ESTIMACIÓN DE VELOCIDADES MEDIANTE TOMOGRAFÍA BASADA EN RAYOS SÍSMICOS.

VELOCITY ESTIMATION USING SEISMIC RAY TOMOGRAPHY.

CRISTHIAN DARIO ZULUAGA HERRRERA

Requisito parcial para optar al título de Maestría en Ingeniería

Asesor:
Hebert Montegranario Riascos

Universidad de Antioquia
Facultad de Ingeniería.

2018

Agradecimentos

Quiero agradecer sinceramente a aquellas personas que compartieron sus conocimientos conmigo para hacer posible la conclusión de esta tesis. Especialmente agradezco a Juan Carlos Muñoz por brindarme su apoyo y conocimiento en esta investigación. Gracias al profesor Hebert Montegranario por sus recomendaciones. Gracias a la Universidad de Antioquia por darme la posibilidad de formarme en mis estudios de maestría.

Este trabajo de tesis es apoyado por la Empresa Colombiana de Petróleo ECOPETROL y COLCIENCIAS como parte de la beca del proyecto de investigación No. 0266-2013.

Índice general

Resumen

En esta trabajo se presenta un trazador de rayos el cual es bastante versátil y que posee características de la teoría de rayos importantes a la hora de hacer tomografía, ya que permite calcular trayectorias de rayos reflejados y refractados simultáneamente, generar rayos con varias fuentes y varios receptores, escoger puntos en interfaces de interés para generar reflexiones en dichos puntos, entre otras. El trazador está basado en el método del camino más corto y fundamentado en el principio de Fermat, en donde el camino del rayo a lo largo del espacio posee un tiempo de propagación que es estacionario [Moser 1991]. A partir de este modelado, se implementó una tomografía de tiempos de propagación usando las técnicas de reconstrucción algebraica (ART y SIRT), para estimar velocidades de propagación de ondas en el subsuelo. Se realizaron experimentos para resolver sistemas de ecuaciones lineales $Gm = d$, donde G es mal condicionada y *sparse*. También se realizaron experimentos donde se aplicaron el modelado y las técnicas de reconstrucción algebraica en Inversión Basada en Modelos sintéticos.

Abstract

In this work we present a ray tracer which is quite versatile and has characteristics of the ray theory important when doing tomography, since it allows to calculate trajectories of reflected and refracted rays simultaneously, generate

rays with several sources and several receivers, choose points in interest interfaces to generate reflections in said points, among others. The tracer is based on the shortest path method and based on the Fermat principle, where the ray path along the space has a taveltime that is stationary [Moser 1991]. From this modeling, a traveltime tomography was implemented using the algebraic reconstruction techniques (ART and SIRT), to estimate waves velocities in the subsurface. Experiments were performed to solve systems of linear equations $Gm = d$, where G is ill conditioned matrix and sparse. Also, experiments were also performed where modeling and algebraic reconstruction techniques were applied in Inversion Based on synthetic models.

Introducción

La exploración petrolera usa la generación de imágenes del subsuelo para mejorar sus índices de éxitos en proyectos de exploración y delineación de yacimientos. En áreas estructuralmente complejas, las imágenes en escala de profundidad revelan las características del subsuelo, lo que convierte la técnica de imágenes sísmicas en una herramienta bastante utilizada por las compañías de gas y petróleo, para la ubicación de nuevas reservas en regiones con geología compleja. Las aplicaciones de esta técnica se encuentran desde la exploración petrolera, hasta otros campos del conocimiento como la Ingeniería Civil, donde se utiliza para explorar los tipos de suelos donde se planea construir estructuras civiles [Hernández y cols. 2006].

La tomografía es la obtención de imágenes de cortes o secciones de algún objeto. Para este trabajo, se obtendrán modelos de velocidades a partir de datos sintéticos, utilizando la técnica de trazado de rayos y tomografía basada en técnicas de reconstrucción algebraica. Este es un tipo de proceso denominado en matemáticas como Problema Inverso, el cual consiste en la obtención de un conjunto de parámetros de un modelo a través de la interpretación de datos observados o mediciones que en su mayoría son inexactos y poseen ruido [Jones 2010]. Durante los últimos años, la tomografía basada en rayos ha tenido gran evolución gracias a las nuevas herramientas de cómputo que día a día procesan gran cantidad de datos en un tiempo cada vez menor [Castillo y cols. 2013], [Zhang y cols. 2017], [Zhang y cols. 2018],

[Meléndez y cols. 2015]. Los *clúster*, un ejemplo de ello, son un conglomerado de computadoras que tienen un hardware preferiblemente homogéneo y que se comportan como si fueran una única máquina, lo cual los convierte en una herramienta eficiente cuando se procesan los datos observados [Castillo y cols. 2013], [Malony y cols. 2016b], [Monil y cols. 2018].

En la prospección sísmica, especialmente en la exploración petrolera, se necesita saber el camino que siguen los rayos sísmicos en el subsuelo, para determinar los tiempos de propagación de una onda entre fuentes y receptores, teniendo en cuenta las diferentes capas geológicas presentes. El método que se describe en este trabajo, es el de refracción y reflexión de una onda elástica utilizando una aproximación en altas frecuencias, a partir del método del camino más corto en un conjunto de nodos [Moser 1991], se encuentran los tiempos de propagación entre las fuentes y los receptores. Utilizando el trazado de rayos se puede construir un modelo apropiado de velocidades sísmicas a partir de un modelo inicial. Luego, las velocidades sísmicas iniciales del modelo son refinadas, con un funcional de optimización adecuado, con la finalidad de obtener un mejor ajuste entre los tiempos de propagación medidos y calculados [Rodríguez-Pérez 2007], [Huang y cols. 2017], [Fu y Hanafy 2017].

Luego de obtener los tiempos de propagación, usando algún trazador de rayos, se procede a trabajar el problema de la tomografía sísmica. Este problema se reduce a un problema de optimización [Nolet 2012]. La tomografía sísmica está formulada como un algoritmo iterativo que produce un modelo de velocidad en profundidad, el cual minimiza la diferencia entre los tiempos de propagación generados por el trazado de rayos, dados por el modelo inicial tomado, y los tiempos de propagación medidos de los datos [Bishop y cols. 1985]. A partir del ajuste del modelo, se obtiene la distribución de velocidades aproximada de la porción del subsuelo estudiada.

En este trabajo se aborda el problema del trazado de rayos y su uso en la tomografía sísmica. Se muestra como se usa el método del camino más corto

para construir de forma eficiente el trazado de rayos para diferentes medios, especialmente en aquellos con variación lineal en profundidad, y en el cual se implementaron rayos generados por refracción y rayos reflejados en puntos específicos del medio estudiado. Además, la inclusión de múltiples fuentes que mejoran la iluminación en la región de interés. Se presenta una estrategia de paralelización usando MPI que permite acelerar la implementación del trazado de rayos, y que permite facilidades computacionales en problemas de mayor alcance. Para la obtención de estimativos de velocidades, se usó la técnica de reconstrucción algebraica SIRT "Simultaneous Iterations Reconstruction Technique", y para la cual se verificó la efectividad en la solución numérica de sistemas de ecuaciones lineales que involucran matrices rectangulares, *sparse* y mal condicionadas, comunes en las aplicaciones de tomografía sísmica. Además, se muestran los resultados obtenidos usando esta técnica en la inversión basada en modelos para la reconstrucción de velocidades en algunos modelos sintéticos.

El trazador implementado que permite calcular trayectorias de rayos por refracción y de reflexiones en puntos de interés, simultáneamente, ha sido usado por otros autores como [Meléndez y cols. 2015] y [Zhang y cols. 2018], pero no se encontró en la literatura el uso de las técnicas de reconstrucción algebraica en tomografía de tiempos de propagación usando un trazador de este tipo, por lo cual el modelado que se implementó y su aplicación se convierte en un aspecto novedoso dentro de tomografía sísmica.

Capítulo 1

Marco Teórico

En este capítulo se abordarán los conceptos y la teoría necesaria para el desarrollo de este trabajo. Primero una introducción al problema Geofísico y la deducción de las ecuaciones que brindan soluciones aproximadas a la ecuación de onda. Luego, el método del camino más corto y el algoritmo de Dijkstra como elementos fundamentales en la construcción del trazador de rayos tanto refractados y como reflejados. Un estado del arte con una visión general de lo que se ha trabajado en trazado de rayos. Por último, los problemas inversos y el desarrollo de las técnicas de reconstrucción algebraica adaptados a la tomografía sísmica.

1.1. Problema geofísico

La geofísica estudia, entre otras cosas, la estructura de la tierra y su dinámica, sobre la base de medidas de campos físicos que normalmente se realizan desde la superficie del planeta. El objetivo es deducir las propiedades del subsuelo a través del uso de modelos físicos que describen los diferentes fenómenos que ocurren en el subsuelo.

Una de las ramas de la geofísica es la sismología, la cual se encarga del

estudio de terremotos y la propagación de las ondas mecánicas (ondas sísmicas) que se generan en el interior de la tierra. Nuestro interés está en la propagación de las ondas sísmicas en el subsuelo y el comportamiento de éstas cuando atraviesan diferentes capas geológicas del interior de la tierra, las cuales se han depositado en las diferentes eras geológicas. Este estudio se realiza a partir de los métodos de sismología de reflexión y refracción; ambos métodos son usados para encontrar rasgos en el subsuelo a partir de una fuente de ondas sísmicas (dinamita, vibro sismos (Exploración *on-shore*), disparos en el agua (Exploración *off-shore*), etc.), y un tendido de receptores ("sensores") que registran las perturbaciones provenientes de la propagación de estas ondas.

Para estudiar la propagación de las ondas en un medio complejo, las aproximaciones mas comunes en la investigación de ondas sísmicas son: a) métodos basados en las soluciones numéricas directas de la ecuación elastodinámica, tales como los métodos diferencias finitas y elementos finitos [Vidale 1988], y b) métodos de aproximación asintótica en altas frecuencias, o lo que se conoce como trazado de rayos [Cerveny 2005]. Ambos métodos son muy usados para resolver cierto tipo de problemas sísmicos, y tienen sus propias ventajas y desventajas.

En la prospección sísmica uno de los principales problemas es generar imágenes del subsuelo a partir de un modelo matemático para la propagación de las ondas y la formulación de un problema inverso que conduce a la reconstrucción tomográfica. En la etapa del modelado, se construyen modelos sintéticos sobre la propagación de la onda en el subsuelo y se generan los datos calculados o datos sintéticos d_{calc}, para después compararlo con el dato observado d_{obs} (sismogramas) en busca de un modelo que minimice la diferencia entre d_{obs} y d_{calc}, a esto se le conoce como Inversión Basada en Modelos [Sen 2006]. La inversión de datos geofísicos es un problema difícil de resolver pues los datos están invariablemente contaminados por ruido y se adquieren en un número finito de puntos. Por otra parte, los modelos matemáticos que se

usan son complejos, y al mismo tiempo, son simplificaciones de los verdaderos procesos geofísicos. En consecuencia, las soluciones son ambiguas y propensas a errores.

1.2. Ecuación de onda y ecuación de rayo

La ecuación de onda con velocidad escalar variable está dada por

$$\nabla^2\psi - \frac{1}{v^2(\vec{x})}\frac{\partial^2\psi}{\partial t^2} = 0, \tag{1.1}$$

donde ψ es una función que satisface la ecuación y v es la velocidad de propagación de la onda. Para esta ecuación se puede suponer una solución aproximada en la forma de onda plana

$$\psi(\vec{x},t) = A(\vec{x})e^{2\pi i f(t-T(\vec{x}))}. \tag{1.2}$$

Donde $A(\vec{x})$ y $T(\vec{x})$ son funciones desconocidas que describen la amplitud y el tiempo de propagación del frente de onda cuando se varia la posición, respectivamente. Sustituyendo la ecuación (1.2) en la ecuación (1.1) y calculando el laplaciano de la solución supuesta, se tiene que

$$\nabla^2\psi(\vec{x},t) = \vec{\nabla}\cdot\vec{\nabla}\left[A(\vec{x})e^{2\pi i f(t-T(\vec{x}))}\right]$$
$$\vec{\nabla}\cdot\vec{\nabla}\left[A(\vec{x})e^{2\pi i f(t-T(\vec{x}))}\right] = \vec{\nabla}\cdot\left[e^{2\pi i f(t-T(\vec{x}))}\vec{\nabla}A(\vec{x}) - 2\pi i A(\vec{x})e^{2\pi i f(t-T(\vec{x}))}\vec{\nabla}T\right], \tag{1.3}$$

Esta ecuación puede expandirse como

$$\nabla^2\psi(\vec{x},t) = \left\{\nabla^2 A - 4\pi i f\vec{\nabla}A\cdot\vec{\nabla}T - 4\pi^2 f^2 A\left[\vec{\nabla}T\right]^2 - 2\pi i f A\nabla^2 T\right\}e^{2\pi i f(t-T(\vec{x}))}. \tag{1.4}$$

Usando este resultado y $\partial_t^2\psi(\vec{x},t) = -4\pi^2 f^2 A e^{2\pi i f(t-T(\vec{x}))}$ en la ecuación (1.1) e igualando las partes real e imaginaria dadas de las dos ecuaciones, se obtiene

$$\left[\vec{\nabla}T\right]^2 - \frac{\nabla^2 A}{4\pi^2 f^2 A} - \frac{1}{v(\vec{x})^2} = 0 \tag{1.5}$$

y

$$\frac{A}{2}\nabla^2 T - \vec{\nabla}A \cdot \vec{\nabla}T = 0. \tag{1.6}$$

Sin embargo, se espera que el segundo término de la ecuación (1.5) sea insignificante con gradientes de velocidad débiles o cuando las frecuencias son altas, independientemente del gradiente de velocidad. Cuando este término es descartado el resultado es una ecuación diferencial parcial no lineal llamada *Ecuación Eikonal*

$$\left[\vec{\nabla}T(\vec{x})\right]^2 - \frac{1}{v(\vec{x})^2} = 0. \tag{1.7}$$

Aunque no es exacta, para muchas situaciones realistas, una solución a la ecuación Eikonal da tiempos de propagación precisos para un medio complejo. La ecuación diferencial para los caminos del rayo es la ecuación vectorial que está implícita en la ecuación Eikonal (1.7). Para un medio isótropo, los rayos son normales al frente de onda. Un frente de onda se describe como una superficie donde $T(\vec{x})$ es constante. Por consiguiente, $\vec{\nabla}T$ debe ser un vector de puntos en la dirección del rayo y la ecuación Eikonal muestra que $\left|\vec{\nabla}T\right| = v^{-1}$ (De forma geométrica, el gradiente es un vector que se encuentra normal a la curva de nivel en el punto que se está estudiando en el frente de onda).

Considere los frentes de onda en T_1 y T_2 donde $T_1 - T_2$ es muy pequeño, y los puntos P_1 y P_2 como en la figura (1.1). Entonces $\Delta s = (T_2(P_2) - T_1(P_1))\cdot v(P_1)$ es un estimativo de la distancia perpendicular entre estos frentes de onda, y

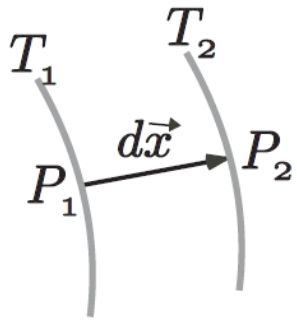


Figura 1.1: El elemento diferencial del rayo $d\vec{x}$ que se extiende desde un punto P_1 en un frente de onda en el tiempo T_1 hasta el punto P_2 en un frente de onda en el tiempo T_2. El segmento del rayo es perpendicular a ambos frentes de onda y tiene longitud $|d\vec{x}| = \Delta s$.

$$\vec{\nabla} T(P_1) = \lim_{P_2 \to P_1} \frac{T_2(P_2) - T_1(P_1)}{\Delta s} \hat{s} = \frac{\hat{s}}{v(P_1)} \tag{1.8}$$

Donde $\hat{s}$ es un vector unitario que es normal a la superficie $T_1(P_1)$, o en otras palabras, $\hat{s}$ son los puntos a lo largo del rayo. Si $d\vec{x}$ es el vector diferencial de P_1 a P_2 entonces $\hat{s}$ puede ser escrito como

$$\hat{s} = \frac{d\vec{x}}{ds} \tag{1.9}$$

donde $ds = |d\vec{x}|$ y s es la longitud de arco a lo largo del rayo. Combinando la ecuaciones (1.8) y (1.9) se tiene la ecuación de rayo

$$\vec{\nabla} T(\vec{x}) = \frac{\hat{s}}{v(\vec{x})} = \frac{1}{v(\vec{x})} \frac{d\vec{x}}{ds} \tag{1.10}$$

El tiempo de propagación T puede ser eliminado de la ecuación (1.10), calculando su derivada con respecto a su longitud de arco, esto es

$$\frac{d\vec{\nabla} T}{ds} = \frac{d}{ds} \frac{1}{v(\vec{x})} \frac{d\vec{x}}{ds} \tag{1.11}$$

El lado izquierdo de la ecuación (1.11) puede ser simplificado como

$$\frac{d\vec{\nabla}T}{ds} = \vec{\nabla}\left[\frac{dT}{ds}\right] = \vec{\nabla}\left[\frac{1}{v(\vec{x})}\right] \tag{1.12}$$

El ultimo paso en esta ecuación se justifica usando la ecuación (1.10) y la identidad $d/ds = \hat{s}\cdot\vec{\nabla}$. Intuitivamente, este paso es valido porque $\vec{\nabla}T$ son puntos a lo largo del rayo y por lo tanto dT/ds también, esta es la derivada escalar con respecto a la longitud de arco a lo largo del rayo y da la magnitud total de $\vec{\nabla}T$.

Así la ecuación de rayo se convierte en

$$\frac{1}{v^2(\vec{x})}\vec{\nabla}v(\vec{x}) = -\frac{d}{ds}\frac{1}{v(\vec{x})}\frac{d\vec{x}}{ds} \tag{1.13}$$

La cual es una ecuación diferencial ordinaria de segundo orden para el vector del rayo $\vec{x}$. Este resultado se puede expresar como un sistema de ecuaciones de primer orden modificando el lado derecho de (1.13) usando

$$\frac{d}{ds} = \frac{dt}{ds}\frac{d}{dt} = \frac{1}{v(\vec{x})}\frac{d}{dt} \tag{1.14}$$

y definiendo el vector de Slowness, $\vec{p} = \vec{\nabla}T$, esto es (de la ecuación 1.10)

$$\vec{p} = \frac{1}{v(\vec{x})}\frac{d\vec{x}}{ds} = \frac{1}{v^2(\vec{x})}\frac{d\vec{x}}{dt} \tag{1.15}$$

Estas últimas dos ecuaciones permiten escribir como un sistema de primer orden la ecuación (1.13)

$$\frac{d\vec{x}}{dt} = v^2(\vec{x})\vec{p} \tag{1.16}$$

y

$$\frac{d\vec{p}}{dt} = -\frac{\vec{\nabla}v(\vec{x})}{v(\vec{x})} \tag{1.17}$$

Soluciones numéricas de las ecuaciones (1.16) y (1.17) pueden ser usadas para propagar el rayo. Un ejemplo de estos métodos puede ser Runge-Kutta de orden 4. Desarrollo realizado a partir de [Margrave 2001].

Otro tipo de aproximaciones a un rayo es usando la teoría de grafos y el método del camino mas corto introducido por [Moser 1991], el cual se discutirá en el capitulo 2.

1.3. Trazado de rayos

El trazado de rayos se basa en la idea de que la energía sísmica en las altas frecuencias, sigue una trayectoria determinada por la trayectoria de rayos, la cual es estacionaria. Físicamente, esta aproximación describe cómo se propaga la energía en la misma dirección, hasta que se refracta o refleja por las variaciones de velocidad. El trazado de rayos en un medio especifico puede ser resuelto por cuatro caminos: numéricamente, analíticamente, semianalíticamente, o por aproximaciones del rayo en celdas [Cerveny 2005].

Uno de los métodos mas comunes es la solución numérica de la ecuación Eikonal, la cual rige la geometría y dirección de propagación del rayo en el subsuelo, y surge de la ecuación de onda acústica con la debida aproximación en altas frecuencias. La solución numérica de la ecuación Eikonal provee los tiempos de propagación de las ondas sísmicas, las cuales se clasifican en dos tipos: ondas de cuerpo (ondas P y las ondas S), y ondas superficiales (ondas Love y ondas Rayleigh). En nuestro caso, solo estamos interesados en los primeros arribos de ondas P, ya que estos están asociados con la energía de la onda que se propaga directamente desde la fuente hasta el receptor. La trayectoria encontrada numéricamente usando la ecuación Eikonal tiene una propiedad importante y está dada por el principio de Fermat: el camino del rayo es una curva a lo largo del espacio cuyo tiempo de propagación es estacionario [Moser 1991]. Esta propiedad puede ser explotada para motivar la

construcción de un rayo entre una fuente sísmica y un receptor, que puede estar basada en este principio. En este caso su construcción se realiza a partir de una analogía entre el camino de un rayo sísmico y el camino mas corto en una malla de puntos. Una estrategia como estas es denominada el método del camino mas corto para calcular rayos sísmicos [Moser 1991].

1.4. Método del camino más corto

El problema del camino más corto es un problema clásico e importante de la combinatoria que puede ser aplicado en muchos contextos, usualmente de naturaleza discreta en los cuales aparecen una cantidad finita de objetos, y la solución exacta del problema puede ser encontrada en un número finito de pasos. Un ejemplo es un mapa de carreteras, la malla consiste de ciudades y conexiones entre ellas; y es posible preguntarse por el camino que tiene longitud mínima entre una ciudad y otra. Estas aplicaciones se resuelven usando un grafo, el cual consiste de un conjunto de puntos denominados nodos o vértices, unidos por enlaces llamados aristas. El problema del camino más corto está definido en términos de un grafo dirigido que consiste de n nodos, que son numerados desde 0 hasta $n-1$. El nodo 0 puede ser la fuente, y es el nodo donde se genera energía sísmica controlada. Las aristas entre vértices tienen asociado un peso igual al tiempo de propagación de un rayo sísmico que cruza de un vértice a otro. El problema es encontrar el camino de longitud mínima ("camino más corto") que minimiza el tiempo de propagación desde la fuente a todos los nodos i, en particular a un nodo o conjuntos de nodos escogidos previamente para ser los receptores. Encontrar el camino más corto desde un vértice a otro usando estos pesos, es una aproximación al rayo sísmico entre ellos, debido al principio de Fermat [Moser 1991].

El modelo de la tierra es representado por una malla, consistentes de nodos conectados por aristas (Figura: 1.2). Las mallas usadas para representar el

modelo de velocidades son dispersas; es decir, cada nodo es conectado con un número restringido de nodos en su vecindad y no con todos los nodos que están cercanos a él. Estas vecindades son denominadas lista de adyacencia de los nodos y contiene todos los nodos tal que existe una arista que los conecta, si tal arista no existe en la matriz de adyacencia hay un cero. El modelo de velocidades es representado como un grafo pesado $G = (V, E, W)$, donde V es el conjunto de vértices, E es el conjunto de aristas y W es el conjunto de los pesos de las aristas.

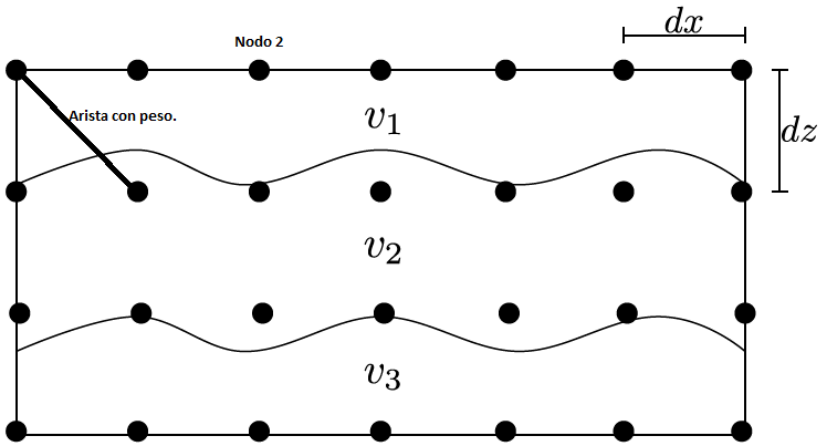


Figura 1.2: Malla rectangular modelo de velocidades. Modificada [Monsegny y Agudelo 2013]

La exactitud del camino más corto seguido por el rayo depende de la cantidad de nodos y aristas usados en la malla. Entre más nodos se utilicen, mayor será la aproximación a la trayectoria seguida por el rayo físico ("real"), pero los costos computacionales serán mucho más altos. Muchos de los rayos que aproximan los caminos variacionales pueden aproximarse por estos algoritmos, aunque algunos no están representados en este método.

1.5. Algoritmo de Dijkstra

La búsqueda del camino mas corto entre dos puntos a través de un conjunto de nodos en un grafo es un problema computacionalmente costoso, que resuelto de forma directa crece como N^2. Sin embargo, se han desarrollado algoritmos que permiten resolver el problema a costos computacionales razonables. Uno de los algoritmos mas exitosos es el algoritmo de Dijkstra.

El algoritmo de Dijkstra, consiste en lo siguiente: Dado un grafo y un vértice fuente en el grafo, encontrar el camino mas corto de la fuente a todos los otros vértices en el grafo dado. Para ver el algoritmo del camino mas corto, consideremos un grafo $G(V, E, W)$ donde V es un conjunto que contiene n nodos y E es un conjunto de aristas tal que $E \subseteq V \times V$. Además, este grafo posee una función de peso $W : V \times V \to \mathbb{R}$ que asigna un número real a cada arista. Para el caso de trazados de rayos sísmicos, este peso se asocia con el tiempo de propagación de un rayo entre un nodo y otro, siempre y cuando estén conectados. W puede ser representado por una matriz $w_{i,j}$ la cual es simétrica en virtud del principio de reciprocidad, es decir, $w_{i,j} = w_{j,i}$ y donde cada entrada de la matriz corresponde al tiempo de propagación entre el nodo i y el nodo j en el grafo. Por convención el tiempo de propagación de un nodo a sí mismo es cero, así $w_{i,i} = 0$ para todo $i \in V$, en nuestro código $TP(src) = 0$ donde src es el nodo fuente. No existen pesos negativos y por ende $w_{i,j} \geq 0$, y para el caso en que los nodos i y j no estén conectados se define $w_{i,j} = w_{j,i} = \infty$. Para propagar un rayo desde un nodo i se buscan todos los vecinos que están conectados con i, este conjunto de nodos se denota por neighb(i), y $TP(i)$ indicará el tiempo de propagación desde la fuente hasta el nodo i.

Una trayectoria de un rayo es una secuencia de nodos y conexiones cuyo tiempo de propagación se define como la suma de las conexiones que forman la trayectoria del rayo. A partir de estas definiciones el algoritmo de Dijkstra

se puede seguir del siguiente pseudocódigo:

1. **Inicialización:**

$$Q := V \qquad TP(i) := \infty \text{ para todo } i \in V$$

$$P := \phi \qquad TP(src) = 0$$

2. **Selección:** encontrar $i \in Q$ con mínimo tiempo de propagación $TP(i)$

3. **Actualización:**

$$TP(j) := \min(TP(j), TP(i) + w_{i,j}) \qquad \text{para todo } j \in \text{neighb}(i) \cap Q$$

Transferir i de Q a P.

4. **Condición para finalizar:**

$$\text{sí } P = V \text{ pare}$$

$$\text{sino, ir al paso 2}$$

Los nodos son divididos en un conjunto P de nodos con tiempo de propagación definitivo y conocido, y un conjunto Q que es el complemento de P. Inicialmente, P es vacío y $Q = V$, es decir, Q contiene todos los vértices . Para la fuente (la cual será denotada como el nodo src) el tiempo de propagación es definitivo ($TP(src) = 0$) y así puede ser transferido a P. Se buscan los vecinos de la fuente y se les asigna a cada uno de ellos el tiempo de propagación a la fuente. Para el vector de tiempos de propagación TP se busca el menor tiempo almacenado entre los nodos que están en Q, el nodo hallado queda marcado como definitivo y puede pasar a P. El algoritmo continua tomando al nodo encontrado como una nueva fuente (principio de Huygens). El tiempo de propagación almacenado en un nodo se actualiza si el tiempo de la nueva fuente mas el tiempo hasta el nodo es menor que el tiempo almacenado en el nodo.

1.6. Estado del arte

En el transcurso del siglo XX, episodios notables marcaron avances en los métodos de prospección sísmica. Si bien muchas de estas tecnologías tardaron tiempo para pasar del formalismo a la práctica, cada una generó finalmente nuevas oportunidades de exploración. En 1920, se introdujeron los disparos analógicos de cobertura simple para detectar capas inclinadas del subsuelo [Albertin y cols. 2002], mostrando esto la necesidad de conocer una aproximación del terreno antes de perforar el suelo. Desde esta época, el problema de encontrar yacimientos de hidrocarburos se ha tornado cada vez más complejo y los costos de realizar excavaciones en busca de estos son altos. Por ende, diferentes técnicas computacionales de tomografía se han utilizado para tener una vista previa del subsuelo a explorar y mejorar los índices de éxito en la detección de hidrocarburos.

El trazado de rayos y los tiempos de propagación son utilizados para este propósito. Estas técnicas son una aplicación de diferentes ramas de la matemática, la física y la ingeniería, las cuales son condicionadas para ser implementadas en computación. Algunas de estas técnicas computacionales son: La solución de la ecuación Eikonal por separación de variables, el método de diferencias finitas [Vidale 1988] y el trazado de rayos en una malla usando el camino más corto [Moser 1991]. Estos métodos de trazados de rayos son usados para la parte de modelado, encontrando los tiempos de propagación de la onda.

Trazado de rayos usando el camino más corto

Las primeras referencias relacionadas al trazado de rayos en una malla son de [Nakanishi y Yamaguchi 1986] quienes necesitaban un trazador de rayos en su estudio de inversión no lineal de los primeros tiempos de llegada, para encontrar el camino del rayo y los tiempos de propagación de un punto fuente a un número de puntos receptores. [Moser 1991], [Moser 1992] utilizó esta idea

e investigó métodos para mejorar la eficiencia de la búsqueda de la trayectoria de un rayo en el tiempo mínimo y además mostró cómo es posible tratar los tiempos de llegada de los rayos reflejados. Su idea consistía de una malla de puntos que son conectados con sus puntos vecinos por una arista y la longitud de dicha arista es tan larga como el tiempo que tarda la onda en ir de un punto a otro. La idea de Moser fue brillante pero la implementación computacional no era muy eficiente.

En teoría de grafos, un algoritmo muy eficiente para determinar el camino más corto en una malla de puntos fue propuesto por [Dijkstra 1959]. Varias modificaciones de este algoritmo han sido usadas generalmente en el trazado de rayos sobre mallas. [Gallo y Pallottino 1986] describen versiones modernas del algoritmo de Dijkstra, que son lo suficientemente eficaces para hacer que el método del camino más corto sea competitivo con otros métodos para el modelado sísmico.

En aras de mejorar cada vez más los tiempos de cómputo, diferentes mejoras en tiempos de cómputo a la técnica del camino más corto fueron propuestas por: [Fischer y Lees 1993], [Cheng y House 1996], [Zhang y cols. 2004], [ZHANG y cols. 2006].

La aparición de herramientas de cómputo mucho más poderosa impulsó el desarrollo en el trazado de rayos. Pues se tenía la facilidad de trabajar con mallas mucho más grandes y obtener resultados en menor tiempo. De esta manera los algoritmos proporcionaron un marco atractivo para la tomografía basada en rayos, algunos de estos algoritmos se encuentran en [PIETA y DWORNIK 2012], [Giroux y Larouche 2013], [Monil y cols. 2018], [Malony y cols. 2016a], [Malony y cols. 2016b].

Otra alternativa por medio de la cual se puede acelerar la solución del problema es el uso de computación en paralelo en GPU, [Monsegny y Agudelo 2013] mostraron una posible descomposición del método del camino más corto, para que una unidad de procesamiento gráfico

pueda realizar los cálculos. De hecho, [Harish y Narayanan 2007] usaron GPUs para resolver el problema del camino más corto para una sola fuente en un grafo general con millones de vértices. Lo cual muestra la efectividad de estos dispositivos a la hora de procesar datos.

Tiempos de propagación mediante diferencias finitas y ecuación Eikonal

Usualmente especificado en la literatura sismológica como "solver"de diferencias finitas para la ecuación Eikonal. Soluciones numéricas directas a la ecuación Eikonal se usan en algunos casos excepcionales. [Pilipenko 1983] da una descripción de los algoritmos y ejemplos numéricos de aplicaciones.

El método de diferencias finitas para calcular los primeros tiempos de llegada a lo largo de una expansión del frente de onda a partir de cuadrados en el caso 2-D y de cubos en el caso 3-D fue propuesto por [Vidale 1988], [Vidale 1990]. Esta expansión del frente de onda se asemeja a la propagación de una onda plana. Las ecuaciones de expansión son diferentes, dependiendo de la posición del punto en el cuadrado y en su distancia desde la fuente. Los principios fundamentales del cálculo de diferencias finitas de los tiempos de propagación, pueden ser demostrados en el caso de una expansión del semiespacio. [Reshef y Kosloff 1986] fueron los primeros en proponer este algoritmo, principalmente para aplicaciones sísmicas de los métodos de reflexión.

Una expansión del frente de onda a lo largo de círculos (en 2-D) y esferas (en 3-D), fue propuesta por [Schneider Jr 1995]. Los cálculos están hechos en un sistema rotado de coordenadas esféricas y la aproximación en ondas esféricas. Las aproximaciones usadas por Vidale no son lo suficientemente precisas en situaciones en las cuales el frente de onda está fuertemente curvado. Para eliminar estos problemas, los algoritmos propuestos por Vidale han sido generalizados y extendidos de muchas formas.

[Qin y cols. 1992] presenta un esquema de diferencias finitas cuya región de solución progresa en forma de expansión de frentes de onda en lugar de una

expansión de cuadrados. Esta aproximación mejora en precisión de los tiempos de propagación, pero tiene un incremento en costo computacional. El método de diferencias finitas para calcular la solución cinemática de la ecuación de onda fue usado por [Podvin y Lecomte 1991] y [Van Trier y Symes 1991].

El algoritmo de Vidale fue extendido por [Hole y cols. 1992] y [Matsuoka y Ezaka 1992] para calcular los tiempos de propagación para las ondas reflejadas en dos dimensiones. Algoritmos similares a los anteriores y basados en el de Vidale fueron propuestos por: [Faria y Stoffa 1994]; [Klimes y Kvasnicka 1994]; [Riahi y Juhlin 1994] y [Hole y Zelt 1995]. De estas modificaciones el algoritmo propuesto por [Podvin y Lecomte 1991] funciona bien en regiones con grandes contrastes de velocidad en interfaces estructurales de diseño arbitrario.

[Sava y Fomel 2001] proponen un método robusto y completo para calcular tiempos de propagación, basado en un sistema de ecuaciones diferenciales parciales, que es equivalente a la ecuación Eikonal pero formulada en un sistema de coordenadas de rayo.

1.7. Problemas inversos

La Teoría de Problemas Inversos es un conjunto de técnicas matemáticas para extraer conocimiento acerca del mundo físico a partir de datos obtenidos mediante observación. Como en la mayoría de los problemas inversos los datos son simplemente una lista de valores numéricos, un vector proporciona un medio conveniente para su representación. Si se realizan N mediciones en un experimento particular, por ejemplo, uno podría considerar estos números como los elementos de un vector $\mathbf{d}$ de longitud N. Los datos son analizados para inferir, de la mejor forma posible, los valores numéricos de los parámetros del modelo. Estos parámetros son escogidos para capturar el carácter esencial de

los procesos que se estudian. Los parámetros del modelo se pueden representar como los elementos de un vector $\mathbf{m}$ de longitud M.

$$\mathbf{d} = [d_1, d_2, d_3, \ldots, d_N]^T$$
$$\mathbf{m} = [m_1, m_2, m_3, \ldots, m_M]^T$$

La declaración básica de un problema inverso es que los parámetros del modelo y los datos están de alguna manera relacionados. Esta relación es denominada modelo, y usualmente, toma la forma de una o mas fórmulas que se espera que sigan los datos y los parámetros del modelo.

Los problemas inversos más simples y mejor comprendidos son aquellos que pueden representarse con la ecuación lineal explícita $\mathbf{Gm} = \mathbf{d}$. Esta ecuación forma la base del estudio de la teoría inversa discreta. Muchos problemas inversos importantes que surgen en las ciencias físicas involucran precisamente esta ecuación. Otros, si bien involucran ecuaciones más complicadas, a menudo se pueden resolver a través de aproximaciones lineales, como es el caso de la tomografía sísmica basada en trazado de rayos. La matriz $\mathbf{G}$ relaciona los datos con el modelo a través del producto $\mathbf{Gm}$, esta matriz se conoce como operador de la teoría. De hecho, contiene toda la información física y matemática del problema que se quiere modelar [Snieder y Trampert 1999].

A partir de los datos observados se pueden realizar estimaciones del modelo, las cuales, son muy diferentes al modelo verdadero en problemas prácticos. El modelo estimado es denotado como $\tilde{\mathbf{m}}$. Hay muchas formas de diseñar un operador inverso que mapea los datos en el modelo estimado, ver [Menke 2012], [Tarantola 1984], [Parker 1994]. Cualquiera sea el estimador escogido, el mapeo lineal mas general de los datos al modelo estimado puede ser escrito como:

$$\tilde{\mathbf{m}} = \mathbf{G}^{-g}\mathbf{d} \tag{1.18}$$

El operador $\mathbf{G}^{-g}$ es llamado la *inversa generalizada* de la matriz $\mathbf{G}$. En general, el número de datos es diferente del número de parámetros del modelo. Por esta razón, la matriz $\mathbf{G}$ usualmente no es cuadrada y por lo tanto su inversa clásica no existe. Para resolver este tipo de problemas, existen dos clases de algoritmos; un método directo cuya solución es encontrada en un solo paso, pero que solo es adecuada para problemas de menor escala. Y métodos iterativos, tales como el método del gradiente conjugado ([Scales 1987]) y las técnicas de reconstrucción algebraica ([Lo y Inderwiesen 1994]), que funcionan bien en sistemas a gran escala.

El esquema de inversión, donde se genera un dato sintético o dato calculado d_{cal} a través de un modelo y parámetros iniciales, para después compararlo con el dato observado d_{obs} en búsqueda de un modelo que minimice la diferencia entre d_{obs} y d_{cal}, se llama Inversión basada en Modelos [Sen 2006]. En ellos se buscan los parámetros del modelo que permiten una configuración óptima y son parte de los Métodos de Inversión para Estimar Parámetros [Gubbins 2004]. Ambos métodos realizan suposiciones iniciales para el modelo o los parámetros del modelo y el objetivo es optimizar el error $e = d_{obs} - d_{cal}$, también conocido como función objetivo [Fichtner 2010]. En la tomografía de tiempos de propagación el objetivo es minimizar el error entre los tiempos observados y los tiempos calculados a partir de métodos iterativos.

1.8. Introducción a la Tomografía de tiempos de propagación

Uno de los métodos de tomografía mas simples para describir es la tomografía de tiempos de propagación (Traveltime Tomography), y en la cual solo se usan los primeros tiempos de llegada de una onda, entre una fuente y un receptor. Para determinar la distribución de velocidades en un medio, la

tomografía intenta resolver un conjunto de ecuaciones simultaneas.
Consideremos un modelo $2D$, el cual está dividido en 9 celdas con velocidades constante v_i. Los puntos A y C, corresponden a un par fuente receptor, y el punto B representa un reflector sobre una interfaz en el medio; como se observa en la figura (1.3).

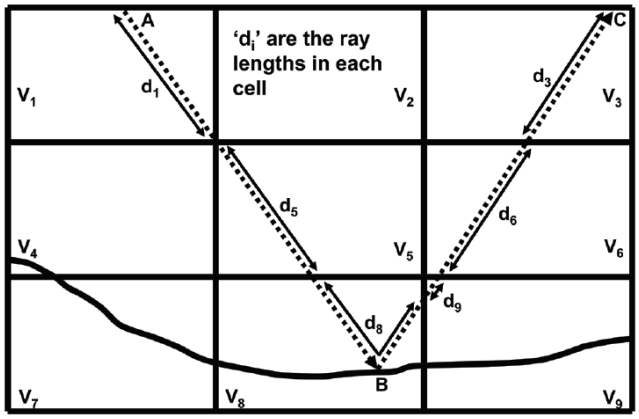


Figura 1.3: Modelo de celdas que describen velocidades en el subsuelo y un rayo con una reflexión en el punto B. (Tomada: [Jones 2010])

El trazado de rayos es usado dentro de los algoritmos de tomografía para determinar los posibles caminos y longitudes de los rayos dentro de cada celda del modelo. Para el rayo de la figura (1.3), que va entre los puntos A, B y C, se tiene un tiempo de propagación asociado t_{ABC} dado por:

$$t_{ABC} = d_1/v_1 + d_5/v_5 + d_8/v_8 + d_9/v_9 + d_6/v_6 + d_3/v_3 \tag{1.19}$$

Para un punto común de reflexión de los rayos en el medio (CMP-gather) se tienen varias medidas de tiempos de propagación, y es posible representar de forma matricial la relación entre los tiempos de propagación, la distancia recorrida por cada rayo en las celdas y la velocidad de cada celda. Por ejemplo, para los cinco rayos escogidos en la figura (1.4), el punto B es un punto común

de reflexión para estos cinco rayos y los tiempos de propagación se pueden representar como:

$$t_i = \sum_{j=1}^{N} d_{ij}/v_j = \sum_{j=1}^{N} d_{ij} s_j \tag{1.20}$$

Donde t_i es el tiempo total de propagación para el i-ésimo rayo, d_{ij} es la longitud recorrida del i-ésimo rayo en la j-ésima celda del modelo de velocidades y $s_j = 1/v_j$ es el *slowness* (reciproco de la velocidad) de la celda y N es la cantidad de celdas del modelo. Para los rayos de la figura 1.4, se tiene $N = 9$, $i = 1, 2, 3, 4, 5$.

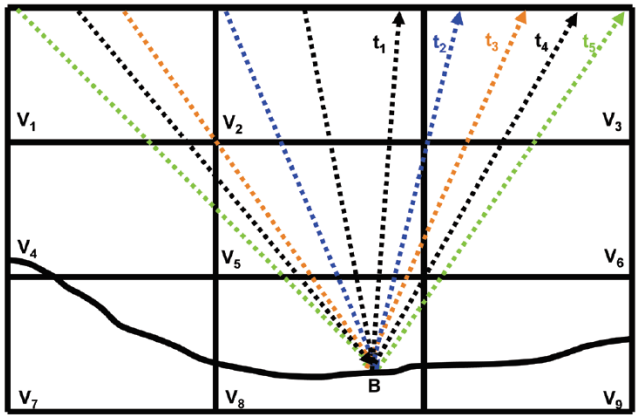


Figura 1.4: Conjunto de 5 rayos con punto de reflexión común en B. (Tomada: [Jones 2010])

La ecuación 1.20 en notación matricial considerando M rayos toma la forma

$$\begin{bmatrix} t_1 \\ t_2 \\ \vdots \\ t_M \end{bmatrix} = \begin{bmatrix} d_{11} & d_{12} & d_{13} & \dots & d_{1N} \\ d_{21} & d_{22} & d_{23} & \dots & d_{2N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ d_{M1} & d_{M2} & d_{M3} & \dots & d_{MN} \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_N \end{bmatrix}$$

$$\mathbf{T} = \mathbf{DS} \tag{1.21}$$

$$\mathbf{T}_{M\times 1} = \mathbf{D}_{M\times N}\mathbf{S}_{N\times 1} \tag{1.22}$$

Muchos de los elementos de $\mathbf{D}$ son cero, pues un rayo no atraviesa todas las celdas del modelo, lo que convierte a la matriz $\mathbf{D}$ en *sparse* y mal condicionada. El objetivo es estimar $\mathbf{S}$, la matriz que contiene el modelo de *slowness*, que en este caso es el vector de parámetros del modelo que se quiere invertir. Este es un problema inverso difícil de resolver, pues generalmente la matriz $\mathbf{D}$ en (1.21) no es invertible, ni cuadrada ya que generalmente la cantidad de rayos (tiempos de propagación) difiere de la cantidad de celdas del modelo. Si se multiplica la ecuación (1.21) por la transpuesta de $\mathbf{D}$ para formar la matriz cuadrada simétrica de $\mathbf{D}$ en el lado derecho de la ecuación y luego buscar la solución de mínimos cuadrados de:

$$\mathbf{D}^T\mathbf{T} = \left(\mathbf{D}^T\mathbf{D}\right)\mathbf{S} \tag{1.23}$$

que es lo mismo que

$$\mathbf{S} = \left(\mathbf{D}^T\mathbf{D}\right)^{-1}\mathbf{D}^T\mathbf{T} \tag{1.24}$$

para resolver la ecuación (1.24) hay dos clases de algoritmos: Un método directo, en el cual la solución es encontrada en un solo paso, pero solo es adecuado para problemas pequeños, pues el cálculo de $\left(\mathbf{D}^T\mathbf{D}\right)^{-1}$ es en términos computacionales bastante costoso. Y un método iterativo, tales como el método del gradiente conjugado o la técnica de reconstrucción algebraica, que funcionan bien en sistemas grandes [Jones 2010] y que se recomiendan para resolver problemas de tomografía sísmica.

El problema de la tomografía se puede seguir como en la figura 1.5. En donde se parte de un modelo inicial S_n y se realiza el trazado de rayos para determinar los tiempos de propagación y las distancias recorridas por los rayos en las celdas del modelo S_n. Luego se obtiene un nuevo modelo calculado a

partir de las actualizaciones dadas por los métodos de reconstrucción algebraica. Si para este modelo el residuo entre los tiempos observados y los calculados para este medio cumplen con una condición de parada, la solución aproximada es regresada. De lo contrario, la inversión continua para S_{n+1}.

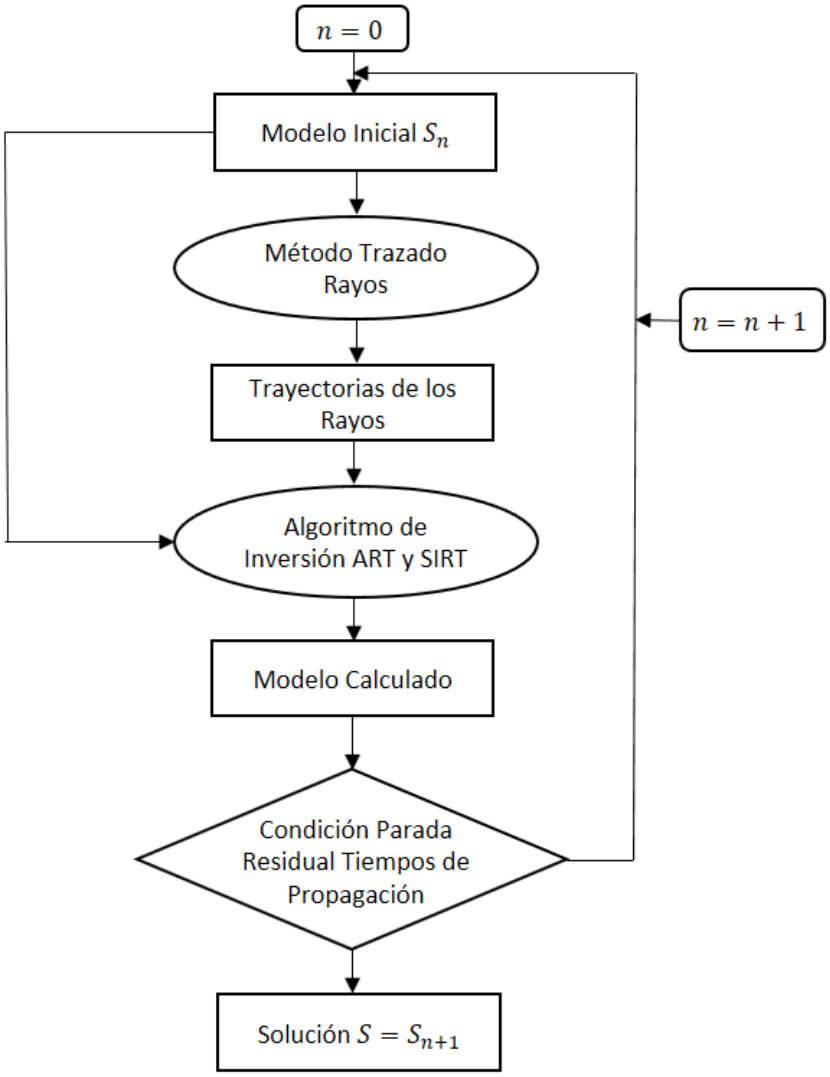


Figura 1.5: Diagrama de flujo para realizar la tomografía de tiempos de propagación.

Nos centraremos en los algoritmos de reconstrucción algebraica.

1.9. Métodos de reconstrucción algebraica

ART (Algebraic Reconstruction Technique) y SIRT (Simultaneous Iterative Reconstruction Technique) son dos implementaciones en tomografía de rayos sísmicos. Ambos métodos actualizan iterativamente un modelo estimado $\mathbf{S}^{est}$ y encuentran una solución aproximada al modelo verdadero $\mathbf{S}^{ver}$.

1.9.1. Técnica de reconstrucción algebraica

La técnica de reconstrucción algebraica o el método de Kaczmarz elude los problemas asociados con matrices *sparse* y da un medio eficiente para encontrar una solución aproximada de la ecuación (1.21) usando un procedimiento iterativo.
A partir de alguna técnica de trazado de rayos se obtiene el operador $\mathbf{D}$ y al aplicarlo a un modelo estimado $\mathbf{S}^{est}$, se encuentran los tiempos de propagación calculados $\mathbf{T}^{cal}$,

$$\mathbf{T}^{cal} = \mathbf{D}\mathbf{S}^{est} \tag{1.25}$$

Para realizar la actualización del modelo estimado actual $\mathbf{S}^{est}$ a un nuevo modelo estimado $\mathbf{S}^{(new)est}$ del cual se espera este mas cercano al modelo verdadero, se usa la diferencia entre los tiempos calculados y los tiempos observados, $\mathbf{T}^{obs} - \mathbf{T}^{cal}$. Esta actualización puede ser escrita en forma de ecuación como:

$$\mathbf{S}^{(new)est} = \mathbf{S}^{est} + \Delta^i\mathbf{S}, \qquad for \;\; i = 1, \ldots, M. \tag{1.26}$$

Donde $\Delta^i\mathbf{S}$ representa el ajuste al modelo actual y el superíndice i significa aplicar la ecuación (1.26) cuando se comparan t_i^{obs} y t_i^{cal} para el i-ésimo rayo. Este ajuste es crucial en el método de Kaczmarz, y viene dado por:

$$\begin{aligned} \Delta^i s_j &= d_{ij} \frac{t_i^{obs} - t_i^{cal}}{\sum_{k=1}^{N} (d_{ik})^2} \\ &= d_{ij} \frac{t_i - \sum_{k=1}^{N} d_{ik} s_k^{est}}{\sum_{k=1}^{N} (d_{ik})^2}. \end{aligned} \tag{1.27}$$

La ecuación (1.27) se puede derivar geométricamente, para mas detalles (ver [Lo y Inderwiesen 1994]). El método de Kaczmarz usa proyecciones ortogonales entre un hiperplano y otro (ver figura 1.6) a partir de una aproximación inicial para encontrar la solución correcta, en este caso, una que describa adecuadamente los datos observados. En adelante, un hiperplano corresponderá a la representación geométrica de una ecuación que haga parte del sistema lineal (1.21). En el caso de dos rayos y dos celdas de velocidad, el sistema lineal es de 2×2 y la representación geométrica de las ecuaciones son rectas en el plano con ejes s_1 y s_2 (ver figura 1.6). Para mas rayos y celdas, la representación geométrica de cada ecuación recibe el nombre de hiperplano.

El algoritmo de reconstrucción algebraica ART, usa la información de cada fila de la matriz **D** (Cada fila corresponde a las distancias recorridas por un rayo en cada una de las celdas) para ajustar el modelo, es decir, el modelo se actualiza inmediatamente de un hiperplano a otro bajo la diferencia entre los datos observados y los calculados. Las iteraciones se pueden apreciar geométricamente en la figura (1.6) y se puede ver como la aproximación inicial G_0 se actualiza mediante el hiperplano 1 a G_1, está se actualiza a G_2 a partir del hiperplano 2, y así sucesivamente, se aproxima a la solución moviéndose de un hiperplano a otro.

Para el i-ésimo rayo correspondiente al hiperplano i, se calcula $t_i^{obs} - t_i^{cal}$ y se usa la ecuación (1.27) para encontrar las actualizaciones de todas las celdas del modelo. El ajuste del rayo i-ésimo a la celda j se denotará por $\Delta^i s_j$ y está dado por la ecuación 1.27.

Es importante notar que la actualización $\Delta^i s_j$ depende de la distancia recorrida

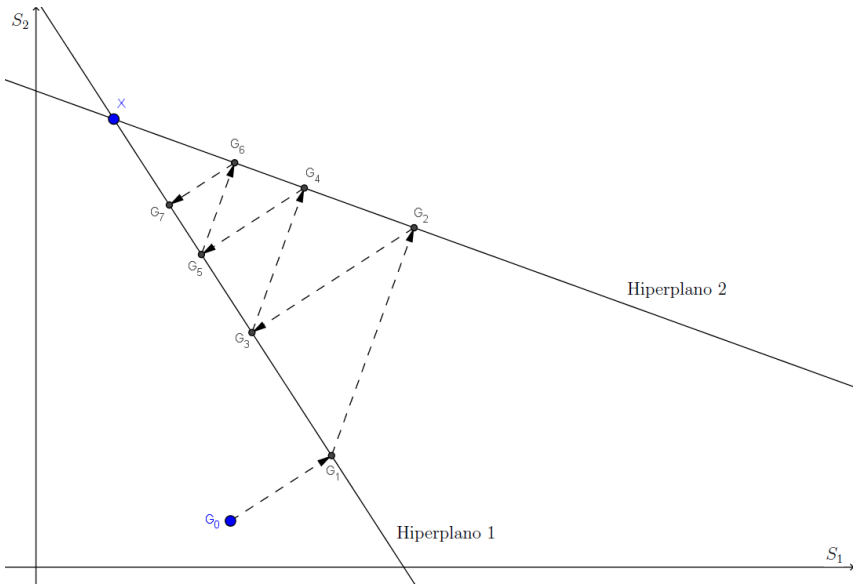


Figura 1.6: Iteraciones del modelo que convergen a una vecindad de la solución correcta. Para dos celdas y dos rayos se obtienen estas rectas, cuando hay mas incógnitas y rayos reciben el nombre de hiperplanos.

por el rayo i en la celda j, es decir, si el rayo no atraviesa la celda j, entonces $d_{ij} = 0$ y la celda no sufre ningún tipo de actualización.

Las celdas del modelo que son atravesadas por el i-ésimo rayo son actualizadas de la siguiente forma:

$$s_j^{(new)est} = s_j^{est} + \Delta^i s_j, \qquad \text{para} \quad i = 1, \ldots, M. \tag{1.28}$$

1.9.2. Técnica de reconstrucción algebraica iterativa simultanea

SIRT difiere de ART en que es necesario recorrer todos los hiperplanos y determinar las correcciones de cada hiperplano a una celda, para luego actualizar el modelo como un promedio de estas correcciones. Este forma de actualizar el

modelo convierte el método de SIRT en uno de los mejores procedimientos de tomografía cuando el ruido sigue una distribución Gaussiana, además, que es altamente resistente a datos *outliers* [Dobróka y Szegedi 2014]. Si el rayo no pasa por la celda la corrección es cero. Las iteraciones se pueden apreciar geométricamente en la figura (1.7) donde se inicia con la aproximación inicial G_0, y para encontrar la siguiente aproximación a la solución, se usa un promedio pesado entre las correcciones G_1' y G_1'' realizadas por los hiperplanos 1 y 2, respectivamente. El procedimiento continua de esta forma generando una secuencia $G_0, G_1, G_2, \ldots$ que converge a la solución X bajo cierto criterio de tolerancia.

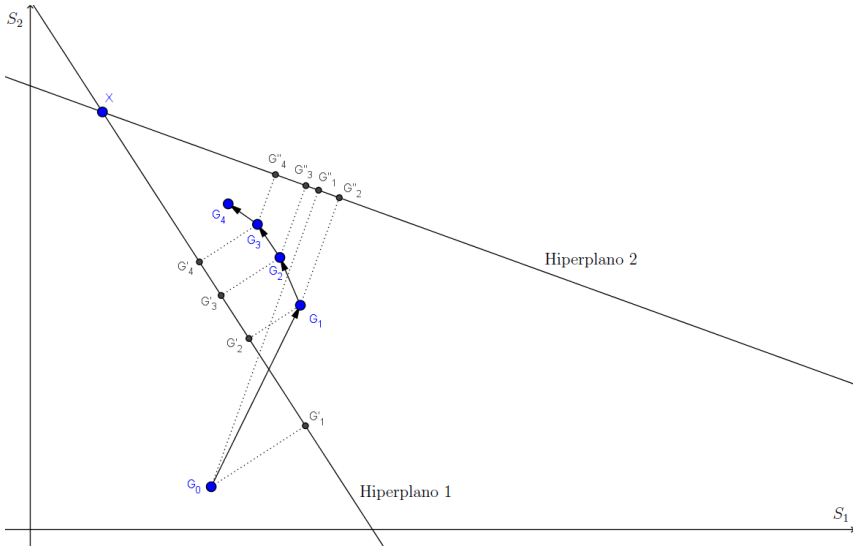


Figura 1.7: Actualizaciones del modelo determinadas por un promedio de las correcciones ΔM_j en SIRT.

Para hallar la actualización de una celda del modelo es necesario calcular previamente el ajuste realizado por todos los rayos, es decir, $\Delta^i s_j$ para $i = 1, \ldots, M$. y sacar un promedio pesado entre las correcciones diferentes de cero ($\Delta^i s_j = 0.0$ si el rayo i no pasa por la celda j). Así la actualización para la

celda j se puede determinar como:

$$\begin{aligned} \Delta s_j &= \frac{1}{W_j} \sum_{i=1}^{M} \Delta^i s_j \\ &= \frac{1}{W_j} \sum_{i=1}^{M} d_{ij} \frac{t_i^{obs} - \sum_{k=1}^{N} d_{ik} s_k^{est}}{\sum_{k=1}^{N} (d_{ik})^2}, \quad \text{para} \;\; j = 1, \ldots, N. \end{aligned} \tag{1.29}$$

Donde W_j es el número de rayos que pasan por la j-ésima celda. El nuevo modelo estimado se actualiza a partir del promedio de correcciones Δs_j como:

$$s_j^{(new)est} = s_j^{est} + \Delta s_j, \qquad \text{para} \;\; j = 1, \ldots, N. \tag{1.30}$$

Capítulo 2

Modelado de reflexiones y refracciones

A continuación se presenta la implementación de un trazador capaz de modelar rayos por refracción y reflexión simultáneamente en un medio con variación lineal de velocidad en profundidad. Este tipo de modelado ha sido usado por [Meléndez y cols. 2015] y [Zhang y cols. 2018] en tomografías sísmicas diferentes a las técnicas de reconstrucción algebraica usadas en este trabajo. Se muestra como se realizó la implementación del trazador de rayos, a partir del método del camino más corto. Además, de una explicación detallada de las funciones y variables involucradas en el algoritmo, y las indicaciones básicas para que otra persona pueda usar el trazador de rayos implementado en este trabajo. Al final del capítulo, se explica una estrategia de paralelización en CPUs de nuestro código con la librería MPI.

2.1. Método del camino más corto

Una propiedad importante de los rayos sísmicos está dada por el principio de Fermat: la trayectoria del rayo es una curva en el espacio cuyo tiempo de

propagación es estacionario. A partir de este principio, es posible enumerar todas las trayectorias desde un punto inicial (fuente) y un punto final (receptor), y realizar la búsqueda del camino mas corto para llegar desde la fuente hasta el receptor. Dicho camino aproxima la trayectoria de un rayo en el subsuelo [Moser 1991].

2.2. Trazado de rayos usando método del camino más corto

A continuación se describe el método para trazar rayos en un medio cuya velocidad aumenta con la profundidad de forma lineal, y que puede ser expresada como la función lineal $v = a + bz$, se pueden usar otros modelos que posean algún cambio en profundidad.

Dada una malla regular de puntos (no necesariamente $dx = dz$), se le asigna a cada nodo una velocidad dependiendo de la profundidad a la que se encuentre y se procede a asignar los pesos (tiempos de propagación entre un nodo y otro).

2.2.1. Vecinos

Para conocer los vecinos con los cuales se conectará un nodo se puede usar un esquema de vecindad rectangular, en donde los vecinos se escogen a partir de un radio de propagación constante para el i-ésimo nodo y trazando la arista entre los vértices de la vecindad y el nodo en cuestión. En la figura (2.1) se muestran dos vecindades, una de radio 1 y la otra de radio 2. El nodo rojo corresponde a la fuente y los nodos azules son los vecinos con los que se puede trazar una arista desde la fuente. Para tener una mayor cobertura de los ángulos de propagación del rayo desde un nodo especifico, se puede escoger un radio mayor para la búsqueda de vecinos y así cubrir una mayor cantidad de

ángulos de propagación del rayo como se muestra en la figura 2.2. Dado que un nodo fuente tiene vecinos en todas las direcciones es posible que los rayos se propaguen con ángulos entre 0° y 360°. En la figura 2.2, se observa los posibles ángulos que pueden barrer los rayos entre 0° y 90°; para la propagación de cuarto orden el ángulo mínimo entre aristas es de 3.2° y el máximo ángulo de 14.0°, esto implica que la discretización de los ángulos de propagación no es homogénea y depende directamente del radio de propagación elegido.

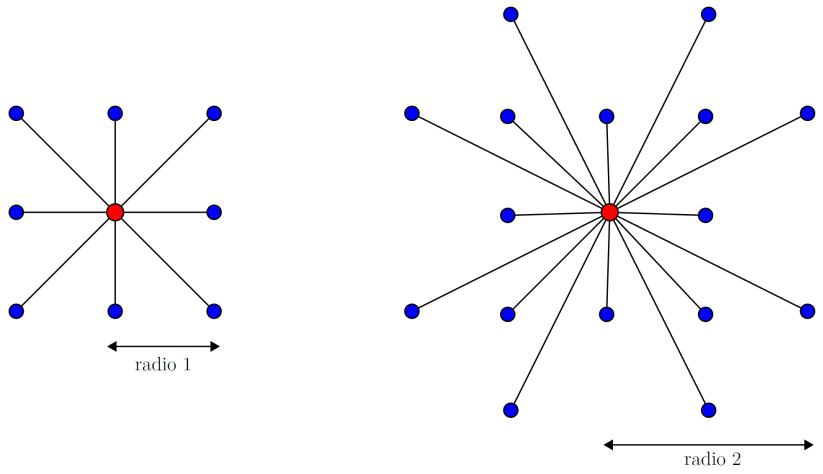


Figura 2.1: Vecindad rectangular de radios $r = 1$ y $r = 2$, respectivamente.

2.2.2. Asignación de pesos

Dado que un nodo tiene un número finito de conexiones con puntos que están cercanos a él, es posible viajar desde un nodo a otro a través de estas conexiones. Para asignar el tiempo de propagación (peso de la arista) entre los puntos que se conectan, se puede hallar la distancia euclidiana y multiplicarla por un promedio de *slowness* $(1/v)$ entre los nodos implicados. Es decir, si el nodo

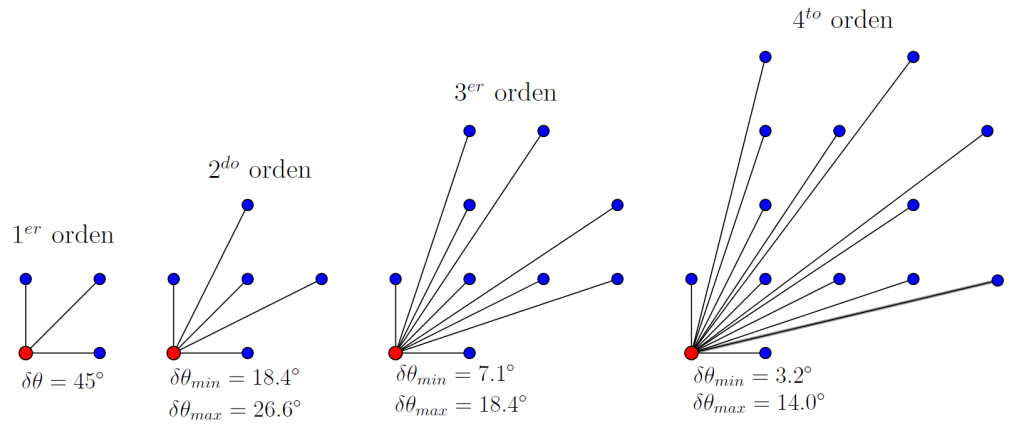


Figura 2.2: Vecinos de un nodo especifico indicando el menor y el mayor ángulo de propagación para diferentes ordenes de propagación.

i de coordenadas espaciales (x_i, z_i) se conecta con el nodo j con coordenadas (x_j, z_j), entonces el tiempo de propagación t_{ij} está dado por:

$$t_{ij} = d_{ij} * \left(\frac{s_i + s_j}{2} \right), \tag{2.1}$$

donde $d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ y s_i es el *slowness* del vértice. Este peso debe ser el mismo en ambas direcciones, en virtud del principio de reciprocidad. Para este trabajo, el código implementado evita almacenar la matriz de adyacencia que se genera en un grafo según las conexiones de los nodos. En términos computacionales esto evita el uso y acceso a la memoria de una matriz bastante grande, ya que es de tamaño $(\texttt{GRIDX*GRIDZ})^2$, donde $GRIDX$ y $GRIDZ$ son la cantidad de nodos para discretizar el medio, y que puede tornarse poco manejable en problemas de gran escala. Así, el problema de cargar los pesos a las aristas se convierte principalmente en cálculos de CPU en cada iteración. Para asignar los pesos a las aristas se usa la ecuación (2.1).

2.2.3. Búsqueda del camino mas corto Dijkstra

Luego de tener una malla que consta de vértices, aristas y pesos, se procede a encontrar el camino mas corto entre un nodo y otro a partir del algoritmo de Dijkstra. El algoritmo realiza la búsqueda de un camino mínimo desde la fuente a cualquier otro punto de la malla, para esto recorre todos los nodos de la malla y asigna a cada nodo visitado un tiempo de propagación acumulado TP desde la fuente. En otras palabras, $TP[j]$ almacena el mínimo tiempo que se puede encontrar entre el nodo j y la fuente. Este tiempo puede cambiar si al unir un vecino u del nodo j, se cumple que el tiempo acumulado en $TP[u]$ más el tiempo de propagación desde u a j es menor que el tiempo acumulado en $TP[j]$, es decir, si

$$TP[u] + t_{uj} < TP[j] \tag{2.2}$$

entonces el tiempo en el nodo j es actualizado como

$$TP[j] = TP[u] + t_{uj}. \tag{2.3}$$

Cuando esto ocurre, aparece un arreglo importante para la reconstrucción del rayo y es el vector predecesor, el cual almacena el identificador del nodo predecesor que esta minimizando el tiempo de propagación desde la fuente. Si la ecuación (2.2) se cumple, entonces el nodo predecesor que minimiza el tiempo de propagación desde la fuente hasta el nodo j es u, y así, $predecesor[j] = u$. Cuando todos los nodos tengan un tiempo de propagación definitivo se cuenta con toda la información necesaria para reconstruir el rayo entre una fuente y un receptor. Una de las ventajas del método del camino mas corto es que encuentra el camino mas corto a todos los nodos de la malla, cualquier vértice diferente a la fuente puede ser escogido como un receptor, para efectos reales los receptores son escogidos en la superficie (nodos superiores de la malla).

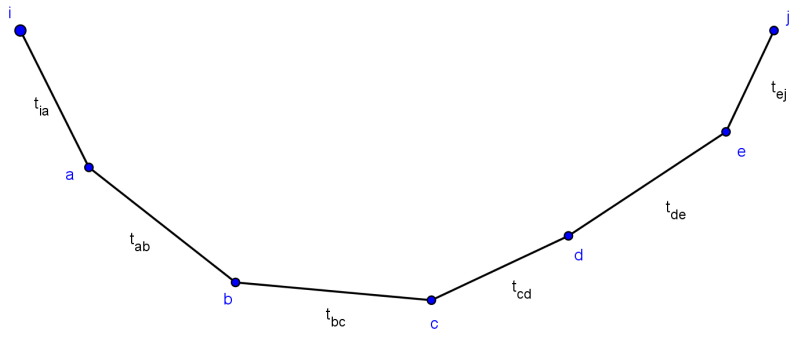


Figura 2.3: Camino con tiempo de propagación mínimo desde la fuente i al receptor j.

2.2.4. Reconstrucción del rayo

Para conocer las coordenadas espaciales por las cuales cruza la trayectoria del rayo se usa el vector *predecesor*. Cada componente de este vector tiene almacenado el vértice del cual proviene el mínimo tiempo desde la fuente, por ejemplo, en el rayo de la figura (2.3) el tiempo que es mínimo desde la fuente i hasta el receptor j con respecto a otros caminos es $t_{ia}+t_{ab}+t_{bc}+t_{cd}+t_{de}+t_{ej}$ y se tiene los siguiente:

$$\begin{aligned} predecesor[j] &= e \\ predecesor[e] &= d \\ \vdots \qquad &= \vdots \\ predecesor[a] &= i \end{aligned}$$

Así es posible reconstruir la trayectoria del rayo iniciando desde el receptor hasta llegar a la fuente.

A partir de estos pasos se puede obtener un trazador de rayos sísmicos y cuyos algoritmos discutiremos en la siguiente sección.

2.3. Implementaciones en ANSI C

En esta sección se discutirá la implementación del trazador de rayos y las rutinas que lo componen. Este algoritmo se puede resumir en los siguientes pasos (ver figura 2.4):

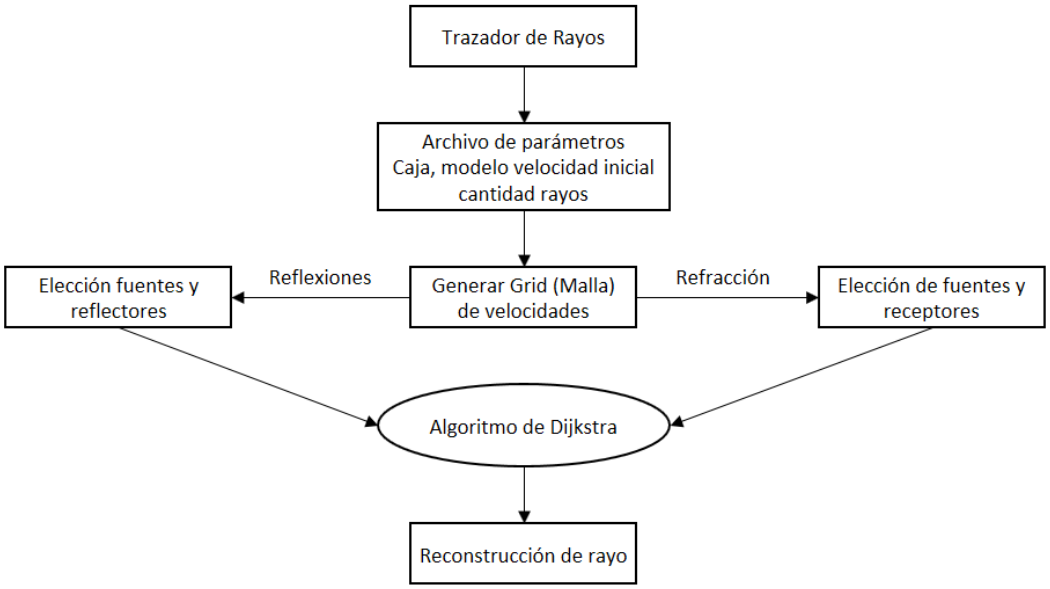


Figura 2.4: Diagrama de flujo para el trazador de rayos.

1. Archivo de entrada de parámetros: en el archivo **param.rays** se ingresan los datos iniciales para el trazado de rayos, en él se encuentran los siguientes datos de entrada:

   - `Xini`, `Zini`, `Xfin` y `Zfin`: son las dimensiones del medio a modelar (caja para el trazado) y son medidas que se dan en metros.
   - `GRADIENTE`: se usa cuando el modelo tiene una variación lineal en profundidad y esta variable determina la pendiente de dicha

ecuación, en el trazado de rayos se trabajó con la relación lineal $v_0 = 1800 + GRADIENTE * z$. Este modelo no necesariamente debe ser lineal, se pueden usar otros modelos de *slowness*.

- `GRIDX` y `GRIDZ`: es la cantidad de nodos en la discretización del modelo 2D para el eje x y eje z, respectivamente.
- `NRAYS`: cantidad de rayos a considerar en el modelado.
- `radiovecinos`: es el orden de propagación para una vecindad rectangular de un nodo de la malla. Las implementaciones mostradas se realizaron con radio 4.
- `GRIDXCELDAS` y `GRIDZCELDAS`: se usan para la tomografía y corresponden a la cantidad de celdas con velocidad constante para realizar inversión.
- `NSHOTS` cantidad de disparos en el modelado. Para que el trazador implementado funcione, es necesario que se cumpla la condición `NSHOTS + NRAYS < GRIDX-1`
- `REFLECTOR_INICIAL` y `REFLECTOR_FINAL`: es el rango donde se realizarán las reflexiones en el fondo del medio escogido.
- `activar_reflexiones`: si se quiere tener en cuenta reflexiones en el fondo esta variable es 1. Para las reflexiones flotantes la variable es 2. Si solo se consideran refracciones usar un valor entero diferente de 1 y 2.

2. Construcción de la malla y algoritmo de Dijkstra: Se genera una malla regular para realizar el trazado de rayos y a cada nodo se le asigna una velocidad (De esto se encarga la subrutina `Construct_Grid`). Luego de esto se procede a realizar la búsqueda del camino mas corto usando el algoritmo de [Dijkstra 1959], ver capitulo 1, y con el cual se tiene acceso a todas las trayectorias con tiempo mínimo desde la fuente a cualquier

punto de la malla. En el código, la función que realiza esta tarea tiene como características:

```
dijkstra(source, predecesor, TP, r, d_eucli_MAX,sc).
```

La función recibe el `id` (identificador del nodo en la malla) de la fuente, el orden de propagación para la búsqueda de vecinos, una distancia máxima entre conexiones de nodos `d_eucli_MAX` que restringe algunos vecinos de aquellos nodos cercanos a la frontera, y `sc` que es un contador de disparos. La función devuelve dos vectores; `predecesor`, útil para la reconstrucción del rayo y `TP` que tiene almacenado los tiempos de propagación mínimos desde la fuente a todos los otros puntos de la malla. Como se advirtió antes, estos tiempos no son una buena aproximación a los tiempos de propagación de los rayos por los promedios que usa para cargar los pesos de las aristas, pero para efectos de encontrar las trayectorias mínimas funciona muy bien. El calculo de los tiempos de propagación se realiza después de encontrar la distancia recorrida por cada rayo en cada celda del modelo, en el capitulo 3 se muestra como se obtienen estos tiempos.

3. Reconstrucción del rayo: La rutina que se encarga de reconstruir las trayectorias seguidas por los rayos es `ray_reconstruction.c`, a partir de la fuente y el vector de predecesores obtenido en el algoritmo de Dijkstra. Es importante definir cuales son los receptores en la malla, pues Dijkstra permite acceder a las trayectorias mínimas desde la fuente a cualquier punto de la malla. La escogencia de los receptores se realizó en la superficie para simular experimentos reales. Los nodos en la superficie pueden ser identificados algebraicamente como múltiplos de la partición en z, es decir, con $\texttt{id's} = h * \texttt{GRIDZ}$ con $h = 1, \ldots, \texttt{GRIDX} - 1$.

La figura 2.5 muestra un rayo entre la fuente (`id` = 0) y el receptor (`id=GRIDZ(GRIDX-1)`) en un medio con variación lineal en profundidad. Y en la figura 2.6 se puede observar un disparo y varios rayos que arriban a receptores en la superficie, en este caso, todos los puntos de la malla se pueden usar como receptores.

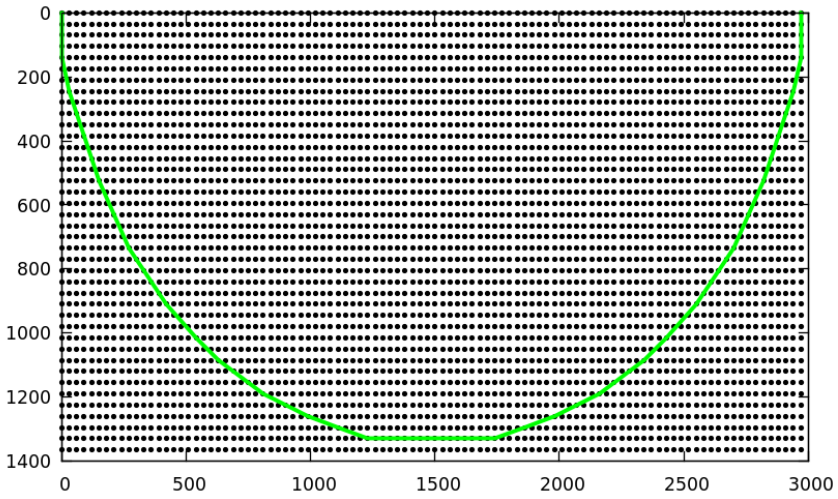


Figura 2.5: Trayectoria de un rayo.

2.3.1. Múltiples fuentes

Para mejorar la iluminación del medio con mas rayos, el trazador permite realizar varios disparos desde fuentes distintas. Para esto, es necesario aplicar el algoritmo de Dijkstra tantas veces como fuentes se deseen ubicar. En las figuras 2.7 y 2.8 se observen los rayos para 1, 5 y 10 disparos con diferentes resoluciones de malla.

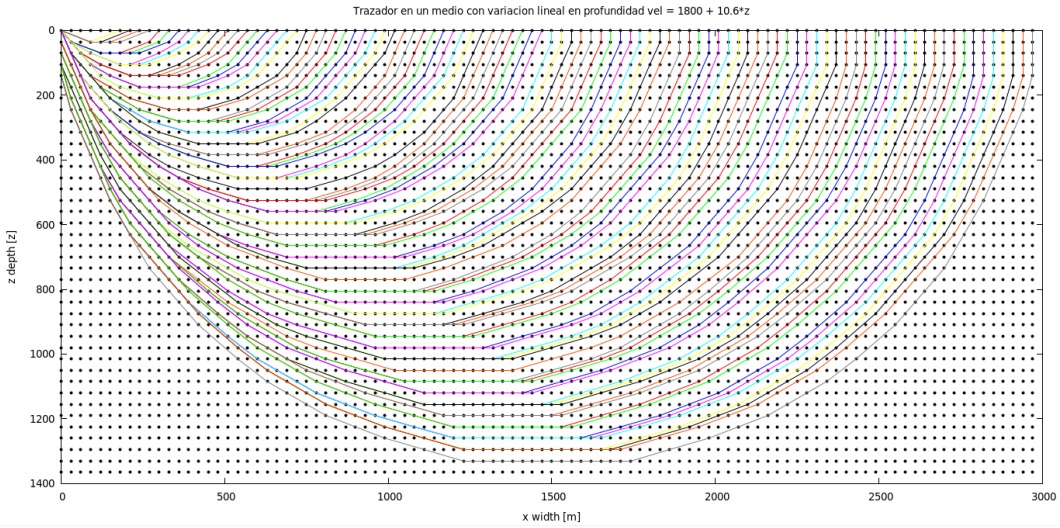
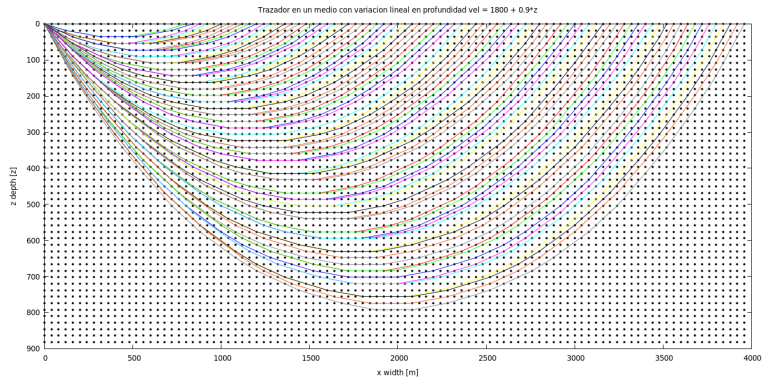


Figura 2.6: Un solo disparo y múltiples receptores en la superficie.

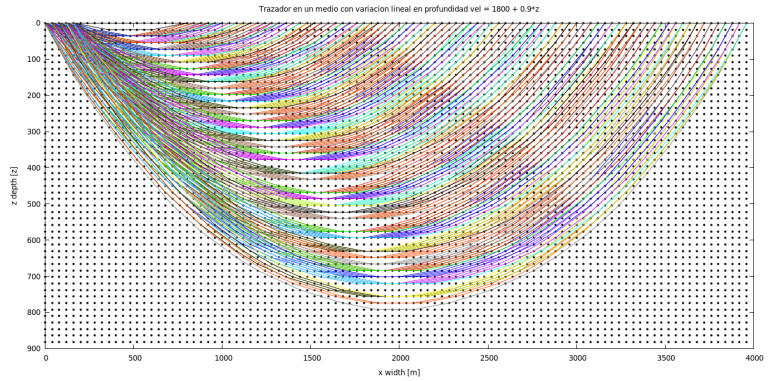
2.3.2. Reflexiones en la última capa

En este modelado es posible introducir reflexiones sobre una interfaz plana basados en un proceso análogo al del software de la empresa NORSAR, el cual dispara rayos desde la fuente y desde un reflector ubicado en el fondo de la región de interés. A partir de una relación trigonométrica que permite asegurar el cumplimiento de las leyes de reflexión, es posible determinar el ángulo de incidencia de un rayo disparado desde la fuente y con llegada en el reflector. Conocido este ángulo se aplica nuevamente el algoritmo de Dijkstra, donde la nueva fuente es el reflector (basado en el principio de Huygens) y realizando una búsqueda sobre los receptores del modelo, para determinar en cual de ellos el ángulo de incidencia desde la fuente coincide con el ángulo de salida del reflector hacia uno de los receptores; cumpliéndose así la Ley de reflexión.

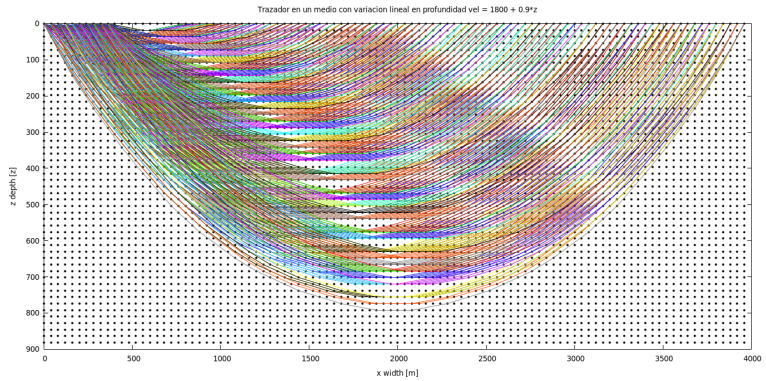
Uno de los problemas de la discretización es que los ángulos barridos por las trayectorias de los rayos no son tan finos, por ende la búsqueda de un reflector cuyo ángulo de incidencia desde la fuente sea igual al ángulo reflejado hacia



(a) 1 disparo y 80 rayos.

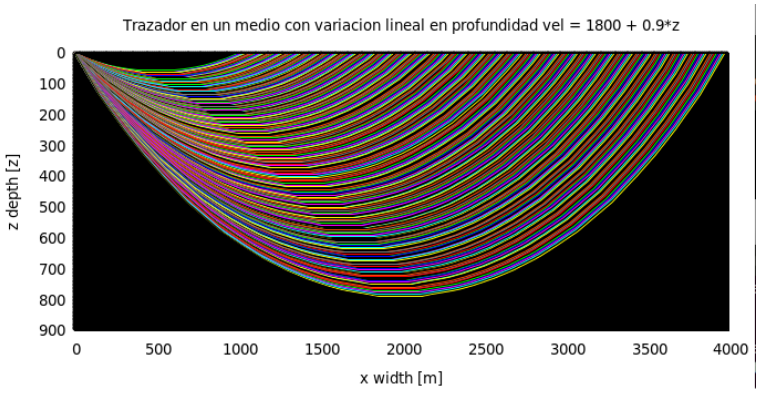


(b) 5 disparos y 80 rayos por disparo.

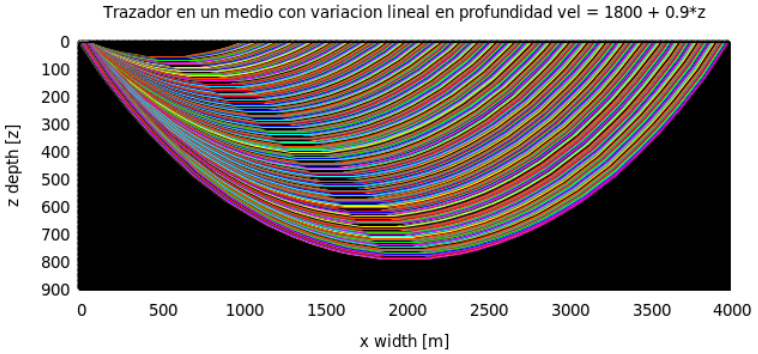


(c) 10 disparos y 80 rayos por disparo.

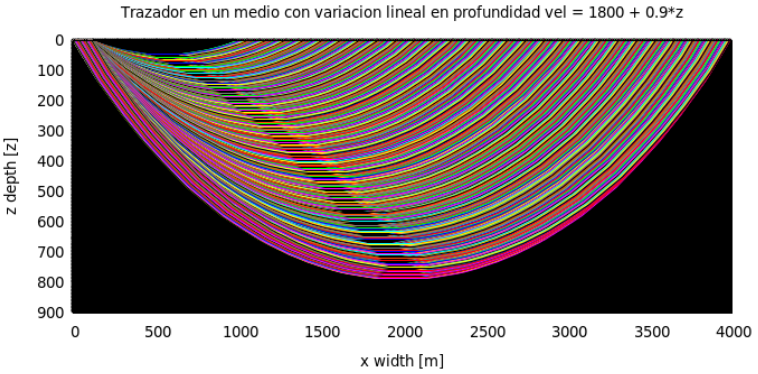
Figura 2.7: Trazado de rayos en un medio con gradiente lineal y una partición de (100×50).



(a) 1 disparo y 240 rayos.



(b) 5 disparos y 240 rayos por disparo



(c) 10 disparos y 240 rayos por disparo.

Figura 2.8: trazado de rayos en un medio con gradiente lineal y una partición de (360×160).

un receptor no era satisfactoria. Para este problema, se usó un criterio en el cual se acepta un punto como un reflector, si la diferencia entre el ángulo de incidencia y el reflejado es menor que $0.05\,rad \approx 2.8°$. Esta condición puede variar significativamente según el radio r de propagación de los rayos, ya que para radios pequeños se barren menor cantidad de ángulos entre $0\,rad$ y $90\,rad$.

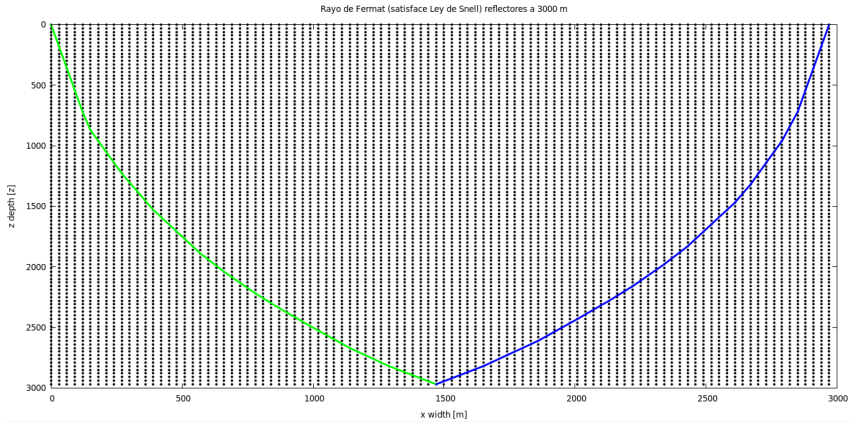
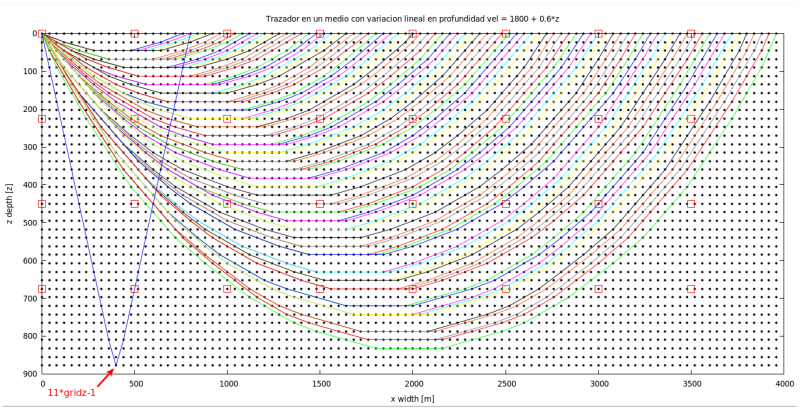
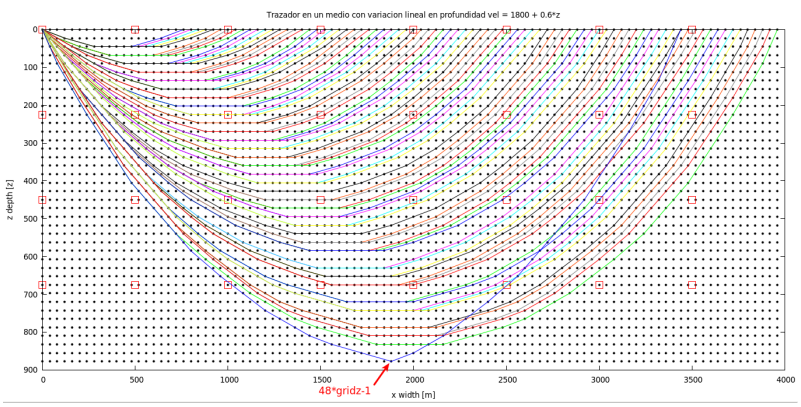


Figura 2.9: Reflexión en una interfaz plana ubicada en el fondo de la región de interés.

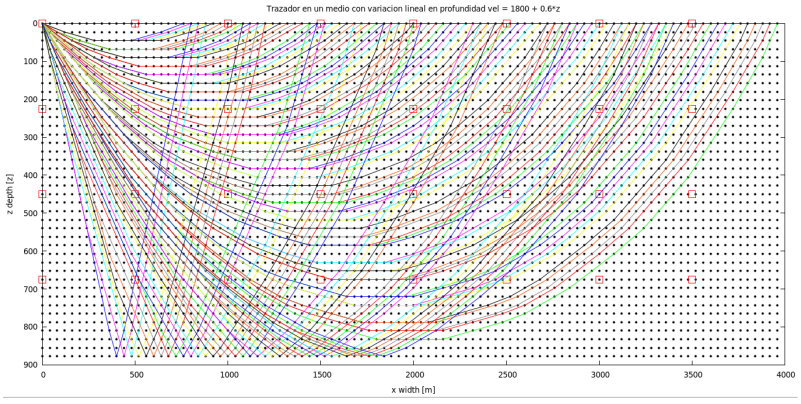
La elección de los reflectores se realiza desde el archivo de parámetros en las variables `REFLECTOR_INICIAL` y `REFLECTOR_FINAL`, la reflexión mostrada en la figura 2.10(a) es para `REFLECTOR_INICIAL=11` y para el `REFLECTOR_FINAL=48` se puede ver la reflexión en 2.10(b). Por último la figura 2.10(c) muestra todos los rayos reflejados entre los nodos reflectores (11 y 48).
En cuestiones de tiempos de computo, las reflexiones son bastantes costosas a comparación de los rayos generados por refracción. Esto, pues es necesario aplicar dos veces la búsqueda del camino mas corto para un solo rayo reflejado, y buscando para cual de los receptores se satisface la Ley de reflexión. Mientras que para encontrar rayos por refracción solo se aplica una vez el algoritmo de búsqueda de las trayectorias mínimas y se pueden encontrar hasta `GRIDX-1` rayos por aplicación.



(a) reflector inicial con $id = 11 * gridz - 1$.



(b) reflector final con $id = 48 * gridz - 1$.



(c) Rayos reflejados empezando en reflector inicial hasta el reflector final.

Figura 2.10: Trazador con rayos reflejados y rayos refractados en un medio con gradiente lineal de 0.6 y una partición de (100×40).

2.3.3. Reflexiones flotantes

Es posible generar rayos que se reflejen en algunos puntos deseados, con el fin de mejorar la iluminación en ciertas regiones y de modelar de forma mas directa los primeros arribos de las reflexiones en las anomalías. Para este fin se escoge en la malla de nodos por donde se trazan los rayos, el `id` en el cual queremos reflejar un rayo. Para trazar los rayos se escoge tanto la fuente como el receptor de la siguiente forma

`source = id_med - k*gridz`, `receptor = id_med + k*gridz`;

donde,

`id_med = (reflector_flo[i]/gridz)*gridz` y `k` $\in \mathbb{Z}$.

Se aplica el algoritmo de Dijkstra, usando como fuente el reflector `reflector_flo[i]` elegido, y trazando las trayectorias desde este punto hasta la fuente y el receptor. Esta forma de hallar la reflexión no garantiza con precisión la ley de reflexión en el reflector, pero es una variante para mejorar la iluminación y encontrar una trayectoria mínima entre una fuente y receptor fijo, para cualquier otro punto de interés en el medio. En la figura 2.11, se observa las reflexiones para 20 reflectores flotantes y cuyos `id's` son los siguientes:

```
reflector_flo[0] = 12300-20;     reflector_flo[8] = 15900-20;
reflector_flo[1] = 12660-20;     reflector_flo[9] = 16500-20;
reflector_flo[2] = 13020-20;     reflector_flo[10] = 12300+4;
reflector_flo[3] = 13380-20;     reflector_flo[11] = 12660+4;
reflector_flo[4] = 14100-20;     reflector_flo[12] = 13020+4;
reflector_flo[5] = 14580-20;     reflector_flo[13] = 13380+4;
reflector_flo[6] = 15060-20;     reflector_flo[14] = 14100+4;
reflector_flo[7] = 15420-20;     reflector_flo[15] = 14580+4;
```

```
reflector_flo[16] = 15060+4;      reflector_flo[19] = 16500+4;
reflector_flo[17] = 15420+4;
reflector_flo[18] = 15900+4;
```

Usando este trazador de rayos se puede obtener una iluminación adecuada del medio para explorar en algoritmos de inversión, en particular, en las técnicas de reconstrucción algebraica (ART y SIRT) basadas en el método de Kaczmarz ([Lo y Inderwiesen 1994]) y que veremos en el siguiente capítulo.

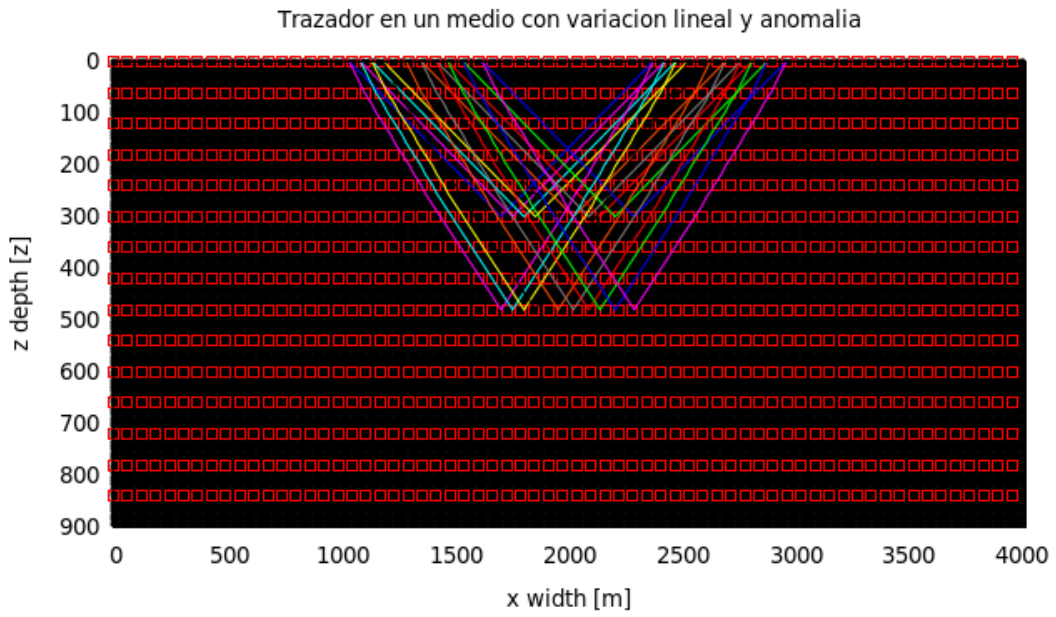


Figura 2.11: reflexión flotantes en diferentes puntos de una región de interés. La malla del trazador es 240×120. Los `id's` de los puntos donde se realizan las reflexiones flotantes varían según la malla elegida en el trazador.

2.4. Estrategia de paralelización

En la actualidad el término "supercomputador" hace referencia a computadores rápidos y poderosos, y generalmente son maquinas que trabajan en paralelo. En aplicaciones científicas y de ingeniería a gran escala, es frecuente la necesidad de usar estos supercomputadores, dada la gran cantidad de tareas que se realizan en datos de investigación y estudio. Por esta razón, surge la pregunta de si las tareas descritas en las secciones anteriores pueden acelerarse mediante el uso de un computador en paralelo. A continuación, se describe una estrategia de paralelización del código `SPM_raytracing.c` para realizar el trazado de rayos en paralelo con la librería `MPI` y que mejora la velocidad de ejecución del trazador de rayos.

En el Modelado descrito en la sección anterior, es necesario ejecutar varias veces la rutina correspondiente al algoritmo de Dijkstra para encontrar las trayectorias de los rayos en el medio de interés. La ejecución de la rutina `Dijkstra.c` se realiza tantas veces como fuentes se quieran para el trazado de rayos por refracción, y para las reflexiones se aplica 2 veces para generar un solo rayo reflejado. Por ejemplo, para el caso de considerar solo refracciones, si se elige `NSHOTS=20` el modelado tendrá 20 fuentes a la izquierda y 20 fuentes a la derecha, para un total de 40 fuentes y en donde para cada fuente se pueden trazar `NRAYS` rayos. Para esta configuración, el total de rayos en el modelado será de `2*NSHOTS*NRAYS`. En el algoritmo en serie, se ejecuta el algoritmo de Dijkstra en una unidad de procesamiento central (CPU), y se ejecuta dicha instrucción una a la vez por fuente, y un tiempo después de que el algoritmo de búsqueda del camino más corto ha terminado, se ejecuta la búsqueda para la siguiente fuente.

Este procedimiento puede resultar costoso computacionalmente cuando el número de rayos y/o disparos aumentan. Una forma de resolver este problema es aprovechar la independencia entre cada uno de estos eventos para diseñar

una estrategia de paralelización eficiente. La estrategia de paralelización es usar simultáneamente múltiples procesadores para realizar el trazado de rayos para optimizar el tiempo de CPU en la ejecución de la rutina `Dijkstra.c`, que es la mas demandante en cuestiones de tiempo, pues realiza una búsqueda intensiva sobre todos los nodos de la malla para determinar las trayectorias con tiempo mínimo entre un nodo y otro. Suponiendo que se tienen P : *procesadores*, la idea es dividir la cantidad de fuentes entre la cantidad de procesadores disponibles,

$$N_i = \frac{2 \cdot \texttt{NSHOTS}}{P}, \quad \text{en el caso que } \texttt{NSHOTS} \text{ sea divisible por } \quad P$$

y así, cada procesador se encargará de calcular las trayectorias de mínimo tiempo para N_i fuentes por procesador y en donde se ejecuta una copia de la rutina de `Dijkstra.c` y de la rutina `ray_reconstruction.c` para reconstruir las trayectorias. El total de rayos trazados por cada procesador, viene dado por:

$$\text{Total de rayos por procesador} = N_i \cdot \texttt{NRAYS}.$$

La organización del algoritmo en paralelo se puede apreciar en la figura 2.12. Para optimizar la velocidad de ejecución es necesario mantener simultáneamente varios procesadores ocupados, dividiendo la cantidad de fuentes lo mas parejo posible entre los procesadores disponibles, obteniéndose el mejor caso, cuando la cantidad de fuentes `NSHOTS` sea divisible por la cantidad de procesadores P. Cuando no ocurre lo anterior, el proceso sufre un **desbalance de carga** que conlleva a un aumento en el tiempo de CPU. La regla para que no ocurra esto es elegir un número de tareas que sea múltiplo entero de el número de procesadores, para que cada procesador disponible ejecute la misma cantidad de tareas. Para el trazado de rayos descrito anteriormente, se debe elegir una cantidad de fuentes que sea múltiplo del número de procesadores P y así garantizar que cada procesador ejecute la búsqueda y la reconstrucción de trayectorias mínimas para la misma cantidad de fuentes.

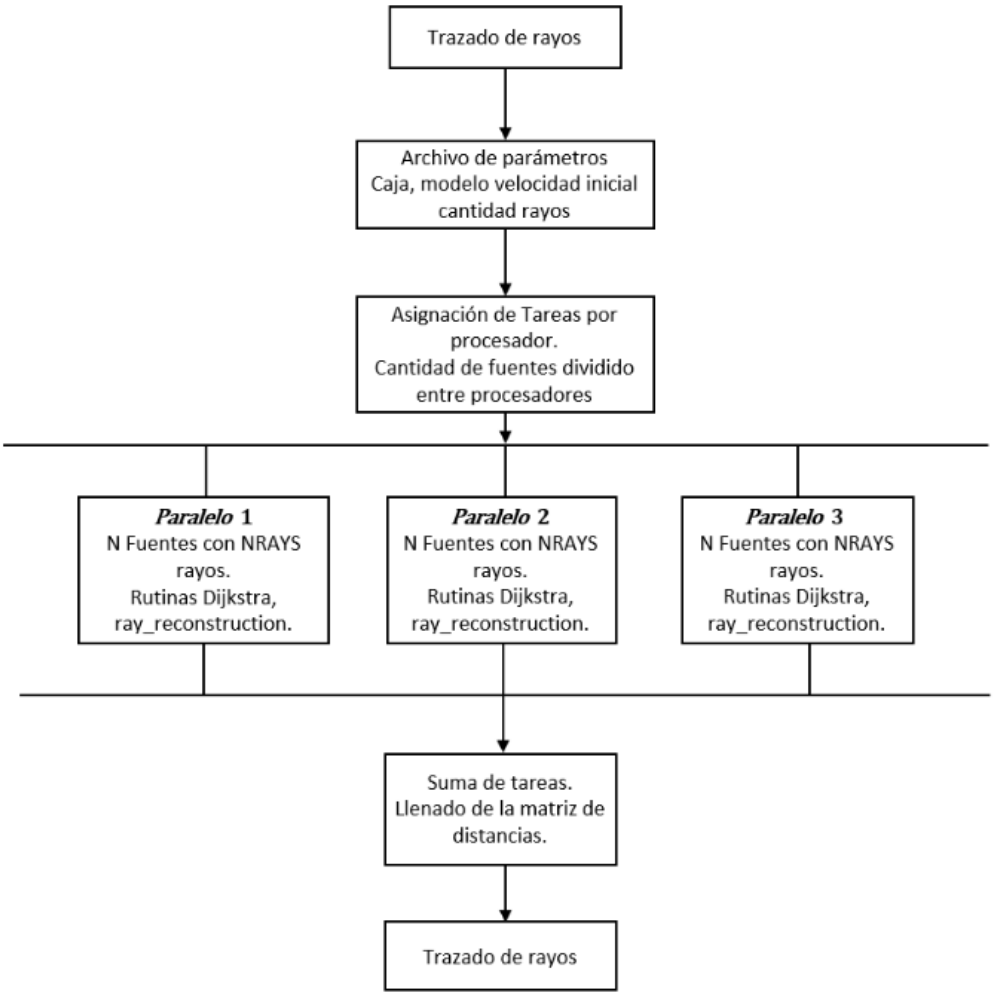


Figura 2.12: Diagrama de flujo para el trazador de rayos en paralelo.

Por ejemplo, si elegimos 22 fuentes (`NSHOTS=11`) para una configuración de procesadores como en la figura 2.13, la forma en que se distribuyen las fuentes entre los procesadores P_1, P_2, P_3 y P_4 es la siguiente: a los procesadores P_1, P_2 y P_3 se le asignan 5 fuentes a cada uno, y al procesador P_4 se le asigna las mismas 5 fuentes incluyendo las 2 fuentes restantes, lo que conlleva a un mayor tiempo de ejecución en este procesador y que en últimas, implica una demora en los procesos de comunicación y de sincronización. Por otro lado, los procesadores P_1, P_2 y P_3 quedarían sin trabajo que realizar hasta que no termine la ejecución en P_4.

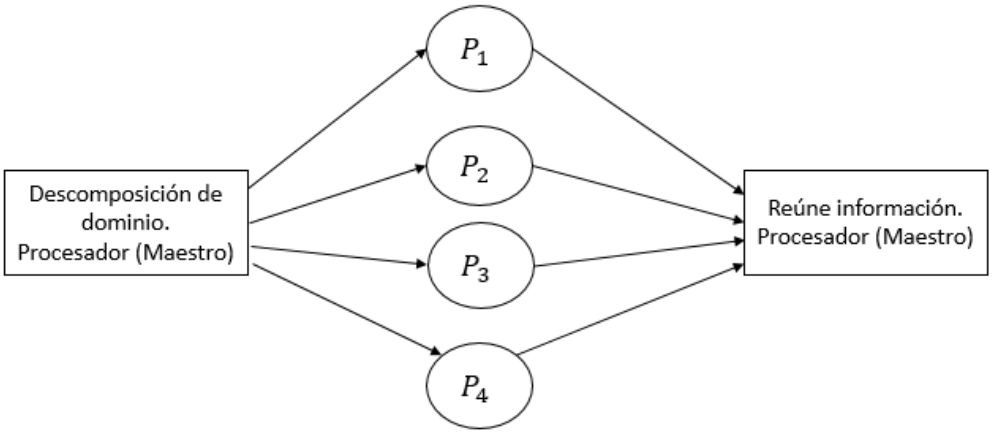


Figura 2.13: Procesador maestro encargado de comunicar y recopilar información.

Otro de los factores determinantes en la velocidad de ejecución de un programa en paralelo, es el proceso de comunicación y sincronización entre procesadores. Ya que pueden afectar notoriamente el tiempo de ejecución. La comunicación es realizada por un nodo procesador llamado maestro, el cual se encarga de descomponer la cantidad de fuentes de manera equitativa y comunicarla a los procesadores P_1, P_2, P_3 y P_4 como en la figura 2.13, y a su vez se

encarga de recopilar la información obtenida por cada uno de los procesadores. Proceder de esta manera conduce a un uso mínimo de procesos de comunicación lo que termina siendo benéfico para el buen rendimiento de la aplicación. La ventaja del trazador de rayos en cuanto a la comunicación, es que los procesos de ejecución del algoritmo de Dijkstra y de reconstrucción del rayo son independientes entre si, y no es necesario realizar procesos de comunicación durante la ejecución de las rutinas. Al final se comunica al nodo maestro los datos obtenidos para iniciar el llenado de la matriz indispensable para estudios de tomografía y el programa implementado se torna nuevamente serial.

Capítulo 3

Resultados en tomografía

En este capitulo presentamos los resultados obtenidos con el trazador simultaneo de rayos refractados y reflejados usando las técnicas de reconstrucción algebraica para la tomografía en modelos sintéticos. Primero se explica el uso de una malla de tomografía con celdas mas grandes que permite que el sistema de ecuaciones lineales con la información del modelo, sea sobredeterminado y sea posible encontrar soluciones aproximadas al problema de tomografía. En la sección 3.2.1 se abarca las soluciones de matrices *sparse* y mal condicionadas, las cuales se resuelven a partir del método de reconstrucción algebraica. Ya en la sección 3.2.1 se muestran los resultados de la inversión basada en modelos aplicada en diferentes modelos sintéticos. Este tipo de inversión es usada en la industria para generar imágenes del subsuelo.

3.1. Algoritmos e implementaciones de tomografía en ANSI C

Las implementaciones de las ecuaciones 1.27 y 1.29 requieren encontrar la matriz de distancias de la ecuación 1.21, es decir, dado un modelo de celdas con velocidad constante, se discrimina para cada rayo la distancia que este recorre

en cada una de las celdas. Para este proceso se usa la rutina `Fill_Matrix(dx, dz, dx_Cell, dz_Cell);` que se encarga de buscar para cada rayo las intersecciones con las celdas del modelo, para luego almacenar en la entrada $S_{\texttt{dist}}(i,j)$ de la matriz de distancias, la longitud del i-ésimo rayo que cruza la celda j.

3.1.1. malla de tomografía y submallas del trazador

Para realizar la tomografía se escogió una malla con celdas mas grandes (menos fina) que la usada en el modelado de trazado de rayos, con el fin de aumentar la cantidad de rayos en el modelo (mayor número de ecuaciones) en comparación con la cantidad de celdas (incógnitas en el sistema de ecuaciones) a tener en cuenta. Esto significa que se puedan trazar mas rayos y así aumentar la iluminación para las celdas de tomografía a considerar. La figura (3.1) ilustra las mallas escogidas para tal fin. Los puntos negros representan los vértices en los cuales se aplica el método del camino mas corto y los rectángulos rojos serán los limites de cada celda para la tomografía.

Luego de tener definida la malla para la tomografía es necesario asignar una velocidad o *slowness* a cada celda. En este paso se implementaron diferentes formas de asignar el *slowness* a la celda de tomografía según los *slowness* de la submalla de trazado de rayos, entre ellas, aproximaciones por splines e interpolación lineal, promedios aritméticos de *slowness* entre los vértices contenidas en cada celda, pero los resultados de la tomografía no fueron satisfactorios. La asignación de *slowness* mas eficaz y natural, y que fue usada para los resultados que se muestran mas adelante, es la de generar el modelo de *slowness* en las celdas de tomografía y a cada punto de la malla donde se trazan los rayos, asignarles el mismo *slowness* de la celda de tomografía que lo contiene. Esta parte la realiza la rutina `slow_cell(dx_Cell,dz_Cell);` donde `dx_Cell` y `dz_Cell` son el largo y el ancho de las celdas de la Tomografía.

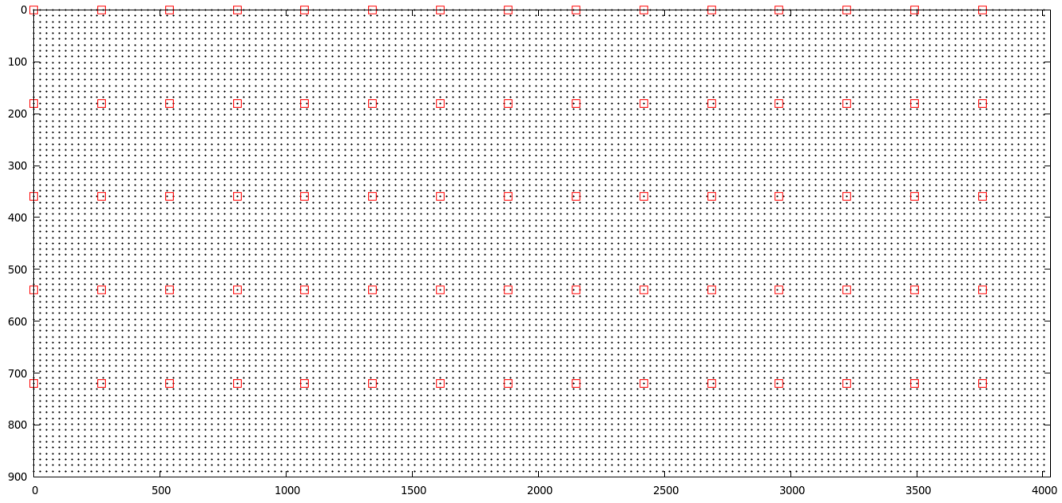


Figura 3.1: Malla para el trazador (puntos negros) y malla para tomografía (rectángulos rojos) .

3.1.2. Llenado de matriz de distancias

La rutina `Fill_Matrix` aprovecha la aproximación al rayo total como una unión entre aristas para encontrar las intersecciones de cada rayo con las celdas por las cuales atraviesa y así determinar las distancias recorridas. Para esto se encuentra la ecuación de la recta que pasa por cada par de nodos que hacen parte del rayo total y que están conectados por una arista (recordar que estos nodos que forman el rayo son generados en el algoritmo de Dijkstra), y luego se determina los puntos de intersección con la celda de tomografía. Por ejemplo, para un segmento de rayo que pasa por los puntos A y B se encuentra la ecuación de recta que pasa por estos dos puntos y se verifica si el segmento $\overline{AB}$ está contenido completamente en la celda actual, en este caso, se calcula la distancia desde A hasta B y se le suma a la distancia recorrida por el rayo en la celda. O si el segmento pasa a otras celdas de tomografía, es necesario hallar el punto de intersección P con la celda en la cual el rayo continua su trayectoria, y cargar la distancia desde A hasta P a la celda actual, y la distancia de P a B

a la nueva celda, y en la cual, se continua con el proceso de llenado de la matriz para el siguiente par de nodos pertenecientes al rayo. Cuando el segmento de rayo corta algún limite de la celda actual, ésta se actualiza para el llenado de distancias dependiendo del lado o vértice donde corte el rayo.
Lo anterior determina el operador de distancias $S_{\texttt{dist}}$ fundamental para realizar los métodos de reconstrucción algebraica ART y SIRT. [Lo y Inderwiesen 1994], [Aster y cols. 2013].

3.2. Resultados

Se presentan los resultados obtenidos para estimar velocidades usando la técnica de reconstrucción algebraica. Primero los resultados del método en la solución de sistemas de ecuaciones lineales 3.2.1 común en los procesos de tomografía sísmica, y segundo los resultados obtenidos en inversión basada en modelos 3.2.2.

3.2.1. Solución de matrices *sparse* usando técnicas de reconstrucción algebraica

El objetivo de estos experimentos es verificar la efectividad del método de inversión para resolver el sistema de ecuaciones lineales $\mathbf{DS} = \mathbf{T}$ descrito en la sección 1.8. Para este fin, se trazaron rayos por refracción (como en la figura 2.7) en los modelos sintéticos (3.2(a), 3.3(a) y 3.4(a)) y se hallaron los tiempos de propagación $\mathbf{T}$ y la matriz de distancias $\mathbf{D}$ respectiva. Para resolver el sistema $\mathbf{DS} = \mathbf{T}$, se usa como modelo inicial el vector $\tilde{\mathbf{S}}$ (3.2(c), 3.3(c) y 3.4(c)) con entradas 0.0 en todas sus componentes. Esto significa que $v \to \infty$, lo cual no tiene sentido físico, pero muestra la solidez del método para encontrar la solución a pesar de tomar dicha condición inicial. El modelo $\tilde{\mathbf{S}}$ se va actualizando a partir de la ecuación 1.30 y dejando fijos la matriz $\mathbf{D}$ y el

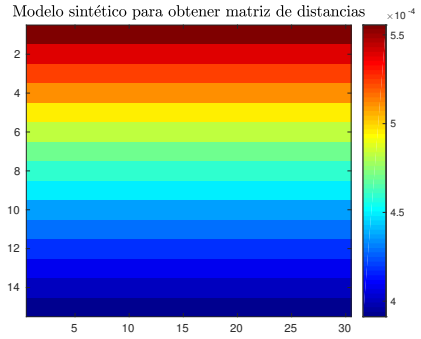
vector $\mathbf{T}$.

Las imágenes 3.2, 3.3 y 3.4 muestran la efectividad de los métodos de reconstrucción algebraica para resolver sistemas de ecuaciones de tamaño $m \times n$, *sparse* y mal condicionados. Y se observa una convergencia de las actualizaciones del modelo $\tilde{\mathbf{S}}$ hacia el modelo verdadero $\mathbf{S}$. Esta convergencia se da en las celdas que son iluminadas por los rayos trazados y en donde también se observa que en las regiones donde no cruza ningún rayo el modelo inicial no sufre ningún tipo de actualización (3.2(b), 3.3(b) y 3.4(b)).

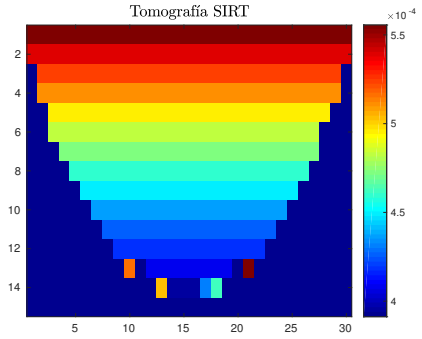
En la figura 3.2, se toma como modelo sintético un medio con velocidad con un gradiente en profundidad $v = 1800 + 1.2 \cdot z$ (aunque toda la parte numérica y de representación de los modelos se realiza con el inverso de la velocidad). La solución del sistema de ecuaciones lineales para este caso es bastante buena ya que recupera todas las capas por donde pasan los rayos refractados. En la figura 3.2(d) se observa la diferencia entre el modelo obtenido por las técnicas de reconstrucción algebraica 3.2(b) y el modelo sintético 3.2(a), el color azul implica que la diferencia entre los modelos es aproximadamente cero. En la parte inferior y limites del trazado de rayos, se puede observar los efectos en la reconstrucción del modelo sintético debidos a la poca iluminación sobre las celdas. A mayor cantidad de rayos atravesando una celda mayor será la velocidad de convergencia del método.

En las figuras 3.3 y 3.4, son mas notorios los efectos de la iluminación en las zonas donde se encuentran las anomalías rectangulares. Los rayos no ingresan en estas zonas debido a que son celdas de baja velocidad, y es una propiedad inherente del trazador de rayos propagarse por las celdas de mayor velocidad, pues son estas zonas las que minimizan el tiempo de propagación del rayo. En la figura 3.4, se observa una correcta aproximación de la anomalía rectangular ubicada en la parte superior del modelo sintético, esto se debe a que se ubicaron receptores en esta zona y los rayos obligatoriamente deben cruzar las celdas de la anomalía superior para llegar a los receptores, lo que no ocurre en las

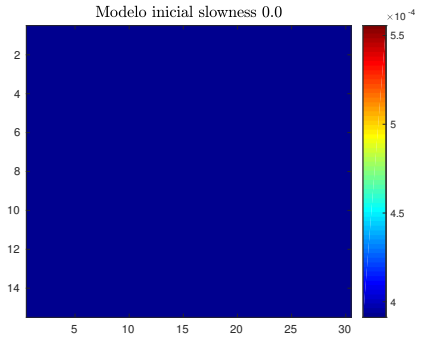
anomalías ubicadas a mayor profundidad.



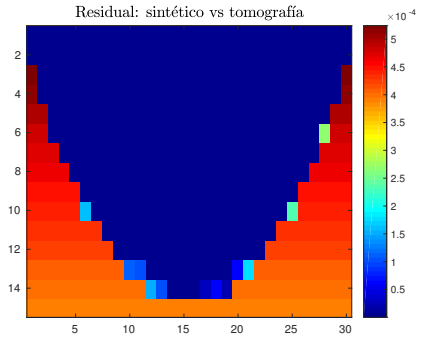
(a) Modelo sintético para obtener los tiempos observados.



(b) Solución del sistema de ecuaciones mediante reconstrucción algebraica.



(c) modelo inicial, celdas con *slowness* = 0.0.

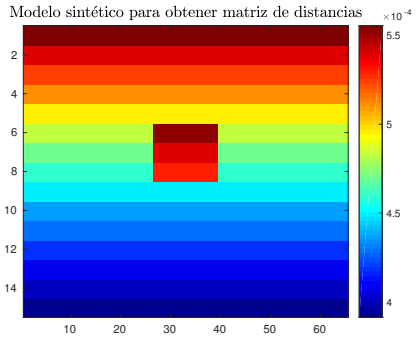


(d) Residual entre modelo sintético y modelo aproximado por las técnicas de reconstrucción.

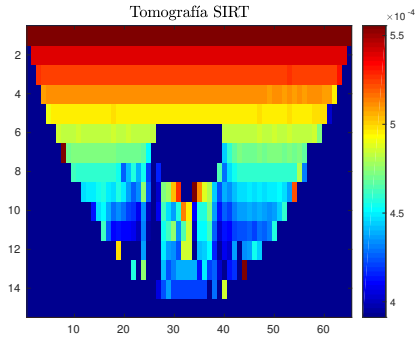
Figura 3.2: Recuperación de un modelo con gradiente lineal a partir de un modelo con *slowness* cero. Solo rayos refractados.

3.2.2. Tomografía SIRT: Inversión basada en modelos.

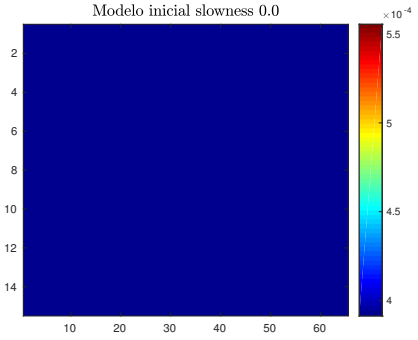
Las imágenes anteriores corresponden a las soluciones de un sistema de ecuaciones lineales *sparse* y mal condicionado de tamaño $m \times n$, en donde la matriz de distancias $\mathbf{D}$ no cambia durante las iteraciones y en la cual se



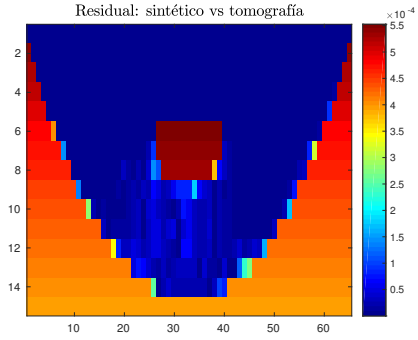
(a) Modelo sintético para obtener los tiempos observados.



(b) Solución del sistema de ecuaciones mediante reconstrucción algebraica.

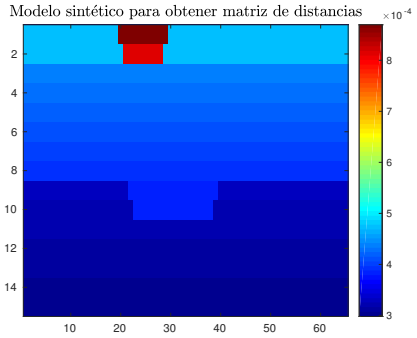


(c) modelo inicial, celdas con *slowness* = 0.0.

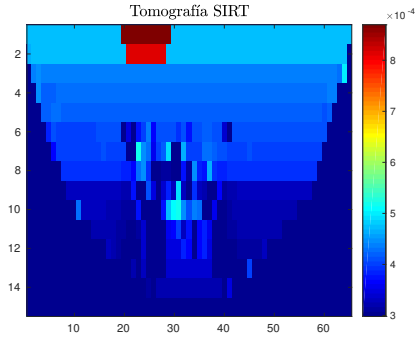


(d) Residual entre modelo sintético y modelo aproximado por las técnicas de reconstrucción.

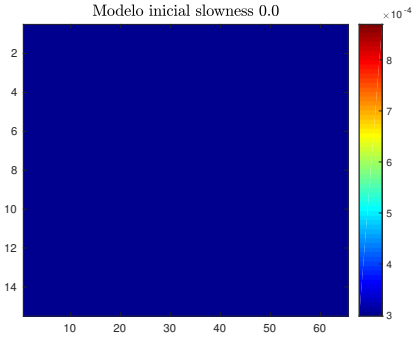
Figura 3.3: Recuperación de un modelo con gradiente lineal y anomalía rectangular a partir de un modelo con *slowness* cero. Solo rayos refractados.



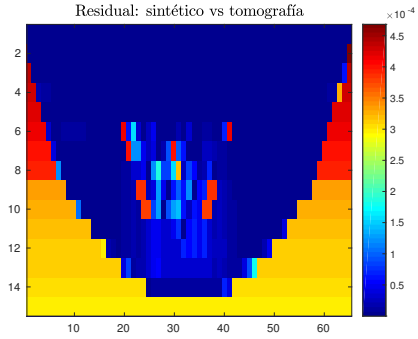
(a) Modelo sintético para obtener los tiempos observados.



(b) Solución del sistema de ecuaciones mediante reconstrucción algebraica.



(c) modelo inicial, celdas con *slowness* = 0.0.



(d) Residual entre modelo sintético y modelo aproximado por las técnicas de reconstrucción.

Figura 3.4: Recuperación de un modelo con anomalías superior e inferior similar al modelo usado por [Zhang y cols. 2004].

verificó que las actualizaciones realizadas al modelo inicial convergen al modelo verdadero.

En los problemas reales de tomografía sísmica se tiene que $\mathbf{D}(\tilde{\mathbf{S}}) = \mathbf{T}$, es decir, la matriz de distancias depende del modelo inicial elegido y por lo tanto debe cambiar en cada actualización del modelo de velocidades. En este caso, el modelo inicial no puede ser un vector de ceros como en el caso anterior 3.2.1, pues los rayos se propagarían todos por la superficie del medio desde las fuentes hasta los receptores. Por esta razón y para garantizar una iluminación similar a la del modelo sintético, el modelo inicial debe tener una configuración cercana al modelo verdadero. El objetivo, es encontrar un modelo que minimice la diferencia entre los tiempos de propagación calculados y los tiempos de propagación observados.

Una simple iteración de los métodos ART y SIRT consiste en realizar una actualización al modelo inicial escogido. En el caso de ART como una contribución rayo por rayo a las celdas del modelo, y en SIRT como un promedio ponderado de todos los rayos que pasan por cada celda del modelo. Luego de la primera actualización del modelo, los rayos son trazados nuevamente usando el modelo ajustado por los métodos de reconstrucción algebraica, y se determinan las nuevas trayectorias de los rayos con la misma configuración de fuentes y receptores. Esto implica que el operador de distancias $\mathbf{D}$ debe ser calculado nuevamente según las nuevas trayectorias en el modelo actualizado. Este proceso se denomina inversión basada en modelos y se repite hasta que se cumpla cierta condición de convergencia con una tolerancia especificada. En las iteraciones se observa una disminución del residuo entre los tiempos observados y los tiempos calculados para cada modelo. Como criterio de parada en las figuras (3.5, 3.6, 3.7, 3.8 y 3.9), se uso el hecho de que el residual disminuye con el paso de las iteraciones, cuando esto deja de ocurrir por mas de 5 iteraciones seguidas el algoritmo se detiene y devuelve la última actualización del modelo. Se realizaron diferentes experimentos de tomografía basada en modelos y se

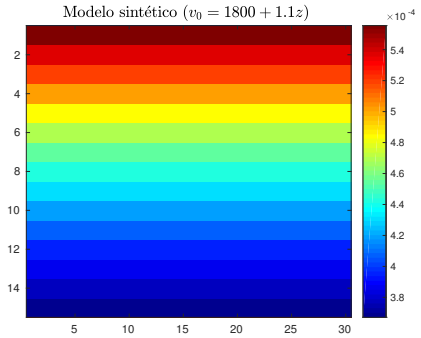
discutirán en las secciones 3.2.3, 3.2.4 y 3.2.5.

3.2.3. Modelo con gradiente lineal

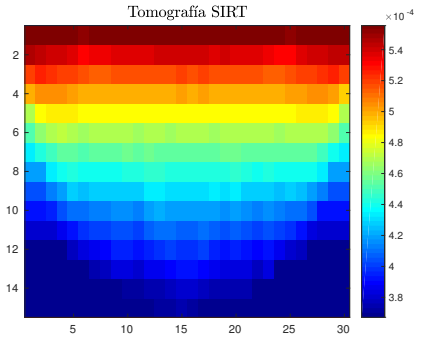
Este primer experimento muestra como se recupera un medio donde la velocidad viene dada como una función lineal. En la figura 3.5, se muestra los resultados obtenidos usando inversión basada en modelos. El modelo sintético tiene una velocidad determinada por la función lineal $v = 1800 + 1.1z$ y fue usado para encontrar los tiempos observados. Para la tomografía se eligió un modelo inicial con velocidad $v = 1800 + 1.4z$, este modelo es diferente en todas las capas al modelo sintético, a excepción de la primera capa ($z = 0.0\,m$) donde la velocidad es de $1800\,m/s$ para ambos. La mejor forma de interpretar los resultados obtenidos, es a partir de la gráfica 3.5(d) que muestra la diferencia entre el modelo sintético 3.5(a) y la tomografía final 3.5(b). Se puede apreciar que en la región por donde pasan los rayos refractados las celdas del modelo se acercan al modelo inicial (entre mas oscuro es el azul, mejor es la solución de tomografía). Como ocurrió en los experimentos de la sección 3.2.1 las esquinas inferiores del modelo de la cual no se tiene iluminación no cambiaron en la inversión. Los residuales entre los tiempos calculados y los tiempos observados en cada iteración, se pueden observar en la figura 3.10(a).

3.2.4. Modelos con gradiente lineal y anomalía rectangular

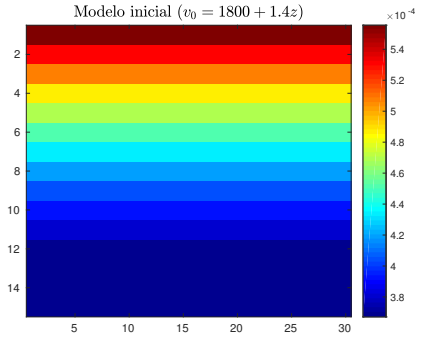
En este experimento se tomó un modelo sintético con gradiente en profundidad dado por $v = 1800 + 0.2z$ y una anomalía rectangular ubicada a mitad de profundidad con una velocidad dada por $v = 1600 + 0.1z$. Para este modelo, se trazaron rayos por refracción y se combinaron con reflexiones flotantes en las caras superior e inferior de la anomalía rectangular, como se muestra en la



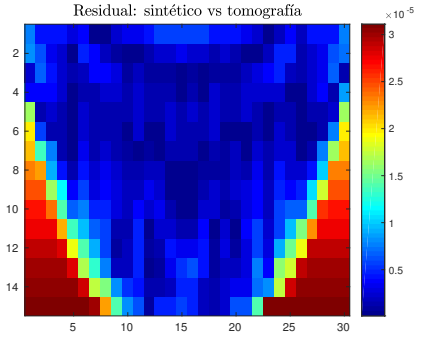
(a) Modelo sintético para obtener los tiempos observados. $v = 1800 + 1.1z$.



(b) Tomografía final usando algoritmo SIRT.



(c) Modelo inicial para calcular tiempos calculados. $v = 1800 + 1.4z$.



(d) Residual entre modelo sintético y tomografía final.

Figura 3.5: Recuperación de un modelos con gradiente lineal partiendo de un modelo de igual cantidad de capas y un gradiente mayor.

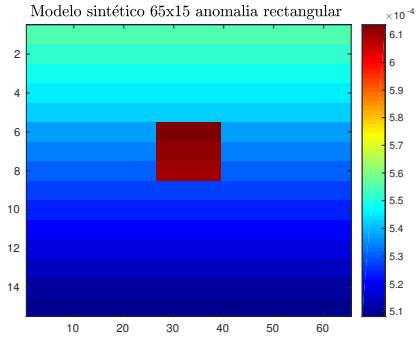
figura 3.7(e). El modelo inicial tomado fue uno con gradiente lineal dado por $v = 1800 + 0.2z$.

La inversión se realizó sobre dos resoluciones para la malla de tomografía, una de tamaño 65×15 en 3.6 y otra de 30×15 en 3.7. En el experimento de mayor resolución (65×15) la tomografía final alcanzó a detectar tenuemente la anomalía rectangular como se puede observar en la figura 3.6(b), cabe anotar, que fueron pocas las iteraciones realizadas por la tomografía en este caso, dado que a partir de los primeros cambios en el modelo inicial el residual entre los tiempos observados y los calculados aumentaban, en la gráfica 3.10(b) se pueden observar la cantidad de iteraciones y como aumenta el residual en este caso.

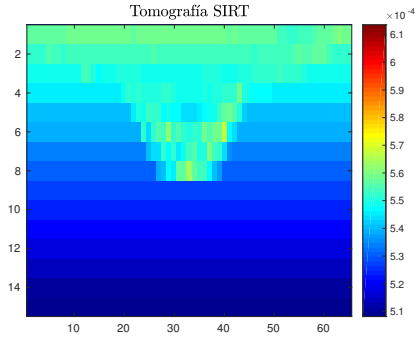
Para la malla de menor resolución (30×15) los resultados fueron mas alentadores, ya que la tomografía final 3.7(b) logró captar con mas detalles casi la totalidad de la anomalía rectangular, aunque no recuperó con precisión la velocidad de las celdas dentro de ésta. Esta mejora en la tomografía, se pudo dar porque las celdas de tomografía son mas grandes comparadas con el experimento anterior, lo que conlleva a que cada celda sea atravesada por una mayor cantidad de rayos. En este caso, se necesitaron mas iteraciones en la inversión y cuyos residuales en cada iteración son mostrados en la gráfica 3.10(c).

3.2.5. Modelos con gradiente lineal y anomalías rectangulares superior e inferior

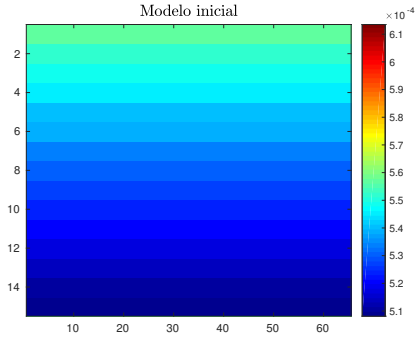
En este experimento se tomó un medio con tres funciones lineales para la velocidad y que dependen de la profundidad de la siguiente forma:
para $z < 120.0$, la velocidad está dada por $v = 1800 + 0.3z$.



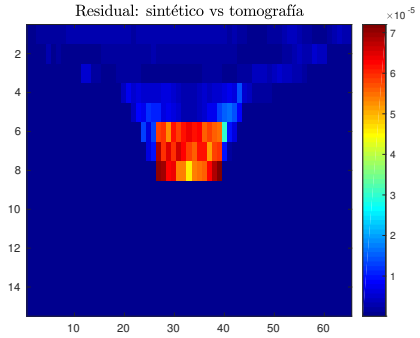
(a) Modelo sintético para obtener los tiempos observados. Velocidad en la anomalía rectangular $v = 1600 + 0.1z$.



(b) Tomografía final usando algoritmo SIRT.

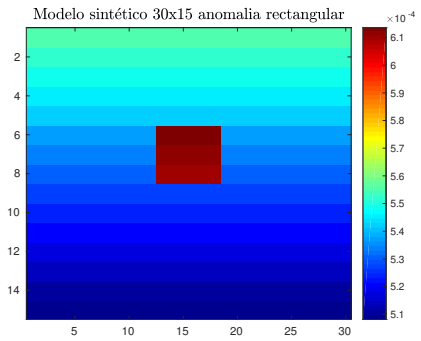


(c) Modelo inicial para calcular tiempos calculados.

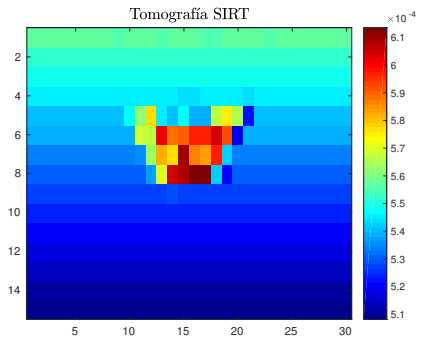


(d) Residual entre modelo sintético y tomografía final.

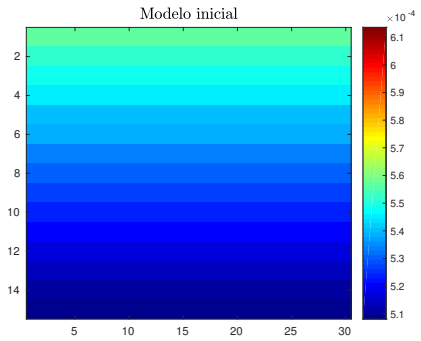
Figura 3.6: Detección de anomalía rectangular. Se usaron pocas refracciones y se combinó con reflexiones flotantes. La resolución del modelo es de 65×15.



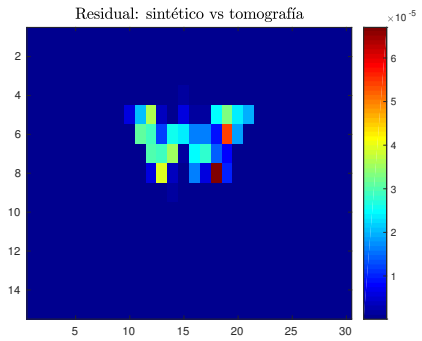
(a) Modelo sintético para obtener los tiempos observados. Velocidad en la anomalía rectangular $v = 1600 + 0.1z$.



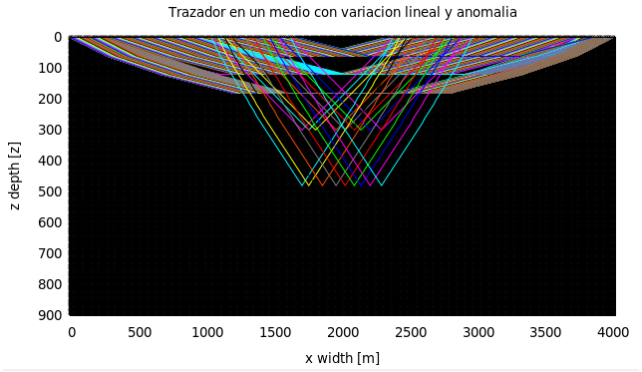
(b) Tomografía final usando algoritmo SIRT.



(c) Modelo inicial para calcular tiempos calculados.



(d) Residual entre modelo sintético y tomografía final.



(e) Trazado de rayos.

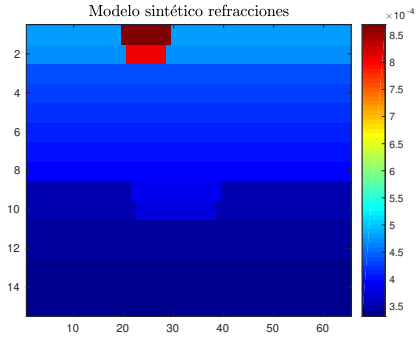
Figura 3.7: Detección de anomalía rectangular. Se usaron pocas refracciones y se combinó con reflexiones flotantes. La resolución del modelo es de 30×15.

para $120.0 \leq z < 480.0$, la velocidad está dada por $v = 2200 + 0.2z$.
para $z > 480.0$, la velocidad está dada por $v = 3000 + 0.05z$.

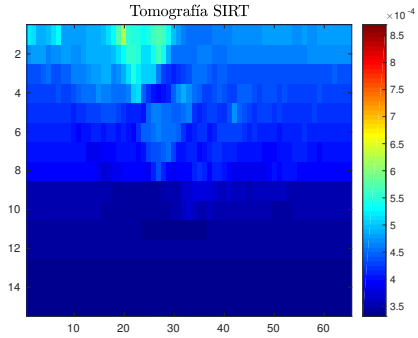
Además de una anomalía en la superficie con velocidad $v = 800 + 0.6z$ y una anomalía inferior con velocidad $v = 2100 + 0.1z$ (Figs. 3.8(a) y 3.9(a)). Como modelo inicial (fig. 3.8(c)) se uso la siguiente configuración de velocidades:
para $z < 48.0$, la velocidad es determinada por $v = 1800 + 0.2z$.
para $z \geq 480.0$, la velocidad es determinada por $v = 3000 + 0.05z$.

Para el resultado obtenido en la figura 3.8 se trazaron solamente rayos por refracción. La tomografía final 3.8(b) en este caso detecta levemente la anomalía superior, y pasa por alto la anomalía inferior. Para la tomografía de la figura 2, se usaron pocos rayos por refracción y varias reflexiones flotantes sobre la anomalía inferior. En este caso se logra detectar la anomalía inferior aunque con poco detalle, pero ya no se detecta la anomalía de la superficie. La cantidad de iteraciones y los residuales se pueden observar en la figuras 3.10(d) y 3.10(e), en donde se observa una disminución del residuo entre los tiempos observados y calculados, pero que no se ve reflejado de la mejor forma en la tomografía final.

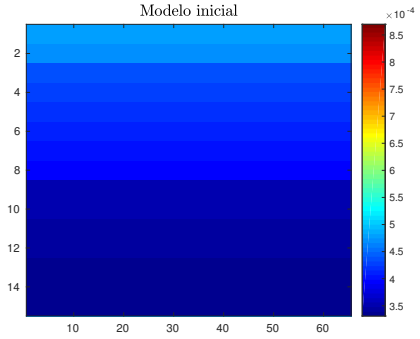
En estos experimentos es necesario explorar (trabajo futuro) con mas detalle como se eligen los parámetros del modelo de velocidades para que la iluminación del medio sea adecuada y que los cambios realizados al modelo en la tomografía no afecten drásticamente el resultado final.



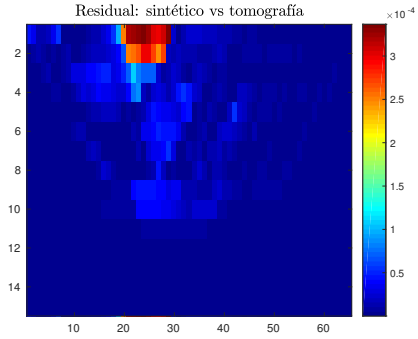
(a) Modelo sintético para obtener los tiempos observados.



(b) Tomografía final usando algoritmo SIRT.

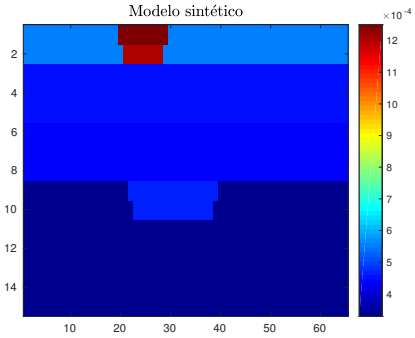


(c) Modelo inicial para calcular tiempos calculados.

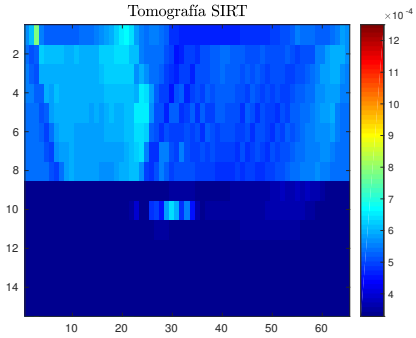


(d) Residual entre modelo sintético y tomografía final.

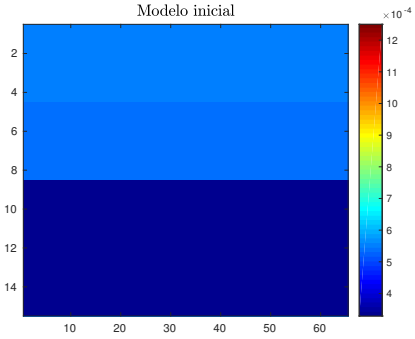
Figura 3.8: Modelo con anomalías superior e inferior similar al modelo usado por [Zhang y cols. 2004]. Solo se usaron refracciones.



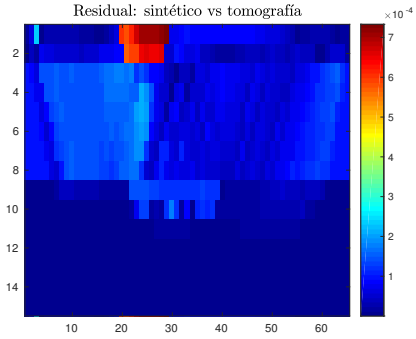
(a) Modelo sintético para obtener los tiempos observados.



(b) Tomografía final usando algoritmo SIRT.

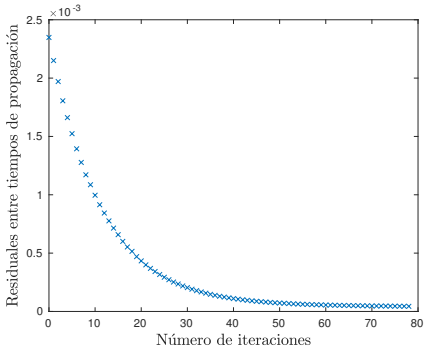


(c) Modelo inicial para calcular tiempos calculados.

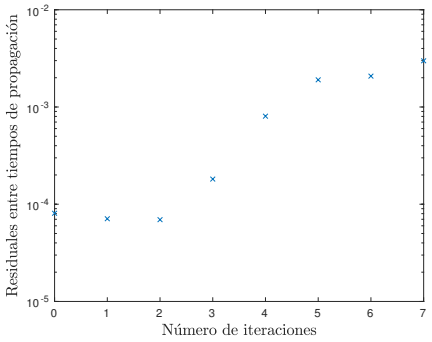


(d) Residual entre modelo sintético y tomografía final.

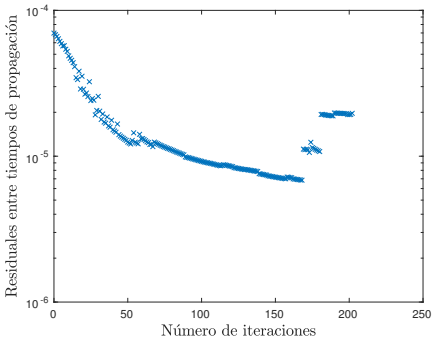
Figura 3.9: Modelo con anomalías superior e inferior similar al modelo usado por [Zhang y cols. 2004]. Solo pocas refracciones y varias reflexiones flotantes.



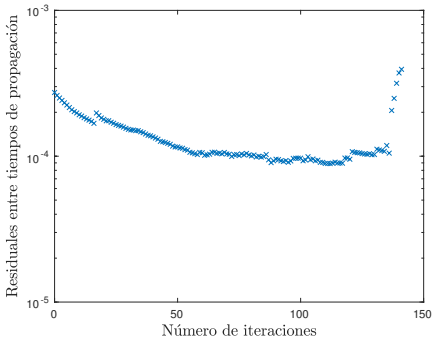
(a) Residuales para el experimento de la gráfica 3.5.



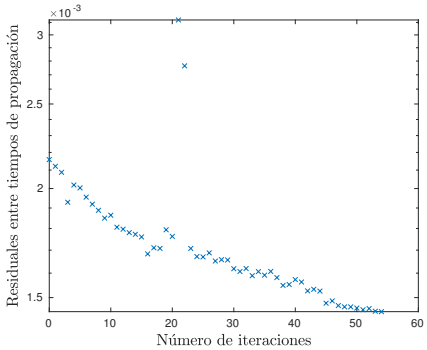
(b) Residuales para el experimento de la gráfica 3.6



(c) Residuales para el experimento de la gráfica 3.7



(d) Residuales para el experimento de la gráfica 3.8



(e) Residuales para el experimento de la gráfica 3.9

Figura 3.10: Gráfica de los residuales entre los tiempos observados y los tiempos calculados.

Capítulo 4

Conclusiones y Discusiones

En este trabajo, se presentó un trazador de rayos basado en el método del camino mas corto, el cual puede modelar simultáneamente rayos por refracción y reflexión, y el cual pudo ser aplicado para encontrar tiempos de propagación de las ondas de primeros arribos en modelos sintéticos, que eventualmente podrían representar versiones simplificadas de geometrías del subsuelo. A partir de esto, se implementó una tomografía de tiempos de propagación usando las técnicas de reconstrucción algebraica, para estimar velocidades de propagación de ondas en el subsuelo, cumpliendo así, con los objetivos propuestos al inicio de este trabajo. Las implementaciones del trazador de rayos y de la tomografía se realizaron en el lenguaje ANSI C y se uso el software Matlab para realizar los gráficos de tomografía. En la sección 2.4 se propone una estrategia de paralelización que mejora el desempeño computacional del trazado de rayos y por ende la ejecución de tomografía basada en modelos.

A continuación discutiremos algunos resultados del modelado y la inversión referentes a este trabajo.

4.1. Modelado

El trazado de rayos implementado es un método flexible a la hora de calcular trayectorias con tiempos de propagación mínimo en una malla de puntos. El hecho de poder ejecutar el algoritmo de Dijkstra para diferentes puntos fuente y apoyados en el principio de Huygens, permite encontrar tanto primeras llegadas del rayo como llegadas secundarias a partir de nuevas ejecuciones del método del camino mas corto entre dos puntos. Es decir, rayos reflejados y refractados simultáneamente.

Dado que el cálculo del camino más corto tiene una dependencia lineal con la cantidad de nodos a tener en cuenta ([Moser 1991]), y además la cantidad de rayos que se pueden trazar depende de la cantidad de nodos en la malla, es necesario tener en cuenta las ventajas y desventajas que se presentan al pasar de una malla de 240×120 a una, por ejemplo, de 480×240. Este aumento en la resolución me permite calcular mas rayos por fuente debido al aumento de posibles nodos receptores en la superficie, pero automáticamente nuestro tiempo de ejecución por fuente será mucho mas alto pues el algoritmo de Dijkstra deberá recorrer muchos más nodos para determinar las trayectorias de mínimo tiempo.

El método del camino más corto es bastante eficiente a la hora de calcular rayos por refracción, ya que con una sola ejecución del método del camino mas corto, se pueden hallar hasta $N-1$ rayos en el caso de que cualquier punto de la malla pueda ser elegido como receptor, pero para efectos reales se pueden trazar hasta `GRIDX-1` (puntos en la superficie) rayos por ejecución. El tiempo de cómputo para calcular reflexiones ya no es tan buena, debido a que se requieren hasta dos ejecuciones del método, para determinar la trayectoria mínima entre fuente-reflector y reflector-receptor, y donde solo se puede encontrar una trayectoria en particular entre los puntos fuente-reflector-receptor.

4.2. Problema inverso

Se verificó la efectividad de las implementaciones realizadas de los métodos de reconstrucción algebraica para resolver sistemas de ecuaciones lineales mal condicionados y *sparse* (ver figuras 3.2, 3.3, 3.4), ya que se logra una recuperación total del medio partiendo desde un modelo inicial con *slowness* cero en todas las celdas. Para que la solución sea exitosa, es necesario que cada celda o pixel sea atravesado por al menos un rayo (preferiblemente mas de uno). En la figura 3.3, se puede observar como la anomalía rectangular no cambia durante el trazado de rayos y esto se debe a que esa región es de baja velocidad y las celdas contiguas tienen una velocidad mayor, y es una característica del método del camino mas corto que los rayos trazados se propaguen por regiones de alta velocidad, ya que en esas regiones su tiempo de propagación es menor, lo que dificulta la iluminación de este tipo de anomalías. La iluminación es clave en la obtención de una buena imagen de tomografía, ya que como en la figura 3.3 los rayos no atravesaron las celdas en esa zona de baja velocidad, las técnicas de reconstrucción algebraica no pueden dar cuenta de estas celdas, lo que conduce a un mal resultado de inversión. Para mejorar los resultados, los modelos iniciales que se escogieron son muy cercanos al modelo sintético como tal.

Para la implementación en celdas de tomografía, se nos presentaron bastantes desafíos con respecto a la cantidad de celdas y la resolución de las mismas para realizar la tomografía. Uno de los inconvenientes principales fue la cantidad de rayos que se podían trazar basados en el método del camino mas corto, ya que si las celdas de tomografía eran del mismo tamaño que las celdas del trazador, implicaba que el sistema de ecuaciones que se obtenía era subdeterminado dada la cantidad de rayos que se podían trazar (solo en aquellos puntos de la superficie) con respecto al total de celdas de tomografía, lo que producía que la cantidad de ecuaciones era mucho menor que el número de

incógnitas. El problema se pudo resolver en gran medida, tomando celdas de tomografía mucho mas grandes y disminuyendo así la cantidad de incógnitas en el problema inverso.

En un trabajo futuro se pueden explorar otras técnicas de tomografía usando el trazador de rayos construido en este trabajo el cual es bastante versátil y que posee características de la teoría de rayos importantes a la hora de hacer tomografía, ya que permite modelar dos tipos de rayos sísmicos simultáneamente, generar rayos con varias fuentes y varios receptores, escoger puntos en interfaces de interés para generar reflexiones en dichos puntos. Estudiar con mas detalle la selección de los parámetros del modelado y la inversión para entender los efectos que pueden tener en los modelos de velocidades obtenidos en la tomografía.

Bibliografía

[Albertin y cols. 2002] Albertin, A., Kapoor, J., Randall, R., Smith, M., Brown, G., Soufleris, C., Whitfield, P., Dewey, F., Farnsworth, J., Grubitz, G., et al. (2002). *La era de las imágenes en escala de profundidad*. Oilfield Review, 14(1), 2–17.

[Aster y cols. 2013] Aster, R. C., Borchers, B., und Thurber, C. H. (2013). *Chapter Six - Iterative Methods*. En Aster, R. C., Borchers, B., und Thurber, C. H. (eds.): *Parameter Estimation and Inverse Problems (Second Edition)*, págs. 141 – 168. Boston: Academic Press, Second Edition ed.

[Bishop y cols. 1985] Bishop, T., Bube, K., Cutler, R., Langan, R., Love, P., Resnick, J., Shuey, R., Spindler, D., und Wyld, H. (1985). *Tomographic determination of velocity and depth in laterally varying media*. Geophysics, 50(6), 903–923.

[Castillo y cols. 2013] Castillo, J. et al. (2013). *Reducción de los tiempos de cómputo de la Migración Sísmica usando FPGAs y GPGPUs: Un artículo de revisión*.

[Cerveny 2005] Cerveny, V. (2005). *Seismic ray theory*: Cambridge university press.

[Cheng y House 1996] Cheng, N. und House, L. (1996). *Minimum traveltime calculation in 3-D graph theory*. Geophysics, 61(6), 1895–1898.

[Dijkstra 1959] Dijkstra, E. W. (1959). *A note on two problems in connexion with graphs*. Numerische mathematik, 1(1), 269–271.

[Dobróka y Szegedi 2014] Dobróka, M. und Szegedi, H. (2014). *On the generalization of seismic tomography algorithms*. American Journal of Computational Mathematics, 4(1), 37–46.

[Faria y Stoffa 1994] Faria, E. L. und Stoffa, P. L. (1994). *Traveltime computation in transversely isotropic media*. Geophysics, 59(2), 272–281.

[Fichtner 2010] Fichtner, A. (2010). *Full seismic waveform modelling and inversion*: Springer Science & Business Media.

[Fischer y Lees 1993] Fischer, R. und Lees, J. M. (1993). *Shortest path ray tracing with sparse graphs*. Geophysics, 58(7), 987–996.

[Fu y Hanafy 2017] Fu, L. und Hanafy, S. M. (2017). *Ray-tracing traveltime tomography versus wave-equation traveltime inversion for near-surface seismic land data*. Interpretation, 5(3), SO11–SO19.

[Gallo y Pallottino 1986] Gallo, G. und Pallottino, S. (1986). *Shortest path methods: A unifying approach*. Netflow at Pisa, págs. 38–64.

[Giroux y Larouche 2013] Giroux, B. und Larouche, B. (2013). *Task-parallel implementation of 3D shortest path raytracing for geophysical applications*. Computers & Geosciences, 54, 130–141.

[Gubbins 2004] Gubbins, D. (2004). *Time series analysis and inverse theory for geophysicists*: Cambridge University Press.

[Harish y Narayanan 2007] Harish, P. und Narayanan, P. (2007). *Accelerating large graph algorithms on the GPU using CUDA*. En *International Conference on High-Performance Computing*, págs. 197–208. Springer.

[Hernández y cols. 2006] Hernández, J. J., Schmitz, M., Audemard, F., und Malavé, G. (2006). *Marco conceptual del proyecto de microzonificación de Caracas y Barquisimeto*. En *VIII Congreso Venezolano de Sismología e Ingeniería Sísmica, Valencia*.

[Hole y cols. 1992] Hole, J., Clowes, R., und Ellis, R. (1992). *Interface inversion using broadside seismic refraction data and three-dimensional travel time calculations*. Journal of Geophysical Research: Solid Earth, 97(B3), 3417–3429.

[Hole y Zelt 1995] Hole, J. und Zelt, B. (1995). *3-D finite-difference reflection traveltimes*. Geophysical Journal International, 121(2), 427–434.

[Huang y cols. 2017] Huang, G., Zhou, B., Li, H., und Nobes, D. C. (2017). *Seismic traveltime inversion based on tomographic equation without integral terms*. Computers & Geosciences, 104, 29–34.

[Jones 2010] Jones, I. F. (2010). *Tutorial: Velocity estimation via ray-based tomography*. first break, 28(2).

[Klimes y Kvasnicka 1994] Klimes, L. und Kvasnicka, M. (1994). *3-D network ray tracing*. Geophysical Journal International, 116(3), 726–738.

[Lo y Inderwiesen 1994] Lo, T. w. und Inderwiesen, P. (1994). *Geophysical Monograph Series*. Nº 6: Society of Exploration Geophysicists.

[Malony y cols. 2016a] Malony, A. D., McCumsey, S., Byrnes, J., Rasmusen, C., Rasmusen, S., Keever, E., und Toomey, D. (2016a). *A Data Parallel Algorithm for Seismic Raytracing*. En *International Conference on Vector and Parallel Processing*, págs. 89–98. Springer.

[Malony y cols. 2016b] Malony, A. D., Monil, M. A. H., Rasmusen, C., Huck, K., Byrnes, J., und Toomey, D. (2016b). *Towards Scaling Parallel Seismic Raytracing*. En *Computational Science and Engineering (CSE) and*

IEEE Intl Conference on Embedded and Ubiquitous Computing (EUC) and 15th Intl Symposium on Distributed Computing and Applications for Business Engineering (DCABES), 2016 IEEE Intl Conference on, págs. 225–233. IEEE.

[Margrave 2001] Margrave, G. (2001). *Numerical methods of exploration seismology with algorithms in Matlab: The University of Calgary*.

[Matsuoka y Ezaka 1992] Matsuoka, T. und Ezaka, T. (1992). *Ray tracing using reciprocity*. Geophysics, 57(2), 326–333.

[Meléndez y cols. 2015] Meléndez, A., Korenaga, J., Sallarés, V., Miniussi, A., und Ranero, C. R. (2015). *TOMO3D: 3-D joint refraction and reflection traveltime tomography parallel code for active-source seismic data-synthetic test*. Geophysical Journal International, 203(1), 158–174.

[Menke 2012] Menke, W. (2012). *Geophysical data analysis: discrete inverse theory: MATLAB edition*, vol. 45: Academic press.

[Monil y cols. 2018] Monil, M. A. H., Malony, A. D., Toomey, D., und Huck, K. (2018). *Stingray-HPC: A Scalable Parallel Seismic Raytracing System*. En *Parallel, Distributed and Network-based Processing (PDP), 2018 26th Euromicro International Conference on*, págs. 204–213. IEEE.

[Monsegny y Agudelo 2013] Monsegny, J. und Agudelo, W. (2013). *Shortest path ray tracing on parallel GPU devices*. En *2013 SEG Annual Meeting*. Society of Exploration Geophysicists.

[Moser 1992] Moser, T. J. (1992). *The shortest path method for seismic ray tracing in complicated media*. Geologica Ultraiectina, 83, 1–180.

[Moser 1991] Moser, T. (1991). *Shortest path calculation of seismic rays*. Geophysics, 56(1), 59–67.

[Nakanishi y Yamaguchi 1986] Nakanishi, I. und Yamaguchi, K. (1986). *A numerical experiment on nonlinear image reconstruction from first-arrival times for two-dimensional island arc structure.*. Journal of Physics of the Earth, 34(2), 195–201.

[Nolet 2012] Nolet, G. (2012). *Seismic tomography: with applications in global seismology and exploration geophysics*, vol. 5: Springer Science & Business Media.

[Orellana y Hamos] Orellana, Y. und Hamos, H. *Algunas aplicaciones del método de regularización de Tikhonov*.

[Parker 1994] Parker, R. L. (1994). *Geophysical inverse theory*: Princeton university press.

[PIETA y DWORNIK 2012] PIETA, A. und DWORNIK, M. (2012). *Parallel implementation of ray tracing procedure in anisotropic medium*. TASK QUARTERLY, 16(1), 135–143.

[Pilipenko 1983] Pilipenko, V. (1983). *Numerical example of time-field method in up-to-date seismic prospecting*. Izv. Akad. Nauk SSSR, Fizika Zemli, (1), 36–42.

[Podvin y Lecomte 1991] Podvin, P. und Lecomte, I. (1991). *Finite difference computation of traveltimes in very contrasted velocity models: a massively parallel approach and its associated tools*. Geophysical Journal International, 105(1), 271–284.

[Qin y cols. 1992] Qin, F., Luo, Y., Olsen, K. B., Cai, W., und Schuster, G. T. (1992). *Finite-difference solution of the eikonal equation along expanding wavefronts*. Geophysics, 57(3), 478–487.

[Reshef y Kosloff 1986] Reshef, M. und Kosloff, D. (1986). *Migration of common-shot gathers*. Geophysics, 51(2), 324–331.

[Riahi y Juhlin 1994] Riahi, M. A. und Juhlin, C. (1994). *3-D interpretation of reflected arrival times by finite-difference techniques*. Geophysics, 59(5), 844–849.

[Rodríguez-Pérez 2007] Rodríguez-Pérez, Q. (2007). *Estructura tridimensional de velocidades para el sureste de México, mediante el análisis de trazado de rayos sísmicos de sismos regionales*. Tesis de Doctorado, M. Sc. thesis, 83 pp., Instituto de Geofísica, Universidad Nacional Autónoma de México, Mexico City, Mexico.[Links].

[Sava y Fomel 2001] Sava, P. und Fomel, S. (2001). *3-D traveltime computation using Huygens wavefront tracing*. Geophysics, 66(3), 883–889.

[Scales 1987] Scales, J. A. (1987). *Tomographic inversion via the conjugate gradient method*. Geophysics, 52(2), 179–185.

[Schneider Jr 1995] Schneider Jr, W. A. (1995). *Robust and efficient upwind finite-difference traveltime calculations in three dimensions*. Geophysics, 60(4), 1108–1117.

[Sen 2006] Sen, M. K. (2006). *Seismic inversion*: Richardson, TX.: Society of Petroleum Engineers.

[Snieder y Trampert 1999] Snieder, R. und Trampert, J. (1999). *Inverse problems in geophysics*. En *Wavefield inversion*, págs. 119–190: Springer.

[Tarantola 1984] Tarantola, A. (1984). *Linearized inversion of seismic reflection data*. Geophysical prospecting, 32(6), 998–1015.

[Van Trier y Symes 1991] Van Trier, J. und Symes, W. W. (1991). *Upwind finite-difference calculation of traveltimes*. Geophysics, 56(6), 812–821.

[Vidale 1988] Vidale, J. (1988). *Finite-difference calculation of travel times*. Bulletin of the Seismological Society of America, 78(6), 2062–2076.

[Vidale 1990] Vidale, J. E. (1990). *Finite-difference calculation of traveltimes in three dimensions*. Geophysics, 55(5), 521–526.

[Zhang y cols. 2004] Zhang, J.-Z., Chen, S.-J., und Xu, C.-W. (2004). *A method of shortest path raytracing with dynamic networks*. Chinese Journal of Geophysics, 47(5), 1013–1018.

[ZHANG y cols. 2006] ZHANG, M.-G., CHENG, B.-J., LI, X.-F., und WANG, M.-Y. (2006). *A fast algorithm of the shortest path ray tracing*. Chinese Journal of Geophysics, 49(5), 1315–1323.

[Zhang y cols. 2017] Zhang, M., Xu, T., Bai, Z., Liu, Y., Hou, J., und Yu, G. (2017). *Ray tracing of turning wave in elliptically anisotropic media with an irregular surface*. Earthquake Science, 30(5-6), 219–228.

[Zhang y cols. 2018] Zhang, X., Bai, Z., Xu, T., Gao, R., Li, Q., Hou, J., und Badal, J. (2018). *Joint tomographic inversion of first-arrival and reflection traveltimes for recovering 2-D seismic velocity structure with an irregular free surface*. Earth and Planetary Physics, 2(3), 220–230.