UNIVERSIDAD NACIONAL AUTÓNOMA
DE MÉXICO

FACULTAD DE INGENIERÍA

Un código libre de hidrodinámica relativista

T E S I S

PARA OBTENER EL TÍTULO DE :

INGENIERO EN COMPUTACIÓN

P R E S E N T A :

DANIEL OLVERA CABRERA

FACULTAD DE INGENIERIA
UNAM

DIRECTOR DE TESIS: DR. SERGIO MENDOZA RAMOS

2008

Agradecimientos

En primer lugar quisiera agradecer a toda mi familia. A mis padres cuyo apoyo siempre estuvo presente. A mis hermanos, en particular al más pequeño que siempre me solicitó esta tesis.

A Raquel por toda su ayuda, compresión y amor.

Agradezco a mi asesor Sergio Mendoza por todo su apoyo y tiempo dedicado a este trabajo, así como el contagiarme y mostrarme de lleno las bondades, ventajas y la filosofía del software libre y la investigación científica.

También agradezco a mi querida Facultad de Ingeniería, en donde encontré grandes retos y satisfacciones.

Agradezco a mi grupo de sinodales: Fis. Raymundo Hugo Rangel Gutiérrez, Ing. Orlando Zaldivar Zamorategui, M. C. Alberto Arturo Herrera Becerra y M. I. Alejandro Padrón Godínez

Por último agradezco al Instituto de Astronomía por el apoyo en la realización de mi trabajo y la dirección de Asuntos del Personal Académico por su apoyo a través de la beca del proyecto DGAPA–UNAM (IN19203-3).

Índice general

Índice de figuras

Prefacio

En la actualidad, existen diversos códigos hidrodinámicos (Stone & Norman, 1992) en el mundo para resolver una cantidad extensa de problemas físicos. En el ámbito astrofísico, la dinámica de gases o hidrodinámica es una de las ramas más importantes para describir diversos fenómenos en el universo. Existen una buena cantidad de códigos hidrodinámicos (casi todos ellos newtonianos) que utilizan diversos métodos para resolver estos problemas.

Los fenómenos astrofísicos que generan las más altas energías requieren de partículas de gas que se mueven a velocidades muy cercanas a la de la luz y que se encuentran generalmente en pozos de potencial gravitacional excesivamente fuertes. Por esta razón, en los últimos años ha sido necesario construir códigos relativistas de la dinámica de gases que puedan modelar a la naturaleza de manera coherente. Esto no era posible en el pasado debido a la poca potencia en cómputo en comparación con la gran complejidad de los sistemas de ecuaciones relativistas. Los pocos códigos hidrodinámicos relativistas que se utilizan en la actualidad, en su gran mayoría no tienen licencias libres de distribución y por lo tanto no se han desarrollado tan fuertemente como deberían.

La intención de este trabajo es realizar un código hidrodinámico de relatividad especial para resolver distintos problemas astrofísicos. Debido a la gran ventaja que tienen las licencias libres en la programación actual, el código ha sido licenciado bajo la "*GNU General Public License*" (apéndice §E) `www.gnu.org`.

Notación

En el presente trabajo se utiliza:

$\rho =$ densidad de masa

$v =$ velocidad

$p =$ presión

$a =$ velocidad del sonido

$\epsilon =$ energía interna no–relativista por unidad de volumen

$\varepsilon = \epsilon/\rho$ energía interna no–relativista por unidad de masa

$s =$ entropía por unidad de masa (entropía específica)

$w =$ entalpía específica

$\gamma = \left(1 - (v/c)^2\right)^{1/2}$, factor de Lorentz

$n =$ número de partículas por unidad de volumen propio

$e =$ energía interna relativista por unidad de volumen propio

$\omega =$ entalpía relativista por unidad de volumen propio

$\sigma =$ entropía por unidad de volumen propio

$\kappa =$ índice politrópico

Se considera implícita la suma sobre índices repetidos (notación de Einstein). Los índices latinos en tensores o vectores toman valores 1,2 ó 3 en referencia a las dimensiones espaciales, $(x^1, x^2, x^3) = (x, y, z)$. Los índices griegos $\alpha, \beta, ...$ toman valores 0, 1, 2 ó 3 refiriéndose a la coordenada temporal $x^0 = ct$ y al espacio $x^\alpha = (ct, x^i)$. El tensor métrico $\eta_{\alpha\beta}$ para el espacio-tiempo de Minkowski es $\eta_{00} = 1$, $\eta_{ij} = -1$ para $i = j$ y $\eta_{ij} = 0$ cuando $i \neq j$. Una consideración muy importante en este trabajo es que el sistema de unidades que tomamos es geometrodinámico, en donde la velocidad de la luz es $c = 1$ y la constante de gravitación de Newton $G = 1$.

Capítulo 1

Hidrodinámica Relativista

En este capítulo se presentan los principios básicos de la hidrodinámica relativista, incluyendo el tema de las ondas de choque en su versión relativista. Además se presentan los fundamentos matemáticos para tratar las ecuaciones de la hidrodinámica mediante métodos numéricos.

§1.1. Introducción a la hidrodinámica relativista

La hidrodinámica es el estudio del comportamiento del los fluidos en movimiento. Cuando un gas desarrolla velocidades v del flujo cercanas a la de la luz, su estudio debe incluir los efectos relativistas del movimiento. Igualmente, en caso de que la energía interna por unidad de volumen ϵ sea comparable a la energía por unidad de volumen en reposo del gas ρc^2, deben considerarse las cantidades definidas en forma relativista. En ambos casos la herramienta que describe el flujo de tales gases es la hidrodinámica relativista.

§1.2. Tensor de energía-momento

A fin de obtener las ecuaciones de la hidrodinámica relativista, primero necesitamos definir el tensor de energía-momento $T^{\alpha\beta}$ (Landau & Lifshitz, 1987) para un fluido en movimiento en un espacio plano, el cual tiene la forma

$$T^{\alpha\beta} = \begin{pmatrix} e & 0 & 0 & 0 \\ 0 & p & 0 & 0 \\ 0 & 0 & p & 0 \\ 0 & 0 & 0 & p \end{pmatrix}. \tag{1.1}$$

en el sistema de referencia propio.

Para encontrar una expresión de $T^{\alpha\beta}$ en cualquier sistema de referencia, utilizamos el cuadrivector de velocidad

$$u^{\alpha} = \left(\gamma, \gamma\frac{\mathbf{v}}{c}\right), \tag{1.2}$$

donde $\mathbf{v}$ es la velocidad del fluido y $\gamma = (1 - (v/c)^2)^{-1/2}$ es su factor de Lorentz. En el sistema de referencia propio, $u^0 = 1$ y $u^i = 0$. Con este vector se puede escribir el tensor de energía–momento en el sistema propio como

$$T^{\alpha\beta} = \omega u^{\alpha} u^{\beta} - p g^{\alpha\beta}, \tag{1.3}$$

donde $\omega = e + p$ es la entalpía por unidad de volumen medida en el sistema de referencia propio. Dado que una igualdad tensorial en relatividad es válida para cualquier sistema de referencia, la ecuación (1.3) resulta ser la forma general del tensor de energía–momento para un fluido ideal.

En la mecánica de fluidos relativista, en un espacio-tiempo plano, las ecuaciones de campo para un sistema cerrado están determinadas por la divergencia nula del tensor de energía-momento (Landau & Lifshitz, 1987):

$$\frac{\partial T^{\beta}_{\alpha}}{\partial x^{\beta}} = \frac{1}{c}\frac{\partial T^{0}_{\alpha}}{\partial t} + \frac{\partial T^{i}_{\alpha}}{\partial x^{i}} = 0. \tag{1.4}$$

La cual contiene las leyes de conservación de la energía y momento, como veremos en la sección §1.9

§1.3. Ecuación de continuidad

Además de la expresión (1.4), es necesaria la ecuación de continuidad relativista de masa o conservación del número de partículas. Denotamos con n el número de partículas por unidad de volumen propio. En el sistema de laboratorio, cada partícula se mueve con una velocidad $\mathbf{v}$ y la densidad de partículas medida en dicho sistema es γn. El número de partículas contenido en un volumen Ω fijo en el sistema de referencia del laboratorio es

$$\int_{\Omega} \gamma n \mathrm{d}V.$$

El número total de partículas que sale del volumen Ω por unidad de tiempo es

$$\oint_{\partial\Omega} \gamma n \mathbf{v} \cdot \mathbf{da},$$

donde $\partial\Omega$ es la superficie frontera de Ω. Por otro lado, la tasa de decremento de partículas en dicho volumen es

$$-\frac{\partial}{\partial t} \int_{\Omega} \gamma n \, dV.$$

Si no existen fuentes o sumideros de partículas en este volumen, las dos cantidades anteriores se igualan en la forma integral de la ecuación de conservación del número de partículas

$$\frac{1}{c}\frac{\partial}{\partial t} \int_{\Omega} (\gamma n) \, dV = -\frac{1}{c} \oint_{\partial\Omega} \gamma n \mathbf{v} \cdot \mathbf{da} = -\frac{\mathbf{1}}{\mathbf{c}} \int_{\Omega} \nabla \cdot [\gamma \mathbf{nv}] \, \mathbf{dV}. \tag{1.5}$$

Como esta última igualdad es válida para cualquier volumen fijo Ω, los integrandos son iguales de modo que la ecuación anterior es

$$\frac{1}{c}\frac{\partial}{\partial t} (\gamma n) = -\frac{1}{c} \nabla \cdot [\gamma \mathbf{n} \boldsymbol{v}] \, . \tag{1.6}$$

En términos de cuadrivectores la ecuación anterior es la ecuación de conservación del número de partículas o ecuación de continuidad (Landau & Lifshitz, 1987):

$$\frac{\partial n u^{\alpha}}{\partial x^{\alpha}} \, = \, 0. \tag{1.7}$$

§1.4. Ecuación de conservación de entropía

Al tomar la derivada del tensor de energía–momento

$$\frac{\partial T^{\beta}_{\alpha}}{\partial x^{\beta}} = u_{\alpha}\frac{\partial\left(\omega u^{\beta}\right)}{\partial x^{\beta}} + w u^{\beta}\frac{\partial u_{\alpha}}{\partial x^{\beta}} - \frac{\partial p}{\partial x^{\alpha}} = 0,$$

y proyectarla en dirección del cuadrivector de velocidad u^{α} (es decir, efectuamos la contracción de $u^{\alpha}\partial T^{\beta}_{\alpha}/\partial x^{\beta}$) obtenemos:

$$\frac{\partial(\omega u^{\beta})}{\partial x^{\beta}} - u^{\beta}\frac{\partial p}{\partial x^{\beta}} = 0.$$

ya que $u^{\alpha}u_{\alpha} = 1$ y por lo tanto $u^{\alpha}\partial u_{\alpha}/\partial x^{\beta} = 0$.

Utilizando la igualdad $\omega u^{\beta} = n u^{\beta}(\omega/n)$ y la ecuación de continuidad (1.7) en la relación anterior se obtiene que

$$n u^{\beta}\left\{\frac{\partial}{\partial x^{\beta}}\left(\omega/n\right) - \frac{1}{n}\frac{\partial p}{\partial x^{\beta}}\right\} = 0. \tag{1.8}$$

La primera ley de la termodinámica en su forma no relativista puede escribirse como

$$\mathrm{d}\epsilon = T\mathrm{d}\mathtt{s} - p\mathrm{d}V, \tag{1.9}$$

o bien,

$$\mathrm{d}w = T\mathrm{d}\mathtt{s} + V\mathrm{d}p, \tag{1.10}$$

donde T es la temperatura, $w = \epsilon + pV$ es la entalpía específica[†] medida en el sistema de referencia del laboratorio, ϵ es la energía interna específica, s la entropía específica y $V = 1/\rho$ el volumen específico. La ecuación (1.10) puede generalizarse de manera relativista como

[†] En hidrodinámica newtoniana las cantidades hidrodinámicas son siempre medidas por unidad de masa, en el sistema de referencia del laboratorio. En otras palabras, en el límite newtoniano hablamos de la entalpía específica w, la energía interna específica ϵ, etc. En hidrodinámica relativista, las cantidades están medidas en el sistema de referencia propio. Así pues, se habla en este límite de la entalpía por unidad de volumen ω, la energía por unidad de volumen e, la entropía por unidad de volumen σ, etc.

$$\mathrm{d}(\omega/n) = T\mathrm{d}(\sigma/n) + (1/n)\,\mathrm{d}p, \tag{1.11}$$

siendo σ y ω la entropía y entalpía por unidad de volumen en el sistema propio de referencia del fluido respectivamente. Así pues, con ayuda de la la relación (1.11), la ecuación (1.8) puede escribirse como

$$u^\beta \frac{\partial\,(\sigma/n)}{\partial x^\beta} = \frac{\mathrm{d}\,(\sigma/n)}{\mathrm{d}s} = 0. \tag{1.12}$$

En otras palabras, ya que $\sigma/n =$ const. a lo largo de la trayectoria del fluido, se dice que este mismo se mueve de manera *adiabática*[†] . A esta clase de fluidos se les denomina también ideales puesto que no existen pérdidas (o ganancias) de energía por viscosidad, conductividad térmica y/o radiación.

§1.5. Ecuación de Euler

Para obtener la ecuación de movimiento o ecuación de Euler en su forma relativista proyectamos la ecuación (1.4) en dirección ortogonal a u^α. Un tensor ortogonal a u^α es $\left\{\delta^\lambda_\alpha - u_\alpha u^\lambda\right\}$ ya que la contracción de estos dos es nula. De esta manera se llega a

$$\frac{\partial T^\beta_\alpha}{\partial x^\beta} - u_\alpha u^\lambda \frac{\partial T^\beta_\lambda}{\partial x^\beta} = 0.$$

Si expandimos esta última expresión se obtiene

$$\omega u^\beta \frac{\partial u_\alpha}{\partial x^\beta} = \frac{\partial p}{\partial x^\alpha} - u_\alpha u^\lambda \frac{\partial p}{\partial x^\lambda}, \tag{1.13}$$

la cual es la ecuación de Euler en su forma cuatro-dimensional. Las componentes espaciales de la ecuación (1.13) constituyen la ecuación de Euler en su forma relativista

$$\frac{\gamma\omega}{c^2}\left\{\frac{\partial \boldsymbol{v}}{\partial t} + \boldsymbol{v}\cdot\mathbf{grad}\,\boldsymbol{v}\right\} = -\mathbf{grad}\,p - \frac{\boldsymbol{v}}{c^2}\frac{\partial p}{\partial t}. \tag{1.14}$$

[†] Cuando σ/n es constante en todo el fluido, decimos que el flujo es *isentrópico*.

§1.6. Ecuación de estado (Bondi-Wheeler)

Un proceso politrópico en un gas es un cambio de estado en el cual el calor específico permanece constante a través de todo el proceso y cumple la siguiente relación (Tooper, 1965):

$$p = \Theta \rho_g^{\,k}. \tag{1.15}$$

A partir de la ecuación (1.15) y de la primera ley de la termodinámica (1.11) se obtiene la ecuación de estado Bondi-Wheeler (cf. Tooper, 1965). La cual, para un proceso adiabático se simplifica a:

$$d\left(\frac{e}{\rho_g}\right) = -p\, d\left(\frac{1}{\rho_g}\right), \tag{1.16}$$

donde ρ_g es la densidad del gas y por lo tanto

$$\frac{d\, e}{e\ -\ p} = \frac{d\, \rho_g}{\rho_g}. \tag{1.17}$$

Así, sustituyendo la ecuación (1.15) en (1.17) se obtiene

$$p\, \frac{d\, e}{d\, p} - \frac{e}{k}\ =\ \frac{p}{k}. \tag{1.18}$$

La solución a la ecuación homogénea de (1.18) es

$$e_{\text{hom}} =\ \xi\, p^{1/k}. \tag{1.19}$$

Y la solución de la ecuación particular de (1.18) es

$$e_{\text{part}} = \frac{p}{k\ -\ 1}. \tag{1.20}$$

Así que la solución general a la ecuación (1.18) es

$$e\ =\ \xi\, p^{1/k}\ +\ \frac{p}{k\ -\ 1}. \tag{1.21}$$

Si elegimos la constaste $\xi = c^2\,/\,\Theta^{1/k}$ tenemos que (1.21) queda:

$$e\ =\ c^2\ \left(\frac{p}{\Theta}\right)^{\frac{1}{k}}\ +\ \frac{p}{k\ -\ 1}. \tag{1.22}$$

Sustituyendo (1.15) en (1.22) queda la ecuación de estado más general para un gas politrópico relativista (Tooper, 1965):

$$e \;=\; \rho_{\rm g}\, c^2 \;+\; \frac{p}{k\;-\;1}. \tag{1.23}$$

En algunas situaciones astrofíscas cuando se tienen gases excesivamente relativistas sucede que la presión p del mismo es mucho mayor que su densidad de masa en reposo $\rho_g\, c^2$. Esto equivale a elegir $\xi = 0$ y así obtener la ecuación de estado de Bondi-Wheeler:

$$p = (\kappa - 1)\, e, \tag{1.24}$$

en donde κ es el índice politrópico. En el caso de un gas monoatómico en el que no se ha tomado en cuenta los efectos relativistas, el índice politrópico es $\kappa = 5/3$. En el caso de un gas relativista, por ejemplo un gas de fotones o un gas de electrones moviéndose a velocidades cercanas a la de la luz, $\kappa = 4/3$ (Landau & Lifshitz, 1987).

§1.7. Ondas de choque relativistas

Como en el caso no relativista, los flujos relativistas presentan discontinuidades generadas por gradientes de presión suficientemente grandes. En el espacio-tiempo de Minkowski, tal discontinuidad está representada por una hipersuperficie (McKee & Colgate, 1973), que se reduce a una línea–universo para un flujo unidimensional (ver figura 1.1).

Sobre esta línea de universo en que la presión varía de manera discontinua, cuando un elemento de fluido atraviesa la onda de choque, las cantidades hidrodinámicas cambian de modo que la entropía aumenta, en completa concordancia con la segunda ley de la termodinámica. Utilizando los subíndices 1 y 2 para las cantidades antes y después de la onda de choque respectivamente, esto significa que

$$\sigma_1 < \sigma_2. \tag{1.25}$$

Tal desigualdad determina la forma en que cambian el resto de las cantidades hidrodinámicas (Taub, 1967). En el sistema de referencia de la onda de choque, la velocidad del fluido satisface las siguientes desigualdades

$$v_1 > a_1, \qquad v_2 < a_2,$$

en donde a es la velocidad del sonido.

De igual modo, el aumento de entropía implica que la densidad de partículas y la presión aumentan. Esto es

$$n_1 < n_2, \quad p_2 > p_1,$$

lo que indica el sentido de movimiento de la onda de choque.

Los cambios en las cantidades hidrodinámicas no son arbitrarios. El flujo de momento, energía y masa deben ser conservado a través de la hipersuperficie de discontinuidad. Para escribir las condiciones de salto seguimos el método presentado por McKee & Colgate (1973). Consideramos una hipersuperficie cerrada Φ sobre la línea de universo de la discontinuidad (ver figura 1.1). En ella integraremos las ecuaciones de la hidrodinámica relativista (1.4) y (1.7) escribiéndolas de la siguiente manera:

$$\frac{\partial}{\partial\tau}\left(\beta\frac{p+\epsilon+\rho c^2}{1-\beta^2}\right)+\frac{1}{r^k}\frac{\partial}{\partial \mathrm{r}}\left[r^k\frac{p+\beta^2(\epsilon+\rho c^2)}{1-\beta^2}\right]-\frac{kp}{r}=0, \tag{1.26}$$

$$\frac{\partial}{\partial\tau}\left\{\frac{\epsilon+\beta^2 p+\rho c^2[1-\sqrt{1-\beta^2}]}{1-\beta^2}\right\}+\frac{1}{r^k}\frac{\partial}{\partial \mathrm{r}}\left\{r^k\beta\frac{p+\epsilon+\rho c^2[1-\sqrt{1-\beta^2}]}{1-\beta^2}\right\}=0, \tag{1.27}$$

$$\frac{\partial}{\partial\tau}\frac{\rho}{\sqrt{1-\beta^2}}+\frac{1}{r^2}\frac{\partial}{\partial \mathrm{r^k}}\left[r^k\beta\frac{\rho}{\sqrt{1-\beta^2}}\right]=0, \tag{1.28}$$

donde β es la velocidad y $k = 0, 1$ y 2 para geometría plana, cilíndrica y esférica respectivamente. La densidad de partículas con masa m está dada por ρ_m y $\epsilon = e - \rho c^2$ es la energía interna por unidad de volumen propio.

Las tres ecuaciones anteriores poseen la misma estructura. De hecho cada una se puede escribir como la ecuación diferencial

$$\frac{\partial}{\partial\tau}\left(r^k A\right)+\frac{\partial}{\partial \mathrm{r}}\left(r^k B\right)+C=0,$$

donde $\tau = ct$, y además A y B son funciones de r y t.

Integremos esta ecuación sobre la hipersuperficie Φ que encierra la discontinuidad (ver

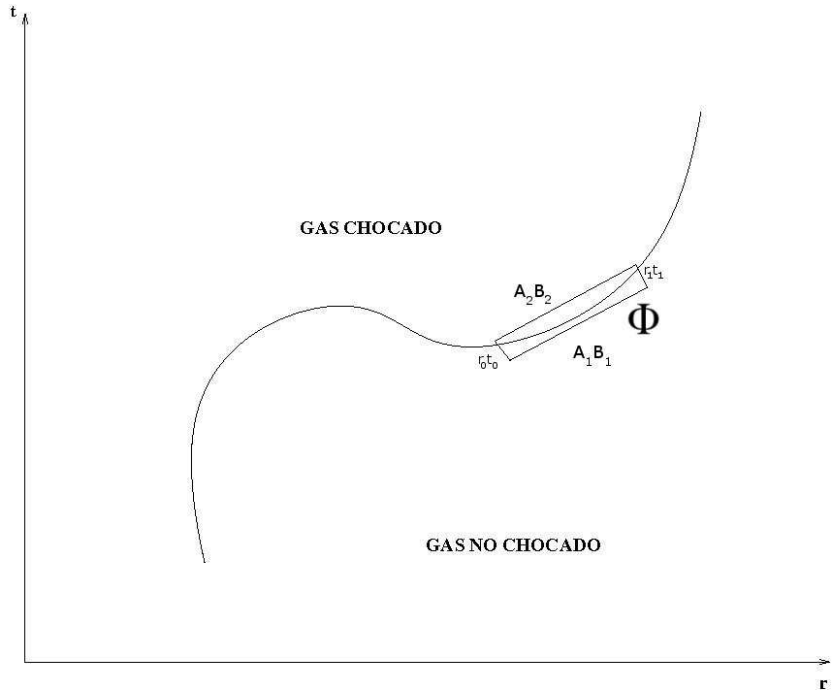

Figura 1.1: La línea continua representa la línea de universo de la onda de choque para el espacio de Minkowski. Las condiciones de salto en el choque se establecen integrando sobre el volumen Φ representado por la línea punteada. Las cantidades hidrodinámicas del fluido cambian de valores $[A_1, B_1]$ a valores $[A_2, B_2]$ cuando atraviesan la superficie de discontinuidad.

figura 1.1), para obtener

$$\int\limits_{\Phi}\int \left[\frac{\partial}{\partial\tau}\left(r^k A\right) + \frac{\partial}{\partial \mathrm{r}}\left(r^k B\right) + C\right] dr d\tau = 0. \tag{1.29}$$

Usando del teorema de Stokes, esta integral se transforma a

$$\oint\limits_{\partial\Phi}\left(r^k A\, dr - r^k B\, d\tau\right) + \int\limits_{\Phi}\int C\, d\tau dr \;=\; 0. \tag{1.30}$$

El volumen de integración Φ se ha trazado de modo que los segmentos normales a la línea–universo de la discontinuidad son pequeños comparados con los paralelos a la misma. En consecuencia, si todas las cantidades en los integrandos de la ecuación (1.30) son finitas en

la discontinuidad, la integración sobre los segmentos de menor longitud en (1.30) será despreciable al integrar sobre la hipersuperficie $\partial\Phi$. Por otra parte, la integral sobre el volumen Φ es también una cantidad de orden superior a la integral de superficie (puesto que dr y $d\tau$ son cantidades de primer orden). De esta manera, la integral sobre Φ en la ecuación (1.30) es despreciable frente a la integral sobre $\partial\Phi$. Por último, cuando el volumen Φ tiende a cero, la coordenada r en el contorno de integración puede tomarse como constante a primer orden en cada segmento. Por lo tanto, la integral (1.30) a primer orden de aproximación se reduce a

$$\oint_{\partial\Phi} (-A\,dr + B\,d\tau) \approx -\left[A_1(r_1 - r_0) - A_2(r_1 - r_0)\right] + B_1(\tau_1 - \tau_0) - B_2(\tau_1 - \tau_0) = 0, \tag{1.31}$$

donde los puntos (r_0, τ_0) y (r_1, τ_1) se encuentran sobre la línea de universo de la onda de choque. Esto implica que el cociente

$$\beta_s = \frac{r_1 - r_0}{\tau_1 - \tau_0}, \tag{1.32}$$

representa la velocidad de la onda de choque. Con esto, la ecuación (1.31) toma la forma

$$\beta_s\,\delta\,A \;=\; \delta\,B, \tag{1.33}$$

donde $\delta Q := Q_2 - Q_1$, para cualquier función $Q(r,t)$.

Sustituyendo $A(r,t)$ y $B(r,t)$ por sus valores correspondientes en cada una de las ecuaciones (1.26) a (1.28), se obtiene

$$\beta_s\,\delta\left[\beta\frac{p+\epsilon+\rho c^2}{1-\beta^2}\right] = \delta\left[\frac{p+\beta^2(\epsilon+\rho c^2)}{1-\beta^2}\right], \tag{1.34}$$

$$\beta_s\,\delta\left\{\frac{\epsilon+\beta^2 p+\rho c^2\left[1-\sqrt{1-\beta^2}\right]}{1-\beta^2}\right\} = \delta\left\{\beta\frac{p+\epsilon+\rho c^2\left[1-\sqrt{1-\beta^2}\right]}{1-\beta^2}\right\}, \tag{1.35}$$

$$\beta_s\,\delta\left[\frac{\rho}{\sqrt{1-\beta^2}}\right] = \delta\left[\beta\frac{\rho}{\sqrt{1-\beta^2}}\right]. \tag{1.36}$$

Estas son las condiciones de salto o condiciones de Taub representadas en un sistema de referencia inercial arbitrario. Las relaciones (1.34) a (1.36) son válidas sobre una discon-

tinuidad unidimensional con cualquier simetría. En el sistema de referencia propio del gas no chocado tenemos que $\beta_1 = 0$ y las relaciones anteriores toman la forma

$$\beta_s\beta_2\frac{p_2+\epsilon_2+\rho_2c^2}{1-(\beta_2)^2} = \frac{p_2+(\beta_2)^2(\epsilon_2+\rho_2c^2)}{1-(\beta_2)^2} - p_1, \tag{1.37}$$

$$\beta_s\left[\frac{\epsilon_2+\beta_2^2p+\rho_2c^2\left[1-\sqrt{1-(\beta_2)^2}\right]}{1-(\beta_2)^2} - e_1\right] = \beta_2\frac{p_2+\epsilon_2+\rho_2c^2\left[1-\sqrt{1-(\beta_2)^2}\right]}{1-(\beta_2)^2}, \tag{1.38}$$

$$\beta_s\left(\frac{\rho_2}{\sqrt{1-(\beta_2)^2}} - \rho_1\right) = \left[\beta_2\frac{\rho_2}{\sqrt{1-(\beta_2)^2}}\right]. \tag{1.39}$$

De aquí se sigue en particular que (McKee & Colgate, 1973)

$$\beta_s - \beta_2 = \frac{\beta_2\left(1-(\beta_2)^2\right)}{(\beta_2)^2p_2+\epsilon_2+\rho_2c^2\left[1-\sqrt{(1-(\beta_2)^2)}\right]-\epsilon_1\left(1-(\beta_2)^2\right)}, \tag{1.40}$$

$$\epsilon_2 = \rho_2c^2\left[\gamma_2-1\right]+\frac{\beta_s\epsilon_1+\beta_2p_1}{\beta_s-\beta_2}, \tag{1.41}$$

$$\frac{\rho_2}{\rho_1} = \frac{\beta_s}{\beta_s-\beta_2}\frac{1}{\gamma_2} = \gamma_2\left[\frac{p_2+\epsilon_2}{p_2+\epsilon_1}\right]+\left[\frac{\rho_2c^2}{p_2+\epsilon_1}\left(\gamma_2-1\right)\right]. \tag{1.42}$$

Al combinar las dos últimas igualdades se obtiene la condición de brinco para la densidad de energía

$$e_2 = \epsilon_2+\rho_2c^2 = \rho_2c^2\gamma_2+\frac{\rho_2}{\rho_1}\gamma_2\epsilon_1+\frac{\rho_2}{\rho_1}p_1\frac{\beta_2}{\beta_s}\gamma_2,$$

de modo que la energía por unidad de masa inmediatamente después del choque esta dada por

$$\frac{e_2}{\rho_2} = \frac{\gamma_2}{\rho_1}\left[e_1+p_1\frac{\beta_2}{\beta_s}\right]. \tag{1.43}$$

Para una onda de choque fuerte en donde p_1 y ϵ_1 son despreciables en comparación con p_2 y e_2 respectivamente, la igualdad (1.41) se reduce a

$$\epsilon_2 = \rho_2c^2\left(\gamma_2-1\right). \tag{1.44}$$

Con esto se puede escribir el contraste de densidades ρ_2/ρ_1 para una onda de choque fuerte

como

$$\frac{\rho_2}{\rho_1} = \frac{1}{p_2}\left[\gamma_2(p_2+\epsilon_2)+\epsilon_2\right]. \tag{1.45}$$

Cuando el gas chocado satisface una ecuación tipo Bondi–Wheeler (1.24) entonces la ecuación anterior toma la forma

$$\frac{\rho_2}{\rho_1} = \gamma_2\frac{\kappa}{\kappa-1}+\frac{1}{\kappa-1}. \tag{1.46}$$

§1.8. Ecuaciones newtonianas de la hidrodinámica

Para derivar las ecuaciones no relativistas de la hidrodinámica notemos primero la diferencia entre los valores de las cantidades termodinámicas en el caso relativista y el caso no relativista. Para comenzar, las cantidades en el caso relativista son definidas con respecto al sistema propio de referencia del fluido. En mecánica no relativista estas cantidades son referidas al sistema de referencia del laboratorio, por ejemplo, un sistema de referencia inercial en donde se observa el movimiento del fluido. En el caso relativista las cantidades termodinámicas como la densidad de energía interna e, la densidad de entropía σ, y la densidad de entalpía ω son todas definidas con respecto al volumen propio del fluido. En mecánica no relativista, estas cantidades están definidas en unidades de la masa del elemento de fluido al que se refieren. Por ejemplo, la energía interna específica ϵ, la entropía específica s y la entalpía específica w son todas medidas por unidad de masa en el sistema de referencia del laboratorio. Adicionalmente, cuando tomamos el límite no relativista, cuando la velocidad de la luz tiende a infinito, debemos recordar que la densidad de energía interna e incluye a la densidad de energía en reposo nmc^2, donde m es la masa en reposo de la partícula de fluido en consideración. En el mismo límite, el volumen debe sufrir una contracción debido a la contracción de Lorentz que se efectúa al efectuar la transformación del sistema propio de referencia propio del fluido al sistema de referencia del laboratorio. Así pues, los siguientes límites deben tomarse al pasar de mecánica de fluidos relativista a no relativista

$$
\begin{aligned}
mn &\xrightarrow[c\to\infty]{} \rho\left(1 - v^2/c^2\right)^{1/2} \approx \rho\left(1 + v^2/2c^2\right), \\
e &\xrightarrow[c\to\infty]{} nmc^2 + \rho\epsilon \approx \rho c^2 + \frac{1}{2}\rho v^2 + \rho\epsilon, \\
\frac{\omega}{n} &\xrightarrow[c\to\infty]{} mc^2 + m\left(\epsilon + \frac{p}{\rho}\right) \approx m\left(c^2 + w\right),
\end{aligned} \tag{1.47}
$$

donde ρ es la densidad de masa del fluido medida en el laboratorio del sistema.

La ecuación de continuidad (1.7), la ecuación de Euler (1.14) en su forma tres–dimensional y la ecuación de la conservación de la entropía (1.12) pueden escribirse respectivamente como

$$\frac{\partial \rho}{\partial t} + \operatorname{div}(\rho \boldsymbol{v}) = 0, \tag{1.48}$$

$$\frac{\partial \boldsymbol{v}}{\partial t} + (\boldsymbol{v} \cdot \mathbf{grad})\, \boldsymbol{v} = -\frac{1}{\rho}\mathbf{grad}\, p, \tag{1.49}$$

$$\frac{\partial s}{\partial t} + (v \cdot \mathbf{grad})\, s = 0. \tag{1.50}$$

§1.9. Ecuaciones de la hidrodinámica en forma conservativa

Un sistema de ecuaciones diferenciales parciales en una sola dimensión, está en forma conservativa cuando se expresa de la siguiente manera (cf. Toro, 1999):

$$\frac{\partial U_\alpha}{\partial t} + \frac{\partial F_\alpha}{\partial x} = 0, \tag{1.51}$$

donde $U_\alpha(x_\beta)$[†] es el vector de variables conservadas y F_α el vector de flujos, el cual es una en función de U_α. Las ecuaciones de la hidrodinámica relativista pueden expresarse de la misma forma. En efecto, si expandimos en una sola dimensión la ecuación de continuidad de la hidrodinámica relativista (1.7) en sus componentes temporal y espacial se observa que tiene la misma forma que la ecuación (1.51)

$$\frac{\partial n u^\alpha}{\partial x^\alpha} = \frac{1}{c}\frac{\partial \gamma n}{\partial t} + \frac{\partial \gamma n v}{\partial x} = 0. \tag{1.52}$$

[†] El índice α depende del número de ecuaciones conservativas y pueden ser tantas como el problema plantee. En este caso, en una dimensión, tenemos 3 ecuaciones por lo que $alpha = 1, 2$ y 3.

También, considerando una sola dimensión, podemos obtener las ecuaciones de conservación de energía y de momento de la divergencia nula del tensor de energía–momento (1.4). La ecuación de conservación de la energía es $\partial T^{0\beta}/\partial x^{\beta} = 0$; por lo que al sustituir los índices correspondientes en las ecuaciones (1.2) y (1.3) se tiene que

$$T^{00} = (e+p)\,\gamma^2 \,-\, p \qquad T^{01} = (e+p)\,\gamma^2\, v. \tag{1.53}$$

Así que al separar en sus componentes temporal y espacial la ecuación de conservación de la energía y sustituir el factor de Lorentz se tiene la misma forma que la ecuación (1.51)

$$\frac{1}{c}\frac{\partial}{\partial t}\left(\frac{e+v^2 p}{1-v^2}\right) + \frac{\partial}{\partial x}\left(\beta\frac{p+e}{1-v^2}\right) = 0. \tag{1.54}$$

De la misma manera, de la ecuación de conservación de momento $\partial T^{1\beta}/\partial x^{\beta}$, se obtiene la ecuación con la forma requerida

$$\frac{1}{c}\frac{\partial}{\partial t}\left(v\frac{p+e}{1-v^2}\right) + \frac{\partial}{\partial x}\left(v^2\frac{p+e}{1-v^2} + p\right) = 0. \tag{1.55}$$

De esta manera tenemos que el sistema que forman las ecuaciones (1.52), (1.54) y (1.55) es conservativo ya que tiene la forma de (1.51). De las derivadas temporales de cada una de las ecuaciones obtenemos el vector de variables conservadas U_α y de las derivadas espaciales el vector de flujos F_α.

De acuerdo a las propiedades matemáticas de dicho sistema, al aplicar el Jacobiano al vector de flujos ($A = \partial F_\alpha/\partial U_\alpha$) y utilizando la regla de la cadena, el sistema de ecuaciones se puede reescribir como

$$\frac{\partial U_\alpha}{\partial t} + M\frac{\partial U_\alpha}{\partial x} = 0, \tag{1.56}$$

llamada la forma cuasi–lineal (Toro, 1999) de las ecuaciones conservativas.

Una importante propiedad del sistema de ecuaciones (1.56), es que es un sistema hiperbólico. Un sistema hiperbólico es aquel en el que el Jacobiano $\partial F(u)/\partial u$ tiene eigenvalores reales y un conjunto completo de eigenvectores. Información sobre la solución está dada por los eigenvalores, pero en general no es posible o es muy complejo derivar una solución exacta de este problema. Es por ello que se utilizan métodos numéricos para aproximar la solución.

En el caso de generalizar las ecuaciones para contemplar las demás dimensiones espaciales,el sistema de ecuaciones toma la forma

$$\frac{\partial U_\alpha}{\partial t} + \frac{\partial F_\alpha}{\partial x} + \frac{\partial F_\alpha}{\partial y} + \frac{\partial F_\alpha}{\partial z} = 0, \tag{1.57}$$

al cual también se le puede aplicar el mismo procedimiento matemático para reescribirla en su forma cuasi–lineal.

Capítulo 2

Métodos numéricos

En este capítulo veremos algunos de los fundamentos de los métodos numéricos para resolver una ecuación diferencial utilizando la discretización, con especial atención a las ecuaciones hiperbólicas. También se detallan algunos métodos básicos para resolver ecuaciones en derivadas parciales.

§2.1. Introducción

Para resolver ecuaciones en derivadas parciales (EDPs), los métodos numéricos reemplazan el problema *contínuo* por un conjunto finito de valores discretos. Estos valores son obtenidos discretizando el dominio de la EDP dentro de una malla. En general existen 2 maneras de discretizar el dominio de una EDP: (a) el método de diferencias finitas o (b) el método de volúmenes finitos.

El método de diferencias finitas hace aproximaciones puntuales de los valores de las derivadas utilizando el teorema de Taylor. Por ejemplo al truncar una serie de Taylor hasta primer orden

$$f(x_0 + \Delta x) = f(x_0) + \frac{(\Delta x)}{1!}\frac{d\,f(x)}{dx} + ... \tag{2.1}$$

y después resolver para la derivada df/dx, obtenemos una aproximación de primer orden

para la primera derivada de la función f(x) (Chung, 2002)

$$\frac{d\,f(x)}{dx} = \frac{f(x_0 + \Delta x) - f(x_0)}{\Delta x} + \mathrm{O}(\Delta x), \tag{2.2}$$

la cual se le conoce como diferencia *hacia adelante*. De igual manera se puede resolver la ecuación (2.1) para obtener las aproximaciones de la primera derivada *hacia atrás* y *centrada*, respectivamente:

$$\frac{d\,f(x)}{dx} = \frac{f(x_0) - f(x_0 - \Delta x)}{\Delta x} + \mathrm{O}(\Delta x), \tag{2.3}$$

$$\frac{d\,f(x)}{dx} = \frac{f(x_0 + \Delta x) - f(x_0 - \Delta x)}{2\,\Delta x} + \mathrm{O}(\Delta x). \tag{2.4}$$

A partir de estas aproximaciones básicas y otras variaciones, además de aproximaciones de orden mayor, los métodos de diferencias finitas, como veremos en los siguientes capítulos, integran una solución.

Por otra parte, en los métodos de volúmenes finitos, en lugar de hacer aproximaciones puntuales, el espacio es dividido en celdas, por lo que el dominio se convierte en una malla. Cada celda corresponde a un promedio el cual se calcula de la siguiente manera

$$\bar{u_i} = \frac{1}{\Delta x}\int_{x_i - 1/2}^{x_i + 1/2} u(x)\mathrm{d}x, \tag{2.5}$$

en donde $\bar{u_i}$ es una celda del dominio y $x_i - 1/2$ y $x_i + 1/2$ indican el borde de la celda.

Estos métodos pueden tener mayor precisión que los de diferencias finitas, pero computacionalmente son más costosos ya que requieren más recursos de memoria de cálculo. Además, debido a que son más complejos, su adaptación a problemas específicos necesita mayores modificaciones.

§2.2. Ecuaciones hiperbólicas

Como se mencionó en la sección §1.9, las ecuaciones de la hidrodinámica en relatividad especial pueden ser escritas como un sistema hiperbólico de leyes conservativas (Martí & Müller, 2003) y luego ser reescritas como el sistema de ecuaciones (1.56).

Una EDP de segundo orden y de dos variables independientes (x, y) en su forma más

general se expresa de la siguiente manera

$$a\frac{\partial^2 q}{\partial x^2} + b\frac{\partial^2 q}{\partial x\,\partial y} + c\frac{\partial^2 q}{\partial y^2} + d\frac{\partial q}{\partial x} + e\frac{\partial q}{\partial y} + f\ q = g.$$

Su clasificación depende del valor del discriminante $\Delta = b^2 - 4ac$ de la siguiente forma. Si $\Delta < 0$ entonces es elíptica, si $\Delta = 0$ entonces es parabólica y cuando $\Delta > 0$ es hiperbólica.

Desde un punto de vista más general, el álgebra lineal nos dice que un sistema hiperbólico es aquel que tiene un conjunto de eigenvalores reales y su correspondiente conjunto independiente de eigenvectores (Toro, 1999).

Otra propiedad importante de los sistemas hiperbólicos es la diagonalización. La matriz M de la expresión (1.56), puede ser diagonalizada y expresada en términos de la matriz diagonal de eigenvalores Λ de la siguiente manera:

$$M = K\Lambda K^{-1} \qquad \Lambda = K^{-1}MK, \tag{2.6}$$

donde K y K^{-1} son el eigenvector derecho y su respectivo eigenvector inverso. La existencia de la matriz inversa hace posible definir un nuevo conjunto de variables, llamadas variables características W en donde

$$W = K^{-1}U \qquad U = KW. \tag{2.7}$$

Así, la ecuación (1.56) con ayuda de (2.6) y (2.7) se puede expresar como:

$$\frac{\partial W}{\partial t} + \Lambda\frac{\partial W}{\partial x} = 0. \tag{2.8}$$

Conocida como la forma *canónica* o forma *característica* del sistema. Del cual se obtienen las curvas características, las cuales tienen la propiedad de que a lo largo de ellas la ecuación diferencial parcial se convierte en una ecuación diferencial ordinaria.

§2.3. Método de Lax-Friedrichs

Uno de los métodos de primer orden más conocidos es el de Lax-Friedrichs. Dicho método es una modificación del método básico de Euler FTCS (Forward Time Central

Space)

$$u(x,t+\Delta t) = u(x,t) - \frac{\Delta t}{2\,\Delta x}\{u(x-\Delta x,t) - u(x+\Delta x,t)\}$$

cuyo problema principal es su inestabilidad; la cual se corrige fácilmente reemplazando el término $u(x,t)$ en la derivada del tiempo por su promedio $(u(x-\Delta x,t)+u(x+\Delta x,t))/2$. (Press & Teukolsy, 1992):

$$u(x,t+\Delta t) = \frac{1}{2}\{u(x-\Delta x,t) + u(x+\Delta x,t)\} - \frac{\Delta t}{2\,\Delta x}\{u(x-\Delta x,t) - u(x+\Delta x,t)\}.$$

Este es un método sólido y estable, pero es disipativo (difusivo). Esto quiere decir que la solución tiende a perder amplitud (la Figura 2.1 muestra dicho efecto), por lo tanto, en presencia de discontinuidades no es el método más adecuado ya que suaviza mucho la solución.

Para ejemplificar esta situación, en la Figura 2.2 se observa la condición inicial del problema del tubo de choque (problema de Riemann). Dicho problema consiste en la suposición de un tubo en una dimensión el cual contiene un fluido en reposo y está separado en dos regiones por un diafragma. Cada región tiene una presión diferente. El diafragma es removido instantáneamente y, debido a la diferencia de presiones, se generan una onda de choque, una discontinuidad de contacto y una onda de rarefacción (Landau & Lifshitz, 1987). La Figura 2.3 muestra el problema del tubo de choque un instante después de haber removido el diafragma. Debido a que el método es disipativo, la onda de choque se suaviza demasiado, por lo cual el método de Lax-Friedrich no es adecuado para simular una discontinuidad.

§2.4. Método predictor-corrector (MacCormark)

El método de segundo orden de MacCormack consiste en 2 pasos, el predictor y el corrector(MacCormack & Paullay, 1972):

$$\bar{u}(x,t+\Delta t) = u(x,t) - \frac{\Delta t}{\Delta x}[u(x,t) - u(x-\Delta x,t)] \tag{2.9}$$

$$u(x,t+\Delta t) = \frac{1}{2}\{u(x,t) + \bar{u}(x,t) - \frac{\Delta t}{\Delta x}[\bar{u}(x+\Delta x,t+\Delta t) - \bar{u}(x,t+\Delta t)]\}. \tag{2.10}$$

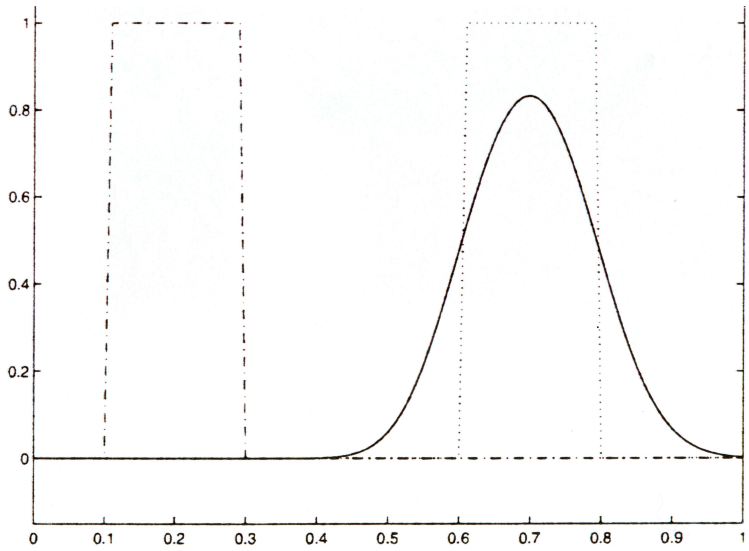

Figura 2.1: La gráfica muestra el comportamiento de una onda cuadrada que pierde amplitud y exactitud debido a los efectos de la disipación numérica.

Este método es mejor que el de Lax-Friedrichs, ya que es de segundo orden y obtiene resultado más exactos, pero en presencia de discontinuidades tiende a ser dispersivo (la Figura 2.4 muestra dicho fenómeno), es decir produce oscilaciones espurias (fenómeno de Gibbs [†]).

§2.5. Viscosidad artificial

En general en una expresión numérica en donde se aproxima hasta la segunda derivada, es decir de segundo orden, a la segunda diferencia se le conoce como viscosidad artificial. Esto es debido a la asociación que existe con el término de viscosidad en las ecuaciones de Navier-Stokes. Se dice que es artificial porque en realidad no esta relacionada con la viscosidad física (Laney, 1998). Diversos métodos numéricos son mejorados al agregar modificaciones a la viscosidad artificial.

El método de MacCormack presenta oscilaciones no físicas como se puede ver en la

[†] Las oscilaciones de Gibbs fueron descubiertas en el contexto de las reconstrucciones polinomiales con series de Fourier, y consiste en un región donde el margen de error aumenta y se manifiesta en marcadas oscilaciones

Figura 2.5, en donde se grafica la ecuación de Burgers de un fluido sin viscosidad. Dicha ecuación es la variación no lineal más simple de la ecuación de advección lineal[†] . Esta ecuación es un buen ejemplo para verificar el desempeño del método ya que presenta discontinuidades.

Para evitar las oscilaciones espurias se recurre a un método de corrección de flujo. El cual consiste en aplicar una difusión correctiva en las regiones donde se localizan oscilaciones no físicas (Book et al., 1975)

La difusión correctiva que se aplica es la siguiente:

$$u(x,t)_D = u(x,t) + \eta\,[u(x,t+\Delta t)\ +\ u(x,t-\Delta t) -\ 2\,u(x,t)], \tag{2.11}$$

en donde η es el coeficiente de difusión/antidifusión.

$$\eta = \begin{cases} \eta_0 \leq 1/4, & \text{si } (\ u(x,t+\Delta t) - u(x,t)\ )\ (\ u(x,t) - u(x,t-\Delta t)\ )\ <\ 0. \\ 0, & \text{si } (\ u(x,t+\Delta t) - u(x,t)\ )\ (\ u(x,t) - u(x,t-\Delta t)\ )\ >\ 0. \end{cases} \tag{2.12}$$

Verificando esta sencilla condición se puede detectar si existe un pico en el fluido, es decir un punto que salga abruptamente dentro de otros dos que tiene un comportamiento similar. De no encontrarse tal comportamiento al coeficiente simplemente se le asigna el valor cero y deja el resultado tal como estaba. En la Figura 2.6 podemos ver la aplicación de la corrección.

El código Aztekas (`www.aztekas.org`) , en su primera versión, utiliza el método predictor-corrector de MacCormack, además se le agrega una difusión correctiva para minimizar el fenómeno de Gibbs. Su estructura y funcionamiento se detalla en el siguiente capítulo.

[†] La ecuación de advección $\partial u/\partial t + a\,\partial u/\partial x = 0$ en donde $a = \text{const}$ es la ecuación hiperbólica más básica, llamada ecuación de transporte, se utiliza para modelar el movimiento de un fluido

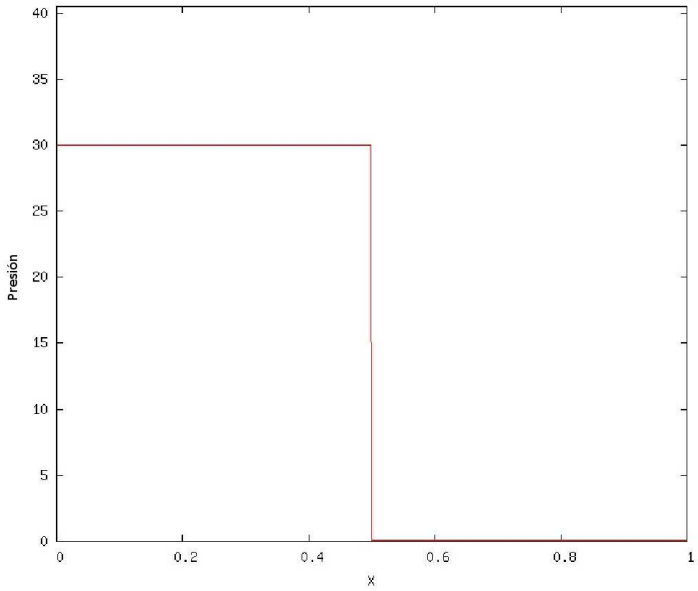

Figura 2.2: La gráfica muestra la presión p de la condición inicial del experimento de tubo de choque. Dicho problema consiste en un tubo separado en dos regiones por un diafragma. El tubo contiene un gas en reposo y cada región tiene diferentes presiones. La condición que se muestra es una perfecta discontinuidad en el valor de la presión.

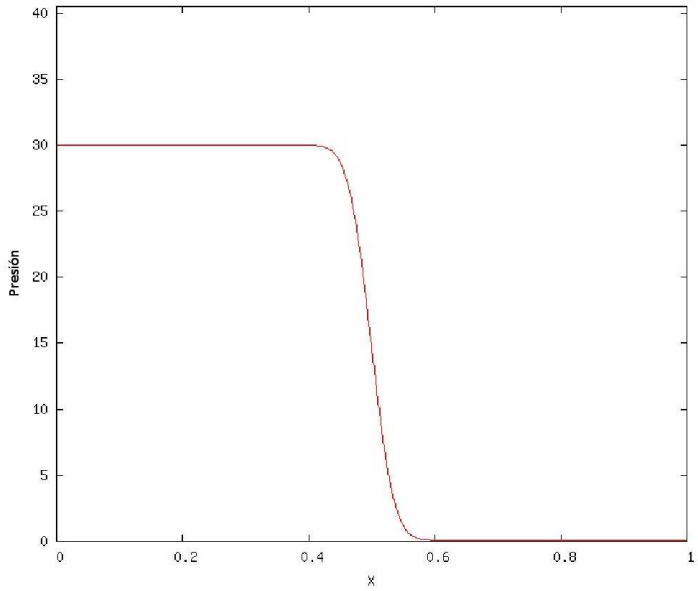

Figura 2.3: La gráfica muestra la presión p como función de la posición x, un instante después de remover el diafragma que separaba en perfecta discontinuidad al tubo. Lo que inmediatamente se forma es una onda de choque, pero debido a la disipación numérica del método de Lax, esta onda se observa suavizada lo cual no representa adecuadamente a la onda de choque.

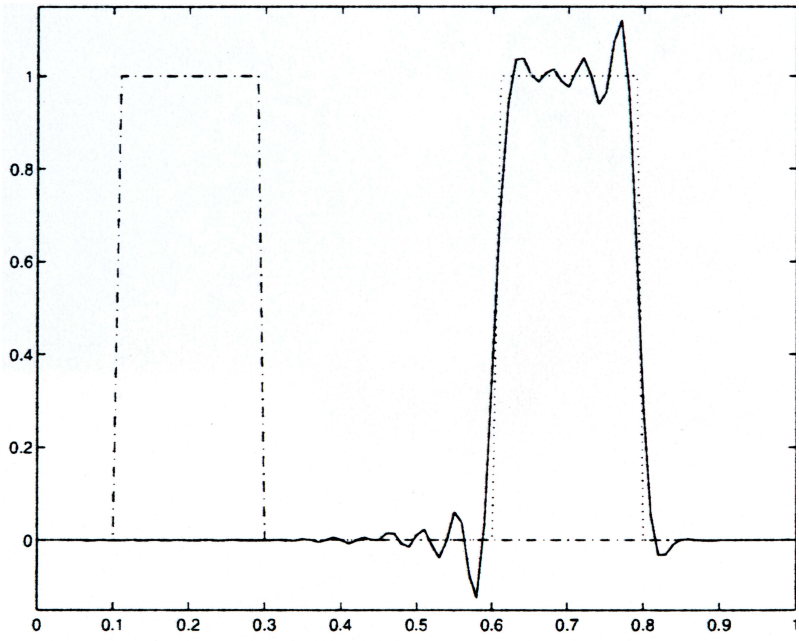

Figura 2.4: La gráfica muestra el comportamiento de una onda cuadrada que presenta oscilaciones espurias debido a los efectos de la dispersión numérica.

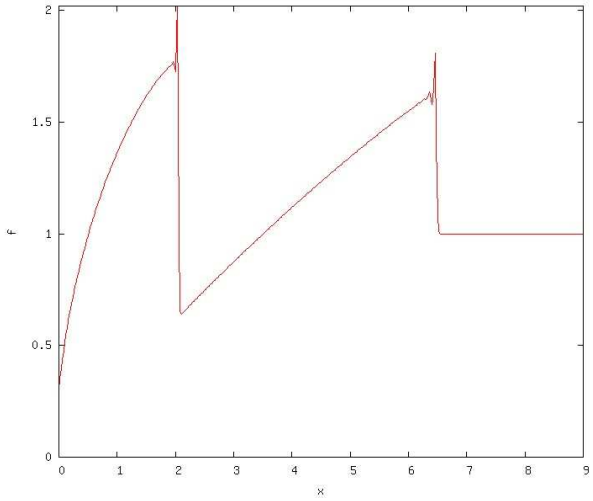

Figura 2.5: La gráfica muestra el comportamiento de la ecuación de Burgers $\partial u/\partial t + u\partial u/\partial x = 0$ utilizando exclusivamente el método de MacCormack, el cual presenta el problema de oscilaciones no físicas al encontrar una fuerte discontinuidad. Dichas oscilaciones son amplificaciones del error numérico del método al no poder manejar consistentemente una discontinuidad

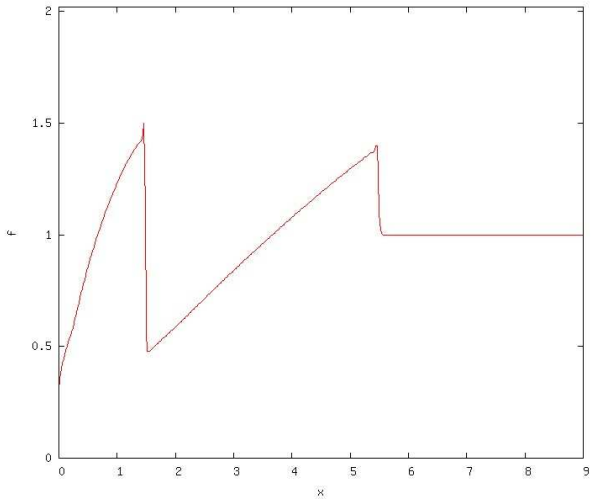

Figura 2.6: La gráfica muestra la corrección de las oscilaciones no físicas de la ecuación de Burgers $\partial u/\partial t + u\partial u/\partial x = 0$ utilizando un método de corrección de flujo, el cual sirve para mejorar el método de MacCormack.

Capítulo 3

Código Aztekas[†]

En este capítulo se mostrará el diseño y las bases sobre las cuales se fundamenta el código aztekas, los módulos que lo componen y algunas pruebas numéricas estándar para validar su funcionamiento.

§3.1. Estructura del código

En general, en la ingeniería de software los pasos básicos para crear un programa son los siguientes: Requerimientos, especificaciones, diseño y mantenimiento.

§3.1.1. Requerimientos

El requerimiento básico es un programa que implemente un método numérico que resuelva las ecuaciones de la hidrodinámica relativista para simular el desarrollo de diversos fenómenos astrofísicos con particular interés en las ondas de choque. Además dicho programa que apoyarse en herramientas de software libre.

§3.1.2. Especificaciones

Como especificaciones se acordó que el código utilizará el método numérico de MacCormack complementado con un método de corrección de flujo para lograr un adecuado manejo de las discontinuidades que se presentan en las ondas de choque. El sistema de

[†] Originalmente se pensó en el nombre de código Azteca, pero debido a que el dominio con ese nombre ya no estaba disponible, se eligió código Aztekas para tener disponible el dominio `www.aztekas.org`

ecuaciones a resolver deberá ser planteado como un sistema de ecuaciones diferenciales parciales en forma conservativa ya que sus propiedades matemáticas permiten generalizar el método de MacCormack para resolver simultáneamente todo un sistema de ecuaciones.

Se plantearán los problemas en una dimensión. El sistema de ecuaciones será de 3x3, siendo las incógnitas la velocidad v, la presión v y el número de partículas n.

La manera en que el programa trabajará será mediante archivos de texto plano. Los archivos de entrada contendrán las ecuaciones que se resolverán y sus condiciones iniciales además de los parámetros de control del programa. Los archivos de salida mostrarán el estado de las tres variables de interés en un determinado tiempo y de manera secuencial para ser graficados y observados de manera consecutiva.

§3.1.3. Diseño

Debido a las características y uso que tendrá el código numérico, el diseño debe enfocarse principalmente al rendimiento y mantenibilidad que ya que el programa no es un sistema crítico, sólo es de cálculo.

La descomposición modular estará orientada a funciones y con un control centralizado mediante subrutinas descendentes ya que el sistema es secuencial.

El código constará de un programa principal y dos de funciones. La figura 3.3 muestra el diagrama de flujo programa principal.

§3.1.4. Pruebas y mantenimiento

La pruebas para validar el funcionamiento del código serán mediante la simulación del tubo de choque. Esta es la prueba estándar para este tipo de código numéricos ya que existe una solución analítica para este problema con la cual se puede comparar el resultado de la simulación numérica.

Se plantea que el programa se modifique ampliamente para agregar más características, principalmente manejar 2 y 3 dimensiones además de contemplar efectos gravitatorios y magnéticos.

§3.2. Lenguaje C y librería GSL

El código esta escrito principalmente en lenguaje C estándar (ANSI C). Debido a su gran portabilidad y amplio uso. Para compilarlo se utiliza el programa GCC (GNU

Compiler Collection `http://gcc.gnu.org`); el cual, además de ser el compilador oficial del proyecto GNU, ha sido adoptado como estándar en la mayoría de los sistemas operativos tipo UNIX y adaptado también a otras arquitecturas.

Otro motivo importante por el cual se ha programado en C fue la posibilidad de utilizar la librería GSL (GNU Scientific Library `www.gnu.org/software/gsl/`). Esta es una librería científica totalmente libre y optimizada para diversas aplicaciones numéricas. En nuestro caso, utilizamos la librería GSL para manejar datos en matrices.

§3.3. Maxima

Para realizar las operaciones simbólicas previas a los cálculos numéricos, utilizamos el software Maxima (`http://maxima.sourceforge.net`). Este programa fue elegido por ser libre y bajo licencia GPL, ya que otros programas muy conocidos en su categoría como Mathematica o Maple son de licencia propietaria.

Maxima es un programa del tipo CAS (Computer algebra system). Es una herramienta que facilita el cálculo simbólico, es decir, una expresión como $a + b$ es interpretada como la suma de dos variables y no como la suma de dos números.

Como se mencionó en la sección §1.9 las ecuaciones de la hidrodinámica relativista en forma conservativa se puede expresar como el sistema (1.51) y mediante simple manipulación algebraica y de matrices se reescribe de la manera (1.56). Dicha manipulación requiere un sistema algebraico como Maxima (Véase apéndice A).

Para iniciar a trabajar con el código, escribimos las ecuaciones en un simple archivo de texto para que el programa lo lea. Primero escribimos la parte temporal del sistema de ecuaciones (1.51)

$$\frac{\partial}{\partial t}\left(\frac{n}{\sqrt{1-v^2}}\right), \quad \frac{\partial}{\partial t}\left(\frac{e+v^2p}{1-v^2}\right), \quad \frac{\partial}{\partial t}\left(v\frac{p+e}{1-v^2}\right),$$

en el formato que requiere Maxima, con lo cual el archivo llamado VECTOR1 queda de la siguiente manera:

```
/*VECTOR1*/
/*Derivadas temporales*/
a1: (n)/(1-v^2)^(1/2);
a2: (e+((v^2)*(p)))/(1-v^2);
```

```
a3: (v*(p+e))/(1-v^2);
```

De igual forma la parte espacial del sistema (1.51),

$$\frac{\partial}{\partial x}\left(\frac{n\,v}{\sqrt{1-v^2}}\right), \quad \frac{\partial}{\partial x}\left(v\frac{p+e}{1-v^2}\right), \quad \frac{\partial}{\partial x}\left(v^2\frac{p+e}{1-v^2}+p\right),$$

la escribimos en el archivo VECTOR2

```
/*VECTOR2*/
/*Derivadas espaciales*/
b1: ((n)*v)/(1-v^2)^(1/2);
b2: (v*(p+e))/(1-v^2);
b3: ((v^2)*(p+e))/(1-v^2) + p;
```

Para que el sistema de ecuaciones se convierta en un sistema compatible determinado, es decir, que tenga una solución única, se debe tener una ecuación más, en este caso se proporciona la ecuación de estado (1.24) en archivo de texto llamado EOS (Equation Of State).

Una vez que Maxima lee los archivos de texto VECTOR1, VECTOR2 Y EOS, calcula todas las derivadas y realiza las operaciones matriciales para obtener la matriz M que se indica en la ecuación (1.56)(cf. apéndice A).

§3.4. Manipulación de datos con Perl

Perl es un un lenguaje de alto nivel y de propósito general, tiene la gran ventaja de ser software libre. Originalmente diseñado para manipular datos y archivos, en la actualidad se utiliza en una infinidad de tareas. Uno de sus grandes usos es la aplicación de expresiones regulares.

Antes de poder usar la matriz calculada en Maxima, se debe hacer un tratamiento de los datos, debido al formato que necesita el lenguaje C y el que arroja Maxima. Es por ello que se utiliza Perl y las expresiones regulares.

Por ejemplo, Maxima eleva al cuadrado una variable como $a\hat{}2$ mientras que en lenguaje C se expresa $pow(a, 2)$. A continuación se muestra la facilidad de hacer este cambio usando Perl:

```
#!/bin/perl
```

```
# subrutina de conversion de potencias a lenguaje C.
sub conversion
{
  $a=$_[0];
  $a=~s/([A-z])\^(\d)/pow($1,$2)/g;
  return $a;
}
  $ejemplo ="v^2 + p^3";
  print "formato de Maxima: $ejemplo \n";
  $ejemplo= conversion $ejemplo;
  print "formato para C   : $ejemplo \n";
```

Lo cual nos da como resultado:

```
formato de Maxima: v^2 + p^3
formato para C   : pow(v,2) + pow(p,3)
```

Con este claro ejemplo se muestra la forma en que Perl aplica una expresión regular para modificar rápidamente una línea de texto.

El programa escrito para realizar la conversión entre el formato de Maxima y el formato del lenguaje C es **Functions.pl**. (véase apéndice B). Este programa además de ajustar la matriz M de la ecuación (1.56) obtenida de los cálculos símbolicos de Maxima, crea otro archivo de funciones para compilarse con el programa principal. Dicho archivo es el func_matrix.c.

El archivo func_matrix.c se compone de 9 funciones. Cada una representa a los elementos de la matriz M calculada por Maxima. Así por ejemplo si el elemento 1,1 de la matriz es $v\hat{}2$, la función tendrá la siguiente forma:

```
double funct_A11(double n, double v, double p, double k)
{
double a11;
a11 = pow(v,2);
return a11;
}
```

§3.5. Datos de salida y Gnuplot

Para facilitar el manejo e intepretación de los resultados, los datos de salida se mandan escribir a un archivo cada determinado tiempo, de acuerdo al valor que el usuario asigne a la variable $timefile$ del archivo de configuración INPUT. Dichas escrituras terminan de acuerdo al tiempo que el usuario indique en la variable $tmax$ del mismo archivo de configuración.

Para visualizar los datos utilizamos el software de graficación Gnuplot (`www.gnuplot.info`). Es un programa libre, muy flexible y portable. Genera gráficas en dos y tres dimensiones, ya sea directo a pantalla o en una gran variedad de formatos de imagen (png, jpeg, eps, latex, postscript, etc.).

El archivo de salida puede contener 4 o 5 columnas dependiendo si se necesita graficar en 2 o 3 dimensiones. La primer columna es la posición en X y en el caso de graficar en 3 dimensiones la segundo columna indica la posición en Y. Las siguientes columnas son el número de partículas, la velocidad y la presión.

La Figura 3.1 y la Figura 3.2, graficadas en 2 y 3 dimensiones respectivamente, muestran un mismo ejemplo. Se trata de la gráfica de velocidad de una onda de choque que es alimentada por una inyección senoidal de materia. Para lograr la representación tridimensional se utilizan curvas de nivel o isocontornos. La barra inferior en la Figura 3.2 indica mediante colores los valores de la velocidad.

Para realizar estas gráficas se tienen dos programas en gnuplot, **grafica2d** y **graficas3d**, los cuales se repiten dentro de un ciclo de acuerdo al número de archivos de datos que se hayan calculado. Los programas que realizan estos ciclos son **graph2D** y **graph3D**. Estos programas leen un pequeño archivo de configuración llamado GRAFICAS, que contiene los valores de las abscisas para cada una de las gráfica y el número de archivos que deben ser graficados. A continuación se muestra el programa graph2D.

```
#!/bin/bash
i=0;
input=[];
echo "valores establecidos para graficar";
while read line
do
input[$i]=$line;
i=$(($i+1));
```

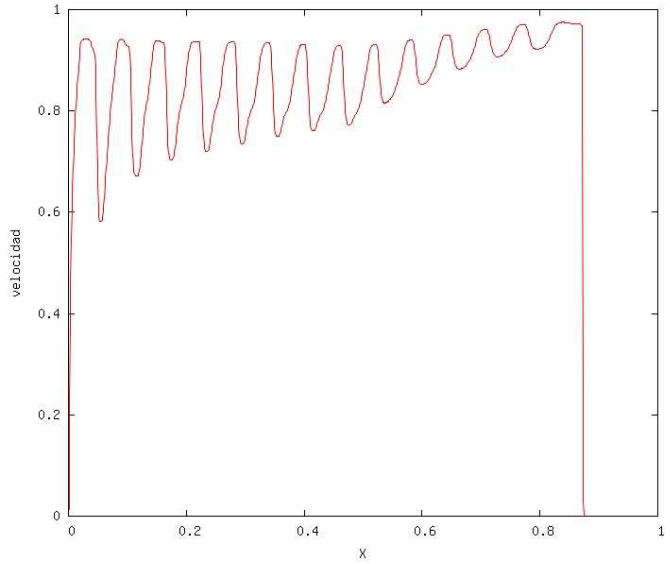

Figura 3.1: La gráfica muestra una onda de choque con inyección senoidal de materia. El programa escribe los archivos de datos para ser graficados en dos dimensiones si la variable *dimension* en el archivo de configuración es igual 2.

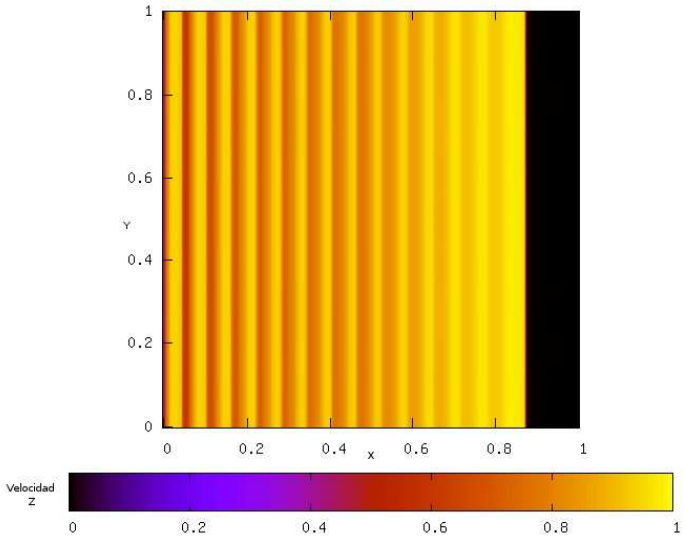

Figura 3.2: La gráfica muestra una onda de choque en 3 dimensiones mediante curvas de nivel de una onda de choque con inyección senoidal de material. El programa escribe los archivos de datos para ser graficados en tres dimensiones si la variable *dimension* en el archivo de configuración es igual 3.

```
done < "GRAFICAS"
n=${input[1]};
echo "n: $n";
v=${input[2]};
echo "v: $v";
p=${input[3]};
echo "p: $p";
t=${input[4]};
echo "numero de archivos de datos: $t";
dat=1
while [ $dat -lt $t ];
do
g=$(echo "scale=5; $dat/1000.0"| bc)
./grafica2 $dat $g $n $v $p
dat=$((dat+1))
done
```

§3.6. Mencoder y Mplayer

Como se mencionó en la sección anterior, los cálculos son guardados en archivos periódicamente. Esto permite ver la evolución del fenómeno estudiado colocando de manera secuencial las gráficas de cada uno de estos archivos. De esta forma los resultados pueden ser fácilmente presentados y comparados.

Las herramientas de software que permiten realizar esta labor son el Mencoder y Mplayer (`www.mplayerhq.hu`). Mencoder es el codificador de video que permirte crear las secuencias de video en una gran variedad de formatos. Mplayer es un reproductor de video capaz desplegar una gran variedad de formatos de video. Ambos fueron elegidos porque estan bajo la licencia GPL, tiene una gran versatilidad y están en constante desarrollo.

Aqui se muestra el script que utilizamos para generar 3 películas a partir de las imágenes en formato JPG creadas gnuplot de cada uno de las archivos de resultados.

```
#!/bin/bash
mencoder "mf://N.*.jpg"  -oac copy -mf fps=$1 -o test_N_fps$1.avi -ovc lavc
-lavcopts vcodec=msmpeg4v2:vbitrate=800
```

```
mencoder "mf://V.*.jpg"  -oac copy -mf fps=$1 -o test_V_fps$1.avi -ovc lavc
-lavcopts vcodec=msmpeg4v2:vbitrate=800
mencoder "mf://P.*.jpg"  -oac copy -mf fps=$1 -o test_P_fps$1.avi -ovc lavc
-lavcopts vcodec=msmpeg4v2:vbitrate=800
```

§3.7. Shell Bash

Para manipular y automatizar todas las aplicaciones anteriormente mencionadas se utiliza como herramienta de programación el shell Bash. El shell es el intérprete de comandos del sistema operativo y uno de los más importantes es Bash (acrónimo de Bourne again shell). Bash es compatible con su antecesor Bourne shell (sh) e incorpora caracterśticas de otros como Korn shell (ksh) y C shell (csh).

El script que ejecuta todos los programas de Maxima, perl y el compilador GCC para obtener el ejecutable del código Aztekas es el siguiente:

```
#!/bin/bash
./Calcula_matrix 1>/dev/null
perl Functions.pl
gcc func_planar.c func_matrix.c planar.c -lgsl -lgslcblas -lm -o aztekas
```

§3.8. Funciones principales del código

En la Figura 3.3 podemos ver de manera clara y resumida la secuencia de pasos que realiza el programa principal.

Como primer paso, ejecuta la función ReadData() la cual realiza la lectura del archivo de configuración INPUT que, como se mencionó en la sección §3.5, contiene importantes parámetros que determinaran el funcionamiento del código.

El archivo INPUT esta compuesto de los siguientes datos:

```
mesh:     500
n1:       10.0
n2:       0.05
v1:       0.0
v2:       0.0
```

```
p1:       13.3
p2:       0.01
tmax:     0.09
timefile: 0.004
dt:       0.00002
eta:      0.10
dimension: 3
```

mesh se refiere al número de divisiones que tendrá el dominio de X, es decir, que tan refinada será la malla; $n1$ y $n2$ son valor inicial de la densidad del número de particulas en la primera y segunda mitad del dominio; de igual manera $v1$, $v2$, $p1$ y $p2$ se refieren a la velocidad y presión inicial. El parámetro *tmax* indica hasta que tiempo llegarán los cálculos numéricos; $timefile$ marca el periodo de tiempo que transcurre cada vez que se manda a escribir los datos en los archivos de salida; *dt* es el tamaño de cada paso del tiempo y *eta* es el conficiente de viscosidad. Por último, *dimension* indica al programa como serán los archivos de datos para las gráficas, en dos o tres dimensiones por lo que su valor sólo puede ser 2 o 3.

El segundo paso del diagrama consiste en ejecutar la función INITFLOW() la cual prepara las condiciones iniciales del problema de acuerdo a lo que se establezca en en archivo INPUT. Dichas condiciones iniciales se guardan en 3 vectores (densidad de número de partículas, velocidad y presión), cada uno con el número de celdas que indica el parámetro *mesh*. La tercera etapa, depende si ha llegado al tiempo máximo de duración de los cálculos. Si el tiempo que indica la variable *time* es menor al tiempo máximo *tmax* entonces se procede al siguiente paso que es escribir a archivo la información que está en los 3 vectores de datos mediante la función OUTPUT().

El siguiente paso consiste de actualizar la variable *time* de acuerdo al valor que se le haya dado a *dt*. Mientras más pequeña sea *dt* más ciclos ejecutará el programa. A continuación se agregan las condiciones de frontera, las cuales consisten en intercambiar el último valor que hay en cada vector de datos con el valor anterior inmediato. De esta manera se crea el efecto de continuidad al final del dominio computacional.

La etapa siguiente es la ejecución de la función TIMESTEP(), la cual es la más importante del programa pues es la que realiza los cálculos numéricos y actualiza todos los valores que hay en los vectores de datos. A partir de este paso se repite el algoritmo y transcurridos los ciclos necesarios para que la varible *time* llegue al su valor máximo, el

programa finaliza.

§3.9. Pruebas del código

Como se mencionó en el capítulo 2, la prueba estándar para verificar el desempeño de un código hidrodinámico es el tubo de choque. La versión relativista de dicho problema es abordada por Martí & Müller (2003), el cual proporciona una solución analítica de dicho problema y además un programa para poder realizar pruebas y poder comparar códigos numéricos con dicha solución analítica.

Las condiciones para reproducir un tubo de choque son las siguientes:

	izquierda	derecha
Presión	1000.0	0.01
Densidad	1.0	1.0
Velocidad	0.0	0.0

Estas condiciones generan la propagación de una onda de choque. La Figura 3.4 muestra la gráfica inicial de la presión.

Estos valores iniciales son mucho más extremos que los mencionados en el capítulo 2, con lo cual se pone a prueba la capacidad del código para manejar fuertes discontinuidades, como las que se presentan en las ondas de choque. La Figura 3.5 y la Figura 3.6 muestran las gráficas de presión para el tiempo $t = 0.04$s y $t = 0.20$s respectivamente. Las gráficas muestran la solución numérica junto con la solución analítica.

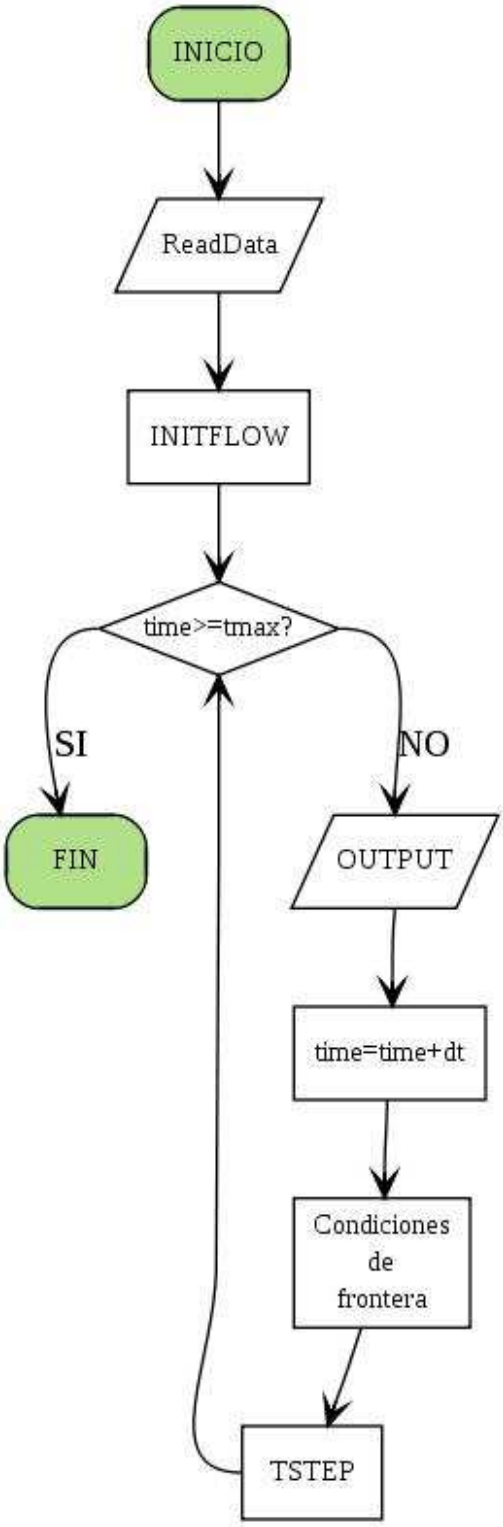

Figura 3.3: Diagrama de flujo del programa principal. El primer paso es la lectuta de datos(ReadData), el segundo las condiciones iniciales(INITFLOW) el tercero es la comprobación si se ha llegado al tiempo máximo establecido, el cual lleva al ciclo que consiste en escritura de datos(OUTPUT), actualización de la variable *time*, la aplicación de las condiciones de frontera y el cálculo numérico de todas las variables del problema(TSTEP).

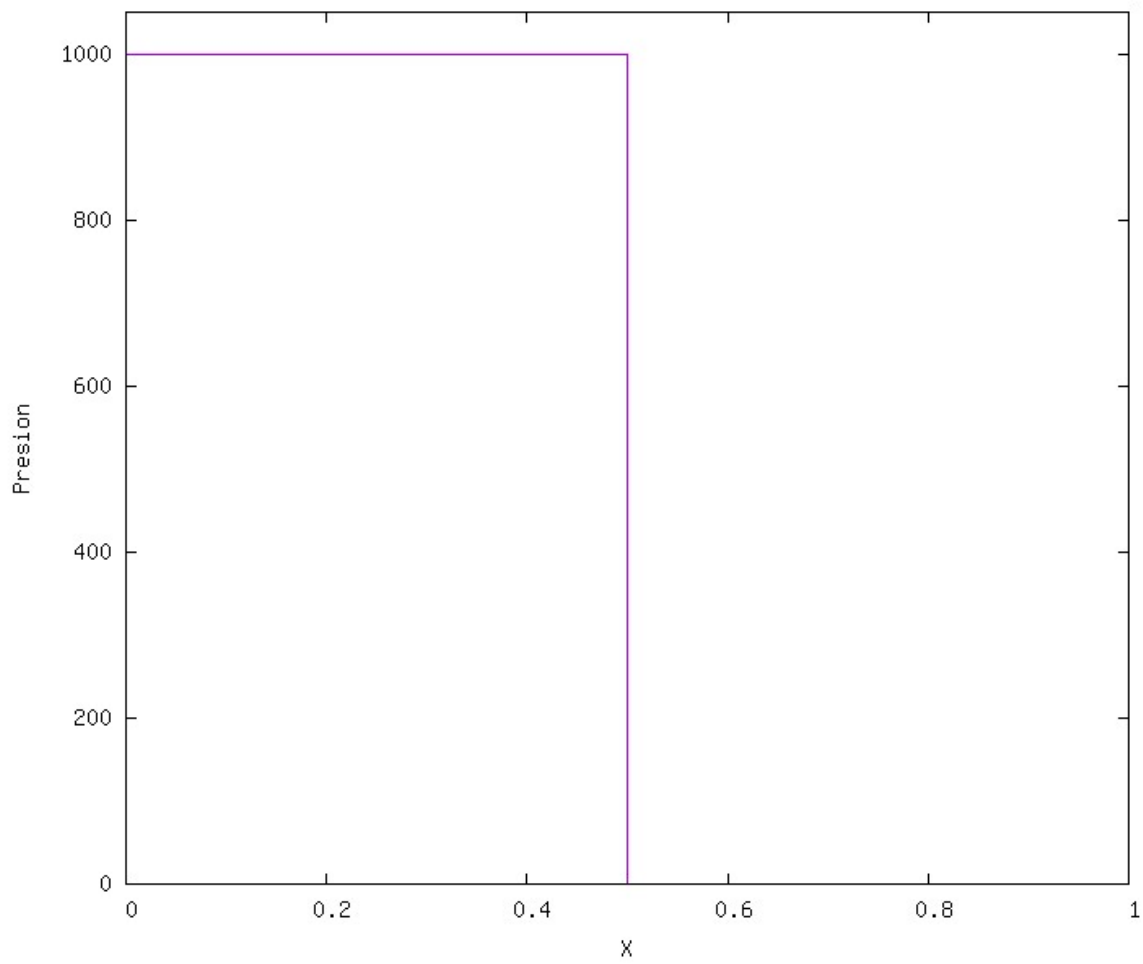

Figura 3.4: La gráfica muestra la condición inicial de la presión para hacer la prueba estándar de un código hidrodinámico relativista (cf. Martí & Müller, 2003).

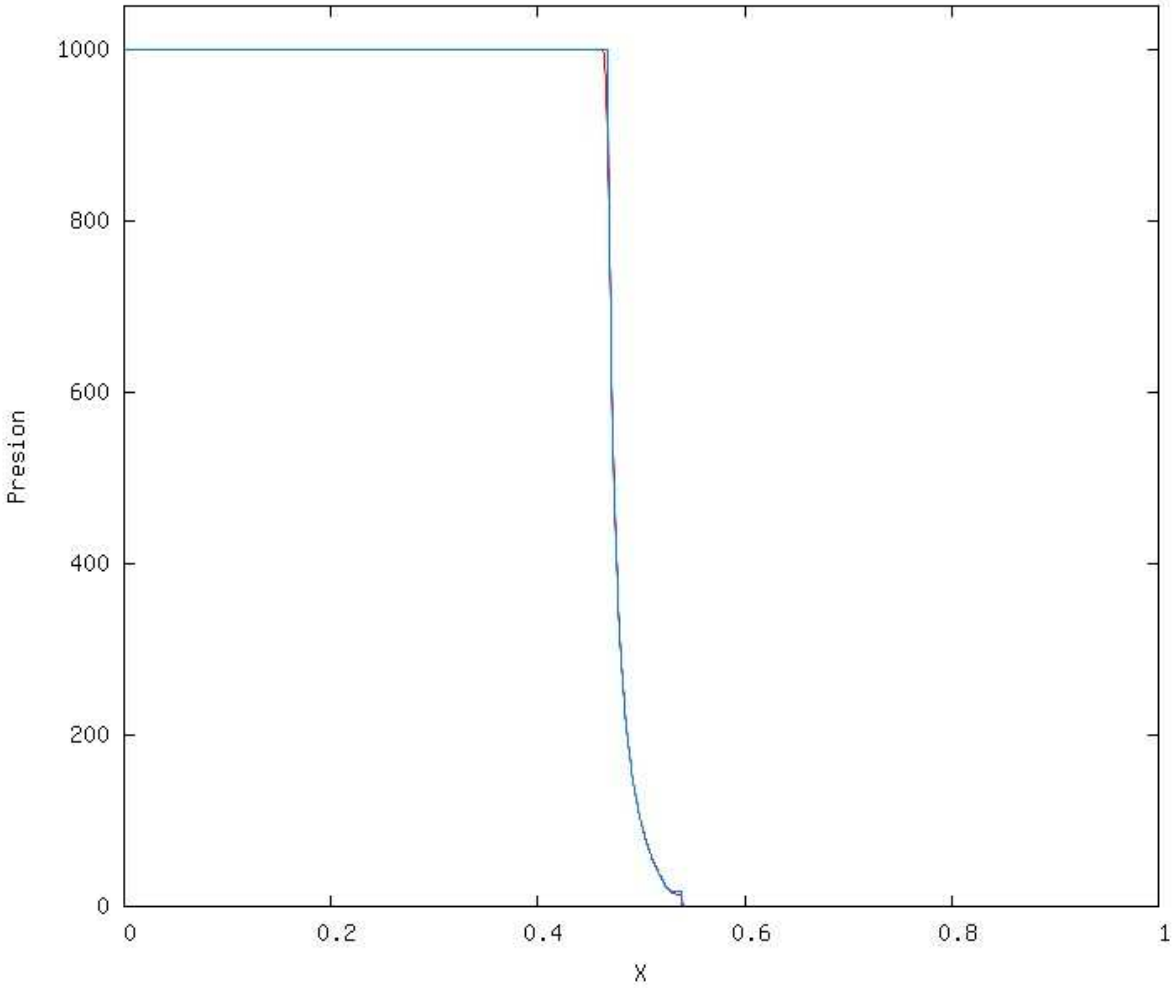

Figura 3.5: La gráfica muestra la evolución de la presión en el tubo de choque relativista para el tiempo $t = 0.04s$. de la solución numérica y de la solución analítica.

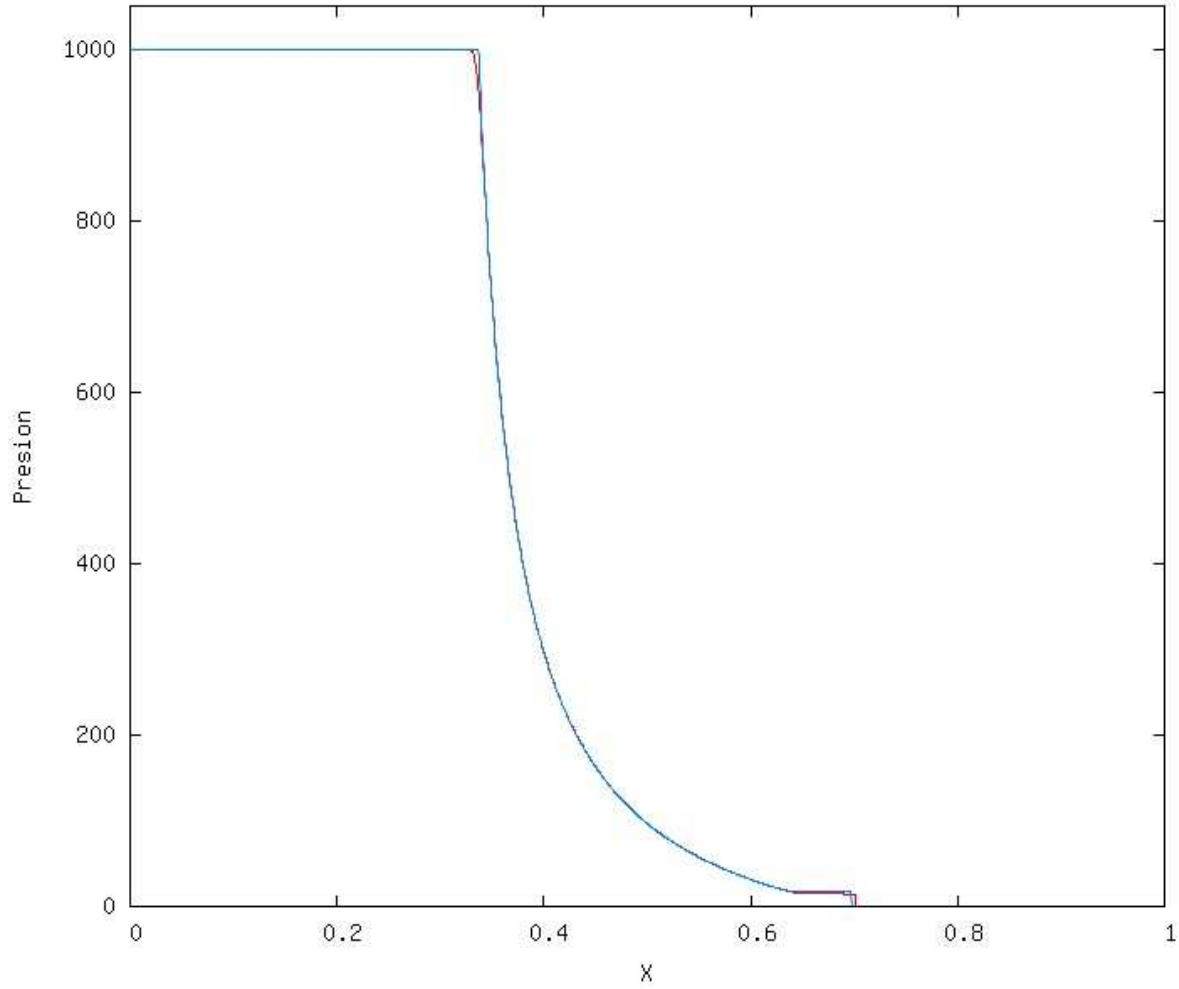

Figura 3.6: La gráfica muestra la evolución de la presión en el tubo de choque relativista para el tiempo $t = 0.20s$. de la solución numérica y de la solución analítica.

Capítulo 4

Choques internos en jets relativistas[†]

En este capítulo mostraremos una aplicación del código aztekas para un caso específico de la astrofísica relativista. Veremos los fundamentos de las onda de choque dentro de un jet relativista y posteriormente, mediante simulaciones numéricas, comprobaremos dicha teoría.

§4.1. Introducción

Los "nudos" o "manchas" aparentemente superlumínicos observados a lo largo de jets relativistas de cuasares y microcuasares son generalmente interpretados como ondas de choque moviéndose a través de un jet. Rees (1966) fue el primero que matemáticamente predijo los aparentes movimientos superlumínicos observados en jets extragalácticos, los cuales son debidos a efectos geométricos y a las velocidades relativistas del movimiento propio de las "manchas" dentro del jet. El mismo autor propuso que esas "manchas" observadas son producidas por variaciones de velocidad del flujo que se mueve dentro del jet (Rees, 1978). Adicionalmente observaciones de blazares a lo largo de muchos años muestran que la variabilidad de la intensidad y polarización son en su mayoría generados por choques transversales propagados a lo largo del jet (cf. Hagen-Thorn et al., 2007; Spada

[†] Este trabajo se realizó con la colaboración Sergio Mendoza y Juan Carlos Hidalgo[1]

[1] Astronomy Unit, Queen Mary College, University of London, Mile End Road, London E1 4NS, United Kingdom.

et al., 2001; Sahayanathan & Misra, 2005). Otra situación donde aparecen ondas de choque internas es en el modelo Fireball para destellos largos de rayos gama(GRB). En tal modelo, la existencia de ondas de choque internas es un efectivo mecanismo para la generación de los rayos gama observados, estas ondas de choque son muy probablemente causadas por variaciones de velocidad en el flujo (Rees & Meszaros, 1994).

Detallados modelos numéricos han sido desarrollados para explicar el origen de la radiación característica asociada a los choques internos (Blandford & Konigl, 1979; Hughes et al., 1985; Marscher, 1996), con especial atención a la polarización de la radiación observada como una prueba de los choques internos que aceleran el material. En este artículo exploramos la formación de superficies de trabajo internas que se propagan en jets relativistas originados de las variaciones periódicas de la velocidad y de la masa inyectada. Considerando una propagación unidimensional de partículas de masa, presentamos una extensión del modelo no relativista formulado por Cantó et al. (2000), en el cual, la formación y evolución de superficies de trabajo internas es modelado para eyecciones correspondientes a jets estelares (Raga & Kofman, 1992; Raga et al., 2004). Este modelo tiene la ventaja de proveer expresiones analíticas para la energía cinética irradiada por las partículas de masa que colisionan dentro de la superficie de trabajo. Asumimos que los gradientes de presión entre las partículas de fluido son despreciables y que la escala de tiempo de la radiación es mucho más corta que el tiempo que le toma formar un superficie de trabajo particular. De esta manera podemos obtener expresiones analíticas para la velocidad de las partículas de los frentes de choque y para la luminosidad del gas que ha sido chocado. Nuestro modelo analítico es comparado con el código numérico aztekas (cf. capítulo 3) que resuelve todas las ecuaciones relativistas de la hidrodinámica en una dimensión como lo presenta Blandford & McKee (1976); Hidalgo & Mendoza (2005). En nuestra simulación los límites para la validación de nuestro modelo son discutidos.

§4.2. Dinámica de la superficie de trabajo

La formación de choques a lo largo de un jet relativista se ha explicado mediante diferentes fenómenos tales como la presencia de inhomogeneidades en el medio circundante, desviaciones y cambios en la geometría de los jets y fluctuaciones temporales en los parámetros de eyección (ver por ejemplo Mendoza, 2000; Mendoza & Longair, 2001, 2002; Cantó et al., 2000, y sus referencias). En este caso nos enfocaremos en las fluctuaciones de eyección. Cuando la velocidad de emisión de partículas varía con el tiempo, una

parcela que fue emitida después pero más rápida, eventualmente golpea a la que fue emitida previamente pero de menor velocidad. Esto produce un par de choques divididos por una discontinuidad de contacto, la llamada superficie de trabajo, que viaja en el jet con una velocidad promedio. A este tipo de formación de superficies radiativas se le conoce como modelo de choque interno (Rees & Meszaros, 1994; Daigne & Mochkovitch, 1998). Aunque varias extensiones y aspectos particulares de este modelo han sido presentados en la literatura, no hay muchas descripciones analíticas de este fenómeno. Aquí presentamos una aproximación analítica de la formación de una superficie de trabajo interna en un jet relativista. Asumimos que las escalas de tiempo de radiación son menores comparadas con los tiempos dinámicos característicos del problema. En consecuencia la presión del fluido es despreciable y las colisiones son descritas balísticamente. Esta suposición es válida si el flujo dentro de jet es prácticamente adiabática y no turbulento (Sahayanathan & Misra, 2005).

Para seguir la evolución de la superficie de trabajo, consideramos que la fuente de eyección del material es en la dirección x con una velocidad $v(\tau)$ y una tasa de eyección de masa $\dot{m}(\tau)$, ambas variables dependen del tiempo τ medido desde el origen del jet.

Una vez que el material es eyectado del origen, este sigue un camino de corriente libre (free streaming) (Raga et al., 1990). Esta aproximación es válida debido a que el número de Mach es grande para el flujo y porque los procesos radiativos que enfrían el fluido ocurren en tiempo menores a los asociados al problema estudiado en este artículo (Spada et al., 2001). La formación de la una superficie de trabajo es estudiada como la intersección de dos diferentes parcelas de material eyectado en tiempos $\tau_1 = 0$ y τ_2 con velocidades de flujo $v_1 = v(\tau_1)$ y $v_2 = v(\tau_2) = v_1 + \alpha\,\tau_2$ respectivamente (ver Figura 4.1), donde $\alpha := (\mathrm{d}u/\mathrm{d}\tau)_{\tau_1}$. Si $\alpha > 0$, la segunda parcela eventualmente alcanzará a la primera. Al tiempo τ_2, la distancia entre ambas parcelas esta dada por $v_1\,(\tau_2 - \tau_1)$ y el tiempo t_m (medido en el marco de referencia del origen) cuando ambas parcelas se unen

$$\begin{aligned} t_\mathrm{m} =& \frac{1}{\alpha} v_1 \gamma^2\,(v_1)\left\{1 - v_1^2 - \alpha v_1 \tau_2\right\}, \\ =& \frac{v_1}{\alpha}\left\{1 - \gamma^2 \alpha v_1 \tau_2\right\}, \end{aligned} \tag{4.1}$$

donde $\gamma\,(u) := 1/\sqrt{\left(1-u^2\right)}$ representa el factor de Lorentz del fluido con velocidad u, y asumiendo que la velocidad de la luz es $c = 1$. La superficie de trabajo se forma a la distancia

$$d_f = (t_m + \Delta t)v_1, \tag{4.2}$$

del origen.

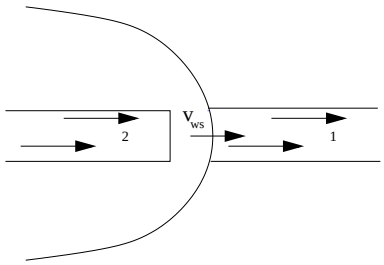

Figura 4.1: Cuando un flujo rápido 2 se mueve sobre un flujo lento 1, se genera una superficie de trabajo que se mueve con velocidad $v_{\rm ws}$ como resultado de esta interacción.

Siguiendo el formalismo no relativista propuesto inicialmente por Cantó et al. (2000), asumimos que la superficie de trabajo es delgada y que no tiene pérdidas de masa (por ejemplo, eyecciones laterales de materia (ver Falle & Raga, 1993, 1995)). Debido a que el fluido se aproxima como un flujo de corriente libre, su velocidad $v(x,t)$ como una función de x y t es simplemente

$$v(x,t) = v_0(\tau) = \frac{x}{t-\tau}. \tag{4.3}$$

Esta relación implica que la posición $x_{\rm ws}$ de la superficie de trabajo del flujo "aguas abajo" esta dada por

$$x_{\rm ws} = v_1(t - \tau_1), \tag{4.4}$$

y el correspondiente flujo "aguas arriba" toma la forma

$$x_{\rm ws} = v_2(t - \tau_2). \tag{4.5}$$

Por otra parte, debido a que el flujo es de corriente libre, la velocidad de la superficie de trabajo esta dada por la velocidad $v_{\rm ws}$ de su centro de masa, el cual está determinada por (Landau & Lifshitz, 1994)

$$v_{\mathrm{ws}} = \frac{1}{M_\gamma} \int_{\tau_1}^{\tau_2} \gamma\left(v(s)\right) \dot{m}(s) v(s) \mathrm{d}s, \tag{4.6}$$

donde la masa pesada M_γ eyectada entre los tiempos τ_1 y τ_2 es

$$M_\gamma = \int_{\tau_1}^{\tau_2} \gamma\left(v(s)\right) \dot{m}(s) \mathrm{d}s. \tag{4.7}$$

Con esta velocidad, la posición de la superficie de trabajo es

$$x_{\mathrm{ws}} = (t - \tau_2) v_{ws} + \frac{1}{M_\gamma} \int_{\tau_1}^{\tau_2} \gamma\left(v(s)\right)\, \dot{m}(s)\, v(s)\, (\tau_2 - s)\, \mathrm{d}s. \tag{4.8}$$

De las ecuaciones (4.4) y (4.5) se sigue que la posición de la superficie de trabajo como una función de los tiempos τ_1 y τ_2 es

$$x_{\mathrm{ws}} = \frac{v_1 v_2}{v_2 - v_1} \left(\tau_2 - \tau_1\right). \tag{4.9}$$

De la misma manera, del mismo conjunto de ecuaciones (4.4) y (4.5) es posible calcular el tiempo t como una función τ_1 y τ_2

$$t = \frac{\tau_2 v_2 - \tau_1 v_1}{v_2 - v_1}. \tag{4.10}$$

Para un valor dado de la posición x_{ws}, las expresiones (4.4), (4.5) y (4.8) establecen una relación entre los tiempos τ_1 y τ_2. Tomando τ_2 como parámetro, podemos construir la posición y la velocidad de la superficie de trabajo como una función de τ_2 y calcular los valores de las cantidades relevantes del problema, tales como la energía disponible en movimiento de la superficie de trabajo. Esta relación es uno a uno en tanto que la velocidad de eyección $v(\tau)$ se incremente uniformemente.

A fin de calcular la cantidad de energía cinética radiada mientras la superficie de trabajo se va moviendo, tomamos en cuenta la energía E_0 del material que ha sido eyectado, la cual es aproximadamente [†]

$$E_0 = \int_{\tau_1}^{\tau_2} \dot{m}(s)\, \gamma\left(v(s)\right)\, \mathrm{d}s, \tag{4.11}$$

y la energía E_{ws} del material dentro de la superficie de trabajo está dada por

[†] El término $\int \dot{\gamma}(s) m(s)\, ds$ es subdominante en nuestro problema en tanto que la variación de la velocidad no disminuya dramáticamente a valores pequeños.

$$E_{ws} = m\gamma_{ws}, \tag{4.12}$$

donde el factor Lorentz $\gamma_{\rm ws}$ del material de la superficie de trabajo es $\gamma_{\rm ws}^{-2} = 1 - v_{\rm ws}^2$.

Si ahora asumimos que la pérdida de energía a largo del jet, $E_r = E_0 - E_{\rm ws}$, es completamente radiada entonces la luminosidad $L := {\rm d}E_{\rm r}/{\rm d}t$ de la superficie de trabajo está dada por

$$\begin{aligned} L = &\frac{\dot{m}(\tau_2)}{{\rm d}t/{\rm d}\tau_2}\left\{\gamma_{\rm ws} + \frac{m}{M_\gamma}\gamma_{\rm ws}^3\gamma_2\left(v_{\rm ws}v(\tau_2) - v_{\rm ws}^2\right) - \gamma_2\right\} - \\ &- \frac{\dot{m}(\tau_1)}{{\rm d}t/{\rm d}\tau_2}\frac{{\rm d}\tau_1}{{\rm d}\tau_2}\left\{\gamma_{\rm ws} + \frac{m}{M_\gamma}\gamma_{\rm ws}^3\gamma_1\left(v_{\rm ws}v(\tau_1) - v_{ws}^2\right) - \gamma_1\right\}, \end{aligned} \tag{4.13}$$

donde los factores de Lorentz $\gamma_{1,2}^{-2} := 1 - v^2(\tau_{1,2})$ y, como anteriormente lo hicimos, mantenemos τ_2 como un parámetro libre de esta expresión. En consecuencia, la luminosidad L en la ecuación (4.13) se encuentra escribiendo las expresiones para las variables τ_1, v_1, v_2, $v_{\rm ws}$ y las derivadas ${\rm d}\tau_1/{\rm d}\tau_2$ y ${\rm d}t/{\rm d}\tau_2$ como funciones del parámetro libre τ_2.

§4.3. Un flujo de descarga constante

Como un ejemplo de nuestra descripción analítica, consideraremos un caso particular de una descarga constante $\dot{m}$ para calcular la luminosidad L que es obtenida a través de oscilaciones simples de la velocidad de emisión de las partículas. Asumimos que la velocidad inyectada v tiene una forma periódica

$$v(\tau) = v_0 - \epsilon^2 \sin\tau. \tag{4.14}$$

En la cual la constante ϵ es una cantidad pequeña. Este tipo de velocidades de emisión oscilatorias han sido ampliamente usadas para la descripción de choques internos tanto en el caso newtoniano como en el relativista (Cantó et al., 2000; Panaitescu et al., 1999). Para este ejemplo usamos un sistema de unidades en el cual $\dot{m} = 1$. Además establecemos la unidad de tiempo para que la frecuencia de oscilación sea $\omega = 1$. Como consecuencia de esta suposición, la luminosidad L es tal que sus dimensiones son las mismas que las dimensiones de $\dot{m}$

$$[L] = [\dot{m}], \tag{4.15}$$

y es también adimensional.

Si asumimos que el flujo inyectado el altamente relativista, entonces es posible resolver analíticamente las ecuaciones (4.6)–(4.13) para $\mathrm{O}(\gamma^{-1})$. Sin embargo las expresiones analíticas son largas y complejas, esto es debido a que el factor de Lorentz aparece frecuentemente, lo cual no ocurre en el caso no relativista. Hemos calculado integraciones numéricas de las ecuaciones (4.6)–(4.13), usando los valores $v_0 = 0.9$ y $\epsilon^2 = 0.09$. Los resultado se muestran la Figura 4.2 y son muy similares en forma a los obtenidos por Cantó et al. (2000).

El abrupto destello en la luminosidad muestra que la energía cinética debe ser radiada de manera muy efectiva, lo cual permite el aumento de las emisiones de fotones de alta energía en varias longitudes de onda dependiendo de la fuerza del choque.

Los mecanismos radiativos y la caracterización del espectro de objetos particulares será tema de una artículo subsecuente. Para un estudio detallado de los mecanismos radiativos de los choques internos en GRBs ver Daigne & Mochkovitch (1998).

§4.4. Solución numérica en 1–D

A fin de comparar nuestros cálculos analíticos y los resultados que se muestran en la Figura 4.2 hemos utilizado el código aztekas.

Para hacer una comparación numérica con los resultados obtenidos en la sección §4.3, en la posición $x = 0$ de nuestro dominio inyectamos a todo tiempo t materia con una inyección constante de masa $\dot{m}$. Asumimos que la masa inyectada tiene una velocidad dada por la ecuación (4.14). Establecemos un flujo constante de descarga $\dot{n} = 1$, lo cual implica que la densidad del número de partículas en el punto $x = 0$ esta dada por

$$n(t, x = 0) = \frac{\dot{n}}{\gamma v(t)} = \frac{\sqrt{1 - v^2(t)}}{v(t)}. \tag{4.16}$$

A fin de analizar un solo destello de luminosidad, como el descrito en la Figura 4.2, el valor de la velocidad se asume constante después $t = 2\pi$, esto es $v(t > 2\pi, x = 0) = 0.9$. Las condiciones iniciales para el flujo son tales que $v(t = 0, x) = 0.9$, $p(t = 0, x) = 0.001$ y $n(t = 0, x) = \sqrt{1 - v^2(t = 0, x)}/v(t = 0, x)$. Las ondas de choque obtenidas por

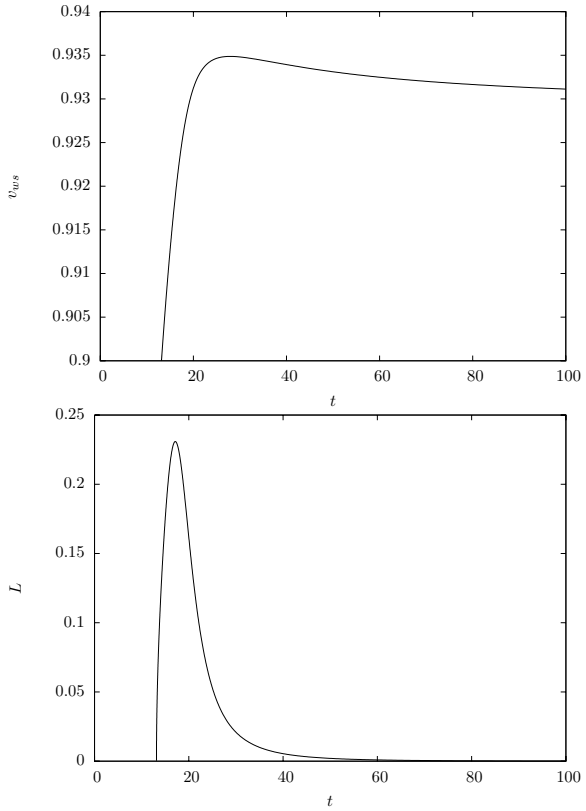

Figura 4.2: La figura muestra la velocidad $v_{\rm ws}$de la superficie de trabajo y su luminosidad L como una función del tiempo para un solo periodo de oscilación con una velocidad de entrada dada por la ecuación (4.14) y un flujo constante de descarga.

la variación de la velocidad del flujo son capturadas numéricamente introduciendo un viscosidad artificial (Book et al., 1975). Una vez que la posición de ambas ondas de choque son conocidas, entonces a cada momento, la energía $E_{\rm ws}$ del flujo entre ambas ondas de choque (la superficie de trabajo) es calculada como la suma $\sum n\gamma\Delta x$ donde la sumatoria se hace para cada celda numérica de tamaño Δx que se encuentran entre ambos choques. La energía del flujo de entrada se calcula con la ecuación (4.11) y la derivada que aparece en el cálculo de la luminosidad $L := \mathrm{d}\left(E_0 - E_{\rm ws}\right)/\mathrm{d}t$ se calcula numéricamente y suavizada utilizando un corrector de flujo (cf. sección §2.4). La figura 4.3 muestra el resultado numérico comparado con la descripción analítica descrita anteriormente. Como se muestra en la figura, la aproximación analítica es una buena manera para describir la dinámica y la energía de las superficies de trabajo formadas por la variación de la velocidad de entrada.

También hemos realizado cálculos para dos casos más en los cuales la presión p ha sido escrita como ζp en las ecuaciones hidrodinámicas, con $\zeta = 0.01,\ 0.02$. Asumimos que la presión y la densidad siguen la relación politrópica $p \propto n^{4/3}$, la cual se ajusta a la relación (1.24) como lo describe Tooper (1965). No se modifican las condiciones iniciales y de frontera. Sin embargo, la energía de entrada E_0 ahora es dada

$$E = \int_0^t \left[1 + \left(3 + v^2\right)\frac{1}{n}\frac{\mathrm{d}p}{\mathrm{d}t} - \left(3 + v^2\right)\frac{p}{n^2}\right]\gamma \mathrm{d}t. \tag{4.17}$$

La energía $E_{\rm ws}$ dentro de la superficie de trabajo se calcula como la suma $E_{\rm ws} = \sum n\gamma\Delta x + \sum p\gamma\left(3 + v^2\right)\Delta x$, donde la sumatoria se toma en todas la celdas de ancho Δx del dominio que hay entre ambas ondas de choque. El resultado obtenido se presenta en la Figura 4.3. Como se puede ver, el pico de luminosidad se forma al mismo tiempo. Sin embargo, la intensidad del pulso se incrementa conforme ζ aumenta. El caso $\zeta = 1$, que corresponde a un flujo ultrarelativista no se muestra en la Figura 4.3 debido a que su pico de luminosidad tiene un valor mucho más grande.

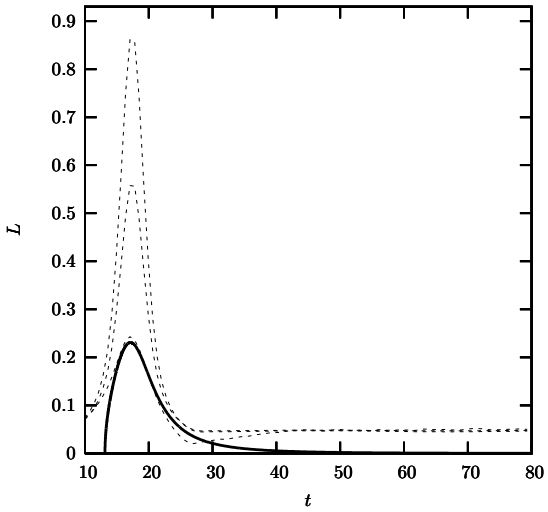

Figura 4.3: La figura muestra la luminosidad adimensional L de la superficie de trabajo como una función del tiempo adimensional para un simple periodo ($t = 2\pi$) de oscilación con una velocidad de entrada dada por la ecuación (4.14) y un flujo constante de descarga. La linea continua muestra la aproximación analítica bajo la suposición de presión nula. De abajo hacia arriba, las curvas punteadas muestran los cálculos numéricos para los casos $\zeta = 0, 0.01, 0.02$. La razón por la cual la luminosidad no llega a cero en un tiempo suficientemente grande ($\gtrsim 35$) para la solución numérica es debido a que la velocidad de entrada alcanza un valor constante $v = 0.9$ después de un periodo de oscilación, mientras que la aproximación analítica llega a un valor nulo.

Capítulo 5

Conclusiones

El título de la presente tesis, "Un código libre de hidrodinámica relativista", indica que no pretende ser el único código para resolver las ecuaciones de la hidrodinámica relativista. Existen muchas maneras de resolver un problema, y la resolución de las ecuaciones de la hidrodinámica relativista no es la excepción.

En este trabajo se presentó el desarrollo de un software especializado para la astronomía. La motivación principal fue la de crear una herramienta que pudiera ser utilizada por cualquier persona que se interesara y sin ninguna restricción. Es por ello que se siguió la filosofía del software libre la cual indica que el conocimiento debe compartirse. De esta manera, el código fuente del programa estará disponible para todo mundo y además, como lo indica la licencia de software libre GPL (apéndice §E), también se le otorga a cualquier usuario el permiso de modificar el programa de acuerdo a sus necesidades. De esta forma, el código crecerá y mejorará.

Por otro lado, la resolución de grandes y cada vez más complejos sistemas de ecuaciones se ha convertido en un prolífico campo del ciencia actual. Cada vez más ramas del conocimiento se apoyan en modelos numéricos para crear simulaciones de sus temas de trabajo o para predecir resultados de diversos problemas, particularmente en la astronomía, donde no es posible recrear en laboratorio los fenómenos que estudia. Por lo que la elaboración de un algoritmo numérico desde cero, podría resultar a primera vista una labor repetitiva, pero esto no es así por dos razones. La primera es que la programación de un código propio permite participar activamente en el debate científico por la búsqueda del mejor código, además de que brinda experiencia y una serie de conocimientos que solamente el escribir un código numérico puede dar. La segunda razón es por motivos más prácticos,

muchas veces es necesario cambiar el código para agregar nuevos fenómenos u optimizarlo para alguna aplicación muy particular. En este proceso, el uso de un código escrito por otras personas no permite que esta modificación sea rápida y eficiente. Es mucho más fácil entender las limitaciones y los errores de las simulaciones si conocemos el código a detalle.

Adicionalmente, concluimos que el método elegido para la primer versión del código Aztekas (sección §2.4 y §2.5) permite realizar pruebas satisfactorias (sección §3.9). De esta manera fue posible utilizarlo en su primera aplicación práctica (capítulo 4), la construcción de una solución completamente relativista sobre el problema de un jet con velocidad de inyección variando periódicamente. Se mostró que las aproximaciones analíticas concuerdan muy bien con una solución numérica bajo la suposición de gradientes nulos de presión en el flujo.

Es importante destacar que a pesar de que el código se creo para resolver las ecuaciones de la hidrodinámica relativista, el programa esta pensado para resolver sistemas de ecuaciones en forma conservativa (1.51). Esto quiere decir que el código es más general por lo que podría ser utilizado para resolver otras problemas.

Por último, a futuro se espera trabajar con el código en 2 y 3 dimensiones y agregar fenómenos magnéticos y gravitatorios, además de incluir a más científicos y programadores a este proyecto de software libre.

Apéndice

§A. Manipulación algebraica de un sistema hiperbólico de ecuaciones

Como se mencionó en la sección (§1.9) las ecuaciones de la hidrodinámica relativista (1.52), (1.54) y 1.55 desde el punto de vista del álgebra matricial, pueden expresarse como un sistema de ecuaciones que tiene la forma

$$\frac{\partial A}{\partial t} + \frac{\partial B}{\partial x} = 0,$$

donde A y B son vectores con 3 variables únicamente ya que al utilizar la ecuación de estado (1.24) simplificamos las ecuaciones y eliminamos una variable, con lo cual podemos expresar el sistema en función de las n, v y p y la constante k (índice politrópico). De esta forma los vectores A y B quedan de la siguiente manera:

$$A = \begin{pmatrix} \frac{n}{\sqrt{1-v^2}} \\ \frac{pk}{(k-1)(1-v^2)} - p \\ \frac{vpk}{(k-1)(1-v^2)} \end{pmatrix} \qquad B = \begin{pmatrix} \frac{nv}{\sqrt{1-v^2}} \\ \frac{vpk}{(k-1)(1-v^2)} \\ \frac{v^2pk}{(k-1)(1-v^2)} + p \end{pmatrix}$$

Para poder trabajar con este sistema, primero definimos un vector simple U con las variables que nos interesan

$$U = \begin{pmatrix} n \\ v \\ p \end{pmatrix}$$

y aplicamos la regla de la cadena

$$\frac{\partial A}{\partial U}\frac{\partial U}{\partial t} + \frac{\partial B}{\partial U}\frac{\partial U}{\partial x} = 0.$$

Lo cual implica calcular el Jacobiano de cada vector, con lo que se produce una matriz de 3x3. Posteriormente calculamos la matriz inversa del Jacobiano del vector A y despejamos la expresión.

$$\frac{\partial A}{\partial U}^{-1} \{ \frac{\partial A}{\partial U}\frac{\partial U}{\partial t} + \frac{\partial B}{\partial U}\frac{\partial U}{\partial x} = 0 \},$$

con lo cual se logra expresar el sistema de ecuaciones con la siguiente forma

$$\frac{\partial U}{\partial t} + M\frac{\partial U}{\partial x} = 0, \tag{A.1}$$

en donde M es la matriz que resulta de la multiplicación de la matriz inversa del Jacobiano de A por la matriz Jacobiana de B

El programa que realiza todo procedimiento anteriormente descrito el el siguiente

```
#!/bin/bash
maxima<<EOF
/*Lectura de las ecuaciones*/
  batch("EOS");
  batch("VECTOR1");
  batch("VECTOR2");
 /*derivadas vector1  respecto a n, v  p*
  exp_11: diff(a1, n);
  exp_12: diff(a1, v);
  exp_13: diff(a1, p);
  exp_21: diff(a2, n);
  exp_22: diff(a2, v);
  exp_23: diff(a2, p);
  exp_31: diff(a3, n);
  exp_32: diff(a3, v);
  exp_33: diff(a3, p);
  a: matrix([exp_11,exp_12,exp_13],[exp_21,exp_22,exp_23],[exp_31,exp_32,exp_33])$
 /*Inversa de la matriz a*/
  a_int: invert(a)$

 /*derivadas vector2 respecto a n, v  p*
  exp_11: diff(b1, n)$
  exp_12: diff(b1, v)$
  exp_13: diff(b1, p)$
```

```
  exp_21: diff(b2, n)$
  exp_22: diff(b2, v)$
  exp_23: diff(b2, p)$
  exp_31: diff(b3, n)$
  exp_32: diff(b3, v)$
  exp_33: diff(b3, p)$

  b: matrix([exp_11,exp_12,exp_13],[exp_21,exp_22,exp_23],[exp_31,exp_32,exp_33])$
  /*multiplicacion de matrices*/
  M: a_int . b$
  /*escritura del resultado al archivo MATRIZ_M*/
    stringout("MATRIZ_M",ratsimp(M[1,1]),ratsimp(M[1,2]),
    ratsimp(M[1,3]),ratsimp(M[2,1]),ratsimp(M[2,2]),ratsimp(M[2,3]),
    ratsimp(M[3,1]),ratsimp(M[3,2]), ratsimp(M[3,3]))$
EOF
```

§B. Código fuente del programa Functions.pl

Programa escrito en Perl que cambia el formato del archivo MATRIZ_M entregado por Maxima, para crear un archivo de funciones, con su respectiva cabecera, en lenguaje C para para compilarse con el programa principal.

```
#!/bin/perl

sub change
{
  $a=$_[0];

$a=~s/([a-z])\^(\d)/pow($1,$2.0)/g;

if($a =~ m/\)\^/)
{
 do{
   $i = index($a,")^");
```

```
   $j = $i+1;
   my $flag=0;
   my $end = substr($a,$j,1);

      do{
         $j--;

         $end = substr($a,$j,1);

          if($end eq ')' )
         {$flag++;}
          elsif($end eq '(' )
         { $flag--;}

         }while $flag != 0;

    substr $a, $j, 1, '<';
    substr $a, $i, 1, '>';
    if($a =~ m/\>\^/)
    {
      $a=~s/<(.+?)>\^(\d)/pow($1, $2.0)/;
    }
    if($a =~ m/\>\^\(/)
    {
      $a=~s/<(.+?)>\^\((\d)\/(\d)\)/pow($1,$2.0\/$3.0)/;
    }

  }while($a =~ m/\)\^/);
}

return $a;
}

open(Matrix, 'MATRIZ_M');
```

```
@lines = <Matrix>;
#print @lines;
$a11 = $lines[1];
$a12 = $lines[2];
$a13 = $lines[3];
$a21 = $lines[4];
$a22 = $lines[5];
$a23 = $lines[6];
$a31 = $lines[7];
$a32 = $lines[8];
$a33 = $lines[9];

$a11= change $a11;
$a12= change $a12;
$a13= change $a13;
$a21= change $a21;
$a22= change $a22;
$a23= change $a23;
$a31= change $a31;
$a32= change $a32;
$a33= change $a33;

open(Matrix, ">func_matrix.h");
print Matrix "double funct_A11(double n, double v, double p, double k);\n";
print Matrix "double funct_A12(double n, double v, double p, double k);\n";
print Matrix "double funct_A13(double n, double v, double p, double k);\n";

print Matrix "double funct_A21(double n, double v, double p, double k);\n";
print Matrix "double funct_A22(double n, double v, double p, double k);\n";
print Matrix "double funct_A23(double n, double v, double p, double k);\n";

print Matrix "double funct_A31(double n, double v, double p, double k);\n";
print Matrix "double funct_A32(double n, double v, double p, double k);\n";
print Matrix "double funct_A33(double n, double v, double p, double k);\n";
```

```
close(Matrix);

open(Matrix, ">func_matrix.c");
print Matrix "#include<stdio.h>\n";
print Matrix "#include<math.h>\n";

print Matrix "double funct_A11(double n, double v, double p, double k)\n";
print Matrix "{\n";
print Matrix "double a11;\n";
$a11= change $a11;
print Matrix "a11 = $a11";
print Matrix "return a11;\n";
print Matrix "}\n";
print Matrix "\n";
print Matrix "double funct_A12(double n, double v, double p, double k)\n";
print Matrix "{\n";
print Matrix "double a12;\n";
$a12= change $a12;
print Matrix "a12 = $a12";
print Matrix "return a12;\n";
print Matrix "}\n";
print Matrix "\n";
print Matrix "double funct_A13(double n, double v, double p, double k)\n";
print Matrix "{\n";
print Matrix "double a13;\n";
$a13= change $a13;
print Matrix "a13 = $a13";
print Matrix "return a13;\n";
print Matrix "}\n";
print Matrix "\n";

print Matrix "double funct_A21(double n, double v, double p, double k)\n";
print Matrix "{\n";
print Matrix "double a21;\n";
```

```
$a21= change $a21;
print Matrix "a21 = $a21";
print Matrix "return a21;\n";
print Matrix "}\n";
print Matrix "\n";
print Matrix "double funct_A22(double n, double v, double p, double k)\n";
print Matrix "{\n";
print Matrix "double a22;\n";
$a22= change $a22;
print Matrix "a22 = $a22";
print Matrix "return a22;\n";
print Matrix "}\n";
print Matrix "\n";
print Matrix "double funct_A23(double n, double v, double p, double k)\n";
print Matrix "{\n";
print Matrix "double a23;\n";
$a23= change $a23;
print Matrix "a23 = $a23";
print Matrix "return a23;\n";
print Matrix "}\n";
print Matrix "\n";

print Matrix "double funct_A31(double n, double v, double p, double k)\n";
print Matrix "{\n";
print Matrix "double a31;\n";
$a31= change $a31;
print Matrix "a31 = $a31";
print Matrix "return a31;\n";
print Matrix "}\n";
print Matrix "\n";
print Matrix "double funct_A32(double n, double v, double p, double k)\n";
print Matrix "{\n";
print Matrix "double a32;\n";
$a32= change $a32;
```

```
print Matrix "a32 = $a32";
print Matrix "return a32;\n";
print Matrix "}\n";
print Matrix "\n";
print Matrix "double funct_A33(double n, double v, double p, double k)\n";
print Matrix "{\n";
print Matrix "double a33;\n";
$a33= change $a33;
print Matrix "a33 = $a33";
print Matrix "return a33;\n";
print Matrix "}\n";
print Matrix "\n";

close(Matrix);
```

§C. Programas para graficación de datos en Gnuplot

§C.1. grafica2d

```
#!/bin/bash
gnuplot<<EOF
unset key
set terminal jpeg large size 720, 600
set output "N$2.jpg"
set xrange [0:1]
set yrange [0:$3]
set ylabel "numero de particulas"
set xlabel "X"
plot "DATOS$1" using 1:2 with lines

set output "V$2.jpg"
set xrange [0:1]
set yrange [0:$4]
```

```
set ylabel "velocidad"
set xlabel "X"
plot "DATOS$1" using 1:3 with lines

set output "P$2.jpg"
set xrange [0:1]
set yrange [0:$5]
set ylabel "presion"
set xlabel "X"
plot "DATOS$1" using 1:4 with lines

set terminal x11
set size 1, 1
EOF
```

§C.2. grafica3d

```
#!/bin/bash
gnuplot<<EOF
set nokey
set dgrid3d 500,500,4
set view 0,0
set contour base
set style data lines
set cntrparam cubicspline
set cntrparam levels 6
set cntrparam  levels incremental 0.1,0.5
set pm3d map
set samples 400; set isosamples 400
set colorbox horizontal
set colorbox user origin 0.1, 0.05 size 0.8, 0.06
set size square
#set multiplot
```

```
set terminal jpeg large size 720, 600
set output "N$2.jpg"
set cbrange [0.0:11.01]
splot  [0:1] [0:1] [0.0:11] "DATOS$1" using 1:2:3

set output "V$2.jpg"
set cbrange [0:1]
splot  [0:1] [0:1] [0.0:1] "DATOS$1" using 1:2:4

set output "P$2.jpg"
set cbrange [0.0:14]
splot  [0:1] [0:1] [0.0:14]  "DATOS$1" using 1:2:5

set size 1.0,1.0
set size nosquare
EOF
```

§D. Código fuente del programa Aztekas

§D.1. programa principal

Código fuente del archivo planar.c

```
int main(void)
{
  ReadData(&par);//Lee archivo input y guarda los datos en una estructura
  double dt, dx, time, tmax, dtprint, tprint;
  int mesh = par.mesh;
  double U1[mesh+1];
  double U2[mesh+1];
  double U3[mesh+1];
  int itprint;

   //Establece parametros iniciales y cada cuando imprimir en pantalla
```

```
 //la evolucion de los datos
  INITFLOW( U1, U2, U3, &dx, &dtprint, &tmax, &par);
  tprint  = 0.0;
  itprint = 1;   //inicializa la numeracion de los archivos datos de salida
  time    = 0.0;
  dt      = 0.0;

 while(time <= tmax ){

     if(time >= tprint){
printf("Time:%.10f ,dt: %f, dx:%f dt/dx:%f \n", time, dt, dx,(dt/dx));

      if(par.dimension==3)
      {
      //Guarda datos en archivos para pelicula
         OUTPUT(U1,U2,U3,&dx,&itprint, &par);
      }

      if(par.dimension==2)
      {
      //Guarda datos en archivos para grafica
       OUTPUT_2(U1,U2,U3,&dx,&itprint, &par);
      }

tprint = tprint+dtprint;
++itprint;
     }
     dt   = par.dt;
     time = time+dt;

   //Integracion de T a T+DT
   TSTEP(U1, U2, U3, &dx, &dt, &time, &par);

//En la FRONTERA, el ultimo dato es sustituido
```

```
     //por el inmediato anterior
     U1[mesh] = U1[mesh-1];
     U2[mesh] = U2[mesh-1];
     U3[mesh] = U3[mesh-1];
 }

    file = fopen("GRAFICAS", "a");
    fprintf(file,"%d\n", itprint);
    fclose(file);
 return 0;
}
```

§D.2. Funciones

Código fuente del archivo de funciones func_planar.c

```
/*
 *   This program is free software: you can redistribute it and/or modify
 *   it under the terms of the GNU General Public License as published by
 *   the Free Software Foundation, either version 3 of the License, or
 *   (at your option) any later version.
 *
 *   This program is distributed in the hope that it will be useful,
 *   but WITHOUT ANY WARRANTY; without even the implied warranty of
 *   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 *   GNU General Public License for more details.
 *
 *   You should have received a copy of the GNU General Public License
 *   along with this program.  If not, see <http://www.gnu.org/licenses/>.
 */
#include<stdio.h>
#include<math.h>
#include<string.h>
#include<gsl/gsl_matrix.h>
#include "func_planar.h"
```

```
#include "func_matrix.h"

int OUTPUT(double *u1, double *u2, double *u3, double *dx, int *itprint, par_ *par)
{
 FILE *file;
 int  i, j;
 char dat[4];
 char archivo[9];
 int  loop, num;
 double y;
 loop = par->mesh;
 num  = *itprint;
 snprintf(dat,4,"%d",num);
 strcpy(archivo, "DATOS");
 strcat(archivo, dat);
 file = fopen(archivo, "w");
 y = 0.0;
 for(j = 0; j <= 100; j++)
  {
   for(i = 0; i <= loop; i++)
       {
         fprintf(file, "%f %f  %f  %f  %f \n", (i)*(*dx), y, u1[i],u2[i],u3[i]);
       }
       y = y + 0.01;
  }
fclose(file);
return 0;
}


int OUTPUT_2(double *u1,double *u2, double *u3, double *dx, int *itprint, par_ *par)
{
 FILE *file;
 int i;
```

```
 char dat[4];
 char archivo[9];
 int num, loop;
 loop = par->mesh;
 num  = *itprint;
 snprintf(dat,4,"%d",num);
 strcpy(archivo, "aztDATOS");
 strcat(archivo, dat);
 file = fopen(archivo, "w");
   for(i = 0; i <= loop; i++)
       {
         fprintf(file, "%f %f %f %f \n", (i)*(*dx), u1[i], u2[i], u3[i]);
       }
fclose(file);
return 0;
}

int FLUXCORRECTOR(double *U1, double *U2, double *U3, par_ *par)
{
  int i;
  int  mesh = par->mesh;
  double UC_1[mesh+1];
  double UC_2[mesh+1];
  double UC_3[mesh+1];
  double eta;
  double test1, test2, test3;

  for (i=0; i<=mesh; i++)
  {
    test1 = ( (U1[i+1]-U1[i]) * (U1[i]-U1[i-1]) );
    if(test1 < 0)
      eta = par->difusion;
    else
      eta = 0.0;
```

```
     UC_1[i] = U1[i] + eta*( U1[i+1]+U1[i-1]-(2.0*U1[i]));

    test2 = ( (U2[i+1]-U2[i]) * (U2[i]-U2[i-1]));
    if( test2 < 0 )
     eta = par->difusion;
    else
     eta=0.0;

     UC_2[i] = U2[i] + eta *( U2[i+1]+U2[i-1]-( 2.0*U2[i] ) );

    test3=((U3[i+1]-U3[i]) * (U3[i]-U3[i-1]));
    if( test3 < 0 )
     eta=par->difusion;
    else
     eta=0.0;

    UC_3[i]= U3[i] + eta *( U3[i+1]+U3[i-1]-(2.0*U3[i]));
  }

  for(i=0; i<=mesh; i++)
  {
    U1[i] = UC_1[i];
    U2[i] = UC_2[i];
    U3[i] = UC_3[i];
  }

return 0;
}

int TSTEP(double *U1, double *U2, double *U3, double *dx, double *dt, double *time, par
{
  double dtx;
  int i, im, ip;
```

```
 double F1, F2, F3, derivada;
 int  mesh = par->mesh;
 double frec = par->frec;
 double UP_1[mesh+1];
 double UP_2[mesh+1];
 double UP_3[mesh+1];
 double UC_1[mesh+1];
 double UC_2[mesh+1];
 double UC_3[mesh+1];
 gsl_matrix *M = gsl_matrix_alloc(3,3);
 gsl_vector *a = gsl_vector_alloc(3);
 gsl_vector *F = gsl_vector_alloc(3);

 dtx = (*dt)/(*dx);
for (i = 0; i <= mesh; i++)
{
  ip = i+1;
  im = i-1;

  F1 = U1[i] - U1[im]; //restas que se utilizan en el calculo de la derivada
  F2 = U2[i] - U2[im];
  F3 = U3[i] - U3[im];

  if(i==0)   //derivada a la derecha
  {
     F1 = U1[ip] - U1[i];
     F2 = U2[ip] - U2[i];
     F3 = U3[ip] - U3[i];
  }
  if(i==mesh)   //derivada a la izquierda
  {
     F1 = U1[i] - U1[im];
     F2 = U2[i] - U2[im];
     F3 = U3[i] - U3[im];
```

```
  }

  gsl_vector_set(F,0,F1);
  gsl_vector_set(F,1,F2);
  gsl_vector_set(F,2,F3);
  MATRIX(U1,U2,U3,M,i);
  //Metodo de MacCormack
  //*********************************************
  //PREDICTOR
  //*********************************************
  gsl_matrix_get_row(a,M,0);
  gsl_vector_mul( a, F);  //{M11[ U1(x+dx,t)-U1(x-dx,t) ]+M12[U2()...] + M13[U3()...]
  gsl_vector_scale(a,dtx);  // = dt/dx * {M11[...]+M12[...]+M13[...] }
  derivada =  (gsl_vector_get(a,0) + gsl_vector_get(a,1) + gsl_vector_get(a,2));
  UP_1[i]  = ( U1[i] - (derivada) );          //      <-------  U1=n

  gsl_matrix_get_row(a,M,1);
  gsl_vector_mul(a,F);   //{M21[ U1(x+dx,t)-U1(x-dx,t) ]+ M22[U2()...] + M23[(U3)...]
  gsl_vector_scale(a,dtx);
  derivada =  (gsl_vector_get(a,0)+ gsl_vector_get(a,1)+ gsl_vector_get(a,2));
  UP_2[i]  = ( U2[i] - (derivada) ); //     <-------------  U2=VELOCIDAD

  gsl_matrix_get_row(a,M,2);
  gsl_vector_mul(a,F);
  gsl_vector_scale(a,dtx);
  derivada =  (gsl_vector_get(a,0)+ gsl_vector_get(a,1)+ gsl_vector_get(a,2));
  UP_3[i]  = ( U3[i] - (derivada) ); //<---------------  U3=PRESION
 }

for (i = 0; i<=mesh; i++)
{
  ip = i+1;
  im = i+1;
```

```
F1 = UP_1[ip]-UP_1[i]; //restas que se utilizan en el calculo de la derivada
F2 = UP_2[ip]-UP_2[i];
F3 = UP_3[ip]-UP_3[i];

if(i==0)   //derivada a la derecha
{
   F1 = UP_1[ip]-UP_1[i];
   F2 = UP_2[ip]-UP_2[i];
   F3 = UP_3[ip]-UP_3[i];
}
if(i==mesh)   //derivada a la izquierda
{
   F1=UP_1[i]-UP_1[im];
   F2=UP_2[i]-UP_2[im];
   F3=UP_3[i]-UP_3[im];
}

gsl_vector_set(F,0,F1);
gsl_vector_set(F,1,F2);
gsl_vector_set(F,2,F3);
MATRIX(U1,U2,U3,M,i);

//Metodo de MacCormack
//*********************************************
//CORRECTOR
//*********************************************
gsl_matrix_get_row(a,M,0);
gsl_vector_mul(a,F);  //{M11[ U1(x+dx,t)-U1(x-dx,t) ]+M12[U2()...] + M13[U3()...]  }
gsl_vector_scale(a,dtx);  // = dt/dx * {M11[...]+M12[...]+M13[...] }
derivada  = (gsl_vector_get(a,0) + gsl_vector_get(a,1) + gsl_vector_get(a,2));
  UC_1[i] = 0.5 * ( U1[i] + UP_1[i] -(derivada) );

gsl_matrix_get_row(a,M,1);
gsl_vector_mul(a,F);   //{M21[ U1(x+dx,t)-U1(x-dx,t) ]+ M22[U2()...] + M23[(U3)...]  }
```

```
   gsl_vector_scale(a,dtx);
   derivada = (gsl_vector_get(a,0)+ gsl_vector_get(a,1)+ gsl_vector_get(a,2));
   UC_2[i]  = 0.5 * ( U2[i] + UP_2[i] - (derivada) );

   gsl_matrix_get_row(a,M,2);
   gsl_vector_mul(a,F);
   gsl_vector_scale(a,dtx);
   derivada = (gsl_vector_get(a,0)+ gsl_vector_get(a,1)+ gsl_vector_get(a,2));

   UC_3[i]  = 0.5 * ( U3[i] + UP_3[i] - (derivada) );

 }
 FLUXCORRECTOR( UC_1, UC_2, UC_3, par);

  //copiar las UP sobre las U
  for(i=0; i<=mesh; i++)
  {
    U1[i] = UC_1[i];
    U2[i] = UC_2[i];
    U3[i] = UC_3[i];
  }

gsl_matrix_free(M);
gsl_vector_free(a);
gsl_vector_free(F);

return 0;
}

int MATRIX(double *U1, double *U2, double *U3, gsl_matrix *A, int i)
{

 double k;
 k = (5.0/3.0);
```

```
 double n, v, p;
 double A11, A12, A13, A21,  A22, A23, A31, A32, A33;
 n = U1[i];
 v = U2[i];
 p = U3[i];

 A11 = funct_A11(n,v,p,k);
 A12 = funct_A12(n,v,p,k);
 A13 = funct_A13(n,v,p,k);

 A21 = funct_A21(n,v,p,k);
 A22 = funct_A22(n,v,p,k);
 A23 =funct_A23(n,v,p,k);

 A31 = funct_A31(n,v,p,k);
 A32 =funct_A32(n,v,p,k);
 A33 = funct_A33(n,v,p,k);

 gsl_matrix_set(A,0,0, A11 );
 gsl_matrix_set(A,0,1, A12 );
 gsl_matrix_set(A,0,2, A13 );

 gsl_matrix_set(A,1,0, A21 );
 gsl_matrix_set(A,1,1, A22 );
 gsl_matrix_set(A,1,2, A23 );

 gsl_matrix_set(A,2,0, A31 );
 gsl_matrix_set(A,2,1, A32 );
 gsl_matrix_set(A,2,2, A33 );
return 0;

}

int INITFLOW(double *U1, double *U2, double *U3, double *dx,double *dtprint, double *tmax
```

```
{
 int i;
 double xmax;

//Size of the mesh
 xmax = 1.0;

*dx = xmax/(par->mesh);

 printf("\nEstablecemos las condiciones iniciales en el tiempo t=0\n");

//CONDICIONES INICIALES
//DENSIDAD
//Primera mitad
 for(i=0; i<=((par->mesh)/2)-1; i++){
     U1[i] = par->n1;
 }
//Segunda mitad
 for(i=((par->mesh)/2); i<=(par->mesh); i++){
     U1[i] = par->n2;
 }
//-------------------------------------------
//VELOCIDAD
//Primera mitad
 for(i=0; i<=((par->mesh)/2)-1; i++){
  U2[i] = par->v1;
 }
//Segunda mitad
 for(i= ((par->mesh)/2); i<=(par->mesh); i++){
  U2[i] = par->v2;
 }
//-------------------------------------------
 //PRESION
//Primera mitad
```

```
for(i=0; i<=((par->mesh)/2)-1; i++){
  U3[i] = par->p1;
 }
//Segunda mitad
 for(i=((par->mesh)/2); i<=(par->mesh); i++){
  U3[i] = par->p2;
 }

printf("U1 = numero de particulas n= %f \n",U1[0]);
printf("U2 = velocidad = %f\n", U2[0]);
printf("U3 = Presion p = %f\n\n", U3[0]);

 *tmax    = par->tmax;     //Establece el tiempo que dura la simulacion
 *dtprint = par->timefile; //Establece cada cuando escribe datos a pantalla

 return 0;
}

int ReadData(par_ *par)
{
 FILE *file;
 int data;
 float data_f;
 char test[15];
 file = fopen("INPUT", "r");

 fscanf(file,"%s %d", test, &data);
 par->mesh = data;

 fscanf(file,"%s %f", test, &data_f);
 par->n1 = data_f;

 fscanf(file,"%s %f", test, &data_f);
 par->n2 = data_f;
```

```
 fscanf(file,"%s %f", test, &data_f);
 par->v1 = data_f;

 fscanf(file,"%s %f", test, &data_f);
 par->v2 = data_f;

 fscanf(file,"%s %f", test, &data_f);
 par->p1 = data_f;

 fscanf(file,"%s %f", test, &data_f);
 par->p2 = data_f;

 fscanf(file,"%s %f", test, &data_f);
 par->tmax = data_f;

 fscanf(file,"%s %f", test, &data_f);
 par->timefile = data_f;

 fscanf(file,"%s %f", test, &data_f);
 par->dt = data_f;

 fscanf(file,"%s %f", test, &data_f);
 par->frec = data_f;

 fscanf(file,"%s %f", test, &data_f);
 par->difusion = data_f;

fscanf(file,"%s %d", test, &data);
 par->dimension = data;


 fclose(file);
 return 0;
```

```
}
```

§E. GNU General Public License

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program–to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

Terms and Conditions

0. Definitions.

   "This License" refers to version 3 of the GNU General Public License.

   "Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

   "The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

   To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

   A "covered work" means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

   The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

   A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

   The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

   The “Corresponding Source” for a work in object code form means all the source code

needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

   You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

   You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

   You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

   a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

   b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".

   c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its

parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally

and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning

of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered

work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

   You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

   However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

   Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

   Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

   You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted,

prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given

local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

End of Terms and Conditions

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>

Copyright (C) <textyear>  <name of author>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
```

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program>  Copyright (C) <year>  <name of author>

This program comes with ABSOLUTELY NO WARRANTY; for details type ‘show w’.
This is free software, and you are welcome to redistribute it
under certain conditions; type ‘show c’ for details.
```

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see `http://www.gnu.org/licenses/`.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read `http://www.gnu.org/philosophy/why-not-lgpl.html`.

Bibliografía

Blandford, R. D. & Konigl, A., 1979. Relativistic jets as compact radio sources. *Astrophys. J.*, **232**, 34.

Blandford, R. D. & McKee, C. F., 1976. Fluid dynamics of relativistic blast waves. *Physics of Fluids*, **19**, 1130–1138.

Book, D. L., Boris, J. P. & Hain, K., 1975. Flux-Corrected Transport. II - Generalizations of the method. *Journal of Computational Physics*, **18**, 248–283.

Cantó, J., Raga, A. C. & D'Alessio, P., 2000. Analytic solutions to the problem of jets with time-dependent injection velocities. *Monthly Notices of the Royal Astronomical Society*, **313**, 656–662.

Chung, T. J., 2002. *Computational Fluid Dynamics*. Cambridge University Press.

Daigne, F. & Mochkovitch, R., 1998. Gamma-ray bursts from internal shocks in a relativistic wind: temporal and spectral properties. *Monthly Notices of the Royal Astronomical Society*, **296**, 275–286.

Falle, S. A. E. G. & Raga, A. C., 1993. The structure of knots in variable stellar jets. I – Symmetric knots. *Monthly Notices of the Royal Astronomical Society*, **261**, 573–583.

Falle, S. A. E. G. & Raga, A. C., 1995. The structure of knots in variable stellar jets-II. Asymmetric knots. *Monthly Notices of the Royal Astronomical Society*, **272**, 785–799.

Hagen-Thorn, V. A. et al., 2007. The Outburst of the Blazar AO 0235+164 in 2006 December: Shock-in-Jet Interpretation.

Hidalgo, J. C. & Mendoza, S., 2005. Self-similar imploding relativistic shock waves. *Physics of Fluids*, **17**, 6101–+.

Hughes, P. A., Aller, H. D. & Aller, M. F., 1985. Polarized Radio Outbursts in Bl-Lacertae - Part Two - the Flux and Polarization of a Piston-Driven Shock. *Astrophysical Journal*, **298**, 301–+.

Landau, L. D. & Lifshitz, E. M., 1987. *Fluid Mechanics, Course on theoretical Physics V.6*. Pergamon Press, London, 2nd ed.

Landau, L. D. & Lifshitz, E. M., 1994. *The Classical Theory of Fields, Course on theoretical Physics V.2*. Pergamon Press, London, 4th ed.

Laney, C. B., 1998. *Computational Gasdynamics*. Cambridge University Press.

MacCormack, R. W. & Paullay, A. J., 1972. Computational efficiency achieved by time splitting of finite difference operators. . American Institute of Aeronautics and Astronautics AIAA paper-1972-154.

Marscher, A. P., 1996. Variability of the Non-thermal Emission in the Jets of Blazars. In *Blazar Continuum Variability*, Astronomical Society of the Pacific Conference Series.

Martí, J. M. & Müller, E., 2003. Numerical Hydrodynamics in Special Relativity. *Living Reviews in Relativity*, **6**(7). URL `http://www.livingreviews.org/lrr-2003-7`.

McKee, C. R. & Colgate, S. A., 1973. Relativistic Shock Hydrodynamics. *Astrophysical Journal*, **181**, 903–938.

Mendoza, S., 2000. *Shocks and Jets in Radio Galaxies and Quasars*. Ph.D. thesis, Cavendish Laboratory, Cambridge University U.K., available at `http://www.mendozza.org/sergio/phdthesis`.

Mendoza, S. & Longair, M. S., 2001. Deflection of jets induced by jet-cloud and jet-galaxy interactions, astro-ph/0008015. *Monthly Notices of the Royal Astronomical Society*, **324**, 149.

Mendoza, S. & Longair, M. S., 2002. Formation of internal shock waves in bent jets, astro-ph/0109125. *Monthly Notices of the Royal Astronomical Society*, **331**, 323–332.

Panaitescu, A., Spada, M. & Mészáros, P., 1999. Power Density Spectra of Gamma-Ray Bursts in the Internal Shock Model. *Astrophysical Journal*, **522**, L105–L108.

Press, W. H. & Teukolsy, S. A., 1992. *Numerical Recipes*. Cambridge University Press, 2nd ed.

Raga, A. C., Beck, T. & Riera, A., 2004. Interpreting the Observations of Herbig-Haro Jets. *Astrophysics and Space Science*, **293**, 27–36.

Raga, A. C., Binette, L., Canto, J. & Calvet, N., 1990. Stellar jets with intrinsically variable sources. *Astrophysical Journal*, **364**, 601–610.

Raga, A. C. & Kofman, L., 1992. Knots in stellar jets from time-dependent sources. *Astrophysical Journal*, **386**, 222–228.

Rees, M. J., 1966. Apearance of Relativistically Expanding Radio Sources. *Nature*, **211**, 468.

Rees, M. J., 1978. The M87 jet - Internal shocks in a plasma beam. *Monthly Notices of the Royal Astronomical Society*, **184**, 61P–65P.

Rees, M. J. & Meszaros, P., 1994. Unsteady outflow models for cosmological gamma-ray bursts. *Astrophysical Journal*, **430**, L93–L96.

Sahayanathan, S. & Misra, R., 2005. Interpretation of the Radio/X-Ray Knots of AGN Jets within the Internal Shock Model Framework. *Astrophysical Journal*, **628**, 611–616.

Spada, M., Ghisellini, G., Lazzati, D. & Celotti, A., 2001. Internal shocks in the jets of radio-loud quasars. *Monthly Notices of the Royal Astronomical Society*, **325**, 1559–1570.

Stone, J. M. & Norman, M. L., 1992. ZEUS-2D: A radiation magnetohydrodynamics code for astrophysical flows in two space dimensions. I - The hydrodynamic algorithms and tests. *Astrophysical Journal*, **80**, 753–790.

Taub, A. H., 1967. Relativistic Hydrodynamics. In *Relativity Theory and Astrophysics. Vol.1: Relativity and Cosmology*, 170–+.

Tooper, R. F., 1965. Adiabatic Fluid Spheres in General Relativity. *Astrophysical Journal*, **142**, 1541–+.

Toro, E. F., 1999. *Riemann Solvers and Numerical Methods for Fluid Dynamics*. Springer, 2nd ed.