UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA
DIRECCIÓN
60-I-1553

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

SR. OSCAR SUÁREZ SERRANO
P r e s e n t e

En atención a su solicitud, me es grato hacer de su conocimiento el tema que propuso el profesor Ing. Eduardo Amador Terrazas y que aprobó esta Dirección para que lo desarrolle usted como tesina de su examen profesional de Ingeniero Geofísico:

MODELADO Y PROCESAMIENTO DE DATOS CON SEISMIC UNIX

Ruego a usted cumplir con la disposición de la Dirección General de la Administración Escolar en el sentido de que se imprima en lugar visible de cada ejemplar del informe el título de éste..

Asimismo, le recuerdo que la Ley de Profesiones estipula que se deberá prestar servicio social durante un tiempo mínimo de seis meses como requisito para sustentar examen profesional.

A t e n t a m e n t e
"POR MI RAZA HABLARÁ EL ESPÍRITU"
CD. Universitaria, D. F., a 4 de Diciembre de 2008
EL DIRECTOR

MTRO. JOSÉ GONZALO GUERRERO ZEPEDA

JGGZ*RJPYS*srs

UNIVERSIDAD NACIONAL AUTONOMA

FACULTAD DE INGENIERIA

MODELADO Y PROCESAMIENTO DE DATOS CON SEISMIC UNIX

T E S I S

QUE PARA OBTENER EL TITULO DE:

I N G E N I E R O G E O F I S I C O

P R E S E N T A :

OSCAR SUAREZ SERRANO

MEXICO, D.F. **2009**

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA
DIRECCIÓN
60-I-1553

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

SR. OSCAR SUÁREZ SERRANO
P r e s e n t e

En atención a su solicitud, me es grato hacer de su conocimiento el tema que propuso el profesor Ing. Eduardo Amador Terrazas y que aprobó esta Dirección para que lo desarrolle usted como tesina de su examen profesional de Ingeniero Geofísico:

MODELADO Y PROCESAMIENTO DE DATOS CON SEISMIC UNIX

Ruego a usted cumplir con la disposición de la Dirección General de la Administración Escolar en el sentido de que se imprima en lugar visible de cada ejemplar del informe el título de éste..

Asimismo, le recuerdo que la Ley de Profesiones estipula que se deberá prestar servicio social durante un tiempo mínimo de seis meses como requisito para sustentar examen profesional.

A t e n t a m e n t e
"POR MI RAZA HABLARÁ EL ESPÍRITU"
CD. Universitaria, D. F., a 4 de Diciembre de 2008
EL DIRECTOR

MTRO. JOSÉ GONZALO GUERRERO ZEPEDA

JGGZ*RJPYS*srs

AGRADECIMIENTOS

Esta tesis está dedicada a mis padres, a quien les agradezco de todo corazón por su amor, cariño y comprensión. Siempre estarán en mi corazón.

Agradezco a mis hermanas por su compañía y apoyo, se que siempre contare con ellas.

Agradezco a mi tía Juana y a su familia por el apoyo que siempre me han brindado.

Agradezco a mis amigos de todas mis etapas como estudiante, en especial a mis amigos del cubo, se que siempre podre contar con ellos.

Agradezco a mis profesores que tuve a lo largo de los años, gracias a ellos esto es posible.

Principalmente agradezco a Dios por darme la vida y ponerme en una familia tan hermosa, por haberme brindado tantas dichas y bendiciones.

Dedicada a mis padres y hermanas ya que sin su apoyo esto nunca hubiera sido posible.

INDICE

Resumen

Seismic Unix (SU) es un paquete de software libre el cual fue creado y desarrollado por la escuela de Minas de Colorado, este paquete nos sirve entre otras cosas para el modelado y procesado de datos sísmicos. El código fuente de SU es abierto, lo que significa que cualquier usuario con conocimientos de programación puede modificarlo de acuerdo a sus necesidades, esta características es lo que distingue a SU de las demás aplicaciones para el modelo y procesamiento de datos sísmicos, ya que al ser este gratuito podemos usarlo sin ninguna restricción y lo principal es que no tenemos que estar pagando licencias que como sabemos en algunos casos son un poco costosas, otra característica importante que tiene infinidad de aplicaciones desarrolladas por terceras personas, que hacen tareas cada vez más elaboradas y complejas, además de que podemos desarrollar nuestras propias aplicaciones.

El modelado sísmico, y a su vez, el procesamiento de datos sintéticos son instrumentos que nos dan una posible respuesta sísmica de un medio que conocemos en este caso con los modelos que son propuestos. Estos principios generan una visión que es más fácil de asimilar de los fundamentos teóricos que son usados en el proceso de adquisición y en el procesamiento de datos reales, siguiendo este contexto, gracias a los datos sintéticos podemos descifrar modelos de datos reales mucho más complejos, también gracias a estos podemos establecer estrategias más eficientes que en su vez disminuyen el riesgo en la toma de decisiones.

Utilizando los programas o rutinas de Seismic Unix, en este trabajo se realiza un modelado básico de datos sísmicos y su correspondiente secuencia de procesamiento, hasta llegar a la sección apilada, para poder llegar a esto se genera una representación 2D del subsuelo lo cual se hace a través de la construcción de perfiles de propiedades del subsuelo, seguidamente se realiza un modelado de la propagación de ondas acústicas en los perfiles creados anteriormente.

Este modelo es realizado un par de veces, simulando de esta forma el proceso de adquisición de una línea sísmica. A los datos obtenidos se les aplica una secuencia básica de procesos, para poder llegara a una sección apilada.

El procedimiento explicado anteriormente es realizado para tres modelos que son propuestos a lo largo de este trabajo, primero tenemos el modelo de ejemplo en cual consta de estratos horizontales y otros.

Las imágenes obtenidas al final del procesado de cada uno de los modelos concuerdan, con los modelos propuestos, es no quiere decir que si manejamos adecuadamente los herramientas de Seismic Unix podemos llegar a obtener resultados muy confiables.

Por último se muestra un Código donde se encuentran todos los pasos para llegar a obtener la imagen de los modelos de entrada que deseemos.

CAPITULO I

1. INTRODUCCION

Introducción

Seismic Unix (SU) es un paquete se software, que fue creado por el Centro para Fenómenos Ondulatorios (CWP) de la Escuela de Minas de Colorado (CEM).

Seismic Unix opera como una extensión del sistema operativo UNIX. Este paquete es código abierto el cual es constantemente actualizado. Hoy en día, Seismic Unix tiene la bondad de que se puede realizar tanto el procesamiento como el modelado de los datos sísmicos, y como es un programa freeware, lo podemos modificar según nuestras necesidades o realizar aplicaciones más complejas, claro con el consentimiento previo del autor, por lo regular los autores no ponen ninguna restricción al hacer esto, solo basta con avisarles y mandarles una copia de los cambios que se realizaron.

Este programa, casi no es empleado por la industria porque su manejo resulta un poco tedioso su, está desarrollado para que se emplee en lo que es principalmente el área docente (investigación, área académica, alumnos, etc.), a pesar de que es una paquetería con muchas aplicaciones no es usado en la industria porque su manipulación es un poco complicada, ya que esta se hace vía comandos y no gráficamente. Es por esa razón principalmente que no es ocupado en la industria. A pesar de de ello no le resta importancia ya que esta paquetería es muy importante para la investigación en universidades, esta es la principal herramienta que utilizan los investigadores alrededor del mundo así como estudiantes.

El presente trabajo tiene como finalidad explicar el uso de esta herramienta, con Seismic Unix realizaremos el procesamiento y modelado sísmico, en este sentido, este trabajo pretende crear un conjunto de datos sintéticos, en caso de que no tuviéramos datos reales, con su correspondiente secuencia de proceso con el fin de:

- Conocer algunas funciones de Seismic Unix
- Contar con grupo de Scripts que puedan servir de base para el desarrollo de otros con un mayor grado de complejidad
- Generar una buena documentación que permita una interacción más cómoda de usuarios con el paquete Seismic Unix, para que se manipulación no se les haga demasiado compleja.

Estas metas serán alcanzadas bajo el siguiente esquema de trabajo:

1. Programación de una serie se Scripts para la realización de modelos simples del subsuelo.
2. Programación de Scripts para la generación de datos sintéticos pre apilados a partir de modelos del subsuelo creados en el punto anterior.
3. Programación de Scripts para la aplicación de secuencias de procesamiento a los datos creados en el punto anterior.

El contenido de este trabajo se encuentra se dividido en una serie de capítulos.

A continuación se recita una breve reseña de cada uno de ellos, tratando de dar una explicación detallada de las ideas principales de cada capítulo.

- Para iniciar el trabajo me parecía importante aunar un poco más a fondo en lo que es Seismic Unix .Para no extenderme demasiado en este capítulo, se centrara en ofrecer los instrumentos necesarios para que los lectores, por si mismos puedan hallar la información que les sea de utilidad.

  Para ello se describen las funciones (comandos) de ayuda más importantes de Seismic Unix .También se dan una serie de referencias bibliográficas y links de internet que son de gran utilidad.

- Capitulo 3: En este capítulo comenzaremos a describir las herramientas de Seismic Unix y los procedimientos que serán empleados para el correcto cumplimiento de las metas antes señaladas. En este sentido, iniciamos las explicaciones de los comandos destinados al modelado sísmico. Unos de los comandos que describiremos es el UNIF2, el cual es empleado para realizar perfiles del subsuelo.

  Seguidamente me refiero al comando SUFDMOD2, el cual permite modelar la propagación de ondas acústicas en un medio, el cual es representado por los perfiles creados con el comando UNIF2.

- Capitulo 4: En este apartado se realizara un modelado de una adquisición sísmica 2D. Básicamente se mostrara un Script que recrea, a través de los siguientes comandos UNIF2 y SUFDMOD2, la propagación de las ondas acústicas en un conjunto de puntos que representan la posición de la fuente en el proceso de adquisición de una línea sísmica.
  Este procedimiento dará como resultado un conjunto de Tiros Sintéticos relacionados con cada disparo de la adquisición modelada.

- Capitulo 5: En este apartado será destinado a explicar la forma de hacer un procedimiento básico con los programas contenidos en SU. Extendiendo la idea, se utilizaran y explicaran algunos de los comandos destinados a: visualización y ajuste del encabezado, análisis de velocidad, corrección NMO y apilamiento. Para este fin se muestra un conjunto de Scripts que realizan dichas y además se muestran las figuras que resumen los resultados de cada uno de los Scripts.

- Capitulo 6: Ya con las herramientas claras, en este apartado se describe la forma de obtener la sección apilada a partir de datos 2D que fueron creados a través del modelado de adquisición sísmica realizado en la capitulo 4.

  Luego adecuaremos algunos de los procedimientos prendidos en secciones anteriores para el correcto manejo de las datos sísmicos las cuales están formadas por varios Tiros Sintéticos, en otras palabras en este capítulo procesaremos el conjunto de Tiros Sintéticos, obtenidos en el capítulo cuarto con el único fin de obtener la sección apilada.

- Capitulo 7: Casi ya a para finalizar este trabajo vamos a abordar un par de ejemplos, en los cuales se encuentran envueltos todos los procedimientos explicados a lo largo de este trabajo. Aunque solo se mostraran los resultados obtenidos, no se muestra la metodología ya que este fue explicada durante todo el proceso del trabajo, se muestran los resultados de 2 modelos propuestos que son un modelo que simula un Valle y una Falla Simple.

- Por último en el apéndice D se muestra el script final, con el cual se realizaron los dos últimos modelos, en este Script solo basta con cambiar algunos de los parámetros iníciales (modelo de entrada, velocidades, densidades) para poder obtener la imagen ya procesada, al final de este Script se obtiene una imagen.

CAPITULO II

2. Principios de Seismic Unix

2.1 Contenido del paquete SU

2.2 Instalación de SU

2.3 Manejo de las funciones

2.4 Ayuda en SU

2.5 Formato utilizado por SU

2.6 Limitaciones de SU

2.7 Principios de Linux y Shell-Scripting

2.7.1 Que es Linux

2.7.2 La terminal de Linux

2.7.3 Shell-Scripting

Capitulo 2

Seismic Unix

Para finales de la década de los 80's, específicamente para 1987, en el Centro de Fenómenos Ondulatorios "Center Waves Phenomenos" de la Escuela de Minas de Colorado, se estaba formando un grupo de trabajo, liderado por Jack Cohen y Shuki Ronen, destinado a la generación de un conjunto de herramientas computacionales para las tareas relativas al procesamiento sísmico, todas estas herramientas pensadas para trabajar en un ambiente tipo UNIX. Es así como nace Seismic Unix, el cual es gratuito, este paquete trabaja como una extensión del Shell de UNIX (o cualquier derivación de este por ejemplo Linux).

Para descargar este paquete solo basta con dirigirnos a la siguiente dirección electrónica **http: //www.cwp.mines.edu/cwpcodes/** , en donde estará disponible la última versión de este al momento de estar realizando este trabajo se encontraba disponible la versión 4.0.

SU posee ayudas, tutoriales y documentación de los programas con que cuenta, para que usuarios novatos puedan empezar a usarlos; sin embargo no exploraremos ese tema en este trabajo, para mayor información puede revisar la referencia bibliográfica [11] donde se explica en detalle todo lo relacionado al manejo se SU.

2.1 Contenido del paquete SU

De manera muy general, el paquete contiene herramientas para:

- Procesamiento sísmico:
  1. Visualización y ajuste del encabezado (encabezado).
  2. Orden del Punto Medio Común CMP).
  3. Análisis de velocidad
  4. Corrección por decaimiento con la distancia (NMO).
  5. Apilamiento
  6. Migración

- Modelado:

  1. Creación de perfiles de propiedades del subsuelo: Modelos uniformemente muestreados y modelos triangulares generados con el método de triangulación Delaunay (B. Delaunay, 1934).

2. Generación de los datos sísmicos sintéticos a partir de los modelos creados en el punto anterior:

- Mediante el modelado de la propagación de ondas acústicas utilizando el método de diferencias finitas (William F. Ames, 1977).
- Mediante trazado de rayo.
- Manipulación de la data sísmica (traza) en formato SEG-Y.
- Manejo de la transformada de Fourier.
- Filtrado de señales.
- Transformadas de la ondicula.
- Realización de graficas en formato PostScript.
- Despliegue de gráficos en pantalla (x-windows graphics)

Para obtener mayor información sobre todo lo que ofrece el paquete SU favor de acudir a la referencia bibliográfica [3].

2.2 Obtener e instalar SU

Podemos obtener **SU** de la siguiente dirección electrónica **http:www.cwp.mines.edu/cwpcodes/** o de uno de los discos incluidos en este trabajo con el nombre de utilidades, la versión que contiene este disco es la versión 4.0.

Los requisitos para poder realizar tal instalación son:

- Una maquina con procesador x86, x64 o Power PC, con cualquier sistemas operativo instalado ya sea Linux o Windows , memoria RAM de 64, monitor a color SVGA, tarjeta de red , conexión a internet , disco duro de 10 GB, teclado y ratón.
- Un compilador de C-ANSI todos los ambientes ya lo traen de fábrica ya sea Windows, Linux o MacosX.

Pasos para una correcta instalación de SU, para dicha instalación en este caso use Linux, en su variante de Ubuntu 7.04 (Feisty Faw) o posterior, he resumido toda la instalación en las siguientes líneas en caso de que se tangan problemas, revisar el apéndice [1] en donde explico paso a paso de una forma gráfica como realizar, la instalación desde cero.

Los pasos a seguir son los siguientes.

Antes que nada tener instalado en la maquina Ubuntu en se versión 7.04 o posterior, este sistema es muy parecido a Windows por lo que no creo que haya problema en su manejo, en caso de no saber cómo instalarlo en el apéndice [1], explico cómo hacerlo, ya que tenemos este también necesitaremos el paquete SU, el cual podemos descargar de la dirección antes mencionada.

Primer paso abriremos una terminal en Ubuntu en donde pondremos lo siguiente.

cd / ----- para cambiar el directorio raíz.

sudo mkdir /cwp ----- para crear la carpeta donde se instalara SU.

sudo chown nombre de usuario /cwp ----- guarda las cambios en el usuario que va a trabajar.

Ya que hemos hecho esto moveremos el archivo que descargamos a la carpeta que acabamos de crear o sea en cwp, en caso de tener problemas podemos moverla con el siguiente comando, **sudo nautilus**.

Paso siguiente tendremos que modificar el archivo environment el cual se encuentra en /etc, para ello haremos lo siguiente, en una terminal escribiremos.

cd /etc ----- nos cambia el directorio etc.

sudo cp environment environment.backup ----- crea una copia del archivo por si tenemos que remplazarlo.

Ahora abriremos el archivo con un editor de texto para cambiarlo de la siguiente forma.

sudo gedit environment

Agregamos '**YOUR_CWPROOT/bin**' al final de la línea PATH (quitando las comillas), en mí caso que así '**:/cwp/bin'**

Agregamos una nueva línea al final del archivo '**CWPROOT="/cwp"** '(sin comillas)

Salvamos y salimos del editor

Cerramos y abrimos sesión para revisar que se hayan efectuado los cambios.

echo $PATH

echo $CWPROOT

Ahora verificaremos que tengamos instaladas las siguientes librerías en la maquina, lesstif2 lesstif2-dev, g77, gcc, libglu1-mesa, libglu1-mesa-dev, freeglut3, freeglut.dev, libxmu6, libxmu-dev, libxi6, libxi1-dev, xlibs xlibs-dev, lesstif, lesstif-dev, xlibmesa-glu, libglut3 , motif, es recomendable tener todas estas librerías , pero en cierta maquinas algunas no están disponibles, pero aun así SU trabaja perfectamente.

Lo siguiente que tendremos que hacer es descomprimir el archivo que copiamos en la carpeta cwp , nos vamos a dicha carpeta desde una terminal lo hacemos de la siguiente forma.

cd /

cd /cwp

Ejecutamos el siguiente comando.

tar zxfv cwp_su_all_40.tgz ----- este comando descomprime el archivo.

Nos creara una carpeta llamada src, en la cual tendremos que modificar el archivo Makefile.config con un editor de texto.

sudo gedit Makefile.config

Lo editamos de la siguiente forma:

- En la línea en donde está escrito esto ENDIANFLAG=-DCWP_BIG_ENDIAN agregáramos un # al principio de la misma.
- En la línea donde tenemos CC=cc ponemos CC=gcc
- En la línea donde tenemos OPTF agregamos un # al principio.
- Agregamos un # en la línea donde aparezca FC=ifort
- En la sección de MOTIF agregamos lo siguiente al final.
  ° IMOTIF= /usr/include
  ° LMOTIF = /usr/lib
- Salvamos y quitamos el editor

Lo único que nos queda es instalar SU de la siguiente forma.

En una terminal de Linux estando en la carpeta src en donde está el archivo Makefile tecleamos lo siguiente.

- **make install**
- **make xtinstall**
- **make make xminstall**
- **make finstall**
- **make mglinstall**
- **make utils**

De esta forma se instala SU en cualquier plataforma de Linux, si has tenido problemas con la instalación revisa el apéndice [1], en l disco de utilidades están todos los archivos ya modificados para que solo los sustituyas por los tuyos.

2.3 Ayuda y referencias.

Seismic Unix puede ser pensado como un lenguaje (o metalenguaje). En este sentido, como en cualquier lenguaje, existe una cierta cantidad de vocabulario que debe ser dominado antes de poder realizar operaciones útiles. Como SU presenta una extensa cantidad de programas, resulta de interés la existencia de comandos que enumeren y expliquen las utilidades y el vocabulario más importante. En torno a esto, el paquete Seismic Unix presenta una serie de programas destinados a brindar ayuda y documentación relacionada con las funciones contenidas. Estas facilidades pueden ser obtenidas escribiendo los siguientes comandos en un terminal de LINUX: [referencia libro de SU]
A continuación de describen algunos comandos con los cuales podemos obtener ayuda de SU, a mi parecer estos son los más importantes para conseguir ayuda en SU.

- **suhelp**: A través de este comando son desplegados, en el terminal, los programas y Shell Scripts más importantes.

- **suname**: Despliega una descripción corta de los programas y la localización del código fuente.

- **sudoc**: Muestra la documentación de cada programa, Shell Script y funciones de biblioteca principales. Descripción de los comandos y de sus parámetros más importantes.

  Se debe colocar, en el terminal: **sudoc** nombre_del_programa. De esta manera se desplegara la información referente al comando tecleado en la terminal.

- **gendocs**: Genera una lista completa de los programas con su descripción corta en formato LATEX.

- **sufind**: A partir de una cadena de caracteres dada, busca los programas y Shell Scripts en cuya documentación aparezca el string especificado. Esta función es de gran importancia cuando se quiere encontrar un programa destinado a un uso específico. Por ejemplo, supongamos que se quiere realizar una corrección NMO y no se conoce el nombre del comando (programa) que realiza esta tarea. Para resolver este problema se puede colocar "**sufind** nmo" y se desplegarán todos los programas y Shell Scripts que, en su documentación, contengan la cadena de caracteres "nmo".

Además de los comandos citados anteriormente, existen otras categorías de ayuda, como son las siguientes.

- **demos**: Seismic Unix también contiene una serie de demostraciones del manejo de sus programas, que como ya sabemos debe realizarse a través de Shell Scripting. Esta, sin duda alguna es la herramienta de ayuda más importante que ofrece el SU. Esta facilidad consiste en una serie de Shell Scripts localizados en $CWPROOT/src/demos.

  Estos demos pueden ser ejecutados con solo leer las instrucciones establecidas en el archivo "README" de cada clasificación. Las ejecuciones brindarán resultan de gran utilidad para entender el funcionamiento de los programas.

- **examples**: En el mismo contexto que los "demos", esta utilidad consiste en una serie de scripts que muestran el manejo de los programas del SU. Se encuentra en el directorio $CWPROOT/src/su/examples. Aquí se muestran ejemplos más elaborados (de pocas aplicaciones) pero con la diferencia de que no son ejecutables de forma inmediata. Es necesario introducir los datos o archivos de entrada pertinentes y si no se tiene una experiencia previa en el manejo de las funciones resulta difícil lidiar con estos Scripts.

Además de estas ayudas tenemos otras, como son las bibliográficas o el manual de SU.

- http://sepwww.stanford.edu/oldsep/cliner/files/suhelp/suhelp.html
  En donde podemos encontrar una buena clasificación de los comandos de SU, con su respectivas instrucciones para su correcto funcionamiento.
- Otra referencia importante es el libro (notas del curso) de SU "The New SU User's Manual; John W. Stockwell, Jr. And Jack K. Cohen, 2005 en donde podemos encontrar toda lo relacionado con SU.

2.4 Uso de las funciones.

Como se menciono anteriormente, SU opera como una extensión del sistema o lenguaje de programación basado en Unix (cualquier distribución de Linux).

Es por esta razón que el manejo de las funciones de Seismic Unix debe realizarse a través del Shell Scripting , si aun no tiene claro que es el Shell Scripting o es nuevo en esto, le recomiendo que lea el apéndice [2], para una mejor comprensión de este.

2.5 Formato utilizado por Seismic Unix

Formato SEG-Y y el formato de datos de SU

El formato SU consiste en que las trazas del formato SEG-Y están escritas en formato binario es el cual es el lenguaje nativo de la maquina.

A continuación se describe más ampliamente el formato SEG-Y.

El formato SEG-Y está conformado por tres partes:

- La primera parte es un header que consiste en 3200 bytes (conformado por 40 líneas de texto con 80 caracteres por línea) que describe las características de la cinta.

- La segunda parte consiste en un header binario de 40 bytes que contiene información sobre cómo fue grabada la cinta.

- La tercera porción del formato SEG-Y contiene las trazas sísmicas. Cada traza posee 240 bytes de header.

El formato SU está basado en la porción de las trazas del formato SEG-Y. La principal diferencia entre las trazas con formato SU y las trazas con formato SEG-Y, es que la porción de datos de las trazas SU se encuentran escritas en el formato binario.

El formato SU consiste solo en las trazas SEG-Y, las otras dos partes no son preservadas. Por lo tanto un archivo con formato SEG-Y no podrá ser utilizado para ejecutar los programas del paquete SU. Para convertir una data con formato SEG-Y a formato SU se puede emplear el comando SEGYREAD.

Los programas del paquete de Seismic Unix para el manejo de datos sísmicos, requieren que las trazas sísmicas se encuentren en el formato de SU para que se puedan manipular de manera correcta, dentro de SU se encuentran el comando para poder realizar la conversión a *.su, además dentro de los recursos que posee SU hay varias rutinas ya implementadas por terceras

personas para poder convertir casi cualquier formato sísmico al formato que maneja SU.

Todos los programas que comienzan con “su” se encuentran dentro de las siguientes posibilidades.

1. Requieren un archivo de entrada en formato SU.
2. Retornan un archivo en formato SU.
3. Requieren y retornan archivos en formato SU.

Los programas que escapan a la característica mencionada anteriormente, es decir los que no comienzan con “**SU**” manejan los datos en forma binaria *.dat

2.6 Limitaciones de SU

Seismic Unix no es como la mayoría de las herramientas para el manejo de datos, ya que este no cuenta con una interfaz grafica (GUI “Graphical User Interface”), aunque esta limitación no lo hace menos importante que los demás paquetes computacionales, existen proyectos en los cuales se está tratando de implementarle una interfaz grafica , con lo cual en un futuro podremos disponer de SU con una interfaz muy intuitiva, pero esta también trae consigo limitaciones a la hora se usar SU, ya que como mencionamos anteriormente esta paquetería trabaja como extensión de Unix,

y el uso de una interfaz grafica limitara el acceso total de las funciones que ofrece este sistema , con lo cual restringiremos las capacidades propias de SU.

Otra limitación importante es que SU no está implementado para aplicaciones 3D de alto nivel en su última versión la cual es 4.0, pero hay trabajos hechos por terceras personas con los cuales podremos tener aplicaciones 3D en SU. Un punto fuerte para SU es que es una excelente herramienta para procesar datos 2D de las diferentes áreas de la geofísica.

2.7 Principios de Linux y Shell-Scripting.

Todas las aplicaciones de SU corren en el sistema operativo UNIX, lo que hace necesario manejar los fundamentos básicos de este. En otras palabras, SU opera bajo la llamada a comandos específicos en el entorno de una terminal[1] (ventana parecida a la de MS-DOS), de un sistema operativo UNIX[2], o una derivación de este. SU no tiene entorno grafico propio, lo que implica una manera de trabajo del tipo de línea de comandos.

Para tener una base solida a la hora de trabajar con SU, necesitaremos aprender a manejar lo que llamamos terminal de Unix (en Linux es lo mismo ya que esta posee las mismas características tanto en Linux como Unix), y saber generar los llamados Shell-Scripting, que son un conjunto de comandos las cuales están en un solo archivo de texto, con las cuales podremos regenerar una rutina o rutinas dentro de la terminal. En este sentido daremos una breve explicación del sistema operativo Linux[3].

2.7.1 ¿Que es Linux?

Linux es un sistema operativo, el cual es mantenido por miles de programadores alrededor del mundo, existen varias distribuciones de Linux, en este caso usaremos una de las distribuciones, que más se asemejan a Windows[4,] la distribución que usaremos es Ubuntu en si versión 7.04, cabe mencionar que esta distribución es actualizada cada 6 meses.

Dentro de Linux tenemos algo llamado núcleo “kernel”, lo cual es el núcleo de la maquina, el kernel es el que permite la interacción de las dispositivos y el software de la maquina, este corre a nivel usuario , este kernel fue concebido por Linus Torvalds de allí es que recibe el nombre de Linux.

El sistema operativo como tal es producto del trabajo de muchos desarrolladores a nivel mundial, en el marco del proyecto llamado GNU[5]. Formalmente, todas las distribuciones que existen actualmente del sistema operativo Linux deberán ser llamadas GNU/Linux; sin embargo, se ha popularizado mas el segundo nombre.

Linux es un sistema operativo de código abierto “open source”, lo que implica que su código fuente está disponible para sus usuarios, y puede ser modificado o alterado a criterio de los mismos para mejorar, crear u optimizar ciertas y determinadas tareas que el sistema requiera. El hecho que Linux sea código abierto ha incentivado a la formación de una comunidad mundial de

programadores y otros usuarios de Linux, que se han dado a la tarea (la mayoría voluntariamente) de ir haciendo a Linux más y más eficiente, estable, multidisciplinario y versátil, mediante la modificación precisamente de su código fuente. También existe en Internet una gran cantidad de información relacionada con casi todos los problemas posibles (desde el menor nivel, hasta los problemas más complicados del manejo) que se presentan al utilizar Linux, y sus posibles soluciones (en caso de que se tengan). A continuación hablaremos sobre las principales características del terminal de Linux, y explicaremos unos pocos de sus comandos principales, necesarios para empezar a trabajar con **SU**.

2.7.2 La terminal de Linux.

En los actuales momentos, con todo el desarrollo que ha tenido y con todas las distribuciones que se han creado, Linux es un sistema operativo sumamente versátil: existen distribuciones de Linux pueden tener un entorno grafico tan amigable como la última versión de Windows (UBUNTU, SUSE, por ejemplo), sin dejar de lado la posibilidad al usuario de utilizar la línea de comando (lo cual se torna en un trabajo mucho más eficiente al adquirir destreza).

De manera formal, un terminal es un intérprete de comandos. Los intérpretes de comandos son especies de compiladores, los cuales traducen y ejecutan, línea a línea, las instrucciones especificadas por el usuario. Leen líneas de texto escritas por el usuario, todo dentro de una secuencia única. A diferencia de los intérpretes de comandos, los compiladores traducen[6] a lenguaje de maquina un código de fuente el cual es almacenado como unarchivo ejecutable, y que es independiente del programa originalmente escrito.

A modo de comparación, un compilador traduce detalladamente un escrito de un idioma a otro, y deja constancia de ello en un documento que denominamos código fuente que en el contexto de programación avanzada se denomina “Script”; en cambio, un intérprete de comandos traduce a otro idioma la conversación de una persona, al tiempo que esta sigue hablando, sin dejar constancia de dicha traducción. En un intérprete “Shell”, a medida que escribes una línea de comandos, esta es traducida (a lenguaje de maquina) y ejecutada conforme sea necesario para el interprete. Este modo de programación ofrece al usuario agilidad y velocidad para realizar tareas cortas y precisas, pero puede tornarse ineficiente y poco práctico para tareas más largas y elaboradas, debído a que el proceso de interpretación es más costoso, en cuanto a tiempo de ejecución, que el proceso de compilación.

La terminal de Linux es un intérprete de comandos que te permite comunicarte con el núcleo de la maquina e interactuar con ciertos dispositivos lógicos que controlan el hardware, así como manejar las aplicaciones del sistema operativo. Básicamente lo que puedes hacer a través del entorno grafico[7], se puede hacer por la terminal, lo que se puede hacer en el terminal no necesariamente se puede hacer en el entorno grafico. Además, una vez aprendidos los comandos básicos y haber ganada un poco de agilidad realizando tareas rutinarias en la terminal, es mucho más eficiente trabajar en la terminal que en el entorno grafíco.

De todas maneras, para manejar SU solo necesitamos aprender pequeñas cosas básicas sobre la terminal, así que de ninguna forma tendremos que memorizar una gran cantidad de comandos para poder aprovechar las herramientas brindadas por Seismic Unix.

Sin embargo, antes de hablar de los comandos, debemos hablar de las redirecciones. En Linux existen dos archivos importantes que tienen que ver con la interacción usuario-maquina, ellos son la entrada estándar y la salida estándar. En la mayoría de los casos, el archivo de entrada estándar corresponde a todos los caracteres que hayan sido escritos por el teclado, es decir, el teclado es la entrada estándar. La salida estándar se refiere a la información que proviene de la maquina hacia el usuario, que generalmente es la que se muestra en la pantalla (monitor).

Las abreviaciones de estos archivos son stdin y stdout. Cualquier entrada del tipo comando o aplicación se especifica como stdin, será por medio del teclado, y cualquier salida que algún comando o aplicación se especifica como stdout, será por la pantalla. Aunque lo dicho atrás es cierto, no necesariamente es definitivo; la terminal le permite al usuario re direccionar tanto la entrada como la salida estándar a archivos, según sea el caso.

Además de eso, el usuario puede enlazar tareas en una misma línea de comandos de la terminal, haciendo uso de las llamadas a nuevas salidas.

El siguiente paso es conocer y familiarizarse con los comandos más usados en el Shell, los cuales te permitirán realizar tareas bastante sencillas, útiles y necesarias para trabajar con Seismic Unix

Lo primero que se debe saber sobre los comandos de la terminal es que estos poseen dos tipos de parámetros de entrada; uno es el argumento, el cual puede ser un archivo, directorio, comando aplicación al cual se le va a ejecutar un comando determinado; es decir, el comando actuara sobre este argumento.

El otro parámetro que recibe un comando son los llamados banderas “flags”, que hacen que el comando funcione de una u otra manera, dependiendo del flag que se escriba. La mayoría de los flags los preceden un signo menos (-).

El comando debe ser escrito seguido de un espacio y a continuación se escriben los parámetros de entrada anteriores, sin importar el orden.

Un comando el cual merece una atención especial es chmod. Este comando modifica los permisos de lectura, escritura y ejecución de los archivos seleccionados además de permitir variables alfanuméricas.

En Linux, todos los archivos tienen unos permisos algunos comunes otros especiales, la cual admite o restringe las acciones especificadas anteriormente.

Si, por ejemplo, un archivo tiene permiso solo de lectura, este puede ser leído por el usuario más no se podrá escribir nada sobre él, ni tampoco podrá ejecutarlo. Además de esto, los permisos son diferentes para cada tipo de usuario. Existen tres tipos de usuarios: **Propietario**, que es el que crea el archivo en su sesión.

Usuario de Grupo, es un conjunto de usuarios agrupados bajo un mismo nombre; **Otros**, que son todos aquellos usuarios que no son ni el propietario ni pertenecen al grupo del propietario.

A la hora de trabajar con los Shell Scripts el cambio de permisos es muy importante, ya que nos permitirá dar permiso de ejecución a los Scripts que hayamos creado. Sin este permiso ningún archivo puede ser ejecutado.

2.7.3 Shell-Scripting.

La programación de Shell Scripts es fundamental a la hora de trabajar de manera efectiva en la terminal. Los llamados Scripts no son más que archivos de texto, los cuales contienen un número determinado de comandos (enlazados o no) que son ejecutados al momento de ser interpretadas por el Shell.

Los Scripts nos permiten darle al Shell dos, tres, cuatros o más líneas de código, las cuales especifiquen secuencias de tareas determinadas. Si es necesario ejecutar un número determinado de veces esta secuencia de tareas sobre argumentos distintos, se multiplica la utilidad del los scripts, ya que ahorran el tiempo de volver a escribir todos los comandos anteriores una y otra vez en el terminal.

El lenguaje de Shell Scripting más ampliamente usado es Bash, el cual viene del acrónimo Bourne-Again-Shell. En la mayoría de las derivaciones de UNIX, Bash es el lenguaje por defecto del Shell. El uso de Shell Scripts nos brinda una manera rápida, aunque menos elegante en comparación con lenguajes compilados, de resolver ciertos problemas computacionales.

Los scripts se vuelven poco prácticos en ciertas circunstancias (manejo de grandes volúmenes de datos, muchas operaciones matemáticas dentro del script, cuando el tiempo de ejecución juega un papel importante en el desempeño de la tarea, entre muchas otras), sin embargo, para trabajar con Seismic Unix los Scripts son esenciales para aprovechar al máximo el alcance del paquete.

Mediante los Scripts se realizan la mayoría de las tareas importantes que se pueden hacer con Seismic Unix, sobre todo en aquellas donde trabajar directamente en el Shell se puede convertir en algo poco funcional e inefectivo. En el apéndice 2 se verá lo necesario para crear Scripts sin problema alguno.

Si se requiere muchas más información del Shell-Scripting puedo referirse a la siguiente dirección electrónica http://www.freeos.com/guides/lsst/.

CAPITULO III

3. -Modelado básico usando Seismic Unix

3.1 Modelado de perfiles de propiedades del subsuelo

3.2 Modelado de la propagación de ondas acústicas mediante diferencias finitas

Capítulo 3

Modelado básico usando Seismic Unix.

3.1 Modelado de perfiles de propiedades del subsuelo.

Los comandos que posee **SU**, nos permiten generar una representación grafica del subsuelo mediante algunas de sus propiedades físicas, con el que se puede caracterizar. Uno de los comandos usados para este fin el siguiente UNIF2, el cual se explicara a continuación.

UNIF2

Utilidad: Este comando nos permite generar un modelo bidimensional del subsuelo, con las propiedades de velocidad o densidad del cual esta uniformemente muestreado a partir de un archivo ASCII, en este archivo se encuentra definido tanto la ubicación como la forma e interfaces del modelo.

En otras palabras **UNIF2** nos permite generar modelos bidimensionales del subsuelo uniformemente muestreados de un modelo de capas, en donde en cada capa la velocidad o densidad es una función lineal de la posición.

El archivo de entrada para UNIF2 es un archivo del tipo ASCII como ya se había mencionado en el párrafo anterior, este deberá contener las coordenadas (X, Z) de una serie de puntos que van a definir la tendencia de cada interfaz contenida en el modelo. En este sentido el archivo de entrada constara de dos columnas, en la cual la primera contendrá los valores de X y la segunda los valores de Z. De esta forma, cada puto estará ubicado en líneas consecutivas, fijando el número de líneas destinadas a cada interfaz igual al número de puntos con los que se haya decidido construir la interfaz correspondiente.

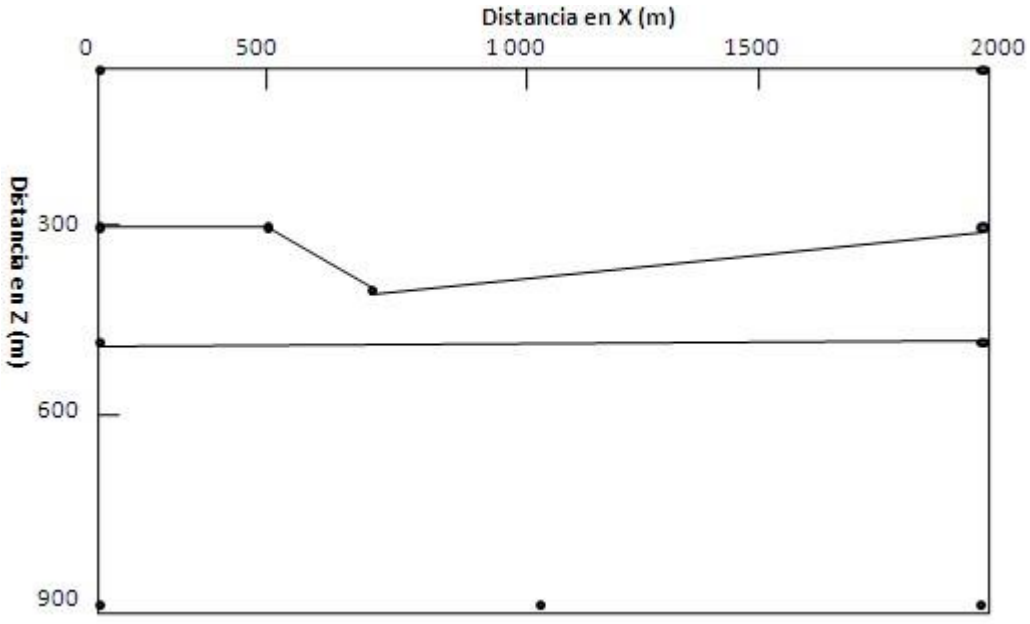
Como es de esperar, cada interfaz constara de un mínimo de dos puntos, sin embargo esta puede ser definida mediante cualquier cantidad de puntos por encima del mínimo establecido que son 2 puntos. Generalmente la cantidad de puntos aumenta conforme aumenta la complejidad de cada interfaz (tortuosidad) y no tiene que ser igual para todas las interfaces del modelo. El par 1-99999 tiene la función de separar las entradas para interfaces

consecutivas, esto quiere decir que luego de definir todos los puntos de una interfaz debemos introducir en la siguiente línea, el punto 1 – 99999 para continuar con la caracterización de la siguiente interfaz en la línea siguiente. A continuación podemos observar un archivo de entrada para unif2.

Ejemplo 1: Archivo de entrada para unif2.

X (m)	Z (m)
0	0
2000	0
1	-99999
0	300
500	300
700	450
2000	300
1	-99999
0	500
2000	500
1	-99999
0	900
1000	900
2000	900

Tabla 1: Aquí se presenta un ejemplo de información numérica de entrada, para la realización de un modelo bidimensional con el comando UNIF2, en donde la primera columna X (m) representa a a la posición de los interfaces en superficie mientras que Z (m) representa a la posición de los interfaces en la profundidad.



En esta figura se muestra como estaría conformado el modelo de capas si este se realizara con el comando UNIF2, como se puede apreciar tenemos los puntos marcados en donde se muestran las interfaces de cada estrato.

Interfaz 1 o tope superior: (0,0); (2000,0)

Interfaz 2: (0,300); (500,300); (600,450); (2000,300)

Interfaz 3: (0,500); (2000,500)

Interfaz 4 o tope inferior: (0,900); (1000,900);(2000,900)

En donde el primer valor corresponde a la posición en x (m) distancia en la superficie y el segundo valor corresponde a la posición en z (m) en profundidad.

Como se explico, la primera columna de la tabla anterior, asigna los valores a X de los puntos que definen las interfaces del modelo y la segunda corresponde a los valores de Z y el punto 1 -99999 separa las entradas para interfaces consecutivas. Por lo tanto, el archivo anterior (Ejemplo 1) está conformado por un total de 4 interfaces incluyendo el tope superior e inferior del modelo los cuales podrían no ser consideradas como interfaces en el modelo del subsuelo, sin embargo deben ser definidos para establecer

los límites superior e inferior dentro del modelo en construcción, si esto no se hiciera el modelo tendería a infinito al no tener ninguna restricción.

En este ejemplo se puede corroborar que no necesariamente todas las interfaces deben estar conformadas por el mismo número de puntos. Podemos ver que la Interfaz 2 fue construida mediante cuatro puntos, siendo esta la de mayor complejidad, mientras que las otras 3 se forman mediante solo dos puntos, lo que indica que son interfaces rectas.

Parámetros:

A continuación se dará una breve explicación de algunos de los parámetros más importantes de **UNIF2**, esta información fue tomada de la ayuda que trae el paquete, para obtenerla solo basta con escribir en la terminal el nombre de la función y la información será desplegada.

Parámetros requeridos: Ninguno

Parámetros opcionales: Varios

Parámetros de muestreo: Definen las dimensiones del modelo

nx= .....Numero de muestras en x (2da dimensión).

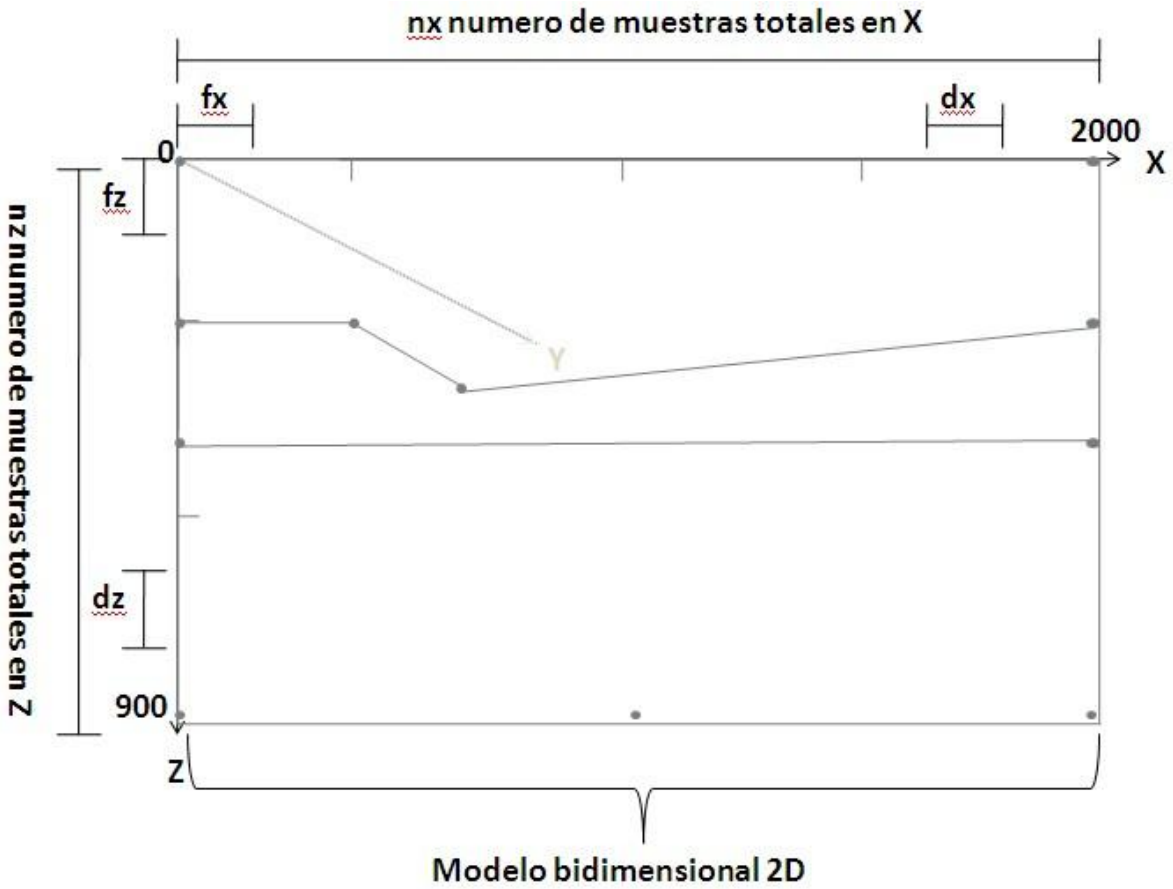
nz= .....Numero de muestras en z (1ra dimensión).

dx= .....Intervalo de muestreo en x.

dz= .....Intervalo de muestreo en z.

fx= .....Distancia correspondiente a la primera muestra en x.

fz= .....Profundidad correspondiente a la primera muestra en z.



En esta figura se explica cada uno de los parámetros mínimos que son requeridos para realizar correctamente un modelo bidimensional, en donde nx es el numero de muestras en X , nz el numero de muestras en Z, dx es el intervalo de muestreo en X, dz intervalo de muestreo en Z, fx la distancia correspondiente a la primera muestra y finalmente fz que es la distancia de a la primera muestra en Z.

v00=1500, 2000,2500…. Para el caso que será abordado en este trabajo, en donde los modelos estarán formados por capas con propiedades constantes ya que de no ser así el modelo se complicaría demasiado, este parámetro corresponde a los valores de la propiedad en cada capa (generalmente velocidad).

Cuando se construye un modelo o perfil del subsuelo utilizando **UNIF2**, se crea un mayado 2-D de muestras uniformemente espaciadas utilizando los parámetros de muestreo (nz, dz, nx y dx). En este paso se definirá la profundidad y la longitud del modelo ajustando el numero de muestras nz, nx y la distancia entre ellas dz, dx. La forma y ubicación de las interfaces del modelo será definida mediante el archivo ASCII incluido dentro de la rutina.

Es importante señalar que la primera muestra del modelo corresponde a la distancia cero, no a uno, por lo tanto la longitud y la profundidad del modelo será igual a:

Profundidad= (nz dz)- dz

Longitud= (nx dx) – dx

Los parámetros fx y fz corresponden a la ubicación de la primera muestra en x y en z respectivamente. En otras palabras, indican la distancia y profundidad a partir de las cuales se quiere construir el modelo.

El comando **UNIF2** generalmente es usado para realizar un perfil de velocidades. Por esto, el parámetro **v00** se refiere, en principio, a los valores de velocidad en (mseg). Sin embargo en este parámetro también pueden ser especificados valores de densidad en (gr/cm^3) o cualquier otra propiedad que deseemos representar.

Archivo de salida (outfile): Archivo en formato binario que contiene el perfil de propiedades generado. Este archivo puede ser visualizado como una mayado uniformemente muestreado de valores de la propiedad representada (generalmente velocidad). Este tipo de archivos es referido de la siguiente forma**: vfile [nx] [nz]**.

Para poder visualizar el perfil correspondiente al archivo de salida de **UNIF2** debemos emplear un comando que nos permita graficar archivos con formato binario .Se ha utilizado el comando **XIMAGE** para desplegar las graficas en una ventana (x-window graphic) y el comando **PSIMAGE** para obtener la imagen en un archivo con formato PostScript. Ambos comandos pertenecen al paquete Seismic Unix.

El siguiente código realiza un perfil de velocidades del subsuelo:

Código 3.1: Creación de un Perfil de velocidades mediante el comando UNIF2

```
#! / bin / sh

unif2 < Archivo_de_entrada \ nz=121 dz=25 nx=121 dx=25 \ v00=1500,2500,3500 > Perfil_de_velocidad
```

El archivo de entrada utilizado en el script anterior se muestra a continuación:

Nombre del archivo: modelo1

0	**0**
3000	**0**
1	**-99999**
0	**1500**
3000	**1500**
1	**-99999**
0	**2500**
3000	**2500**

1	**-99999**
0	**3025**
3000	**3025**

En esta tabla se muestra como queda conformado el modelo de usáremos como ejemplo, aquí podemos ver que el modelo está conformado por 3 interfaces.

El archivo generado mediante el código anterior (3.1) fue graficado a través del comando **PSIMAGE**. Tal grafica corresponde a la figura 3.1 y el código que la realiza es el siguiente.

Código 3.2: Graficación del perfil de velocidad generado en el código 3.1

```
## Graficación del perfil de velocidades en formato postscript
psimage < Perfil_de_velocidad \n1=121 d1=25 n2=121 d2=25 \legend=1 units='Velocidad (m/seg ) ' \label1='Profundidad (m) ' label2='Distancia (m) ' \width=4.5 height=2.5 lx =0.6 d1num=500 d2num=1000 \legendfont=times roman8 labelsize =12 titlesize =18 > Velocidad.ps
```

Para saber qué es lo que realiza el comando **PSIMAGE** solo basta con escribir el nombre en la terminal de Linux (**psimage**).

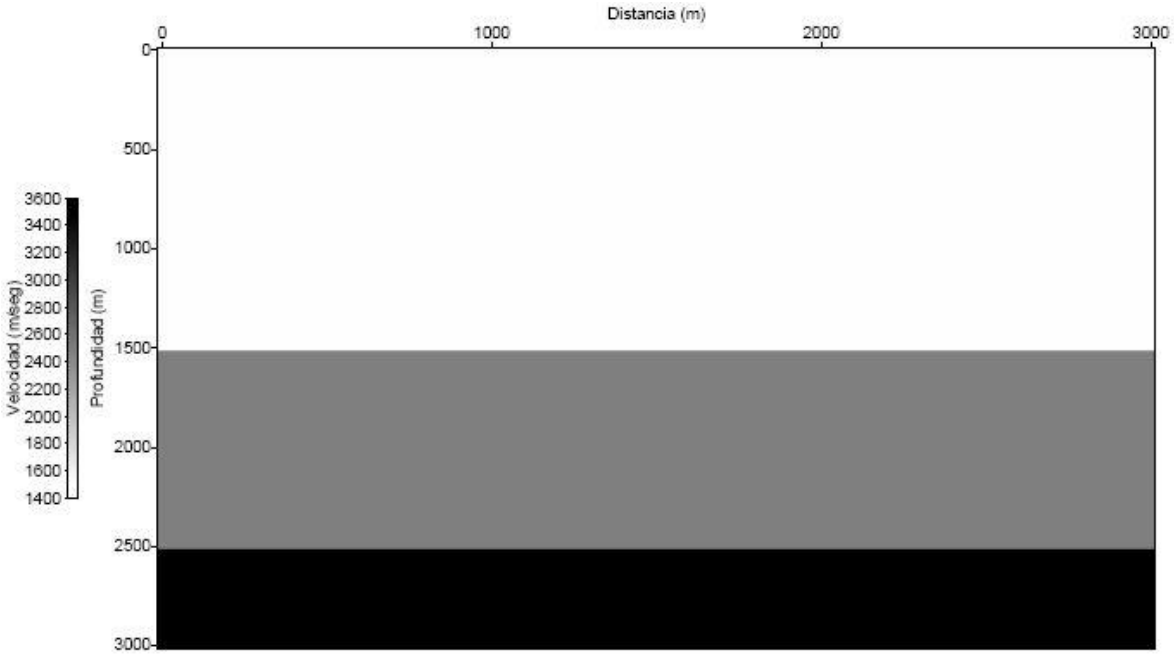


Figura 3.1: Perfil de velocidad generado mediante UNIF2

Ahora si el parámetro v00 = 1500, 2000, 3000 del código del perfil de velocidades (Script 3.1) es cambiado por valores de densidad en (gr/c^3) (v00=1.0, 2.25, 2.25), se obtiene un Perfil de densidades el cual se muestra a continuación En este perfil los valores de densidad para la segunda y tercera capa son los mismos. Por esa razón solo se representan dos capas en la siguiente figura

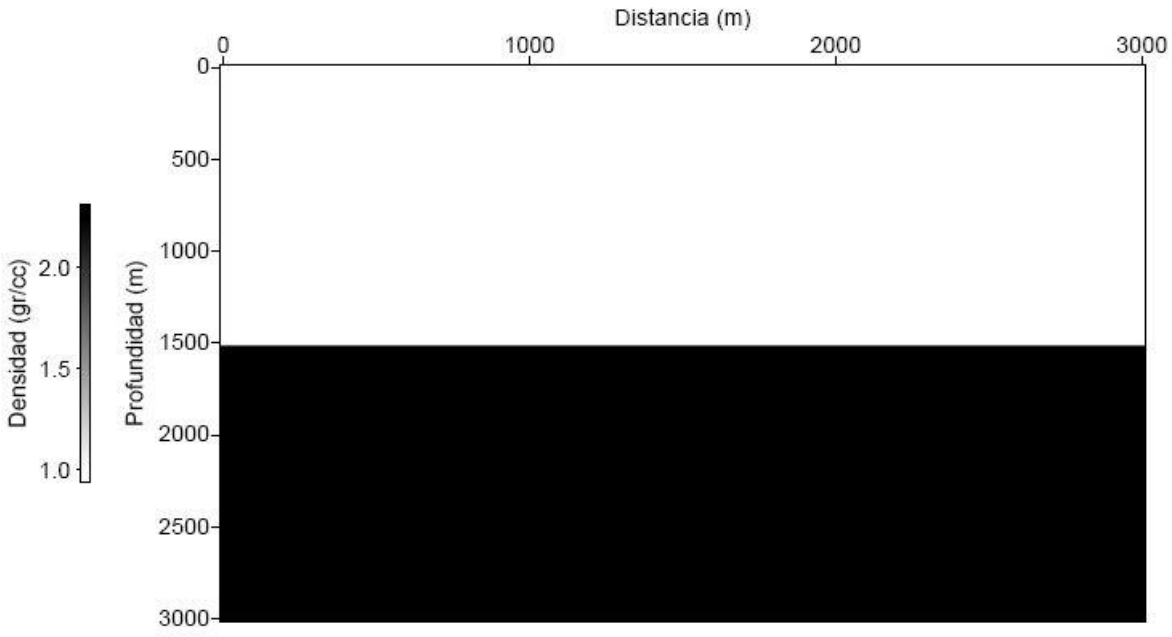


Figura 3.2: Perfil de densidad generado mediante UNIF2

3.2 Modelado de la propagación de ondas acústicas mediante diferencias finitas.

En este punto vamos a recrear la propagación de ondas acústicas en un modelo 2-D del subsuelo utilizando el método de diferencias finitas. El modelo del subsuelo será definido mediante dos perfiles de propiedades, uno de velocidad y otro de densidad, creados a través del comando UNIF2 todos los comandos empleados aquí pertenecen al paquete SU.

El método de diferencias finitas.

El método de diferencias finitas es una clásica aproximación para encontrar soluciones numéricas a las ecuaciones que componen el modelo matemático de un sistema continuo.

Básicamente, en una solución por diferencias finitas, las derivadas son reemplazadas por aproximaciones finitas, convirtiendo entonces un problema de ecuaciones diferenciales en un problema algebraico de menor complejidad. En este sentido podremos utilizar este método para encontrar soluciones numéricas a las ecuaciones que describen la propagación de ondas acústicas en un sistema bidimensional.

En estos conceptos está basado el modelado de la propagación de ondas acústicas que vamos a recrear mediante los Códigos de Seismic Unix.

Cuando se realiza un modelo de propagación de ondas utilizando el método de diferencias finitas, es indispensable tomar en cuenta que dicho método presenta ciertas condiciones que deben cumplirse para que el sistema sea estable.

En torno a esto, algunos de los parámetros involucrados deben tener cierta relación entre ellos. En la **referencia [4]** podemos encontrar, de manera resumida, la teoría relacionada con el método de diferencias finitas enfocado a la propagación de ondas acústicas en un sistema bidimensional.

Aquí se mencionan y explican las condiciones de estabilidad que debe cumplir el sistema para generar resultados coherentes.

El comando **SUFDMOD2** del paquete Seismic Unix nos permite realizar un modelado de la propagación de ondas acústicas a partir de un modelo del subsuelo representado por un perfil de velocidades uniformemente muestreado como el que fue obtenido a través del comando **UNIF2**.

Este comando ajusta uno de sus parámetros con el fin de resguardar la condición de estabilidad del sistema .Sin embargo, para obtener buenos resultados, con sentido geofísico, es necesario que los parámetros opcionales que sean ingresados por el usuario presenten valores adecuados, estas condiciones también son discutidas en la **Referencia [4]**.

A continuación vamos a explicar, el comando **SUFDMOD2** (Archivos de entrada, parámetros más importantes y archivos de salida), para luego mostrar algunos códigos que realizan dicha tarea.

SUFMOD2

Utilidad: Modelado (de segundo orden) mediante Diferencias Finitas para la ecuación de ondas acústicas.

Sintaxis

Sufdmod2 < infile nx= nz= tmax= xs= zs= [parámetros opcionales dfile= hsfile=] > outfile

Archivo de entrada (infile): Archivo en formato binario que contiene un mayado uniformemente muestreado de valores de velocidad (vfile [nx][nz]). Los archivos generados mediante UNIF2 son archivos de entrada para SUFDMOD2. También se puede considerar la influencia de la densidad igualando el parámetro “dfile” a un perfil de densidad (dfile [nx][nz]) con las mismas características del perfil de velocidad.

Parámetros:

Parámetros Requeridos: nx= nz= tmax= xs= zs= (definidos a continuación)

Parámetros de muestreó:

- **nz**=Numero de muestras en z (Primera dimensión) [Parámetro requerido].
- **dz**=Intervalo de muestreo en z.
- **nx**=Numero de muestras en x (Segunda dimensión) [Parámetro requerido].
- **dx**=Intervalo de muestreo en x (intervalo entre receptores "gx").
- **fx**=Primera muestra en x.
- **fz**=Primera muestra en z.
- **nt**=1+tmax/dt: Numero de muestras temporales (dt es determinado para que la condición de estabilidad del modelado se cumpla).

Parámetros de contienen archivos de salida:

- **hsfile**=Archivo de salida para la línea horizontal del sismograma [nx][nt] (Shot Gather).
- **Vsfile**=Archivo de salida para la línea vertical del sismograma [nz][nt] (Shot Gather para un perfil sísmico vertical "PSV").
- **Ssfile**=Archivo de salida para la ondicula del sismograma [nt].

Parámetros relacionados con la fuente:

- **zs**=Ubicación en z de la fuente [Parámetro requerido].
- **xs**=Ubicación en x de la fuente [Parámetro requerido].

Parámetros relacionados con los receptores.

- **hsz**=Coordenadas en z de la línea horizontal del sismograma (Profundidad de los receptores).
- **vsx**=Coordenadas en x de la línea vertical del sismograma.
- **tmax**=Tiempo máximo para el modelado de la propagación. Matemáticamente, este parámetro corresponde con el mayor tiempo para el cual será resuelta la ecuación de propagación de ondas acústicas. En términos de una adquisición sísmica, es el tiempo máximo de grabación [Parámetro requerido].

Parámetros relacionados con la ondicula del sismograma:

- **fpeak**=0.5+fmax: Frecuencia pico de la ondicula (Hz).
- **fmax**=vmin/(10.0*h): Frecuencia máxima de la ondicula (Hz).

Otros parámetros:

- **abs**=1,1,1,1: Condiciones de absorción en los bordes del modelo 1,1,1,1 = Tope superior, izquierda, tope inferior, derecha. =0,1,1,1 para condición de superficie libre en el tope superior del modelo.

Archivo de salida (outfile): Archivo con formato "SU" que contiene la caracterización de las ondas acústicas para todos los instantes de tiempo. Si el parámetro "**hsfile**" es especificado, obtendremos un archivo que contiene el Disparo Sintetico correspondiente a la propagación de ondas modelada, por supuesto también en formato SU. De igual forma, si se especifican los parámetros "**vsfile**" o "**ssfile**" se obtendrán archivos concernientes al Disparo Sintetico para el caso de un "**PSV**" (**vsfile**) y la ondícula del sismograma (**ssfile**).

El archivo de salida de **SUFDMOD2** (**outfile**) debe ser visualizado mediante una película que muestre la posición de las ondas acústicas para todos los instantes de tiempo. Para esto se puede utilizar el comando **SUXMOVIE**, el cual despliega la película en una ventana o el comando **SUPSMOVIE** para generar un archivo que contiene todas las imágenes que conforman la película en formato PostScript. Por otra parte, los archivos correspondientes a los parámetros "**hsfile**", "**vsfile**" y "**ssfile**" pueden ser visualizados mediante los comandos **SUXWIGB** y **SUXIMAGE** (despliega las graficas en ventanas) o a través de **SUPSWIGB** y **SUPSIMAG**E para obtener la grafica en un archivo con formato PostScript.

Por otra parte, los archivos correspondientes a los parámetros "**hsfile**", "**vsfile**" y "**ssfile**" pueden ser visualizados mediante los comandos **SUXWIGB** y **SUXIMAGE** (para graficas desplegadas en ventanas) o a través de **SUPSWIGB** y **SUPSIMAGE** para obtener la grafica de un archivo con formato Postscript.

Los nombres de los comandos de Seismic Unix que realizan graficas, al igual que muchos de los comandos de SU, es tan compuestos de términos que describen sus características. El termino SU se refiere al formato que acepta. -X- o -PS- indican el tipo de gráfica que generan; X: Gráficas desplegadas en ventanas (x-window graphic), PS: Archivo con formato PostScript. Y -MOVIE-, -WIGB- o -IMAGE- nos dan información acerca de la visualización de las gráficas; MOVIE: Se refiere a una película, WIGB: Gráficas en donde se puede visualizar el conjunto de trazas que conforman el archivo (Ejemplo: Figura 3.4).

El siguiente código (3.3) realiza un modelado de la propagación de ondas acústicas en un perfíl 2-D del subsuelo utilizando el método de diferencias finitas. El modelo del subsuelo se encuentra representado por un perfíl de velocidades y uno de densidades. El perfíl de velocidad corresponde al archivo "Perfíl De Velocidades "generado en el codigo 3.1 y el perfíl de densidad corresponde a un archivo llamado "Perfíl De Densidades" que fue generado de forma similar al de velocidades, pero sustituyendo el parámetro v00=1500, 2500,3500 por v00=1.0, 2.25, 2.25, los valores mostrados están en gr/c^3. Las gráficas de estos dos perfiles se encuentran en las Figuras 3.1 y 3.2 respectivamente.

Codigo 3.3: Modelado de la propagación de ondas acústicas mediante diferencias finitas en un medio representado por los perfiles de las Figuras 3.1y 3.2

```
#! /bin/sh

##### Modelado por diferencias finitas

sufdmod2 < Perfil_de_velocidades dfile=Perfil_de_densidadd \nz=121 dz=25 nx=121 dz=25 \zs=0 xs=1500 fpeak=5 fmax=10 \hsz=0 tmax=4 abs=1,1,1,1 mt=2 \hsfile=gather > Propagacion de ondas
```

Propagacion_de_ondas: Este contiene el modelo de propagación de ondas acústicas.

Gather: Contiene la información de todos los Tiros Sintéticos.

Para visualizar este archivo vamos a utilizar el comando **SUPSMOVIE** mediante el cual obtendremos un conjunto de imágenes que describen la propagación de las ondas en el modelo citado anteriormente. El código que gráfica este archivo es el siguiente:

Codigo 3.4: Graficación del archivo "Propagacion_de_ondas" retornado en el Codigo 3.3

```
#! /bin/sh

### Graficacion del archivo "Propagacion_de_ondas"

Suspsmovie < Propagacion_de_ondas title="Propagacion de ondas acústicas" \label1='Profundidad (m)' label2='Distancia (m)' \n1=121 d1=25 n2=121 d2=25 \clip=0.3 hbox=5 wbox=5 d1num=500 d2num=500 lx=0 labelsize=17 > Ondas.ps
```

En el código anterior n1, d1, n2 y d2 son los parámetros de muestreo. Estos valores son análogos a los parámetros nz, dz, nx y dx, respectivamente, utilizados en el código| 3.3 ambos grupos de parámetros deben poseer valores iguales. El parámetro "clip" es de gran importancia para lograr una buena visualización de las graficas generadas. Los demás valores están relacionados con la estética de la figura y pueden ser revisadas en la ayuda de SU **[Referencia 3].**

El codigo 3.4 genera un conjunto de imágenes que describen la propagación de las ondas acústicas en el modelo de entrada. En la figura 3.3 se muestran tres imágenes asociadas con el inicio, punto medio y final de la propagación modelada.

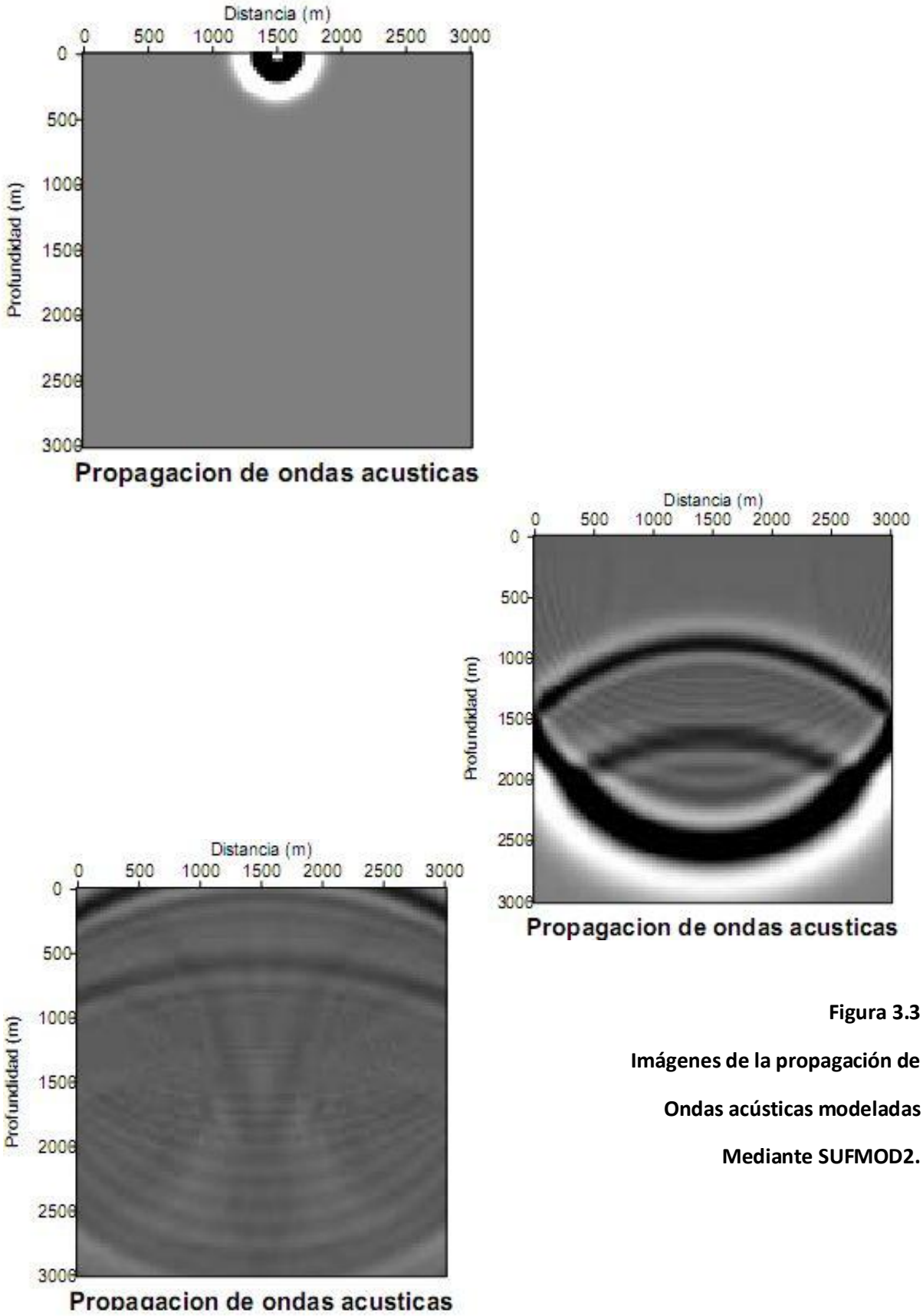


Figura 3.3

Imágenes de la propagación de

Ondas acústicas modeladas

Mediante SUFMOD2.

El otro archivo es el gather: Este archivo corresponde al tiro sintético de la propagación de ondas modelada. Como el numero de muestras en x empleadas en el modelado realizado mediante **SUFDMOD2** (Codigo 3.3 fue de 121).

Por lo tanto el tiro sintético asociado estará conformado por un total de 121 trazas separadas una distancia igual al espaciamiento entre las muestras. Para graficar este archivo vamos a emplear el comando **SUSPWIGB**, el cual fue creado con el fin de graficar archivos con la extensión SU. Este comando retorna las graficas en un archivo del tipo PostScript.

La explicación de los parámetros asociados a **SUSPWIGB** puede ser revisada en la ayuda de SU **[Referencia 3]**. Para el mismo fin también se puede utilizar este comando **SUXWIGB**, el cual despliega las graficas en una ventana (x-window graphic). A continuación se muestra el codigo empleado para graficar el archivo "gather".

Codigo: 3.5 Graficacion del archivo "gather"

```
#! /bin/sh

suspswigb < gather clip=0.17 \title= "Shot Gather correspondiente a un disparo en la posicion 1500 m" \label1= 'Tiempo (seg)' label2= 'Distancia (m)' \wbox=8.5 hbox=8.5 titlesize=18 d2num=500 > gather.ps
```

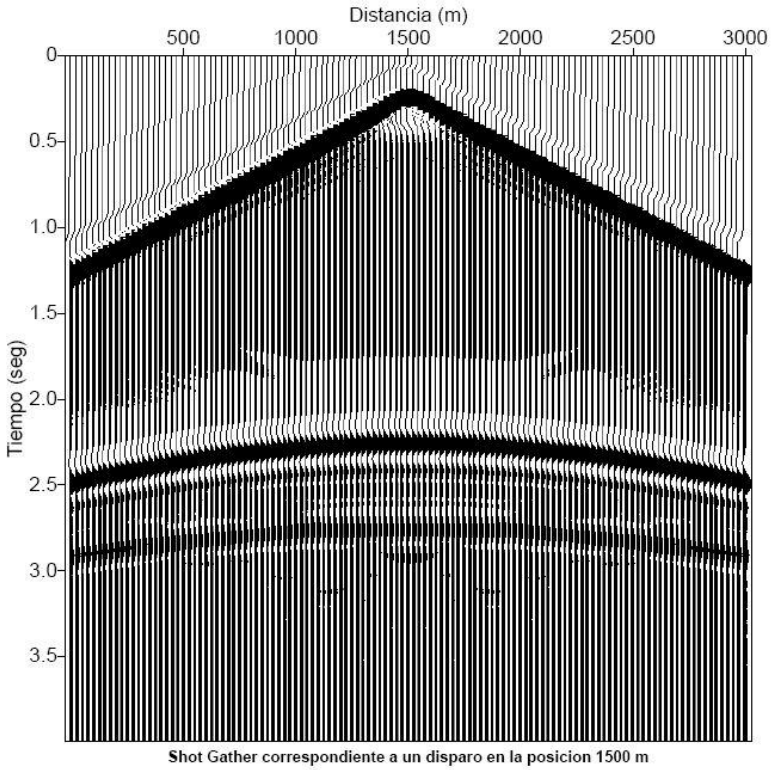


Figura 3.4: Shot Gather obtenido a partir del modelado de la propagación de ondas acústicas realizado a través del comando SUFDMOD2.

CAPITULO IV

4. Modelado de una adquisición sísmica 2D

4.1 Geometría de adquisición
4.2 Parámetros de adquisición
4.3 Script empleado para realizar el modelo
4.4 Generación y Graficación del modelo
4.5 Generación de los Tiros Sintéticos

Capitulo 4

Modelado de adquisición sísmica 2D.

Para modelar una adquisición sísmica tenemos que definir en primer lugar, la estructura y propiedades del subsuelo. Es decir debemos generar una representación del medio a través de la definición de sus propiedades físicas y estructurales. En la Sección 3.1 se explica una forma de cómo se puede realizar esto, este tipo de representaciones se pueden realizar mediante el comando **UNIF2** el cual genera perfiles del subsuelo 2D.

Con lo que se verá a continuación seremos capaces de generar nuestros propios modelos de partidas para posteriormente recrear la adquisición sísmica de los modelos generados con el comando **UNIF2**.

Para representar la sísmica vamos a realizar un modelo de propagación de ondas acústicas en los perfiles creados anteriormente. Para esto utilizaremos el método de diferencias finitas explicado en la Sección 3.2 mediante el uso del comando **SUFDMOD2** a continuación veremos los parámetros que se requieren para poder realizar la adquisición sísmica como lo es la geometría de adquisición y los parámetros de adquisición.

4.1 Geometría de adquisición

En la geometría de adquisición que vamos a emplear, los receptores estarán dispuestos proporcionalmente a ambos lados de la fuente, lo que hace que el offset tome igual número de valores positivos y negativos.

El primer grupo de receptores corresponde a la posición 0 (cero) en metros. Esto último implica que la posición del primer disparo será igual al offset máximo. A diferencia de un proceso de adquisición sísmica real, en este modelado si tenemos un grupo de receptores ubicado en el mismo punto de la explosión (offset=0).

4.2 Parámetros de adquisición

1. Offset máximo (offset max): 1500 metros

2. Intervalo entre puntos de disparos (ID): 50 metros

3. Intervalo entre grupos de receptores (IR): 25 metros

4. Tiempo de grabación (tmax): 5 segundos

5. Intervalo de muestreo temporal: Es ajustado por el programa **SUFDMOD2** para que el modelado por diferencias finitas sea consistente.

4.3 Código empleado para la realización del modelo

Una adquisición sísmica consiste en un conjunto de disparos (**Shots**) a lo largo del área de interés, cada disparo va a estar representado por un modelado de propagación de ondas acústicas en el punto definido por la posición de la fuente. Dicho modelo, será relacionado con cada disparo, debe realizarse en un medio (subsuelo) cuya ubicación y extensión concuerden con el tendido correspondiente.

Lo anterior implica que debemos realizar un programa iterativo que contenga un número de ciclos igual a la cantidad de disparos realizados. De esta forma serán generados los **Tiros Sintéticos** de la adquisición sísmica recreada.

El **Código** que se ha creado, es capaz de realizar un modelo, con unos pocos parámetros que pueden ser entendidos por cualquier persona con conocimientos básicos de geofísica. Es decir, los parámetros intrínsecos de cada comando son ajustados automáticamente por medio de parámetros más entendibles definidos previamente.

En la siguiente parte vamos a explicar la forma como fueron ajustados los parámetros:

Definición y ajuste de parámetros.

Parámetros definidos por el usuario: Esta categoría está formada por la mínima cantidad de parámetros que deben ser definidos en el **Código**. Por supuesto, los demás parámetros pueden ser redefinidos, pero para esto es obligatorio que el usuario tenga ciertos conocimientos para poder comprenderlos. Si el usuario es novato, puede correr el **Código** sin ningún problema definiendo los parámetros mostrados en esta categoría. Estos parámetros pueden ser divididos en dos grupos:

1. **Parámetros relacionados con la descripción del subsuelo:**

    - Longitud del modelo en metros (Longitud)
    - Profundidad del modelo en metros (Profundidad)
    - Definición de las interfaces (Archivo de texto): Archivo de entrada UNIF2.
    - Velocidad de cada capa m/seg (v00):v00=v1, v2, v3,…
    - Densidad de cada capa en gr/cc (v00): Análogo al parámetro anterior. v00=d1, d2, d3,….

2. **Parámetros de adquisición:**

    - Offset máximo (offset_max)
    - Intervalo entre puntos de disparos (ID)
    - Intervalo entre grupo de receptores (IR)
    - Tiempo de grabación (tmax)

Ajuste de otros parámetros de adquisición: A través de los elementos especificados anteriormente, el **Código** ajustara algunos parámetros de adquisición que van a facilitar el manejo de las variables en las operaciones realizadas y a lo largo de este **Código**. Estos parámetros son:

- **Longitud del tendido (LT):** Como la geometría de adquisición utilizada dispone igual número de receptores a ambos lados de la fuente, la longitud del tendido será el doble del offset máximo.

$$LT=2*offset\_max$$

- Número de grupos de receptores (NR): Este parámetro es ajustado a través de la longitud del tendido y el intervalo entre grupos de receptores (IR) según la fórmula.

$$NR = \frac{LT}{IR} + 1$$

El factor de uno (1) que sumamos a la fracción, se debe a que tenemos un grupo de receptores para offset cero.

- Número de disparos (ND): Valor ajustado según la Longitud del modelo (Longitud), longitud del tendido (LT) y el intervalo entre disparos (ID):

$$ND = \frac{Longitud - LT}{ID}$$

Ajuste de los parámetros intrínsecos de los comandos empleados: Algunos de los parámetros que emplean los comandos son ajustados a través de los parámetros expuestos arriba. Por ejemplo: nx, dx, nz y dz son los parámetros de muestreo, ellos define el número de muestras (nx, nz) y la distancia entre ellas (dx, dz) para los valores de propiedades representados en los perfiles realizados mediante **UNIF2**.

Como estos perfiles serán ingresados al comando SUFDMOD2 para realizar el modelado de ondas acústicas, los parámetros de muestreo también están relacionados con los parámetros de adquisición. En este sentido, cada muestra en x (nx) definida en el comando **SUFDMOD2** generara una traza en los archivos de salida. Como cada traza está relacionada con un grupo de receptores, nx debe ser igual al número de grupos de receptores (NR) y dx igual a la distancia entre receptores (IR).

Como nos parece apropiado que las muestras en x y z tengan el mismo espaciamiento, ajustamos la distancia entre muestras en z (dz) igual a la distancia entre muestras en x (dx). De esta forma dz=IR y el número de muestras en z será igual a la profundidad (Profundidad) dividida entre el intervalo entre receptores (IR) más uno. El parámetro tmax es el tiempo máximo del modelado de la propagación de ondas acústicas, pero en términos de una adquisición sísmica es el tiempo máximo de grabación.

Este último parámetro ya fue definido con la misma cadena de caracteres (tmax), lo que ajusta de una vez el parámetro intrínseco del comando **SUFDMOD2** (tmax). Los demás parámetros definidos están relacionados con cada comando y fueron explicados en el Capítulo 3. Estos parámetros también pueden ser revisados en la ayuda de SU [**Referencia 3**].

- nx=NR [Parámetro de UNIF2 y SUFDMOD2]
- dx=IR [UNIF2 y SUFDMOD2]
- nz=(Profundidad/IR)+1 [UNIF2 y SUFDMOD2]
- dz=IR [UNIF2 y SUFDMOD2]
- tmax: Ya fue definido en los parámetros de adquisición (Sección 4.2). [SUFDMOD2]
- fpeak=5 [SUFDMOD2]
- fmax=10 [SUFDMOD2]
- mt=4 [SUFDMOD2]
- clip=0.25 - Este parámetro pertenece a los comandos de graficación [SUXIMAGE, SUXMOVIE, SUPSIMAGE, SUPSMOVIE, SUXWIGB, SUPSWIGB etc.]

Para explicar la forma de realizar el modelo vamos a considerar un medio formado por tres capas con propiedades constantes (velocidad y densidad) divididas por dos interfaces horizontales (Modelo 1). El **Código** que hemos escrito puede ser fragmentado en dos partes. En la primera parte se genera y gráfica la representación del subsuelo a partir de la cual se hará el modelo de la adquisición (Sección 4.4). La segunda corresponde a la generación de los **Tiros Sintéticos** mediante el modelo de la propagación de ondas acústicas en el medio construido en el primer fragmento (Sección 4.5).

4.4 Generación y graficación del modelo inicial

A continuación mostramos los valores que hemos empleado para construir el modelo:

- Longitud= 5000 m
- Profundidad= 3000 m

Descripción de las interfaces: Interfaces horizontales.

- Interfaz 1: Se encuentra a 1500 metros de profundidad
- Interfaz 2: 2500 metros

Para poder introducir la información relacionada con las interfaces es necesario ingresar el siguiente archivo.

El cual ha sido llamado **Modelo 1**.

0	**0**
5000	**0**
1	**-99999**
0	**1500**
5000	**1500**
1	**-99999**
0	**2500**
5000	**2500**
1	**-99999**
0	**3025**
5000	**3025**

En esta tabla se muestran los valores del modelo, del cual vamos a realizar la adquisición sísmica, como podemos apreciar este consta de tres interfaces, además de tener una longitud de 5000m y una profundidad de 3000m.

- Archivo_Entrada=Modelo1
- Velocidad de capa: v00=1500, 2500, 3500 m/seg
- Densidad de cada capa: v00=1.0, 2.25, 2.25 gr/c^3

El **Código 4.1** genera y gráfica los perfiles de partida (velocidad y densidad) de la adquisición sísmica utilizando los valores definidos arriba. Los resultados obtenidos pueden ser visualizados en las Figuras 4.1 y 4.2.

Código 4.1: Generación y graficación de los perfiles de partida (velocidad, densidad) que serán utilizados para modelar una adquisición sísmica (Modelo 1)

```
#!  /bin/sh

##Parametros definidos por el usuario

Archivo_Entrada=Modelo1


Longitud=5000

Profundidad=3000

v00=1500,2500,3500

#NOTA: Los valores de Densidad deben ser ingresados directamente

# en el comando UNIF2 en la parte destinada a la generación del

# perfil de densidad, ya que corresponde al mismo parámetro v00

offset_max=1500

ID=50

IR=25

tmax=5

## Ajuste de otros parámetros de adquisición

LT=$((2*offset_max))
```

```
ND=$(((Longitud-LT)/ID))

NR=$((LT/IR+1))

##Ajuste de los parámetros intrínsicos de los commandos

nx=$((LT/IR+1))

dx=$IR

nz=$((Profundidad/IR+1))

dz=$IR

tmax=$tmax

fpeak=5

fmax=10

mt=4

clip=0.25 #(Puede variar del comando)

#Generación del Perfil de velocidad del modelo completo

unif2 < $Archivo_Entrada \nx=$((Longitud+1)) dx=1 nz=$((Profundidad+1))  dz=1 \
v00=1.0,2.25,2.25 > densidad

# Graficación del perfil de velocidad (Formato PostScript)

psimage < velocidad title="Perfil de Velocidad de onda P" \n2=$(($Longitud+1)) d2=1
n1=$(($Profundidad+1)) d1=1 \legend=1  units='Velocidad (m/seg)' \label11='Profundidad (m)'
label2='Distancia (m)' \width=10.0 height=6.0 lx=0.6 d1num=500 d2num=1000
\legendfont=times_roman8 labelsize=12 titlesize=18 \> Velocidad_completo.ps

#Graficación del perfil de densidad (Formato PostScript)

psimage < densidad title="Perfil de Densidad" \n2=$(($Longitud+1)) d2=1
n1=$(($Profundidad+1)) d1=1 \legend=1  units='Densidad (gr/cc)' \label11='Profundidad (m)'
label2='Distancia (m)' \width=10.0 height=6.0 lx=0 d1num=500 d2num=1000 \>
Densidad_completo.ps
```

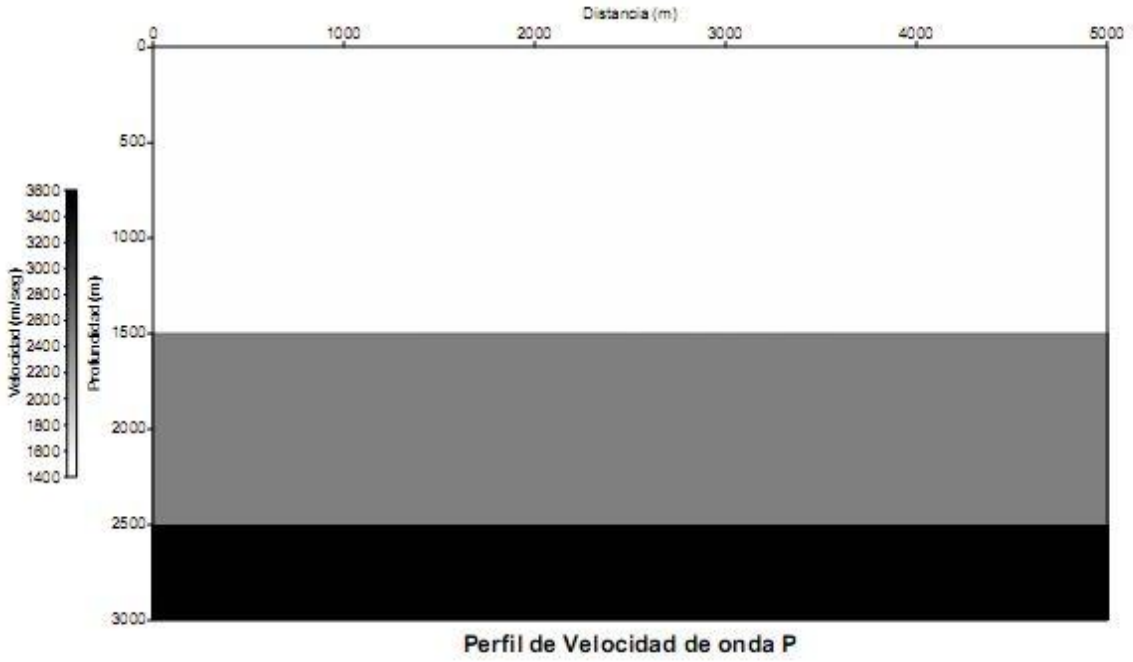


Figura 4.1: Perfil de velocidad a partir del cual se hará el modelo de una adquisición sísmica (Modelo 1).

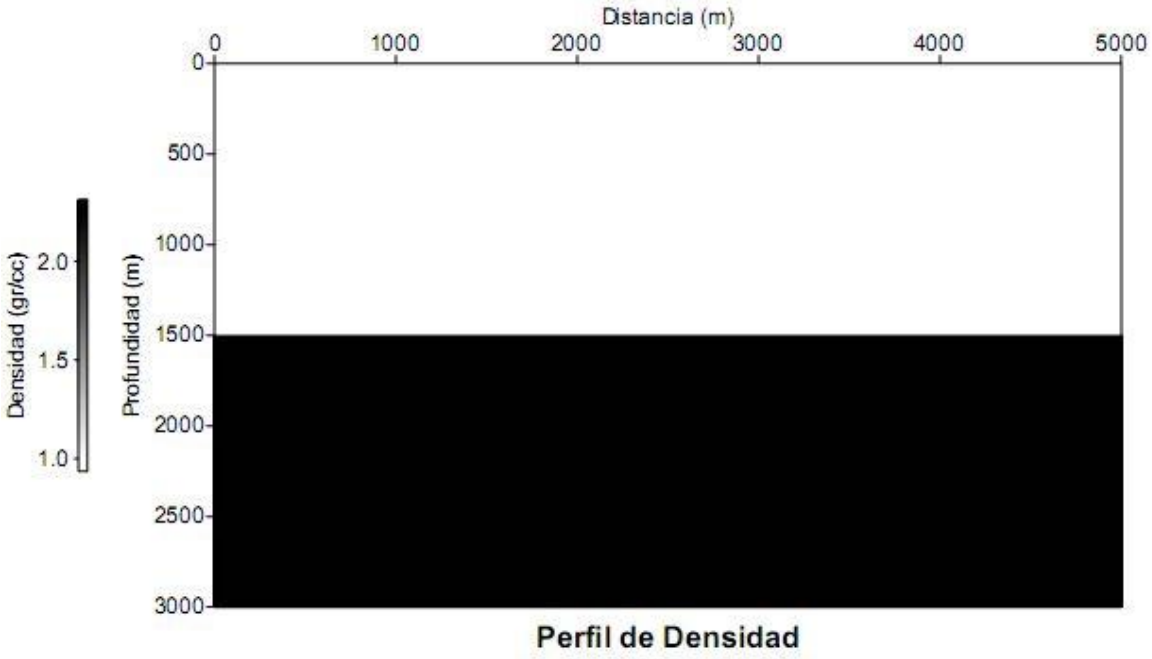


Figura 4.2: Perfil de densidad a partir del cual se hará el modelo de una adquisición sísmica (Modelo1).

4.5 Generación de los Tiros Sintéticos.

En esta parte se generan el conjunto de Tiros Sintéticos resultantes del modelado de la adquisición sísmica. Para esto fue necesario construir cada uno de los perfiles de velocidad y densidad relacionados con cada disparo. Es decir, un perfil para el disparo en la posición 1500 m, el cual concierne al tendido desde 0 hasta 3000 metros. Luego para el tendido correspondiente al disparo en la posición 1550 m (tendido desde 50 hasta 3050 metros) y así sucesivamente. Esto se hace porque para cada disparo hay que realizar un modelado de ondas acústicas y para esto es necesario tener los perfiles de velocidad y densidad que intervienes en cada disparo. Para crear los **Tiros Sintéticos** se realizó un código iterativo que efectúa un número de ciclos igual a la cantidad de disparos que conforman la adquisición.

Por medio del **Código 4.2** se obtienen todos los Tiros Sintéticos del modelo. A través de los valores definidos en los parámetros de entrada, el **Código** ajusta 40 disparos.

Por esto son retornados 40 archivos principales, en cada uno se encuentra un **Tiro Sintético** correspondiente a cada disparo. Estos archivos son nombrados según la posición del disparo que lo generó, es decir, el Shot Gather relacionado con el disparo en la posición 1500 m se llama "gather {1500}", el generado por el disparo a 1550 m es "gather {1550}" y así sucesivamente.

Los parámetros de entrada para este Código son los mismos que fueron definidos en el **Código 4.1.**

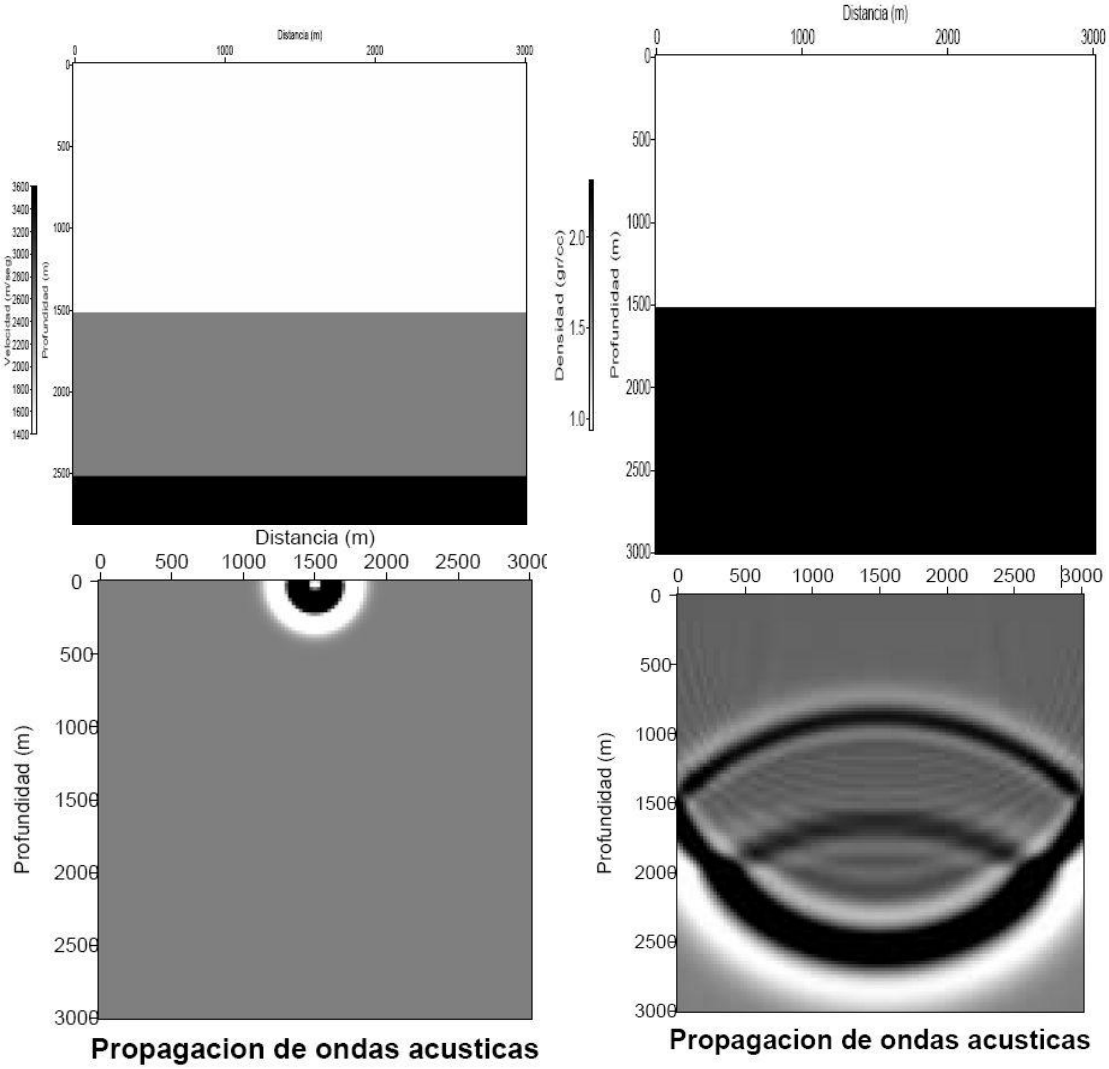
Código 4.2: Generación de los Shot Gathers correspondientes al modelado de una adquisición sísmica 2D (Modelado 1)

```
#! /bin /sh

#Nota: Los parametros de entrada son los definidos en el Script 4.1

LIMIT=$((ND*ID-ID))

for ((fx=0 ; fx <= LIMIT ; fx=$((fx+ID)) ))

do

#Generacion de los perfiles de velocidad correspondiente a cada tendido

unif2 < $Archivo_Entrada fx=$fx nx=$nx dx=$dx nz=$nz dz=$dz \v00=$v00 > vel.out

#Generacion de los perfiles de densidad correspondiente a cada tendido

unif2 < $Archivo_Entrada fx=$fx nx=$nx dx=$dx nz=$nz dz=$dz \v00=1.0,2.25.2.25 > den.out

#Graficacion del perfil de velocidad del tendido (x window graphic)
```

ximage < vel.out title="Perfil de Velocidad del tendido" \f2=$fx n2=$nx d2=$dx n1=$nz d1=$dz \legend=1 units='Velocidad (m/seg)' \label1='Profundidad (m)' label2='Distancia (m)' \wbox=800 hbox=800 lx=0.0

Para cada ciclo del Código 4.2 se obtienen las siguientes gráficas:

- Perfil de velocidad relacionado con el tendido correspondiente (Figura 4.3)
- Perfil de densidad del tendido correspondiente (Figura 4.3)
- Perfil de propagación de ondas acústicas (Figura 4.3)
- Tiro Sintético del disparo correspondiente (Figura 4.3)



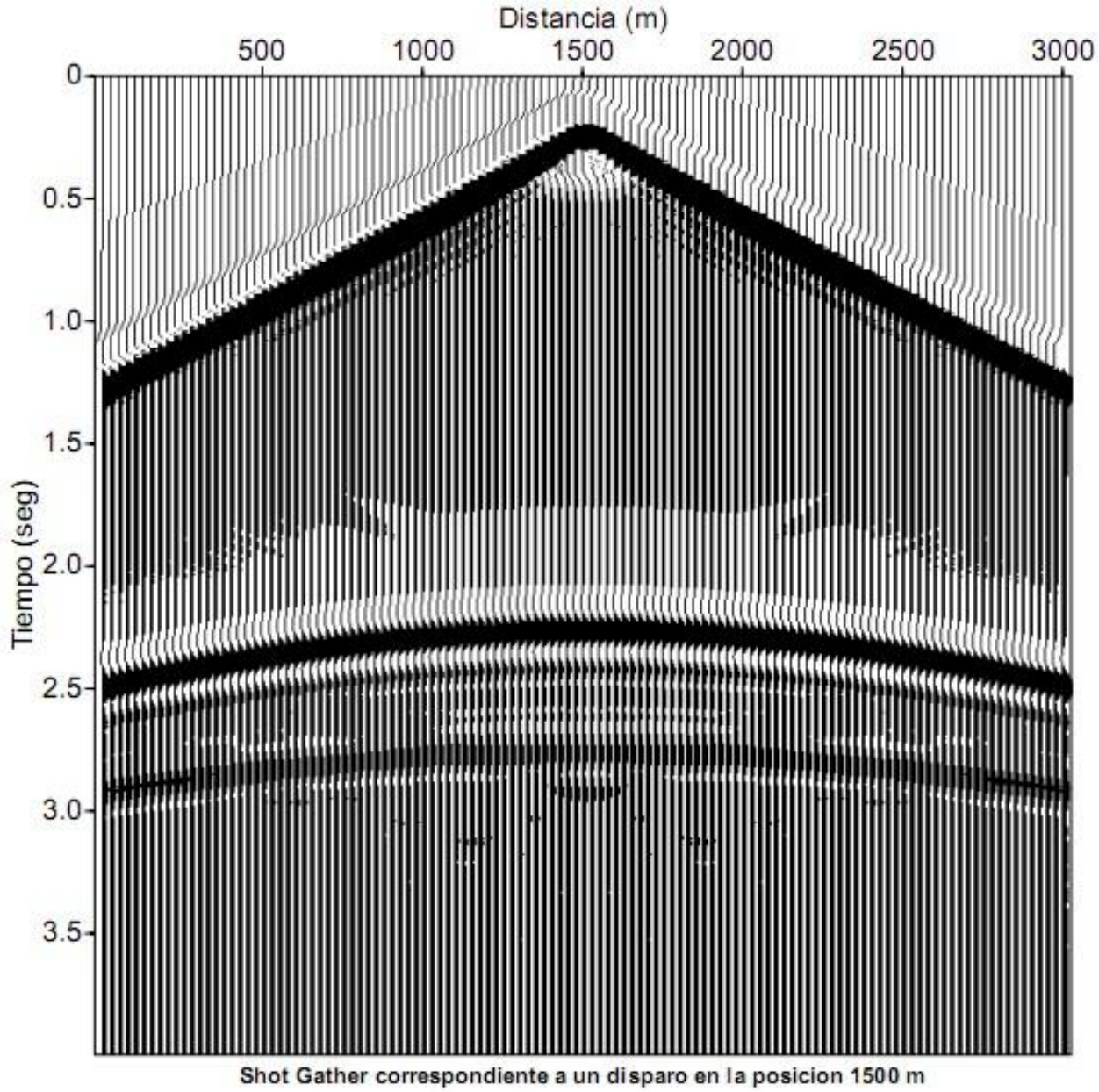


Figura 4.3: Resultados obtenidos para el primer ciclo (primer disparo) del Script 4.2. En las dos primeras graficas se muestran los perfiles de velocidad y en seguida el de densidad, un poco más abajo se muestra la propagación de ondas en los perfiles citados con anterioridad en el punto del tendido de 1500 mts. el cual corresponde a 0 a 3000 mts.

CAPITULO V

5. Procesamiento básico con Seismic Unix

5.1 El encabezado de un archivo sísmico

5.1.1 Visualización del encabezado

5.1.2 Ajuste del encabezado

5.2 Análisis de velocidad

5.3 Corrección por NMO

5.4 Apilamiento

Capitulo 5

Procesamiento básico con Seismic Unix

5.1 El encabezado de un archivo sísmico

5.1.1 Visualización del encabezado

Los comandos del paquete Seismic Unix que manejan archivos sísmicos requieren que los datos estén en formato SU (trazas sísmicas). Este formato está basado en parte del formato SEG-Y que contiene las trazas (Sección 2.5). En este orden, el formato SU adopta el **Encabezado** de las trazas del formato SEG-**Y [Referencia 2]**.

El **Encabezado** de un archivo con formato SU, por ejemplo un **Tiro Sintético**, contiene un conjunto de parámetros que son entendidos por los programas adecuados del paquete de Seismic Unix para poder realizar las operaciones pertinentes. Los comandos **SURANGE** y **SUGETHW** de SU, nos permiten observar el encabezado de las trazas de un archivo con formato SU. Para explicar el funcionamiento de estos dos comandos vamos a emplear el archivo "gather" obtenido en la sección 3.2 a través del Script 3.3.

Surange

El comando **SURANGE** nos da el rango de los parámetros del Header, como se observa a continuación:

surange < gather

La línea del comando anterior genera los siguientes resultados:

tracl 1 121 1 (1 – 121)

tracr 1 121 1 (1 – 121)

trid 1

offset -1500 1500 (-1500 -1500)

sx 1500

ns 960

dt 4166

los parámetros que nos interesa explicar en estas instancias son:

tracl - Secuencia numérica de las trazas de un archivo sísmico.

offset - Distancia entre la fuente y los receptores.

sx - Ubicación x de la fuente.

gx - Ubicación x de los receptores

SUGETHW [SU GET HEADER WORD]

Utilizando el comando **SUGETHW** podemos desplegar toda la secuencia de valores de los parámetros mostrados arriba. Esto se hace de la siguiente forma:

sugethw < gather key=tracl,trac,offset,sx,gx | more

"key" es utilizado para indicar cuáles son los parámetros del header que se quieren observar.

El comando **MORE** permite desplegar solamente la secuencia de valores que cabe en la pantalla. Los valores restantes irán apareciendo al oprimir la tecla ENTER. De esta manera se desplegara lo siguiente en la pantalla:

t r a c l =1 t r a c r =1 o f f s e t =−1500 sx=1500 gx=0

t r a c l =2 t r a c r =2 o f f s e t =−1475 sx=1500 gx=25

t r a c l =3 t r a c r =3 o f f s e t =−1450 sx=1500 gx=50

t r a c l =4 t r a c r =4 o f f s e t =−1425 sx=1500 gx=75

t r a c l =5 t r a c r =5 o f f s e t =−1400 sx=1500 gx=100

. . . . .

. . . . .

. . . . .

t r a c l =121 t r a c r =121 o f f s e t =1500 sx=1500 gx=3000

Los resultados mostrados arriba corresponden a los de un solo Shot Gather. Para explicar el header de un archivo que contenga varios SHOTS vamos a emplear el siguiente archivo **modeldata.**

surange < modeldata

768 t r a c e s :

tracl	**1 768 (1 – 768)**
tracr	**1 768 (1 – 768)**
fldr	**1 12 (1 – 12)**
tracf	**1 64 (1-64)**
cdp	**350 4600 (1450-3500)**
trid	**1**
offset	**-6400 -100 (-100 - -6400)**
sx	**300 1400 (1400 – 300)**
gx	**400 7800 (1500 – 6700)**
counit	**1**
ns	**501**
dt	**4000**
d2	**0.100000**
f2	**0.100000**

En este caso es necesario que expliquemos otros tres parámetros:

fldr - Número de archivo (Número de Shot Gather)

tracf - Secuencia numérica de las trazas en el archivo original (Secuencia en cada Shot Gather)

cdp - Punto medio común

Los valores del parámetro "**fldr**" indican que el archivo "**modeldata**" está conformado por un total de 12 Tiros Sintéticos. En este caso el parámetro "**tracl**" contiene la secuencia numérica de todas las trazas del archivo sin importar el número de Tiros Sintéticos. El parámetro "**tracf**" se refiere a la secuencia de las trazas de cada archivo (cada Tiro Sintético).

Utilizando el comando S**UGETHW**:

sugethw < modeldata key=tracl, fldr, tracf, cdp, offset, sx, gx | more

Nos mostrará la siguiente secuencia de datos de todos los disparos.

tracl=1 fldr=1 tracf=1 cdp=1450 offset=−100 sx=1400 gx=1500

tracl=2 fldr=1 tracf=2 cdp=1500 offset=−200 sx=1400 gx=1600

tracl=3 fldr=1 tracf=3 cdp=1550 offset=−300 sx=1400 gx=1700

tracl=4 fldr=1 tracf=4 cdp=1600 offset=−400 sx=1400 gx=1800

tracl=5 fldr=1 tracf=5 cdp=1650 offset=−500 sx=1400 gx=1900

tracl=6 fldr=1 tracf=6 cdp=1700 offset=−600 sx=1400 gx=2000

. . . . . .

. . . . . .

. . . . . .

tracl=64 fldr=1 tracf=64 cdp=4600 offset=−6400 sx=1400 gx=7800

tracl=65 fldr=2 tracf=1 cdp=1350 offset=−100 sx=1300 gx=1400

tracl=66 fldr=2 tracf=2 cdp=1400 offset=−200 sx=1300 gx=1500

tracl=67 fldr=2 tracf=3 cdp=1450 offset=−300 sx=1300 gx=1600

. . . . . .

Secuencia de todos los disparos restantes (Desde el 3 hasta el 12)

. . . . . .

tracl=764 fldr=12 tracf=60 cdp=3300 offset=−6000 sx=300 gx=6300

tracl=765 fldr=12 tracf=61 cdp=3350 offset=−6100 sx=300 gx=6400

Como se pudo ver, el comando **SUGETHW** nos permite observar la secuencia de todos los parámetros que se encuentran en el header de un archivo con formato SU. Es posible seguir esta secuencia en las partes punteadas del ejemplo anterior.

A través de los resultados mostrados por los comandos **SURANGE** Y **SUGETHW** podemos llegar a las siguientes conclusiones:

El archivo "**modeldata**" concuerda con una adquisición sísmica con los siguientes parámetros:

Número de disparos: 12

Número de grupos de receptores: 64

Intervalo entre puntos de disparo: 100

Intervalo entre grupos de receptores: 100

Ubicación x de los disparos sx :=1400,1300,1200 …. 300

Ubicación x de los grupos de receptores (gx): 1400,1300.1200 ….. 300

En algunos casos puede ser necesario interactuar con el header de las trazas para editar o definir algunos parámetros. Previendo esto fueron programados los comandos **sushw** y **suchw** del paquete SU. Estos comandos son de gran importancia, por ello serán explicados a continuación:

5.1.2. Ajuste del Header

El siguiente contenido fue obtenido de la **referencia [2]**.

Este comando nos permite ajustar uno o varios parámetros del header de un archivo con formato SU.

Sintaxis: **SUSETHW < SU data.in [Parámetros opcionales] > SU data.out**

Parámetros requeridos: Ninguno.

Parámetros opcionales:

key=cdp,...Parámetro(s) del encabezado "key" a ser ajustado.

a=0,..... Valor(s) en la primera traza.

b=0,..... Incremento dentro del grupo(s).

c=0,..... Incremento del grupo(s).

d=0,..... Cambio(s) en el número de traza.

j=ULONG_MAX,ULONG_MAX, ...Número de elementos del grupo.

Ejemplo 1

sushw < data.su key=dt a=2000 > data.out.su

En este caso hemos ajustado "**dt**" a un valor constante igual a 2000.

Ejemplo 2

Si queremos ajustar "**sx**" en las primeras 5 trazas igual a 6400, las segundas 5 a 6300, y seguir decreciendo con un factor de -100 por cada grupo de 5 trazas, debemos colocar:

sushw < data.su key=sx a=6400 c=-100 j=5 > data.new.su

Ejemplo 3

Para ajustar el campo del offset correspondiente a cada grupo de 5 trazas a los valores: (200, 400, 600, 800, 1000) se coloca:

sushw < data.su key=offset a=200 b=200 j=5 | sugethw key=offset | more

Del Ejemplo 3 se desplegaría:

o f f s e t =200

o f f s e t =400

o f f s e t =600

o f f s e t =800

o f f s e t =1000

o f f s e t =200

o f f s e t =400

. . . . . .

También se pueden ajustar cualquier cantidad de parámetros en una sola ejecución del comando **sushw.**

Ejemplo 4

En el siguiente ejemplo vamos ajustar tres parámetros al mismo tiempo que el dt=2000 para todas las trazas, el sx vaya disminuyendo cada 100 en cada grupo de 5 trazas y que el offset vaya aumentando cada 100 en cada traza para un grupo de 5 trazas.

sushw < data.su key=dt,sx,offset a=2000,6400,200 b=0,0,200 c=0,-100,0 j=0,5,5 | sugethw key=dt,sx, offset | more

dt=2000 sx=6400 o f f s e t =200

dt=2000 sx=6400 o f f s e t =400

dt=2000 sx=6400 o f f s e t =600

dt=2000 sx=6400 o f f s e t =800

dt=2000 sx=6400 o f f s e t =1000

dt=2000 sx=6300 o f f s e t =200

dt=2000 sx=6300 o f f s e t =400

dt=2000 sx=6300 o f f s e t =600

dt=2000 sx=6300 o f f s e t =800

dt=2000 sx=6300 o f f s e t =1000

dt=2000 sx=6200 o f f s e t =200

dt=2000 sx=6200 o f f s e t =400

El Encabezado de una traza sísmica con formato SU también puede ser ajustado manualmente convirtiéndolo en un archivo ASCII que puede ser modificado mediante cualquier editor de texto

Esto se logra a través de la implementación de "**output=geom**" en el comando **SUGETHW:**

sugethw < sudata key=key1,key2,... output=geom > hdrfile

El archivo de salida "**hdrfile**" presenta formato ASCII y contiene los valores de los parámetros indicados por key. A partir de este archivo podemos realizar cualquier variación de los parámetros extraídos. Luego de que hayamos hecho todos los cambios requeridos, el archivo "**hdrfile**" puede ser convertido a formato binario a través del comando **A2B**:

a2b < hdrfile n1=nfields > binary_file

Donde "**nfields**" es el número de parámetros extraídos a través de key=...lista especificada arriba. Luego de la carga del nuevo header se hace de la siguiente forma:

sushw < sudata infile=binary_file key=key1,key2,...(lista de arriba) > sudata.edited

Este método resulta una buena vía cuando la geometría que queremos ajustar es muy compleja.

SUCHW [SU CHANGE HEADER WORD] [Referencia 2]

Como su nombre lo indica, este comando tiene la función de cambiar o procesar parámetros del Encabezado (header words) en datos con formato SU.

Es importante tener en cuenta que algunos de los parámetros que deben ser definidos en el encabezado de una traza sísmica pueden ser procesados a partir de parámetros existentes. Tal es el caso del parámetro "**cdp**".

El comando **suchw** nos va a permitir realizar este tipo de tareas.

suchw - Cambia parámetros del header usando uno o dos campos ya definidos.

Sintaxis: **suchw < SU data.in [Parámetros opcionales] > SU_data.out**

Parámetros requeridos: Ninguno.

Parámetros opcionales:

key1=cdp,... Parámetro(s) "key" resultante (salida)

key2=cdp,... Par' metro(s) "key" de entrada

key3=cdp,... Parámetro(s) "key" de entrada

a=0,..... Reforma total del parámetro(s)

b=1,..... Factor del primer parámetro "key(s)"

c=0,..... Factor del segundo parámetro "key(s)"

d=1,..... Factor total

Los valores para los parámetros "key1" son calculados mediante los valores de "key2" y "key3" a través de la siguiente fórmula:

val (key1) = (a + b * val(key2) + c * val(key3)) / d

Ejemplo 1

Para cambiar el valor del parámetro "**cdp**" por una cantidad constante, por ejemplo -1, se debe colocar:

suchw < sudata.in key1=cdp a=-1 > sudata.out

De esta forma, el valor del cdp será igual a -1 para todas las trazas.

Ejemplo 2

Para sumar una cantidad constante, por ejemplo 1000, en algún campo del encabezado, por decir "**tracl**", se coloca:

suchw < infile key1=tracl key2=tracl a=1000 > outfile

Ejemplo 3

Una de las aplicaciones más importantes de este comando es cuando se quiere ajustar el valor de "**gx**" (posición de los receptores) mediante la suma del "**offset**" y la posición de la fuente "sx" y luego calcular el valor de "**cdp**" a través del promedio entre "**sx**" y "**gx**". Aquí estamos usando la posición actual del cdp como el número de cdp, en lugar de la enumeración convencional 1,2,3.

Este procedimiento es hecho de la siguiente forma:

suchw < indata key1=gx key2=offset key3=sx b=1 c=1 |

suchw key1=cdp key2=gx key3=sx b=1 c=1 d=2 > outdata

Es posible realizar ambas operaciones en la misma línea:

suchw < indata key1=gx,cdp key2=offset,gx key3=sx,sx b=1,1 c=1,1 d=1,2 > outdata

5.2 Análisis de velocidad

Uno de los métodos más utilizados para realizar un análisis de velocidad a partir de una traza(s) sísmica es el Espectro de velocidades. De aquí se derivan varios tipos de medidas de coherencia que pueden ser usados como atributos en el cálculo de espectros de velocidad.

Para este trabajo se utiliza un atributo conocido como semblanza. En el Apéndice A se encuentra la teoría que envuelve la realización de un análisis de velocidad y se especifica el caso particular de un análisis de semblanza.

En resumen, lo que se hace en un análisis de semblanza, es calcular cada cierto número de muestras temporales de un **CMP gather**, el valor del atributo de (semblanza) para un rango de velocidades convenientemente ajustado. El archivo generado será visualizado en un mapa que muestre el valor de semblanza para cada punto (**Vstk, t**). Donde **Vstk** es la velocidad de apilamiento y **t** es el tiempo en el **CMP Gather**. La velocidad correspondiente al valor máximo de semblanza para un evento en particular representa la velocidad que mejor apila tal evento. Mediante el paquete Seismic Unix podemos realizar un análisis de semblanza a partir de un CMP gather mediante el programa **SUVELAN**.

SUVELAN

Utilidad: Calculo de semblanza para las velocidades de apilamiento de un CMP gather.

sintaxis**: suvelan < infile [Parámetros Opcionales] > outfile**

Archivo de entrada (infile): CMP gather en formato SU

Parámetros

Parámetros requeridos: Ninguno

Parámetros opcionales:

nv=en este caso 50 ..... Número de velocidades

dv= en este caso 50.0 ..... Intervalo de muestreo de las velocidades

fv= en este caso 1500.0 ..... Primera velocidad

Lo que se define mediante **nv**, **dv** y **fv** es el rango de velocidades para las cuales se calculará el valor de semblanza. Este rango debe ser ajustado tomando en cuenta las velocidades máxima y mínima que posiblemente se puedan encontrar. El parámetro **dv** (intervalo de muestreo de las velocidades) está relacionado con la precisión del método, esto porque a medida que **dv** se hace más pequeño, la distancia entre las velocidades para las cuales se calculará la semblanza se hace menor, lo cual nos va a permitir obtener una mayor exactitud de los valores calculados.

Archivo de salida (Outfile): El archivo de salida contiene un mayado de valores de semblanza. Es decir, un valor para cada punto (Vstack, t), donde Vstack: Velocidad de apilamiento; t: Tiempo del CMP GATHER.

El código 5.1 muestra un caso bastante básico en la utilización del comando **SUVELAN**. En este Código se realiza un análisis de semblanza para el **CMP gather** que se encuentra en la figura 5.2.1 (**CMPgather.in**). Los resultados obtenidos a través de **suvelan** son representados en el mapa de semblanza de la Figura 5.2.1 b (**semblanza.out**).

Código 5.1: Análisis de velocidad a través del cálculo de la semblanza para el CMP gather de la Figura 5.2.1

```
#! /bin /sh

suvelan < CMPgather.in nv=120 dv=75.0 fv=4000.0 > semblanza.out
```

En el Código anterior el archivo de entrada (**CMPgather.in**) es un **CMP gather** sintético de cuatro (4) eventos generado mediante el comando **susynlv** y cuya gráfica es la correspondiente a la Figura 5.2.1, la cual fue realizada a través del comando **SUPSIMAGE.**

Los parámetros ajustados (**nv**=120 **dv**=75.0 **fv**=4000.0) indican que el valor de semblanza será calculado para las velocidades comprendidas entre 4000 y [4000+((120*75)-75)] m/seg. Es decir, entre 4000 y 12925 m/seg.

Con un intervalo entre cada valor de velocidad de 75 m/seg (4000, 4075, 4150, 4225, 4300,...,12925). El archivo de salida (**semblanza.out**) contiene los valores de semblanza calculados.

Este archivo se muestra en la Figura 5.2.1

En la Figura 5.2.1 se puede observar que cada evento del **CMP GATHER** corresponde con un pico de semblanza. El par (**Vstack,**t) relacionado con alguno de los picos de de semblanza nos da los valores de **Vnmo** y **Tnmo** que deben ser ingresados a la fórmula de corrección **NMO** para corregir el evento relacionado. En la práctica no se hace distinción entre la velocidad de apilamiento y la velocidad **NMO**. Por esto, las velocidades obtenidas mediante el análisis de semblanza, las cuales son velocidades de apilamiento, serán utilizadas como velocidades nmo (V_{nmo}) y serán ingresadas en la fórmula de corrección por NMO para eliminar la influencia del offset en los **CMP gathers**.

A partir del mapa de semblanza (Figura 5.2.1 (b)) se deben visualizar los puntos que presenten el mayor valor de semblanza para cada evento. Las gráficas desplegadas mediante **SUXIMAGE** tienen la opción de almacenar, en un archivo de texto, los valores de los ejes 1 y 2 correspondientes a la ubicación del cursor del mouse al marcar la tecla “s” (en nuestro caso el eje 1 es tiempo y el eje 2 es velocidad).

De esta manera, se debe desplazar el cursor hasta el punto que muestre el mayor valor de semblanza (pico de semblanza) para un evento reconocido y presionar la tecla s para almacenar los valores **(Vnno, Tnmo**).

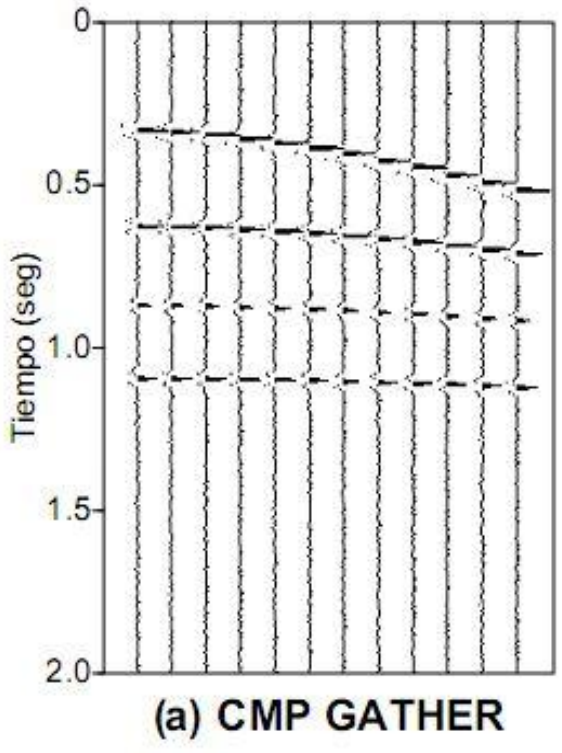
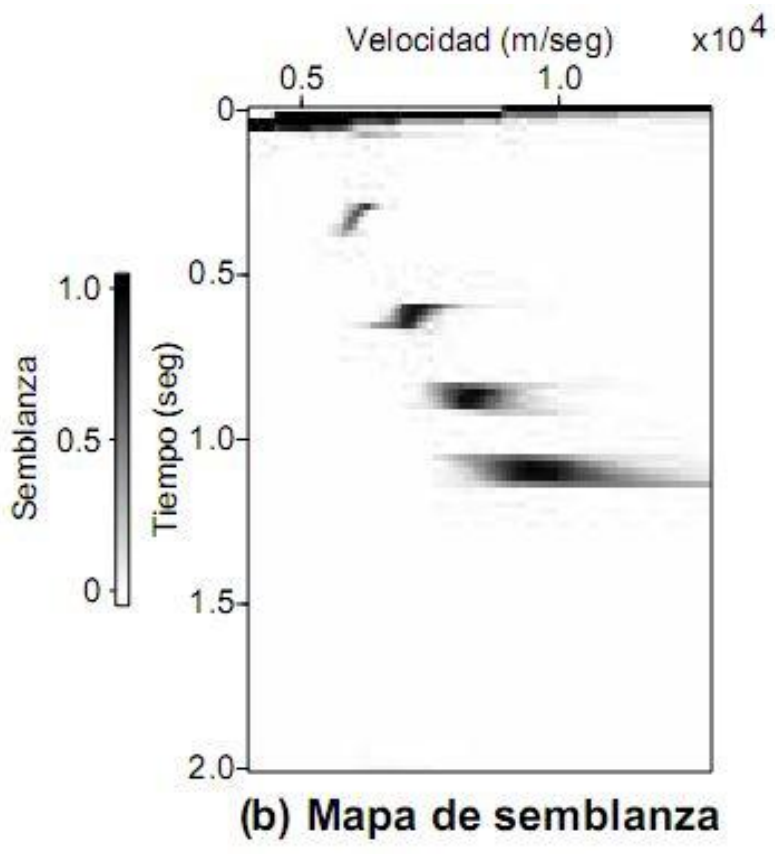


Figura 5.2.1: (a): CMP Gather de 4 eventos. (b): Mapa de Semblanza asociado al CMP Gather de (a)



A partir del mapa de semblanza (Figura 5.2.1 (b)) se deben visualizar los puntos que presenten el mayor valor de semblanza para cada evento. Las gráficas desplegadas mediante **SUXIMAGE** tienen la opción de almacenar, en un archivo de texto, los valores de los ejes 1 y 2 correspondientes a la ubicación del cursor del mouse al marcar la tecla "s" (en nuestro caso el eje 1 es tiempo y el eje 2 es velocidad).

De esta manera, se debe desplazar el cursor hasta el punto que muestre el mayor valor de semblanza (pico de semblanza) para un evento reconocido y presionar la tecla s para almacenar los valores **(Vnno, Tnmo**).

Debemos repetir este procedimiento para cada evento y luego oprimir la tecla "q" para terminar la selección de puntos y cerrar el mapa. Después de realizar este procedimiento se habrá creado un archivo de texto con el nombre especificado en el parámetro "**mpicks**" de **XIMAGE**.

Por ejemplo, si se coloca mpicks=mpicks.cdp, se obtendrá un archivo de texto con el nombre mpicks.cdp el cual contendrá los valores correspondientes a las velocidades y los tiempos seleccionados (Vnmo, Tnmo). El archivo se verá como el siguiente:

mpick.cdp: Archivo obtenido de la selección de puntos realizada en el mapa de semblanza de la Figura 5.1. La primera columna corresponde con los tiempos seleccionados **(tnmo**) y la segunda con las velocidades (**Vnmo**).

1.101 9791

0.87 8234

0.6292 7010

0.3435 5972

En el archivo anterior el número de líneas debe ser igual a la cantidad de picos de semblanza seleccionados y por ende igual a la cantidad de eventos localizados en el **CMP GATHER**.

El par ubicado en la primera línea corresponde con el primer punto seleccionado en el mapa de semblanza. Ahora, el comando que realiza la corrección **NMO** necesita que los valores temporales estén ordenados de manera creciente, sin depender de la forma como el usuario lleve a cabo la selección de los valores. Por esta razón es necesario ordenar el archivo **mpicks.cdp** de la forma requerida. Para esta tarea utilizáremos el comando **SORT**.

En resumen, lo que hace "**SORT**" es ordenar líneas de texto. El parámetro -n de este comando indica que el ordenamiento es numérico. De esta manera, mediante la siguiente línea de comando el archivo **mpicks.cdp** es ordenado de forma creciente, obteniendo así el archivo **mpicksort.cdp** el cual se muestra abajo.

sort < mpicks.cdp -n > mpicksort.cdp

mpicksort.cdp: Corresponde al mismo archivo mpick.cdp pero ordenado de forma creciente.

0.3435 5972

0.6292 7010

0.87 8234

1.101 9791

A pesar de que este archivo (**mpicksort.cdp**) se encuentra ordenado de forma creciente, el comando que realiza la corrección **NMO** necesita que los datos se encuentren en un formato específico llamado formato par. Por lo tanto, el archivo **mpicksort.cdp** debe ser transformado a este formato, lo cual se logra mediante el comando **MKPAFILE**.

mkparfile: Convierte archivos con formato ASCII en archivos con formato par.

Mediante la siguiente línea seria transformado el archivo mpicksort.cdp en un archivo con formato par:

mkparfile < mpicksort.cdp string1=tnmo string2=vnmo > par.cdp

El archivo generado (par.cdp) contiene los mismos valores almacenados en mpicksort.cdp pero en formato par. Este archivo se muestra a continuación:

par.cdp:

tnmo=0.3435,0.6292,0.87,1.101

vnmo=5972,7010,8234,9791

Mediante el archivo **par.cdp** se podrá realizar la corrección **NMO** para el **CMP gather** contenido en el archivo **CMPgather.in** y cuya gráfica es la correspondiente a la Figura 5.2.1 (a).

En el Código 5.2 se encuentran todos los pasos necesarios para obtener el archivo de texto **par.cdp** utilizado por el comando **SUNMO** para realizar la corrección **NMO** del **CMP Gather** de la Figura 5.1 (a).

Código 5.2: Análisis de velocidad: Obtención del archivo "par" necesario para realizar la corrección NMO del CMP gather de la Figura 5.2.1(a)

```
#! /bin /sh

suxwigb < CMPgather.in title="(a) cmp gather" label1=Tiempo (seg) xbox=700 \mpicks=mpicks.cdp &

suvelan < CMPgather.in nv=120 dv=75 fv=4000.0 > semblanza.out

suximage < semblanza.out f2=4000.0 d2=75.0 mpicks=mpicks.cdp \legend=1 units="Semblanza" \label1="Tiempo (seg)" label2="Velocidad (m/seg)" \title="(b) Mapa de semblanza" \

sort < mpicks.cdp –n > mpicksort.cdp

mkparfile < mpicksort.cdp string=tnmo string2=vnmo > par.cdp
```

5.3 Corrección por NMO

El comando de SU destinado a la corrección por **NMO** es el siguiente **SUNMO**. El archivo **par.cdp** contiene las velocidades y los tiempos **NMO** (**Vnmo**, **Tnmo**) obtenidos a partir del análisis de semblanza realizado para el **CMP gather** de la Figura 5.2.1 (a).

A través de este archivo podemos efectuar la corrección **nmo** del **CMP gather** mencionado. El Código 5.3 realiza esta tarea y gráfica los resultados obtenidos en formato PostScript. El **CMP gather** corregido se encuentra en la Figura 5.3.

Código 5.3: Corrección nmo del CMP gather de la Figura 5.2.1(a), correspondiente al archivo CMPgather.in

```
#! / bin/ sh

## Correción NMO

sunmo < CMPgather.in par=par.cdp > CMPcorregido

## Graficación del CMO Gather corregido

suspswigb < CMPcorregido d2num=10 wbox=3.5 hbox=5 \lable1='Tiempo (seg)' > CMPgatherCorregido.ps
```

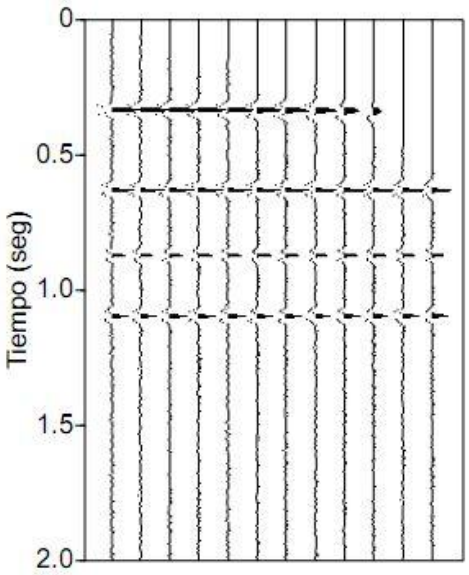


Figura 5.3: CMP gather de la figura 5.2.1 (a) corregido por nmo

5.4 Apilamiento

En esta sección explicaremos como se realiza el apilamiento de la trazas en este caso será solo una traza, pero más adelante se realizara el mismo procedimiento para un conjunto de trazas sísmicas que conforman un **CMP gathers.**

Código 5.4: Apilamiento de las trazas del CMP Gather corregido por NMO de la Figura 5.3.

```
#! /bin /sh

sustack < CMPcorregido > trazaApilada

suspswigb < TrazaApilada title="Traza Apilada" d2num=10 wbox=3.5 hbox=5 \lable1="Tiempo (seg)" > TrazaApilada.ps
```

A partir del **CMP Gather** corregido por **NMO** de la Figura 5.3, el cual corresponde al archivo "**CMPcorregido**" obtenido en el Código 5.3, podemos realizar el apilamiento de las trazas a través del comando **SUSTACK** perteneciente al paquete Seismic Unix. El Código 5.4 realiza esta tarea y gráfica los resultados obtenidos.

En el Código 5.4, el archivo "**TrazaApilada**" es el resultado obtenido del apilamiento de las trazas del **CMP Gather** corregido por **NMO** de la Figura 5.3. Este archivo es gráficado a través del comando **SUPSWIGB**, obteniendo así la Figura 5.5

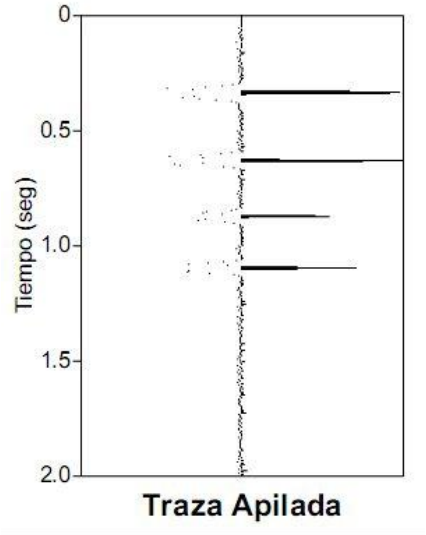


Figura 5.5: Apilamiento del CMP Gather corregido por NMO de la Figura 5.2.1

CAPITULO VI

6. Generación de secciones a partir de datos 2D

6.1 Concatenación de los datos (unión de las trazas sísmicas), ajuste del encabezado y orden del CMP

6.1.1 Concatenación de los Tiros Sintéticos

6.1.2 Ajuste del encabezado

6.1.3 Orden del CMP

6.2 Análisis de velocidad

6.3 Corrección por NMO

6.4 Apilamiento

Capitulo 6

Generación de secciones apiladas a partir de datos 2D

6.1 Concatenación de los datos (unión de las trazas sísmicas), ajuste del Encabezado y orden del CMP.

6.1.1. Concatenación de Tiros Sintéticos (Unión de los Tiros Sintéticos)

En el capituló 4 obtuvimos un conjunto de Tiros Sintéticos mediante los comandos **UNIF2** y **SUFMOD2**, los cuales fueron usados para crear modelos sintéticos. Este conjunto de Tiros recrea el proceso de adquisición de una línea sísmica. Por esta razón, los archivos correspondientes cada uno de los Tiros Sintéticos deben ser almacenados en un solo archivo, porque se hace esto, porque esta de esta forma podremos manejar los datos como un solo conjunto y no como varios, lo cual facilitara la aplicación de algunos procedimientos, que deberán ser aplicados a los datos para obtener una sección apilada.

Específicamente, en este trabajo necesitaremos manejar los datos en un solo archivo para emplear los programas del paquete SU que realizan la corrección por **NMO** y **el apilamiento**.

Para poder realizar esto vamos a usar el siguiente comando el cual se llama **CAT**, este nos va a permitir **concatenar** (unir) todos los archivos de los Tiros Sintéticos en uno solo.

Considerando los 4 de los Gathers generados en el capítulo 4: (**gather {1500}, gather {1550}, gather {1600}, gather {1650}**), el comando **CAT** nos permite obtener un solo archivo que contenga los cuatro Gathers una al lado de otro.

Esto se hace de la siguiente forma:

cat gather{1500} gather{1550} gather{1600} gather{1650} > Shot_Gathers

El archive resultante (**Shot_Gathers**) contiene los Tiros Sintéticos correspondientes a los cuatro primeros disparos del modelado de la adquisición sísmica. Dichos disparos corresponden a las posiciones 1500, 1550, 1600 y 1650 metros respectivamente. La grafica de los cuatro Tiros Sintéticos se hace mediante el comando **SUPSWIGB**.

supswigb < Shot_Gathers >

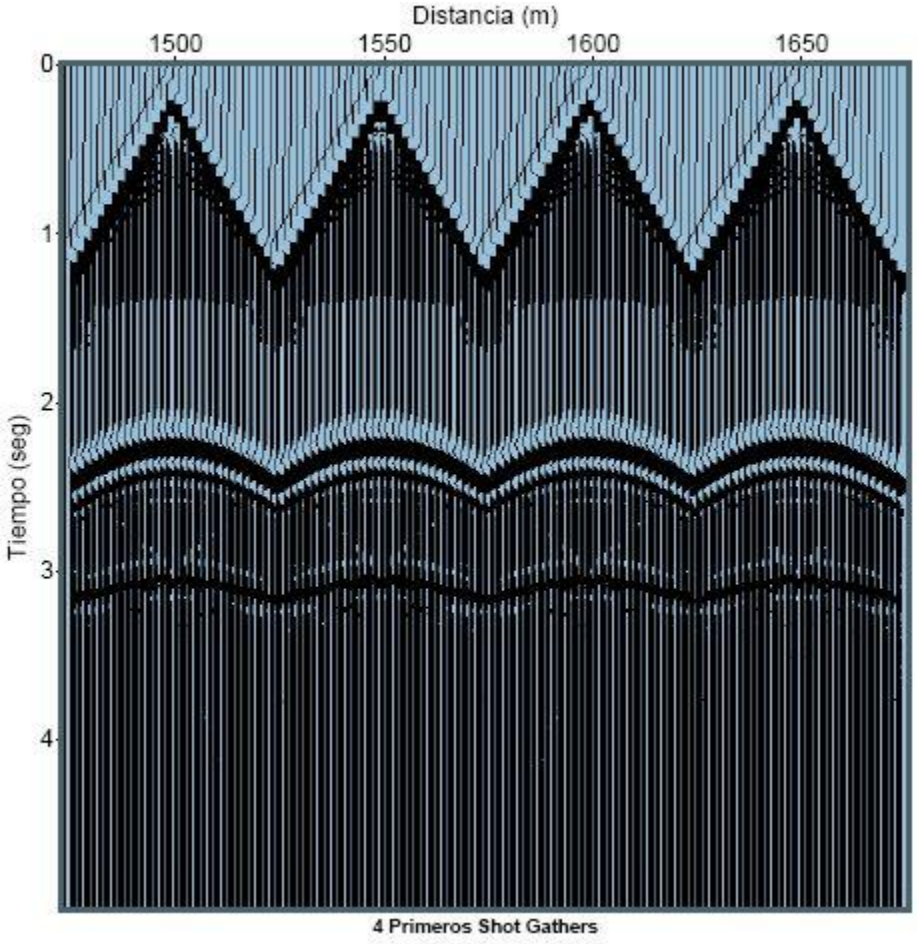


Figura 6.1 Primeros Shot Gathers obtenidos del modelo 2D de la adquisición sísmica.

El procedimiento descrito anteriormente puede ser realizado para cualquier cantidad de Tiros Sintéticos. Sin embargo, cuando el número de archivos (**Shot Gathers**) a introducir es demasiado alto, resulta muy tedioso realizar esta tarea, por esta razón, es de gran interés programar un código que una cualquier cantidad de archivos.

En este trabajo se muestra un Código que realiza esta tarea para un conjunto de Tiros Sintéticos cuyos nombres deben ser de la siguiente forma: **nombre {x + nA}**; donde a es un valor constante y n un numero entero desde 0 en adelante **(n=0….N)**.

En este orden vamos a partir de un conjunto de Tiros Sintéticos denominados de la siguiente forma: nombre {x0}, nombre {x1}, nombre {x2}……. nombre {xn}.

Donde **x0=x+0A, x1=x+1A, x2=x+2ª…. xn=x+nA**.

Para el caso de los archivos creados en el capítulo 4, x=1500 y A=50. De esta manera los nombres obtenidos son: nombre {1500}, nombre {1550}, nombre {1600}…… nombre {1500+50*N}. Lo que concuerda con los nombres de los archivos citados (gather {1500}, gather {1550}, gather {1600}….).

Haciendo una serie de ajustes en los parámetros para obtener un código general que determine las condiciones de acuerdo a los parámetros de adquisición:

Primer_Archivo (el primer Tiro Sintetico), offset máximo, numero de disparos (ND) y el intervalo entre disparos (ID). A través de estos datos el código es capaz de unir todos los Shot Gathers generados en el modelo de una adquisición sísmica.

Código 6.1: Concatenación de los Shot Gathers obtenidos en el capítulo 4

```
#/bin /sh

### Parametros de entrada
# Primer Shot Gather

Primer_Archivo=gather { 1500 }

offset_max=1550

# Numero de disparos

ND=40

# Intervalo entre disparos

ID=50

suwind < $Primer_Archivo j=1 > SHOTS

LIMITE=$((ND*ID-ID+offset_max))

for ((x=$((offset_max+ID)) ; x <= LIMITE ; x=$((x+ID))  ))

do

cat SHOTS gather [$x} > shots

suwind < shots j=1 > SHOTS

done
```

6.1.2. Ajuste del Encabezado de una traza sísmica

Los comandos que manejan las trazas sísmicas se basan en la información contenida en el encabezado, por medio de esta información podemos realizar las operaciones pertinentes que deseemos.

Por lo tanto, si esta información no es correcta, los procedimientos aplicados a los datos generaran serán erróneos. En el encabezado de una traza sísmica se encuentra toda la información relacionada con los parámetros de adquisición (**distancia entre geófonos, intervalo de muestreo, offset, etc.**), por lo tanto, los valores registrados deben corresponder con la geometría y parámetros de adquisición empleados.

En el **capítulo 4** se encuentran todos los parámetros y la geometría utilizada en el modelado de la adquisición realizada. De acuerdo con estos datos podemos saber cómo debe estar configurado el Encabezado de la traza sísmica generada. Observando tal Encabezado, correspondiente al archivo "**SHOTS**" obtenido en el código anterior (6.1) nos percatamos que varios parámetros del encabezado (llamados en SU "**key Header words**") no corresponden a la configuración que este debería de tener, la cual fue explicada en el **capítulo 4**. Es por esta razón que fue necesario recurrir a los comandos destinados al ajuste de los parámetros del encabezado explicados en el **capítulo 5.1.**

Código 6.2 Ajuste del header del archivo SHOTS

```
#! /bin /sh

LT=3000

NR=121

ND=40

IR=25

ID=50

sushw < SHOTS key=gx a=0 b=$IR  c=$ID j=$NR > SHOTS.HEA

sushw < SHOTS.HEA key=tracl , tracr , tracf , sx , fldr , offset
a=1, 1, 1, $(($LT/2)),1,-$(($LT/2)) b=1,1,1,0,0,$IR c=0,0,0,$ID,1,0
j=$(($NR*$ND)) ,$(($NR*$ND)) ,$NR,$NR,$NR,$NR > SHOTS.HEADER

suschw < SHOTS.HEADER key1=cdp key2=gx key3=sx b=1 c=1 d=2 > SHOTS.CMP
```

En este Código, mediante el comando **SUSHW** realizamos la configuración, en la primera línea, del parámetro "**gx**" y en la segunda los parámetros **tracl**, **tracr**, **tracf**, **sx**, **fldr** y **offset** (todo lo contenido, desde **sushw** < **SHOTS.HEA… hasta…** > **SHOTS: HEADER**, debe ser ejecutado como una solo línea). A través de **SUCHW** analizaremos el parámetro "**cdp**" el cual, obviamente, será necesario el momento de ordenar las trazas por **CMP**.

Los parámetros **LT**, **NR**, **ND**, **IR** e **ID** fueron definidos anteriormente (**capítulo 4**).

El archivo **SHOTS.CMP** contiene todos los Tiros Sinteticos obtenidos en el capitulo 4. En la siguiente figura solo se muestran los primeros cuatro disparos de este archivo.

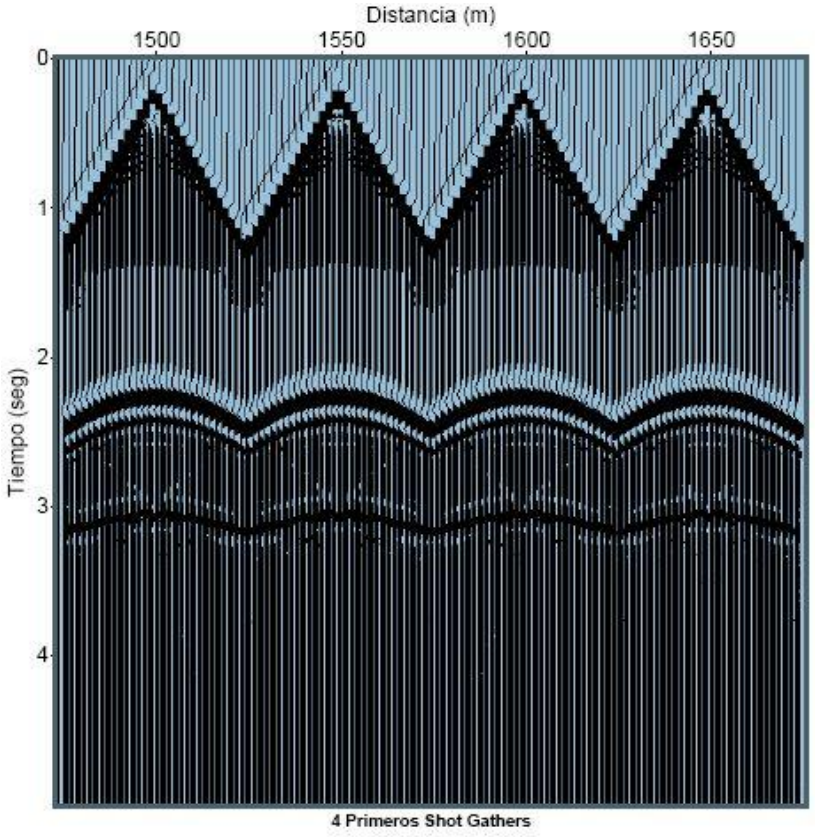


Figura 6.2 Primeros Shot Gathers obtenidos del modelo 2D de la adquisición sísmica.

6.1.3. Orden del CMP

Las trazas sísmicas correspondientes al total de los Tiros Sintéticos (**SHOTS.CMP**) deber ser ordenada por **CMP** para poder realizar la corrección por **NMO** y al apilamiento de los datos. El comando de Seismic Unix (SU) que realiza esta tarea es **SUSORT**. El siguiente Código ordena las trazas contenidas en el archivo **SHOTS.CMP** y grafica 4 de **los CMP Gathers** obtenidos. Esta grafica se muestra en la figura 6.3

Código 6.3 Ordenamiento del archivo SHOT.CDP por CMP y graficación de 4 CMP de los Gathers obtenidos.

```
#! /bin /sh

# Ordenamiento por CMP

susort  < SHOTS:CMP > CMP     cdp

# Graficación de los 4 Gathers (PostScript)

suwind < CMP key=cdp min=2000 max=2040 |supswigb clip=0.25 f2=1994 d2=0.3125 d2=12.5 \
title="CMP Gather" \windowtitle="CMP" \ label1='Tiempo (seg)' label2='Distancia (m) > CMP:ps
```

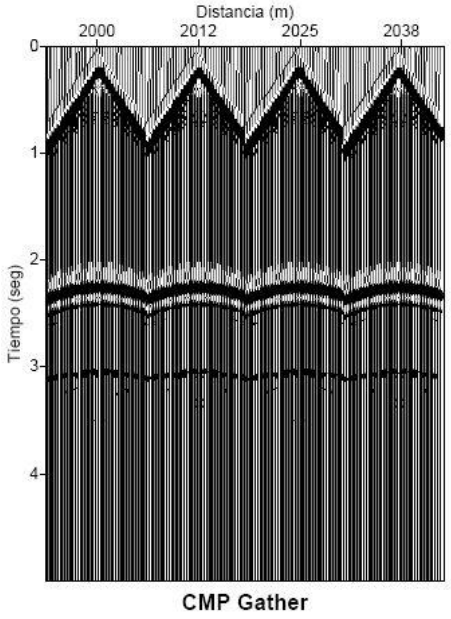


Figura 6.3 CMP Gathers obtenidos a través del modelado de la adquisición sísmica.

6.2 Análisis de velocidad del modelo sísmico generado

Anteriormente aprendimos como hacer un análisis de velocidad para un solo **CMP Gather** a través del cálculo de la semblanza de las velocidades de apilamiento. Cuando tenemos los datos sísmicos (ordenados por CMP) como la que fue obtenida en la sección 6.1.3 en el Código 6.3 debemos realizar un análisis de semblanza correctamente distribuido a lo largo de la línea sísmica. De esta forma será obtenida una función de velocidad de apilamiento para cada **CMP**, lo que nos dará una aproximación de las velocidades a lo largo del total de la línea sísmica. En este sentido, el objetivo principal de esta sección es explicar una rutina que realiza un análisis de velocidad para una línea sísmica con formato SU utilizando los comandos de Seismic Unix. Este código está adaptado de los ejemplos que vienen con el paquete.

Como se menciono con anterioridad, vamos a realizar el mismo procedimiento que en la sección 5.2, pero en lugar de hacerlo solo para uno, lo aremos para todo el conjunto de **CMP Gathers** pertenecientes a la línea sísmica. Con lo dicho. Se puede deducir que el Código que realiza un análisis de velocidad para toda la línea sísmica (Código 6.3) será, prácticamente, una versión iterativa de los procedimientos explicados en la sección 5.2. Tomando en cuenta esto, a continuación serán explicados únicamente los segmentos de código que no hayan sido explicados anteriormente (sección 5.2)

Antes de comenzar con la explicación del Código es importante que tengamos en cuenta que los resultados obtenidos deben ser adecuados para que puedan ser utilizados por el comando que realiza lo corrección por **NMO** (**SUNMO**). Tomando en cuenta esto será prioridad de este Código que los valores calculados sean almacenados en un archivo como el que se muestra a continuación.

Tipo de archivo que se debe obtener luego de realizar el análisis de velocidad:

cdp=1500,2000,2500,3000,3500

tnmo=0.321145,0.626391,0.864865,1.10016

vnmo=6034.07,7051.1,8275.7,9583.31

tnmo=0.333863,0.629571,0.871224,1.10016

vnmo=6013.31,7217.15,8358.72,729687.09

tnmo=0.372019,0.63911,0.868045,1.0938

vnmo=6200.12,7196.4,8420.99,9604.07

En el archivo anterior los valores **tnmo** y **vnmo** que se encuentran en la primera y segunda línea corresponden con cdp=1500, los que se encuentran en la tercera y cuarta corresponden con cdp=2000 y así sucesivamente.
Por otra parte la cantidad de valores de **vnmo** o **tnmo** (separados por comas) deben ser igual al número de reflectores que sean identificados en el **los CMP Gathers**. Para obtener un archivo de esta forma será necesario realizar un conjunto de operaciones a los valores obtenidos mediante el análisis de la semblanza.

Para facilitar la explicación, se dividirá el Código 6.2 en un conjunto de pasos, los cuales serán explicados por separados, los cuales serán comentados en el cuerpo del Código.

1. Ajuste de los parámetros de entrada

- **velpal=**CMP Total de la línea sísmica ordenada por CMP
- **vpicks=**stkvel.p1 Nombre del archivo en donde serán almacenados los valores (t_{NMO},v_{NMO}) resultantes. Este es el archivo que será ingresado al comando que realiza la corrección **NMO**

Selección de los **CMP Gathers**:

- **cdpmin=**1800 Primer CMP al que se le realizará el análisis de la semblanza
- **cdpmax**=2100 Ultimo CMP analizado
- **dcdp**=100 Intervalo entre cada CMP analizado

Los tres parámetros anteriores son los que definen los CMP Gathers a los cuales se les realizara el análisis de la Semblanza. Con los valores anteriores se trabajara con los siguientes, **CMP=1800**, **1900**, **2000**, **2100**.

Selección del rango de velocidades a utilizar

- **nv**=500 Numero de velocidades para las cuales se calculara el valor de la semblanza asociado a ellas
- **dv**=10 Intervalo entre las velocidades

- **fv**=500 Primera velocidad

Selección de los valores del filtro pasa banda

- **f**=1,10,100,120 Frecuencias
- **amps**=0,1,1,0 Amplitudes

Los siguientes parámetros corresponden al comando **UNISAM**, el cual será empleado para graficar la función de la velocidad del apilamiento

- **nout**=3001 Numero de valores en el archivo de salida (Archivo binario)
- **dxout**=0.004 Intervalo de muestreo en x del archivo de salida
- **nd**=$ND Numero de disparos de la adquisición sísmica

2. Definición del ciclo

En este código fue necesario utilizar 2 ciclos.
El primero es empleado para que el programa realice el análisis de la semblanza ingresado (seleccionando previamente los parámetros de entrada). Por esto, tal ciclo debe iterar entre los valores "**cdpmin**" y "**cdpmax**" y el incremento debe ser igual al "**dcdp**".
Por otra parte, cuando la grafica que muestra la función de la velocidad del apilamiento sea desplegada, el programa nos preguntara si estamos de acuerdo con los resultados obtenidos. Si respondemos "si" el programa pasara al análisis del segundo **CMP Gather** (segunda iteración del primer ciclo), si respondemos "no" (n) tendremos una nueva oportunidad de realizar el análisis para el mismo **CMP**. Con esta finalidad es implementado el segundo ciclo.
En resumen el programa comienza con el análisis del primer **CMP Gather** establecido por el ciclo 1 en su primera iteración, luego dentro del segundo ciclo se realizarán todas las operaciones pertinentes al análisis de la semblanza, las cuales serán repetidas para el mismo **CMP Gather** hasta que indiquemos (marcando una S) que estamos de acuerdo con la función de velocidad obtenida. Los ciclos son mostrados a continuación.

```
cdp=$cdpmin

while [ $cdp – le $cdpmax ]

do

ok=false

while [ $ok = false ]

do
```

3. Extracción de cada CMP Gather

Como la línea sísmica está formada por un número muy grade de **CMP Gathers**, debemos extraer un conjunto de ellos. A estos se les realizara el análisis de semblanza y por tanto se obtendrán funciones de velocidad únicamente para los **CMP Gathers** seleccionados (en los parámetros de entrada). Por esto último es importante que la selección se realice de una forma apropiada para que los valores obtenidos describan de una buena manera las velocidades a lo largo del total de la línea sísmica. El siguiente código, extrae y grafica los **CMP Gathers** correspondientes a cada ciclo.

```
## Extracción de cada CMP Gather.

suwind < $velpanel key=cdp min=$cdp max=$cdp count=$ND > panel.$cdp

## Graficacion de los CMP Gathers extraidos.

suxwigb < panel.$cdp title="CMP Gather para cdp=$cdp" \xbox=700 clip=0.6 mpicks=mpicks.$cdp
```

El comando **SUWIND** buscara a lo largo de toda la línea sísmica las trazas cuyos CMP se encuentren dentro del rango establecido mediante los parámetros "**min**" y "**max**".

En este programa "**min**" y "**max**" presentan el mismo valor (**cdp**), el cual corresponde a cada uno de los valores de los **CMP Gathers** definidos por los parámetros "**cdpmin**", "**cdpmax**" y "**dcdp**". Por lo tanto, en este caso, SUWIND recopilara todas las trazas que presenten el valor del único **CMP** especificado a través de la variable "cdp", lo que implica que el archivo de salida contendrá el CMP **GATHER** referido mediante el valor (en el ciclo correspondiente) del parámetro "**cdp**". El parámetro "**count**" tiene la función de fijar un límite para el número de trazas recopiladas. Esto se hace para disminuir el tiempo de ejecución del comando. Es decir, mediante este parámetro el comando detiene el análisis cuando haya cumplido con un número de trazas recopiladas igual al valor especificado por "**count**". En este Código "**count**" es igualado al número de disparos (**ND**) debido a que este es el número mayor de veces que puede ser muestreado un punto durante una adquisición sísmica y por lo tanto el número mayor de trazas que puede presentar un **CMP Gather**. Lo anterior implica que luego de que el comando "**SUWIND**" haya recopilado un número de trazas igual al número de disparos, el análisis de las demás trazas será en vano y por lo tanto solo producirá un aumento en el tiempo de ejecución del programa. Un ajuste inteligente del parámetro "**count**" es de extrema importancia cuando se está analizando datos muy grandes.

El comando **SUXWIGB** es empleado para graficar el **CMP Gather** que fue extraído mediante **SUWIND**.

4. Aplicación de ganancia y filtrado del CMP Gather.

Antes de que el **CMP Gather** sea ingresado al comando que realiza el análisis de la semblanza, es necesario que este posea una buena relación señal-ruido para que los valores de la semblanza se puedan distinguir con facilidad. En busca de esto de le aplica un ganancia y un filtro. Dicha ganancia será manejada mediante el siguiente comando **SUGAIN**, y el filtro mediante **SUFILTER**.

Sugain < panel.$cdp tpow=2 | sufilter f=$f amps=$amps > panel1f.$cdp

El parámetro "**tpow**" indica que la línea sísmica será multiplicada por un factor igual a $\mathbf{t^{pow.}}$

Con **SUFILTER** aplicamos un filtro pasa-banda, es cual esta especificado por las valores de frecuencia (**f**) y amplitud (**amps**) mostrados en el paso 1.

5. Analisis de la semblanza

Después de todo lo anterior, hemos llegado al paso más circunstancial del programa, el punto en donde por fin vamos a calcular la semblanza para las velocidades de apilamiento del **CMP Gather**. Esta tarea será realizada exactamente como se explico en la parte 5.2, las siguientes líneas de código corresponden al análisis de la semblanza.

```
## Análisis de semblanza.

suvelan < pane1F.$cdp nv=$nv dv=$dv fv=$fv > semblanza

## Graficación del mapa de semblanza.

suximage < semblanza  wclip=0.6 bclip=1 f2=$fv d2=$dv \units="semblanza" \label="Tiempo (seg)" label2="Velocidad (m/seg)" \title="Mapa de Semblanza para el CMP $cdp" \picks=mpicks.$cdp \
```

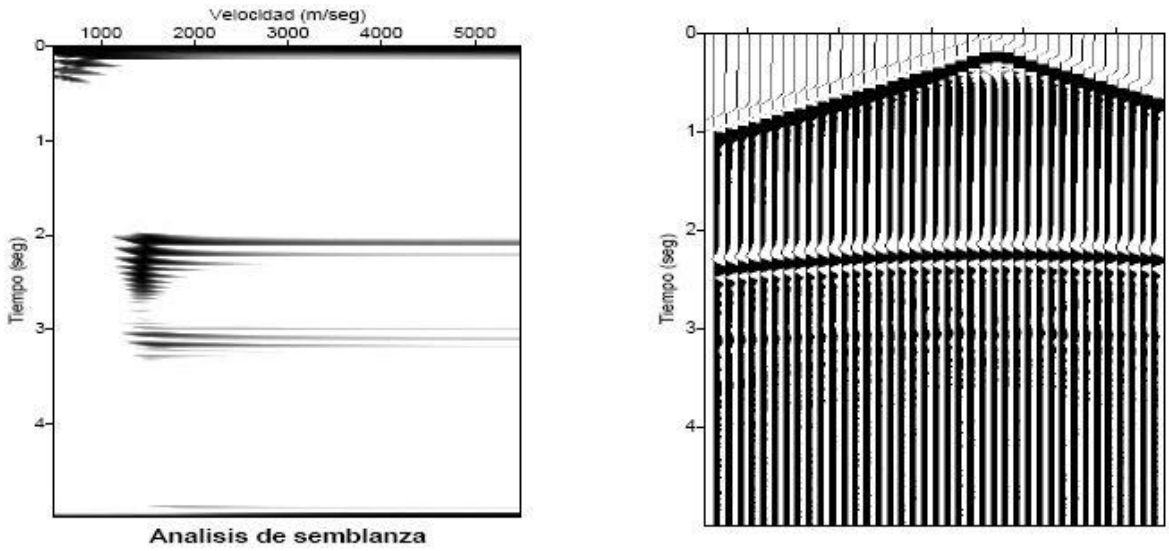


Figura 6.4 A la derecha. CMP Gather correspondiente al primer ciclo del Script anterior (cdp=cdpmin=1800). A la izquierda: Mapa de semblanza obtenido para el CMP Gather mostrado a un costado.

En la figura 6.4 podemos observar: A la derecha el **CMP Gather** que fue extraído a través del comando **SUWIND**, correspondiente al primer ciclo del proceso (**cdo=cdpmin=1800**) y a la izquierda el mapa de la semblanza obtenido para el **CMP Gather** mostrado a un costado.

Se puede apreciar que cada evento coincide con el máximo valor de la semblanza. Estos picos deben ser seleccionados de la forma como se explico en la sección 5.2 para almacenar dichos valores ($\mathbf{v_{NMO}}$ y $\mathbf{t_{NMO}}$) mediante los cuales se realizara la corrección por **NMO** de cada evento.
A partir del mapa de la figura 6.4 obtuvimos el archivo con el nombre **mpic.1800**.

Archivo **mpic.1800**: Valores de velocidad de apilamiento (2da columna) correspondientes a los valores de tiempo mostrados en la primera columna. Cada uno de los valores está relacionado con uno de los picos de la semblanza que son mostrados en la figura 6.4 los cuales a su vez, corresponden con los eventos localizados en el **CMP Gather** mostrado.

El archivo **mpic.1800** como su nombre lo indica fue obtenido a partir del **CMP Garther** en la posición 1800 (**cdp=1800**) en el cual fue el primer CMP que fue ajustado en los parámetros de entrada (**cdpmin**), estos nos indica que nos encontramos en la primera iteración del codigo, inmediatamente se muestra esta parte.

3.052 1563.02
2.05275 1495

6. Obtención del archivo *.par

Para que los valores almacenados en el archivo **mpic.1800** puedan ser utilizados por el comando que realiza la corrección por **NMO**, es necesario que los valores se encuentran en un archivo con formato par **"*.par**" , anteriormente se explico cómo obtener este archivo (sección 5.2).

sort < mpicks.$cdp –n | mkparfile string1-tnmo string2-vnmo > par.$cdp

Para la primera iteración del código anterior se obtendrá un archivo con nombre **par.1800** el cual tiene el formato adecuado para que pueda ser interpretado por el comando **SUMNO**, este archivo contiene la función de velocidad del apilamiento para el **CMP** en la posición de 1800 (**cdp=1800**).

Por lo tanto puede ser usado para efectuar la corrección NMO del CMP Gather correspondiente en este caso en el **CMP** de 1800 metros, tal archivo se muestra a continuación.

tnmo=2.05275,3.052
vnmo=1495,1563.02

7. Generación de la grafica de velocidad de apilamiento

Graficando la velocidad de apilamiento tendremos una idea de la validez de los valores de obtenidos, esto se logra partiendo del hecho de que dicha grafica debe ser parecida a la de una hipérbola, por lo tanto si la grafica que obtenemos no se parece en nada a una hipérbola tendremos que repetir el análisis de la semblanza para cada **CMP Gather** en donde la grafica no se parezca a la de una hipérbola.

Esta grafica será obtenida a partir de los valores de se encuentran en el archivo "**par.$cdp**" , el comando que realiza esta tarea emplea los archivos en forma binaria , por tal motivo tendremos que convertir los valores correspondientes del archivo **par.$cdp** a forma binaria, para realizar la conversión usaremos el comando **UNISAM**, el cual muestrea uniformemente una función **y(z)** especificada a través de los valores (**x,y**).

Este comando emplea un archivo similar a "**par.cdp**", salvo que los valores en x y y deben estar definidos como xin y yin respectivamente, por esta razón debemos editar el archivo **par.$cdp** de manera tal que los términos $\mathbf{t_{NMO}}$ y $\mathbf{v_{NMO}}$ sean remplazados por xin y yin, para esto vamos a emplear el comando "**SED**", tal y como se muestra.

sed $<$ par.$\$$cdp ' s/tnmo/xin/s/vmo/yin/ ' $>$ unisam.p

El archive generado para el primer **CMP Gather** "**unisam.p**" es el siguiente.

xin= 2.09161,3.05166
yin= 1458.65,1470.563

El archivo anterior puede ser utilizado para ejecutar el comando "**UNISAM**" de la siguiente forma:

Unisam < nout=$nout fxout=0.0 dxout=$dxout \par=unisam.p method=spline > varias.u

El archivo obtenido en el paso anterior "**varias.u**" está en formato binario, además este contiene una función la cual esta uniformemente muestreada a partir de los valores del archive "**unisam.p**", que es una función de $\mathbf{v_{NMO}}$ vs $\mathbf{t_{NMO}}$.

Para graficar este archivo (**varias.u**) se emplea el coamndo **XGRAP**:

Xgraph < varias.u n=$nout nplot=1 d1=$dxout f1=0.0 \label1="Tiempo (seg)" label2="Velocidad (m/seg)" \title="Funcion de Velocidad del Apilamiento: CMP Gayher $cdp" \linecolor=2 style=seismic >

La grafica de la función de la velocidad es la siguiente.

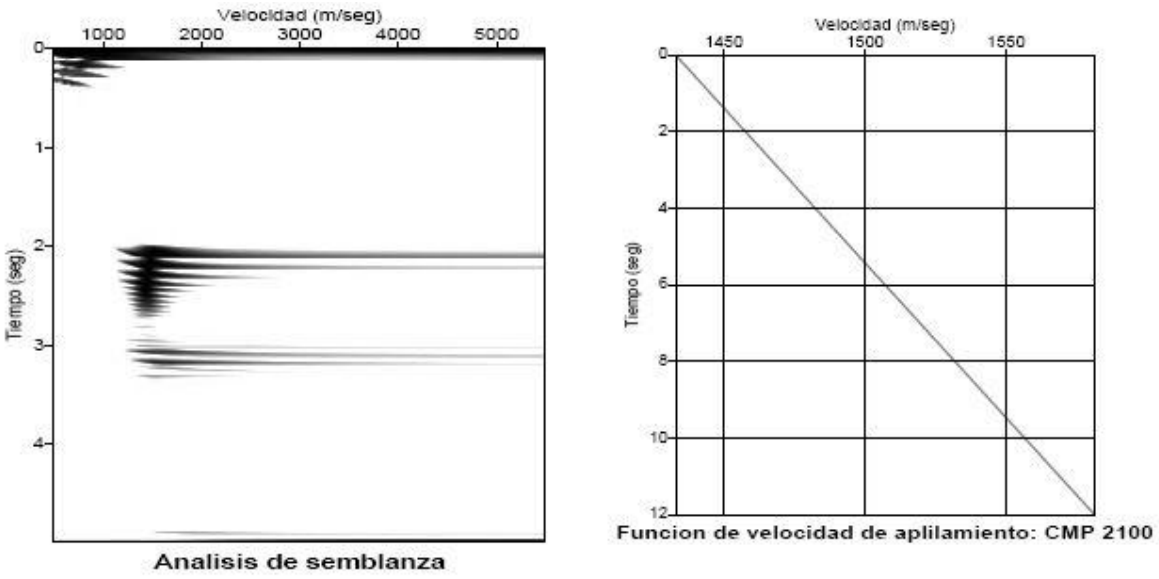


Figura 6.5 Función de velocidad generada a partir del análisis de la semblanza.

8. Análisis de la gráfica obtenida

En el momento es que la grafica sea desplegada en la pantalla, se detendrá la ejecución del programa, inmediatamente después de oprimir la tecla enter nos aparecerá un mensaje en la terminal, el cual es una pregunta, en que si estamos de acuerdo con la función representada por dicha gráfica.

El mensaje es el siguiente.

"¿La selección es correcta? (s/n)"

En esta parte nosotros debemos responder "s" para un sí o "n" para un no y enseguida oprimir de nuevo la tecla enter, en que basaremos nuestra elección en la grafica mostrada y en los valores de semblanza que hemos introducido, si estamos de acuerdo con la grafica desplegada contestaremos "si" y en el caso de que nuestra respuesta sea "no" tendremos otra oportunidad para seleccionar nuevos valores del mapa de semblanza después de seleccionar los nuevos valores tendremos que hacer todo la anterior. Cuando ya estemos de acuerdo con lo que se nos despliega en la pantalla diremos que "si", entonces estos valores se almacenaran y pasaremos al siguiente ciclo del programa , en el cual se realizara el mismo análisis para el **CMP Gather** siguiente .

La parte que realiza esto se muestra enseguida

```
## Consulta de la grafica obtenida

Pause

Echo <  "¿La selección es correcta? (s/n)" | tr-d "\012" > /dev/

tty

read response

case $response in

n*) ok=false ;;

*) ok=true ;;

Esac

Done < /dev/tyy

cdp='bc-1<<end

END

done
```

Explicación del código anterior.

pause: Se encarga de detener la ejecución del programa hasta que la tecla "enter" sea presionada.

echo: Despliega caracteres o cadenas de caracteres en la terminal. La ejecución del código continuara luego de presionar la tecla "**enter**".

tr –d "\012" > /dev/tty: En esta línea el comando "**tr**" combinado con el parámetro "**-d**" **(tr –d)** se encargara de borrar la información correspondiente a la representación hexadecimal, la cual corresponde a la tecla "**enter**". Seguidamente se guardara en el **bufer /dev/tty** solamente la información referente a la tecla oprimida (**s/n**).

read response: Nos permite tratar la información almacenada en la variable "**response**" mediante una serie de casos. En esta rutina se han definido dos.

- **Caso 1**: **texbfn***) ok=false: Si la información almacenada en la variable "response" coprresponde al carácter "n", la variable "ok" será igualada a "false". El asterisco (*) que se encuentra luego de "n" nos permite colocar cualquier otro carácter después de dicha tecla y obtener el mismo resultado, es decir n*) se refiere al caso de "n" seguido de cualquier otro carácter o caracteres.

- **Caso 2**: ok=true : El asterisco indica cualquier diferencia a "n" producirá ok=true.

esac: Indica el fin de "case".

done </dev/tty: Indica el fin del segundo "do".

El código continúa con las siguientes operaciones:

```
Cdp='bc -1 <<END
$cdp + $dcdp
END
```

El segmento de código anterior redefine la variable "**cdp**" igual a su valor actual mas el valor de la variable "**dcdp**", esto se hace para que en el siguiente ciclo el archivo de entrada sea el siguiente **CMP Gather** ajustado, y de esta manera realizar el análisis descrito para el siguiente **CMP**, luego de esto se volverán a repetir todos los pasos explicados anteriormente hasta que la condición: **[$cdp –le $cdpmax]**, definida en el primer ciclo llegue a fallar.

Al finalizar todos los ciclos obtendremos un conjunto de archivos cuyos nombres serán **par.$cdp** donde el **$** corresponde con la posición de cada **CMP** ajustado en los parámetros de entrada. En este caso se obtuvieron los siguientes archivos.

Archivo 1: Con nombre "**par.1800**"

tnmo=2.05275,3.052

vnmo=1495,1563.002

Archivo 2: Cuyo nombre es **par.1900**

tnmo=20.2023,3.05269
vnmo=1466.19,1500.79

Archivo 3: Cuyo nombre es **par.2000**

tnmo=2.06636,3.05381
vnmo=1466.28,1495.58

Archivo 4: Cuyo nombre es **par.2100**

tnmo=2.09161,3.05166
vnmo=1458.65,1470.53

9. Generación de un archivo que contenga todos los archivos "par".

Para finalizar debemos generar un archivo que contenga todos los valores almacenados en cada uno de los archivos **par.$cdp**. De esta forma se hallara un archivo como el que fue mostrado al inicio de esta explicación, para logar esto se ha usado el siguiente código.

Generacion de un archivo que contenga todos los archivos "par".

```
>$vpicks

echo "cdp=" | tr –d "\012" >> $vpicks

cdp=$cdpmin

echo "$cdp" | tr –d "\012" >> $vpicks

cdp=`bc -l <<END

$cdp + $dcdp
end`

while [ $cdp -le $cdpmax]

done
echo "$cdp" | tr –d "\012" >>

$vpicks

cdp=`bc –l' <<END
```

```
$cdp + $dcdp
end‘

done

echo >>$cdpmin

while [ $cdp – le $cdpmax ]

do

cat par.$cdp >>$vpicks

cdp=‘bc –l <<END
end‘

done

echo “sunmo par file : $vpicks is ready”

cdp=$cdpmin

while [ $cdp –le $cdpmax ]

do

# rm mpicks.$cdp par.$cdp

cdp=‘bc –l <<END

        $cdp + $dcdp
end‘

done
```

 # rm unisam.p

Este código retornara un archivo con el nombre especificado en el parámetro **vpicks** definido en los parámetros iníciales, para este caso es **vpicks=stkvel.p1**.

El archivo obtenido es el siguiente

Stkvel.p1:

cdp=1800,1900, 2000, 2100
tnmo=2.05275, 3.052
vnmo=1495,1563.02
tnmo=2.05023,3.05269
vnmo=1466.19,1500.79
tnmo=2.06636 ,3.05381
vnmo=1466.28,1495.58
tnmo=2.09161, 3.05166
vnmo=1458.65,1470.53

El Código que se muestra a continuación, contiene todos los pasos ya explicados.

Codigo 6.2 Análisis de velocidad para una línea sísmica con formato Seismic Unix.

```
#! /bin /sh

## Análisis de velocidad

## Paso (1) "Ajuste de los parámetros" (sección 2)

## total de la traza sísmica ordenada por CMP

velpanel=CMP

## Nombre del archivo en donde serán almacenados los valores ($t_{NMO}, v_{nmo}$)
##resultantes

vpicks=stkvel.p1

## Selección de los CMP Gathers que serán analizados

cdpmin=1800    # Primer CMP Gather
cdpmax=1200   # Último CMP analizado
dcdp=100        # Intervalo entre cada una de los CMP analizados

## Selección del rango de velocidades que vamos a utilizar

nv=500    # Numero de velocidades
dv=10     # Intervalo entre las velocidades
fv=500    # Primera velocidad
```

```
## Selección de los valores del filtro pasa banda

f= 1 , 10 , 100 , 120  - Frecuencias
amps= 0 , 1 , 1 , 0    - Amplitudes

## Parámetros correspondientes al comando UNISAM empleado para graficar la función de
## velocidad de apilamiento calculada

nout=3001 – \' Numero de valores en al archivo de salida (Archivo en formato binario)
dxout=0.004 – intervalo de muestreo en x del archivo de salida
ND=40

## Segundo paso. Definición del clico

cdp=$cdpmin
while [ $cdp – le $cdpmax ]
do
        ok=false

    while [ $ok = false]

    do

## Tercer paso. Extracción del CMP Gather

suwind < $velpanel key=cdp min=$cdp max=$cdp count=$fold > panel.$cdp

## Graficacion de los CMP Gathers extraidos

suxwigb < panel.$cdp title="CMP Gather para cdp=$cdp" Xbox=700 clip=0.6
\mpicks=mpicks.$cdp &

## Paso 4. Aplicación de la ganancia y filtrado del CMP Gather

sugain < panel.$cdp tpow=2 | sufilter f=$f amps=$amps > panelF.$cdp

## Paso 5. Análisis de semblanza

suvelan < panelF.$cdp nv=$nv dv=$dv fv=$fv > semblanza

## Grafica del mapa de semblanza

suximage < semblanza wclip=0.6 bclip=1 f2=$fv d2=$dv \units="semblanza" \label1="Tiempo
(seg)" label2="Velocidad (m/seg)" \mpicks=mpics.$cdp \

## Grafica de los CMP Gather extraídos (formato PostScript)
```

```
supsimage < panel.$cdp clip=0.25 \title="CMP Gather para CMP=$CDP" \windowtitle="CMP" \label1="Tiempo (seg)" label2="Distancia (m)" > cmp1Ima.ps

## Grafica del mapa de semblanza (en formato PostScript)

supsimahe  < semblanza wclip=0.6 bclip=1 f2=$fv d2=$dv \units="semblanza" \label1="Tiempo (seg)" label2="Velocidad (m/seg)" \title="Analisis de semblanza" > semblanza.ps

## Paso 6. Obtención del archivo .par

sort < mpicks.$cdp –n | mkparfile string1=tnmo string2=vnmo > par.$cdp

## Paso 7. Grafica de la función de la velocidad del apilamiento

sed < par.$cdp ' s / tnmo / xin / s / vnmo /yin / ' > unisam.p

unisam nout=$nout fxoput=0.0 dxout=$dxout \par=unisam.p method=spline > varias.u

xgraph < varias.u n=$nout nplot=1 d1=$dxout f1=0.0 \label1="Tiempo (seg)" label2="Velocidad (m/seg) \title="Funcion de velocidad de apilamiento: CMP $cdp" \grid1=solid grid2=solid \linecolor=2 style=seismic &

psgraph < varias.u n=$nout nplot=1 d1=$dxout f1=0.0 \label1="Tiempo (seg)" label2="Velocidad (m/seg)" \title="Funcion de velocidad de apilamiento : CMP $CDP" \grid1=solid grid2=solid \linecolor=2 style=Seismic > función.ps

## Análisis de la grafica obtenida y del "cdp"

pause

echo "La selección es correcta (s/n)" | tr –d "\012" > / dev/ tty

read response

case $response in

n*) ok=false ;;

*) ok=true ;;

esac

done < /dev /tty

cdp='bc -1 <<END

$cdp + $DCDP
```

```
END‘

done

set +x

## Paso 9. Generación de un solo archivo a partir de todos los archives par.$CDP

>$vpicks

echo “cdp=” | tr –d “\012” >> $vpicks

cdp=$cdpmin

echo “$cdp” | tr –d “\012” >> $vpicks

cdp=‘bc -l <<END

$cdp + $dcdp

END‘

while [ $cdp -le $cdpmax]

do

echo “$cdp” | tr –d “\012” >> $vpicks

cdp=‘bc –l’ <<END

$cdp + $dcdp

END‘

done
echo >> $vpicks

cdp  >> $cdpmin

while [ $cdp – le $cdpmax ]

do

cat par.$cdp >>$vpicks

cdp=‘bc –l <<END

END‘
```

```
done

echo "sunmo par file : $vpicks is ready"

cdp=$cdpmin

while [ $cdp –le $cdpmax ]

do

# rm mpicks.$cdp par.$cdp

cdp='bc –l <<END

$cdp + $dcdp

END'

done

# rm unisam.p
```

6.3 Corrección NMO de la línea sísmica.

Con el archivo "**stkvel.p1**" obtenido a partir del análisis de velocidad podemos realizar la corrección por **NMO** para los datos ordenados por **CMP** que corresponden al archivo "**CMP**".

Este procedimiento es igual al empleado en la corrección para un solo **CMP Gather**, a continuación se muestra el Código que realiza esta corrección, también se muestra la grafica de los **CMP Gathers** corregidos.

Código 6.4: Corrección por NMO de los CMP Gathers

```
# Corrección por NMO

sunmo < CMP par=stkvel.p1 > nmodata

suwind < nmodata key=cdp min=2000 max=2050 | suspswigb clip=0.2 \title="CMP Gathers corregidos por NMO" \windowtitle="NMO" \label1="Tiempo (seg)" label2="Distancia (m)" \ > NMO.ps
```

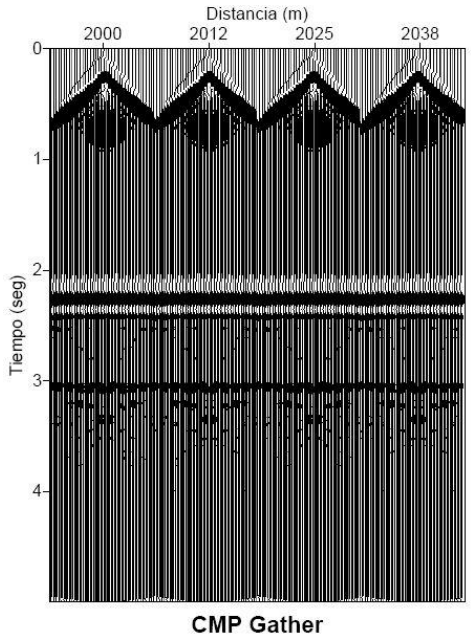


Figura 6.6 CMP Gathers después de la corrección por NMO.

6.4 Apilamiento de la línea sísmica

Después de haber eliminado el efecto del offset en los **CMP Gathers** podemos realizar el apilamiento de la línea de forma similar de cómo fue explicado en la sección 5.4. El Código 6.5 realiza el apilamiento y grafica la sección resultante, la cual corresponde a la figura 6.7, en esta figura también se muestra otra manera de cómo podemos visualizar la sección apilada, estas graficas fueron hechas mediante el comando **SUPSWIGB**.

Codigo 6.5: Apilamiento de los CMP Gathers corregidos que se muestran en la figura 6.6

```
## Apilamiento de la línea sísmica

sustack < nmodata > stackdata

supsimage < stackdata title="Seccion apilada" clip=0.2 \windowtitle="Stack" \label1="Tiempo (seg)" label2="Distancia (m) > stack.ps
```

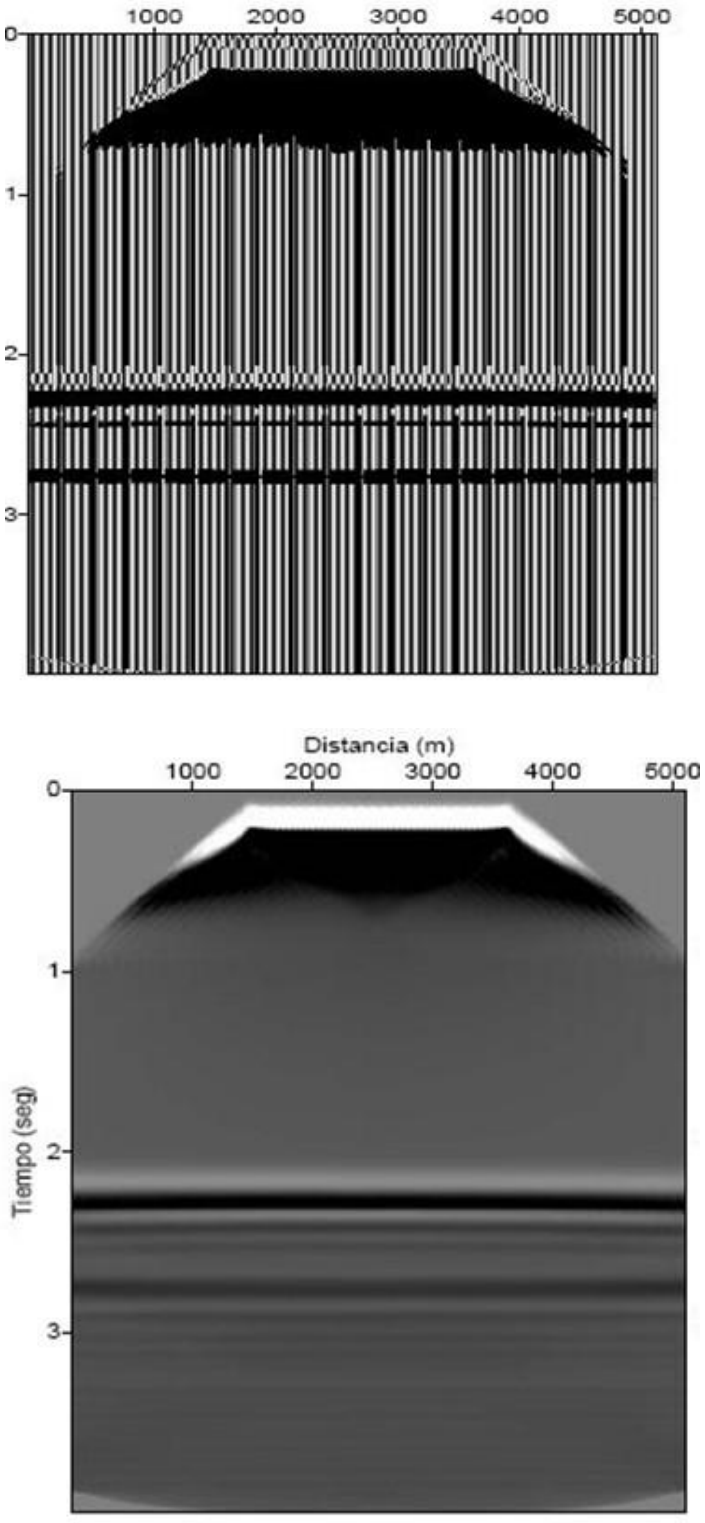


Figura 6.7. Dos formas de visualizar la sección apilada para un modelo por dos capas horizontales.

CAPITULO VII

7. Algunos modelos obtenidos con Seismic Unix

7.1 Modelo de una falla simple

7.2 Modelo de interfaces inclinadas formando un valle

Capitulo 7

Algunos modelos con su secuencia de proceso

En el apéndice D se encuentra un Script que es capaz de modelar una adquisición sísmica y generar la sección apilada partiendo de algunas propiedades del subsuelo que serán definidas, como lo fueron para el primer modelo. Este Script está basado en todas las características especificadas a lo largo de este trabajo.

El modelo tiene que estar formado por una serie de capas con velocidades constantes y se tiene que usar la geometría de adquisición como se uso en el primer modelo modificarla a sus necesidades.

Este Script está conformado por una gran parte de los Scripts que fueron mostrados y explicados durante todos los capítulos de este trabajo, y al igual que muchos de ellos, esta diseñado para usuarios con pocos conocimientos en SU como es mi caso.

Para esto solo se tienen que definir parámetros de conocimiento general como lo son: Longitud y Profundidad modelo de partida, las velocidades y densidades de cada capa, ubicación de las interfaces (el cual se hace a través de un archivo ASCII) y los parámetros de adquisición: offset máximo, intervalo entre grupos de receptores, intervalo entre disparos y tiempo de grabación.

Utilizando el Script citado anteriormente en este capítulo vamos a realizar algunos modelos con su secuencia de procesamiento, hasta llegar a la sección apilada.

En este capítulo son empleados los mismos valores mostrados durante el capítulo cuatro relacionados con los parámetros de adquisición y las propiedades del modelo de partida. Así mismo, son utilizados los mismos valores de los parámetros relacionados con el análisis de velocidad mostrado en la sección 6.2, los únicos cambios son en la configuración de las interfaces y en las longitudes del modelo, los cuales serán especificadas más adelante.

El siguiente paso será realizar un modelo parecido al que se realizo durante los capítulos anteriores, con unas pequeñas variaciones, como una interfaz inclinada, las densidades diferentes y velocidades, pero no tiene mucha importancia esto, ya que con solo cambiar el archivo de entrada , el cual tiene los valores de las interfaces, las cuales difieren muy poco de los valores del modelo que fueron utilizados en el trabajo, y el resultado será prácticamente el mismo es por esta razón que no tiene caso realizar este modelo, para ello se realizara un modelo un poco más complejo en donde se cambiaran varios parámetros y sobre todo las interfaces son un poco más complejas, estas interfaces están dispuestas de tal manera que forman un valle.

7.1. Modelo 2. Modelo de una falla simple.

Para este modelo se usara el Script 8 del apéndice D es el mismo que fue usado en el modelo anterior, como se hizo en el modelo anterior se cambiara el modelo estructural de entrada, en otras palabras lo que se variaron fueron las interfaces, el siguiente archivo muestra como se variaran las interfaces.

"**Modelo_Falla**"

0	**0**
5000	**0**
1	**-99999**
0	**1500**
2000	**1500**
3000	**2500**
5000	**2500**
1	**-99999**
0	**4025**
5000	**4025**

En esta tabla se muestran los puntos que definen al modelo de capas, el cual consta de dos capas con la forma de una falla simple.

En las siguientes figuras se muestran los modelos de velocidad y densidad.

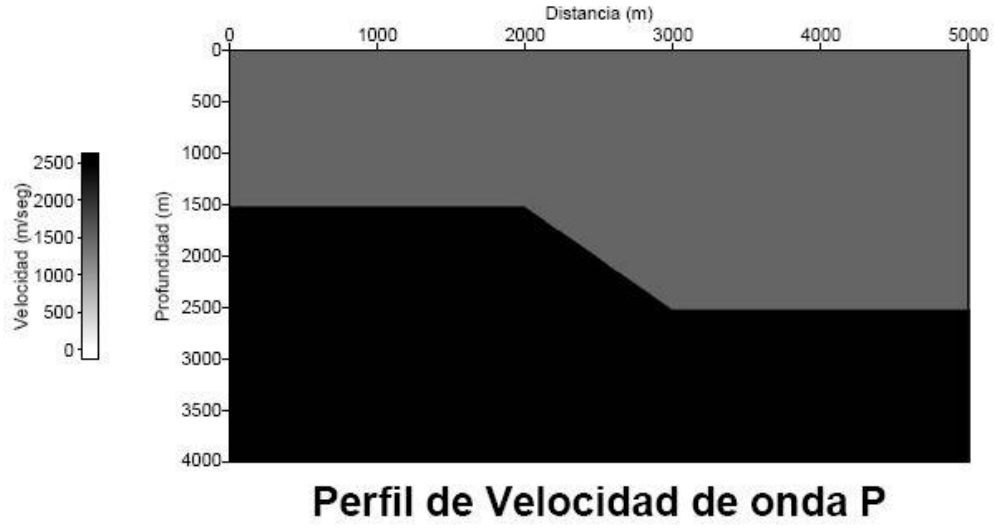


Figura 7.1: Perfil de velocidades del modelo "Falla_Simple"

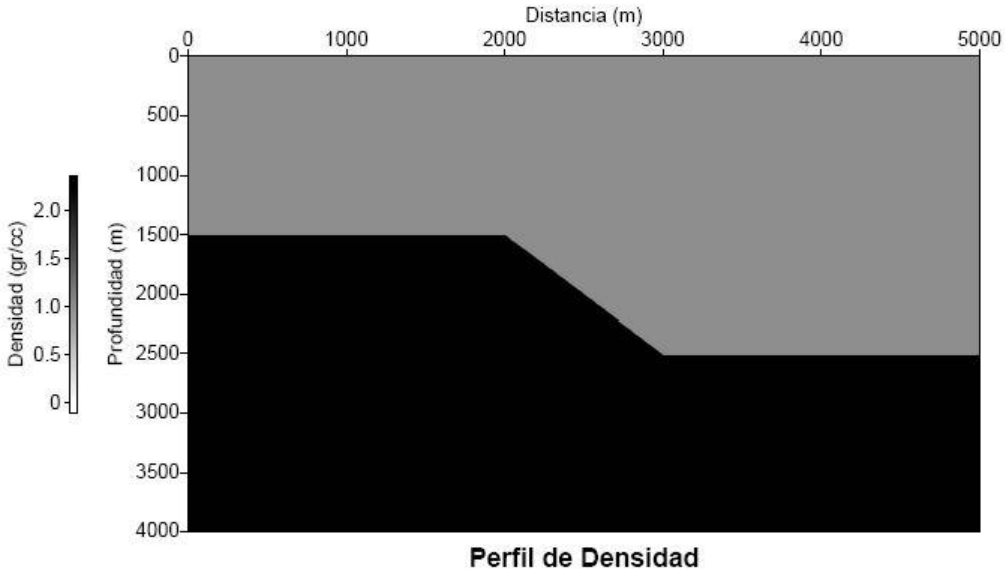
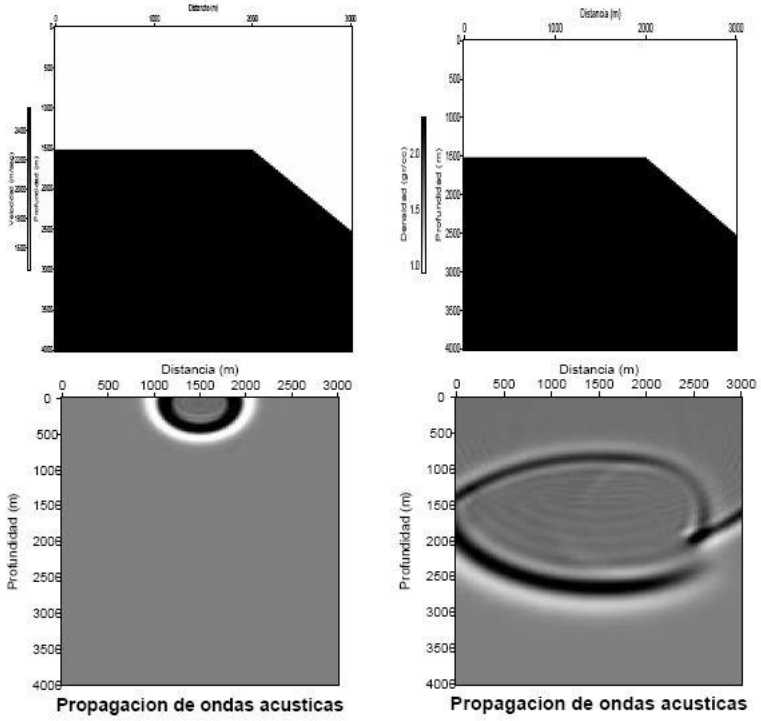


Figura 7.2: Perfil de densidades del modelo "Falla_Simple".

Para el primer disparo se obtuvieron las siguientes figuras.



7.3: Resultados obtenidos para el primer disparo, en la parte superior se muestran los perfiles de velocidad (izquierda) y densidad (derecha), abajo se muestra la propagación de ondas.

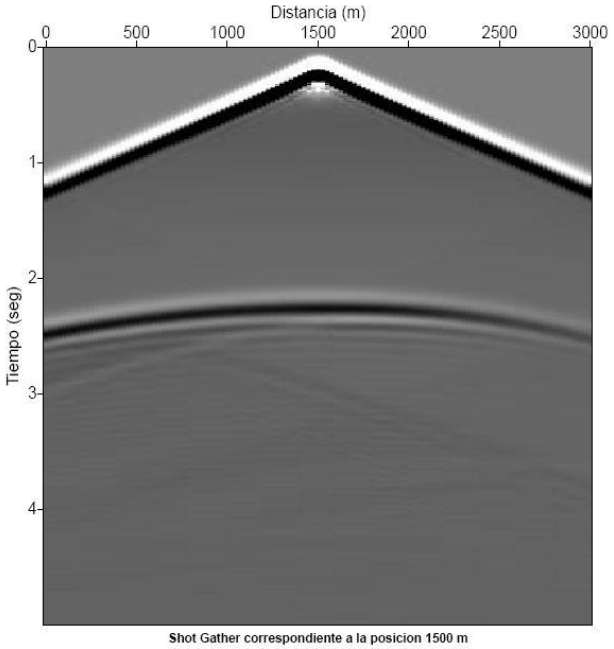


Figura 7.4: Shot Gather correspondiente al primer disparo.

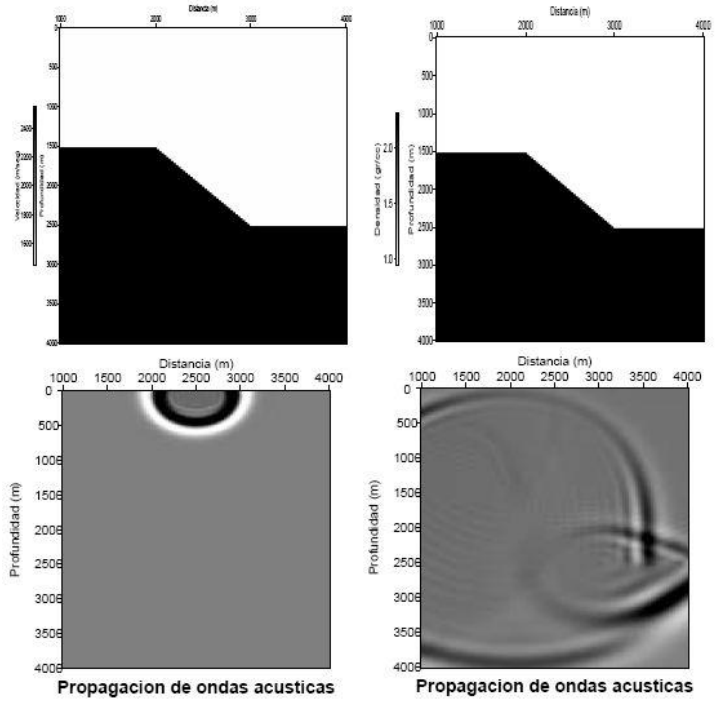


Figura 7.5: Resultados obtenidos para el disparo 21, en la parte superior se muestran los perfiles de velocidad (izquierda) y densidad (derecha) , en la parte de abajo se muestran los perfiles de propagación de ondas.

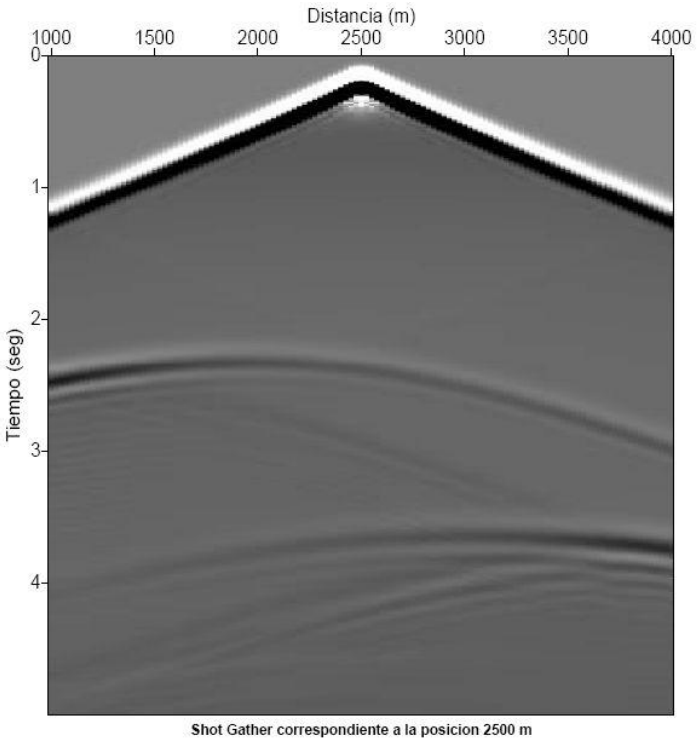


Figura 7.6: Shot Gather correspondiente al disparo 21 en 2500 metros.

En la siguiente grafica se muestran los cuatro Shots Gathers obtenidos.

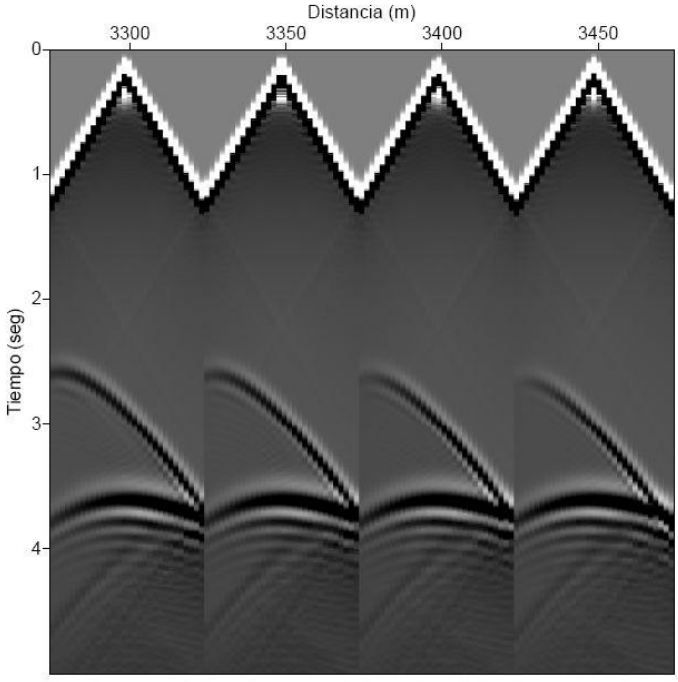


Figura 7.7: En esta figura se muestran los cuatro Shot Gathers obtenidos.

En la figura 7.8 se muestran los cuatro CMP Gathers, los cuales corresponden a la posición 2000, 2012, 2025,2038 metros.

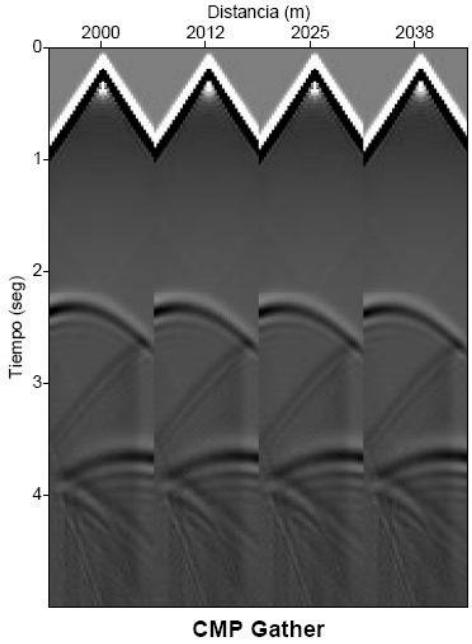


Figura 7.8: CMP Gathers correspondientes a las posiciones 2000, 2012, 2025,2038 metros.

Después de realizar un análisis de semblanza, de igual manera que en los dos modelos anteriores, se obtuvo una función de velocidad, la cual nos permitió realizar la corrección NMO, los resultados se muestran en la figura 7.9.

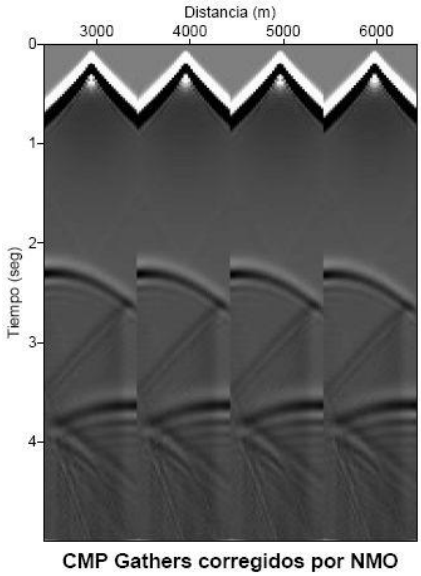


Figura 7.9: CMP Gathers de la figura 7.8 los cuales fueron corregidos por NMO.

Después de realizar el apilamiento de la línea sísmica, se obtiene la siguiente figura.

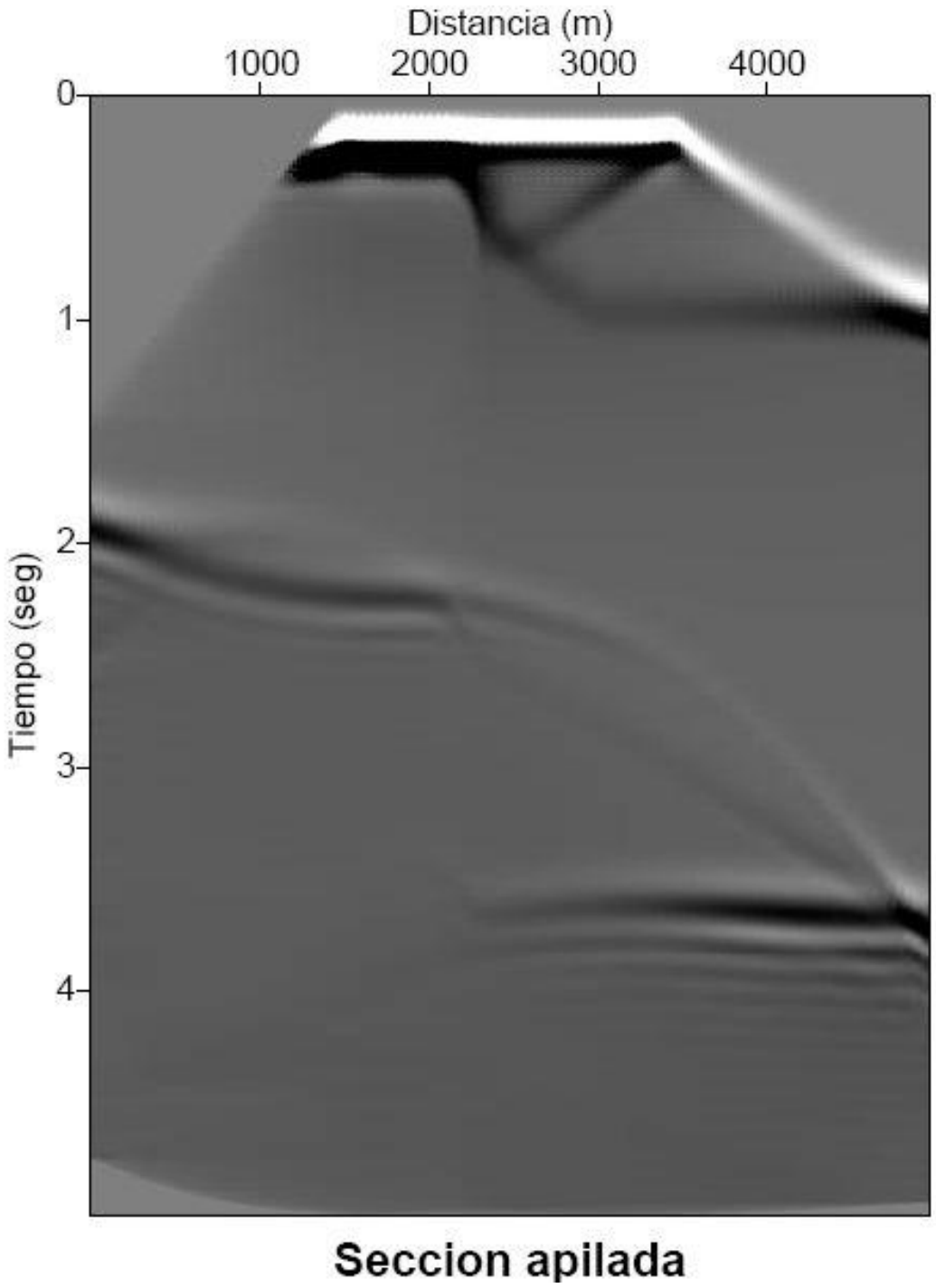


Figura 7.10. Seccion apilada obtenida del modelo "Falla_Simple"

7.2 Modelo 1. Interfaces formando un Valle

En este modelo es un poco más complejo. En este caso construiremos una representación del subsuelo formada por 2 capas inclinadas que integran una especie de valle.
Para lograr esto solo fue necesario cambiar 2 de los parámetros del Script 8: El archivo de entrada "Archivo Entrada" y la profundidad del modelo. Ambos parámetros están relacionados con la construcción del modelo del subsuelo inicial. Todos los demás valores se conservaron. En este caso, la profundidad del modelo fue de 4000 metros:
Profundidad=4000

El archivo de entrada empleado es el siguiente.

"Interfaces_Inclinadas"

0	**0**
5000	**0**
1	**-99999**
0	**1500**
1500	**2500**
3500	**2500**
5000	**1500**
1	**-99999**
0	**4025**
5000	**4025**

En esta tabla se muestran los puntos que definen al modelo de capas, el cual consta de dos capas con la forma un valle.

A través de estos dos pequeños cambios (profundidad y el modelo de las interfaces) en el script 8 se obtuvieron las siguientes graficas para este modelo.

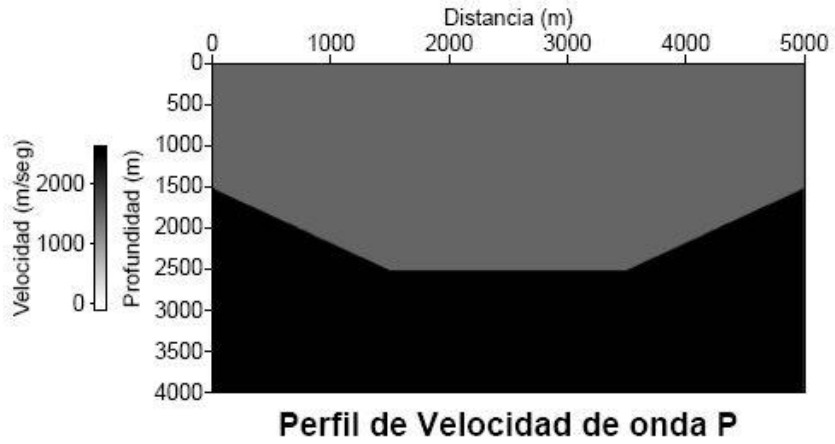


Figura 7.11: Perfil de velocidad del modelo de interfaces inclinadas.

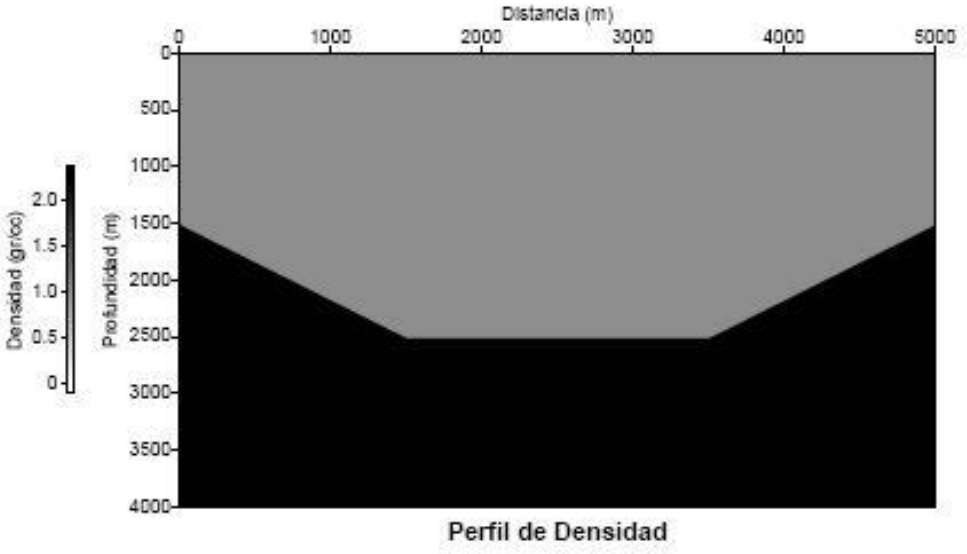


Figura 7.12: Perfil de densidad del modelo de interfaces inclinadas.

Para el primer disparo se obtuvieron los siguientes resultados.

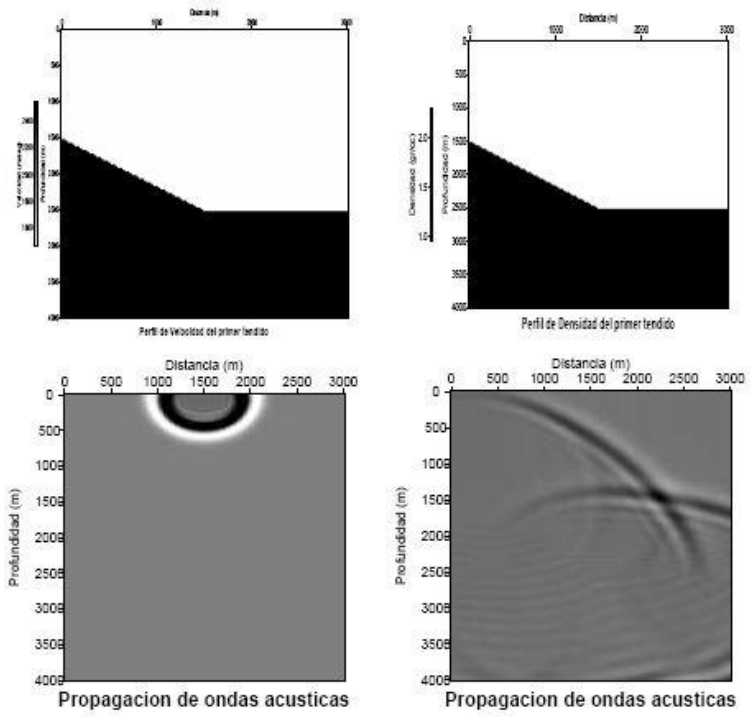


Figura 7.13: Estas graficas corresponden el primer disparo, en la parte superior se muestra los perfiles de velocidad (derecha) y densidad (izquierda), en la parte de debajo de muestra la propagación de las ondas acústicas.

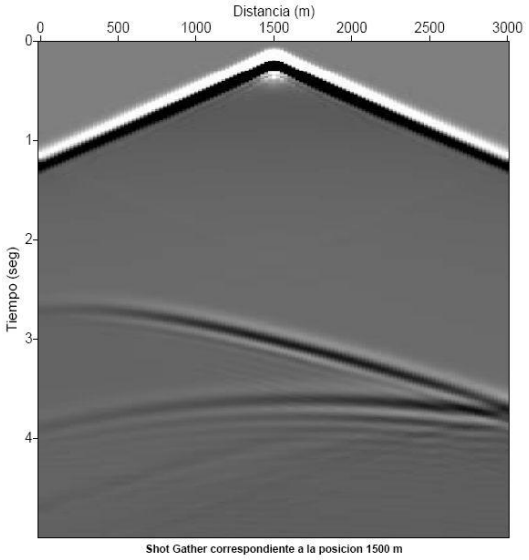


Figura 7.14: Shot Gather correspondiente al primer disparo.

Para el disparo numero 21 de la adquisición, el cual corresponde a la posición de 2500 metros se obtuvieron las siguientes graficas.

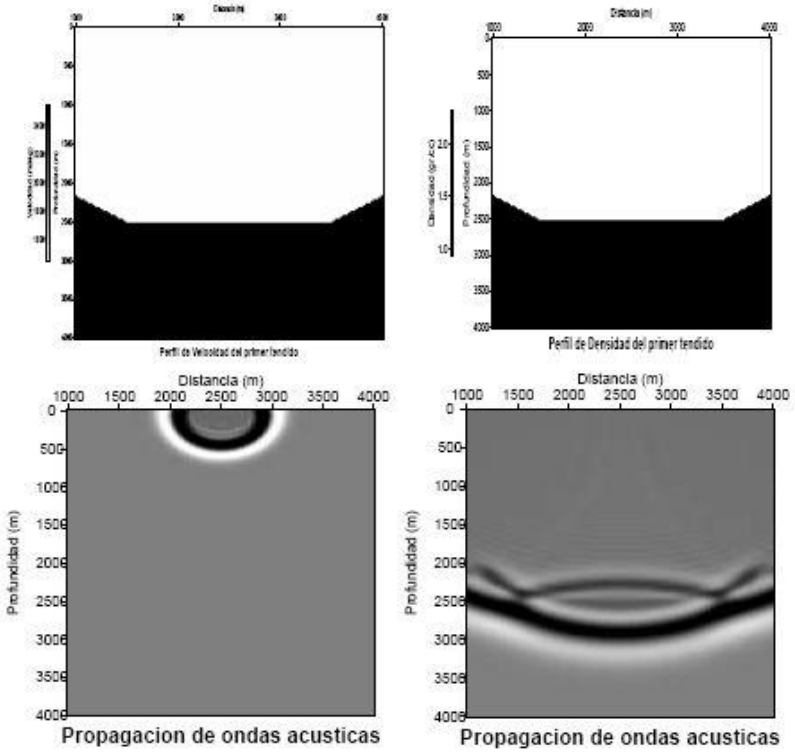


Figura 7.15: Graficas obtenidas para el disparo 21 de la adquisición que está a 2500 metros, en la parte superior de muestran las perfiles de velocidad y densidad, mientras que en la parte de abajo se muestran los perfiles de la propagación de las ondas acústicas.

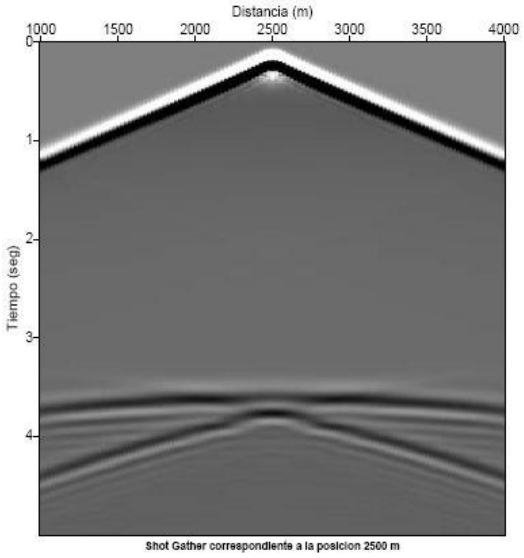


Figura 7.16: Shot Gather correspondiente al disparo 21.

En la siguiente figura de observan los cuatro Shot Gathers obtenidos, un poco más abajo se muestran cuatro CMP Gathers con una cobertura amplia.

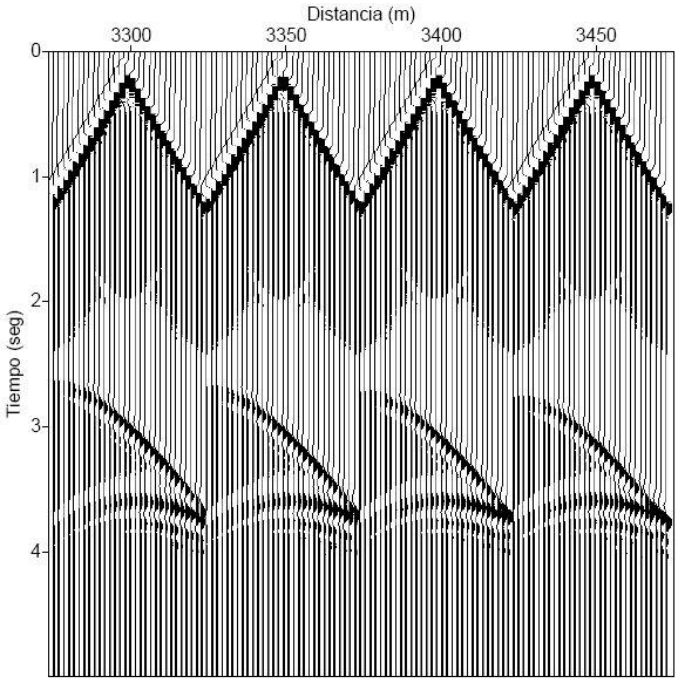


Figura 7.17: Grafica que muestra cuatro Shot Gather del modelo.

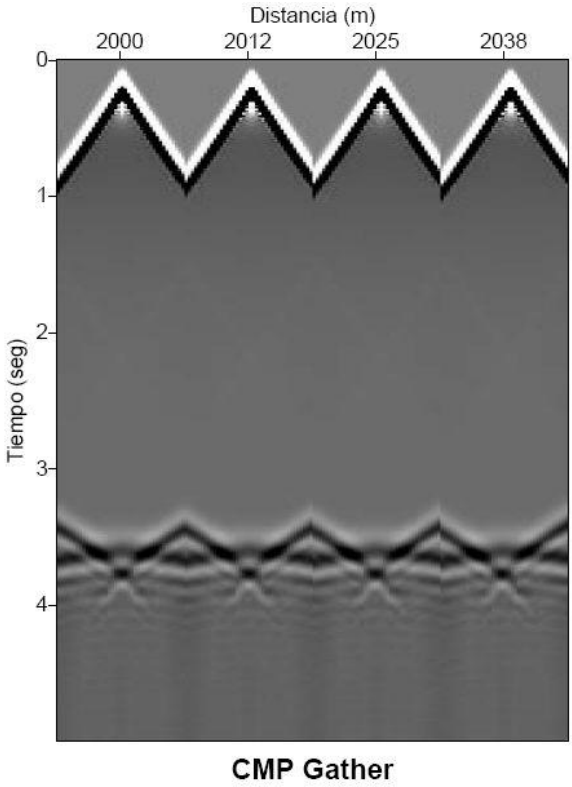


Figura 7.18: CMP Gather correspondiente a las posiciones 2000,2012.2025 y 2038 metros.

Al igual que en el primer modelo, se utilizaron cuatro CMP Gathers distribuidos a lo largo del total de la línea sísmica , para realizar el análisis de velocidad, se usaron los siguientes CMP=1800,1900,2000,2100 metros.
En la siguiente figura se muestra el análisis de semblanza realizado al CMP 2100.

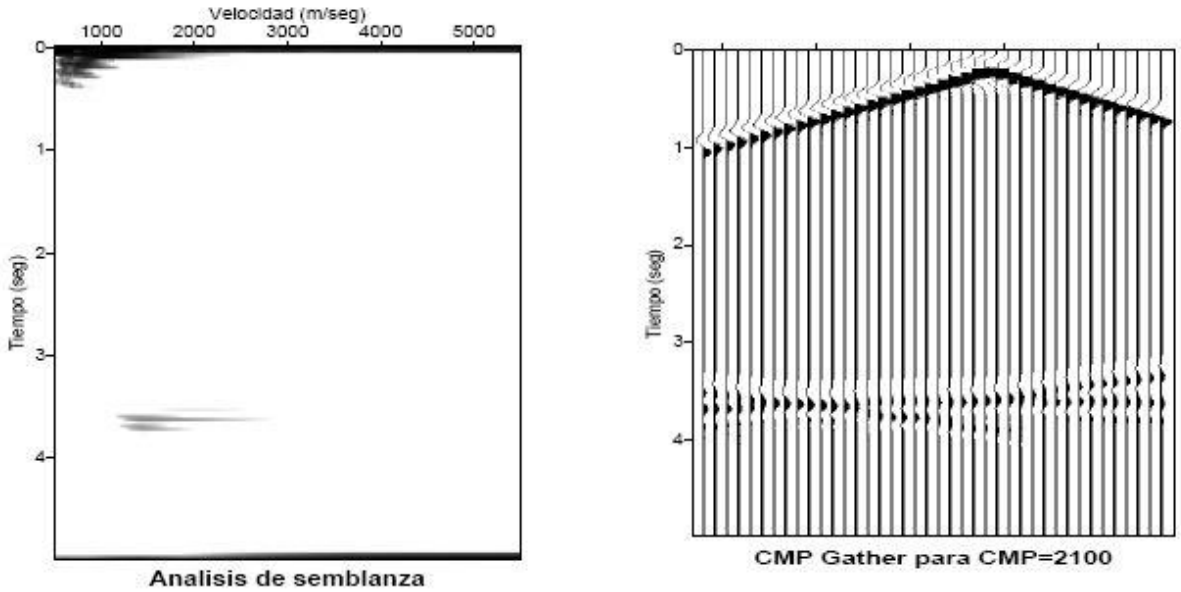


Figura 7.19: análisis de semblanza para el CMP Gather de 2100 metros.

A través de la función de velocidad obtenida mediante el análisis de semblanza se pudo realizar la corrección NMO, la cual se muestra a continuación.

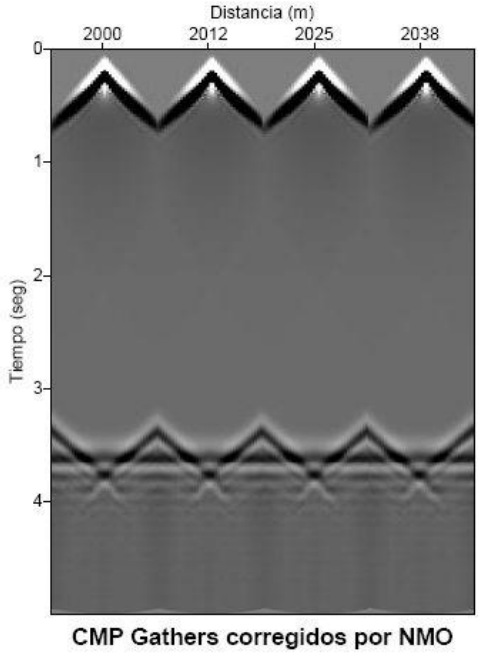


Figura 7.20: CMP Gathers de la figura 2.1.9 corregidos por NMO.

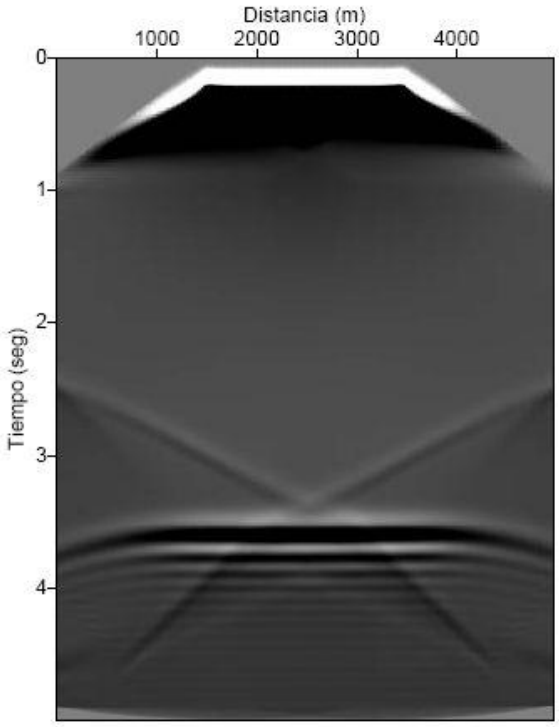


Figura 7.21: En esta figura se muestra la sección ya apilada.

Migración

Como paso adicional podemos migrar los modelos sintéticos creados estos se hace a través del siguiente Script, para esto vamos a usar el comando SUMIGPS,pero antes de realizar la migración vamos a eliminar las primeras llegadas directas, de esta forma se obtuvo una sección a partir de 1.5 seg. la cual se procedió a migrar.

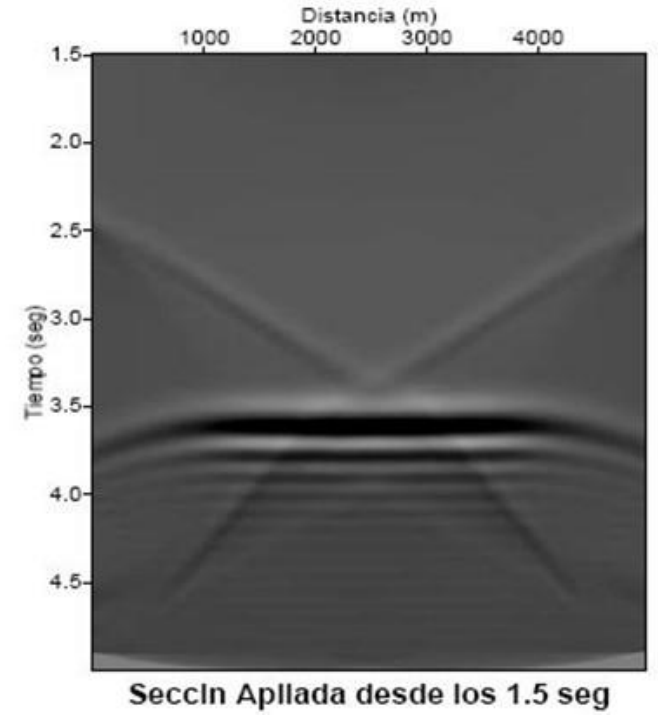
El script toma el último archivo creado, el cual corresponde a la sección apilada y a su vez a través del comando SUWIND, elimina las llegadas directas, después de eliminar las llegadas se procede a migrar por medio del comando SUMIGPS.

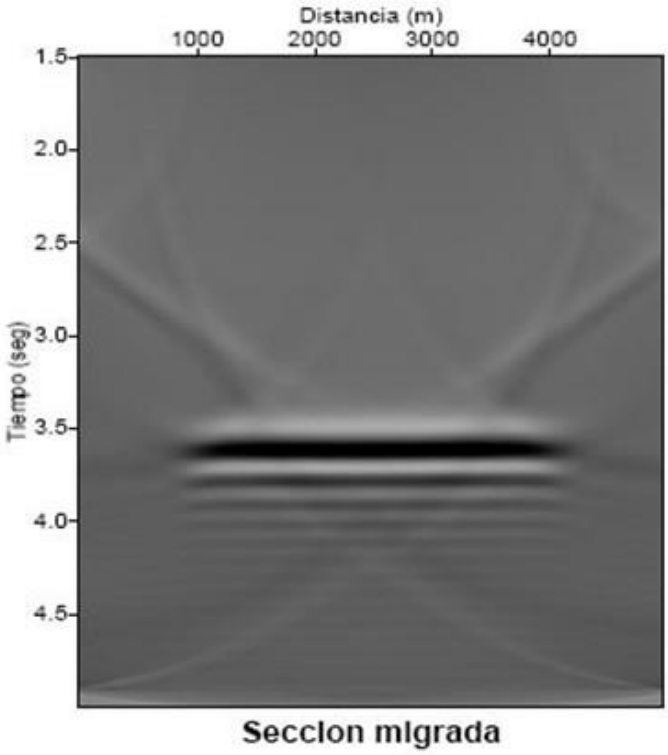
Script para el cual se hace la migración del modelo 2 (Valle).

```
"!  /bin/sh

## Eliminación de las llegadas directas

suwind < stack1_5 tmig=1.5 > stack1_5

## Migración de cambio de fase

sumigps > stack1_5 tmig=3.625 vmig=2500 > stack_migrada1_5

## Graficación de "stack1_5"

supsimage < stack1_5 title="Seccion apilada desde 1.5 seg" clip=0.05 \label1='Tiempo (seg)' label2='Distancia (m)' > stack1_5.ps

## Graficación de la sección migrada

supsimage < stack_migrada1_5 title="Seccion migrada"  clip=0.05 \label1='Tiempo (seg)' label2='Distancia (m)' > stack_migrada1_5.ps
```

En esta parte no se realizo el análisis de velocidades para la migración, las velocidades usadas se obtuvieron observando el modelo de partida y a través de ensayo y error, la velocidad optima de migración encontrada, es la velocidad de la capa que se encuentra por debajo del evento.

En las dos figuras mostradas, podemos observar que en la primer figura se muestra la sección apilada, mientras que en la otra se muestra la sección migrada.





7.22: comparación entre la sección apilada y la sección migrada del modelo 1 (Interfaces formando un Valle).

CAPITULO VIII

8.- Conclusiones y recomendaciones

Capitulo 8

Conclusiones y recomendaciones.

Conclusiones.

- Los programas del paquete Seismic Unix que fueron utilizados a lo largo de este trabajo constituyen una herramienta de gran utilidad en el modelado sísmico, siguiendo esta idea por medio de estos programas podemos obtener representaciones del subsuelo o perfiles de propiedades, todo esto lo podemos hacer con solo definir un par de parámetros como son los parámetros de velocidad y densidad, para posteriormente hacer el modelado de la propagación de ondas acústicas y hacer su análisis correspondiente.

- Con estos mismos programas es posible realizar una secuencia estándar de procesamiento, que fue lo que se hizo a lo largo de este trabajo, se pudo apreciar que las secciones obtenidas a partir de la secuencia de procesamiento aplicada a los datos sintéticos presentan una gran similitud con modelos de partida. Esto nos indica que los procedimientos empleados fueron correctos y nos dan una idea básica de cómo es el procesamiento de datos sísmicos con Seismic Unix.

- En este trabajo se explican se manera básica las herramientas que posee Seismic Unix para el moldeamiento y procesamiento de datos sísmicos, las cuales pueden ser empleadas para el desarrollo de proyectos de una mayor complejidad, como puede ser el procesamiento de datos reales o el modelado de secciones más complejas.

- El comando SUFDMOD2 realiza un modelado en donde no pueden ser eliminadas las reflexiones (Primeras llegadas), pero existe un parámetro que genera una disminución considerable de estos efectos, ya que estas llegadas producen problemas considerables en los datos generados, dichos efectos se traducen en patrones de reflexiones que generan eventos ficticios, los cuales, si no supiéramos las características del modelo de partida nos llevaría a interpretaciones erróneas, es por tal motivo que debemos saber diferenciar las primeras llegadas en las trazas sísmicas.

- La estabilidad que caracteriza al modelado de la propagación de ondas acústicas a través del método de diferencias finitas, nos obliga a establecer ciertos criterios a la hora de definir algunos parámetros involucrados, siguiendo esta idea algunos de los parámetros deben de guardar cierta relación entre ellos, estos son: las dimensiones totales del modelo, el intervalo de muestreo y la frecuencia máxima que caracterizará la propagación de ondas acústicas.

- Cabe mencionar que Seismic Unix es un paquete muy poco conocido, pero esto no le resta importancia, ya es un paquete que tiene grandes alcances en la interpretación de datos sísmicos si estos son aplicados correctamente, un punto en contra es que la manipulación de los datos y de todo el procedimiento se hace vía comando, es decir carece de interfaz grafica (GUI) al principio es un poco complicado, porque hoy en día es raro manejar estos tipos de programas sin interfaz grafica, pero con un poco de práctica se ve que no es tan complicado, el único punto que veo tiene un gran inconveniente es la instalación del programa, ya que ésta es un poco compleja si nunca antes se ha manejado alguna distribución de Linux, pero al final de este trabajo se muestra como hacer esta instalación paso a paso, además de que se anexan 2 discos uno con las utilidades y otro con el sistema listo para instalarse en cualquier maquina, no será necesario instalar ningún componente adicional (software).

- Por último se han hecho programas que se asemejan mucho a lo que hace Seismic Unix, uno de ellos Visual SUNT el cual puede descargarse de la página http://www.wgeosoft.ch/Software/default2.html el cual tiene un precio desde $1200 dll. Hasta los $2100 .dll por licencia, la única diferencia entre este y Seismic Unix, es que Visual SUNT posé una interfaz grafica (GUI).

- Mi contribución no es muy grande, ya que en el presente trabajo no se programo ninguna nueva función, solo si hicieron los códigos ó rutinas para el procesamientos de datos sísmicos de los modelos propuestos, mi aportación mas grande es que dejo una referencia bibliográfica de cómo usar Seismnic Unix, desde su instalación hasta la ejecucion del mismo. Además de que se dejan una serie de Códigos con los cuales se pueden empezar a trabajar de inmediato en el modelado y procesado de datos sísmicos, por ultimo otra aportación es que se deja un DVDROOM con todo lo necesario para su instalación en cualquier maquina con procesador X86.

Recomendaciones.

- En el modelo de una falla geologica simple se puede apreciar que la sección obtenida después de la migración presenta ciertas discrepancias con el modelo del subsuelo, para mejorar estos resultados podemos probar otros algoritmos de migración o realizar una migración pre-apilamiento, esto permitirá evaluar a mayor escala las funcionalidades de migración que ofrece Seismic Unix, de igual forma es muy importante realizar un análisis de velocidad de migración, la cual no fue tratada en este trabajo.

- El Código generado (apéndice D), el cual trata de envolver la mayoría de los procedimientos explicados a lo largo de este trabajo,se hizo con el fin de que usuarios con pocos conocimientos en el manejo de SU, puedan realizar análisis y modelos sísmicos de una manera eficaz. Esta herramienta es de gran utilidad pedagógica, permitiendo a los estudiantes recrear y observar los conceptos básicos involucrados en el modelado y procesamiento de datos sísmicos.

Comentarios.

- El tiempo de ejecución de los programas jugó un papel importante en el modelado de realizado. Debido a que, obviamente, el tiempo de ejecución aumenta conforme a la cantidad de muestras utilizadas en el modelo.
  En el trabajo realizado se ajustó un número de muestras razonablemente manejable en cuanto al tiempo de ejecución. Ya con esta condición en el número de muestras y tomando en cuenta las longitudes del modelo, se deriva un intervalo de muestreo el cual está ligado con los valores de frecuencia que admite el método de diferencias finitas utilizado. Todo esto se hizo con la finalidad de que los eventos sean más fáciles de distinguir y el tiempo de generación de los modelos no sea muy largo, ya que se uso una PC personal con las siguientes características generales.

  Procesador Intel Core 2 QUAD Q6600 4x2.4 Ghz
  4 Gigas de RAM a 800 Mhz
  Tarjeta grafica GForce con memoria de 1024 MB

  En algunas ocasiones el tiempo de espera en la generación de los modelos fue de casi una hora como máximo bajo un hambiente de trabajo Linux con una Desktop de penúltima generación.

REFERENCIAS.

[1] Ozdogan Yilmaz, 2000, Seismic Data Processing.

[2] Seismic Unix, Universidad de Bergen 2006.
http://www2.geo.uib.no/seismic-unix/

[3] Seismic Unix Help.
http://sepwww.stanford.edu/oldsep/cliner/files/suhelp.

[4] "Modelado con Seismic Unix", Jos´e Antonio Lara,Madrid, España 2003.

[5] Taner, M. T. y Koehler, F., 1969, Velocity spectra digital computer derivation and applications,Geophysics.

[6] Scales John, Theory of Seismic Imaging, Samizdat Press, Colorado School of Mines,1994.

[7] Ryan J. Early ,Seismic Unix en Ubuntu. Instalación de Seismic Unix en Ubuntu 7.04., Rutgers Geophysics,2006.

[8] Cooper M., "Advanced Bash-Scripting Guide", Version 4.1, 2006.

[9] Caicedo M., Mora P., "Temas de Propagaci´on de Ondas", Universidad Caracas, 2004.

[10] Using Seismic Unix to Teach Seismic Processing,Michigan Technological University

[11] J. Stockwell, J. Cohen, The New SU Users Manual. Center for Wave Phenomena, Colorado School of Mines, V3.2, August 2002.

[12] Proyecto "Modelado de propagaci´on de ondas en medios visco-el´asticos". USB-PDVSA.

[13] Henry Thorson Consulting,Graphical User Interface Example,TKSU (Interactive front to SU seismic processing), 2002.

[14] Manual de Ubuntu 8.04. www.Ubuntu.es

[15] Laurence Lines y Rachel Newrick, SEG. Fundamentos de Interpretacion Geofisica

[16] Jonh Clearbout. Imaging the Earth Interior.

[17] http://www.cwp.mines.edu/cwpcodes/index.html

[18] Perez Monica. Utilizacion del paquete Seismci Unix para la comprension de la Sismica. USB

APENDICES

. Apéndice A.

Instalación de Seismic Unix desde cero.

Hay diferentes maneras de instalar Seismic Unix, estas son :

- Instalándolo en Cygwin.
- Instalándolo en MacosX cualquier versión.
- Instalándolo en Linux (Ubuntu) en modo nativo.
- Instalándolo en Linux (Ubuntu) desde una maquina virtual que corre bajo Windows cualquier versión de este.

Como se menciono anteriormente existen varias formas de instalar Seismic Unix, aquí solo se verán las dos últimas, ya que la instalación en Cygwin es muy complicada, además de que el sistema se vuelve inestable bajo Windows Vista, con respecto a la otra en MacosX no se hará ya que son muy pocas personas las que usan este sistema operativo de Apple, pero la instalación es casi idéntica a la de Linux, ya que MacosX corre bajo Unix.

Comenzando la instalación.

Primero veremos cómo instalar SU en una maquina virtual, la cual corre desde cualquier versión de Windows, la única diferencia que hay entre esta instalación y la del modo nativo es que primero instalaremos la maquina virtual para luego instalar Linux en su versión de Ubuntu , acto seguido instalaremos SU, para poder realizar esta instalación necesitaremos una maquina un poco potente con un procesador de 2.0 GHz o superior bastara en memoria con 1.0 GB o superior con tarjeta de gráficos 128 MB o superior estos serán los requisitos mínimos, porque se necesitan tantos recursos es porque estaremos corriendo los dos sistemas operativos simultáneamente, no es muy recomendable esta opción ya que algunas veces el sistema se vuelve inestable o consume demasiados recursos, esta opción solo es recomendable si solo queremos probar SU por unos momentos, no se recomienda para trabajar con grandes volúmenes de datos.

Primer paso.

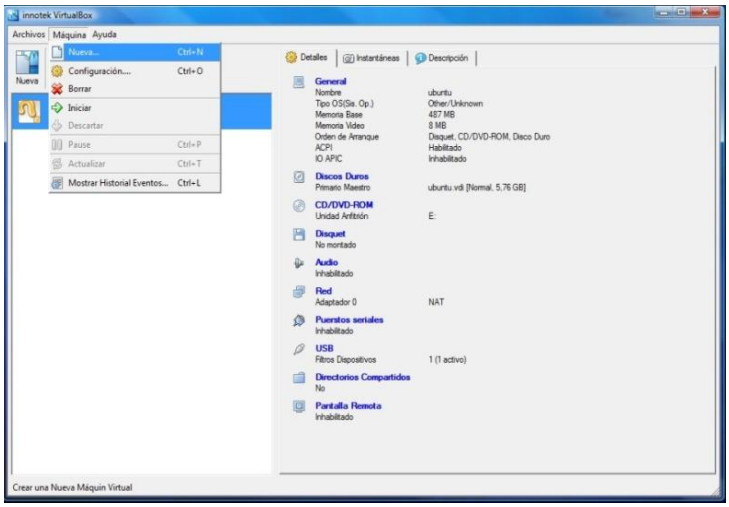
Descargaremos el siguiente programa VirtualBox de la siguiente dirección electrónica **http://www.virtualbox.org/** o lo podremos instalar desde el CD de utilidades anexado, este programa es del tipo Freeware por lo tanto no estaremos no tendremos ninguna resticcion. Cuando tengamos el programa procederemos a instalarlo de acuerdo a las instrucciones del mismo.

Dentro del programa, nos dispondremos a crear un maquina virtual de la siguiente manera.

- En el menú nos iremos a máquina, hay nos iremos nueva, hay seguiremos los instrucciones del asistente para la creación de una nueva máquina virtual.

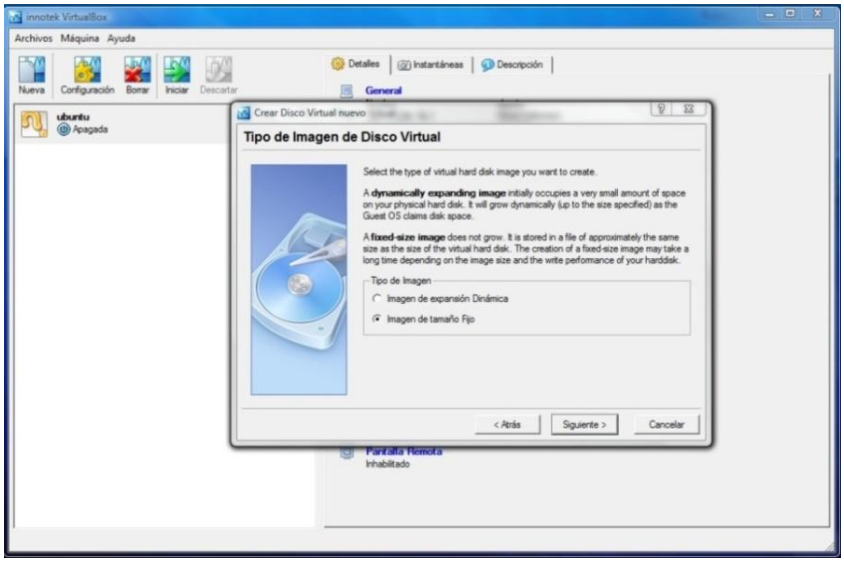
En las siguientes capturas se muestra el proceso de cómo crear una nueva máquina virtual.

Ejecutamos el programa y creamos una nueva máquina virtual. Le damos un nombre a la maquina virtual y el tipo de sistema operativo.



Acto seguido seleccionamos el tamaño de memoria se recomiendan 512 MB

Después tendremos crear un disco duro virtual, damos clic en nuevo, nos saldrá un asistente el cual nos indicara como crear el disco duro damos en siguiente y seleccionamos imagen de tamaño fijo, damos en siguiente nos pedirá el nombre del disco y tamaño se recomiendan 8 Gigas damos en siguiente y en terminar ya hemos creado la imagen.



Ya casi hemos terminado de crear la maquina virtual solo falta configurarla, para ello seleccionaremos la maquina que acabamos de crear después nos iremos a la ventana de configuración, en donde en la sección de CD/DVD-ROM montaremos la unidad, después en audio seleccionamos habilitar audio, en red la habilitamos por ultimo en USB habilitamos el controlador de USB, damos OK.

Hasta aquí solo hemos instalado la maquina virtual, falta instalar el sistema operativo y posteriormente SU.

Para instalar Ubuntu solo debemos iniciar la maquina virtual creada e introducir el CD de Ubuntu en la unidad óptica y seguir las instrucciones en la pantalla.

La instalación de Ubuntu se hace de la siguiente forma, primero iniciaremos la maquina virtual creada, introduciremos el CD en la unidad óptica, en la pantalla que se muestra cambiaremos el idioma con f2, luego le damos iniciar o instalar Ubuntu, dejaremos que cargue, ya estando el escritorio le daremos clip en el icono que dice instalar.

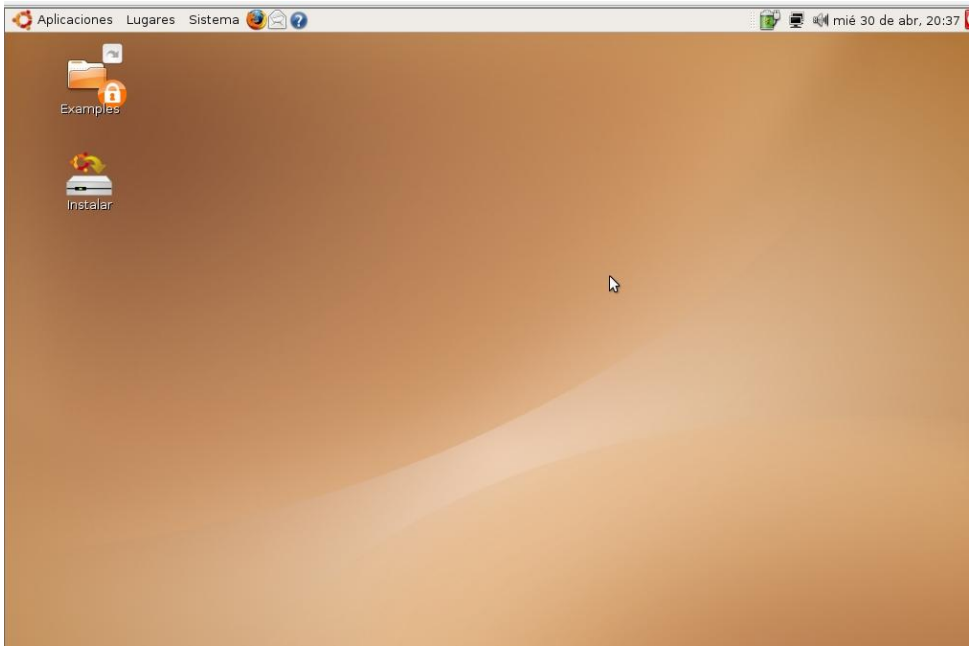
Pasos para una correcta instalación.

- Dar clip en el icono instalar.
- Seleccionar el idioma de instalación y dar clip en siguiente.
- Seleccionar la ciudad y dar clip en siguiente.
- Seleccionar la distribución del teclado en este caso es Latinoamérica dar clip en siguiente.

- Seleccionar la partición del disco en manual y dar clip en siguiente.
- Aquí crearemos dos particiones una de 600 MB para intercambio y el resto para la instalación de Ubuntu. Damos clip en New partition table, después en New partition, aquí seleccionamos el tamaño de la partición que es de 600 MB, en Use as seleccionamos swap y damos clip en aceptar.
- Para la creación de la partición en donde vamos a trabajar, seleccionamos el espacio libre (free space), damos clip en New partition y damos clip en aceptar por ultimo seleccionamos la partición que acabamos, damos clip en Edit partition y seleccionamos punto de montaje (Mount point) la primera opción debe de quedar así / damos clip en aceptar, damos clip en adelante.
- Migramos documentos y configuraciones lo más recomendable es que no migremos nada, damos clip en adelante.
- En esta ventana seleccionaremos un nombre de usuario y contraseña, damos clip en adelante.
- Si todo esta correcto daremos clip en Install, esperaremos que se complete la instalación para después reiniciar, si has seguido estos pasos correctamente habrás instalado Ubuntu sin problema.

- Para iniciar cada vez que desees Ubuntu solo bastara con seleccionarlo del menú de VirtualBox

Captura de Ubuntu ya instalado.



Instalación de Ubuntu como un segundo sistema operativo (en forma nativa).

Para poder instalar Ubuntu necesitaremos como requisitos mínimos los siguientes: procesador a 700 MHz o superior, 364 Mb en RAM o superior, 8 GB de disco duro para instalación y swap, tarjeta VGA con soporte de resolución 1024x768 o superior y tarjeta de red. Como podemos apreciar los requisitos bajan enormemente cuando instalamos el sistema en forma nativa y no virtual.

Para proseguir con la instalación debemos crear dos particiones desde Windows (cualquier versión), estas particiones pueden ser creadas con diversos programas que hay en el mercado, no citare ninguno, ni cómo hacerlo ya que en su gran mayoría cada programa tiene su forma de realizar esta tarea, las particiones que necesitaremos serán una con un punto de montaje / de al menos 8 GB de espacio y la otra del tipo swap con al menos 512 Mb de espacio.

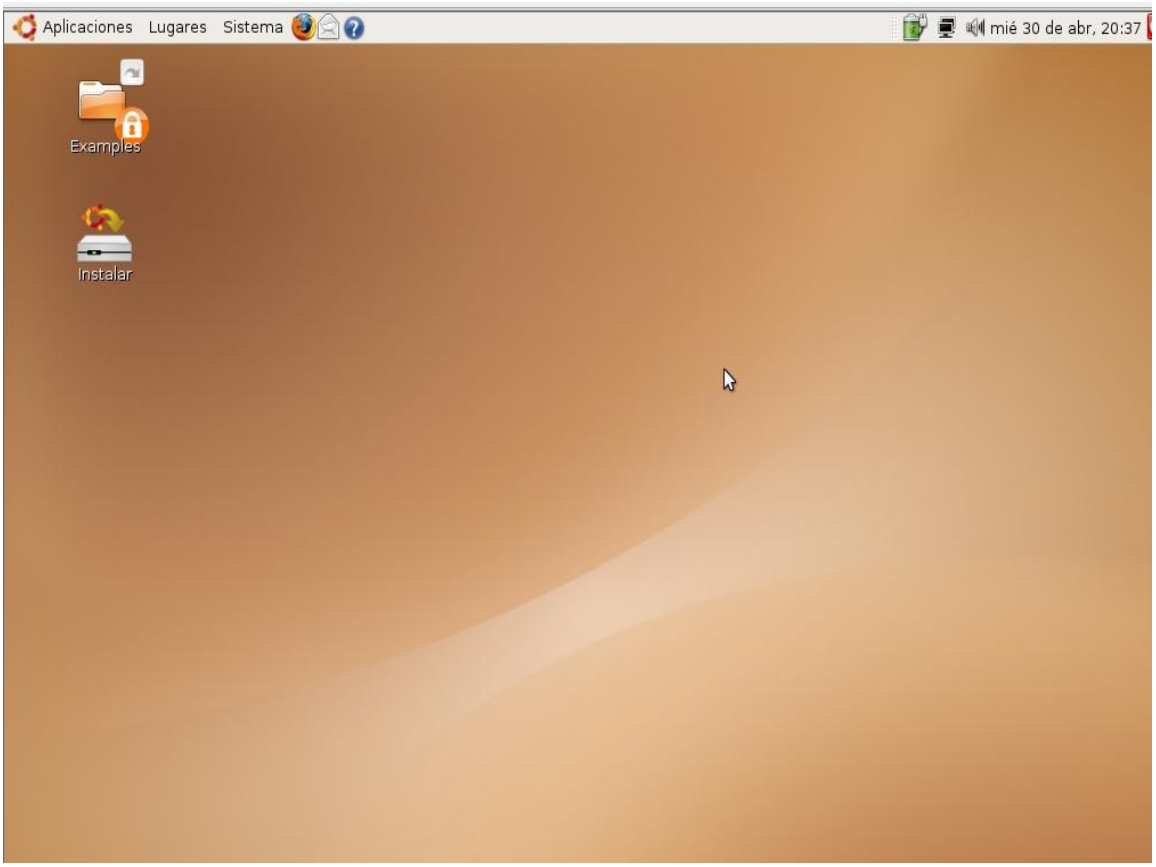
Cuando ya tengamos creadas las dos particiones necesarias y tras haber actualizado el sistema, colocaremos el CD en la unidad óptica y dispondremos a reiniciar la maquina.

Para una correcta instalación haremos lo siguiente, en la pantalla que nos muestra inmediatamente seleccionaremos el idioma oprimiendo f2 y esperaremos a que cargue completamente Ubuntu.

Pasos a seguir para instalar Ubuntu sin complicaciones.

- Dar clip en el icono instalar.
- Seleccionar el idioma de instalación y dar clip en siguiente.
- Seleccionar la ciudad y dar clip en siguiente.
- Seleccionar la distribución del teclado en este caso es Latinoamérica dar clip en siguiente.
- Seleccionar la partición del disco en manual y dar clip en siguiente.
- Aquí crearemos dos particiones una de 600 MB para intercambio y el resto para la instalación de Ubuntu. Damos clip en New partition table, después en New partition , aquí seleccionamos el tamaño de la partición que es de 600 MB, en Use as seleccionamos swap y damos clip en aceptar.
- Ya que hemos llegado a este punto, cómo ya creamos las particiones de vamos usar anteriormente solo bastara con editarlas, así que para la partición más pequeña damos clip en Edit partition y seleccionamos swap damos clip en aceptar, para la siguiente asemos lo mismo solo que en lugar de seleccionar swap seleccionamos Ext3 y punto de montaje / damos clip en aceptar y damos clip en adelante.
- Migramos documentos y configuraciones lo más recomendable es que no migremos nada, damos clip en adelante.
- En esta ventana seleccionaremos un nombre de usuario y contraseña, damos clip en adelante.
- Si todo esta correcto daremos clip en Install, esperaremos que se complete la instalación para después reiniciar, si has seguido estos pasos correctamente habrás instalado Ubuntu sin problema.
- Reiniciaremos la maquina nos saldrá un menú contextual elegimos la primera y dentro de unos segundos estaremos en Ubuntu.

Captura de pantalla del escritorio de Ubuntu.



Instalación de Seismic Unix.

La instalación de Seismic Unix, la vamos a realizar en Ubuntu 7.04., los pasos a seguir son los siguientes.

- Debemos descargar de la siguiente dirección electrónica http://www.cwp.mines.edu/cwpcodes el programa de instalación que es un archivo gzipped es cual tiene el siguiente nombre cwp_su_all_40.tgz o copiarlo del CD de utilidades anexado.
- una vez hecho esto debemos abrir una terminal (nos vamos al menú superior, hay en accesorios y luego en terminal) en Ubuntu y teclear lo siguiente. Lo que pongo en azul es lo que debemos poner en la terminal.

    - **cd /**
    - **sudo mkdir /cwp**
    - **sudo chown nombre_de_usuario /cwp**

- Enseguida moveremos el archivo a la nueva carpeta que creamos la cual está en lugares/equipo/sistema de archivos
- En este paso vamos a modificar el archivo environment en una terminal pondremos lo siguiente.

    - **cd /etc**
    - **sudo cp environment environment.backup**

- Ahora abriremos este archivo con cualquier editor y cambiaremos lo siguiente.

    - **sudo gedit environment**

- Agregaremos estas línea al final de la primera línea ':/cwp/bin' sin las comillas

- Luego agregaremos esto al final en una línea aparte ' CWPROOT="/cwp" ' también sin comillas, salvamos y salimos del editor.
  Reiniciamos sesión y verificamos que los cambios estén hechos de la siguiente forma.
  Tecleamos en una terminal

    - **echo $PATH**
    - **echo $CWPROOT**

  Si hemos tenido problemas con el archivo, el cd de utilidades se incluye este archivo, para que solo lo sustituyan por el que tienen, para que no tengan ningún tipo de dificultades.

- Después tendremos que actualizar algunas partes del sistema e instalar algunos programas o librerías que son necesarias para el correcta funcionamiento de SU, hay dos formas de realizar esta tarea de forma automática y de forma manual.
  Veremos primero la forma automática: Para esta forma solo bastara, con abrir una terminal y poner lo siguiente es necesario una conexión a internet.

    - **Sudo apt-get install lesstif2 lesstif2-dev g77, gcc libglu1-mesa libglu1-mesa-dev freeglut3 freeglut.dev libxmu6 libxmu-dev libxi6 libxi1-dev xlibs xlibs-dev, lesstif lesstif-dev xlibmesa-glu libglut3 motif**

También hay otra forma con el programa llamado Gestor de paquetes de Synaptic en el cual se encuentra en el menú superior/Sistema/Administración/Gestor de paquetes de Synaptic aquí solo basta con buscar cada programa o librería y seleccionarlo para su instalación, es la instalación que más se recomienda, ya que nunca da problemas.
La forma manual es mas tediosa ya que tendremos que instalar cada librería o programas de forma manual, para la instalación de cada uno solo bastara con ejecutarlos en una terminal y seguir las instrucciones, para los que no

cuenten con internet en el disco de utilidades están todas las librerías y programas para que sean instalados.

- Después de haber instalado todo lo anterior, nos colocaremos en la carpeta de /cwp que creamos anteriormente, en donde colocamos el archivo de descargamos de la siguiente dirección electrónica http://www.cwp.mines.edu/cwpcodes el cual vamos a descomprimir de la siguiente forma.
    - **tar zxvf cwp_su_all40.tgz**

- Una vez descomprimido tendremos que configuara el archivo Makefile.config que se encuentra dentro de la carpeta que descomprimimos anteriormente, para modificar dicho archivo lo haremos desde un editor de texto o desde la terminal de la siguiente forma.
    - **gedit Makefile.config**

  Una vez abierto cambiaremos lo siguiente este paso es crucial para que SU funcione correctamente, ya que la instalación si se esta trabajando bajo Linux o MacosX o Solaris para cada sistema operativo la configuración del Makefile.config es diferente, como en este caso de trabajara con Ubuntu la configuración del Makefile.config queda de la siguiente manera.
    - En la línea donde tenemos esto ENDIANFLAG=-DCWP_BIG_ENDIAN agregaremos este carácter al principio #.
    - Cambiaremos la línea 'CC=cc' por 'CC=gcc'.
    - Agregarnos este carácter # a las líneas del compilador fortran
    - Agregaremos un # la línea FC=ifort porque en su lugar usaremos el compilador g77
    - Por ultimo agregaremos estas líneas al final de la parte de MOTIF.
        - IMOTIF=/usr/include
        - LMOTIF=/usr/lib

Salvamos y quitamos el editor, si hubiese una complicación en el disco de utilidades, se incluye este archivo, para que solo lo remplacen por el que tienen.

- Para finalizar la instalación aremos lo siguiente, en una terminal teclearemos esto.

    - **make install**
    - **make xtinstall**
    - **make xminstall**
    - **make finstall**
    - **make mglinstall**
    - **make utils**

Si todo ha salido correctamente solo basta con teclear unif2 y se nos desplegara en la pantalla la ayuda de este comando o este comando suplane | suxwigb & donde nos mostrara una grafica.

. Apéndice B

PRINCIPIOS EN LINUX Y SHELL SCRIPTING.

REDIRECCIONES y CONECTORES (PIPES)

Redirecciones: Sirven para cambiar la entrada estándar (stdin) y/o la salida estándar (*stdout*) de un comando usado en el shell.
El símbolo < re direcciona la entrada estándar a un archivo, cuyo nombre debe ser especificado inmediatamente después de < El símbolo > re direcciona la salida estándar a un archivo cuyo nombre va especificado luego de >; si este archivo no existe, es creado, y si existe, se borra el contenido y se sobre escribe sobre él. El símbolo >> cumple la misma función que >, con la diferencia que si el archivo donde se re direcciona la salida ya existe, la información se escribe al final del mismo, sin borrar su contenido previo. Por ejemplo:

[ubuntu]:> ls >lista.txt

">redirecciona la salida de ls hacia el archivo lista.txt."

Conectores (Pipes): Se utilizan para conectar comandos en una misma línea de comando. Los conectores están representados por el signo **|** (pipe en ingles), y va localizado entre dos comandos tales que la salida del primer comando sea la entrada del siguiente.
Por ejemplo:

[ubuntu]:> cat archivo.txt | grep hola | ....

"|" permite que la salida de *cat* sea la entrada de *grep*, y también que la salida de *grep* sea la entrada de otro comando, y así sucesivamente.
Estas dos acciones representan los complementos más básicos que se necesitan saber para empezar a aprender a utilizar la terminal de Linux.

NOTA: Vale acotar que todos los comandos que se presentaran a continuación tienen su equivalente en el entorno grafico. Sin embargo, si se va a trabajar con SU, más vale acostumbrarse a realizar estas tareítas arcaicas...

NOTA2: Todos estos comandos, y muchos otros, poseen un manual que explica que hace el comando, como usarlo, cuáles son sus *flags* y muestra ejemplos. El comando *man* seguido del nombre del comando abrira este manual.

ALGUNOS COMANDOS DEL SHELL DE LINUX

- ***cd***: Este es quizás el comando que más se usa en el Shell. Este comando te lleva al directorio cuya dirección se especifica inmediatamente después de ***cd***. Por ejemplo:

[ubuntu]:> cd /home/osk/Documentos

Esto hará que te ubiques en el directorio especificado por la dirección /home/osk/Documentos.
Uno podría cambiarse directorio actual, por medio de *cd*, de 2 maneras: una es escribiendo *cd* seguido de la ruta absoluta a la cual se quiere ir ***(ej: /home/osk/Documentos o cualquier otra)***, y otra es usando la ruta relativa, la cual te ubica en el directorio que tú buscas, tomando como referencia tu directorio actual de trabajo. Las rutas pueden ser relativas a tu directorio de trabajo actual, en este caso"***./"***, y relativas al directorio padre de tu directorio actual, en ese caso ***"./"***. Por ejemplo:

[ubuntu]:> cd ../

Este comando te ubica en el directorio padre de tu directorio actual. En el caso que nos encontremos ubicados en la dirección /home/osk/Documentos, la ejecución del ejemplo anterior me llevaría al directorio /home/osk.

- ***pwd***: Este comando te imprime en pantalla tu ubicación actual dentro de la terminal. Por ejemplo:

  [ubuntu]:> pwd
  /home/osk/Documentos
  [ubuntu]:>

- ***ls***: Este comando lista el contenido de archivos y directorios dentro de un directorio, cuya ubicación debe ser dada. Por defecto, ls (sin argumentos) listará el contenido de los archivos y directorios que estén dentro del directorio actual de trabajo. Este comando tiene ciertos flags importantes
-

  como -l, el cual muestra la permisologia de los archivos dentro de la ruta especificada y -s que muestra el tamaño en bytes de cada archivo.

  [ubuntu]:> ls
  archivo1.txt
  archivo2.txt
  archivo3.txt

- ***cat***: Conecta archivos. Si la entrada de ***cat*** son dos archivos, este devuelve ambos archivos en uno solo. Por ejemplo.

  [ubuntu]:> cat archivo1.txt archivo2.txt >all.txt

  Donde el contenido del archivo all.txt corresponde al contenido de archivo1.txt junto con archivo2.txt. También ***ca***t sirve para imprimir en pantalla el contenido de un archivo. Por ejemplo:

  [ubuntu]:> cat archivo1.txt

Hola, mi nombre es Oscar
Vivo en D.F.
Soy estudiante de geofísica...

- ***mkdir***: Crea un directorio. El nombre del nuevo directorio debe incluir su ruta (dirección absoluta o relativa), en caso que no se especifique la ruta, el directorio se creara en el directorio actual de trabajo. Por ejemplo:

  [ubuntu]:> pwd
  /home/osk/Documentos
  [ubuntu]:> mkdir Directorio1
  [ubuntu]:> ls
  archivo1.txt
  archivo2.txt
  archivo3.txt
  Directorio1/
  [ubuntu]:>

- ***cp***: Copia archivos o directorios. La sintaxis es ***cp*** seguido del archivo(s) a copiar, seguido del destino de la(s) copia(s) (ruta absoluta o relativa). Por ejemplo:

  [ubuntu]:> cp archivo1.txt /home/osk/Documentos/Directorio1/

  Copia el archivo archivo1.txt, en el directorio Directorio1. Otra manera de hacer esto mismo es escribir la ruta relativa del directorio destino (=:/Directorio1/). Para copiar directorios es necesario colocar el flag * r, el cual hace que ***cp*** copie recursivamente todos los archivos dentro del directorio especificado, y dentro de todos los directorios hijos del primero.

- ***rm:*** Borra archivos y/o directorios. Para borrar directorios es necesario colocar el flag * r.

- ***mv***: Mueve archivos y/o directorios. La sintaxis es igual a cp. También mv sirve para renombrar archivos y directorios.

- ***chmod***: Este comando es muy importante al momento de la creación de Shell-Scripts, ya que modifica los permisos de lectura, escritura y ejecución. En Linux, todos los archivos tiene una permisologia determinada, la cual admite o restringe las acciones especificadas anteriormente. Si, por ejemplo, un archivo tiene permisologia solo de lectura, este puede ser leído por el usuario más no se podrá escribir sobre él, ni tampoco se podrá ejecutarlo. Además de esto, los permisos son diferentes para cada tipo de usuario. Existen tres tipos de usuarios: Propietario, que es el que crea el archivo en su sesión; usuario de Grupo, el cual tiene su sesión en el mismo grupo del propietario; y Otros, que son todos aquellos usuarios que no son ni el propietario ni pertenecen al grupo del propietario. El comando ***ls*** * l lista los archivo y directorios de un directorio determinado, junto con la información de permisologias de cada archivo y directorio. Por ejemplo:

[ubuntu]:> ls -l
total 4
-rw-r--r-- 1 jlara jlara 0 2007-03-20 18:13 archivo1.txt
-rw-r--r-- 1 jlara jlara 0 2007-03-20 18:13 archivo2.txt
-rw-r--r-- 1 jlara jlara 0 2007-03-20 18:13 archivo3.txt
drwxr-xr-x 2 jlara jlara 4096 2007-03-20 18:14 Directorio1
[ubuntu]:>

La siguiente lista muestra el contenido del directorio /home/osk/Documentos, conjunto una serie datos importantes acerca de cada archivo. Se explicara de manera general como se lee esta información.

La primera letra de la lista identifica el tipo de archivo (en este caso, el único archivo identificado es el directorio, con la letra “d”), luego, las tres letras siguientes definen los permisos de \lectura"(“r” read), “escritura” (“write”) y “ejecución”(e “x”ecution) de cada archivo, para el Propietario. Podemos ver que los tres primeros archivos tienen permiso de lectura y escritura, mas no de ejecución. Las siguientes tres letras definen los permisos para el Grupo, y las siguientes tres letras definen los permisos para los Otros usuarios. La información siguiente se refiere a los enlaces que tienen cada archivo, los nombres del propietario y el grupo, tamaño, fecha de modificación y hora y el nombre del archivo.

El comando ***chmod*** hace posible el cambio de permisologia de los archivos. Ahora bien, cada permiso tiene un “peso” en número, donde el permiso de lectura vale 4, el de escritura vale 2, y el de ejecución vale 1. El siguiente ejemplo muestra la manera de ejecutar ***chmod***:

[ubuntu]:> chmod 644 archivo1.txt
[ubuntu]:>

Donde se ha dado permiso de lectura y escritura (4+2) al usuario Propietario sobre el archivo archivo1.txt, y se ha dado permiso sólo de lectura (4) a los otros usuarios. Es decir, en la secuencia de tres números luego de ***chmod,*** el primer número ajusta los permisos del Propietario, el segundo número ajusta los permisos del Grupo y el tercero ajusta los permisos de Otros. Si se quiere dar todos los permisos a todos los usuarios sobre un archivo, se coloca la secuencia de números 777.

Habiendo entendido esto nos preguntamos, para qué sirve saber y controlar la permisologia de los archivos?.

La respuesta inmediata seria por seguridad, para proteger ciertos archivos o directorios que no deseamos que otros usuarios vean, modifiquen o ejecuten. La otra razón tiene que ver la generación de Shell-Scripts.

Todos los archivos, al crearse, poseen por defecto permisologia de lectura y escritura para el Propietario,y permiso de lectura para los demás usuarios; ahora, los Shell-Scripts son archivos de texto ejecutables, es decir, debemos ejecutarlos para que estos realicen las tareas para las cuales se hicieron. Para poder ejecutar un script, entre otras cosas, debemos primero cambiar su permisologia de tal manera que podamos ejecutarlo. Aquí vemos una manera de realizar esto:

[ubuntu]:> chmod 744 archivo2.txt
[ubuntu]:>

Aquí le damos al archivo todos los permisos para el Propietario (4 + 2 + 1 = 7), y permiso de lectur para el Grupo y Otros.

CARACTERES ESPECIALES DE BASH:
#
Se coloca para comentar líneas dentro del código. En ciertas ocasiones, particularmente para SU, habrá problemas en la ejecución del script cuando se crean líneas de comentario dentro de los comandos de SU; es decir, líneas de comentario que aparezcan en la mitad de un comando especifico, en el caso que este comando tenga varias líneas de especificación
de parámetros en el script. Por ejemplo:

```
# Este comentario no provocara error
sufdmod2 < modelfile.out

nx=100 nz=100 dx=5 dz=5 tmax=5

#Este comentario podra provocar error
fmax=30 mt=3 > cubo.out
```

Aunque los comentarios son muy importantes cuando se está programando en cualquier lenguaje, la función más importante de # en bash es llamar al operador. Todos los scripts hechos en bash deben empezar con la sintaxis siguiente:

```
#! /bin/bash
....
....
```

El caracter es obligatorio dentro de esta primera línea, donde obligatoriamente esta debe ser la primera línea del Script.
Esto significa que el intérprete que se encuentra en el directorio /bin/bash será el encargado de ejecutar las líneas de texto que se encuentran dentro de ese archivo.

Escapan caracteres. Cuando un carácter tiene una significancia distinta a su valor literal, con se “escapa" ese carácter para que su sentido literal se conserve. También es útil para saltos de líneas cuando los comandos son muy largos.

$
Sustitución de variables. Cuando se intenta llamar al valor de una variable predefinida anteriormente, se usa $ como prefijo al nombre de a variable, para que el valor dentro de ese nombre sea el referido en el caso determinado. Por ejemplo:

```
… …
variable=30
… …
variable2=$variable +1
```

APENDICE C

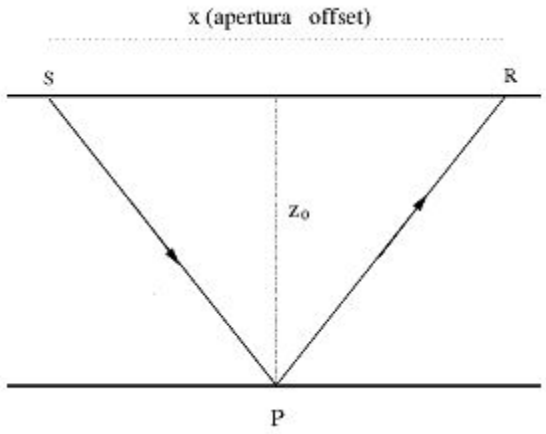
Normal Moveout y Análisis de velocidad

Un registro sónico representa una medida directa de la velocidad con que viajan las ondas sísmicas en el interior de la tierra. Por otra parte, mediante los datos sísmicos podemos obtener un conjunto de medidas indirectas de velocidad a través de las cuales podemos caracterizar un medio. La base para la determinación de velocidades a partir de datos sísmicos de reflexión es el Normal Moveout (NMO). [Yilmaz, 2000]

Introducción al Normal Moveout (NMO)

La Figura muestra el caso simple de una capa horizontal. El tiempo de viaje a lo largo de la trayectoria del rayo desde la posición del disparo (S) al punto en profundidad (P) y luego hasta la posición del receptor (R) es t(x). Usando el teorema de Pitágoras, la ecuación de tiempo de viaje es obviamente.

$$t^2(x) = t^2(0) + \frac{x^2}{v^2}$$



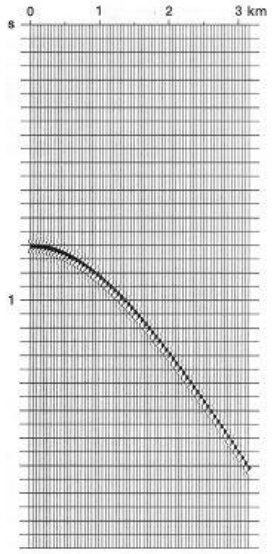
Geometría NMO para un reflector horizontal

Donde x es la distancia entre la fuente y el receptor (offset), v es la velocidad de propagación de ondas en el medio (supuesto homogéneo) que se encuentra sobre el reflector y t(0) es el tiempo de viaje para apertura (offset) = 0, es decir t(0) = $2\frac{zo}{v}$

La proyección en superficie del punto iluminado (P) se encuentra en el punto medio entre S y R, condición que solo ocurre para el caso de reflectores horizontales.

La ecuación $t^2(x) = t^2(0) + \frac{x^2}{v^2}$ describe una hipérbola en el plano (dominio) de tiempo doble de viaje versus offset, esta hipérbola se aproxima asintóticamente a la recta: t(x) = x/v + t, cuya pendiente es claramente la velocidad de propagación en la capa.

La siguiente imagen representa un conjunto de trazas correspondientes a las trayectorias del rayo asociadas con cada par fuente-receptor para el mismo punto en superficie (CMP), este conjunto de trazas se denomina CMP gather. El rango de aperturas fuente-receptor en la grafica es de 0 a 3150 m, con una separación de 50 m entre cada traza. La velocidad del medio arriba del reflector es 2264 m/s, en la grafica se observa claramente la asíntota recta de pendiente 1/v.



CMP Gather con un rango de apertura de 0 a 3150 m, con una separación de 50 m entre cada traza

Corrección NMO

La diferencia entre el tiempo doble de viaje en un offset dado t(x) y el tiempo doble de viaje para offset nulo t (0) es llamada Normal Moveout (NMO) (Δt_{NMO}(x) = t(x) −t (0)).

La corrección NMO consiste en cambiar el tiempo de viaje t(x) a (Δt_{NMO}(x) = t(x) −Δt_{NMO}(x). Obviamente esto da como resultado t_{NMO}(x) = t(0). Pero obviemos esta tautología por un momento a ver a donde llegamos.

Como ya comentamos, la corrección NMO para una apertura x se define como la diferencia entre t(x) y t (0). Para un solo reflector podemos utilizar la siguiente ecuación:

$$\Delta t_{NMO} = t(x) - t(0)\{\sqrt{1+\left(\frac{x}{vt(0)}\right)^2} - 1$$

Si la apertura es "mucho menor" que la profundidad (z_0) se puede llevar a cabo una expansión en series de Taylor alrededor de x = 0. Lo que arroja el resultado aproximado:

$$\Delta t_{NMO} \approx \frac{x^2}{2v^2 t_0}$$

Un artículo de enorme importancia, Taner y Koehler (Taner, M. T. y Koehler, F., 1969) demostraron que en el caso de un subsuelo estratificado de N capas horizontales el tiempo doble de viaje hasta el N-ésimo reflector está dado por:

$$t^2(x) = C_0 + C_1 x^2 + C_2 x^4 + C_3 x^6 + \cdots,$$

donde C0 es el tiempo doble de viaje hasta la el N-ésimo reflector a lo largo de una trayectoria de incidencia normal (offset 0), $C_1 = 1/v_{rms}^2$, la velocidad cuadrática media está dada por:

$$v_{rms} = \frac{1}{t_0}\sum_{i=1}^{N} v_i^2 \Delta t_i$$

y los coeficientes que acompañan a las potencias superiores de la apertura fuente receptor son funciones más complicadas de los espesores y velocidades de las capas.

En la aproximación de apertura pequeña es posible cortar la serie hasta el segundo orden en x para obtener finalmente:

$$t^2 \approx t_0 + \frac{x^2}{v_{rms}^2}$$

Este resultado implica que, en el límite de aperturas pequeñas comparadas con la profundidad del reflector objetivo, las trayectorias de los eventos de reflexión en el dominio CMP gather son hiperbólicas, lo que permite utilizar la formula de corrección NMO aproximada:

$$\Delta t_{NMO} = \frac{x^2}{2t_0 v_{SQRT}^2}$$

En un caso realista el único dato del que dispone el geofísico de procesamiento es el conjunto de trazas de cada CMP, obviamente esto no incluye el valor de v_{rms}, que se convierte entonces en una incógnita del problema de apilamiento y que denominaremos de ahora en adelante v_{NMO} o

velocidad de apilamiento. El problema de estimación del valor de la velocidad de apilamiento para cada reflector es denominado análisis de velocidades y será comentado a continuación.

Análisis de Velocidad

El espectro de velocidades: El CMP gather de la Figura (a) contiene una simple hipérbola de reflexión de una interfaz horizontal, la velocidad del medio arriba del reflector es 3000 m/s. Suponga que este gather es corregido por NMO y apilado usando un rango de velocidades constantes desde 2000 a 4300 m/s. La Figura (b) despliega las trazas apiladas Resultantes para cada velocidad perteneciente al rango mencionado, y se encuentran graficadas en un plano de velocidad versus tiempo doble de viaje para offset cero. Esto es llamado el espectro de velocidades. De esta forma hemos transformado la data desde el dominio de offset versus tiempo doble de viaje (Figura 8.3(a)) a el dominio de velocidad de apilamiento vs tiempo doble para offset cero (Figura (b)).[Yilmaz, 2000]

Podemos ver que la máxima amplitud luego del apilamiento ocurre con una velocidad de 3000 m/s. Por lo tanto ´esta debe ser la velocidad que tiene que usarse para corregir el evento del CMP gather. [Yilmaz, 2000]

Un CMP gather asociado con un modelo de capas es mostrado en la Figura (a), Basa- do en las amplitudes apiladas, las siguientes selecciones para la función de velocidad de apilamiento son hechas a partir del espectro de velocidades (Figura (b)): 2700, 2800 y 3000 m/s. Estas selecciones corresponden a los tres eventos, superficial, medio y profundo respectivamente. [Yilmaz, 2000]

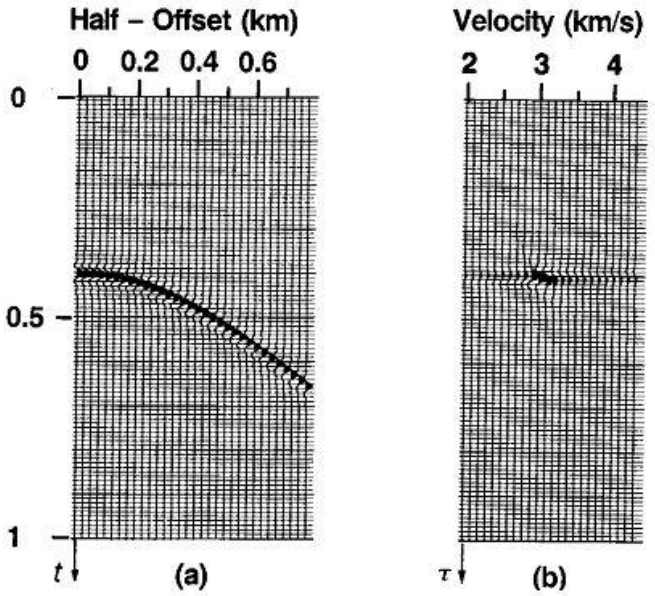


Figura en donde se gráfica el offset vs velocidad, cada traza en el gather (b) es un apilado de las trazas del CMP Gather (a) usando una velocidad de corrección NMO constante [Tomada de: Yilmaz, 2000]

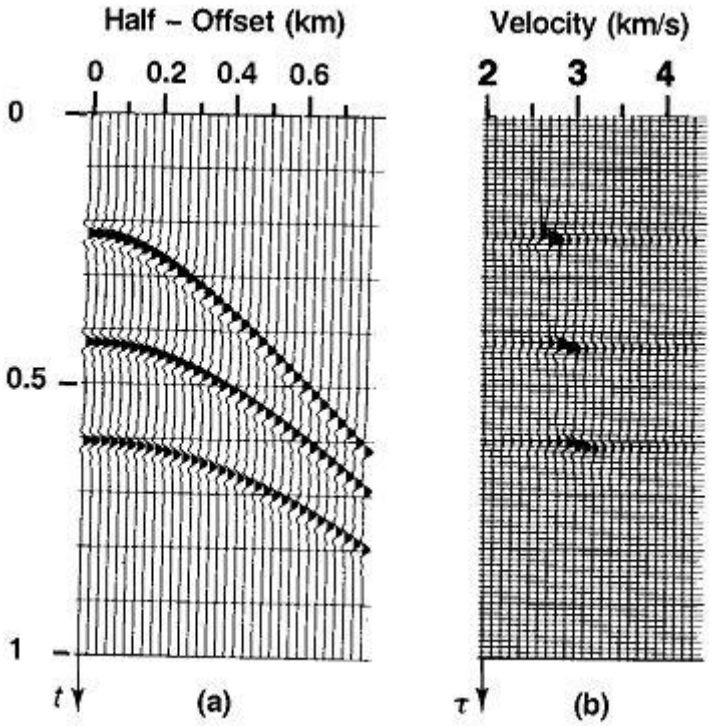
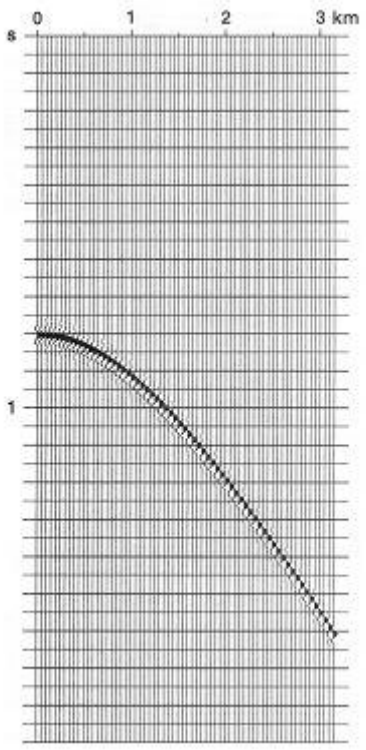


Figura en donde se gráfica el offset vs velocidad, cada traza en el gather (b) es un apilado de las trazas del CMP Gather (a) usando una velocidad de corrección NMO constante [Tomada de: Yilmaz, 2000]

Las cantidades desplegadas en los espectros de velocidades de las Figuras (a) y (b) son las amplitudes apiladas. Cuando la relación señal-ruido (S/N) de la data de entrada es pobre, las amplitudes apiladas pueden no ser las mejores. Lo ideal en el análisis de velocidad es hacer selecciones correspondientes a la mejor coherencia de la señal a lo largo de la trayectoria hiperbólica sobre la extensión completa del CMP gather. Neidell y Taner (1971) resumieron varios tipos de medidas de coherencia que pueden ser usadas como atributos en el cálculo de espectros de velocidad. [Yilmaz, 2000]

Considere un CMP gather con un solo evento (siguiente figura). La amplitud apilada es definida como:

$$S_t = \sum_{i=1}^{M} f_{i,t(i)}$$



Donde $f_{i,t(i)}$es el valor de amplitud en la traza i a un tiempo doble de viaje t(i) y M es el número de trazas en el CMP gather. El tiempo doble de viaje t(i) a lo largo de la hipérbola es:

$$t(i) = [t^2(0) + \frac{x_i^2}{V_{st}^2}]^{1/2}$$

La amplitud de apilamiento normalizada es definida como:

$$NS = \frac{|s_t|}{\sum_i |f_{i,t(i)|}}$$

El rango de NS es 0 ≤ NS ≤ 1. La ecuación anterior implica que la coherencia como es definida aquí, es la amplitud apilada normalizada.

Otra cantidad que es usada en el cálculo de espectros de velocidades es la no-normalizada suma de la crosscorrelación (unnormalized crosscorrecation sum) dentro de una ventana de tiempo que sigue el patrón correspondiente a la hipérbola apilada a través del CMP gather. La expresión para la no-normalizada suma de la crosscorrelación está dada por:

$$CC = \frac{1}{2}\sum_t \left\{ \left[\sum_{i=0}^{M} f_{i,t(i)} \right]^2 - \sum_{i=0}^{M} f_{i,t(i)}{}^2 \right\} = \frac{1}{2}\sum_t \left[S_t^2 - \sum_t f_{i,t(i)}{}^2 \right]$$

Donde CC puede ser interpretado como un medio de la diferencia entre la energía de salida y la energía de entrada de la sección . Una forma normalizada de CC es otro atributo que a menudo es usado en el cálculo de espectros de velocidad y esta dado por la siguiente fórmula:

$$NCC = \frac{2}{M(M-1)} \sum_t \sum_{k=1}^{M-1} \sum_{i=1}^{M-k} \frac{f_{i,t(i)} f_{i+k,t(i+k)}}{\left[\sum_t f_{i,t(i)}{}^2 \sum_t f_{i+k,t(i+k)}{}^2 \right]^{1/2}}$$

La energía normalizada de la suma de la crosscorrelacion (energy-normalized crosscorrelation sum) corresponde con:

$$ECC = \frac{2}{(M-1)} \frac{CC}{\sum_t \sum_{i=1}^{M} f_{i,i(t)}{}^2}$$

El rango de ECC es –[(1/(M-1))] < ECC ≤ 1. Finalmente la semblanza que es el cociente entre la energía de salida y de entrada normalizado (normalized output-to-input energy ratio) viene dada por:

$$NE = \frac{1}{M} \frac{\sum_t^2 1}{\sum_t \sum_{i=1}^{M} f_{i,t(i)}^{\ 2}}$$

La siguiente expresión muestra la relación entre NE y ECC:

$$ECC = \frac{1}{M-1}(M * NE - 1)|$$

El rango de NE es 0 ≤ NE ≤ 1

El espectro de velocidad normalmente no es desplegado tal como se muestra en las figuras (a) y (b), en su lugar generalmente es usado un mapa de contornos. En este trabajo se empleo una grafica que muestra el valor de la semblanza a partir de una degradación de colores, los cuales están referenciados a una escala.

Principios básicos de la Migración sísmica

La Migración sísmica presenta dos objetivos fundamentales. El primero es retornar eventos a su verdadera posición y el segundo está relacionado con el colapso de difracciones generadas por las discontinuidades geológicas:

Ubicación de eventos inclinados

En la Figura (a) se puede ver que para el caso de un reflector horizontal, la proyección en superficie (M) del punto iluminado D, corresponde con el punto medio entre la fuente (S) y el receptor (R).

Ahora, cuando estamos en presencia de reflectores inclinados (Figura (b)) la proyección del punto de reflexión no concuerda con la distancia media entre la fuente y el receptor. Sin embargo, cuando se procesan datos sísmicos, todas las señales recibidas son asignadas al punto medio entre la fuente y el receptor.

Es decir, se asume que las capas son planas, horizontales y homogéneas. Obviamente esto se hace porque no se conocen las características de las interfaces. El planteamiento anterior genera que los eventos obtenidos en una sección sísmica luego del apilamiento presenten posiciones

incorrectas. En este sentido, la Migración tiene como finalidad “ubicar eventos sísmicos inclinados en su posición real”.

Considere el reflector inclinado CD que se muestra en la sección geológica en profundidad de la Figura 2(a). Se ha obtenido una sección cero-offset a lo largo de 0x. La primera incidencia normal proveniente del evento es grabada en la localidad A. En este ejemplo se asume un medio con velocidad constante (v = 1), de manera que las coordenadas de tiempo y profundidad sean intercambiables. La reflexión grabada en el punto

A es indicada por el punto C’ en la sección temporal cero-offset mostrada en la Figura 8.6 (b).

La última llegada es grabada en el punto B (Figura 2(a)), la cual es indicada por el punto D’ en la Figura 2(b). (En este ejemplo las difracciones en los bordes del reflector CD son ignoradas para simplificar el caso).

Comparando la sección geológica de la figura 2(b) que esta en tiempo. La verdadera posición del reflector CD es superpuesta en la sección temporal para facilitar la comparación. Claramente, la verdadera posición del reflector CD no es la misma que la representación temporal encarnada por C’D’. [Yilmaz, 2000]

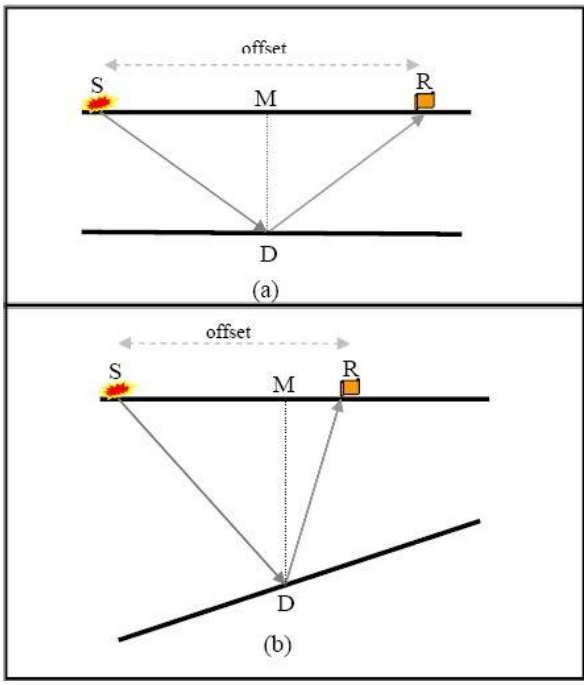


Figura 1: Geometría de reflexión para: (a) una interfaz horizontal y (b) una interfaz inclinada. En (a) la proyección en superficie (M) del punto iluminado D, corresponde con el punto medio entre la fuente (S) y el receptor (R), lo cual no se cumple en (b).

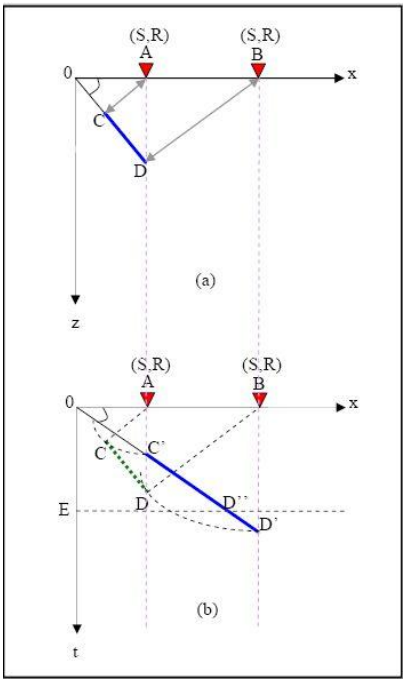


Figura 2: principios de la Migración. El reflector C'D' mostrado en la sección temporal (b), obtenido a partir de reflexiones con offset cero, no corresponde con su verdadera posición en profundidad CD. Recreada de: Yilmaz, 2000.

A partir del ejemplo ilustrado por la Figura 2 se puede notar la necesidad migrar el evento C'D' hacia su verdadera posición CD. Las siguientes observaciones pueden ser hechas a partir de la descripción geométrica de migración en explicada: [Yilmaz, 2000]

1. El ángulo de buzamiento del reflector CD es mayor que el de C'D'. Por lo tanto, la Migración debe aumentar el ángulo de buzamiento.

2. La longitud de el reflector CD es menor que la de C'D'. Lo que sugiere que la Migración debe acortar la longitud de los reflectores.

3. Migración debe mover los reflectores buzamiento arriba.

El desplazamiento vertical (tiempo) y horizontal, dt y dx que debe ser aplicado al reflector C'D' para retornarlo a su verdadera posición CD y el ángulo de buzamiento después de la migración $\bar{\theta}$t (todas cantidades de la sección sísmica migrada) pueden ser expresados en términos de la velocidad del medio v, tiempo de viaje t, y el buzamiento aparente del reflector no migrado θt, Chun y Jacewitz (1981) derivaron las siguientes formulas: [Yilmaz 2000]

$$d_x = \frac{(v^2 t * tan\theta_t)}{4}$$

$$d_t = t\left\{1 - \left[1 - \frac{(v^2 * tan^2\theta_t)}{4}\right]^{1/2}\right\}$$

$$tan\bar{\theta}_t = \frac{tan\theta_t}{\left[1 - \frac{(v^2 * tan^2\theta_t}{4}\right]^{1/2}}$$

Donde $tan\ \theta_t = \frac{\Delta t}{\Delta x}$.

A través de los principios deducidos mediante la geómetra de la Figura 2, se pueden entender los conceptos migratorios involucrados en un sinclinal. En la Figura 3 se observa una sección offset-cero correspondiente a un sinclinal. En (a) mostramos la sección antes de la migración y en (b) luego de la migración. Note que a través de la migración el segmento A es movido buzamiento arriba (hacia la izquierda) y el segmento B se desplaza hacia la derecha. También se puede notar que los ángulos de buzamiento han sido aumentados. Estos procesos eliminan la intercepción entre ambos reflectores y definen la geometría real del evento.

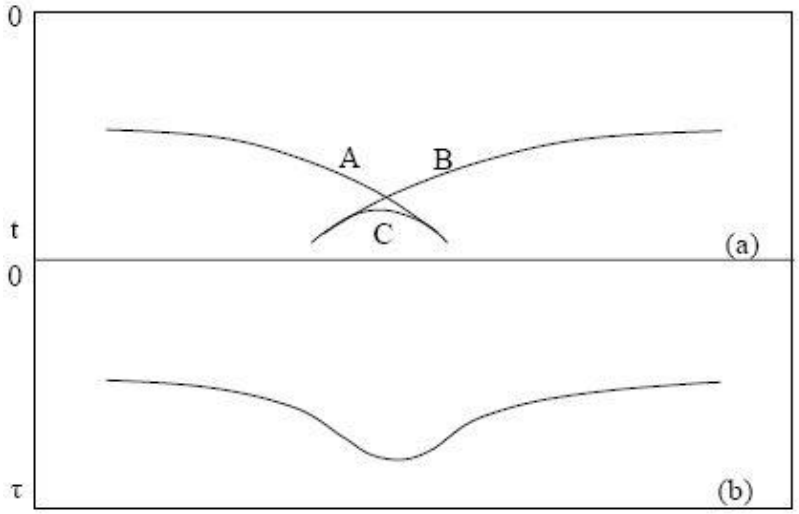


Figura 3: Curvas reflectoras correspondientes a un sinclinal. (a): antes dela migración. (b) después de la migración.

Como fue comentado al principio de la discusión, el segundo objetivo de la Migración sísmica es el colapso de difracciones asociadas a las discontinuidades geológicas. A continuación se presenta un breve resumen en relación a este tema:

Colapso de difracciones a través de la migración sísmica

Las ondas sísmicas son difractadas en las discontinuidades geológicas. Para entender el fenómeno y la forma como ocurre este proceso consideremos el Principio de Huygens.

Principio de Huygens: Cada punto de un frente de ondas (propagándose en un medio isotrópico) puede ser considerado como una fuente secundaria de ondas esféricas [Scales, 2006]. La Figura 4 ilustra el principio de Huygens. Cada onda esférica secundaria satisface la ecuación:

$$(z - z_0^1)^2 + (y - y_0)^2 + (x - x_0)^2 = v^2(t - t_0)^2$$

Donde (x_0, y_0, x_0) es el origen de la onda esférica y t_0 es su tiempo de inicio.

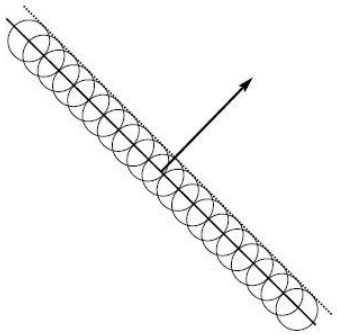


Figura 4: Principio de Huyens: Cada puno de un frente de ondas puede ser considerado como una fuente secundaria de ondas esféricas [Tomada de: Scales, 2006].

Claerbout (1985) usa el ejemplo de la Figura 5 para describir el principio físico de la migración. Asume que una barrera existe a alguna distancia Z de una playa y que hay un agujero en la barrera. Imagine una onda plana proveniente del océano en donde el frente de onda es paralelo a la barrera. El agujero en la barrera a actuado como una fuente secundaria y a generado los frentes de ondas semicirculares que se propagan hacia la orilla. Si se coloca un cable de receptores a lo largo de la playa para detectar las ondas que se aproximan se obtiene la sección temporal que se muestra en la Figura 5 (b). El agujero en la barrera es llamado en términos físicos un punto de apertura, lo cual es similar a un punto fuente, ya que ambos generan frentes de ondas circulares. El punto de apertura en la barrera actúa como una fuente secundaria de Huygens.[Yilmaz, 2000]

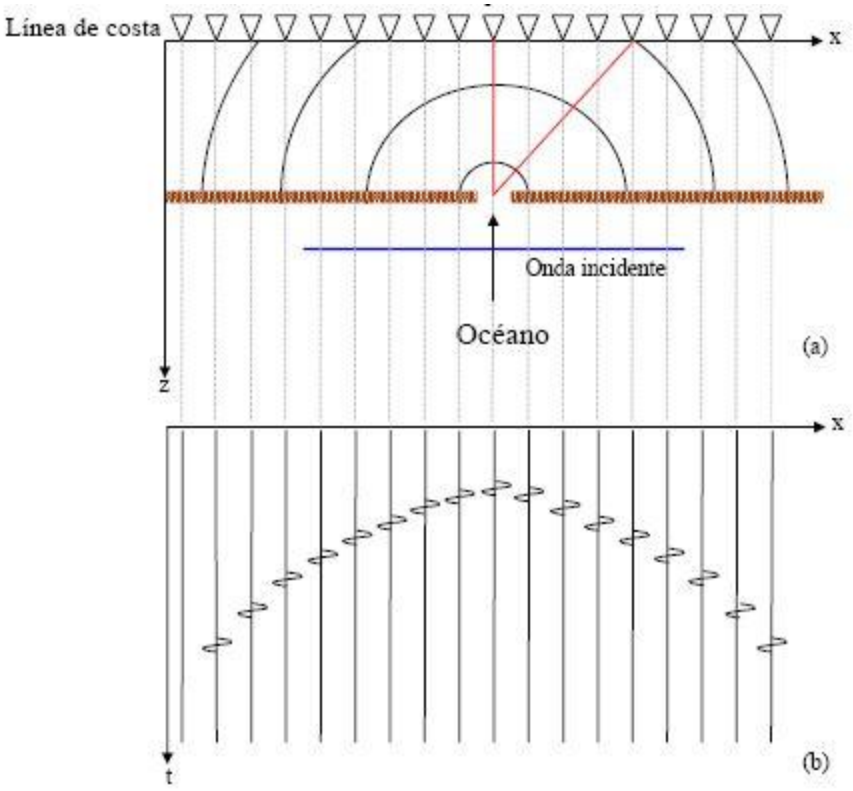


Figura 5: Generación de frentes de ondas circulares a partir de la incidencia de una onda plana en un punto de apertura: El agujero en la barrera actúa como una fuente secundaria de Huygens, causando los frentes de ondas circulares que se aproximan hacia la costa. Recreado de:[Yilmaz, 2000].

La difracción de las ondas acústicas que se genera en torno a las discontinuidades geológicas es un proceso similar al ejemplo resumido en la Figura 5. Por tanto, podemos decir que tales difracciones se comportan como una fuente secundaria de Huygens. De esta forma generan frentes de ondas semicirculares en el plano (x,z) y la respuesta en el plano (x,t) es la hipérbola de difracción mostrada en la Figura 6(b).[Yilmaz, 2000].

Imagine que el subsuelo está conformado por un conjunto de puntos a lo largo de cada horizonte de reflexión que actúan como el agujero en la barrera del ejemplo anterior. A partir de este planteamiento podemos afirmar que estos puntos se comportan como fuentes secundarias de Huygens y producen curvas hiperbólicas de tiempos de viaje. Este fenómeno se produce por la difracción de las ondas acústicas en las discontinuidades geológicas.

En la Figura 6 se muestra la representación de un punto de difracción en el plano (x,t). De forma similar, en la Figura 6 se muestra un evento formado por un conjunto de puntos de difracción y su representación en la sección temporal. En esta ultima figura se debe notar como las fuentes (los puntos en la interfaz reflectora) se encuentran cerca, la superposición de las hipérbolas producen la imagen de la interfaz reflectora (Figura 7). [Yilmaz, 2000]

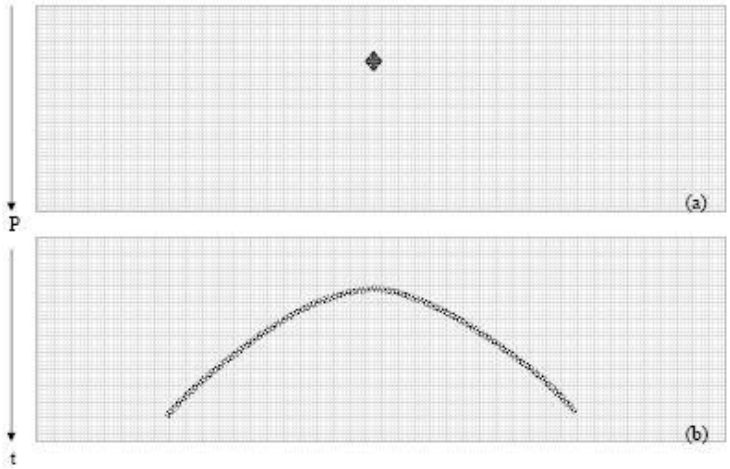


Figura 6: Representación de un punto de difracción en el plano (x,t): Un punto que representa una fuente secundaria de Huygens en la sección de profundidad (a) genera una hipérbola de difracción en la sección temporal offset-cero (b). El eje vertical es el tiempo doble del viaje.

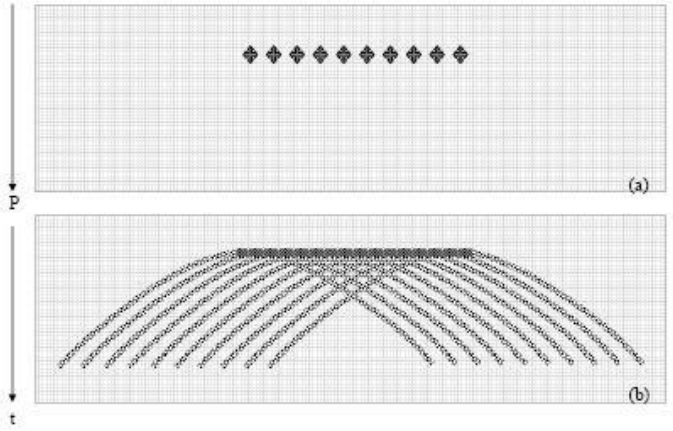


Figura 7: Un conjunto de puntos que actúan como fuentes secundarias de Huygens en la sección en profundidad (a) generan una serie de hipérbolas de difracción en la sección temporal offset-cero (b).

Si consideramos un reflector horizontal, las difracciones se producen en los extremos del evento figura 8.

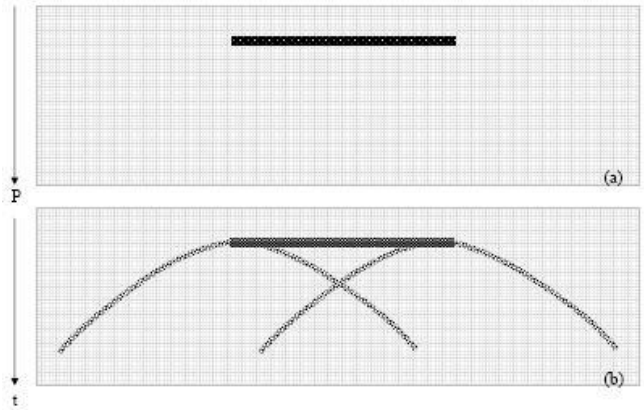


Figura 8: En el reflector horizontal que se muestra en la sección en profundidad (a) se producen difracciones en los extremos, lo cual genera las hipérbolas de difracción que se muestran en la sección temporal (b)

APENDICE D

Script que modela una adquisición sísmica y genera una sección apilada partiendo de la definición de algunas propiedades del subsuelo y los parámetros de adquisición.

Script 8. Script que modela una adquisición sísmica y genera una sección apilada partiendo de la definición de algunas propiedades del subsuelo y los parámetros de adquisición.

```
#! /bin/sh

#### Generación de una sección apilada a partir del modelo de una adquisición sísmica 2D

#### Hay dos secciones en donde se deben ajustar los parámetros, que es la sección siguiente
#### y antes de comenzar el análisis de velocidad.

### Ajuste de los parámetros (sección 1).
### Parámetros definidos por el usuario.

## Archivo de entrada: Define  la ubicación y forma de las interfaces.
Archivo_Entrada=Modelo1

## Longitud del perfil (metros)
Longitud=5000

## Profundidad del perfil (metros)
Profundidad=3000

## Velocidad de cada capa (m/seg): v00=v1 ,v2 ,v3 ….
v00=1500,2500,3500

### Nota: Los valores de densidad deben ser ingresados directamente en el comando UNIF2 en
### la parte destinada a la generación del perfil de densidad ya que corresponde el mismo
### parámetro v00

## Offset máximo (metros).
offset_max=1500

## Intervalo entre disparos (ID).
ID=50

## Intervalo entre grupos receptores (IR) (metros).
IR=25

## Tiempo máximo de grabación (seg)
tmax=5


### Ajuste de los parámetros de adquisición a través de los que fueron definidos anteriormente
### Estos son definidos para facilitar el manejo de las variables.
```

```
## Longitud del tendido (metros): 2 veces del offset máximo, debido a nuestra geometría de
## adquisición.
LT=$((2*offset_max))

## Numero de disparos (ND).
ND=$(((Longitud-LT)/ID))

## Numero de grupos de receptores (NR).
NR=$((LT/IR+1))

## NR=Igual al número de muestras que serán ingresados al comando SUFDEMOD2.
## Ajuste de los parámetros usados por los programas.
nx=$NR
dx=$IR
nz=$((Profundidad/IR+1))
dz=$IR
tmax=$tmax
fpeak=5
fmax=10
mt=4
clip=0.15  # Este valor puede variar.

## Generación del perfil de velocidad del modelo completo.
unif2 < $Archivo_Entrada \nx=$((Longitud+1)) dx=1 nz=$((Profundidad+1)) dz=1 \v00=$v00 >
velocidad

## Generación del perfil de densidad del modelo completo.
unif2 < $Archivo_Entrada \nx=$((Longuitud+1)) dx=1 nz=$((Profundidad+1)) dz=1
\v00=1.0,2.25,2.25 > densidad

## Graficación del perfil de velocidad.
ximage < velocidad title="Perfil de Velocidad de onda P" \n2=$((Longitud+1)) d2=1
n1=$((Profundidad+1)) d1=1 \legend=1 units='Velocidad (m/seg)' \label1='Profundidad (m)'
label2='Distamcia (m)' \wbox=1000 hbox=600 lx=0.0 \

## Graficación del perfil de densidad.
ximage < densidad title="Perfil de Densidad" \n2=$((Longitud+1)) d2=1 n1=$((Profundidad+1))
d1=1 \legend=1 units='Densidad (gr/cc)' \label1='Profundidad (m)' label2='Distamcia (m)'
\wbox=1000 hbox=600 lx=0.0 \




## Graficación del perfil de velocidad (Formato PostScript)
```

```
psimage < velocidad title="Perfil de velocidad de onda P" \n2=$((Longitud+1)) d2=1 n1=$((Profundidad+1)) d1=1 \legend=1 units='Velocidad (m/seg )' \label1='Profundidad (m)' label2='Distancia (m)' \width=4.5 height=2.5 lx=0.6 d1num=500 d2num=1000 \legendfont=times_roman8 labelsize =12 titlesize =18 > Velocidad_completo.ps

## Graficación del perfil de densidad (Formato PostScript).

psimage < densidad title="Perfil de densidad" \n2=$((Longitud+1)) d2=1 n1=$((Profundidad+1)) d1=1 \legend=1 units='Densidad (gr/cc )' \label1='Profundidad (m)' label2='Distancia (m)' \width=10.0 height=6.0 lx=0 d1num=500 d2num=1000 \legendfont=times_roman8 labelsize =12 itlesize =18 > Densidad_completo.ps

### Generación de los Shots Gathers.

LIMIT=$((ND*ID-ID))
for ((fx=0 ; fx<=LIMIT ; fx=$((fx+5*ID)) ))
do

## Generación de los perfiles de velocidad correspondientes a la longitudes de cada tendido.


unif2 < $Archivo_Entrada fx=$fx nx=$nx dx=$dx nz=$nz dz=$dz \v00=$v00 > vel.out

#Generación de los perfiles de densidad correspondiente a cada tendido

unif2 < $Archivo_Entrada fx=$fx nx=$nx dx=$dx nz=$nz dz=$dz \v00=1.0,2.25.2.25 > den.out

#Graficación del perfil de velocidad del tendido (x window graphic)

ximage < vel.out title="Perfil de Velocidad del tendido" \f2=$fx n2=$nx d2=$dx n1=$nz d1=$dz \legend=1 units='Velocidad (m/seg)' \label1='Profundidad (m)' label2='Distancia (m)' \wbox=800 hbox=800 lx=0.0

## Graficación del perfil de densidad del tendido (x Windows graphic).

ximage < den.out title="Perfil de densidad del tendido" \f2=$fx n2=$nx d2=$dx n1=$nz d1=$dz \legend=1 units='Densidad (gr/cc)' \label1='Profundidad (m)' label2='Distancia (m)' \wbox=800 hbox=800 lx=0.0

## Graficación del perfil de velocidad del tendido (PostScript).

psimage < vel.out title="Perfil de Velocidad del primer tendido" \f2=$fx n2=$nx d2=$dx n1=$dz d1$dz \legend=1 units='Velocidad (m/seg)' \label1='Profundidad (m)' label2='Distancia (m)' \width=10.0 height=6.0 lx=0.6 d1num=500 d2num=1000 \legendfont=times_ronam8 labelsize=12 titlesize=18 \ > velocidad{$((fx+offset_mx))}.ps

## Graficación del perfil de densidad del tendido (PostScript).
```

```
psimage < den.out title="Perfil de Densidad del primer tendido" \f2=$fx n2=$nx d2=$dx n1=$dz
d1=$dz \legend=1 units='Densidad (gr/cc)' \label1='Profundidad (m)' label2='Distancia (m)'
\width=10.0 height=6.0 lx=0. d1num=500 d2num=1000 \legendfont=times_ronam8
labelsize=12 titlesize=18 \ > densidad{$((fx+offset_mx))}.ps

## Modelado mediante diferencias finitas.

sufdmos2 < vel.out dfile=den.out xs=((fx+offset_max)) zs=0 fx=$fx \nx=$nx nz=$nz dx=$dx
dz=$dz \tmax=$tmax fpeak=$fpeak fmax=$fmax \abs=1,1,1,1 mt=$mt verbose=0
\hsfile=gather{$((fx+offset_max))} > ONDAS

## Despliegue de la película de la parte de la propagación de ondas.

suxmovie < ONDAS f2=$fx n1=$nz n2=$nx d1=$dz d2=$dx \title="Propagacion de ondas"
\label1="Profundidad  (m)" label2="Distancia (m)" \clip=0.3 loop=1

## Graficación del Shot Gather.

suxwigb < Gather{$((fx+offset_max))} f2=$fx clip=$clip \title="Shot Gather correspondiente a la
posicion $fx m" \label1='Tiempo (seg)' label2='Distancia (m)' \

## Pelicula de la propagación de las ondas.

suspmovie < ONDAS title="Propagacion de ondas acústicas" \f2=$fx n2=$nx d2=$dx n1=$nz
d1=$dz lx=0 \label1='Profundidad (m)' label2='Distancia (m)' \clip=0.3 wbox=5 hbox=5
d1num=500 d2num=500 labelsize=17 \ > Ondas{$((fx+offset_max))}.ps

## Graficación del Shot Gather.

suspwigb < Gather{$((fx+offset_max))} f2=$fx clip=0.2 \title="Shot Gather en la posicion
$((fx+offset_max)) m" \label1='Tiempo (seg)' label2='Distancia (m)' \wbox=8.5 hbox=8.5
titlesize=14 d2num=500 \labelsze=18 labelsize=12 > gather{$((fx+offset))}.ps

## Graficación del Shot Gather (imagen).

suspimage < gather {$((fx+offsset_max))} f2=$fx clip=0.2 \title="Shot Gather en la posicion
$((fx+offset_max)) m" \label1='Tiempo (seg)' label2='Distancia (m)' \wbox=8.5 hbox=8.5
titlesize=14 d2num=500 \labelsize=18 labelsize=12 > gatherIma{$((fx+offset_max))}.ps
done

## Union de los Shot Gathers.
# Primer Shot Gather.

Primer_Archivo=gather {1500}
suwind < $Primer_Archivo j=1 > SHOTS
LIMITE=$((ND*ID-ID+offset_max))
for ((x=$((offset_max+ID)) ; x <= LIMITE ; x=$((x+ID)) ))
```

```
do
cat SHOTS gather{$x} > SHOTS
done

## Ajuste del Header.

sushw < SHOTS key=tracl,tracr,tracf,sx,fldr,offset a=1,1,1,1, $((LT/2)),1,-$((LT/2))
b=1,1,1,0,0,$dx c=0,0,0,$ID,1,0 j=$(($nx*$ND)),$(($nx*$ND)),$nx,$nx,$nx,$nx > SHOTS.HEA

sushw < SHOTS.HEA key=gx a=0 b=$bx c=$IS j=$nx >  SHOTS.HEADER

suchw < SHOTS.HEADER key1=cdp key2=gx key3=sx b=1 c=1 d=2 > SHOTS.CMP

## Graficación de todos los Shots Gathers (x window)

suwind < SHOTS.CMP j=1 |
suxwigb clip=0.1 f2=1475 d2=1.652892 d2num=50 \title="Shots Gathers" \label1='Tiempo (seg)'
label2='Distancia (km)' \

## Los cuatro primeros Shots Gathers (PS).

suwind < SHOTS.CMP j=4 |
suwind key=fldr min=1 max=4 |
suspswigb clip=0.09 f2=3275 d2=1.652 d2num=50 \title='4 primeros Shot Gathers'
\label1='Tiempo (seg)' label2='Distancia (m)' \wbox=8.5 hbox=8.5 titlesize=14 labelsize=18
labe2size=12 \ > Shots.ps

## Los cuatro primeros Shots Gathers (Image).

suwind < SHOTS.CMP j=4 |
suwind key=fldr min=37 max=40 |
suspsimage clip=0.09 f2=3275 d2=1.652 d2num=50 \title= "4 primeros Shot Gathers"
\label1='Tiempo (seg)' label2='Distancia (m)' \wbox=8.5 hbox=8.5 titlesize=14 labelsize=18
labe2size=12 \ > ShotsIma.ps

### Orden CMP

susort < SHOTS.CMP > CMP.cdp

## Graficación de todos los CMP Gathers (x-window graphic)

suwind < CMP j=1 |
suxwigb clip=0.1 f2=742 d2=0.3125 d2num=12.5 \title="CMP Gathers" \windowtitle="CMP"
\label1='Tiempo (seg)' label2='Distancia (m)' \


## Graficación de los $ Shots Gathers.
```

```
suwind < CMP key=cdp min=2000 max=2040|

supswigb clip=0.1 f2=1994 d2=0.3125 d2num=12.5 \title="CMP Gather" \windowtitle="CMP"
\label1='Tiempo (seg)' label2='Distancia (m)' > CDP.ps

## Graficación de los 4 Shots Gathers.

suwind < CMP key=cdp min=2000 max=2040|

supsimage clip=0.1 f2=1994 d2=0.3125 d2num=12.5 \title="CMP Gather" \ windowtitle="CMP"
\label1='Tiempo (seg)' label2='Distancia (m)' > CDPIma.ps

### Análisis de Velocidad.

### (1) "Ajuste de parámetros" (seccion dos).

### La línea sísmica ordenada por CMP.

velpanel=CMP

## Nombre del archivo en donde serán almacenados los valores.

vpicks=stkvel.p1

## Selección de los CMP Gathers que van a ser analizados.

cdpmin=1800 #Primer CMP Gather

cdpmax=2100 #Ultimo CMP analizado

dcdp=100 #Intervalo entre los CMPs analizados

## Selección del rango de velocidades a utilizar.

nv=500 # Numero de velocidades

dv=10 # Intervalo entre las velocidades

fv=500 # Primera velocidad

## Selección de los valores del filtro pasa banda.

f=1,10,100,120 #Frecuencias

amps=0,1,1,0 #Amplitudes


## Parámetros correspondientes al comando UNISAM empleando, para graficar la función de
```

```
## Velocidades de apilamiento calculado.

nout=3001 # Numero de valores en el archivo de salida

dxout=0.004 # Intervalo de muestreo en x del archivo de salida

# Otros parámetros.

normpow=0

slowness=0

fold=40

### (2) Definicion del ciclo.

cdp=$cdomin

while [$cdp- le $cdpmax]

do

ok=false

while [$ok=false]

do

## (3) Extracción de cada CMP Gather.

suwind < $velpanel key=cdp min=$cdp max=$cdp count=fold > panel.$cdp

### Graficación de los CMP's Gathers extraidos.

suxwigb < panel.$cdp title="CMP Gather para cdp=$cdp" Xbox=700 clip=0.5
\mpicks=mpicks.$cdp &

## (4) Aplicación de la ganancia y filtrado del CMP Gather.

sugain < panel.$cdp tpow=2 |

sufilter f=$f amps=$amps > panelF.$cdp

## (5) Análisis de Semblanza.

suvelan < panelF.$cdp nv=$nv dv=$dv fv=$fv > semblanza
```

```
## Graficación del mapa de Semblanza.

suximage < semblanza wclip=0.6 bclip=1 f2=$fv d2=$dv \units='semblanza' \label1='Tiempo
(seg)' label2='Velocidad (m/seg)' \title="Velocity Scan for CMP.$cdp" \mpicks=mpicks.$cdp \

## Graficación de los CMP Gathers extraidos (PS)

supsimage < panel.$cdp clip=0.25 \title="CMP Gather para CMP=$cdp" \windowtitle="CMP"
\label1='Tiempo (seg)' label2='Distancia (m)' > CMP1Ima.ps

### Graficación del mapa de Semblanza (PS).

supsimage < semblanza wclip=0.6 bclip=1 f2=$fv d2=$dv \units="semblanza" \label1='Tiempo
(seg)' label2='Velocidad (m/seg)' \title='Analisis de semblanza' > semblanza.ps

## (6) Obtención del archivo *.par

sort < mpicks.$cdp  -n |

mkparfile string1=tnmo string2=vnmo > par.$cdp

## (7) Graficación de la función de velocidad de apilamiento.

sed < par.$cdp '

s/tnmo/xin/

s/vnmo/yin/

' > unisam.p

unisam nout=$nout fxout=0.0 dxout=$dxout \par=unisam.p method=spline > varias.u

xgraph < varias.u n=$nout nplot=1 d1=$dxout f1=0.0 \ label1="Tiempo (seg)" label2="Velocidad
(m/seg)" \title="Funcion de velocidad de apilamiento: CMP $cdp" \grid=solid grid2=solid
\linecolor=2 style=seismic &

psgraph < varias.u n=$nout nplot=1 d1=$dxout f1=0.0 \label1='Tiempo (seg)' label2='Velocidad
(m/seg)' \title="Funcion de velocidad de apilamiento : CMP $cdp" \grid1=solid grid2=solid
\linecolor=2 style=Seismic > funcion.ps

## (8) Análisis de  la grafica y redefinición del cdp.

pause

echo "¿La selección es correcta? (s/n)" | tr –d "\012" > /dev/tty

read response
```

```
case $response in

n*) ok=false ;;

*) ok=true ;;

esac

done < /dev/tty

cdp=‘bc-1 <<END

$cdp+$dcdp

END‘

done

set + x

## (9) Generación de un solo archivo a partir de todos los archivos par.$cdp.

> $vpicks

echo “cdp=” | tr –d “\012” >> $vpicks

cdp=$cdpmin

echo “$cdp” | tr –d “\12” >> $vpicks

cdp=‘bc-1” << END

$cdp+$dcdp

End‘

done

echo >> $vpicks

cdp=$cdpmin

while [$cdp –le $cdpmax]

do

cat par.$cdp >> $vpicks

cdp=‘bc -1’ << END
```

```
$cdp + $dcdp

END‘

done

echo “sunmo par file : $vpicks is ready”

cdp=$cdpmin

while [ $cdp –le $cdpmax]

do

# rm mpicks.$cdp par.$cdp

cdp=‘bc -1 << END

$cdp + $dcdp

END‘

done

# rm unisam.p

### Corrección por NMO

sunmo < CMP par=stkvel.p1 > nmodata

## Graficacion de los CMP Gathers corregidos por NMO (x-windows graphic)

suwind < nmodata j=1 > NMO

sugain < NMO tpow=2 gpow=0.5 NMO_sugain

suxwigb < NMO clip=0.3 \title=“NMO” \windowtitle“NMO” \label1=‘Tiempo (seg)’
label2=‘Distancia (m)’ \wbox=${WIDTH} hbox=${HEICHTOFF2} \xbox=${WIDTHOFF1}
ybox=${HEIGHTOFF2} &

Graficación de cuatro CMP corregidos por NMO (PS)

suwind < nmodata key=cdp min=2000 max=2040 |

supswigb clip=0.1 f2=1994 d2=0.3125 d2num=12.5 \title=“CMP Gathers corregidos por NMO”
\windowtitle=“NMO” \label1=‘Tiempo (seg)’ label2=‘Distancia (m)’ \ > NMO.ps
```

```
## Graficación de cuatro CMP Gathers corregidos por NMO (Image).

suwind < nmodata key=cdp min=2000 max=2040 |

supsimage clip=0.1 f2=1994 d2=0.3125 d2num=12.5 \title="CMP Gathers corregidos por NMO"
\windowtitle="NMO" \label1='Tiempo (seg)' label2='Distancia (m)' \ > NMOIma.ps

## Apilamiento

sustack < nmodata normpow=1.0 > stackdata

## Grafica de la sección apilada (x-windows graphic).

suwind < stackdata j=1 |

suxwigb title="Seccion apilada" clip=0.2 \windowtitlw="Stack" \label1='Tiempo (seg)'
label2='Distancia (m)' \wbox=${WIDTH} hbox=${HEIGHT} \xbox=${WIDTHOFF1}
ybox=${HEIGHTOFF2} &

## Graficación de la sección apilada.

supsimage < stackdata title="Seccion apilada" clip=0.04 \windowtitle="Stack" \label1='Tiempo
(seg)' label2='Distancia (m)' > stackIma.ps
```

NOTAS