:

$$g_{\mu\nu} = g^0_{\mu\nu} + \delta g_{\mu\nu}, \tag{2.9}$$

siendo $g_{\mu\nu}$ la métrica física que queremos aproximar, $g^0_{\mu\nu}$ la **métrica de fondo** o **background**, por su nombre en Inglés, y $\delta g_{\mu\nu}$ la perturbación. Sin embargo, algo adicional que hay que tener en cuenta es que los tensores métricos $g^0_{\mu\nu}$ y $g_{\mu\nu}$ están asociados a dos variedades diferenciables diferentes. Y si bien en la ecuación 2.9 se asume que existe algún tipo de correspondencia entre ellas, lo cierto es que tal correspondencia no es única, lo cual conlleva a que exista una cierta libertad a la hora de establecerla. A esto se le conocer con el nombre de **libertad de calibre** o **gauge**. Para ver esto, consideremos el caso de interés para este trabajo, aquel en el que la métrica de fondo corresponde al modelo Bianchi I. Para esto reemplazamos $g^0_{\mu\nu}$ en la ecuación 2.9 por $(g^{AI})_{\mu\nu}$:

$$\begin{aligned} g_{\mu\nu} &= g^0_{\mu\nu} + \delta g_{\mu\nu} \\ &= (g^{AI})_{\mu\nu} + \delta g_{\mu\nu} \\ &= a^2(\eta)\bar{g}_{\mu\nu}(\eta) + \delta g_{\mu\nu}, \\ &= a^2(\eta)\left[\bar{g}_{\mu\nu}(\eta) + \delta g^c_{\mu\nu}\right], \end{aligned} \tag{2.10}$$

donde $\delta g^c_{\mu\nu} \equiv a^{-2}(\eta)\delta g_{\mu\nu}$. Ahora, la descomposición $3+1$, dada por la ecuación 1.9, aplicada a esta métrica perturbada, puede escribirse de la siguiente manera:

$$(ds^2) = a^2(\eta)\left[-(1+2A)\mathrm{d}\eta^2 + 2\hat{B}_i \mathrm{d}x^i \mathrm{d}\eta + (\gamma_{ij} + h_{ij})\mathrm{d}x^i \mathrm{d}x^j\right], \tag{2.11}$$

donde el escalar A, las componentes vectoriales $\hat{B}_i$ y el tensor h_{ij} corresponden a la descomposición de la perturbación $\delta g^c_{\mu\nu}$.

Continuando con la descomposición de la métrica, se sabe, por el teorema de Helmholtz [13], que un vector espacial puede ser expresado como la suma de un vector B_i sin divergencia y el gradiente de un campo escalar B, es decir:

$$\hat{B}_i = D_i B + B_i, \text{ con } D_i B^i = 0, \tag{2.12}$$

donde D_i es la derivada covariante asociada a γ_{ij} y donde el escalar B y el vector B_i se denotan con la misma letra para resaltar el hecho de que surgen de la descomposición de $\hat{B}_i$.

La extensión de este teorema a tensores espaciales de rango 2 se conoce como descomposición SVT[10]. Aquí, un tensor espacial simétrico de rango dos como h_{ij} puede ser reescrito en términos de aportes tensoriales, vectoriales y escalares, de tal manera que:

$$h_{ij} = 2C\left(\gamma_{ij} + \frac{\sigma_{ij}}{\mathcal{H}}\right) + 2D_iD_jE + 2D_{(i}E_{j)} + 2E_{ij}, \tag{2.13}$$

con $D_iE^{ij} = 0$, $E^i{}_i = 0$ y $D_iE^i = 0$. De manera similar al caso anterior, aquí $E,\ E_i,\ E_{ij}$ se nombran igual para resaltar que provienen de la misma descomposición SVT. Por su parte, la inclusión del corte en esta expresión obedece a una cuestión de conveniencia que se menciona en las líneas siguientes.

Con esto podemos hacer una descomposición total de la métrica 2.11 en términos de escalares, vectores y tensores, pero estos últimos ahora sin divergencia:

$$\begin{aligned}(ds^2) = a^2(\eta)\Bigg[&- (1+2A)\mathrm{d}\eta^2 + 2(D_iB + B_i)\mathrm{d}x^i\mathrm{d}\eta \\ &+ \left\{(1+2C)\gamma_{ij} + 2C\frac{\sigma_{ij}}{\mathcal{H}} + 2D_iD_jE + 2D_{(i}E_{j)} + 2E_{ij}\right\}\mathrm{d}x^i\mathrm{d}x^j\Bigg].\end{aligned} \tag{2.14}$$

Consideremos ahora una transformación infinitesimal de coordenadas de la forma

$$x^\mu \to \hat{x}^\mu = x^\mu - \zeta^\mu(\{x^\nu\}), \tag{2.15}$$

donde ζ^μ son las componentes de un campo vectorial.

Como se trata de una transformación infinitesimal, la métrica física transforma a partir de la derivada de Lie a lo largo del vector auxiliar, es decir:

$$\hat{g}_{\mu\nu} = g_{\mu\nu} + \mathcal{L}_\zeta g_{\mu\nu}.$$

Con esto, su perturbación transforma como

$$\delta\hat{g}_{\mu\nu} = \delta g_{\mu\nu} + \mathcal{L}_\zeta g^0_{\mu\nu}.$$

Si se aplica ahora una descomposición en ζ^μ dada por

$$\begin{aligned}&\zeta^0 = T(x^i,\eta),\\ &\zeta^i = D^iL(x^i,\eta) + L^i(x^i,\eta), \text{ con } D_iL^i = 0,\end{aligned}$$

puede demostrarse entonces, siguiendo el esquema planteado en la referencia [4], que las reglas de transformación de los aportes escalares, debidas a un cambio de coordenadas

[10]Siglas en Inglés de Scalar-Vector-Tensor.

como el de la ecuación 2.15, expresadas en el espacio de Fourier (ver sección 2.5), vienen dadas por:

$$A \to A + T' + \mathcal{H}T, \tag{2.16}$$

$$B \to B - T + \frac{(k^2 L)'}{k^2}, \tag{2.17}$$

$$C \to C + \mathcal{H}T, \tag{2.18}$$

$$E \to E + L. \tag{2.19}$$

Las contribuciones vectoriales, por otra parte, transforman como

$$B_i \to B_i + \gamma_{ij}(L^j)' - 2ik^j\sigma_{mj}\left(\gamma^m{}_i - \frac{k^m k_i}{k^2}\right)L, \tag{2.20}$$

$$E_i \to E_i + L_i. \tag{2.21}$$

Y para el caso de las contribuciones tensoriales, debido a la forma en que se introdujo el corte en la ecuación 2.13, estas transforman como:

$$E_{ij} \to E_{ij}.$$

De esta última expresión se dice que E_{ij}, al no verse afectada ante tales transformaciones, es una **invariante de calibre**. Pero esta no es la única cantidad que puede cumplir esta característica. De hecho, si bien las otras variables vinculadas la descomposición de la métrica no son invariantes de calibre, pueden combinarse sus transformaciones (ecuaciones 2.16-2.21) para hallar otras que sí lo sean. Un ejemplo de ellas son las que se ofrecen en la referencia [4], que vienen dadas por:

$$\Phi \equiv A + \frac{1}{a}\left[a\left(B - \frac{(k^2 E)'}{k^2}\right)\right]', \tag{2.22}$$

$$\Psi \equiv -C - \mathcal{H}\left[B - \frac{(k^2 E)'}{k^2}\right], \tag{2.23}$$

$$\Phi_i \equiv B_i - \gamma_{ij}(E^j)' + 2ik^j\sigma_{mj}\left(\gamma^m{}_i - \frac{k^m k_i}{k^2}\right)E. \tag{2.24}$$

Estas variables son una extensión al modelo Bianchi I de las definidas originalmente por Bardeen en 1980 para la métrica FLRW [14], las cuales pueden recuperarse cuando $\sigma_{ij} = 0$. Nótese que con esto ahora Φ, Ψ y Φ_i absorben a A, B, C, E y B_i, pasando de tener cuatro variables escalares y dos vectoriales a dos escalares y una vectorial, lo que muestra que podemos eliminar dos grados de libertad escalares y uno vectorial de las variables originales. Es a esto precisamente a lo que se le conoce como **libertad de calibre**, a la posibilidad de fijar algunas cantidades según la necesidad. Geométricamente hablando, dado que estas variables hacen parte de la métrica perturbada, eliminar algunas de ellas se traduce en elegir, en la variedad perturbada, unas coordenadas que

cumplan unas características concretas. Hay varias maneras de hacer esto, pero en este trabajo sólo se utilizará lo que se conoce como **calibre de Newton**, que consiste en eliminar los grados de libertad escalares, B y E, de tal manera que en el límite de no expansión con corte cero se recupere la ecuación de Poisson que satisface el potencial de la gravedad newtoniana [3]. Como también podemos eliminar un grado de libertad vectorial, en este trabajo el calibre de Newton estará dado por:

$$\begin{cases} B = E = 0 \\ B^i = 0, \end{cases} \tag{2.25}$$

Con esta elección no sólo se recupera la ecuación de Poisson, sino que, geométricamente hablando, se hace que en el punto de trabajo de la variedad perturbada la coordenada temporal sea ortogonal a las coordenadas espaciales.

Para el caso de la materia también pueden definirse variables invariantes de calibre. En el caso particular del campo escalar canónico, por ejemplo, se puede construir la variable χ [4], que viene dada por:

$$\chi \equiv \delta\varphi + \left[B - \frac{(k^2 E)'}{k^2}\right] \varphi'. \tag{2.26}$$

Obsérvese que si se trabaja en el gauge de Newton en Bianchi I, se tiene entonces que $\chi = \delta\varphi$, hecho que será aprovechado en la implementación en código de la sección 4.3 para escribir las ecuaciones de las perturbaciones en términos de χ.

Para resumir, dados dos puntos, ambos en variedades distintas, una de fondo y otra perturbada, cada una de ellas con una métrica asociada, la libertad de calibre no es otra cosa que aquella de la cual se dispone a la hora de unir o relacionar tales puntos. O, visto de otro modo, como la libertad remanente con la que se cuenta para elegir un sistema de coordenadas en la variedad perturbada habiendo ya elegido uno sobre la variedad de fondo y habiendo identificado un punto con el otro. De esta manera, se puede entonces trabajar ya sea con la métrica 2.14 tal como está o eligiendo un calibre en particular, es decir, eliminando algunos grados de libertad a conveniencia. El gauge de Newton, por su parte, es un caso particular en donde la eliminación los grados de libertad escalares se hace de tal forma que en el límite de no expansión se recupere la gravedad newtoniana. Y finalmente, las variables invariantes de calibre son cantidades, que al no verse alteradas por las transformaciones de gauge, son útiles no sólo para trabajar con ellas directamente, sino para utilizarlas a manera de puente para moverse entre un calibre y otro.

2.5. Espacio de Fourier

Es habitual en muchos casos, tanto en el modelo de Friedmann como en el Bianchi I, no realizar los cálculos y/o análisis de las perturbaciones directamente en su forma diferencial, sino en términos de los modos de Fourier, $\{k^i\}$. Una de las bondades que ofrece esto es que permite convertir las ecuaciones diferenciales en ecuaciones algebraicas, lo

cual facilita un poco la parte operativa no sólo en el trabajo manual, sino también en el desarrollo de rutinas de cálculo en herramientas computacionales como las que se emplean en este caso. Ahora, si bien en este trabajo la manipulación de las ecuaciones se hace sobre todo en su forma diferencial con el fin de mostrar el uso de la instrucciones incorporadas en el software, es útil tener nociones de cómo se puede implementar el espacio de Fourier en la herramienta, ya que esta no tiene incorporado un concepto de integración y sólo manipula tensores abstractos, lo que imposibilita el uso o adaptación de la transformada de Fourier propia de Mathematica. Por lo anterior, es necesario implementarla a partir de reglas de transformación y tensores definidos con los mismos paquetes. Para hacer esto, se presentará aquí un breve resumen de las ecuaciones que serán de utilidad para este proceso. Para un análisis mucho más detallado del trabajo con el espacio de Fourier en el modelo Bianchi I y de sus diferencias respecto al caso de Friedmann-Lemaître, consúltese la sección II de la referencia [4].

Dado un campo escalar, $f(\eta, x^i)$, su descomposición en los modos de Fourier viene dada por

$$f(\eta, x^i) = \frac{1}{(2\pi)^{3/2}} \int \hat{f}(\eta, k_j) e^{\mathrm{i}k_j x^j} \mathrm{d}^3 k_j.$$

O, de manera equivalente, mediante su transformación inversa:

$$\hat{f}(\eta, k_i) = \frac{1}{(2\pi)^{3/2}} \int f(\eta, x^j) e^{-\mathrm{i}k_j x^j} \mathrm{d}^3 x^j,$$

donde $k_i = k\hat{k_i}$ es el covector de onda de norma k en la dirección del vector unitario $\hat{k}_i$. Como se verá en la implementación (sección 4), estos son dos de los elementos que se deberá definir en el programa para trabajar en el espacio de Fourier.

Algo interesante aquí es que esta transformación tiene la propiedad de hacer que las derivadas covariantes puramente espaciales D_i, asociadas a γ_{ij}, pasen a ser $\mathrm{i}k_i$ [13], lo que debe aprovecharse luego como una regla de reemplazo en el código. El covector de onda cumple además con que $(k_i)' = 0$ y a partir de él se define el contravector $k^i = \gamma^{ij} k_j$. Y dado que γ_{ij} depende del tiempo, se tiene que

$$(k^i)' = -2\sigma^{ij} k_j \neq 0.$$

El cambio temporal del vector unitario, por su parte, viene dado por

$$(\hat{k}^i)' = (\sigma^{jl} \hat{k}_j \hat{k}_l) \hat{k}^i - 2\sigma^{ij} \hat{k}_j,$$

lo que también será aprovechado en la herramienta.

Por otra parte, un vector V_i en el espacio de Fourier puede ser expresado como:

$$V_i = k_i V + \bar{V}_i,$$

donde $k^i\bar{V}_i = 0$. Esto último significa que $\bar{V}_i$ está dentro un plano ortogonal a k^i, por lo que decimos que es transversal. Sea $\{e^{1i}, e^{2i}\}$[11] una base de dicho plano (que también debe definirse en el software), es posible entonces descomponer el vector $\bar{V}_i$ como:

$$\begin{aligned}\bar{V}_i(\eta, k_i) &= \left(e^{1j}V_j(\eta, k_i)\right) e^1{}_i + \left(e^{2j}V_j(\eta, k_i)\right) e^2{}_i \\ &= V_1(\eta, \hat{k}_i)e^1{}_i + V_2(\eta, \hat{k}_i)e^2{}_i, \end{aligned} \tag{2.27}$$

donde además los vectores base ortogonales a k^i cumplen con:

$$(e^a_i)' = -\sum_{b=1,2} \sigma_{jl}e^j_a e^l_b e^b{}_i + 2\sigma_{ij}e^j_a.$$

De manera análoga, un tensor de rango dos ortogonal a $\hat{k}^i$ puede ser descompuesto como

$$V_{ij}(\eta, k_i) = \sum_{\lambda=+,\times} V_\lambda(\eta, k^i)\varepsilon^\lambda{}_{ij}(\hat{k}_i), \tag{2.28}$$

donde a ε^+_{ij} y $\varepsilon^\times_{ij}$ (a incorporarse también en la rutina) se les conoce como tensores de polarización y vienen dados por:

$$\varepsilon^\lambda_{ij} = \frac{e^1_i e^1_j - e^2_i e^2_j}{\sqrt{2}}\delta^\lambda{}_+ + \frac{e^1_i e^2_j - e^2_i e^1_j}{\sqrt{2}}\delta^\lambda{}_\times. \tag{2.29}$$

Puede probarse, además, que dado un tensor espacial simétrico V_{ij} en el espacio de Fourier, este puede descomponerse como [4]:

$$V_{ij} = \frac{1}{3}\left[\gamma^{ab}V_{ab}\right]\gamma_{ij} + \frac{3}{2}\left[Z^{ab}V_{ab}\right]Z_{ij} + 2\hat{k}_{(i}\left[P^a{}_{j)}\hat{k}^b V_{ab}\right] + \Lambda^{ab}_{ij}V_{ab}, \tag{2.30}$$

donde

$$\begin{aligned} Z^i{}_j &= \hat{k}^i\hat{k}_j - \frac{1}{3}\delta^i{}_j, \\ P_{ij} &= e^1{}_i e^1{}_j + e^2{}_i e^2{}_j = \gamma_{ij} - \hat{k}_i\hat{k}_j, \\ \Lambda^{ab}_{ij} &= P^a{}_i P^b{}_j - \frac{1}{2}P^{ab}P_{ij}. \end{aligned}$$

[11] Téngase en cuenta que esta base es propia del modo k^i. Si se tiene otro modo k^j, con $j \neq i$, entonces este tendrá su propia base asociada. Además, los índices 1 y 2 no deben entenderse como índices tensoriales, pues son sólo etiquetas que usamos para identificar los dos vectores. Estos pueden aparecer arriba o abajo sin que pierdan su carácter de etiqueta. El índice i, por su parte, sí es de carácter tensorial, por estar asociado a k^i, así que debe tratarse como tal.

Habiéndose dicho todo lo anterior, pueden usarse las ecuaciones 2.27 y 2.28 para expresar la descomposición del vector E_i y el tensor E_{ij} en términos de la base adaptada, ya que por construcción son ortogonales a k^i. Se tiene entonces:

$$E_i = E_1 \; e^1{}_i + E_2 \; e^2{}_i, \tag{2.31}$$

$$E_{ij} = \sum_{\lambda=+,\times} E_\lambda \varepsilon^\lambda{}_{ij}, \tag{2.32}$$

donde E_1, E_2, E_+ y $E_\times$ serán otras de las cantidades que junto con los vectores de la base adaptada y los tensores de polarización, deberán definirse en la implementación en código.

Si se desea trabajar también con las variables de Bardeen será necesario descomponer además el vector Φ_i:

$$\Phi_i = \Phi_1 \; e^1{}_i + \Phi_2 \; e^2{}_i, \tag{2.33}$$

donde Φ_1 y Φ_2 pasarían a sumarse entonces a la lista de definiciones obligadas.

Por último, si se aplica la descomposición dada por 2.30 a σ_{ij}, se puede verificar que se llega finalmente a que el corte puede ser reescrito en términos de la base $\{\hat{k}_i, e^1{}_i, e^2{}_i\}$ como

$$\sigma_{ij} = \frac{3}{2}\left[\hat{k}_i\hat{k}_j - \frac{1}{3}\gamma_{ij}\right]\sigma_{\parallel} + 2\sum_{a=1,2}\sigma_{Va}\hat{k}_{(i}e^a{}_{j)} + \sum_{\lambda=+,\times}\sigma_{T\lambda}\varepsilon^\lambda{}_{ij}, \tag{2.34}$$

donde

$$\begin{aligned}\sigma_{\parallel} &= \sigma_{ij}\hat{k}^i\hat{k}^j,\\ \sigma_{T\lambda} &= \sigma_{ij}\varepsilon_\lambda{}^{ij},\\ \sigma_{Va} &= \sigma_{ij}\hat{k}^i e_a{}^j,\end{aligned}$$

siendo estas cantidades los últimos elementos que será necesario definir en el código para llevar las ecuaciones de las perturbaciones al espacio de Fourier, incluso aunque xPand no lo haga directamente. En la siguiente sección se explorarán entonces las herramientas de software que se utilizarán y las instrucciones más importantes para llevar a cabo el objetivo.

3. Software

El software que se propone como herramienta de cómputo es xPand [5], [15], un paquete de Mathematica enfocado en el cálculo de perturbaciones cosmológicas. Este paquete se basa, a su vez, en xTensor y xPert [16], ambos subpaquetes de la distribución xAct [6], una suite dedicada al cálculo tensorial. En esta sección se describen los comandos básicos necesarios para realizar la implementación. En particular, se describen instrucciones correspondientes a las siguientes versiones:

- xTensor 1.2.0.
- xPert 1.0.6.
- xPand 0.4.3.

Todo corriendo sobre Wolfram Mathematica 13.1. Más detalles de los recursos utilizados pueden encontrarse en el apéndice B.

3.1. xTensor

Este paquete es el que se enfoca en la manipulación tensorial como tal. Lo hace de manera abstracta, es decir, sin trabajar en ningún sistema de coordenadas en particular. Su funcionamiento se basa en la idea de que trabajar con tensores es trabajar con espacios tangentes y cotangentes de variedades diferenciables, por lo que el flujo de trabajo básico con esta herramienta inicia siempre por definir la variedad sobre la cual se desarrolla todo el análisis, luego la métrica principal con la que vamos a trabajar y finalmente los tensores que se desea manipular. Los comandos con los que se realizan estas acciones son:

- `DefManifold[M, n, {a, b, c, ...}]`

  Define una variedad diferenciable M de n dimensiones con índices $\{a, b, c, \dots\}$. Dado que xTensor opera de manera abstracta, los índices definidos en la variedad deben ser entendidos también como índices abstractos [17], por lo que no representan coordenadas, sino el tipo de tensor.

- `DefMetric[signdet, g[-a, -b], CD]`

  Define un tensor métrico g_{ab} y una derivada covariante asociada, sin torsión, de nombre «CD» y que puede ser aplicada luego a una expresión `expr` escribiendo `CD[ind][expr]` o `CD[ind]@expr`, donde `ind` es el índice de la derivada. La opción «`signdet`» corresponde al signo del determinante de la métrica y debe reemplazarse por -1, 1 o 0 dependiendo de si la métrica es de signo negativo, positivo o nulo, respectivamente.

- `DefTensor[T[-m,n], M]`

  Define un tensor ${T_m}^n$ de rango 2 sobre la variedad M. Al trabajar con tensores, los índices siempre deben ser tratados como argumentos de funciones comunes de Mathematica. Los índices contravariantes se representan con signos positivos, mientras que los covariantes por signos negativos. Si no tiene índices, entonces es un tensor escalar.

Algunos comandos de utilidad en el uso de xTensor son:

- `?xAct`xTensor`*`

  No es propiamente un comando de xTensor, sino de Mathematica. Nos muestra todas las funciones definidas dentro del paquete. Haciendo clic sobre cada una de ellas se obtiene una breve explicación sobre su funcionamiento. Esto también se logra escribiendo la instrucción sobre la cual se desea información y anteponiéndole un signo de cierre de pregunta, de tal manera que si se necesita información sobre la función `ToCanonical`, se pueden obtener detalles de ella simplemente usando el comando `?ToCanonical`.

- `expr//MyInstruction`
  o también `MyInstruction@expr`

  Esta es sintaxis propia de Mathematica y, por su practicidad, será muy usada en el código. Denota que a `expr` se le va a aplicar la orden `MyInstruction`. Es equivalente entonces a hacer a decir que se ejecutará la función `MyFunction` con `expr` como su argumento, es decir, `MyFunction[expr]`

- `ToCanonical[expr]`

  Reorganiza los índices de la expresión matemática `expr` de acuerdo con las condiciones de simetría de los tensores involucrados. Este comando realiza tareas como intercambiar índices mudos o subirlos y bajarlos, por ejemplo, según sea necesario.

- `ContractMetric[expr, metric]`

  Contrae en `expr` todas las instancias encontradas de la métrica, `metric`. En caso de que haya más de una métrica definida y esta última opción sea un listado de métricas, se ejecutará el comando de manera secuencial para cada una de ellas. Si no se especifica cuál métrica contraer, el comando las contraerá todas.

- `DefConstantSymbol[symbol]`

  Define `symbol` como una cantidad que se comportará como constante real ante cualquier forma de derivación.

- `DefScalarFunction[f]`

  Define `f` como una función escalar de uno o varios argumentos escalares.

- `LieD[v][expr]`
  o `LieD[v]@expr`

  Aplica a `expr` la derivada de Lie a lo largo del campo vectorial contravariante `v`.

- `expr/.MiRegla`
  o también `ReplaceAll[expr,MiRegla]`

  No siendo una orden de xPand, sino de Mathematica, la instrucción «`/.`» se usa para aplicar la regla de transformación o reemplazo dada por `MiRegla` a `expr`. `MiRegla` debe ser una instrucción o lista de instrucciones de la forma {`lhs->rhs`} o {`lhs:>rhs`}, donde `rhs` indica el valor por el cual deberá ser reemplazado `lhs`. Si `rhs` es una instrucción ejecutable y la regla se usa con "`->`", entonces `rhs` se evaluará antes de hacer el reemplazo. En caso de usarse "`:>`" se dice que la regla es retardada y `rhs` se evaluará al final.

- `expr//.MiRegla`

  También propio de Mathematica, esta instrucción hace lo mismo que la anterior, pero aplicando `MiRegla` de forma repetida hasta que `expr` no se vea afectada por ella.

- `MakeRule[{lhs,rhs}]`

  Instrucción de xTensor que crea una regla retardada de reemplazo que convierte `lhs` en `rhs`, pero subiendo y bajando índices en los tensores, para considerar así todas las posibles reglas equivalentes. La regla generada puede usarse con la instrucción "`/.`" explicada previamente.

- `AutomaticRules[symbol,MyRules]`

  Define sobre `symbol` una regla o listado de reglas `MyRules` que serán aplicadas por el programa de manera automática, sin intervención del usuario, cada vez que se encuentren en una expresión. Un ejemplo típico podría ser que si un determinado vector tridimensional $\hat{v}$ es unitario, el usuario podría estar interesado en reemplazar toda aparición de $\hat{v} \cdot \hat{v}$ por 1.

- `$Metrics`

  Muestra todas las métricas definidas.

- `PrintAs ->` "ϕ"

  Esta instrucción se usa dentro de las órdenes de tipo definición (aquellas que empiezan por `Def`) para indicar que el elemento que se está definiendo debe presentarse en pantalla de una manera específica. En este caso, como ϕ. Se usa sobre todo para garantizar la legibilidad de los resultados. Puede usarse como instrucción independiente, en cuyo caso se hace `PrintAs[NombreDeLaDefinición]^=`"ϕ".

- `Ricci[CD][a, b]`

  Con este comando se invoca el tensor de Ricci asociado a la derivada covariante `CD`. Las letras `a` y `b` corresponden a los índices. Como todo tensor trabajado con xTensor, si el índice aparece con signo positivo indica que son contravariantes. Si aparecen con un signo menos indica que son covariantes. En este caso, por ejemplo, `a` y `b` son índices contravariantes.

- `RicciScalar[ ]`

  Con este comando invocamos el escalar de Ricci a la hora de calcular.

- `RicciToEinstein[expr,covd]`

  Reescribe `expr` de tal manera que todos los tensores de Ricci asociados a la derivada covariante de `covd` queden expresados en términos del tensor de Einstein.

- `ScreenDollarIndices[expr]`

  Cuando se ejecutan instrucciones de cálculo con xTensor, a menudo el programa utilizará índices mudos de manera automática. Podría así darse el caso en que internamente estos índices entren en conflicto en operaciones en donde dos de ellos tengan el mismo nombre, pero hagan referencia a sumas distintas. Un ejemplo de esto se daría al trabajar en una variedad en donde sólo se han definido dos índices, a y b. En tal caso, una expresión como $A_a B^a C_b D^b \nabla_c E^c$ no sería válida, ya que la letra c no se ha definido como índice. Para usar letras válidas, el programa intentaría hacer ser algo como $A_a B^a C_b D^b \nabla_a E^a$, pero claramente aquí habría un conflicto, ya que el índice a sólo puede aparecer una vez arriba y una abajo. Para evitar esto, el software hará algo como $A_{a\$00001} B^{a\$00001} C_b D^b \nabla_{a\$00002} E^{a\$00002}$. Esto indicaría que aunque los dos índices llevan el mismo nombre, en realidad son diferentes. Esto puede causar, sin embargo, que en un resultado final aparezca algo como $A_{a\$00001} B^{a\$00001} + \nabla_{a\$00002} E^{a\$00002}$. Pero en este caso ya no hay riesgo de conflicto, puesto que los índices pertenecen a términos diferentes, por lo que esta expresión puede ser reescrita simplemente como $A_a B^a + \nabla_a E^a$. Lo mismo en una situación en la que se tenga algo como $A_{a\$00001} B^{a\$00001} \nabla_{a\$00002} E^{a\$00002}$, que podría ser reescrita simplemente como $A_a B^a \nabla_b E^b$. Esto es precisamente lo que hace la función ScreenDollarIndices, presentar los resultados omitiendo esos indicadores adicionales y reescribiendo todo, siempre que sea posible, en términos de los índices definidos al introducir la variedad con el comando `DefManifold`.

  Este comando, de tipo misceláneo, resulta entonces bastante útil a la hora de garantizar la legibilidad de los resultados obtenidos con xTensor.

- `STFPart[expr,metric]`

  Entrega la parte simétrica sin traza de `expr` respecto a la métrica `metric`.

- `$Tensors`

  Muestra todos los tensores definidos en la sesión.

- `VarD[fieldPert, CD][exprPert]`

  Asumiendo un funcional que se anula al integrar respecto a un cierto volumen, esta instrucción calcula la derivada variacional de `exprPert` respecto a `fieldPert`, tomando a CD como la derivada que se usará en la integración por partes. Debido a la forma en que trabaja xTensor y de cómo interpreta y trabaja la derivada funcional, `exprPert` y `fieldPert` deben ser perturbaciones definidas y/o calculadas previamente con xPert.

3.2. xPert

Este paquete se encarga del cálculo y la manipulación de perturbaciones tensoriales a orden arbitrario. Funciona bajo el entorno de xTensor y es utilizado internamente por xPand para construir las perturbaciones en diferentes métricas. La principal diferencia entre xPert y xPand es que el primero trabaja sin definir una métrica de fondo en particular, mientras que xPand trabaja con métricas de fondo concretas.

Dado que xPert funciona sobre xTensor, el flujo de trabajo con este paquete debe iniciar igual que el descrito en la sección 3.1, definiendo primero la variedad, luego la métrica y luego los tensores de interés.

Algunos comandos de xPert que serán utilidad son:

- `?xAct`xPert`*`

  Muestra todas las funciones definidas dentro de xPert. Al igual que para xTensor, aquí se puede dar clic en cada una de ellas para obtener una breve explicación sobre su funcionamiento.

- `DefMetricPerturbation[metric, metricpert, ϵ]`

  Este comando define, a partir de la métrica `metric`, un nuevo tensor `metricpert[LI[n], -a, -b]` con `a` y `b` como índices covariantes y con una etiqueta LI[n] que identifica el orden «`n`» de perturbación. Con esto, `metricpert` debe interpretarse entonces como la perturbación a n-ésimo orden del tensor métrico `metric`, mientras que ϵ es el parámetro perturbativo.

- `DefTensorPerturbation[Tpert[LI[n], inds], T[inds]]`

  Esta instrucción define un nuevo tensor `Tpert[LI[n],inds]` como una perturbación a orden n del tensor `T[inds]` con índices `inds`.

- `Perturbation[expr, n]`

  Este comando entrega la perturbación de `expr` a orden n, utilizando para ello la regla de Leibniz.

- `ExpandPerturbation[expr]`

  Con esta función se expanden todas las perturbaciones de `expr` en términos de las perturbaciones del tensor métrico.

3.3. xPand

Como se mencionó en la sección 3.2, xPand es el paquete encargado de hacer las perturbaciones cosmológicas en métricas de fondo concretas. Utiliza la descomposición 3+1 descrita en la sección 1.2 y en su versión 0.4.2 opera con las métricas de Minkowski, FLRW (plana o con curvatura) y Bianchi I, A y B. Este paquete funciona sobre el entorno de xTensor y utiliza xPert para la definición de las perturbaciones, por lo que en este caso el flujo de trabajo inicia de la misma forma que para los casos anteriores, definiendo una variedad y luego una métrica. Una característica de xPand es que trabaja en métricas concretas sin recurrir realmente a la elección de una base en particular, por lo cual no llega a trabajar propiamente en términos de las componentes de los tensores, sino en sus formas abstractas. Esto hace que no tenga que recurrir al uso de herramientas adicionales creadas para tal fin como xCoba [18], también de la suite xAct. Para lograr esto, utiliza el enfoque geométrico descrito en la sección 1.2, defiendo una variedad a la cual vincula una métrica ambiente general con ayuda de xTensor. Sobre esta define también un vector abstracto normal a una hipersuperficie y, a partir de estas dos cantidades, define la versión en cuatro dimensiones de la métrica inducida sobre ella (ecuaciones 1.8 y 1.7). Finalmente define una métrica física como una transformación conforme de la métrica ambiente general definida inicialmente (ecuación 2.5) y basa su cálculo perturbativo en tal cantidad, logrando así su cometido.

Ya en el trabajo con la herramienta, la primera métrica que se define, la ambiente, será la que se tomará como métrica conforme y, en el caso particular de Bianchi I, debe ser entendida como el tensor $\bar{g}_{\mu\nu}(\eta)$ en las ecuaciones 2.5 y 2.10, que junto con el cuadrado del factor de escala, forman entonces la métrica de fondo del espaciotiempo físico [5]. El programa no trabaja entonces directamente sobre la métrica física de fondo, sino sobre $\bar{g}_{\mu\nu}(\eta)$. Y a partir de una transformación conforme obtiene la otra, misma que luego es perturbada al orden deseado, aplicando la descomposición SVT vista en la sección 2.4 para expresar la métrica de Bianchi I en forma de la ecuación 2.14, pero siempre en términos de la extensión a 4 dimensiones de la métrica inducida por el ambiente sobre la hipersuperficie.

Con esto en mente, dado que xPand no incluye el factor de escala al definir la métrica inducida, la expansión en la hipersuperficie generada es cero, ya que esta queda completamente contenida en $a(\eta)$, como se explicó en la sección 2.3. Se tiene entonces que al trabajar con xPand en espaciotiempos Bianchi I (y para sus formas más generales [5]),

$$K_{ij} = \sigma_{ij},$$

mientras que para el modelo FLRW y Minkowski, se tendrá entonces que $K_{ij} = 0$.

3.3.1. Comandos de xPand

Veremos a continuación varios comandos correspondientes al paquete xPand.

- `?xAct`xPand`*`

Muestra todas las funciones definidas dentro del paquete. Funciona igual que sus análogas para xTensor y xPert.

- `SetSlicing[`$\boldsymbol{g}$`,`$\boldsymbol{n}$`,nNorm,`$\boldsymbol{\gamma}$`,cd,{cdPost,cdPre},SpaceTimeType]`

  Este comando realiza la descomposición $(N-1)+1$ a una variedad de fondo de dimensión N y con métrica (conforme) de fondo $\boldsymbol{g}$. El comando construye una métrica inducida $\boldsymbol{\gamma}$, junto con su correspondiente derivada covariante «`cd`» y sus respectivas representaciones `cdPost` y `cdPre`, además del vector normal $\boldsymbol{n}$. La opción «nNorm» permite establecer la norma del vector normal, que por defecto es -1. El parámetro `SpaceTimeType` define el tipo de métrica de la variedad ambiente. Las configuraciones posibles de `SpaceTimeType` son:

Minkowski	Métrica de Minkowski
FLFlat	Métrica FLRW plana
FLCurved	FLRW con curvatura
BianchiI	Espaciotiempos Bianchi tipo I.
BianchiA	Para espaciotiempos Bianchi A.
BianchiB	Espaciotiempos Bianchi en su forma más general

  Esta instrucción define también, de forma automática, las reglas de conmutación entre la derivada de Lie a lo largo del vector normal a la hipersuperficie y la derivada covariante asociada a la métrica inducida sobre ella.

- `DefMetricFields[g,gpert,h,parameter]`

  Este comando define una métrica perturbada «`gpert`» a partir de una métrica de fondo «`g`». Y dado que el interés está en trabajar con una descomposición de la variedad, es necesario especificar cuál es la métrica «`h`» que el ambiente induce sobre la hipersuperficie. La opción `parameter` es opcional y define el símbolo que será usado como parámetro de la expansión en series. Por defecto está configurado como ϵ.

- `DefMatterFields[uf,ufpert,h,parameter]`

  Esta función es análoga a la anterior, sólo que para el caso de la materia. Define un campo vectorial perturbado «`ufpert`» a partir de su valor de fondo «uf». Los parámetros «`h`» y «`parameter`» son los mismos que para el caso anterior.

- `DefProjectedTensor[name[indices], h,`
  `SpaceTimesOfDefinition->{«Background»,«Perturbed»},`
  `TensorProperties->{«SymmetricTensor»,«Traceless»,«Transverse»}]`

  Define un tensor espacial, `name`, en la hipersuperficie correspondiente a la métrica inducida `h`, de tal manera que satisfaga la descomposición SVT.

  Las opciones `SpaceTimesOfDefinition` y `TensorProperties` no son obligatorias, ya que sus valores por defecto son los que se muestran arriba. La primera de ellas

determina si el tensor se define sobre la variedad de fondo o sobre la variedad perturbada; la segunda, se usa para describir las propiedades del tensor espacial definido.

- `ToxPand[expr,gpert,uf,ufpert,h,"gauge",order]`

  Este comando ejecuta una secuencia de tres pasos:

  1. Realiza una transformación conforme sobre `expr` con el factor de escala «a» (ecuación 2.5), determinando así la métrica física de fondo. Este paso se omite cuando se trabaja con la métrica de Minkowski, puesto que no involucra expansión.
  2. Perturba `expr` a orden `order` usando xPert.
  3. Tomando a `h` como la métrica conforme inducida, separa el resultado según las reglas especificadas por el `gauge` impuesto en la ecuación 2.14.

  La palabra `gauge` de ser reemplazada por uno de los términos clave admitidos: `AnyGauge` [12], `FluidComovingGauge`, `ScalarFieldComovingGauge`, `FlatGauge`, `IsoDensityGauge`, `NewtonGauge` y `SynchronousGauge`.

  Vale la pena hacer la aclaración de que el calibre de Newton que admite xPand en Bianchi I no coincide exactamente con el de la ecuación 2.25. Esto lo podemos verificar ejecutando `SetSlicing` en un espaciotiempo Bianchi I y luego perturbando el tensor métrico a primer orden con `ToxPand`, eligiendo como calibre la opción `NewtonGauge`. Al hacer esto encontramos que xPand no elimina B_i, sino E_i, lo cual no supone un problema, porque la restricción sobre el calibre de Newton no está en la parte vectorial, sino en la escalar: hacer $E = B = 0$, que es lo que hace el programa. De igual manera, con el fin de mantener las definiciones introducidas en este trabajo, en la sección 4 se define una regla que permite utilizar el calibre de Newton introducido en la ecuación 2.25 para el modelo Bianchi I.

- `ExtractComponents[expr,h,ListOfProjectors]`

  Extrae de `expr` la componente dada por `ListOfProjectors`, que para tensores de rango dos puede ser {"`Time`","`Time`"}, {"`Time`","`Space`"} o {"`Space`","`Space`"}, mientras que para vectores puede ser {"`Time`"} o {"`Space`"}.

- `VisualizeTensor[expr,h]`

  Con este comando se presenta una tabla con todas las proyecciones de `expr`, tanto a lo largo del vector normal, como de la hipersuperficie.

- `$ConformalTime=True/False`

  Esta instrucción se utiliza para indicar a xPand si los resultados debe presentarlos en términos del tiempo conforme (`True`) o del cósmico (`False`). El valor por

[12] Con esta opción se trabaja de manera general, sin elegir ningún calibre en particular.

defecto es `True`, por lo que los resultados se presentan en términos del tiempo conforme a menos que indique lo contrario.

- `$FirstOrderTensorPerturbations=True/False`

  Se usa para especificar si los aportes tensoriales deben ser tenidos en cuenta (`True`) o no (`False`).

- `$FirstOrderVectorPerturbations=True/False`

  Igual que la instrucción anterior, pero para los modos vectoriales.

3.3.2. Perturbaciones escalares para FLRW plana

En la subsección anterior se presentaron los comandos básicos para trabajar con xPand. Ahora, como una primera aproximación al uso práctico de la herramienta, veremos aquí cómo obtener las perturbaciones de dos cantidades geométricas usando el paquete: la primera es la perturbación del tensor métrico en sí mismo, que al ser trivial permite familiarizarse con el flujo de trabajo de la herramienta. La otra cantidad es el escalar de Ricci, cuyo cálculo mediante el software permite hacer una comparativa rápida con el cálculo manual. Habiendo dicho esto, lo primero que debemos hacer es cargar el paquete. Esto lo podemos hacer con la instrucción:

```
<<xAct`xPand`;
```

Luego definimos la variedad:

```
DefManifold[M,4,{α,β,γ,μ,ν,λ,σ}];
```

Definimos también la métrica conforme de fondo:

```
DefMetric[-1,g[-α, -β], CD, {";", "∇"}, PrintAs -> "ḡ"];
```

Nótese que hasta aquí no se ha aclarado qué tipo de métrica es, sólo que su determinante es negativo, por lo que debemos especificarlo a continuación, en la aplicación de la separación $3+1$, donde no sólo se aclara el tipo de métrica trabajada, sino el vector normal $\boldsymbol{n}$, la métrica inducida $\boldsymbol{h}$ y la derivada covariante, `cd`, asociada a ella, junto con la forma en que esta va a ser representada. En este caso, como D. Ingresamos entonces:

```
SetSlicing[g, n, h, cd, {"|", "D"}, "FLFlat"];
```

Con la siguiente instrucción cambiamos la notación de las componentes escalares de la descomposición SVT, para que coincida con la que manejada en este trabajo:

```
PrintAs[φγ] ^= "A";
PrintAs[ψγ] ^= "C";
```

Luego de esto se define también la perturbación de la métrica de fondo. Esto lo hacemos con `DefMetricFields`, porque no se está usando xPert directamente, sino xPand:

```
DefMetricFields[g, dg, h];
```

Ya con esto se pueden perturbar las cantidades deseadas. Veamos por ejemplo las perturbaciones a primer orden del tensor métrico y del escalar de Ricci. Estas perturbaciones se calculan usando la función `ToxPand` y como esta es una función que recibe siete argumentos, cuatro de los cuales serán siempre iguales cuando sólo trabajamos con un modelo de espaciotiempo (FLFlat, BianchiI, etc.), una práctica común en el uso de xPand es definir una función MyToxPand que reciba sólo los datos que sí podrían cambiar de manera regular, que serían la expresión que se va a expandir (`expr`), el calibre en el que se va a trabajar (`gauge`) y el orden al que se va a realizar la perturbación (order). Esto lo hacemos con:

```
MyToxPand[expr_,gauge_,order_]:=ToxPand[expr,dg,u,du,h,gauge,order]
```

En la sección 3.3 vimos que `xPand` permite ignorar (False) o considerar (True) las perturbaciones vectoriales y las tensoriales. Como en este caso se está trabajando con la métrica FLRW y aquí las contribuciones escalares, vectoriales y tensoriales de las perturbaciones están desacopladas [3], podemos ignorar las vectoriales y quedarnos sólo con las tensoriales[13].

```
$FirstOrderVectorPerturbations = False;
$FirstOrderTensorPerturbations = True;
```

Por defecto el sistema trabaja en el tiempo conforme. Si deseamos trabajar con el tiempo cósmico, basta con hacer

```
$ConformalTime = False;
```

Ya con esto, podemos calcular las perturbaciones escalares y tensoriales del tensor métrico en un fondo FLWR sin curvatura en el calibre deseado. Eligiendo el calibre de Newton, nos queda entonces la instrucción:

```
MyToxPand[g[-μ, -ν], "NewtonGauge", 1]
```

Lo anterior genera el siguiente resultado:

$$g_{\mu\nu} = a^2\bar{h}_{\mu\nu} - a^2\bar{n}_\mu\bar{n}_\nu + \epsilon\left[2a^2E_{\mu\nu} - 2a^2A\bar{n}_\mu\bar{n}_\nu - 2a^2C\bar{h}_{\mu\nu}\right]. \tag{3.1}$$

Aquí vale la pena aclarar que los índices que se definen con la variedad no tienen por qué ser necesariamente letras griegas. Esto es irrelevante para el software y obedece sólo a una cuestión de convención del usuario, pues el programa los interpreta siempre como índices abstractos. Cuando el usuario vaya a interpretar los índices como componentes, sin embargo, debe tener en cuenta que xPand trabaja, en este caso, en una variedad 4-dimensional, por lo que sus índices van de 0 a 3. Siendo así, las cantidades ya **proyectadas** deben interpretarse como sus versiones extendidas, tal como se explicó en la sección 1.2.3.

Continuando con el desarrollo, puede que no siempre estemos interesados en trabajar con un resultado completo como el de la ecuación 3.1, así que con el comando `ExtractOrder` podemos extraer el orden deseado. En este caso puede ser el orden cero, que sería el valor de fondo o uno, que corresponde ya a la perturbación lineal. Si queremos la perturbación lineal, ejecutamos entonces el comando:

[13]Las escalares siempre están activas.

```
ExtractOrder[MyToxPand[g[-μ, -ν], "NewtonGauge", 1], 1]
```

Con esto obtenemos:

$$\delta g_{\mu\nu} = 2a^2 E_{\mu\nu} - 2a^2 A \bar{n}_\mu \bar{n}_\nu - 2a^2 C \bar{h}_{\mu\nu}.$$

Con la función `VisualizeTensor` se pueden además ver todas las proyecciones de un tensor, así que se puede aprovechar para ver las del tensor métrico. Esto lo conseguimos con la instrucción:

```
VisualizeTensor[ExtractOrder[MyToxPand[g[-μ,-ν],"NewtonGauge",1],1],h]
```

Con esto obtenemos la tabla siguiente:

	n	h
n	$-2a^2A$	0
h	0	$2a^2E_{\mu\nu} - 2a^2C\bar{h}_{\mu\nu}$

Podemos también extraer una proyección concreta utilizando para ello la instrucción `ExtractComponents`. Por ejemplo, para extraer la proyección espacial de la perturbación, ejecutamos las órdenes:

```
MyToxPand[g[-μ, -ν], "NewtonGauge", 1]
ExtractOrder[%, 1] [14]
ExtractComponents[ %, h, {"Space", "Space"}]
```

Nótese que aquí primero se da la instrucción de calcular las perturbaciones con `MyToxPand`, luego se extrae el primer orden y finalmente se extrae la proyección espacial. Esto da como resultado:

$$\delta g_{mn} = 2a^2 \left({}^{(1)}E_{mn}\right) - 2a^2 \bar{h}_{mn} C,$$

donde se han cambiado los índices $\mu \to m$ y $\nu \to n$ sólo por resaltar el carácter espacial de la proyección. Tampoco está de más insistir, con el fin de evitar confusiones, que $\bar{h}_{mn}$ es la métrica inducida por la métrica conforme de fondo, no por la métrica física de fondo, aunque sí está relacionada con ella mediante la transformación conforme de la ecuación 2.5, razón por la cual el resultado involucra los factores a^2.

Así mismo puede obtenerse el escalar de Ricci perturbado:

```
MyToxPand[RicciScalarCD[], "NewtonGauge", 1]
```

Con esto se tiene que:

$$R = 12H^2 + 6\dot{H} + \epsilon\left(\frac{4}{a^2} D_\alpha D^\alpha C - \frac{2}{a^2} D_\alpha D^\alpha A - 24H^2 A - 12\dot{H}A - 6H\dot{A} - 24H\dot{C} - 6\ddot{C}\right),$$

[14] Aquí el signo `%` lo usamos para indicarle al programa que la instrucción debe aplicarse al resultado inmediatamente anterior.

donde

$$\delta R = \frac{4}{a^2} D_\alpha D^\alpha C - \frac{2}{a^2} D_\alpha D^\alpha A - 24H^2 A - 12\dot{H}A - 6H\dot{A} - 24H\dot{C} - 6\ddot{C}. \tag{3.2}$$

Vemos entonces que a partir de unas pocas instrucciones se logra economizar el trabajo puramente manual del apéndice A, que incluso para un caso simple como este puede llegar a ser un poco extenso, lo que motiva el uso de herramientas de álgebra computacional que permitan automatizar los cálculos, sobre todo cuando la complejidad de los mismos aumenta el tiempo de solución sin aportar ningún beneficio real al análisis del problema.

4. Implementación

En esta sección se presentará la implementación en código del cálculo de las perturbaciones lineales en el espaciotiempo de Bianchi I para el campo escalar canónico. Para ello se muestra primero, por completez, cómo se pueden obtener las ecuaciones de movimiento y de campo Einstein para una métrica arbitraria. Luego, en la segunda parte, se obtiene la forma explícita de estas ecuaciones para un fondo Bianchi I y finalmente, en la tercera parte de esta sección, se presenta ya el cálculo de las perturbaciones lineales, presentando los resultados tanto en su forma diferencial como en el espacio de Fourier. Todo el código utilizado en esta sección se encuentra disponible en el apéndice C y en un cuaderno de Mathematica de acceso público en la referencia [7].

4.1. Ecuaciones generales para el campo escalar

Para este caso, la definición de la variedad y la métrica de fondo es exactamente igual que la trabajada en la sección 3.3.2 con fondo FLRW. La diferencia se da al momento de hacer la descomposición $3+1$, donde debemos cambiar `FLFlat` por `BianchiI` [15].

```
SetSlicing[g, n, γ, cd, {"|", "D"}, "BianchiI"];
```

Como se explicó en la sección 3.3, xPand usa la métrica conforme como métrica ambiente, lo que hace que la curvatura extrínseca de la hipersuperficie generada no tenga traza, coincidiendo así con el corte. Podemos entonces hacer que el programa imprima la curvatura directamente como el σ. Esto lo indicamos con:

```
PrintAs[Kγ] ^= "σ";
```

Ya con este detalle y con la descomposición del ambiente creada, se pueden calcular las cantidades de interés.

4.1.1. Ecuaciones de Campo de Einstein

Al definir la métrica y su perturbación, ya sea con xPand o con xPert, se están definiendo las cantidades del sector gravitacional. Para encontrar las ecuaciones de Eins-

[15] Cada vez que se realiza una descomposición $3+1$ con SetSlicing se definen automáticamente cantidades que son necesarias para el trabajo con xPand. Si se vuelve a ejecutar el comando en la misma sesión de Mathematica sin ninguna modificación, se podrían presentar conflictos, porque el programa intenta declarar nombres ya definidos. Para evitar esto, pueden seguirse tres caminos:

- Reiniciar el núcleo de Mathematica con el comando `Quit[]`, volver a cargar los paquetes y ejecutar nuevamente las definiciones teniendo en cuenta los cambios requeridos.
- Utilizar `SetSlicing` con nuevos nombres de vector normal, métrica inducida y derivada covariante, seguido de los comandos `DefMetricFields` y `DefMatterFields` para definir las perturbaciones asociadas a la nueva métrica inducida.
- Utilizar el comando `UnDef[]` de xTensor para eliminar definiciones previamente creadas. Esto permite utilizar los mismos nombres sin tener que reiniciar el núcleo.

tein es necesario también definir las cantidades correspondientes al sector de materia, por lo que en este caso se define el campo escalar como un tensor de rango cero.

```
DefTensor[Phi[], M, PrintAs -> "ϕ"];
```

Se define también la función escalar correspondiente al potencial.

```
DefScalarFunction[V];
```

Se definen las constantes. En este caso, una:

```
DefConstantSymbol[κ, PrintAs -> "κ"];
```

Se define la densidad lagrangiana de materia correspondiente al campo escalar canónico:

```
Lmat = -(1/2)Sqrt[-Detg[]] g[μ,ν]*CD[-μ][Phi[]]*CD[-ν][Phi[]]-V[Phi[]]
```

Ahora la correspondiente a la gravedad:

```
Lg = Sqrt[-Detg[]] 1/(2 κ) RicciScalarCD[]
```

La densidad lagrangiana total será entonces:

```
L = Lmat + Lg;
```

Ya con esto, generamos la perturbación de `L` con las funciones `Perturbation` y `ExpandPerturbation` de xPert:

```
Lpert = ToCanonical[
            ContractMetric[
                ExpandPerturbation[Perturbation[L,1]]]]
```

Finalmente se calcula la derivada variacional de la densidad lagrangiana utilizando `Lpert` y la perturbación de la métrica.

```
2 κ VarD[dg[LI[1],μ,ν],CD][Lpert] /.{delta[-LI[1],LI[1]]->1}
0 == %*(1/ Sqrt[-Detg[]])
% //Expand //RicciToEinstein //ContractMetric //ToCanonical
```

La instrucción `/.{delta[-LI[1],LI[1]]->1}` es una regla que usamos para que el programa reemplace por 1 los $\delta^1{}_1$ que surgen del cálculo.

Con esto se obtiene finalmente que

$$G_{\mu\nu} = -\frac{1}{2}\kappa g_{\mu\nu}\nabla_\alpha\phi\nabla^\alpha\phi + \kappa\nabla_\mu\phi\nabla_\nu\phi - \kappa g_{\mu\nu}V(\phi). \tag{4.1}$$

que corresponde a las ecuaciones de campo de Einstein para el caso del campo escalar canónico.

4.1.2. Ecuación de Movimiento

En este caso es necesario calcular la derivada variacional de la densidad lagrangiana, pero ahora respecto al campo escalar:

```
VarD[PhiPert[LI[1]], CD][Lpert]
% /.{delta[-LI[1], LI[1]] -> 1}
%*(1/Sqrt[-Detg[]]) //Simplify
0 == %
```

Con lo anterior se obtiene:

$$\nabla_\alpha \phi \nabla^\alpha \phi - V_\phi(\phi) = 0, \tag{4.2}$$

que es la forma general de la ecuación de Klein-Gordon para una métrica arbitraria.

4.2. Ecuaciones de fondo en Bianchi I

4.2.1. Ecuaciones de Friedmann

Por definición, las cantidades de fondo son siempre el orden cero en las perturbaciones, así que las ecuaciones de Friedmann para la métrica Bianchi I las podemos obtener expandiendo a orden cero las ecuaciones 4.1 y extrayendo las componentes a partir de allí.

La expansión a orden cero la hacemos entonces con:

```
EinsEqsBG = MyToxPand[EinsEqs, "NewtonGauge", 0]
```

En este caso `EinsEqs` es una variable que contiene el miembro no nulo de las ecuaciones de Einstein. El resultado de la operación se ha almacenado en la variable `EinsEqsBG` sólo por comodidad.

La componente tiempo-espacio se puede obtener con:

```
0 == ExtractComponents[EinsEqsBG, γ, {"Time", "Space"}]
```

Lo anterior devuelve lo siguiente:

$$\kappa D_\mu \phi \; \phi' = 0. \tag{4.3}$$

Para hacer un pequeño análisis sobre este resultado, vale la pena tener en cuenta que el tensor escalar ϕ se definió a partir de la función `DefTensor`, de xTensor, que lo vincula a la métrica ambiente. Y aunque para xPand esto significa que se trata de la métrica conforme de fondo, para xTensor es una métrica cualquiera, por lo que un tensor escalar definido allí podría, en principio, tener dependencia temporal y espacial. Sin embargo, aquí de forma natural, a partir la componente tiempo-espacio de las ecuaciones de Einstein, vemos que emerge la condición de que en un espaciotiempo homogéneo como el Bianchi I, un campo escalar dependiente del tiempo no puede tener dependencia espacial. Esto es un ejemplo de las diferencias que pueden encontrarse al definir un tensor con `DefTensor` de xTensor y hacerlo con `DefProjectedTensor`, de xPand. Los tensores escalares definidos con esta última función, ya asumen esta condición para sus valores de fondo, anulando automáticamente todas las derivadas covariantes espaciales que actúen sobre ellos.

Habiendo entendido las implicaciones del resultado anterior, es hora de extraer la componente tiempo-tiempo de estas ecuaciones con la orden

```
0 == ExtractComponents[EinsEqsBG, γ, {"Time", "Time"}]
```

Con esto se obtiene:

$$\mathcal{H}^2 = \frac{1}{6}\sigma^2 + \frac{1}{3}\kappa a^2 V(\phi) + \frac{1}{6}D_\alpha\phi D^\alpha\phi + \frac{1}{6}\kappa(\phi')^2,$$

pero si se tiene en cuenta la no dependencia espacial del campo escalar, entonces se llega a

$$\boxed{\mathcal{H}^2 = \frac{1}{6}\sigma^2 + \frac{1}{3}\kappa a^2 V(\phi) + \frac{1}{6}\kappa(\phi')^2}, \tag{4.4}$$

siendo esta entonces la primera ecuación de Friedmann en el espaciotiempo de Bianchi I.

Ahora extraemos la componente espacio-espacio:

```
0 == ExtractComponents[EinsEqsBG, γ, {"Space", "Space"}]
```

Con esto se obtiene:

$$\begin{aligned}\mathcal{H}'\gamma_{\mu\nu} =& \mathcal{H}\sigma_{\mu\nu} - \sigma_\mu{}^\alpha\sigma_{\nu\alpha} + \frac{1}{2}\sigma'_{\mu\nu} - \frac{1}{2}\mathcal{H}^2\gamma_{\mu\nu} - \frac{1}{4}\sigma_{\alpha\beta}\sigma^{\alpha\beta}\gamma_{\mu\nu} \\ &+ \frac{1}{2}\kappa a^2 V(\phi)\gamma_{\mu\nu} + \frac{1}{4}\kappa\gamma_{\mu\nu}D_\alpha\phi D^\alpha\phi - \frac{1}{2}\kappa D_\mu\phi D_\nu\phi + \frac{1}{4}\kappa\gamma_{\mu\nu}(\phi')^2.\end{aligned}$$

Si nuevamente se tiene en cuenta la no dependencia espacial del campo escalar (ecuación 4.3), esto se reduce a:

$$\begin{aligned}\mathcal{H}'\gamma_{\mu\nu} =& \mathcal{H}\sigma_{\mu\nu} - \sigma_\mu{}^\alpha\sigma_{\nu\alpha} + \frac{1}{2}\sigma'_{\mu\nu} - \frac{1}{2}\mathcal{H}^2\gamma_{\mu\nu} \\ &- \frac{1}{4}\sigma_{\alpha\beta}\sigma^{\alpha\beta}\gamma_{\mu\nu} + \frac{1}{2}\kappa a^2 V(\phi)\gamma_{\mu\nu} + \frac{1}{4}\kappa\gamma_{\mu\nu}(\phi')^2.\end{aligned}$$

Reemplazando $\mathcal{H}^2$ de la primera ecuación de Friedmann:

$$\mathcal{H}'\gamma_{\mu\nu} = \mathcal{H}\sigma_{\mu\nu} - \sigma_\mu{}^\alpha\sigma_{\nu\alpha} + \frac{1}{2}\sigma'_{\mu\nu} - \frac{1}{3}\sigma_{\alpha\beta}\sigma^{\alpha\beta}\gamma_{\mu\nu} + \frac{1}{3}\kappa a^2 V(\phi)\gamma_{\mu\nu} - \frac{1}{3}\kappa\gamma_{\mu\nu}(\phi')^2. \tag{4.5}$$

Calculando la traza se obtiene:

$$\boxed{\mathcal{H}' = -\frac{1}{3}\kappa(\phi')^2 + \frac{1}{3}\kappa a^2 V(\phi) - \frac{1}{3}\sigma^2}, \tag{4.6}$$

donde $\sigma^2 \equiv \sigma_{\alpha\beta}\sigma^{\alpha\beta}$. Esta corresponde entonces a la segunda ecuación de Friedmann para espaciotiempos Bianchi I.

Vemos entonces que ya se tiene la traza de la parte espacial de las ecuaciones de Einstein, así que sólo falta ver qué ocurre con la parte sin traza. Si deseamos utilizar el software, utilizamos la instrucción `STFPart` aplicada a la ecuación 4.5, especificando

la métrica respecto a la cual se extrae la traza. Como aquí estamos analizando la parte espacial, usamos $\gamma_{\mu\nu}$. El resultado de hacer esto es:

$$[\sigma'_{\mu\nu}]^{NT} = -2\mathcal{H}[\sigma_{\mu\nu}]^{NT} + 2[\sigma_\mu{}^\alpha \sigma_{\nu\alpha}]^{NT}.$$

Pero a partir de la ecuación 2.3 puede probarse que el último término es cero para el caso del campo escalar [1], por lo que se llega finalmente a que[16]:

$$\boxed{[\sigma'_{\mu\nu}]^{NT} = -2\mathcal{H}[\sigma_{\mu\nu}]^{NT}}, \tag{4.7}$$

expresión que podemos entender entonces como una tercera ecuación de Friedmann en espaciotiempos Bianchi I, ya que también surge del fondo.

4.2.2. Ecuación de Klein-Gordon

Para obtener el valor de fondo de la ecuación de Klein-Gordon también se debe expandir a orden cero. Si almacenamos el miembro no nulo de la ecuación 4.2, podemos calcular su forma explícita para esta métrica mediante la instrucción:

```
KGEqExp = MyToxPand[KGEq, "AnyGauge", 0]
0 == %
```

Con lo anterior obtenemos:

$$D_\alpha D^\alpha \phi - 2\mathcal{H}(\phi') - \phi'' - a^2 V_\phi(\phi) = 0$$

Pero teniendo en cuenta la homogeneidad del campo escalar (ecuación 4.3), se concluye que:

$$\varphi'' + 2\mathcal{H}\varphi' + a^2 V_\varphi(\varphi) = 0,$$

donde observamos que en el modelo Bianchi I, la ecuación de Klein-Gordon conserva la misma forma que en el FLRW (ecuación 2.8).

4.3. Perturbaciones lineales en Bianchi I

Llegados a este punto, es momento de encontrar las ecuaciones de las perturbaciones. Para esto, ya en la sección anterior se mostró que en un fondo homogéneo, el campo escalar también debe ser homogéneo, por lo que todas las derivadas espaciales actuando sobre él se anulan. Esto se puede tener en cuenta definiendo el campo con la instrucción:

[16] Hay que resaltar que lo que se anula es la parte sin traza de $\sigma_\mu{}^\alpha \sigma_{\nu\alpha}$, no todo el término. Tengamos en cuenta que este, al ser totalmente simétrico, se descompone como

$$\sigma_\mu{}^\alpha \sigma_{\nu\alpha} = [\sigma_\mu{}^\alpha \sigma_{\nu\alpha}]^{NT} + \frac{1}{3}\sigma_{\alpha\beta}\sigma^{\alpha\beta}\gamma_{\mu\nu},$$

por lo que si $[\sigma_\mu{}^\alpha \sigma_{\nu\alpha}]^{NT} = 0$, entonces $\sigma_\mu{}^\alpha \sigma_{\nu\alpha} = \frac{1}{3}\sigma^2\gamma_{\mu\nu}$. Esto es importante tenerlo en cuenta al implementarlo en el código, para evitar errores.

```
DefProjectedTensor[Phi[],γ]
```

Con esto se define automáticamente el valor de fondo y su perturbación. Como en este caso ya se ha trabajo con `Phi[]`, utilizaremos otro tensor, φ`[]`, que cumple con las mismas características, sólo que ha sido definido automáticamente por xPand al ejecutar el comando `DefMatterFields`.

Aquí se definen las ecuaciones de Einstein de manera similar a como se explicó en la sección 4.1.1, pero ahora para el tensor φ`[]`. El resultado se almacena en la variable `EinsEqs`.

Nuevamente nos aseguramos de que los nombres de los campos escalares asociados a la descomposición SVT de la métrica inducida coincidan con los nombres que estamos trabajando:

```
PrintAs[ϕγ] ^= "A";
PrintAs[ψγ] ^= "C";
```

Ahora es necesario definir algunas reglas que serán de utilidad en la manipulación de las expresiones. La primera es una que tenga en cuenta la tercera ecuación de Friedmann, lo que va a permitir reescribir primeras y segundas derivadas temporales del tensor de corte en términos del mismo sin derivadas.

```
KRules = Flatten@Join[
    MakeRule[
     Evaluate[{Kγ[LI[0], LI[2], -μ, -ν],
       LieD[n[α]][-2 Hγ[ LI[0], LI[0]] Kγ[LI[0], LI[0], -μ, -ν] +
       2 Kγ[ LI[0], LI[0], -μ, α] Kγ[ LI[0], LI[0], -ν, -α]]}]],

    MakeRule[
     Evaluate[{Kγ[LI[0], LI[1], -μ, -ν],
       -2 Hγ[ LI[0], LI[0]] Kγ[ LI[0], LI[0], -μ, -ν] +
        2 Kγ[ LI[0], LI[0], -μ, α] Kγ[LI[0], LI[0], -ν, -α]}]],

    MakeRule[Evaluate[{
      Kγ[ LI[0], LI[0], -α, λ] Kγ[ LI[0], LI[0], -β, -λ],
        1/3  Kγ[LI[0],LI[0],-λ,-μ] Kγ[LI[0],LI[0],λ,μ] γ[-α, -β]}]]
    ];
```

Nótese que esta regla está construida en secuencia, por lo que al aplicarla una vez, las segundas derivadas temporales se escriben en términos de primeras y las primeras en términos sin derivadas, esto hace que sea necesario aplicar la regla varias veces de forma consecutiva hasta eliminar completamente las derivadas temporales. Se podría hacer en una sola instrucción, pero este método ofrece una mayor legibilidad del código.

También relacionado con el corte, definimos también una regla sencilla para expresar $\sigma_{\alpha\beta}\sigma^{\alpha\beta}$ como σ^2:

```
KCuadrado =
  MakeRule[{Kγ[LI[0],LI[0],-α,-β] Kγ[ LI[0], LI[0],α,β],Kγ[]^2}];
```

Esto podría definirse como una regla automatizada, pero dejarla como regla «manual» da mayor control a la hora de aplicar derivadas sobre algunas expresiones que involucran el corte y a la hora de descomponer en el espacio de Fourier.

Es de utilidad definir también una regla que permita ir fácilmente al calibre de Newton en Bianchi I que se introdujo con la ecuación 2.25. Para esto hacemos entonces:

```
NewtonGaugeBianchiI =
  {Esγ[LI[1], LI[_]] -> 0,
  Bsγ[LI[1], LI[_]] -> 0,
  Bvγ[LI[1], LI[_], _] -> 0};
```

Como en este caso se trabajará en términos de las variables invariantes de calibre introducidas en la sección 2.4, es necesario definir tales cantidades. Para ello debemos recordar que estas se definen a partir de elementos de la descomposición SVT de la perturbación de la métrica inducida, por lo que deben estar definidas en la variedad perturbada, no en la de fondo. Ejecutamos entonces las instrucciones:

```
DefProjectedTensor[ΦsB[], γ,
 PrintAs -> "Φ", SpaceTimesOfDefinition -> {"Perturbed"}]
DefProjectedTensor[ΦvB[-μ], γ,
 PrintAs -> "Φ", SpaceTimesOfDefinition -> {"Perturbed"}]
DefProjectedTensor[ψB[], γ, PrintAs -> "ψ",
 SpaceTimesOfDefinition -> {"Perturbed"}]
DefTensor[χ[], M, PrintAs -> "χ"]
```

Se puede verificar, a partir de la ecuación 2.24, que $D^i\phi_i = 0$, por lo que es útil introducir una automatización que aplique esta regla. Tenemos entonces:

```
AutomaticRules[ΦvB, MakeRule[Evaluate[{cd[μ]@ΦvB[-μ], 0}]]];
```

Conviene ahora definir una regla que permita aprovechar las ecuaciones 2.22-2.26 para reescribir las expresiones en términos de las invariantes de gauge partiendo del calibre de Newton:

```
BardeenRules = Flatten@Join[

    MakeRule[
     Evaluate[{φγ[LI[1], LI[x_]], ΦsB[LI[0], LI[x]]}]],

    MakeRule[
     Evaluate[{ψγ[LI[1], LI[x_]], ψB[LI[0], LI[x]]}]] ,

    MakeRule[
     Evaluate[{φ[LI[1], LI[x_]], χ[LI[0], LI[x]]}]],

    MakeRule[
     Evaluate[{Evγ[LI[1], LI[2], -μ],
       LieD[n[β]][-ΦvB[LI[0], LI[0], -μ] +
         2 Kγ[-μ, -α] Evγ[LI[1], LI[0], α]]}]],

    MakeRule[
     Evaluate[{Evγ[LI[1], LI[1], -μ],
        -ΦvB[LI[0], LI[0], -μ] +
```

```
        2 Kγ[-μ, -α] Evγ[LI[1], LI[0], α]}]]
    ];
```

Por otra parte, aunque xPand posee algoritmos para forzar la formación de divergencias y laplacianos cuando haya lugar a ellos [5], en el desarrollo de esto proyecto no se han obtenido resultados satisfactorios con esto, así que es útil definir una regla que haga este trabajo:

```
ForceDivRule = Flatten@Join[
    (*Para tensores de rango 1*)
    {cd[x1_]@
        cd[x2_]@cd[x3_]@cd[x4_]@expr_[a1_, a2_, x5_] /; (x5 == -x1) ->
       Evaluate[cd[x2]@cd[x3]@cd[x4]@cd[x1]@expr[a1, a2, x5]]},
    {cd[x1_]@
        cd[x2_]@cd[x3_]@cd[x4_]@expr_[a1_, a2_, x5_] /; (x5 == -x2) ->
       Evaluate[cd[x1]@cd[x3]@cd[x4]@cd[x2]@expr[a1, a2, x5]]},
    {cd[x1_]@
        cd[x2_]@cd[x3_]@cd[x4_]@expr_[a1_, a2_, x5_] /; (x5 == -x3) ->
       Evaluate[cd[x1]@cd[x2]@cd[x4]@cd[x3]@expr[a1, a2, x5]]},

    {cd[x1_]@cd[x2_]@cd[x3_]@expr_[a1_, a2_, x5_] /; (x5 == -x1) ->
      Evaluate[cd[x2]@cd[x3]@cd[x1]@expr[a1, a2, x5]]},
    {cd[x1_]@cd[x2_]@cd[x3_]@expr_[a1_, a2_, x5_] /; (x5 == -x2) ->
      Evaluate[cd[x1]@cd[x3]@cd[x2]@expr[a1, a2, x5]]},

    {cd[x1_]@cd[x2_]@expr_[a1_, a2_, x5_] /; (x5 == -x1) ->
      Evaluate[cd[x2]@cd[x1]@expr[a1, a2, x5]]},

    (*Para tensores de rango 2*)
    {cd[x1_]@
        cd[x2_]@cd[x3_]@
          cd[x4_]@expr_[a1_, a2_, x5_,
            x6_] /; (x5 == -x1) || (x6 == -x1) ->
      Evaluate[cd[x2]@cd[x3]@cd[x4]@cd[x1]@expr[a1, a2, x5, x6]]},
    {cd[x1_]@
        cd[x2_]@cd[x3_]@
          cd[x4_]@expr_[a1_, a2_, x5_,
            x6_] /; (x5 == -x2) || (x6 == -x2) ->
      Evaluate[cd[x1]@cd[x3]@cd[x4]@cd[x2]@expr[a1, a2, x5, x6]]},
    {cd[x1_]@
        cd[x2_]@cd[x3_]@
          cd[x4_]@expr_[a1_, a2_, x5_,
            x6_] /; (x5 == -x3) || (x6 == -x3) ->
      Evaluate[cd[x1]@cd[x2]@cd[x4]@cd[x3]@expr[a1, a2, x5, x6]]},

    {cd[x1_]@
        cd[x2_]@cd[x3_]@
          expr_[a1_, a2_, x5_, x6_] /; (x5 == -x1) || (x6 == -x1) ->
      Evaluate[cd[x2]@cd[x3]@cd[x1]@expr[a1, a2, x5, x6]]},
    {cd[x1_]@
        cd[x2_]@cd[x3_]@
          expr_[a1_, a2_, x5_, x6_] /; (x5 == -x2) || (x6 == -x2) ->
```

```
      Evaluate[cd[x1]@cd[x3]@cd[x2]@expr[a1, a2, x5, x6]]},

    {cd[x1_]@
        cd[x2_]@expr_[a1_, a2_, x5_,
          x6_] /; (x5 == -x1) || (x6 == -x1) ->
      Evaluate[cd[x2]@cd[x1]@expr[a1, a2, x5, x6]]}
    ];
```

En caso de desear una generalización de esta regla a n derivadas covariantes, podría definirse de forma más óptima a partir de recursividad. Para este proyecto, sin embargo, es suficiente con la opción planteada.

Para definir otra regla que será de utilidad, tengamos en cuenta que si a dos veces la ecuación 4.4 se le suma la ecuación 4.6 y si al doble de la ecuación 4.4 se le resta el doble de la ecuación 4.6 se obtienen las siguientes identidades:

$$\kappa a^2 V = 2\mathcal{H}^2 + \mathcal{H}',$$
$$\kappa(\varphi')^2 = 2\mathcal{H}^2 - 2\mathcal{H} - \sigma^2.$$

Estas relaciones son sencillas, pero útiles para simplificar un poco las ecuaciones, así que se definirá una regla que las aplique:

```
BGRules = Flatten@Join[
    {κ aγ[ LI[0], LI[0]]^2 V[ φ[ LI[0], LI[0]]] ->
    2 Hγ[LI[0], LI[0]]^2 + Hγ[LI[0], LI[1]]},

    {κ aγ[ LI[0], LI[0]]^2 V[ Scalar[ φ[ LI[0], LI[0]]]] ->
    2 Hγ[LI[0], LI[0]]^2 + Hγ[LI[0], LI[1]]},

    {κ Scalar[ aγ[ LI[0], LI[0]]]^2 V[ Scalar[ φ[ LI[0], LI[0]]]] ->
    2 Hγ[LI[0], LI[0]]^2 + Hγ[LI[0], LI[1]]},

    {κ Scalar[ aγ[ LI[0], LI[0]]]^2 V[ φ[ LI[0], LI[0]]] ->
    2 Hγ[LI[0], LI[0]]^2 + Hγ[LI[0], LI[1]]},


    MakeRule[Evaluate[{κ*φ[LI[0], LI[1]]^2 ->
    2 Hγ[LI[0], LI[0]]^2 - 2 Hγ[LI[0], LI[1]] - Kγ[-μ, -ν] Kγ[μ, ν]}]],

    MakeRule[Evaluate[{κ Scalar[ φ[ LI[0], LI[1]]]^2 ->
    2 Hγ[LI[0], LI[0]]^2 - 2 Hγ[LI[0], LI[1]] - Kγ[-μ, -ν] Kγ[μ, ν]}]]
    ];
```

Si además se desean expresar los resultados en términos del espacio de Fourier, es necesario definir los tensores involucrados, como se explicó en la sección 2.5. Definimos entonces los escalares E_1, E_2, E_+, $E_\times$, Φ_1 y Φ_2 de las ecuaciones 2.31 a 2.33.

```
DefProjectedTensor[Ep[], γ, PrintAs -> "E+",
 SpaceTimesOfDefinition -> {"Perturbed"}]
DefProjectedTensor[Ec[], γ, PrintAs -> "E×",
 SpaceTimesOfDefinition -> {"Perturbed"}]
```

```
DefProjectedTensor[E1[], γ, PrintAs -> "E_1",
 SpaceTimesOfDefinition -> {"Perturbed"}]
DefProjectedTensor[E2[], γ, PrintAs -> "E_2",
 SpaceTimesOfDefinition -> {"Perturbed"}]

DefProjectedTensor[Φ1[], γ, PrintAs -> "Φ_1",
 SpaceTimesOfDefinition -> {"Perturbed"}]
DefProjectedTensor[Φ2[], γ, PrintAs -> "Φ_2",
 SpaceTimesOfDefinition -> {"Perturbed"}]
```

Definimos ahora en la variedad de fondo los vectores de la base ortonormal $\{\hat{k}_i, e_i^1, e_i^2\}$ adaptada a $k_i = k\hat{k}_i$.

```
DefProjectedTensor[Nk[], γ, PrintAs -> "|k|",
 SpaceTimesOfDefinition -> {"Background"}]
DefProjectedTensor[k[-α], γ, PrintAs -> "k̂",
 SpaceTimesOfDefinition -> {"Background"}]
DefProjectedTensor[e1[-α], γ, PrintAs -> "e_1",
 SpaceTimesOfDefinition -> {"Background"}]
DefProjectedTensor[e2[-α], γ, PrintAs -> "e_2",
 SpaceTimesOfDefinition -> {"Background"}]
```

Definimos también los tensores de polarización ε_{ij}^{+} y $\varepsilon_{ij}^{\times}$ dados por la ecuación 2.29.

```
DefProjectedTensor[εp[-α, -β], γ, PrintAs -> "ε_+",
  SpaceTimesOfDefinition -> {"Background"}]
DefProjectedTensor[εc[-α, -β], γ, PrintAs -> "ε_×",
  SpaceTimesOfDefinition -> {"Background"}]
```

Ahora es necesario definir los escalares $\sigma_{\parallel}$, σ_{V1}, σ_{V2}, σ_{T+} y $\sigma_{T\times}$ asociados a la descomposición dada por la ecuación 2.34 sobre el tensor de corte.

```
DefProjectedTensor[σpar[], γ, PrintAs -> "σ_||",
 SpaceTimesOfDefinition -> {"Background"}]
DefProjectedTensor[σV1[], γ, PrintAs -> "σ_V1",
 SpaceTimesOfDefinition -> {"Background"}]
DefProjectedTensor[σV2[], γ, PrintAs -> "σ_V2",
 SpaceTimesOfDefinition -> {"Background"}]
DefProjectedTensor[σp[], γ, PrintAs -> "σ_+",
 SpaceTimesOfDefinition -> {"Background"}]
DefProjectedTensor[σc[], γ, PrintAs -> "σ_×",
 SpaceTimesOfDefinition -> {"Background"}]
```

Definimos algunas reglas automatizadas para los vectores de la base, $\{\hat{k}^i, e^1, e^2\}$, para que el paquete tenga en cuenta su carácter ortonormal:

```
AutomaticRules[e1, MakeRule[Evaluate[{e1[-α] e2[α], 0}]]];
AutomaticRules[e1, MakeRule[Evaluate[{e1[-α] k[α], 0}]]];
AutomaticRules[e2, MakeRule[Evaluate[{e2[-α] k[α], 0}]]];

AutomaticRules[e1,
  MakeRule[
```

```
   Evaluate[{e1[-α] e2[-β] g[α, β], 0}]];
AutomaticRules[e1,
  MakeRule[
   Evaluate[{e1[-α] k[-β] g[α, β], 0}]];
AutomaticRules[e2,
  MakeRule[
   Evaluate[{e2[-α] k[-β] g[α, β], 0}]];

AutomaticRules[e1,
  MakeRule[Evaluate[{e1[-α] e1[α], 1}]];
AutomaticRules[e2,
  MakeRule[Evaluate[{e2[-α] e2[α], 1}]];
AutomaticRules[k, MakeRule[Evaluate[{k[-α] k[α], 1}]];
AutomaticRules[k, MakeRule[Evaluate[{k[LI[0], LI[1], -α], 0}]];
```

Definimos también algunas reglas automáticas que se pueden deducir de la ecuación 2.29 para los tensores de polarización:

```
AutomaticRules[εp,
  MakeRule[
   Evaluate[{εp[-μ, -ν] g[μ, ν], 0}]];
AutomaticRules[εp,
  MakeRule[
   Evaluate[{k[-α] εp[α, β], 0}]];
AutomaticRules[εp,
  MakeRule[
   Evaluate[{εp[α, β] εp[- α, -β], 1}]];
AutomaticRules[εp,
  MakeRule[
   Evaluate[{εp[α, β] εc[- α, -β], 0}]];
AutomaticRules[εc,
  MakeRule[
   Evaluate[{εc[-μ, -ν] g[μ, ν], 0}]];
AutomaticRules[εc,
  MakeRule[
   Evaluate[{εc[α, β] εc[-α,-β], 1}]];
AutomaticRules[εc,
  MakeRule[
   Evaluate[{k[-α] εc[α, β], 0}]];

AutomaticRules[εp,
  MakeRule[
   Evaluate[{εp[-α, -β] εp[ β, σ] εp[-σ, α], 0}]];
AutomaticRules[εp,
  MakeRule[
   Evaluate[{εp[-α, -β] εp[ β, σ] εc[-σ, α], 0}]];
AutomaticRules[εp,
  MakeRule[
   Evaluate[{εp[-α, -β] εc[ β, σ] εc[-σ, α], 0}]];
AutomaticRules[εc,
  MakeRule[
```

```
    Evaluate[{εc[-α, -β] εc[ β, σ] εc[-σ, α], 0}]]];
```

Definimos otras reglas que, haciendo uso de la ecuación 2.29, permitan expresar los tensores de polarización en términos de los vectores base. Puede que no siempre se desee hacer esto, por lo que no se definirá como una regla automática, sino manual.

```
εRules = Flatten@Join[
    MakeRule[
     Evaluate[{e1[α] εp[-α, -β], 1/Sqrt[2] e1[-β]}]],
    MakeRule[
     Evaluate[{e1[λ] εp[-α, -β] g[ α, -λ], 1/Sqrt[2] e1[-β]}]],
    MakeRule[
     Evaluate[{e2[α] εp[-α, -β], -( 1/Sqrt[2]) e2[-β]}]],
    MakeRule[
     Evaluate[{e2[λ] εp[-α, -β] g[ α, -λ], -(1/Sqrt[2]) e2[-β]}]],

    MakeRule[
     Evaluate[{e1[α] εc[-α, -β], 1/Sqrt[2] e2[-β]}]],
    MakeRule[
     Evaluate[{e1[λ] εc[-α, -β] g[ α, -λ], 1/Sqrt[2] e2[-β]}]],
    MakeRule[
     Evaluate[{e2[α] εc[-α, -β], 1/Sqrt[2] e1[-β]}]],
    MakeRule[
     Evaluate[{e2[λ] εc[-α, -β] g[ α, -λ], 1/Sqrt[2] e1[-β]}]]

    MakeRule[
      Evaluate[{εc[-μ,-α]εc[α,-ν],
        1/2 e1[LI[0],LI[0],-μ] e1[LI[0],LI[0],-ν] +
        1/2  e2[ LI[0], LI[0], -μ] e2[ LI[0], LI[0], -ν]}]],
    MakeRule[
      Evaluate[{εc[-μ, -α] εp[α, -ν],
        1/2  e1[ LI[0], LI[0], -ν]   e2[ LI[0], LI[0], -μ] -
        1/2  e1[ LI[0], LI[0], -μ]   e2[ LI[0], LI[0], -ν]}]],
    MakeRule[
      Evaluate[{εp[-μ, -α] εp[α, -ν],
        1/2  e1[ LI[0], LI[0], -μ]   e1[ LI[0], LI[0], -ν] +
        1/2  e2[ LI[0], LI[0], -μ]   e2[ LI[0], LI[0], -ν]}]]

    ];
```

Es útil definir también algunas reglas que descompongan el E_μ y $E_{\mu\nu}$ y sus derivadas temporales en el espacio de Fourier haciendo uso de las ecuaciones 2.31 y 2.32; después de todo, hasta ahora se han definido los escalares asociados a la descomposición, pero no se ha definido ninguna regla que reescriba estos tensores en términos de tales cantidades, por lo que es necesario introducirla. Esto lo hacemos con:

```
ERules = Flatten@Join[
    MakeRule[
     Evaluate[{Evγ[LI[1], LI[0], -α],
       ToCanonical@
```

```
        ContractMetric[
         E1[LI[0], LI[0]] e1[-α] +
          E2[LI[0], LI[0]] e2[-α]]}]],

    MakeRule[
     Evaluate[{Etγ[LI[1], LI[0], -α, -β],
       ToCanonical@
        ContractMetric[
         Ep[LI[0], LI[0]] εp[-α, -β] +
          Ec[LI[0], LI[0]] εc[-α, -β]]}]],
    MakeRule[
     Evaluate[{Etγ[LI[1], LI[1], -α, -β],
       ToCanonical@
        ContractMetric[
         LieD[n[μ]][
          Ep[LI[0], LI[0]] εp[-α, -β] +
           Ec[LI[0], LI[0]] εc[-α, -β]]]}]],
    MakeRule[
     Evaluate[{Etγ[LI[1], LI[2], -α, -β],
       ToCanonical@
        ContractMetric[
         LieD[n[μ]]@
          LieD[n[μ]][
           Ep[LI[0], LI[0]] εp[-α, -β] +
            Ec[LI[0], LI[0]] εc[-α, -β]]]}]]
    ];
```

Definimos ahora una regla para hacer lo mismo, pero para la descomposición del tensor de corte (ecuación 2.34):

```
RuleDecomposeKγ =
 MakeRule[{Kγ[LI[0], LI[0], -α, -β],
   Evaluate[
    3/2 (k[-α] k[-β] - 1/3 γ[-α, -β]) σpar[] + σV1[] ( k[-α] e1[-β] +
        k[-β] e1[-α]) + σV2[] ( k[-α] e2[-β] +
        k[-β] e2[-α]) + σp[] εp[- α, -β] + σc[] εc[-α, -β]]}]
```

Finalmente, en la variable `FourierModes` agrupamos varias de las reglas anteriores junto con otras adicionales para la descomposición del vector Φ_μ y para describir cómo deben tratarse las derivadas temporales de este, de los vectores de la base adaptada y de los tensores de polarización.

```
FourierModes = Flatten@Join[ERules, RuleDecomposeK,
    MakeRule[
     Evaluate[{ΦvB[LI[0], LI[0], -α],
       ToCanonical@
        ContractMetric[Φ1[LI[0],
            LI[0]] e1[-α] + Φ2[LI[0],
            LI[0]] e2[-α]]}]],
    MakeRule[
     Evaluate[{ΦvB[LI[0], LI[1], -α],
       ToCanonical@
        ContractMetric[
```

```
        LieD[n[β]]@(Φ1[LI[0],
             LI[0]] e1[-α] + Φ2[LI[0],
             LI[0]] e2[-α])]}]],

   MakeRule[
    Evaluate[{εp[LI[0],
       LI[1], -μ, -ν], -Kγ[α, β] εp[-α, -β] (γ[-μ, -ν] -
          k[-μ] k[-ν]) - Kγ[α, β] (γ[-α, -β] -
          k[-α] k[-β]) εp[-μ, -ν] + 2 (Kγ[α, -μ] εp[-ν, -α] +
           Kγ[α, -ν] εp[-μ, - α])}]],
   MakeRule[
    Evaluate[{εc[LI[0],
       LI[1], -μ, -ν], -Kγ[α, β] εc[-α, -β] (γ[-μ, -ν] -
          k[-μ] k[-ν]) - Kγ[α, β] (γ[-α, -β] -
          k[-α] k[-β]) εc[-μ, -ν] + 2 (Kγ[α, -μ] εc[-ν, -α] +
           Kγ[α, -ν] εc[-μ, - α])}]],

   MakeRule[
    Evaluate[{εp[LI[0], LI[2], -μ, -ν],
      LieD[n[β]]@(-Kγ[α, β] εp[-α, -β] (γ[-μ, -ν] -
            k[-μ] k[-ν]) - Kγ[α, β] (γ[-α, -β] -
            k[-α] k[-β]) εp[-μ, -ν] + 2 (Kγ[α, -μ] εp[-ν, - α] +
            Kγ[α, -ν] εp[-μ, - α]))}]],

   MakeRule[
    Evaluate[{εc[LI[0], LI[2], -μ, -ν],
      LieD[n[β]]@(-Kγ[α, β] εc[-α, -β] (γ[-μ, -ν] -
            k[-μ] k[-ν]) - Kγ[α, β] (γ[-α, -β] -
            k[-α] k[-β]) εc[-μ, -ν] + 2 (Kγ[α, -μ] εc[-ν, - α] +
            Kγ[α, -ν] εc[-μ, - α]))}]],

   MakeRule[{e1[LI[0], LI[1], -μ],
     Evaluate[-Kγ[-α, -β] e1[α] e1[β] e1[-μ] -
       Kγ[-α, -β] e1[α] e2[β] e2[- μ] + 2 Kγ[-μ, -α] e1[α]]}],
   MakeRule[{e2[LI[0], LI[1], -μ],
     Evaluate[-Kγ[-α, -β] e2[α] e1[β] e1[-μ] -
       Kγ[-α, -β] e2[α] e2[β] e2[- μ] + 2 Kγ[-μ, -α] e2[α]]}]
   ];
```

Con esto ya se tienen algunas reglas que, como veremos en la siguiente sección y como se detalla en el apéndice C, permiten manipular las expresiones con mayor comodidad.

4.3.1. Ecuación de Klein-Gordon para las perturbaciones

Para obtener las perturbaciones de la ecuación de Klein-Gordon, definimos primero la parte no nula de la ecuación general con la siguiente instrucción:

```
KGEq = CD[-μ]@CD[μ][φ[]] - V'[φ[]]
```

Ahora se expande:

```
KGEqExp = MyToxPand[KGEq, "AnyGauge", 1]
```

Luego extraemos el primer orden de `KGEqExp`:

```
KGpert = ExtractOrder[KGEqExp, 1] // Expand
```

Igualando a cero se obtiene entonces la ecuación para un calibre arbitrario:

$$\begin{aligned}0 =& -2\mathcal{H}\delta\varphi' - \delta\varphi'' + 4\mathcal{H}\varphi' A + 2\varphi'' A + \varphi' A' + 3\varphi' C' \\ &+ 2\sigma^{\mu\alpha}\varphi' D_\alpha D_\mu E + \varphi' D_\mu D^\mu B - \varphi' D_\mu D^\mu E' + D_\mu D^\mu \delta\varphi \\ &- a^2 \delta\varphi V_{\varphi\varphi}(\varphi)\end{aligned}$$

Para expresarla en términos de χ y de los potenciales de Bardeen, nos movemos primero al calibre de Newton para Bianchi I:

```
KGpert /. NewtonGaugeBianchiI
```

Desde este calibre ya podemos reescribir la ecuación usando la regla `BardeenRules`:

```
%/.BardeenRules
```

Igualando a cero y realizando algunas manipulaciones algebraicas menores (ver apéndice C.3.2), se obtiene:

$$2\mathcal{H}\chi' + \chi'' - D_\mu D^\mu \chi + a^2 \chi V_{\varphi\varphi}(\varphi) = 4\mathcal{H}\varphi'\Phi + 2\varphi''\Phi + \varphi'\Phi' + 3\varphi'\Psi',$$

que corresponde entonces a la ecuación de movimiento para la perturbación del campo escalar en un espaciotiempo de fondo Bianchi tipo I.

4.3.2. Ecuaciones de Einstein para las perturbaciones

En este caso expandimos nuevamente sin elegir un calibre en particular.

```
EinsEqsExp = MyToxPand[EinsEqs, "AnyGauge", 1]//Expand [17]
```

Como una vez más el interés está en el comportamiento de las perturbaciones, extraemos el primer orden del resultado anterior:

```
EinsPert = ExtractOrder[EinsEqsExp, 1]//Expand
```

Aquí aparecen primeras y segundas derivadas del tensor de corte respecto al tiempo conforme, así que podemos aprovechar la tercera ecuación de Friedmann (ecuación 4.7) para reescribirlas. Esto lo hacemos con la regla `KRules`:

```
EinsPert /. KRules // Expand
```

Luego de aplicar tres veces la instrucción anterior, se obtienen las perturbaciones de las ecuaciones de Einstein, que era el principal objetivo, aunque hasta ahora estarían en bruto, sin ningún tipo de manipulación, por lo que resta presentar de manera explícita cada una de las componentes. Esto se hará nuevamente basándose en el calibre de Newton en Bianchi I:

[17] Como ya en este punto las expresiones que genera xPand son demasiado extensas, sólo se presentan los resultados finales por cuestiones de practicidad.

```
EinsPertNewton = EinsPert /.  NewtonGaugeBianchiI
```

Con esto podemos encontrar ya la forma explícita de las componentes, para lo cual hay que tener en cuenta que las ecuaciones de Einstein de las perturbaciones tienen la siguiente forma general:

$$\delta G_{\mu\nu} = \kappa \delta T_{\mu\nu},$$

que por la descomposición $3+1$ tiene componentes tiempo-tiempo, tiempo-espacio y espacio-espacio. En términos generales estas componentes dan origen a un conjunto de ecuaciones que describen el sistema: cuatro escalares, dos vectoriales y una tensorial.

De la componente tiempo-tiempo tenemos:

$$\delta G_{00} = \kappa \delta T_{00} \qquad \text{1ra ecuación escalar}$$

De la componente tiempo-espacio tenemos:

$$\begin{aligned} &\delta G_{0n} = \kappa \delta T_{0n} && \text{1ra ecuación vectorial} \\ &D^n \delta G_{0n} = D^n \kappa \delta T_{0n} && \text{2da ecuación escalar} \end{aligned}$$

De la componente espacio-espacio tenemos:

$$\begin{aligned} &\gamma^{mn} \delta G_{mn} = \gamma^{mn} \kappa \delta T_{mn} && \text{3ra ecuación escalar} \\ &[\delta G_{mn}]^{NT} = \kappa [\delta T_{mn}]^{NT} && \text{Ecuación tensorial} \\ &D^n [\delta G_{mn}]^{NT} = \kappa D^n [\delta T_{mn}]^{NT} && \text{2da ecuación vectorial} \\ &D^m D^n [\delta G_{mn}]^{NT} = \kappa D^m D^n [\delta T_{mn}]^{NT} && \text{4ta ecuación escalar} \end{aligned}$$

Estas ecuaciones pueden ser llevadas al espacio de Fourier, en donde además pueden ser expresadas en términos de los modos transversales descritos en la sección 2.5, así que pasemos ahora a ver la forma explícita de cada una de ellas usando el software.

Primera ecuación escalar

Como hemos visto, la primera ecuación escalar es la componente tiempo tiempo de las ecuaciones, así que teniendo ya almacenadas en `EinsPertNewton` las ecuaciones de Einstein para las perturbaciones en el calibre de Newton, podemos extraer la componente tiempo-tiempo con la siguiente instrucción:

```
EinsTimeTime=ExtractComponents[EinsPert, γ, {“Time”, “Time”}]
```

Igualando este a cero se obtiene la forma explícita de la ecuación $\delta G_{00} = \kappa \delta T_{00}$.

$$\begin{aligned} 0 =& E_{\alpha\beta}{}' \sigma^{\alpha\beta} + \kappa \varphi' \delta\varphi' + 2\kappa a^2 V(\phi) A + 2\sigma^2 C + \frac{\mathcal{H}'}{\mathcal{H}^2} \sigma^2 C \\ &+ 6\mathcal{H} C' - \frac{1}{\mathcal{H}} \sigma^2 C' - 2 D_\alpha D^\alpha C + \sigma^{\beta\alpha} D_\alpha E_\beta{}' \\ &+ \frac{1}{\mathcal{H}} \sigma^{\alpha\beta} D_\beta D_\alpha C + \kappa a^2 \delta\varphi^2 V_\varphi(\varphi). \end{aligned}$$

Esta ecuación puede ser expresada en términos de los potenciales de Bardeen haciendo:

```
EinsTimeTime/.BardeenRules//Expand
```

Esto hará que aparezcan términos de la forma $\sigma_{\alpha\lambda}\sigma^{\lambda}{}_{\beta}$, por lo que se hace necesario aplicar `KRules`, para que elimine su parte sin traza. Luego de esto, con las manipulaciones algebraicas detalladas en C.3.4, se llega finalmente a que

$$\begin{aligned} &2\mathcal{H}^2\Phi + \mathcal{H}'\Phi + \frac{1}{2}\kappa\varphi'\chi' + 3\mathcal{H}\Psi' - D_\alpha D^\alpha\Psi + \frac{1}{2}\kappa a^2\chi V_\varphi(\varphi) = \\ &-\frac{1}{2}E_{\alpha\beta}{}'\sigma^{\alpha\beta} - \sigma^2\Psi - \frac{\mathcal{H}'\sigma^2}{2\mathcal{H}^2}\Psi + \frac{\sigma^2}{2\mathcal{H}}\Psi' + \frac{1}{2}\sigma^{\beta\alpha}D_\alpha(\Phi_\beta) - \frac{1}{2\mathcal{H}}\sigma^{\alpha\beta}D_\beta D_\alpha\Psi. \end{aligned} \tag{4.8}$$

Podemos llevar lo anterior al espacio de Fourier. Para ello, primero se transforman las derivadas covariantes con:

```
% //. cd[α_][expr_] :> I Nk[] k[α] expr
```

Luego se aplican las reglas de transformación a los tensores involucrados. Esto lo hacemos con:

```
% /. FourierModes // Expand // ContractMetric
```

Puede ser necesario aplicar la regla varias veces hasta que la expresión no cambie. Aplicando ε`Rules` para simplificar un poco la expresión, se llega finalmente a que la primera ecuación asociada a los modos escalares viene dado por:

$$\begin{aligned} &2\mathcal{H}^2\Phi + \mathcal{H}'\Phi + \frac{1}{2}\kappa\varphi'\chi' + 3\mathcal{H}\Psi' + k^2\Psi + \frac{1}{2}\kappa a^2\chi V_\varphi(\varphi) = \\ &+\sum_{\lambda=+,\times}\left(E_\lambda\sigma_\lambda\sigma_{\parallel} - \frac{1}{2}E'_\lambda\sigma_\lambda\right) - \sqrt{2}E_+\sigma_{V1}^2 - 2\sqrt{2}E_\times\sigma_{V1}\sigma_{V2} + \sqrt{2}E_+\sigma_{V2}^2 \\ &+\frac{1}{2}\sum_{a=1}^{2}ik\sigma_{Va}\Phi_{Va} + \sigma^2\left[\left(\frac{\Psi}{2\mathcal{H}}\right)' - \Psi\right] + \frac{1}{2\mathcal{H}}k^2\sigma_{\parallel}\Psi. \end{aligned} \tag{4.9}$$

Con esto hemos obtenido ya la primera ecuación escalar tanto en su forma diferencial como en el espacio de Fourier.

Primera ecuación vectorial

La primera ecuación vectorial viene dada por $\delta G_{0n} = \kappa\delta T_{0n}$, por lo debemos extraer la componente tiempo-espacio. Esto se hace con:

```
EinsTimeSpace = ExtractComponents[EinsPertNewton,γ,{"Time","Space"}]
```

Expresando en términos de los potenciales de Bardeen e igualando a cero, como se detalla en el apéndice C.3.5, se obtiene entonces que la forma explícita de la primera ecuación vectorial es:

$$\begin{aligned} 2\mathcal{H}D_\nu\Phi - \kappa\varphi'D_\nu\chi - \frac{1}{2}D_\alpha D^\alpha\Phi_\nu + 2D_\nu\Psi' = \sigma_{\nu\alpha}D^\alpha\left[\Phi + \Psi + \left(\frac{\Psi}{\mathcal{H}}\right)'\right] \\ - 2\sigma^{\alpha\beta}D_\beta E_{\nu\alpha} + \sigma^{\alpha\beta}D_\nu E_{\alpha\beta} - \frac{\sigma^2}{\mathcal{H}}D_\nu\Psi. \end{aligned}$$

Para llevar al espacio de Fourier, hacemos:

```
EinsTimeSpace //. cd[α_][expr_] :> I Nk[] k[α] expr
EinsTimeSpaceInFourier = % /. FourierModes // Expand // ContractMetric
```

Donde ahora podemos proyectar respecto de $e^{1\nu}$. En nuestro caso se haría con la instrucción:

```
EinsTimeSpaceInFourier*e1[ν]
```

Haciendo lo mismo para $e^{2\nu}$, se obtiene entonces que la primera ecuación asociada a los modos vectoriales se puede escribir como:

$$\Phi_b = \frac{2i}{k}\sigma_{Vb}\left[\Phi + \Psi + \left(\frac{\Psi}{\mathcal{H}}\right)'\right] - \frac{4i}{k}\sum_{\lambda=+,\times}\sum_{a=1}^{2}\sigma_{Va}e_b{}^{\nu}e_a{}^{\beta}\varepsilon^{\lambda}{}_{\nu\beta}E_{\lambda}, \tag{4.10}$$

donde $b = 1, 2$.

Segunda ecuación escalar

La segunda ecuación escalar viene dada por $D^n\delta G_{0n} = D^n\kappa\delta T_{0n}$, así que ya con `EinsTimeSpace` expresada en términos de las invariantes de gauge, la divergencia espacial se calcula con:

```
cd[ν][EinsTimeSpace]
```

En este caso ν es el índice libre en `EinsTimeSpace`. Con esto, siguiendo los pasos detallados en C.3.6, se llega a que la forma explícita de la segunda ecuación escalar es:

$$\begin{aligned}\mathcal{H}D_\alpha D^\alpha\Phi - \frac{1}{2}\kappa\varphi' D_\alpha D^\alpha\chi + D_\alpha D^\alpha\Psi' = \frac{1}{2}\sigma^{\alpha\beta}D_\beta D_\alpha\left[\Phi + \Psi + \left(\frac{\Psi}{\mathcal{H}}\right)'\right] \\ + \frac{1}{2}\sigma^{\alpha\beta}D_\nu D^\nu E_{\alpha\beta} - \frac{1}{2\mathcal{H}}\sigma^2 D_\alpha D^\alpha\Psi.\end{aligned}$$

Llevando al espacio de Fourier, obtenemos entonces que la segunda ecuación asociada a los modos escalares viene dada por:

$$\mathcal{H}\Phi - \frac{1}{2}\kappa\varphi'\chi + \Psi' = \frac{1}{2}\sigma_{\parallel}\left[\Phi + \Psi + \left(\frac{\Psi}{\mathcal{H}}\right)'\right] - \frac{\sigma^2}{2\mathcal{H}}\Psi + \frac{1}{2}\sum_{\lambda=+,\times}E_\lambda\sigma_{T\lambda}.$$

Tercera ecuación escalar

La tercera ecuación escalar viene dada por la traza espacial de las ecuaciones de Einstein, es decir, $\gamma^{mn}\delta G_{mn} = \gamma^{mn}\kappa\delta T_{mn}$, por lo que primero debemos extraer las componentes espacio-espacio con la instrucción:

```
EinsSpaceSpace = ExtractComponents[EinsPertNewton,γ,{"Space","Space"}]
```

Ahora obtenemos la traza con:

```
γ[μ, ν] EinsSpaceSpace // Expand // ContractMetric
```

Reescribiendo el resultado anterior en términos de las variables invariantes de calibre e igualando a cero se llega entonces a que la tercera ecuación escalar viene dada por:

$$2\mathcal{H}^2\Phi + \mathcal{H}'\Phi + \mathcal{H}\Phi' - \frac{1}{2}\kappa\varphi'\chi' + 2\mathcal{H}\Psi' + \Psi'' + \frac{1}{3}D_\alpha D^\alpha(\Phi - \Psi) + \frac{1}{2}\kappa a^2\chi V_\varphi(\varphi) =$$
$$\frac{1}{2}E_{\alpha\beta}{}'\sigma^{\alpha\beta} - \frac{1}{2}\sigma^2\left[-2\Psi + \left(\frac{\Psi}{\mathcal{H}}\right)'\right] - \frac{1}{6\mathcal{H}}\sigma^{\alpha\beta}D_\beta D_\alpha\Psi - \frac{1}{2}\sigma_{\alpha\beta}D^\beta\Phi^\alpha.$$

Llevando al espacio de Fourier, siguiendo los pasos descritos en C.3.7, se obtiene la tercera ecuación asociada a los modos escalares:

$$2\mathcal{H}^2\Phi + \mathcal{H}'\Phi + \mathcal{H}\Phi' - \frac{1}{2}\kappa\varphi'\chi' + 2\mathcal{H}\Psi' + \Psi'' - \frac{1}{3}k^2(\Phi - \Psi) + \frac{1}{2}\kappa a^2\chi V_\varphi(\varphi) =$$
$$\frac{k^2}{6\mathcal{H}}\sigma_{\parallel}\Psi - \frac{1}{2}\sigma^2\left[-2\Psi + \left(\frac{\Psi}{\mathcal{H}}\right)'\right] - \frac{1}{2}\sum_{b=1}^{2} ik\sigma_{Vb}\Phi_b$$
$$+ \sum_{\lambda=+,\times}\left[\frac{1}{2}E'_\lambda\sigma_{T\lambda} - E_\lambda\sigma_{T\lambda}\sigma_{\parallel} + 2\sum_{a,b=1}^{2} e_a{}^i e_b{}^j\sigma_{Va}\sigma_{Vb}E_\lambda\varepsilon^\lambda{}_{ij}\right].$$

Ecuación tensorial

La ecuación tensorial corresponde a la parte sin traza de la componente espacio-espacio de las ecuaciones de Einstein. Esta puede calcularse con

$$\left(\delta G_{mn} - \frac{1}{3}\gamma^{ij}\delta T_{ij}\gamma_{mn}\right) = \kappa\left(\delta T_{mn} - \frac{1}{3}\gamma^{ij}\delta T_{ij}\gamma_{mn}\right),$$

o utilizando la función `STFPart`. Usando esta última, ejecutamos entonces la instrucción:

```
STFPart[EinsSpaceSpace, γ] // Expand;
```

donde `EinsSpaceSpace` es la componente espacio-espacio extraída en el caso anterior.

Realizando las manipulaciones algebraicas descritas en C.3.8 al resultado obtenido e igualando a cero, se obtiene que la parte sin traza de la componente espacio-espacio de las ecuaciones de Einstein de las perturbaciones viene dada por:

$$E''_{\mu\nu} + 2E'_{\mu\nu}\mathcal{H} - D_\alpha D^\alpha E_{\mu\nu} + \frac{1}{3}\gamma_{\mu\nu}D_\alpha D^\alpha(\Phi - \Psi) - 2\mathcal{H}D_{(\mu}\Phi_{\nu)}$$
$$- D_{(\mu}\Phi'_{\nu)} + D_\mu D_\nu(\Psi - \Phi) = 4E'_{(\mu\alpha}\sigma_{\mu)}{}^\alpha - 4E^{\alpha\beta}\sigma_{\mu\alpha}\sigma_{\nu\beta}$$
$$+ \sigma_{\mu\nu}\left[\Phi' + \Psi' + \left(\frac{\Psi}{\mathcal{H}}\right)'' - \frac{1}{\mathcal{H}}D_\alpha D^\alpha\Psi\right]$$
$$+ \frac{2}{\mathcal{H}}\sigma_{(\mu}{}^\alpha D_\alpha D_{\nu)}\Psi - 2\sigma_{(\mu\alpha}D^\alpha\Phi_{\nu)} - \frac{2}{3\mathcal{H}}\gamma_{\mu\nu}\sigma^{\alpha\beta}D_\alpha D_\beta\Psi.$$

Llevando ahora al espacio de Fourier y proyectando respecto ${\varepsilon^+}_{ij}$ y ${\varepsilon^\times}_{ij}$ se llega, respectivamente, a que las ecuaciones asociadas a los modos tensoriales vienen dadas por:

$$E_+'' + 2E_+'\mathcal{H} + k^2 E_+ = \frac{2}{3}E_+\sigma^2 + 2E_+(\sigma_\times)^2 - 2E_\times\sigma_\times\sigma_+ - \sqrt{2}ik\sigma_{V1}\Phi_1 + \sqrt{2}ik\sigma_{V2}\Phi_2 + \sigma_+\left[\Phi' + \Psi' + \left(\frac{\Psi}{\mathcal{H}}\right)'' + \frac{k^2\Psi}{\mathcal{H}}\right]$$

y

$$E_\times'' + 2E_\times'\mathcal{H} + k^2 E_\times = \frac{2}{3}E_\times\sigma^2 + 2E_\times(\sigma_+)^2 - 2E_+\sigma_\times\sigma_+ - \sqrt{2}ik\sigma_{V2}\Phi_1 - \sqrt{2}ik\sigma_{V1}\Phi_2 + \sigma_\times\left[\Phi' + \Psi' + \left(\frac{\Psi}{\mathcal{H}}\right)'' + \frac{k^2\Psi}{\mathcal{H}}\right].$$

Segunda ecuación vectorial

Ahora debemos encontrar la forma explícita de

$$D^n\left(\delta G_{mn} - \frac{1}{3}\gamma^{ij}\delta T_{ij}\gamma_{mn}\right) = \kappa D^n\left(\delta T_{mn} - \frac{1}{3}\gamma^{ij}\delta T_{ij}\gamma_{mn}\right).$$

En términos generales, esto se obtiene aplicando la derivada covariante espacial a resultado de la instrucción `STFPart`:

```
cd[ν]@STFPart[EinsSpaceSpace, γ] // Expand//ToCanonical
```

Con esto, luego de aplicar las reglas pertinentes, igual que en los casos anteriores, se llega a que la segunda ecuación vectorial viene dada por:

$$D_\alpha D^\alpha\left[\frac{2}{3}D_\mu(\Psi-\Phi) - \mathcal{H}\Phi_\mu - \frac{1}{2}\Phi_\mu'\right] = \sigma_{\mu\alpha}D^\alpha\left[\Phi' + \Psi' + \left(\frac{\Psi}{\mathcal{H}}\right)''\right] - \frac{1}{3\mathcal{H}}\sigma^{\alpha\beta}D_\alpha D_\beta D_\mu\Psi - 2\sigma^{\alpha\beta}D_\beta E_{\mu\alpha}' - \sigma_{\alpha\beta}D^\alpha D^\beta\Phi_\mu + \sigma_{\alpha\beta}D_\mu D^\alpha\Phi^\beta.$$

Llevando lo anterior al espacio de Fourier y proyectando respecto a e_1 y e_2, como se detalla en C.3.9, se obtiene:

$$\begin{aligned} 2\mathcal{H}\Phi_1 + \Phi_1' = {} & -\frac{2\sqrt{2}i}{k}E_+'\sigma_{V1} - \frac{2\sqrt{2}i}{k}E_\times'\sigma_{V2} - \frac{4i}{k}\sum_{\lambda=+,\times}E_\lambda\sigma_{T\lambda}\sigma_{V1} \\ & - \frac{2\sqrt{2}i}{k}E_+\sigma_\parallel\sigma_{V1} - \frac{2\sqrt{2}i}{k}E_\times\sigma_\parallel\sigma_{V2} - \frac{\sigma_+\Phi_1}{\sqrt{2}} - \frac{\sigma_\times\Phi_2}{\sqrt{2}} + \frac{5}{2}\sigma_\parallel\Phi_1 \\ & + \frac{2i}{k}\sigma_{V1}\left[\Phi' + \Psi' + \left(\frac{\Psi}{\mathcal{H}}\right)''\right] \end{aligned} \tag{4.11}$$

y

$$
\begin{aligned}
2\mathcal{H}\Phi_2 + \Phi_2' = &-\frac{2\sqrt{2}i}{k}E_\times'\sigma_{V1} - \frac{2\sqrt{2}i}{k}E_+'\sigma_{V2} - \frac{4i}{k}\sum_{\lambda=+,\times}E_\lambda\sigma_{T\lambda}\sigma_{V2} \\
&- \frac{2\sqrt{2}i}{k}E_\times\sigma_\parallel\sigma_{V1} + \frac{2\sqrt{2}i}{k}E_+\sigma_\parallel\sigma_{V2} - \frac{\sigma_\times\Phi_1}{\sqrt{2}} + \frac{\sigma_+\Phi_2}{\sqrt{2}} + \frac{5}{2}\sigma_\parallel\Phi_2 \\
&+ \frac{2i}{k}\sigma_{V2}\left[\Phi' + \Psi' + \left(\frac{\Psi}{\mathcal{H}}\right)''\right].
\end{aligned}
\tag{4.12}
$$

Con esto se han obtenido ya las ecuaciones de los modos asociados a la segunda ecuación vectorial, así que sólo falta encontrar la última ecuación de los modos escalares.

Cuarta ecuación escalar

Finalmente, la cuarta ecuación escalar viene dada por

$$
D^m D^n\left(\delta G_{mn} - \frac{1}{3}\gamma^{ij}\delta T_{ij}\gamma_{mn}\right) = \kappa D^m D^n\left(\delta T_{mn} - \frac{1}{3}\gamma^{ij}\delta T_{ij}\gamma_{mn}\right).
$$

Esto puede ser calculado a partir de lo ya encontrado o utilizando directamente la orden siguiente:

```
cd[μ]@cd[ν]@STFPart[EinsSpaceSpace, γ] // Expand//ToCanonical
```

Con lo anterior se obtiene:

$$
\begin{aligned}
\frac{2}{3}D_\alpha D^\alpha D_\beta D^\beta(\Psi - \Phi) = &\, 2\sigma_{\alpha\beta}D_\lambda D^\lambda D^\beta\Phi^\alpha - 4\sigma^{\alpha\beta}\sigma^{\lambda\mu}D_\mu D_\beta E_{\alpha\lambda} \\
&+ \sigma^{\alpha\beta}D_{\alpha\beta}\left[\Phi' + \Psi' + \left(\frac{\Psi}{\mathcal{H}}\right)'' + \frac{1}{3\mathcal{H}}D_\lambda D^\lambda\Psi\right].
\end{aligned}
$$

Y siguiendo los pasos presentados en C.3.10 se puede llevar la ecuación al espacio de Fourier, con lo cual se obtiene que la cuarta y última ecuación asociada a los modos escalares viene dado por:

$$
\begin{aligned}
\frac{2}{3}k^2(\Phi - \Psi) = &-2\sqrt{2}E_+(\sigma_{V1})^2 - 4\sqrt{2}E_\times\sigma_{V1}\sigma_{V2} - 2\sqrt{2}E_+(\sigma_{V2})^2 \\
&+ 2ki\sum_{a=1,2}\sigma_{Va}\Phi_a + \sigma_\parallel\left[\Phi' + \Psi' + \left(\frac{\Psi}{\mathcal{H}}\right)'' - \frac{k^2\Psi}{3\mathcal{H}}\right],
\end{aligned}
$$

5. Discusión

Acerca de los resultados obtenidos

En la primera parte de la sección anterior, partiendo de la acción de Hilbert-Einstein y usando las funciones de xPert, se obtuvo la forma general de las ecuaciones de campo de Einstein y la ecuación de movimiento para la densidad lagrangiana del campo escalar canónico. Luego se utilizó xPand para perturbar estas expresiones y así obtener sus formas explícitas en el modelo Bianchi I, llegando entonces a las ecuaciones de Klein-Gordon y de Friedmann, tanto para los valores de fondo como para sus perturbaciones lineales. Al hacer esto se encontró no sólo que en un modelo homogéneo como el Bianchi I los valores de fondo de los campos escalares deben ser igualmente homogéneos (ecuación 4.3), sino que la ecuación de Klein-Gordon conserva la misma forma que en el modelo FLRW plano. Las ecuaciones de Friedmann, por su parte, presentan diferencias respecto a este último, debido a la aparición del término σ^2 en ellas (ecuaciones 4.4 y 4.6). La diferencia más marcada, sin embargo, se da en la aparición de una tercera ecuación de Friedmann (4.7), vinculada a la parte sin traza de la evolución temporal del tensor de corte. Pero recordemos que como en este caso se consideró vorticidad igual a cero, se pudo establecer una relación entre la expansión, el tensor de corte y la curvatura extrínseca (ecuación 2.3). Más aún, como la separación 3+1 se hizo a partir de la métrica conforme y esto llevó a que la expansión sea nula, hablar del tensor de corte es hablar del tensor de curvatura extrínseca ($K_{ij} = \sigma_{ij}$)[18], lo que hace notar que estas ecuaciones en realidad están añadiendo el efecto de este tensor. O, dicho de otra manera, están teniendo en cuenta la forma en que la hipersuperficie está embebida en la variedad 4-dimensional. Esto refleja el aspecto geométrico que subyace en ellas, dejando a las ecuaciones de Friedmann del modelo FLRW plano como un caso particular en el que el tensor de curvatura extrínseca es cero.

Lo anterior fue en cuanto a las ecuaciones de fondo. Si pensamos ahora en las ecuaciones de las perturbaciones, se encuentra que la correspondiente a la ecuación de Klein-Gordon no involucran el tensor de corte y conserva la forma de su homóloga en el modelo FLRW. Por su parte, las perturbaciones de las ecuaciones de Einstein sí lo involucran. Y, de hecho, lo hacen de tal manera que estas quedan acopladas. Esto puede verse de forma más clara al revisarlas en el espacio de Fourier, en donde podemos notar que incluso si se reducen las ecuaciones tomando los modos vectoriales (ecuaciones 4.10-4.12) y reemplazándolos en las otras ecuaciones, al final se tendrían ecuaciones más compactas que quedarían expresadas completamente en términos de los modos tensoriales y escalares. Pero esto no significa que se hayan desacoplado o anulado las cantidades vectoriales, sino que sus efectos han quedado absorbidos en el acople dado entre los otros modos. Despreciar los modos vectoriales sería eliminar su

[18] De hecho, esta fue la razón de usar la instrucción `PrintAs[Kγ] = "σ"`, que indica al software que la curvatura extrínseca de la hipersuperficie asociada a la métrica conforme debe presentarse en pantalla como el tensor de corte, ya que xPand opera directamente en términos de K_{ij} y es el usuario quien lo identifica con el corte.

contribución a la relación final entre los modos tensoriales y escalares, por lo que vale la pena hacer la aclaración.

Este acople entre los modos difiere completamente de lo que ocurre con el modelo FLRW, en donde las perturbaciones correspondientes a los modos escalares, vectoriales y tensoriales están completamente desacopladas, por lo que pueden estudiarse por separado. Esto es lo que se presenta en la referencia [3] como **teorema de descomposición** y, de hecho, justifica por qué xPand posee, como se mencionó en la sección 3, una instrucción para «apagar» las contribuciones vectoriales y/o tensoriales de las perturbaciones de manera completamente independiente. Sin embargo, como vemos, esto ya no es válido para el modelo Bianchi I, por lo que se debe tener especial cuidado en estos casos, ya que en Bianchi I no es posible estudiar de manera independiente las ecuaciones asociadas a los modos escalares, vectoriales y tensoriales.

A pesar de todo lo descrito anteriormente, puede observarse que si el tensor de corte se anula, las ecuaciones se desacoplan completamente y se recuperan las expresiones asociadas al modelo FLRW plano, tanto para el fondo como para las perturbaciones, lo que permite concluir que, bajo las condiciones trabajadas, la introducción de la anisotropía del modelo Bianchi I se ve reflejada en la aparición del tensor de corte en las ecuaciones de Einstein, siendo este entonces el elemento clave que marca la diferencia respecto al modelo isotrópico plano, al cual se retorna siempre que dicho tensor se anule.

Acerca del algoritmo

El primer aspecto que hay que destacar de la implementación y, en general, de xPand y de la suite xAct, es que puede observarse que una buena parte del trabajo con estas herramientas radica en definir reglas de reemplazo adecuadas que permitan alcanzar el resultado deseado, desempeñando así un papel central dentro del algoritmo, donde pueden entenderse como la axiomática básica sobre la cual reposa el desarrollo matemático del programa, por lo que deben aplicarse una y otra vez, incluso de manera consecutiva, hasta alcanzar el resultado final[19], como se ilustró en la sección 4.3 y en el apéndice C. Con estas mismas ideas incluso se logró demostrar que si bien xPand no implementa por defecto los resultados en el espacio de Fourier, es posible hacerlo a partir de reglas definidas sobre la base del mismo paquete y de xTensor, logrando extender así las funcionalidades del software. Y a pesar de que en este trabajo la transformación a dicho espacio se realizó a ecuaciones ya manipuladas en su forma diferencial, esto no necesariamente tiene por qué ser así; bien podría haberse aplicado directamente a las expresiones perturbadas que devuelve xPand sin que haya ninguna diferencia. El decidir hacer la manipulación de las expresiones directamente en su forma diferencial obedece principalmente a fines ilustrativos, para mostrar de forma práctica cómo a partir de reglas se puede forzar la aparición de divergencias y laplacianos en las ecuaciones[20]. La utilidad de esto radica en que a pesar de que xPand posee algoritmos internos

[19]Estas reglas actúan de manera similar a como lo hace la restricción de que, por ejemplo, una determinada base de $\mathbb{R}^3$ sea ortonormal.

[20]Esto es útil para que los vectores y tensores sin divergencia se anulen.

automáticos para ello [5], en el desarrollo del proyecto esto no siempre se vio reflejado en los resultados, haciendo que el programa realizara cálculos adicionales innecesarios y presentando expresiones más complejas y extensas de lo que realmente eran, desviando la atención a términos que deberían haberse anulado.

Finalmente, valga recordar que en este caso se trabajó en un calibre en particular, el de Newton. Esto resultó ser de utilidad para reducir el número de pasos intermedios para llegar a expresar las perturbaciones en términos de las invariantes de gauge introducidas en la sección 2.4. Con esto sólo fue necesario definir la regla del calibre[21], la de transformación a las invariantes y luego aplicarlas, algo que se pudo hacerse en pocas líneas de código. Vemos entonces que hacer uso de los calibres puede facilitar el trabajo no sólo cuando se trabaja a mano, sino también en la escritura de código, ya la que la manipulación de las expresiones se hace más simple y directa. Si en este caso se necesita luego trabajar en otro calibre, basta con definir una regla que aplique las transformaciones 2.22-2.24, 2.26 y luego elegir el gauge deseado.

Acerca de la herramienta

En cuanto al potencial del software, vale la pena mencionarse que si bien el trabajo en este caso se ha limitado a primer orden, el desarrollo puede extenderse a órdenes superiores según la necesidad. Debe tenerse en cuenta, sin embargo, que el tiempo de cómputo también incrementa con el orden y la complejidad de las expresiones a perturbar. Los desarrollos realizados aquí se probaron en dos computadores convencionales (ver apéndice B). El tiempo promedio de ejecución de la instrucción `ToxPand` sobre las ecuaciones de Einstein para el campo escalar canónico, fue de 24 segundos en el más lento y de 14 segundos con el más rápido. Teniendo en cuenta que esta es la instrucción que realiza el cálculo propiamente dicho de las perturbaciones y que los procesadores utilizados poseen frecuencias que bien pueden considerarse «normales» para los estándares actuales, es claro que el trabajo a primer orden con la herramienta no requiere de gran poder de cómputo y su ejecución es bastante rápida si se le compara con el trabajo manual (ver apéndice A).

En línea con el rendimiento, la mayor dificultad que se pudo experimentar con el uso de la herramienta se dio en su consumo de memoria RAM. Si bien en este proyecto no se realizaron pruebas cuantitativas al respecto, cualitativamente se observó que el trabajo con 4GB de memoria RAM puede verse entorpecido y algunas veces imposible si se desea trabajar en más de dos núcleos de Mathematica en simultánea, algo que suele ser útil cuando se desarrollan nuevas rutinas o cuando se está en proceso de aprendizaje y es necesario utilizar las mismas definiciones de un archivo base sin causar conflictos en la sesión. Esto, por otra parte, no debe entenderse como una dificultad de xPand o los paquetes de xAct, sino del software base, es decir, Wolfram Mathematica.

[21] Recordemos que aquí se definió la regla del calibre manualmente sólo para ser estrictos con la ecuación 2.25, pero dado que se quería expresar todo en términos de los potenciales de Bardeen, hacer esto no es realmente relevante.

Apuntes del autor

Un detalle final que sería interesante destacar, aunque ya desde una perspectiva algo más subjetiva, es el potencial pedagógico de las herramientas de software utilizadas durante este proyecto, puesto que al formarse de una disciplina como lo es la Física, el trabajo teórico-experimental es algo de gran ayuda a la hora afianzar conocimientos e interiorizar ideas. Si bien durante el pregrado esto se hace desde los laboratorios o desde herramientas como Interactive Physics [19], la labor se hace un poco más complicada cuanto más se trabaja con ideas o conceptos matemáticos más abstractos. Herramientas como xPand, xTensor o xPert permiten, por una parte, que el estudiante pueda verificar cálculos realizados a mano mientras se familiariza con las ideas. Y por otra, que pueda modificar o explorar de forma relativamente rápida (si se le compara con el trabajo manual) variaciones de sus cálculos, modelos o de las condiciones establecidas sobre los problemas, haciendo las veces de un laboratorio virtual útil para quienes apenas se adentran en áreas como la Relatividad General, la Cosmología o la Geometría Diferencial, donde incluso el entender la documentación del software o sus instrucciones, obliga al estudiante a sumergirse en el formalismo que hay detrás de ella.

6. Conclusiones y perspectivas

En este trabajo se ha presentado xPand como una alternativa viable para el cálculo de perturbaciones cosmológicas. Se ha presentado la teoría matemática y física básica necesaria para hacer uso de la herramienta en espaciotiempos Bianchi I y se ha aplicado al caso particular del campo escalar canónico. En concreto,

- Se ha utilizado xPand para generar las ecuaciones de Friedmann para el espaciotiempo Bianchi I como una perturbación a orden cero de las ecuaciones de Einstein. Con esto se confirma, de forma natural, la homogeneidad del campo escalar de fondo en este modelo y la aparición de una tercera ecuación de Friedmann vinculada a la evolución del tensor de corte. Las ecuaciones de fondo correspondientes al modelo de FLRW quedan incluidas aquí como un caso particular en el que el tensor de corte es cero.

- Se ha empleado el software para generar la forma explícita de la ecuación de movimiento del campo escalar canónico en el espaciotiempo Bianchi I, tanto para el fondo (ecuación de Klein-Gordon) como para las perturbaciones lineales, encontrando que ninguna estas no depende del tensor de corte y conservan la misma forma que en el modelo FLRW plano.

- Se ha utilizado el software para calcular las perturbaciones de las ecuaciones de Einstein para el campo escalar canónico en su forma diferencial en el espaciotiempo Bianchi I, encontrando que estas sí dependen del tensor de corte y, aunque no conservan la forma del modelo FLRW plano, sí se recupera su forma cuando el corte se anula.

- Con todo lo anterior, se puede concluir que bajo las condiciones trabajadas, la introducción de la anisotropía del modelo Bianchi I se ve reflejada directamente en la aparición del tensor de corte en las ecuaciones de Einstein, tanto para el fondo, como para sus perturbaciones, siendo este entonces el elemento clave que marca la diferencia respecto al modelo FLRW plano y al cual se retorna siempre que dicho tensor se anule.

- Se utilizó xTensor, y las herramientas del lenguaje Mathematica para llevar las ecuaciones de las perturbaciones de su forma diferencial al espacio de Fourier, mostrando que si bien xPand no trabaja por defecto en el espacio de Fourier, sus resultados pueden ser extendidos a él mediante tales herramientas.

- Con lo anterior se encuentra también que a diferencia de lo que ocurre en el modelo FLRW plano, los aportes escalares, vectoriales y tensoriales se encuentran acoplados por el tensor de corte. Más aún, los modos vectoriales quedan completamente contenidos en el acople entre los modos escalares y los modos tensoriales. Esto permite afirmar que en espaciotiempos Bianchi I no sólo no es válido el teorema de descomposición, sino que el despreciar los modos vectoriales sería ignorar

su contribución a la relación entre los modos escalares y tensoriales. Se tiene así entonces que las contribuciones de los modos no pueden estudiarse de manera independiente, a como sí ocurre en el modelo FLRW plano.

En el desarrollo de este proyecto se generó un archivo de Mathematica con todo el código utilizado, para que pueda ser estudiado y modificado por los estudiantes del departamento de Física de la Universidad del Valle interesados en familiarizarse con el uso de las herramientas trabajadas. Se espera que este trabajo contribuya a brindar una primera aproximación básica al estudio de las perturbaciones lineales en espaciotiempos Bianchi I a través de paquetes de álgebra computacional.

Trabajos futuros

Puede observarse que las ecuaciones de las perturbaciones lineales obtenidas en este caso aún pueden ser reducidas, por lo que el siguiente paso natural es identificar las variables invariantes de calibre asociadas a los grados de libertad físicos. Aprovechando que en este trabajo se han presentado los elementos básicos necesarios para hacer uso del paquete xPand se propone como trabajo futuro el utilizar los conocimientos proporcionados aquí para explorar el algoritmo de la referencia [20]. En él se presenta un método sistemático para aislar variables asociadas a grados de libertad físicos utilizando xTensor y xPert. Se podría además revisar la compatibilidad de su código con xPand y ver si la inclusión de este podría contribuir a facilitar la extensión del algoritmo a más casos de interés.

APÉNDICES

A. Perturbación del escalar de Ricci

Con el fin de ofrecer una comparativa entre el trabajo a mano y el trabajo realizado con xPand, a continuación se presenta el cálculo manual de la perturbación del escalar de Ricci en un fondo FLRW plano. Para esto, básicamente necesitamos calcular el escalar de Ricci, pero esto no es más que la traza respecto a la métrica perturbada del tensor homónimo, el cual viene dado por:

$$R_{\mu\nu} = \Gamma^{\alpha}{}_{\mu\nu,\alpha} - \Gamma^{\alpha}{}_{\mu\alpha,\nu} + \Gamma^{\alpha}{}_{\beta\alpha}\Gamma^{\beta}{}_{\mu\nu} - \Gamma^{\alpha}{}_{\beta\nu}\Gamma^{\beta}{}_{\mu\alpha},$$

donde $\Gamma^{\alpha}{}_{\mu\nu,\beta} \equiv \partial_\beta \Gamma^{\alpha}{}_{\mu\nu}$. Con esto entonces el escalar de Ricci viene dado por:

$$\begin{aligned} R &= g^{\mu\nu} R_{\mu\nu} \\ &= g^{0\nu} R_{0\nu} + g^{m\nu} R_{m\nu} \\ &= g^{00} R_{00} + \cancelto{0}{g^{m0}} R_{m0} + \cancelto{0}{g^{0n}} R_{0n} + g^{mn} R_{mn} \\ &= g^{00} R_{00} + g^{mn} R_{mn}. \end{aligned} \tag{A.1}$$

Para poder avanzar, necesitamos conocer entonces la forma explícita de g^{00}, g^{mn}, R_{00} y R_{mn}. Empecemos por las componentes del tensor de Ricci:

R_{00}**:**

$$\begin{aligned} R_{00} &= \Gamma^{\alpha}{}_{00,\alpha} - \Gamma^{\alpha}{}_{0\alpha,0} + \Gamma^{\alpha}{}_{\beta\alpha}\Gamma^{\beta}{}_{00} - \Gamma^{\alpha}{}_{\beta 0}\Gamma^{\beta}{}_{0\alpha} \\ &= \cancel{\Gamma^{0}{}_{00,0}} + \Gamma^{i}{}_{00,i} - \cancel{\Gamma^{0}{}_{00,0}} - \Gamma^{i}{}_{0i,0} + \Gamma^{\alpha}{}_{\beta\alpha}\Gamma^{\beta}{}_{00} - \Gamma^{\alpha}{}_{\beta 0}\Gamma^{\beta}{}_{0\alpha} \\ &= \Gamma^{i}{}_{00,i} - \Gamma^{i}{}_{0i,0} + \Gamma^{\alpha}{}_{0\alpha}\Gamma^{0}{}_{00} + \Gamma^{\alpha}{}_{i\alpha}\Gamma^{i}{}_{00} - \Gamma^{\alpha}{}_{00}\Gamma^{0}{}_{0\alpha} - \Gamma^{\alpha}{}_{i0}\Gamma^{i}{}_{0\alpha} \\ &= \Gamma^{i}{}_{00,i} - \Gamma^{i}{}_{0i,0} + \cancel{\Gamma^{0}{}_{00}\Gamma^{0}{}_{00}} + \Gamma^{i}{}_{0i}\Gamma^{0}{}_{00} + \cancel{\Gamma^{0}{}_{i0}\Gamma^{i}{}_{00}} \\ &\quad + \Gamma^{j}{}_{ij}\Gamma^{i}{}_{00} - \cancel{\Gamma^{0}{}_{00}\Gamma^{0}{}_{00}} - \Gamma^{i}{}_{00}\Gamma^{0}{}_{0i} - \cancel{\Gamma^{0}{}_{i0}\Gamma^{i}{}_{00}} - \Gamma^{j}{}_{i0}\Gamma^{i}{}_{0j} \\ &= \Gamma^{i}{}_{00,i} - \Gamma^{i}{}_{0i,0} + \Gamma^{i}{}_{0i}\Gamma^{0}{}_{00} + \Gamma^{j}{}_{ij}\Gamma^{i}{}_{00} - \Gamma^{i}{}_{00}\Gamma^{0}{}_{0i} - \Gamma^{j}{}_{i0}\Gamma^{i}{}_{0j}, \end{aligned}$$

R_{mn}**:**

$$\begin{aligned} R_{mn} &= \Gamma^{\alpha}{}_{mn,\alpha} - \Gamma^{\alpha}{}_{m\alpha,n} + \Gamma^{\alpha}{}_{\beta\alpha}\Gamma^{\beta}{}_{mn} - \Gamma^{\alpha}{}_{\beta n}\Gamma^{\beta}{}_{m\alpha} \\ &= \Gamma^{0}{}_{mn,0} + \Gamma^{i}{}_{mn,i} - \Gamma^{0}{}_{m0,n} - \Gamma^{i}{}_{mi,n} \\ &\quad + \Gamma^{0}{}_{00}\Gamma^{0}{}_{mn} + \Gamma^{0}{}_{i0}\Gamma^{i}{}_{mn} + \Gamma^{i}{}_{0i}\Gamma^{0}{}_{mn} + \Gamma^{i}{}_{ji}\Gamma^{j}{}_{mn} \\ &\quad - \Gamma^{0}{}_{0n}\Gamma^{0}{}_{m0} - \Gamma^{0}{}_{in}\Gamma^{i}{}_{m0} - \Gamma^{i}{}_{0n}\Gamma^{0}{}_{mi} - \Gamma^{i}{}_{jn}\Gamma^{j}{}_{mi}. \end{aligned}$$

Vemos que en ambos casos todo queda expresado en términos de los símbolos de Christoffel, los cuales vienen dados por:

$$\Gamma^{\mu}{}_{\alpha\beta} = \frac{g^{\mu\nu}}{2}\left[\partial_\beta g_{\alpha\nu} + \partial_\alpha g_{\beta\nu} - \partial_\nu g_{\alpha\beta}\right],$$

donde ∂_α es el operador de derivada parcial respecto a α. Aquí vemos que para lograr nuestro cometido, necesitamos la forma explícita de la métrica perturbada, que por el teorema de descomposición [3], podemos hacer que sólo contenga perturbaciones escalares. Si en este caso elegimos el calibre de Newton, la métrica perturbada puede ser expresada como:

$$g_{\mu\nu} = \begin{pmatrix} -1-2\psi & 0 & 0 & 0 \\ 0 & a^2(1+2\phi) & 0 & 0 \\ 0 & 0 & a^2(1+2\phi) & 0 \\ 0 & 0 & 0 & a^2(1+2\phi) \end{pmatrix},$$

donde $\psi = \psi(t, \{x^i\})$ y $\phi = \phi(t, \{x^i\})$ [22], ambas funciones mucho menores que uno. Así mismo tenemos que:

$$g^{\mu\nu} = (g_{\mu\nu})^{-1} = \begin{pmatrix} (-1-2\psi)^{-1} & 0 & 0 & 0 \\ 0 & a^{-2}(1+2\phi)^{-1} & 0 & 0 \\ 0 & 0 & a^{-2}(1+2\phi)^{-1} & 0 \\ 0 & 0 & 0 & a^{-2}(1+2\phi)^{-1} \end{pmatrix}.$$

Pero como estamos tratando con perturbaciones, podemos expandir a primer orden, con lo que tenemos que

$$\begin{aligned} g^{0\nu} &\approx (-1+2\psi)\delta^\nu{}_0 \\ g^{i\nu} &\approx a^{-2}(1-2\phi)\delta^\nu{}_i. \end{aligned}$$

Con esto ya se puede iniciar el cálculo de los símbolos de Christoffel:

Cálculo de $\Gamma^0{}_{00}$

$$\begin{aligned} \Gamma^0{}_{00} &= \frac{g^{\nu 0}}{2}\left[\partial_0 g_{0\nu} + \partial_0 g_{0\nu} - \partial_\nu g_{00}\right] \\ &\approx \frac{1}{2}(-1+2\psi)\delta^\nu{}_0\left[\partial_0 g_{0\nu} + \partial_0 g_{0\nu} - \partial_\nu g_{00}\right] \\ &\approx \frac{1}{2}(-1+2\psi)\partial_0(-1-2\psi) \\ &\approx \dot{\psi}, \end{aligned}$$

donde se han conservado sólo los términos de primer orden en las perturbaciones. Lo mismo se hará para los cálculos siguientes.

[22] La convención que maneja xPand es con los nombres opuestos y el signo invertido en la parte espacial, es decir: $\phi \to \psi$ y $\psi \to -\phi$.

Cálculo de $\Gamma^0{}_{0i}$

$$\begin{aligned}\Gamma^0{}_{0i} &= \frac{g^{0\nu}}{2}\left[\partial_i g_{0\nu} + \partial_0 g_{i\nu} - \partial_\nu \cancelto{0}{g_{0i}}\right] \\ &\approx \frac{1}{2}(-1+2\psi)\delta^\nu{}_0\left[\partial_i g_{0\nu} + \partial_0 g_{i\nu}\right] \\ &\approx \frac{1}{2}(-1+2\psi)\partial_i(-1-2\psi) \\ &\approx \partial_i\psi.\end{aligned}$$

Además, $\Gamma^0{}_{i0} = \Gamma^0{}_{i0} \approx \partial_i\psi$.

Cálculo de $\Gamma^0{}_{ij}$

$$\begin{aligned}\Gamma^0{}_{ij} &= \frac{g^{0\nu}}{2}\left[\partial_j g_{i\nu} + \partial_i g_{j\nu} - \partial_\nu g_{ij}\right] \\ &\approx \frac{1}{2}(-1+2\psi)\delta^\nu{}_0\left[\partial_j[a^2(1+2\phi)\delta_{\nu i}] + \partial_i[a^2(1+2\phi)\delta_{\nu j}] - \partial_\nu[a^2(1+2\phi)\delta_{ij}]\right] \\ &\approx \frac{1}{2}(-1+2\psi)[-\partial_0[a^2(1+2\phi)\delta_{ij}]] \\ &\approx (-1+2\psi)[-a\dot{a}(1+2\phi) - a^2\dot{\phi}]\delta_{ij} \\ &\approx a^2[H + 2H(\phi-\psi) + \dot{\phi}]\delta_{ij},\end{aligned}$$

donde $H \equiv \frac{\dot{a}}{a}$.

Cálculo de $\Gamma^i{}_{00}$

$$\begin{aligned}\Gamma^i{}_{00} &= \frac{g^{i\nu}}{2}\left[\partial_0 g_{0\nu} + \partial_0 g_{0\nu} - \partial_\nu g_{00}\right] \\ &\approx \frac{1}{2}a^{-2}(1-2\phi)\delta^\nu{}_i\left[\partial_0 g_{0\nu} + \partial_0 g_{0\nu} - \partial_\nu g_{00}\right] \\ &\approx \frac{1}{2}a^{-2}(1-2\phi)\left[-\partial_i[-(1+2\psi)]\right] \\ &\approx \frac{\partial_i\psi}{a^2}.\end{aligned}$$

Cálculo de $\Gamma^i{}_{0j}$

$$\begin{aligned}
\Gamma^i{}_{0j} &= \frac{g^{i\nu}}{2}\left[\partial_j g_{0\nu} + \partial_0 g_{j\nu} - \partial_\nu \cancelto{0}{g_{0j}}\right] \\
&\approx \frac{1}{2}a^{-2}(1-2\phi)\delta^\nu{}_i\left[\partial_j g_{0\nu} + \partial_0 g_{j\nu}\right] \\
&\approx \frac{1}{2}a^{-2}(1-2\phi)\left[\partial_0[a^2(1+2\phi)\delta_{ji}]\right] \\
&\approx \frac{1}{2}a^{-2}(1-2\phi)\left[2a\dot{a}(1+2\phi) + 2a^2\dot{\phi}\right] \\
&\approx (1-2\phi)\left[H(1+2\phi) + \dot{\phi}\right] \\
&\approx [H+\dot{\phi}]\delta_{ji}.
\end{aligned}$$

Además, $\Gamma^i{}_{j0} = \Gamma^i{}_{0j} \approx [H+\dot{\phi}]\delta_{ji}$.

Cálculo de $\Gamma^i{}_{0j}$

$$\begin{aligned}
\Gamma^n{}_{ij} &= \frac{g^{n\nu}}{2}\left[\partial_j g_{i\nu} + \partial_i g_{j\nu} - \partial_\nu g_{ij}\right] \\
&\approx \frac{a^{-2}}{2}(1-2\phi)\delta^{n\nu}\left[\partial_j g_{i\nu} + \partial_i g_{j\nu} - \partial_\nu g_{ij}\right] \\
&\approx \frac{a^{-2}}{2}(1-2\phi)\left[\partial_j[a^2(1+2\phi)\delta_{in}] + \partial_i[a^2(1+2\phi)\delta_{jn}] - \partial_n[a^2(1+2\phi)\delta_{ij}]\right] \\
&\approx (1-2\phi)\left[\partial_j\phi\delta_{in} + \partial_i\phi\delta_{jn} - \partial_n\phi\delta_{ij}\right] \\
&\approx \partial_j\phi\delta_{in} + \partial_i\phi\delta_{jn} - \partial_n\phi\delta_{ij}.
\end{aligned}$$

Ahora reemplazamos los símbolos de Christoffel correspondientes en la componente tiempo-tiempo del tensor de Ricci:

$$R_{00} \approx \partial_i\left[-\frac{1}{a^2}\partial_i\psi\right] - 3\partial_0[H+\dot{\phi}] + 3[H+\dot{\phi}]\dot{\psi} + \frac{3}{a^2}\partial_i\phi\ \partial_i\psi - \left(\frac{\partial_i\psi}{a^2}\right)\partial_i\psi - 3[H+\dot{\phi}]^2.$$

Si nos quedamos a primer orden en las perturbaciones obtenemos:

$$R_{00} \approx -\frac{1}{a^2}\partial_i\partial_i\psi - 3[\dot{H}+\ddot{\phi}] + 3H\dot{\psi} - 3[H^2 + 2H\dot{\phi}]. \tag{A.2}$$

Ahora hacemos lo mismo, pero para la componente espacio-espacio:

$$\begin{aligned}
R_{mn} = {} & \partial_0(\delta_{mn}a^2[H + 2H(\phi-\psi) + \dot{\phi}]) + \partial_i([\delta_{mi}\partial_n + \delta_{ni}\partial_m - \partial_{mn}\partial_i]\phi) \\
& + \partial_n\partial_m\psi - \delta_n([\delta_{mi}\delta_i + \delta_{ii}\delta_m - \delta_{mi}\delta_i]\phi) + \dot{\psi}\delta_{mn}a^2(H + 2H[\phi-\psi] + \dot{\phi}) \\
& + (H+\dot{\phi})\delta_{ii}\delta_{mn}a^2(H + 2H[\phi-\psi] + \dot{\phi}) \\
& - \delta_{in}a^2(H + 2H[\phi-\psi] + \dot{\phi})(H+\dot{\phi})\delta_{mi} \\
& - (H+\dot{\phi})\delta_{ni}\delta_{mi}a^2(H + 2H[\phi-\psi] + \dot{\phi}),
\end{aligned}$$

donde de entrada se han despreciado los términos correspondientes a ${\Gamma^0}_{i0}{\Gamma^i}_{mn}$, ${\Gamma^i}_{ji}{\Gamma^j}_{mn}$, ${\Gamma^0}_{0n}{\Gamma^0}_{m0}$ y ${\Gamma^i}_{jn}{\Gamma^j}_{mi}$, ya que se sólo contribuyen con términos de segundo orden. Simplificando la expresión y quedándose a primer orden en las perturbaciones, se llega a que

$$R_{mn} = \delta_{mn}\left[(2H^2a^2 + a\dot{a})(1 + 2[\phi - \psi]) + a^2H(6\dot{\phi} - \dot{\psi}) + a^2\ddot{\phi} - \partial_i\partial_i\phi\right] - \partial_n\partial_m(\psi + \phi). \tag{A.3}$$

Con esto ya podemos encontrar el escalar de Ricci reemplazando A.2 y A.3 en la ecuación A.1:

$$\begin{aligned} R &= g^{00}R_{00} + g^{mn}R_{mn} \\ &\approx (-1 + 2\psi)\left(-\frac{1}{a^2}\partial_i\partial_i\psi - 3[\dot{H} + \ddot{\phi}] + 3H\dot{\psi} - 3[H^2 + 2H\dot{\phi}]\right) \\ &\quad + \frac{1}{2}(1 - 2\phi)\delta^{mn}\left(\delta_{mn}\left[(2H^2a^2 + a\dot{a})(1 + 2[\phi - \psi]) + a^2H(6\dot{\phi} - \dot{\psi}) + a^2\ddot{\phi} - \partial_i\partial_i\phi\right]\right) \\ &\quad + \frac{1}{2}(1 - 2\phi)\delta^{mn}\left(-\partial_n\partial_m(\psi + \phi)\right). \end{aligned}$$

Si se expanden los productos y nos quedamos en el orden lineal, se llega finalmente a que:

$$\begin{aligned} R &= 6\underbrace{\left[H^2 + \frac{\ddot{a}}{a}\right]}_{\dot{H}+2H^2} + 6\ddot{\phi} - 6H[\dot{\phi} - 4\dot{\phi}] - \frac{4}{a^2}\partial_i\partial_i\phi - \frac{2}{a^2}\partial_i\partial_i\psi - 12\psi\underbrace{\left[H^2 + \frac{\ddot{a}}{a}\right]}_{\dot{H}+2H^2}. \\ &= 6\left[\dot{H} + 2H^2\right] + 6\ddot{\phi} - 6H[\dot{\phi} - 4\dot{\phi}] - \frac{4}{a^2}\partial_i\partial_i\phi - \frac{2}{a^2}\partial_i\partial_i\psi - 12\psi\left[\dot{H} + 2H^2\right]. \end{aligned}$$

Pero aquí vemos que en el primer término, $6\left[\dot{H} + 2H^2\right]$, no intervienen las perturbaciones y eso es porque corresponde al valor de fondo del escalar de Ricci. Por tanto, su perturbación viene dada por:

$$\delta R = 6\ddot{\phi} - 6H[\dot{\psi} - 4\dot{\phi}] - \frac{4}{a^2}\partial_i\partial_i\phi - \frac{2}{a^2}\partial_i\partial_i\psi - 12\psi\left[\dot{H} + 2H^2\right].$$

Para reescribir en términos de la notación que se ha manejado en este trabajo para espaciotiempos Bianchi I, sólo es necesario tener en cuenta que la derivada parcial espacial coincide con la derivada covariante asociada a la métrica inducida, por lo que $\partial_i\partial_i \to D_iD^i$ o $D_\alpha D^\alpha$, si consideramos su extensión 4-dimensional. En esta notación tenemos además $\psi \to A$ y $\phi \to -C$, con lo cual se obtiene:

$$\delta R = -6\ddot{C} - 6H[\dot{A} + 4\dot{C}] + \frac{4}{a^2}D_iD^iC - \frac{2}{a^2}D_iD^iA - 12A\left[\dot{H} + 2H^2\right],$$

que es justamente la ecuación 3.2 y que puede extraerse con el siguiente código:

```
<<xAct`xPand`;
DefManifold[M,4,{α,β,γ,μ,ν,λ,σ}];
DefMetric[-1,g[-α, -β], CD, {";", "∇"}, PrintAs -> "ḡ"];
SetSlicing[g, n, h, cd, {"|", "D"}, "FLFlat"];
PrintAs[φγ] ^= "A";
PrintAs[ψγ] ^= "C";
DefMetricFields[g, dg, h];
MyToxPand[expr_,gauge_,order_]:=ToxPand[expr,dg,u,du,h,gauge,order]
$ConformalTime = False;
MyToxPand[g[-μ, -ν], "NewtonGauge", 1];
ExtractOrder[MyToxPand[RicciScalarCD[], "NewtonGauge", 1],1]
```

Vemos entonces que en pocas líneas de código se ahorra el trabajo realizado de forma manual, que si bien aquí no resulta ser algo complicado, es difícil decir lo mismo en el caso de expresiones más complejas en donde se trabajan incluso espaciotiempos más generales.

B. Recursos

El desarrollo y la prueba del código se realizó en dos equipos diferentes con sistema operativo GNU/Fedora Linux 37. El primero de ellos con las siguientes características:

- Procesador: Intel(R) Core(TM) i3-2120 CPU @ 3.30GHz.
- Arquitectura: x86_64.
- Frecuencia máxima: 3,3 GHz.
- Frecuencia mínima: 1,6 GHz.
- RAM: 12GB a 3333MHz.
- HDD: 500 GB.
- Wolfram Mathematica v12.2.

El segundo equipo con las siguientes características:

- Procesador: Intel(R) Core(TM) i3-1005G1 CPU @ 1.20GHz.
- Arquitectura: x86_64.
- Frecuencia máxima: 3,4 GHz.
- Frecuencia mínima: 0,4 GHz.
- RAM: 8GB a 2667MHz.
- SSD: 128 GB.
- Wolfram Mathematica v13.1.

En ambos equipos se trabajó inicialmente con una memoria RAM de 4 GB, pero fue necesario aumentarla para evitar bloqueos del sistema al trabajar en más de dos núcleos de Mathematica en simultánea.

En ambos casos se trabajó con:

- xTensor 1.2.0.
- xPert 1.0.6.
- xPand 0.4.3.

Financiación

Los recursos de hardware fueron de financiación propia, mientras que las licencias de Mathematica fueron provistas por la Universidad del Valle.

C. Código

A continuación se presenta el código que ha sido utilizado en el desarrollo del proyecto. Se recomienda, sin embargo, descargarlo directamente del enlace proporcionado en la referencia [7], para disponer de la versión más actualizada posible.

```
(*Carga de Paquete.*)
Needs["xAct`xPand`"]
(* Lo anterior actúa igual que la instrucción <<xAct`xPand`*)

(*Utilidad para despejar términos.*)
Despejar[x_][expr_] :=
  Module[{aux},
   x == Solve[expr /. {x -> aux}, aux][[1]][[1]][[2]] /. {aux -> x}];

DefManifold[M, 4, {α, β, μ, ν, λ, σ}];
DefMetric[-1, g[-α, -β], CD, {";", "∇̄"}, PrintAs -> "ḡ"];
$Pre = ScreenDollarIndices;


(*Separando la parte temporal de la espacial*)
SetSlicing[g, n, γ, cd, {"|", "D̄"}, "BianchiI"];
PrintAs[Kγ] ^= "σ";
DefMetricFields[g, dg, γ];
DefMatterFields[u, du, γ];
PrintAs[Kγ] ^= "σ";


MyToxPand[expr_, gauge_, order_] :=
  ToxPand[expr, dg, u, du, γ, gauge, order];
```

C.1. Ecuaciones en una métrica arbitraria

C.1.1. Ecuaciones de campo de Einstein

```
(*Definición del Campo escalar canónico*)
DefTensor[Phi[], M, PrintAs -> "ϕ"];
DefTensorPerturbation[PhiPert[LI[order]], Phi[], M, PrintAs -> "δϕ"];

(*Definición del potencial*)
DefScalarFunction[V];

(*Definición de la constante de κ*)
DefConstantSymbol[κ, PrintAs -> "κ"];

(*Densidades lagrangianas.*)
Lmat=Sqrt[-Detg[]](-(1/2)g[μ,ν]*CD[-μ][Phi[]]*CD[-ν][Phi[]]-V[Phi[]]);
Lg = Sqrt[-Detg[]] (1/(2 κ)) RicciScalarCD[];
L = Lmat + Lg;
```

```
Lpert = ToCanonical[
  ContractMetric[ExpandPerturbation[Perturbation[L, 1]]]]

EinsEqs =
 2 κ VarD[dg[LI[1], μ, ν], CD][Lpert] 1/
      Sqrt[-Detg[]] /. {delta[-LI[1], LI[1]] -> 1} //
    RicciToEinstein // ContractMetric // ToCanonical

0 == EinsEqs

(*Con esta instrucción vemos las proyecciones temporal y espacial de
los miembros no nulos de las ecuaciones.*)
VisualizeTensor[EinsEqs, γ]
```

C.1.2. Ecuación de movimiento

```
KGEq = VarD[PhiPert[LI[1]], CD][Lpert] 1/
    Sqrt[-Detg[]] /. {delta[-LI[1], LI[1]] -> 1} // Simplify

(*La ecuación de Klein-Gordon para una métrica arbitraria queda de la
   siguiente \
manera:*)
0 == %
```

C.2. Ecuaciones de fondo en Bianchi I

C.2.1. Ecuaciones de Friedmann

```
EinsEqsBG = MyToxPand[EinsEqs, "NewtonGauge", 0]

(*Tiempo-Espacio*)
0 == ExtractComponents[EinsEqsBG, γ, {"Time", "Space"}]

(*Tiempo-Tiempo*)
0 == ExtractComponents[EinsEqsBG, γ, {"Time", "Time"}] /.
 MakeRule[{cd[-μ][Phi[]], 0}]

(*Tenemos entonces la primera ecuación de Friedmann en Bianchi I con
la siguiente instrucción:*)

Despejar[Hγ[LI[0], LI[0]]^2][%] // Expand

(*Espacio-Espacio*)
ExtractComponents[EinsEqsBG, γ, {"Space", "Space"}] /.
 MakeRule[{cd[-μ][Phi[]], 0}]

(*Traza*)
(γ[μ, ν]*%) // Expand // ContractMetric
```

```
(*Ahora aprovechamos la primera ecuación de Friedmann en Bianchi I:*)
% /. MakeRule[{
    Hγ[ LI[0], LI[0]]^2,
    1/6  Kγ[ LI[0], LI[0], -α, -β] Kγ[ LI[0], LI[0], α, β] + 1/3 κ aγ[
    LI[0], LI[0]]^2 V[ Phi[]] + 1/6 κ ( LieD[ n[α]][ Phi[]])^2}]
% // Expand // ContractMetric

Despejar[ Hγ[ LI[0], LI[1]]][0 == %] // Expand
(*Con esto obtenemos la segunda ecuación de Friedmann para Bianchi I*)

(*Parte sin traza*)
ExtractComponents[EinsEqsBG, γ, {"Space", "Space"}] /.
 MakeRule[{cd[-μ][Phi[]], 0}]

STFPart[%, γ] // Expand // ContractMetric // ToCanonical

Despejar[ Kγ[ LI[0], LI[1], -μ, -ν] ][0 == %] // Expand // ToCanonical
(*Con esto se obtiene la tercera ecuación de Friedmann*)
```

C.2.2. Ecuación de Klein-Gordon

```
(*Debemos perturbar a orden cero*)
MyToxPand[KGEq, "AnyGauge", 0]

% /. MakeRule[{cd[-\[Mu]][Phi[]], 0}]

%*(-a\[Gamma][]^2) // Expand

(*La ecuación de fondo viene dada entonces por:*)
0 == %
```

C.3. Ecuaciones de las perturbaciones del campo escalar canónico

C.3.1. Preliminares

```
(*Por simplicidad, trabajaremos con φ.*)

Lmat = Sqrt[-Detg[]](-(1/2)g[μ,ν]*CD[-μ][φ[]]*CD[-ν][φ[]]-V[φ[]]);
Lg = Sqrt[-Detg[]] (1/(2 κ)) RicciScalarCD[];
L = Lmat + Lg;

Lpert = ToCanonical[
   ContractMetric[ExpandPerturbation[Perturbation[L, 1]]]];

EinsEqs =
  2 κ VarD[dg[LI[1], μ, ν], CD][Lpert] 1/
       Sqrt[-Detg[]] /. {delta[-LI[1], LI[1]] -> 1} //
     RicciToEinstein // ContractMetric // ToCanonical;
```

```
(*Cambiamos la forma de mostrar la descomposición SVT, para que \
coincida con la convención presentada por [4].*)

PrintAs[ϕγ] ^= "A";
PrintAs[ψγ] ^= "C";


(*Las siguientes reglas son para tener en cuenta la tercera ecuación
   de Friedmann.*)

KRules = Flatten@Join[
    MakeRule[
     Evaluate[{Kγ[LI[0], LI[2], -μ, -ν], LieD[n[α]][-2 Hγ[ LI[0], LI
   [0]] Kγ[LI[0], LI[0], -μ, -ν] + 2  Kγ[ LI[0], LI[0], -μ, α] Kγ[ LI
   [0], LI[0], -ν, -α]]}]],

    MakeRule[
     Evaluate[{Kγ[LI[0], LI[1], -μ, -ν] , -2 Hγ[ LI[0], LI[0]] Kγ[ LI
   [0], LI[0], -μ, -ν] + 2 Kγ[ LI[0], LI[0], -μ, α] Kγ[LI[0], LI[0], -
   ν, -α]}]],

    MakeRule[Evaluate[{Kγ[ LI[0], LI[0], -α, λ] Kγ[ LI[0], LI[0], -β,
   -λ], 1/3  Kγ[ LI[0], LI[0], -λ, -μ] Kγ[ LI[0], LI[ 0], λ, μ] γ[-α,
   -β]}]]
    ];

KCuadrado =
  MakeRule[{Kγ[LI[0], LI[0], -α, -β] Kγ[
      LI[0], LI[0], α, β], Kγ[]^2}];


(*Calibre de Newton para el caso de Bianchi I.*)
NewtonGaugeBianchiI = {Esγ[LI[1], LI[_]] -> 0, Bsγ[LI[1], LI[_]] -> 0,
    Bvγ[LI[1], LI[_], _] -> 0};

(*Si queremos expresarlo en términos de los potenciales de Bardeen:*)
DefProjectedTensor[ΦsB[], γ,
 PrintAs -> "Φ", SpaceTimesOfDefinition -> {"Perturbed"}]
DefProjectedTensor[ΦvB[-μ], γ,
 PrintAs -> "Φ", SpaceTimesOfDefinition -> {"Perturbed"}]
DefProjectedTensor[ψB[], γ, PrintAs -> "ψ",
 SpaceTimesOfDefinition -> {"Perturbed"}]
DefTensor[χ[], M, PrintAs -> "χ"]

AutomaticRules[ΦvB, MakeRule[Evaluate[{cd[μ]@ΦvB[-μ], 0}]]];

(*Regla de transformación que lleva del Gauge de Newton a los
potenciales de Bardeen y a χ*)
(*Aquí se tiene en cuenta que xPand usa la convención de signo
negativo para la contribución escalar de la métrica espacial, por lo
cual, no debe hacerse C=-ψ, sino C=ψ.*)
```

```
BardeenRules = Flatten@Join[
    MakeRule[
     Evaluate[{ ϕγ[LI[1], LI[x_]], ΦsB[LI[0],
        LI[x]]}]],

    MakeRule[
     Evaluate[{ψγ[LI[1], LI[x_]], ψB[LI[0],
        LI[x]]}]] ,

    MakeRule[
     Evaluate[{φ[LI[1], LI[x_]], χ[LI[0], LI[x]]}]],

    MakeRule[
     Evaluate[{Evγ[LI[1], LI[2], -μ],
       LieD[n[β]][-ΦvB[LI[0], LI[0], -μ] +
         2 Kγ[-μ, -α] Evγ[LI[1],
           LI[0], α]]}]],

    MakeRule[
     Evaluate[{Evγ[LI[1],
        LI[1], -μ], -ΦvB[LI[0], LI[0], -μ] +
        2 Kγ[-μ, -α] Evγ[LI[1],
          LI[0], α]}]]
    ];


(*Esta regla es para forzar la aparición de divergencias cuando sea
   posible.*)

ForceDivRule = Flatten@Join[

    (*Para tensores de rango 1*)
    {cd[x1_]@
        cd[x2_]@cd[x3_]@cd[x4_]@expr_[a1_, a2_, x5_] /; (x5 == -x1) ->
       Evaluate[cd[x2]@cd[x3]@cd[x4]@cd[x1]@expr[a1, a2, x5]]},
    {cd[x1_]@
        cd[x2_]@cd[x3_]@cd[x4_]@expr_[a1_, a2_, x5_] /; (x5 == -x2) ->
       Evaluate[cd[x1]@cd[x3]@cd[x4]@cd[x2]@expr[a1, a2, x5]]},
    {cd[x1_]@
        cd[x2_]@cd[x3_]@cd[x4_]@expr_[a1_, a2_, x5_] /; (x5 == -x3) ->
       Evaluate[cd[x1]@cd[x2]@cd[x4]@cd[x3]@expr[a1, a2, x5]]},

    {cd[x1_]@cd[x2_]@cd[x3_]@expr_[a1_, a2_, x5_] /; (x5 == -x1) ->
      Evaluate[cd[x2]@cd[x3]@cd[x1]@expr[a1, a2, x5]]},
    {cd[x1_]@cd[x2_]@cd[x3_]@expr_[a1_, a2_, x5_] /; (x5 == -x2) ->
      Evaluate[cd[x1]@cd[x3]@cd[x2]@expr[a1, a2, x5]]},

    {cd[x1_]@cd[x2_]@expr_[a1_, a2_, x5_] /; (x5 == -x1) ->
      Evaluate[cd[x2]@cd[x1]@expr[a1, a2, x5]]},

    (*Para tensores de rango 2*)
    {cd[x1_]@
```

```
        cd[x2_]@cd[x3_]@
          cd[x4_]@expr_[a1_, a2_, x5_,
            x6_] /; (x5 == -x1) || (x6 == -x1) ->
      Evaluate[cd[x2]@cd[x3]@cd[x4]@cd[x1]@expr[a1, a2, x5, x6]]},
    {cd[x1_]@
        cd[x2_]@cd[x3_]@
          cd[x4_]@expr_[a1_, a2_, x5_,
            x6_] /; (x5 == -x2) || (x6 == -x2) ->
      Evaluate[cd[x1]@cd[x3]@cd[x4]@cd[x2]@expr[a1, a2, x5, x6]]},
    {cd[x1_]@
        cd[x2_]@cd[x3_]@
          cd[x4_]@expr_[a1_, a2_, x5_,
            x6_] /; (x5 == -x3) || (x6 == -x3) ->
      Evaluate[cd[x1]@cd[x2]@cd[x4]@cd[x3]@expr[a1, a2, x5, x6]]},

    {cd[x1_]@
        cd[x2_]@cd[x3_]@
          expr_[a1_, a2_, x5_, x6_] /; (x5 == -x1) || (x6 == -x1) ->
      Evaluate[cd[x2]@cd[x3]@cd[x1]@expr[a1, a2, x5, x6]]},
    {cd[x1_]@
        cd[x2_]@cd[x3_]@
          expr_[a1_, a2_, x5_, x6_] /; (x5 == -x2) || (x6 == -x2) ->
      Evaluate[cd[x1]@cd[x3]@cd[x2]@expr[a1, a2, x5, x6]]},

    {cd[x1_]@
        cd[x2_]@expr_[a1_, a2_, x5_,
          x6_] /; (x5 == -x1) || (x6 == -x1) ->
      Evaluate[cd[x2]@cd[x1]@expr[a1, a2, x5, x6]]}
    ];

(*Esta regla la usamos para aprovechar las dos primeras ecuaciones de
   Friedmann de Fondo.*)
BGRules = Flatten@Join[
    {κ aγ[ LI[0], LI[0]]^2 V[ φ[ LI[0], LI[0]]] ->
    2 Hγ[LI[0], LI[0]]^2 + Hγ[LI[0], LI[1]]},

    {κ aγ[ LI[0], LI[0]]^2 V[ Scalar[ φ[ LI[0], LI[0]]]] ->
    2 Hγ[LI[0], LI[0]]^2 + Hγ[LI[0], LI[1]]},

    {κ Scalar[ aγ[ LI[0], LI[0]]]^2 V[ Scalar[ φ[ LI[0], LI[0]]]] ->
    2 Hγ[LI[0], LI[0]]^2 + Hγ[LI[0], LI[1]]},

    {κ Scalar[ aγ[ LI[0], LI[0]]]^2 V[ φ[ LI[0], LI[0]]] ->
    2 Hγ[LI[0], LI[0]]^2 + Hγ[LI[0], LI[1]]},


    MakeRule[Evaluate[{κ*φ[LI[0], LI[1]]^2 ->
    2 Hγ[LI[0], LI[0]]^2 - 2 Hγ[LI[0], LI[1]] - Kγ[-μ, -ν] Kγ[μ, ν]
   }]],

    MakeRule[Evaluate[{κ Scalar[ φ[ LI[0], LI[1]]]^2 ->
    2 Hγ[LI[0], LI[0]]^2 - 2 Hγ[LI[0], LI[1]] - Kγ[-μ, -ν] Kγ[μ, ν] }]]
```

```
    ];
```

Definiciones útiles si se desea trabajar en el espacio de Fourier

```
(*Para trabajar en el espacio de Fourier.*)
DefProjectedTensor[Ep[], γ, PrintAs -> "E+",
 SpaceTimesOfDefinition -> {"Perturbed"}]
DefProjectedTensor[Ec[], γ, PrintAs -> "E×",
 SpaceTimesOfDefinition -> {"Perturbed"}]
DefProjectedTensor[E1[], γ, PrintAs -> "E1",
 SpaceTimesOfDefinition -> {"Perturbed"}]
DefProjectedTensor[E2[], γ, PrintAs -> "E2",
 SpaceTimesOfDefinition -> {"Perturbed"}]

DefProjectedTensor[Φ1[], γ, PrintAs -> "Φ1",
 SpaceTimesOfDefinition -> {"Perturbed"}]
DefProjectedTensor[Φ2[], γ, PrintAs -> "Φ2",
 SpaceTimesOfDefinition -> {"Perturbed"}]
```

```
DefProjectedTensor[k[-α], γ, PrintAs -> "k̂",
 SpaceTimesOfDefinition -> {"Background"}]
DefProjectedTensor[e1[-α], γ, PrintAs -> "e1",
 SpaceTimesOfDefinition -> {"Background"}]
DefProjectedTensor[e2[-α], γ, PrintAs -> "e2",
 SpaceTimesOfDefinition -> {"Background"}]
DefProjectedTensor[εp[-α, -β], γ, PrintAs -> "ε+",
  SpaceTimesOfDefinition -> {"Background"}]
DefProjectedTensor[εc[-α, -β], γ, PrintAs -> "ε×",
  SpaceTimesOfDefinition -> {"Background"}]
DefProjectedTensor[Nk[], γ, PrintAs -> "|k|",
 SpaceTimesOfDefinition -> {"Background"}]

DefProjectedTensor[σpar[], γ, PrintAs -> "σ||",
 SpaceTimesOfDefinition -> {"Background"}]
DefProjectedTensor[σV1[], γ, PrintAs -> "σV1",
 SpaceTimesOfDefinition -> {"Background"}]
DefProjectedTensor[σV2[], γ, PrintAs -> "σV2",
 SpaceTimesOfDefinition -> {"Background"}]
DefProjectedTensor[σp[], γ, PrintAs -> "σ+",
 SpaceTimesOfDefinition -> {"Background"}]
DefProjectedTensor[σc[], γ, PrintAs -> "σ×",
 SpaceTimesOfDefinition -> {"Background"}]


AutomaticRules[e1, MakeRule[Evaluate[{e1[-α] e2[α], 0}]]];
AutomaticRules[e1, MakeRule[Evaluate[{e1[-α] k[α], 0}]]];
AutomaticRules[e2, MakeRule[Evaluate[{e2[-α] k[α], 0}]]];

AutomaticRules[e1,
```

```
  MakeRule[
   Evaluate[{e1[-α] e2[-β] g[α, β], 0}]]];
AutomaticRules[e1,
  MakeRule[
   Evaluate[{e1[-α] k[-β] g[α, β], 0}]]];
AutomaticRules[e2,
  MakeRule[
   Evaluate[{e2[-α] k[-β] g[α, β], 0}]]];

AutomaticRules[e1,
  MakeRule[Evaluate[{e1[-α] e1[α], 1}]]];
AutomaticRules[e2,
  MakeRule[Evaluate[{e2[-α] e2[α], 1}]]];
AutomaticRules[k, MakeRule[Evaluate[{k[-α] k[α], 1}]]];
AutomaticRules[k, MakeRule[Evaluate[{k[LI[0], LI[1], -α], 0}]]];

RuleDecomposeKγ =
 MakeRule[{Kγ[LI[0], LI[0], -α, -β],
   Evaluate[
    3/2 (k[-α] k[-β] - 1/3 γ[-α, -β]) σpar[] + σV1[] ( k[-α] e1[-β] +
        k[-β] e1[-α]) + σV2[] ( k[-α] e2[-β] +
        k[-β] e2[-α]) + σp[] εp[- α, -β] + σc[] εc[-α, -β]]}]

AutomaticRules[εp,
  MakeRule[
   Evaluate[{εp[-μ, -ν] g[μ, ν], 0}]]];
AutomaticRules[εp,
  MakeRule[
   Evaluate[{k[-α] εp[α, β], 0}]]];
AutomaticRules[εp,
  MakeRule[
   Evaluate[{εp[α, β] εp[- α, -β], 1}]]];
AutomaticRules[εp,
  MakeRule[
   Evaluate[{εp[α, β] εc[- α, -β], 0}]]];
AutomaticRules[εc,
  MakeRule[
   Evaluate[{εc[-μ, -ν] g[μ, ν], 0}]]];
AutomaticRules[εc,
  MakeRule[
   Evaluate[{εc[α, β] εc[-α,-β], 1}]]];
AutomaticRules[εc,
  MakeRule[
   Evaluate[{k[-α] εc[α, β], 0}]]];


AutomaticRules[εp,
  MakeRule[
   Evaluate[{εp[-α, -β] εp[ β, σ] εp[-σ, α], 0}]]];
AutomaticRules[εp,
  MakeRule[
   Evaluate[{εp[-α, -β] εp[ β, σ] εc[-σ, α], 0}]]];
```

```
AutomaticRules[εp,
  MakeRule[
   Evaluate[{εp[-α, -β] εc[ β, σ] εc[-σ, α], 0}]]];
AutomaticRules[εc,
  MakeRule[
   Evaluate[{εc[-α, -β] εc[ β, σ] εc[-σ, α], 0}]];


εRules = Flatten@Join[
    MakeRule[
     Evaluate[{e1[α] εp[-α, -β], 1/Sqrt[2] e1[-β]}]],
    MakeRule[
     Evaluate[{e1[λ] εp[-α, -β] g[ α, -λ], 1/Sqrt[2] e1[-β]}]],
    MakeRule[
     Evaluate[{e2[α] εp[-α, -β], -( 1/Sqrt[2]) e2[-β]}]],
    MakeRule[
     Evaluate[{e2[λ] εp[-α, -β] g[ α, -λ], -(1/Sqrt[2]) e2[-β]}]],

    MakeRule[
     Evaluate[{e1[α] εc[-α, -β], 1/Sqrt[2] e2[-β]}]],
    MakeRule[
     Evaluate[{e1[λ] εc[-α, -β] g[ α, -λ], 1/Sqrt[2] e2[-β]}]],
    MakeRule[
     Evaluate[{e2[α] εc[-α, -β], 1/Sqrt[2] e1[-β]}]],
    MakeRule[
     Evaluate[{e2[λ] εc[-α, -β] g[ α, -λ], 1/Sqrt[2] e1[-β]}]]

    MakeRule[
      Evaluate[{εc[-μ,-α]εc[α,-ν],
        1/2 e1[LI[0],LI[0],-μ] e1[LI[0],LI[0],-ν] +
        1/2  e2[ LI[0], LI[0], -μ] e2[ LI[0], LI[0], -ν]}]],
    MakeRule[
      Evaluate[{εc[-μ, -α] εp[α, -ν],
        1/2  e1[ LI[0], LI[0], -ν]   e2[ LI[0], LI[0], -μ] -
        1/2  e1[ LI[0], LI[0], -μ]   e2[ LI[0], LI[0], -ν]}]],
    MakeRule[
      Evaluate[{εp[-μ, -α] εp[α, -ν],
        1/2  e1[ LI[0], LI[0], -μ]   e1[ LI[0], LI[0], -ν] +
        1/2  e2[ LI[0], LI[0], -μ]   e2[ LI[0], LI[0], -ν]}]]

    ];

ERules = Flatten@Join[
    MakeRule[
     Evaluate[{Evγ[LI[1], LI[0], -α],
       ToCanonical@
        ContractMetric[
         E1[LI[0], LI[0]] e1[-α] +
          E2[LI[0], LI[0]] e2[-α]]}]],

    MakeRule[
     Evaluate[{Etγ[LI[1], LI[0], -α, -β],
```

```
      ToCanonical@
       ContractMetric[
        Ep[LI[0], LI[0]] εp[-α, -β] +
         Ec[LI[0], LI[0]] εc[-α, -β]]}]],
    MakeRule[
     Evaluate[{Etγ[LI[1], LI[1], -α, -β],
       ToCanonical@
        ContractMetric[
         LieD[n[μ]][
          Ep[LI[0], LI[0]] εp[-α, -β] +
           Ec[LI[0], LI[0]] εc[-α, -β]]]}]],
    MakeRule[
     Evaluate[{Etγ[LI[1], LI[2], -α, -β],
       ToCanonical@
        ContractMetric[
         LieD[n[μ]]@
          LieD[n[μ]][
           Ep[LI[0], LI[0]] εp[-α, -β] +
            Ec[LI[0], LI[0]] εc[-α, -β]]]}]]
    ];

FourierModes = Flatten@Join[ERules, RuleDecomposeK,
    MakeRule[
     Evaluate[{ΦvB[LI[0], LI[0], -α],
       ToCanonical@
        ContractMetric[Φ1[LI[0],
            LI[0]] e1[-α] + Φ2[LI[0],
            LI[0]] e2[-α]]}]],
    MakeRule[
     Evaluate[{ΦvB[LI[0], LI[1], -α],
       ToCanonical@
        ContractMetric[
         LieD[n[β]]@(Φ1[LI[0],
              LI[0]] e1[-α] + Φ2[LI[0],
              LI[0]] e2[-α])]}]],

    MakeRule[
     Evaluate[{εp[LI[0],
        LI[1], -μ, -ν], -Kγ[α, β] εp[-α, -β] (γ[-μ, -ν] -
           k[-μ] k[-ν]) - Kγ[α, β] (γ[-α, -β] -
           k[-α] k[-β]) εp[-μ, -ν] + 2 (Kγ[α, -μ] εp[-ν, -α] +
            Kγ[α, -ν] εp[-μ, - α])}]],
    MakeRule[
     Evaluate[{εc[LI[0],
        LI[1], -μ, -ν], -Kγ[α, β] εc[-α, -β] (γ[-μ, -ν] -
           k[-μ] k[-ν]) - Kγ[α, β] (γ[-α, -β] -
           k[-α] k[-β]) εc[-μ, -ν] + 2 (Kγ[α, -μ] εc[-ν, -α] +
            Kγ[α, -ν] εc[-μ, - α])}]],

    MakeRule[
     Evaluate[{εp[LI[0], LI[2], -μ, -ν],
       LieD[n[β]]@(-Kγ[α, β] εp[-α, -β] (γ[-μ, -ν] -
```

```
            k[-μ] k[-ν]) - Kγ[α, β] (γ[-α, -β] -
            k[-α] k[-β]) εp[-μ, -ν] + 2 (Kγ[α, -μ] εp[-ν, - α] +
            Kγ[α, -ν] εp[-μ, - α]))}]],

    MakeRule[
     Evaluate[{εc[LI[0], LI[2], -μ, -ν],
       LieD[n[β]]@(-Kγ[α, β] εc[-α, -β] (γ[-μ, -ν] -
            k[-μ] k[-ν]) - Kγ[α, β] (γ[-α, -β] -
            k[-α] k[-β]) εc[-μ, -ν] + 2 (Kγ[α, -μ] εc[-ν, - α] +
            Kγ[α, -ν] εc[-μ, - α]))}]],

    MakeRule[{e1[LI[0], LI[1], -μ],
      Evaluate[-Kγ[-α, -β] e1[α] e1[β] e1[-μ] -
        Kγ[-α, -β] e1[α] e2[β] e2[- μ] + 2 Kγ[-μ, -α] e1[α]]}],
    MakeRule[{e2[LI[0], LI[1], -μ],
      Evaluate[-Kγ[-α, -β] e2[α] e1[β] e1[-μ] -
        Kγ[-α, -β] e2[α] e2[β] e2[- μ] + 2 Kγ[-μ, -α] e2[α]]}]
    ];
```

C.3.2. Perturbaciones de la ecuación de Klein-Gordon

```
(*Por simplicidad usaremos φ*)

KGEq = CD[-μ]@CD[μ][φ[]] - V'[φ[]]

KGEqExp = MyToxPand[KGEq, "AnyGauge", 1]

KGpert = ExtractOrder[KGEqExp, 1] // Expand

KGpert = KGpert*aγ[LI[0], LI[0]]^2 // Expand

(*Usando el Gauge de Newton:*)
KGpert /. NewtonGaugeBianchiI

(*Con esto, la evolución de las perturbaciones del campo escalar \
vienen dadas por:*)
0 == %

(*Reemplazando por los potenciales de Bardeen:*)
% /. BardeenRules

Despejar[-2* Hγ[LI[0], LI[0]]*χ[ LI[0], LI[1]] - χ[LI[0], LI[2]] + cd
   [-μ][ cd[μ][χ[LI[0], LI[0]]]] - aγ[LI[0], LI[0]]^2*χ[ LI[0], LI
   [0]]* Derivative[2][V][ φ[LI[0], LI[0]]]][%]

%*(-1)

(*En el espacio de Fourier*)
KGpert /. BardeenRules //. cd[α_][expr_] :> I Nk[] k[α] expr;
FKGpert = %
```

```
0 == %;
% - (Nk[ LI[0], LI[0]]^2 χ[ LI[0], LI[0]] +
    2 Hγ[ LI[0], LI[0]] χ[ LI[0], LI[1]] +
    χ[ LI[0], LI[2]] +
    aγ[ LI[0], LI[0]]^2 χ[ LI[0], LI[0]] V''[ φ[ LI[0], LI[0]]]);
%*(-1) // Expand
```

C.3.3. Perturbaciones de las ecuaciones de Einstein

```
EinsEqsExp = MyToxPand[EinsEqs, "AnyGauge", 1] // Expand

EinsPert = ExtractOrder[EinsEqsExp, 1] // Expand;

EinsPert /. KRules // Expand;
% /. KRules // Expand;
% /. KRules;
EinsPert = %;

(*Si nos quedamos en el gauge de Newton*)
EinsPertNewton = EinsPert /. NewtonGaugeBianchiI
```

C.3.4. Primera ecuación escalar

Para encontrar la forma explícita de $\delta G_{00} = \kappa \delta T_{00}$, hacemos:

```
EinsTimeTime =
 ExtractComponents[EinsPertNewton, γ, {"Time", "Time"}]

(*Si se reescribe en términos de los potenciales de Bardeen:*)
EinsTimeTime /. BardeenRules // Expand

% /. KRules // Expand // ContractMetric

% /. BGRules // Expand

% /. KCuadrado // ContractMetric

SEq1 = %;

0 == %

% - (4 Hγ[ LI[0], LI[0]]^2 ΦsB[ LI[0], LI[0]] +
2 Hγ[ LI[0], LI[1]] ΦsB[ LI[0], LI[0]] +
κ φ[ LI[0], LI[1]] χ[ LI[0], LI[1]] +
6 Hγ[ LI[0], LI[0]] ψB[ LI[0], LI[1]] -
2 ( cd[-α][ cd[α][ ψB[ LI[0], LI[0]]]]) +
κ aγ[ LI[0], LI[0]]^2 χ[ LI[0], LI[0]] Derivative[1][V][ φ[ LI[0], LI
   [0]]]) // Expand
```

```
(*Ecuación final Tiempo-Tiempo expresada en términos de los \
potenciales de Bardeen*)
% (-1/2) // Expand

(*En el espacio de Fourier:*)
SEq1 //. cd[α_][expr_] :> I Nk[] k[α] expr

% /. FourierModes // Expand // ContractMetric

% /. FourierModes // Expand // ContractMetric

% /. FourierModes // Expand // ContractMetric

% /. FourierModes // Expand // ContractMetric

% /. εRules // Expand

FSEq1 = %/2 // Expand

0 == %;
% - (2 Hγ[ LI[0], LI[0]]^2 ΦsB[ LI[0], LI[0]] +
Hγ[ LI[0], LI[1]] ΦsB[ LI[0], LI[0]] +
1/2 κ φ[ LI[0], LI[1]] χ[ LI[0], LI[1]] +
Nk[ LI[0], LI[0]]^2 ψB[ LI[0], LI[0]] +
3 Hγ[ LI[0], LI[0]] ψB[ LI[0], LI[1]] +
1/2 κ aγ[ LI[0], LI[0]]^2 χ[ LI[0], LI[0]] Derivative[1][V][ φ[ LI[0],
    LI[0]]]);
%*(-1) // Expand
```

C.3.5. Primera ecuación vectorial

Para encontrar la forma explícita de $\delta G_{0n} = \kappa \delta T_{0n}$, hacemos:

```
EinsTimeSpace =
 ExtractComponents[EinsPertNewton, γ, {"Time", "Space"}]

EinsTimeSpace = EinsTimeSpace /. {
    Kγ[LI[0], LI[0],-α,-β] (cd[β][ cd[-ν][Evγ[ LI[1], LI[0], α]]]) ->
    Kγ[LI[0],LI[0],-α,-β] (cd[-ν][ cd[β][ Evγ[ LI[1], LI[0], α]]])}

(*Si queremos expresar en términos de los potenciales de Bardeen:*)
EinsTimeSpace = EinsTimeSpace /. BardeenRules // Expand

0 == %

% - (1/2 ( cd[-α][ cd[α][ ΦvB[ LI[0], LI[0], -ν]]]) -
    2 Hγ[ LI[0], LI[0]] ( cd[-ν][ ΦsB[ LI[0], LI[0]]]) +
    κ φ[ LI[0], LI[1]] ( cd[-ν][ χ[ LI[0], LI[0]]]) -
    2 ( cd[-ν][ ψB[ LI[0], LI[1]]]))
```

```
% // ToCanonical

(*Si queremos llevarlo al espacio de Fourier*)
EinsTimeSpace //. cd[α_][expr_] :> I Nk[] k[α] expr

EinsTimeSpaceInFourier = % /. FourierModes // Expand // ContractMetric

(*Si queremos extraer la proyección respecto a e1[ν]*)
EinsTimeSpaceInFourier * e1[ν] // Expand

% (2/Nk[]^2) // Expand

FPhiV1 = % + Φ1[];

0 == %%;
% + Φ1[] // Expand

(*Si queremos extraer la proyección respecto a e2[ν]*)
EinsTimeSpaceInFourier*e2[ν] // Expand;

% (2/Nk[]^2) // Expand;

FPhiV2 = % + Φ2[];

0 == %%;
% + Φ2[] // Expand
```

C.3.6. Segunda ecuación escalar

Para encontrar la forma explícita de $D^n\delta G_{0n} = \kappa D^n \delta T_{0n}$, hacemos:

```
(*Calculando cd[ν][EinsTimeSpace] encontramos la segunda ecuación \
escalar:*)
cd[ν][EinsTimeSpace]

% /. ForceDivRule

% /. KCuadrado

SEq2 = %;

0 == -(1/2) % // ToCanonical

Despejar[Hγ[ LI[0], LI[0]] ( cd[-α][ cd[α][ ΦsB[ LI[0], LI[0]]]]) -
    1/2 κ φ[ LI[0], LI[1]] ( cd[-α][ cd[α][ χ[ LI[0], LI[0]]]]) +
    cd[-α][ cd[α][ ψB[ LI[0], LI[1]]]]][%] // Expand

(*Si queremos expresarlo todo en el espacio de Fourier*)
SEq2 //. cd[α_][expr_] :> I Nk[] k[α] expr

%*(-(1/Nk[]^2)) // Expand
```

```
% /. RuleDecomposeKγ // Expand // ContractMetric

% /. FourierModes // Expand

FSEq2 = %/2 // Expand;

0 == %;
% + (Hγ[ LI[0], LI[0]] ΦsB[ LI[0], LI[0]] -
    1/2 κ φ[ LI[0], LI[1]] χ[ LI[0], LI[0]] +
    ψB[ LI[0], LI[1]])
```

C.3.7. Tercera ecuación escalar

Para encontrar la forma explícita de $\gamma^{mn}\delta G_{mn} = \kappa\gamma^{mn}\delta T_{mn}$, hacemos:

```
EinsSpaceSpace = ExtractComponents[EinsPertNewton,γ,{"Space","Space"}]

γ[μ, ν] EinsSpaceSpace // Expand // ContractMetric;

% /. KRules // Expand // ContractMetric;

% /. KRules;

% // ContractMetric

% // ToCanonical

Traza = %;

(*Si queremos expresar todo en términos de los potenciales de \
Bardeen:*)
Traza /. BardeenRules // Expand

% /. KRules

% // ContractMetric

TrazaEnBardeen = % (1/6) // Expand

% // NoScalar

% /. BGRules // Expand

SEq3 = %;

(0 == %) - (-2 Hγ[ LI[0], LI[0]]^2 ΦsB[ LI[0], LI[0]] -
    Hγ[ LI[0], LI[1]] ΦsB[ LI[0], LI[0]] -
    Hγ[ LI[0], LI[0]] ΦsB[ LI[0], LI[1]] +
    1/2 κ φ[ LI[0], LI[1]] χ[ LI[0], LI[1]] -
    2 Hγ[ LI[0], LI[0]] ψB[ LI[0], LI[1]] -
```

```
    ψB[ LI[0], LI[2]] -
    1/3 ( cd[-α][ cd[α][ ΦsB[ LI[0], LI[0]]]]) +
    1/3 ( cd[-α][ cd[α][ ψB[ LI[0], LI[0]]]]) -
    1/2 κ aγ[ LI[0], LI[0]]^2 χ[ LI[0], LI[0]] V'[ φ[ LI[0], LI[0]]])

% /. KCuadrado // ContractMetric

(*Si queremos expresarlo todo en el espacio de Fourier*)
SEq3 //. cd[α_][expr_] :> I Nk[] k[α] expr

% /. KCuadrado // ContractMetric

% /. FourierModes // Expand // ContractMetric

% /. FourierModes // Expand // ContractMetric

% /. FourierModes // Expand // ContractMetric

% // ToCanonical // NoScalar

% /. εRules

% /. εRules

FSEq3 = %*(-1);

0 == %;
    % + (-2 Hγ[ LI[0], LI[0]]^2 ΦsB[ LI[0], LI[0]] -
    Hγ[ LI[0], LI[1]] ΦsB[ LI[0], LI[0]] +
    1/3 Nk[ LI[0], LI[0]]^2 ΦsB[ LI[0], LI[0]] -
    Hγ[ LI[0], LI[0]] ΦsB[ LI[0], LI[1]] +
    1/2 κ φ[ LI[0], LI[1]] χ[ LI[0], LI[1]] -
    1/3 Nk[ LI[0], LI[0]]^2 ψB[ LI[0], LI[0]] -
    2 Hγ[ LI[0], LI[0]] ψB[ LI[0], LI[1]] -
    ψB[ LI[0], LI[2]] -
    1/2 κ aγ[ LI[0], LI[0]]^2 χ[ LI[0], LI[0]] V'[ φ[ LI[0], LI[0]]]);
    %*(-1) // Expand
```

C.3.8. Ecuación tensorial

Para encontrar la forma explícita de

$$\delta G_{mn} - \frac{1}{3}\mathrm{Tr}[\delta G_{kl}]\gamma_{mn} = \kappa\left(\delta T_{mn} - \frac{1}{3}\mathrm{Tr}[\delta T_{kl}]\gamma_{mn}\right),$$

hacemos:

```
SinTraza = STFPart[EinsSpaceSpace, γ] // Expand;

% // ContractMetric // Expand;
```

```
% /. KRules // ContractMetric // Expand;

% /. KRules;

% // ContractMetric;

% /. BardeenRules // Expand;

% /. KRules;

% /. KRules;

% // Expand;

% // ContractMetric;

% /. BardeenRules;

% // Expand // ContractMetric;

% /. KRules // Expand // ContractMetric;

% // ToCanonical // NoScalar;

% /. BGRules // Expand;
% // Simplification;

SinTraza = %;

% /. KCuadrado // ContractMetric;


(*Ecuación para la parte sin traza de las perturbaciones.*)
0 == %


(*Lo siguiente es sólo para reorganizar la ecuación.*)
% - (-( Etγ[ LI[1], LI[2], -μ, -ν]) -
    2 ( Etγ[ LI[1], LI[1], -μ, -ν]) Hγ[ LI[0], LI[0]] +
    cd[-α][ cd[α][ Etγ[ LI[1], LI[0], -μ, -ν]]] -
    1/3  γ[-μ, -ν]  ( cd[-α][ cd[α][ ΦsB[ LI[0], LI[0]]]]) +
    1/3  γ[-μ, -ν]  ( cd[-α][ cd[α][ ψB[ LI[0], LI[0]]]])
    + Hγ[ LI[0], LI[0]] ( cd[-μ][ ΦvB[ LI[0], LI[0], -ν]]) +
    1/2 ( cd[-μ][ ΦvB[ LI[0], LI[1], -ν]]) +
    Hγ[ LI[0], LI[0]] ( cd[-ν][ ΦvB[ LI[0], LI[0], -μ]]) +
    1/2 ( cd[-ν][ ΦvB[ LI[0], LI[1], -μ]]) +
    cd[-ν][ cd[-μ][ ΦsB[ LI[0], LI[0]]]] -
    cd[-ν][ cd[-μ][ ψB[ LI[0], LI[0]]]]) // Expand

(*Si queremos expresarlo todo en el espacio de Fourier*)
SinTraza /. KCuadrado //.
```

```
  cd[α_][expr_] :> I Nk[] k[α] expr;

% /. FourierModes // Expand // ContractMetric;

SinTrazaEnFourier = %;

(*Proyectando respecto a εp[μ,ν]*)
εp[μ, ν]*% // Expand;

% /. FourierModes // Expand;

% /. KCuadrado // ContractMetric;

% /. KRules // Expand // ContractMetric;

% /. KRules // Expand // ContractMetric;

% /. KCuadrado // ContractMetric;

% /. FourierModes // Expand // ContractMetric;

% /. FourierModes // Expand // ContractMetric;

% // ToCanonical;

% /. εRules // Expand;

% /. εRules // Expand;

% /. εRules // Expand;

FTEqEp = (-1)*%;

0 == %

(*Haciendo lo mismo,pero ahora respecto a \
εc[μ,ν]*)
εc[μ, ν]*SinTrazaEnFourier // Expand;

% /. FourierModes // Expand;

% /. KCuadrado // ContractMetric;

% /. KRules // Expand // ContractMetric;

% /. KRules // Expand // ContractMetric;

% /. KCuadrado // ContractMetric;

% /. FourierModes // Expand // ContractMetric;

% /. FourierModes // Expand // ContractMetric;
```

```
% // ToCanonical;

% /. εRules // Expand;

% /. εRules // Expand;

% /. εRules // Expand;

FTEqEc = (-1)*%;

0 == %
```

C.3.9. Segunda ecuación vectorial

Para encontrar la forma explícita de

$$D^m\left(\delta G_{mn}-\frac{1}{3}Tr[\delta G_{kl}]\gamma_{mn}\right)=\kappa D^m\left(\delta T_{mn}-\frac{1}{3}Tr[\delta T_{kl}]\gamma_{mn}\right)$$

hacemos:

```
cd[ν]@SinTraza // Expand // ToCanonical // NoScalar;

% /. ForceDivRule;

(*Aquí usamos la regla de conmutación pertinente para intercambiar la
   \
derivada covariante y la de Lie aplicadas a Subscript[Φ, \
μ]]*)
% /. MakeRule[
   Evaluate[{cd[μ]@ΦvB[LI[0], LI[1], -μ],
     2 Kγ[LI[0],
       LI[0], μ, -ν] cd[-μ]@ΦvB[LI[0],
        LI[0], ν]}]];

% /. KRules /. KRules // Expand // ContractMetric;

% /. BGRules // Expand;

% /. KCuadrado // ToCanonical // NoScalar;

VectorEq2 = %;

0 == %

(*Si queremos llevar todo al espacio de Fourier, hacemos lo \
siguiente:*)
VectorEq2EnFourier =
  VectorEq2 //. cd[α_][expr_] :> I Nk[] k[α] expr;
```

```
(*Proyectamos respecto a e1[μ]*)
%*e1[μ] // Expand;

% /. FourierModes // Expand // ContractMetric;

% /. FourierModes // Expand // ContractMetric;

% /. FourierModes // Expand // ContractMetric;

% /. FourierModes // Expand // ContractMetric;

% (-2/Nk[]^2) // Expand;

% // ToCanonical;

% /. εRules;

% /. εRules;

FPhiPrimeV1 = -(% - Φ1[
LI[0],
LI[1]]);

0 == %%

(*Si queremos llevar todo al espacio de Fourier, hacemos lo \
siguiente:*)
VectorEq2EnFourier =
  VectorEq2 //. cd[α_][expr_] :> I Nk[] k[α] expr;

(*Proyectamos respecto a e1[μ]*)
%*e1[μ] // Expand;

% /. FourierModes // Expand // ContractMetric;

% /. FourierModes // Expand // ContractMetric;

% /. FourierModes // Expand // ContractMetric;

% /. FourierModes // Expand // ContractMetric;

% (-2/Nk[]^2) // Expand;

% // ToCanonical;

% /. εRules;

% /. εRules;

FPhiPrimeV1 = -(% - Φ1[ LI[0], LI[1]]);

0 == %%
```

```
(*Ahora hacemos lo mismo para la proyección respecto a e2[μ].*)
VectorEq2EnFourier*e2[μ] // Expand;

% /. FourierModes // Expand // ContractMetric;

% /. FourierModes // Expand // ContractMetric;

% /. FourierModes // Expand // ContractMetric;

% /. FourierModes // Expand // ContractMetric;

% (-2/Nk[]^2) // Expand;

% // ToCanonical;

% /. εRules;

% /. εRules;

FPhiPrimeV2 = -(% - Φ2[ LI[0], LI[1]]);

0 == %%
% - (2 Hγ[ LI[0], LI[0]] Φ2[ LI[0], LI[0]] +
Φ2[ LI[0], LI[1]]);
%*(-1)
```

C.3.10. Cuarta ecuación escalar

Para encontrar la forma explícita de

$$D^m D^n \left(\delta G_{mn} - \frac{1}{3} Tr[\delta G_{kl}] \gamma_{mn} \right) = \kappa D^m D^n \left(\delta T_{mn} - \frac{1}{3} Tr[\delta T_{kl}] \gamma_{mn} \right)$$

hacemos:

```
cd[μ]@cd[ν]@SinTraza // Expand;

% // ContractMetric;

% /. ForceDivRule;

% // ToCanonical // NoScalar;

(*Aquí usamos la regla de conmutación pertinente para intercambiar la
   \
derivada covariante y la de Lie aplicadas a Subscript[Φ, \
μ]]*)
% /. MakeRule[
   Evaluate[{cd[μ]@ΦvB[LI[0], LI[1], -μ],
     2 Kγ[LI[0],
```

```
       LI[0], μ, -ν] cd[-μ]@ΦvB[LI[0],
        LI[0], ν]}]];

% /. KCuadrado;

SEq4 = %;

0 == %

(*Si queremos llevar todo al espacio de Fourier*)
SEq4 //. cd[α_][expr_] :> I Nk[] k[α] expr;

% /. FourierModes // Expand // ContractMetric;

% (- (1/Nk[]^2)) // Expand;

% /. εRules

FSEq4 = %*(-1) // Expand;

0 == %;
% + (-(2/3) Nk[ LI[0], LI[0]]^2 ΦsB[ LI[0], LI[0]] + 2/3 Nk[ LI[0], LI
   [0]]^2 ψB[ LI[0], LI[0]]) // Expand;
%*(-1)
```

Referencias

[1] P. Peter y J.-P. Uzan, *Primordial Cosmology*. Oxford University Press, 2009.

[2] G. F. Ellis, R. Maartens y M. A. MacCallum, *Relativistic Cosmology*. Cambridge University Press, 2012.

[3] S. Dodelson y F. Schmidt, *Modern Cosmology*. Academic Press, 2020.

[4] T. S. Pereira, C. Pitrou y J.-P. Uzan, «Theory of Cosmological Perturbations in an Anisotropic Universe,» *Journal of Cosmology and Astroparticle Physics*, vol. 2007, n.º 09, págs. 006-006, sep. de 2007. DOI: 10.1088/1475-7516/2007/09/006. dirección: https://iopscience.iop.org/article/10.1088/1475-7516/2007/09/006.

[5] C. Pitrou, X. Roy y O. Umeh, «xPand: An Algorithm For Perturbing Homogeneous Cosmologies,» *Classical and Quantum Gravity*, vol. 30, n.º 16, pág. 165 002, jul. de 2013. DOI: 10.1088/0264-9381/30/16/165002. dirección: https://iopscience.iop.org/article/10.1088/0264-9381/30/16/165002.

[6] J. M. Martín-García, «*xAct: Efficient Tensor Computer Algebra For the Wolfram Language*», 2004. dirección: http://www.xact.es/.

[7] S. Esquea, *xPand Implementation of Canonical Scalar Field Perturbations in Bianchi I spacetimes*, 2023. dirección: https://www.github.com/StevenEsquea/CanonicalScalarFieldPerturbationsInBianchiI.

[8] Ó. A. Palmas Velasco y J. G. Reyes Victoria, *Curso de Geometría Diferencial. Parte 2. Geometría Intrínseca de las Superficies*. Universidad Nacional Autónoma de México, 2012.

[9] Ó. A. Palmas Velasco y J. G. Reyes Victoria, *Curso de Geometría Diferencial. Parte 1. Curvas y Superficies*. Universidad Nacional Autónoma de México, 2005.

[10] J. Lafuente, «Variedades Diferenciables,» *Universidad Complutense de Madrid*, vol. 10, 2014.

[11] E. Gourgoulhon, *3+1 Formalism and Bases of Numerical Relativity*. arXiv, 2007. DOI: 10.48550/ARXIV.GR-QC/0703035. dirección: https://arxiv.org/abs/gr-qc/0703035.

[12] H. Sánchez Morgado y Ó. A. Palmas Velasco, *Geometría Riemanniana*. Universidad Nacional Autónoma de México, 2007.

[13] G. B. Arfken, H. J. Weber y F. E. Harris, *Mathematical Methods for Physicists*. Elsevier, 2013.

[14] J. M. Bardeen, «Gauge-invariant Cosmological Perturbations,» *Physical Review D*, vol. 22, n.º 8, pág. 1882, 1980.

[15] C. Pitrou, X. Roy y O. Umeh, «*xPand, Perturbations Are Not Difficult*», 2012. dirección: http://www2.iap.fr/users/pitrou/xpand.htm.

[16] D. Brizuela, J. M. Martín-García y G. A. M. Marugán, «xPert: Computer Algebra For Metric Perturbation Theory,» *General Relativity and Gravitation*, vol. 41, n.º 10, págs. 2415-2431, feb. de 2009. DOI: 10.1007/s10714-009-0773-2. dirección: https://doi.org/10.1007/s10714-009-0773-2.

[17] R. M. Wald, *General Relativity*. University of Chicago press, 2010.

[18] D. Yllanes y J. M. Martín-García, «*xCoba*», 2005. dirección: http://www.xact.es/xCoba.

[19] D. Simulation, *Interactive Physics: Physics Simulation Software for the Classroom*, 2023. dirección: https://www.design-simulation.com/IP/.

[20] I. Agullo, J. Olmedo y V. Sreenath, «xAct Implementation of the Theory of Cosmological Perturbation in Bianchi I Spacetimes,» *Mathematics*, vol. 8, n.º 2, 2020, ISSN: 2227-7390. DOI: 10.3390/math8020290. dirección: https://www.mdpi.com/2227-7390/8/2/290.

[21] G. Montani, M. V. Battisti, R. Benini y G. Imponente, *Primordial Cosmology*. World Scientific, 2011.

[22] S. Wolfram, *What Should We Call the Language of Mathematica?* Feb. de 2013. dirección: https://writings.stephenwolfram.com/2013/02/what-should-we-call-the-language-of-mathematica/.

[23] *Core Language*, 2008. dirección: https://library.wolfram.com/infocenter/Books/8499/CoreLanguage.pdf.

[24] S. Weinberg, *Cosmology*. OUP Oxford, 2008.