



UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

RECONSTRUCCIÓN DE TOPOGRAFÍA Y BATIMETRÍA PARA LA CONSTRUCCIÓN DE PALEODEMS

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN COMPUTACIÓN

LUCAS FELIPE AMION ARENAS

PROFESORES GUÍA:

Fernando Poblete Gómez

Nancy Hitschfeld Kahler

MIEMBROS DE LA COMISIÓN:

Willy Maikowski Correa

Valentin Barriere

Este trabajo fue parcialmente financiado por el proyecto FONDECYT 1231211

SANTIAGO DE CHILE

2025

RESUMEN DE LA MEMORIA PARA OPTAR AL
TÍTULO INGENIERO CIVIL EN COMPUTACIÓN
POR: LUCAS FELIPE AMION ARENAS
FECHA: 2025
PROF. GUÍA: Fernando Poblete Gómez

RECONSTRUCCIÓN DE TOPOGRAFÍA Y BATIMETRÍA PARA LA CONSTRUCCIÓN DE PALEODEMS

La paleogeografía estudia las condiciones geográficas del pasado geológico mediante herramientas como los PaleoDEMs, modelos digitales que reconstruyen el relieve terrestre de épocas remotas. Actualmente, existen programas como GPlates que buscan facilitar el proceso de creación de un PaleoDEM, pero aun así persisten importantes desafíos, en especial en lo que respecta a la batimetría, donde la pérdida corteza oceánica por el proceso de subducción y la formación de nueva corteza en las dorsales oceánicas dificulta la posibilidad de realizar reconstrucciones paleogeográficas fidedignas.

En este trabajo de memoria se trabaja sobre la base del software llamado Terra Antiqua, el cual se trata de un plugin de QGIS especialmente diseñado para la creación y manipulación de mapas paleogeográficos. Este software permite realizar distintas modificaciones sobre un PaleoDEM, como por ejemplo rellenar espacios vacíos en el mismo, modificar el relieve de una zona en particular o unir múltiples PaleoDEMs en una sola imagen de tipo ráster.

El objetivo de esta memoria es ampliar la funcionalidad de este software, incorporando herramientas del programa GPlates, además de un novedoso algoritmo de reconstrucción de batimetría. En particular, se agrega a dicho plugin la capacidad de reconstruir, tanto capas vectoriales como capas de tipo ráster hacia una edad específica, pudiendo descargar todos los archivos de entrada necesarios desde la aplicación misma.

Como resultado de este trabajo, Terra Antiqua se transforma en una herramienta autosuficiente que ofrece a los investigadores de esta área del conocimiento la posibilidad de realizar todas las etapas de la construcción de un PaleoDEM por medio de una interfaz intuitiva. Sin embargo, queda bastante trabajo por hacer, como por ejemplo agregar la capacidad de usar un modelo de rotación propio para las reconstrucciones seleccionando archivos locales o la incorporación de otros algoritmos similares.

*Caminante, no hay camino.
Se hace camino al andar*

Joan Manuel Serrat

Agradecimientos

En primer lugar, quisiera expresar mi profundo agradecimiento a las personas más importantes en mi vida, por siempre apoyarme y creer en mí.

Gracias a mi abuelo Milton, mi mayor modelo a seguir. Tu ejemplo de perseverancia y dedicación, así como tus siempre tan sabios consejos, me inspiraron a salir adelante.

A mi abuela María Eugenia, cuyo apoyo incondicional ha sido una fuente de consuelo y esperanza. Tus amorosas palabras de aliento, y tu preocupación por mi bienestar, me llenan de ánimo para continuar esforzándome.

Gracias a mi madre, Loreto, mi heroína y mi mayor apoyo, tu dedicación y sacrificios son la base de todo lo que he logrado. Eres un ejemplo de fortaleza y resiliencia que me enseña a no rendirme jamás.

A mi hermano Juan Pablo, gracias por tu sentido del humor y por demostrarme que siempre hay razones para sonreír, incluso en los momentos más difíciles.

Quiero también extender mi gratitud a mis profesores guía, Fernando Poblete y Nancy Hitschfeld, quienes desempeñaron un papel fundamental en este proceso. Gracias por su paciencia y su disposición constante a ayudarme en todo lo que necesitara. Su orientación no solo fue clave para completar este proyecto, sino también para mi crecimiento como profesional.

Tabla de Contenido

1. Introducción	1
1.1. Problema y contexto	1
1.2. Objetivos	3
1.2.1. Objetivo general	3
1.2.2. Objetivos específicos	3
1.3. Solución propuesta	4
1.4. Estructura de la memoria	4
2. Estado del Arte	5
2.1. Generación de Topografía	5
2.2. Generación de Agegrids	5
2.3. Conversión a batimetría	7
2.4. Herramientas de software útiles	7
2.4.1. GPlates	7
2.4.2. pyGPlates	8
2.4.3. GPlately	8
2.4.4. PlateModelManager	9
2.4.5. QGIS	9
2.4.6. Terra Antiqua	10
2.4.7. Discusión	10
3. Situación Inicial	11
3.1. Funcionalidades presentes	11
3.1.1. Compile Topo/Bathymetry	12
3.1.2. Set Paleoshorelines	13
3.1.3. Modify Topo/Bathymetry	13
3.1.4. Create Topo/Bathymetry	13
3.1.5. Remove Artifacts	14
3.1.6. Prepare Masks	14
3.1.7. Standard Processing	14
3.2. Diseño y arquitectura del software	15
3.2.1. Algoritmos	15
3.2.2. Interfaz	15
4. Diseño e implementación de las nuevas funcionalidades	18
4.1. Nuevas funcionalidades	18
4.2. Metodología de desarrollo	18

4.2.1.	Reconstrucción de capas vectoriales	19
4.2.2.	Reconstrucción de topografía	21
4.2.3.	Reconstrucción de batimetría	23
4.2.4.	Descarga de archivos	26
4.2.5.	Documentación	26
5.	Validación y prueba de la herramienta	27
5.1.	Prueba de funcionalidades	27
5.1.1.	Reconstruyendo la topografía	27
5.1.2.	Reconstruyendo la batimetría	28
5.1.3.	Integrando las imágenes	29
5.1.4.	Reconstruyendo COBs	30
5.1.5.	Modificando topografía	32
5.2.	Benchmark del algoritmo de batimetría	34
5.2.1.	Desempeño en memoria y tiempo	34
5.2.2.	Validación de los resultados	35
6.	Conclusiones	37
	Bibliografía	38
	Anexo	40
	A. Instrucciones de instalación	40

Índice de figuras

1.1.	Ejemplo de mapa paleogeográfico generado mediante un modelo de rotación con zonas en blanco debido a la pérdida de corteza oceánica.	2
2.1.	Ejemplo de utilización del programa GPlates.	8
3.1.	Imagen de la interfaz de QGIS con la barra de tareas de Terra Antiqua. . . .	11
3.2.	Interfaz del algoritmo “Compile Topo/Bathymetry” a modo de ejemplo. . . .	12
3.3.	Diagrama de clases de Terra Antiqua.	16
4.1.	Interfaz del algoritmo de reconstrucción de capas vectoriales.	19
4.2.	Interfaz del algoritmo de reconstrucción de topografía.	21
4.3.	Interfaz del algoritmo de reconstrucción de batimetría.	24
5.1.	Parámetros usados en la reconstrucción de topografía de la época de 30 Ma. .	28
5.2.	Resultado de la reconstrucción de topografía para 30 Ma.	28
5.3.	Parámetros usados en la reconstrucción de batimetría de la época de 30 Ma. .	29
5.4.	Parámetros usados para la compilación de topografía y batimetría de 30 Ma. .	30
5.5.	Ráster compilado de topografía y batimetría de 30 Ma.	30
5.6.	Parámetros usados para la reconstrucción de los COBs del modelo Müller 2022 a la época de 30 Ma.	31
5.7.	Resultado de la reconstrucción de los COBs del modelo Müller 2022 a la época de 30 Ma.	31
5.8.	Selección del polígono de interés para la modificación de topografía.	32
5.9.	Parámetros para la modificación de la topografía de los Andes a los 30 Ma. . .	33
5.10.	Resultado de la modificación de la topografía de los Andes a los 30 Ma. . . .	33

Índice de Tablas

5.1.	Resultados de benchmark para el algoritmo de generación de batimetría	35
------	---	----

Índice de Códigos

5.1.	Script de Python que compara los resultados de ambos métodos.	35
A.1.	Instrucciones de instalación dentro del archivo README.md	40

Índice de algoritmos

1	Algoritmo de reconstrucción de capas vectoriales	20
2	Algoritmo de reconstrucción de capas tipo ráster	25

1. Introducción

1.1. Problema y contexto

Este trabajo se enmarca dentro de la disciplina llamada paleogeografía. Esta es un área de estudio que busca determinar las condiciones geográficas existentes en la superficie de la Tierra en épocas geológicas pasadas. Uno de los objetivos de esta disciplina es la generación de PaleoDEMs, es decir, modelos de elevación digital que permiten visualizar cómo era el relieve de la Tierra hace millones de años. La construcción de estos modelos requiere de la interacción de diversas disciplinas de Ciencias de la Tierra, como la estratigrafía, la glaciología, el paleomagnetismo, la tectónica, etc. Por esta razón, la paleogeografía resulta ser una disciplina muy compleja.

El proceso de creación de un PaleoDEM, se puede dividir en dos partes principales, la generación de la topografía y la generación de la batimetría de la época en cuestión. A grandes rasgos, la topografía corresponde a la elevación de los terrenos por encima del nivel del mar, y la batimetría hace referencia a la profundidad del relieve por debajo del nivel del mar.

La topografía de una época específica se puede obtener partiendo desde un DEM de la topografía actual y utilizando un modelo tectónico de rotación para rotar cada uno de los bloques corticales presentes hasta su posición correspondiente a la época de estudio. Existe una gran variedad de dichos modelos de rotación publicados por diferentes autores, basados en diferentes evidencias geológicas, que corresponden a diferentes escenarios posibles de lo que podría haber pasado en la Tierra en el pasado distante.

En cuanto a la batimetría tenemos una situación bastante diferente. Esto se debe a la existencia de zonas de subducción producto de la convergencia entre dos placas tectónicas, en donde se está continuamente destruyendo parte de la corteza oceánica al hundirse una placa tectónica debajo de la otra, además de la presencia de dorsales oceánicas producto de la divergencia de placas, en donde tenemos material del manto terrestre subiendo hacia la superficie en forma de magma, el cual se endurece y pasa a formar parte de la corteza. En resumen, tenemos regiones del planeta en donde la corteza oceánica está desapareciendo y otras regiones donde se está continuamente creando nueva corteza. Esto significa que, según estimaciones realizadas por Torsvik et al. (2010), más de la mitad de toda la corteza oceánica existente actualmente tiene una edad de no más de 120 millones de años [1]. En otras palabras, si queremos estudiar épocas anteriores, no podremos ocupar la misma estrategia usada para la reconstrucción de topografía, es decir, la de tomar el relieve de la corteza oceánica actual y rotarlo hasta su posición correspondiente, porque la corteza más antigua ya no existe.

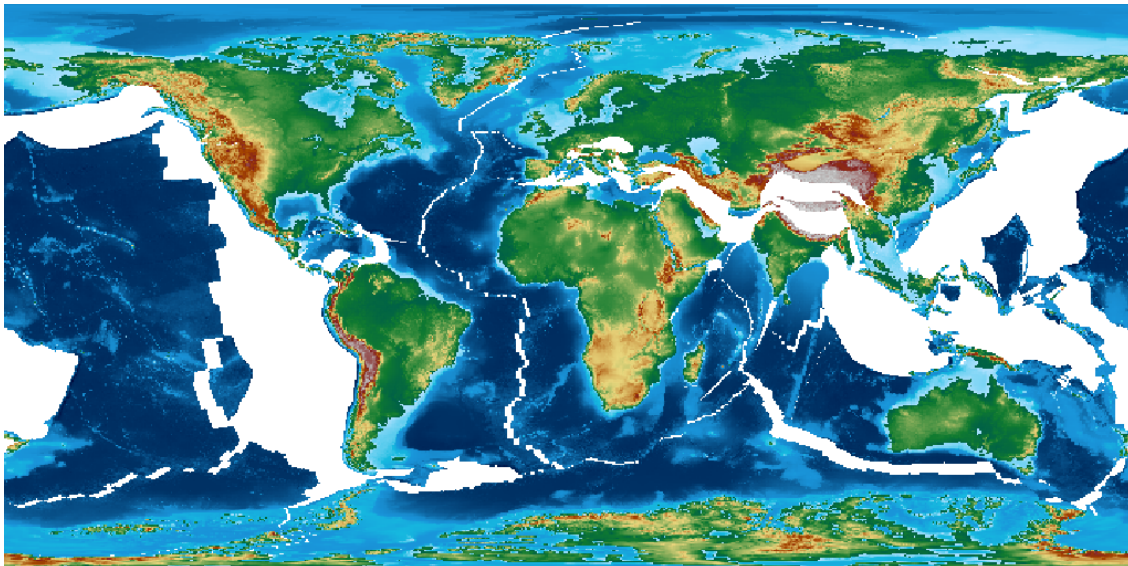


Figura 1.1: Ejemplo de mapa paleogeográfico generado mediante un modelo de rotación con zonas en blanco debido a la pérdida de corteza oceánica.

Al explorar las herramientas de software diseñadas para enfrentar estos desafíos, vemos que ninguna de las que existe actualmente ofrece una solución completa al problema de la generación de un PaleoDEM. Por ejemplo, tenemos el programa GPlates, cuyo propósito es justamente el de llevar a cabo el proceso ya mencionado de aplicar modelos de rotación para reconstruir la posición de las placas tectónicas y bloques corticales. Lamentablemente, este proceso solo permite reconstruir la posición de los diferentes bloques, mas no su paleogeografía, por lo que suele no ser suficiente para generar simulaciones fidedignas de las condiciones de la tierra antigua. Por ejemplo, puede haber evidencia de que en una región en particular existió una gran cadena montañosa que ya no está presente (o viceversa), o que las líneas de costa han cambiado, haciendo que ciertas zonas del mapa que actualmente están por sobre el nivel del mar deban ser sumergidas, etc. En casos como estos nos vemos obligados a recurrir a otro tipo de software para realizar modificaciones adicionales.

Una opción es utilizar un GIS, o Sistema de Información Geográfica. Estos son programas de computador muy completos que contienen una gran variedad de herramientas para trabajar con información geográfica, con los que se puede realizar básicamente cualquier modificación que se desee a un mapa. Existen varios GIS disponibles en el mercado con funcionalidades más o menos equivalentes, pero uno de ellos, llamado QGIS, resulta de especial interés para esta memoria, ya que dentro de su repositorio de plugins existe uno que está especialmente diseñado para realizar reconstrucciones paleogeográficas, llamado Terra Antiqua. Con él se pueden realizar tareas como modificar la elevación de una sección del mapa, delimitada por un polígono específico, aumentando su elevación en el caso de la aparición de una montaña, por ejemplo.

Por otro lado, para la reconstrucción de batimetría no encontramos ningún programa de escritorio que pueda ser de ayuda, no obstante, existen métodos que han sido propuestos en la literatura para llevar a cabo este proceso. Un ejemplo de esto es el trabajo Simon Williams et al. [2] quienes desarrollaron un método para realizar reconstrucciones de la batimetría del planeta que funciona, en cierto modo, de manera contraria a las reconstrucciones de

topografía, ya que se parte desde una época anterior a la que se desea estudiar y se reconstruye hacia adelante en el tiempo, creando nuevos puntos en el mapa a lo largo de las dorsales oceánicas y eliminando dichos puntos al entrar en zonas de subducción. El código asociado a este artículo científico está disponible públicamente en un repositorio de GitHub, en forma de un script escrito en el lenguaje de programación Python. Lamentablemente, este puede ser difícil de usar para usuarios no expertos en programación porque no posee ninguna interfaz de usuario. Cabe mencionar que la generación de la batimetría mediante este método implica en primera instancia la creación de una “Agegrid”, es decir, un mapa de la edad de la roca en lugar de su elevación, pero esta puede ser convertida a batimetría, usando una simple relación lineal entre la edad y la profundidad de la corteza oceánica [3].

1.2. Objetivos

De esta manera, la creación de un mapa paleogeográfico y en particular la creación de un PaleoDEM, requiere de la utilización de una gran variedad de herramientas de software distintas. El objetivo de esta memoria es simplificar este proceso para los investigadores de esta disciplina, de manera que puedan fácilmente crear estas simulaciones para estudiar las condiciones de la tierra en una época específica. Para esto se necesita un software que reúna todas las características necesarias en un solo lugar y que tenga una interfaz intuitiva para que pueda ser usada por usuarios que no tengan conocimientos tan avanzados de programación como para interactuar directamente con las API (Application Programming Interface) de programas como QGIS o GPlates, para la creación de sus propios scripts.

A la luz de lo expuesto anteriormente, el objetivo principal de esta memoria será:

1.2.1. Objetivo general

Diseñar e implementar una herramienta de software capaz de llevar a cabo el proceso de construcción de un PaleoDEM, incluyendo la reconstrucción tanto de la topografía como de la batimetría de la época de estudio. El software implementado debe ser autosuficiente, permitiendo la realización de todo el proceso sin recurrir a herramientas externas.

1.2.2. Objetivos específicos

Entre los objetivos específicos que se espera desarrollar se encuentran los siguientes:

- Permitir la descarga de los archivos necesarios (de tipo vectorial, ráster o modelos de rotación) desde la aplicación misma para que estos no tengan que ser proporcionados por el usuario.
- Permitir la rotación rásteres de topografía, aplicando un modelo tectónico de rotación, como se puede hacer en GPlates.
- Permitir la rotación de archivos vectoriales usando el mismo procedimiento basado en modelos de rotación. Estos archivos pueden servir para definir polígonos que enmarquen las zonas en las que se realizará modificaciones dentro del mapa.
- Permitir la generación de batimetría a partir del procedimiento desarrollado por Simon Williams [2].

- Permitir la visualización en tiempo real de los resultados dentro de la misma aplicación.
- Evaluar la utilidad de las funcionalidades añadidas en un caso de uso real.

1.3. Solución propuesta

Para implementar la solución, se plantea extender el código de Terra Antiqua. Al ser este un plugin de QGIS, otorga acceso a toda la API de esta poderosa aplicación, que ya tiene implementadas muchas funciones útiles a la hora de manipular mapas que se necesitarán en este proyecto. Además, este plugin está escrito en el lenguaje de programación Python, lo que significa que será relativamente sencillo incorporar funcionalidades de la API de Python de GPlates, llamada pyGPlates, y el método de Simon Williams [2], que también está implementado en Python.

1.4. Estructura de la memoria

Este documento está estructurado de la siguiente manera:

- En el capítulo 2 “Estado del Arte”, se describen algoritmos y herramientas disponibles en la actualidad que pueden ser de ayuda para el proceso de reconstrucción paleogeográfica.
- En el capítulo 3 “Situación Inicial”, se describe en detalle el estado en el que se encontraba inicialmente el plugin Terra Antiqua previo a los cambios realizados en esta memoria.
- En el capítulo 4 “Desarrollo”, se explica la metodología del trabajo realizado y los algoritmos implementados.
- En el capítulo 5 “Resultados”, se presenta un ejemplo de ejecución en donde se ponen a prueba las nuevas funcionalidades desarrolladas.
- Y, finalmente, en el capítulo 6 se presentan las conclusiones del trabajo realizado, incluyendo algunas proyecciones acerca de trabajos futuros a realizar.

2. Estado del Arte

En este capítulo se pretende explicar cada uno de los pasos que comprenden la realización de una reconstrucción paleogeográfica, así como las herramientas de software y algoritmos que pueden ser de utilidad en cada uno de ellos.

2.1. Generación de Topografía

La reconstrucción paleogeográfica se puede dividir en topografía y batimetría. En el caso de la topografía, podemos rotar las placas tectónicas para llevar la topografía actual a su posición correspondiente al periodo estudiado. Esto se puede realizar aplicando los ya mencionados modelos tectónicos de rotación. Estos modelos están basados en el teorema de rotación de Euler, el cual establece que cualquier movimiento (o combinación de movimientos) de una placa rígida sobre la superficie de una esfera se puede reducir a una única rotación alrededor de un eje particular, que se denomina polo de Euler [4].

Esto permite describir el movimiento relativo entre dos placas tectónicas con tan solo 4 números: 3 ángulos que definen el eje de rotación o polo de Euler correspondiente y uno más que indica la cantidad de movimiento alrededor de dicho eje. Comúnmente se elige una placa que actuará como ancla y se considera fija, describiendo el movimiento relativo de las demás placas con respecto a ella. Toda esta información se almacena generalmente en un archivo, o una serie de archivos, en formato `.rot` que contienen la información de los movimientos relativos entre cada una de las placas tectónicas. Estos datos generalmente se obtienen mediante la extrapolación de las velocidades relativas observadas en la actualidad.

Además de los archivos `.rot`, cada modelo de rotación debe incluir una serie de archivos vectoriales (polígonos) que definen básicamente las fronteras entre las distintas placas. Ambos tipos de archivo son necesarios para la reconstrucción, ya que el mapa inicial tiene que ser cortado según estos polígonos para poder rotarlo. Por ejemplo, uno de los modelos de rotación más usados actualmente es el modelo Müller 2022. Es uno de los más completos, ya que permite rebobinar incluso hasta los 1000 millones de años atrás [5].

2.2. Generación de Agegrids

La generación de batimetría, es decir, la elevación o relieve de la corteza oceánica, requiere de un enfoque diferente. Esto se debe a la existencia de los procesos de divergencia y convergencia de las placas tectónicas. La divergencia consiste en la situación en la que dos placas tectónicas contiguas se están separando entre sí paulatinamente año a año. Esto produce lo que se llama una dorsal oceánica en el punto de unión entre ambas placas, que es un lugar en

el que el material del manto terrestre sube a la superficie, donde se emplaza y pasa a formar parte de la corteza oceánica. Mientras mayor sea la tasa de separación o movimiento relativo entre ambas placas, más rápida será la velocidad de generación de nueva corteza.

La convergencia de placas tectónicas, por su parte, consiste en el proceso por el cual dos placas tectónicas adyacentes se mueven una hacia la otra, comúnmente produciendo que una de ellas se hunda debajo de la otra, proceso conocido como subducción. Generalmente, se trata de una placa oceánica, las cuales suelen ser más delgadas y densas, hundiéndose debajo de una placa continental. En este caso, tenemos el problema mencionado en el capítulo anterior, en el cual la roca de la placa oceánica que se está hundiendo desaparece paulatinamente, por lo cual perdemos información sobre el pasado geológico de la Tierra.

El artículo de Simon Williams et al. [2] enfrenta esta problemática por medio del siguiente algoritmo:

- Primero, el algoritmo requiere como input un modelo tectónico como los mencionados anteriormente. Este se usará para saber los puntos en los que se encuentran la separación entre el océano y los continentes, así como también para saber donde están las zonas de subducción y las dorsales oceánicas que, actúan como las zonas en donde se destruye o se crea nueva corteza oceánica, respectivamente. Los límites entre las placas se codifican siguiendo las definiciones del lenguaje de marcado de GPlates (GPML). Se usan los límites de placa codificados como `MidOceanRidge` para definir las ubicaciones de la nueva generación del fondo marino y los límites codificados como `SubductionZone` para definir las zonas en que irá desapareciendo la corteza.
- Como primer paso del algoritmo, se tiene que generar la Agegrid para la época elegida como el tiempo de inicio de la reconstrucción. Esto se realiza asumiendo un spreading rate o tasa de separación, constante para todo el mapa, que en el paper es 35 mm/año. Luego, a cada punto del mapa, se le asigna un valor de edad de acuerdo a su distancia hacia la dorsal oceánica más cercana, usando el valor de spreading rate ya mencionado. Para placas que no tienen una dorsal oceánica asociada, se asume un valor constante 5000 km de distancia y se calcula la edad a partir de él. Todas estas suposiciones significan que los valores obtenidos para los primeros años de la reconstrucción no serán muy acertados, sin embargo, a medida de que esta avanza, la corteza inicial irá desapareciendo, dando paso a la nueva corteza creada a partir del algoritmo. Por esta razón se recomienda en el paper elegir un tiempo inicial que sea por lo menos 50 millones de años anterior a la edad final de reconstrucción deseada.
- En cada paso del algoritmo, lo primero que se hace es generar nuevos puntos a lo largo de las dorsales oceánicas con edad inicial cero.
- Luego, para todos los demás puntos en el mapa, se incrementa su edad según el intervalo de tiempo transcurrido y se rotan según lo indicado por el modelo tectónico que se está utilizando.
- A continuación, para cada uno de los puntos en el mapa, se revisa si estos han atravesado alguna zona de subducción. Si es así, estos puntos son eliminados permanentemente.
- Finalmente, si no se ha llegado todavía a la época establecida como tiempo final de la reconstrucción, se repite todo el proceso.

Cabe destacar que este algoritmo no solo permite generar una imagen del tiempo final, sino que también genera mapas para cada punto intermedio de la reconstrucción, aunque esto no es usado en esta memoria.

2.3. Conversión a batimetría

Una vez obtenida la Agegrid correspondiente a la edad de reconstrucción, hay que recordar que lo que tenemos es una imagen de tipo ráster que define un valor de edad de la roca para cada punto del espacio. Sin embargo, lo que se busca es un valor de elevación del lecho marino en su lugar, para tener así un verdadero mapa de la batimetría de la edad que se está estudiando.

Por suerte, existe una relación lineal entre la edad de la roca y la elevación del lecho marino que se pudo extraer a partir de observaciones de anomalías magnéticas en el océano [3]. Los minerales ferromagnéticos contenidos en las rocas son capaces de registrar la orientación del campo magnético terrestre en el momento de su formación. Esto permite a los científicos determinar la edad del lecho marino con bastante precisión. Usando estos datos se llegó a una fórmula que se presenta a continuación:

$$\begin{cases} Z = 2620 + 330\sqrt{A}, & \text{si } A < 90 \\ Z = 5750, & \text{si } A > 90 \end{cases} \quad (1)$$

Ecuación que permite convertir la edad de la corteza oceánica (A), medida en millones de años, a profundidad en metros (Z).

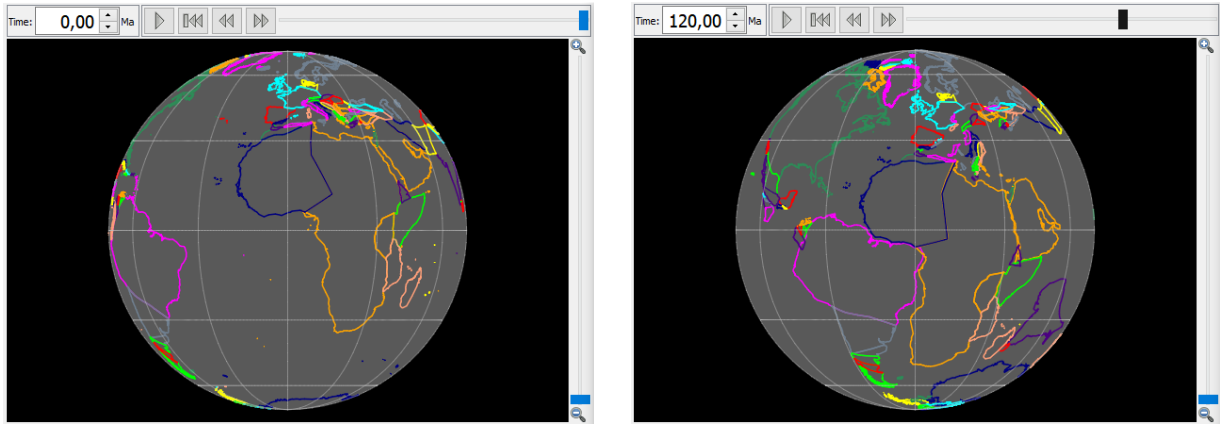
Esta misma relación se usa dentro del plugin Terra Antiqua, en particular en su herramienta de “Convert to Bathymetry” dentro del menú llamado “Standard Processing” (ver sección 3.1.7).

2.4. Herramientas de software útiles

2.4.1. GPlates

Para realizar el procedimiento descrito en la sección 2.1 se puede utilizar el software llamado GPlates. Este software de código abierto es usado para la visualización interactiva de la tectónica de placas [6] y es desarrollado por la empresa EarthByte.

La interfaz de GPlates permite cargar el o los archivos `.rot` junto con los archivos vectoriales que definen un modelo de rotación simplemente arrastrándolos y soltándolos al interior del mapa. Luego, haciendo uso de un slider en la parte superior de la interfaz, se puede modificar el mapa en tiempo real, rebobinando hasta el periodo deseado.



(a) Líneas de costa actuales.

(b) Líneas de costa reconstruidas a su posición de hace 120 millones de años.

Figura 2.1: Ejemplo de utilización del programa GPlates.

Aparte de poder reconstruir archivos vectoriales, GPlates también permite la reconstrucción de archivos tipo ráster, cortándolos según los límites definidos por un archivo vectorial, y rotando cada trozo obtenido de acuerdo con el movimiento de la placa correspondiente.

2.4.2. pyGPlates

GPlates también ofrece una API o Application Programming Interface llamada pyGPlates, la cual nos otorga acceso a toda la funcionalidad del programa dentro del lenguaje de programación Python. Esto permite crear nuestros propios scripts, lo que en general entrega un mayor grado de flexibilidad y personalización que la aplicación de escritorio.

El componente más importante de esta librería se trata de la función `reconstruct()`, la cual permite reconstruir cualquier tipo de rasgo geológico hacia atrás en el tiempo usando un modelo de rotación.

2.4.3. GPlately

También tenemos GPlately, que es una librería de Python escrita sobre pyGPlates que añade algunas funcionalidades interesantes, como por ejemplo la capacidad de reconstruir puntos, calcular velocidades y crear gráficos de líneas de flujo o de trayectorias de movimientos asociados a las reconstrucciones.

En particular, la clase llamada `Raster` que forma parte de GPlately será de vital importancia para esta memoria. Contiene métodos como `reconstruct()`, que automatiza todo el proceso de reconstrucción de una imagen ráster que ya fue descrito, incluyendo el proceso de cortar la imagen según la capa vectorial correspondiente, mover los trozos obtenidos en función del modelo de rotación utilizado, y `resample()` que permite cambiar la resolución de la imagen en cuestión, lo que puede ser útil para disminuir el tamaño del archivo y el tiempo que tomará el proceso de reconstrucción en situaciones en que no se requiera una precisión tan elevada.

2.4.4. PlateModelManager

PlateModelManager es otra librería de Python, también desarrollada por EarthByte, que permite descargar modelos tectónicos de rotación, y todos sus archivos vectoriales asociados, desde su repositorio oficial. Para poder usarla hay que definir un directorio llamado `data_dir` donde se descargarán todos los archivos. Cada modelo se descarga dentro de su propia carpeta, dentro de la cual tendremos una carpeta llamada Rotations con todos los archivos `.rot` asociados al modelo, además de una carpeta para cada capa vectorial descargada. Las capas vectoriales disponibles varían dependiendo del modelo, pero todas ellas son útiles para el proceso de reconstrucción de alguna manera. Algunas de las más importantes son:

- COBs: Su nombre significa Continental Ocean Boundaries. Indican el límite entre las placas de corteza oceánica y las de corteza continental. Se utilizan para cortar los archivos ráster durante los procesos de reconstrucción, tanto de topografía como de batimetría.
- Coastlines: También señala el contorno de los continentes, pero poniendo el límite el punto en que el relieve pasa de estar sobre el nivel del mar a estar bajo el nivel del mar.
- Static Polygons: Similar a los COBs, pero incluye además la corteza oceánica, dividida en polígonos finos que delimitan zonas donde la edad de la roca es similar. Estos polígonos van desapareciendo al reconstruir hacia atrás en el tiempo, por lo que no se pueden usar para reconstrucciones de batimetría de manera confiable.
- Topologies: Contiene información más específica como la ubicación de las zonas de subducción y las dorsales oceánicas, por lo que es crucial para el algoritmo descrito en la sección 2.2.

2.4.5. QGIS

Como se mencionó anteriormente, el proceso de reconstrucción que genera la topografía y batimetría iniciales no es suficiente para la construcción de un PaleoDEM. Hay situaciones en donde hay que modificar manualmente la topografía o batimetría de una región específica, como se explicó más arriba. Para esto puede ser de gran utilidad el software llamado QGIS. Este es uno de los Sistemas de Información Geográfica (GIS) más utilizados, es gratis y de código abierto [7]. Nos otorga una gran variedad de herramientas para la manipulación, tanto de archivos tipo ráster en formato GeoTiff o NetCDF, como archivos vectoriales en formato Shapefile o GPML. Estos archivos pueden ser cargados en la aplicación, simplemente arrastrándolos y soltándolos sobre el mapa, y serán agregados como una serie de capas, las cuales se muestran en un panel en la parte inferior izquierda de la aplicación, donde pueden ser activadas o desactivadas haciendo clic en un pequeño checkbox al lado de cada una. Esto controla cuáles capas serán visibles y cuáles no. Cada una de las capas añadidas al mapa puede luego ser modificada utilizando la gran variedad de herramientas disponibles en la barra de tareas de la parte superior de la interfaz. Finalmente, el mapa resultante puede ser exportado en una gran variedad de formatos.

QGIS es un software muy completo e incluye por defecto una cantidad inmensa de funcionalidades para trabajar con información geográfica. Aun así, QGIS cuenta con una gran biblioteca de complementos o plugins que otorgan funcionalidades más específicas. Estos pueden ser instalados desde la misma interfaz del programa o descargados desde la página web <https://www.qgis.org/>:

//plugins.qgis.org/plugins/ en formato zip y extraídos dentro de la carpeta en donde QGIS espera encontrar los archivos de plugins, es decir, C:\{USER}\AppData\Roaming\QGIS\QGIS3\profiles\default\python\plugins en Windows o /home/{USER}/.local/share/QGIS3/profiles/default/python/plugins en Linux.

Estos plugins de QGIS pueden ser escritos en el lenguaje de programación Python. El equipo de QGIS publica una guía sobre cómo desarrollar estos plugins llamada PyQGIS Developer Cookbook que se encuentra en la siguiente página: https://docs.qgis.org/3.34/en/docs/pyqgis_developer_cookbook/. Además, hay dos complementos oficiales que pueden ser de gran utilidad a la hora de desarrollar plugins de QGIS. Estos son Plugin Reloader, que permite reiniciar un plugin específico para que se reflejen los últimos cambios realizados en el código fuente sin tener que reiniciar la aplicación en su totalidad, y First Aid, un debugger que permite saber con exactitud en qué parte del código hay un problema en caso de que un plugin arroje un error.

2.4.6. Terra Antiqua

Uno de los complementos de Python disponibles en el repositorio de plugins de QGIS es el llamado Terra Antiqua, que fue la base sobre la cual se trabajó durante este trabajo de título. Este plugin, desarrollado por Jovid Aminov, Diego Ruiz y Guillaume Dupont-Nivet, está especialmente diseñado para realizar reconstrucciones paleogeográficas.

Algunas de las funcionalidades presentes en este software son la capacidad de unir dos DEMs en uno solo, modificar el relieve de una zona específica del mapa, entre otras. Una descripción más detallada de este software se encuentra en el siguiente capítulo de este documento.

2.4.7. Discusión

Se decidió trabajar sobre la base de este plugin de QGIS en lugar de desarrollar una herramienta nueva desde cero, ya que de esta manera se evita agregar una más a la lista de herramientas necesarias para realizar cada uno de los pasos de una reconstrucción paleográfica completa, lo que no sería conducente al objetivo inicial de conseguir una herramienta autocontenida que agilice el proceso.

Si bien existen múltiples Sistemas de Información Geográfica con funcionalidades más o menos equivalentes, se eligió trabajar específicamente con QGIS debido a que es un software de código abierto y fácilmente extensible por medio de su sistema de plugins que funcionan gracias al intérprete de Python que viene incluido de forma nativa con la aplicación. Esto significa que se puede fácilmente incorporar la funcionalidad de GPlates por medio de sus APIs escritas también en Python. Además, la presencia de un plugin de QGIS ya existente que está especialmente diseñado para la reconstrucción paleogeográfica significa que no es necesario crear un plugin desde cero, teniendo muchas funcionalidades útiles ya disponibles que no será preciso reimplementar.

3. Situación Inicial

A continuación se describe detalladamente la situación inicial en la que se encontraba el código de Terra Antiqua antes de los cambios realizados como motivo de esta memoria. Esto incluye las funcionalidades que posee, así como la forma en que está programado.

3.1. Funcionalidades presentes

Las distintas funcionalidades de Terra Antiqua se presentan como íconos en una barra de herramientas separada en la parte superior de la interfaz de QGIS. Al hacer clic en cualquiera de estos íconos se abre un cuadro de diálogo que contiene los parámetros de entrada del algoritmo en el lado izquierdo, que permiten personalizar la ejecución del mismo, y un texto de ayuda en el lado derecho que explica el propósito de la herramienta en cuestión y describe con bastante detalle el significado de cada uno de los parámetros de entrada. Algunos de los algoritmos contienen “parámetros avanzados” que están en un menú desplegable que se encuentra oculto por defecto. En la parte inferior de dicho diálogo, también tenemos una barra de progreso que mostrará cuánto falta para la completación del algoritmo, un botón “Help” que abre la página de documentación del plugin (<https://jaminzoda.github.io/terra-antiqua-documentation/>), un botón que permite cerrar el diálogo, un botón con la palabra “Cancel” que permite cancelar el algoritmo en plena ejecución y, finalmente, el botón “Run” que inicia la ejecución del algoritmo. Mientras el algoritmo está en ejecución, veremos un cuadro de texto con mensajes de log que registran las acciones que se van llevando a cabo.

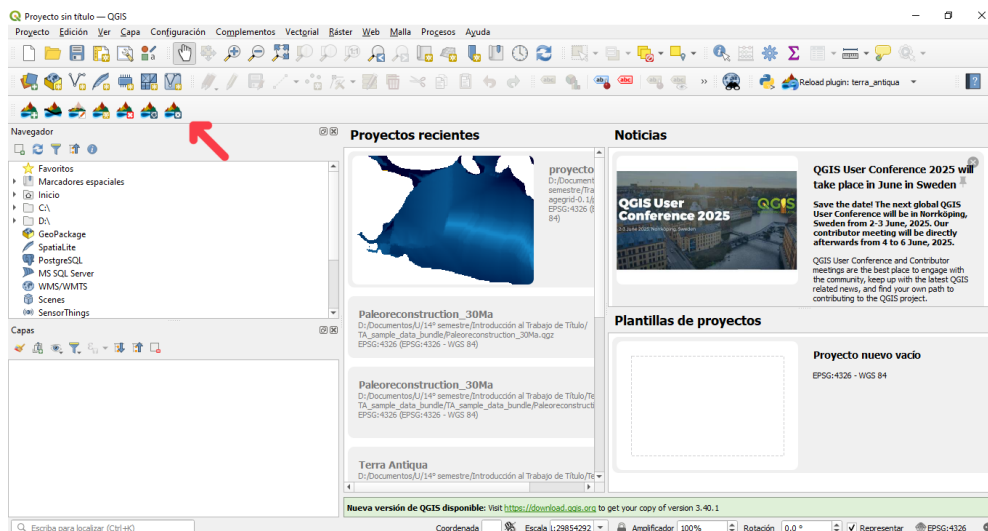


Figura 3.1: Imagen de la interfaz de QGIS con la barra de tareas de Terra Antiqua.

En general, al finalizar la ejecución de cualquiera de estos algoritmos, el resultado será añadido al proyecto como una nueva capa, ya sea vectorial o de tipo ráster, dependiendo del algoritmo en cuestión. Casi todos los algoritmos tienen un parámetro en común llamado “Output file path” que define la ubicación y nombre del archivo que se creará como resultado de la ejecución. Si este parámetro se deja en blanco, el resultado será guardado en la carpeta temporal del usuario `C:\{USER}\AppData\Local\Temp` en Windows o `/tmp/` en Linux y conservará el nombre por defecto que depende del algoritmo.

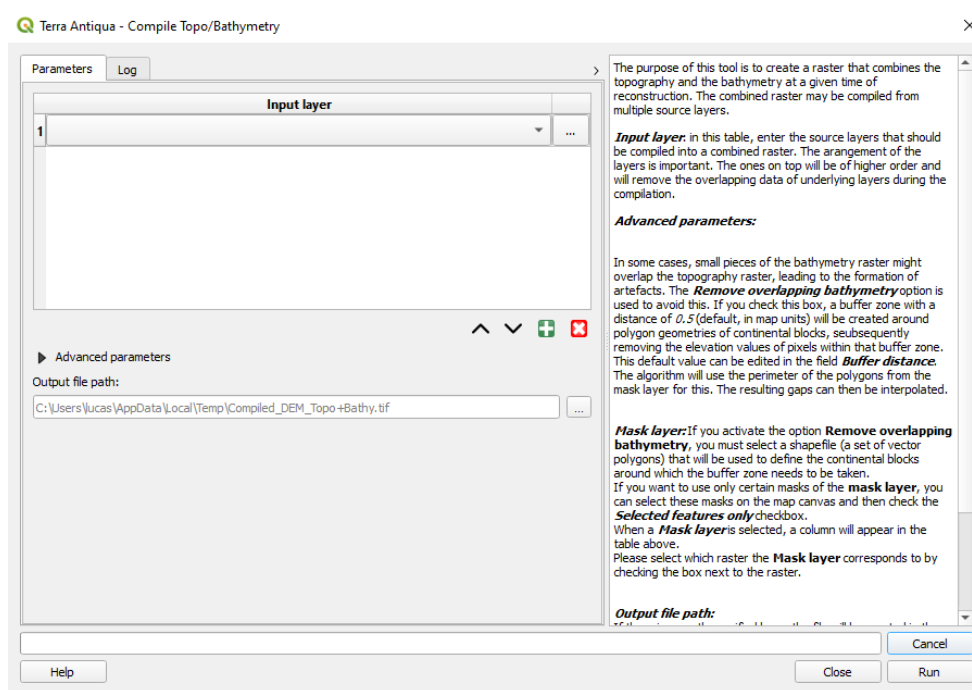


Figura 3.2: Interfaz del algoritmo “Compile Topo/Bathymetry” a modo de ejemplo.

A continuación se describe de manera detallada cada una de las funcionalidades o algoritmos presentes disponibles en Terra Antiqua inicialmente:

3.1.1. Compile Topo/Bathymetry

Esta funcionalidad, como su nombre lo indica, permite unir varias imágenes ráster en una sola. Generalmente, esto se usará para unir la topografía de una época en particular con la batimetría correspondiente. Para usar esta función simplemente hay que seleccionar las dos o más capas que van a ser unidas y hacer clic en el botón “Run”. Los archivos ráster de entrada pueden ser seleccionados de entre las capas ya añadidas al proyecto, pero también se da la opción de seleccionar archivos dentro del computador del usuario, los cuales deben estar en formato GeoTiff, NetCDF, PNG o JPG, y estos serán añadidos al proyecto. Al finalizar la ejecución, el resultado es añadido al mapa como una nueva capa llamada “Compiled_DEM_Topo+Bathy”. Aparte de los archivos de entrada, este algoritmo también posee un parámetro avanzado llamado “Remove overlapping Bathymetry” que permite seleccionar un archivo vectorial que defina los límites entre la topografía y la batimetría para cortar las imágenes y evitar solapamientos, ya que estos pueden ocasionar la aparición de artefactos en la imagen. Si existe solapamiento entre las imágenes ráster que se van a unir, entonces el orden en que fueron seleccionados importa: el primero tendrá precedencia y reemplazará el contenido de las capas siguientes.

3.1.2. Set Paleoshorelines

El propósito de esta funcionalidad es corregir un PaleoDEM según una serie de polígonos que definen las Paleocostas de la época de estudio. Como se describió anteriormente, al seguir el procedimiento estándar de reconstrucción de topografía usando el software GPlates, lo que se hace es simplemente rotar los polígonos actuales hasta su posición hace millones de años. Sin embargo, hay situaciones en que sabemos que la forma de algún continente ha cambiado a lo largo del tiempo, es decir, una zona que ahora es océano antes no lo era o viceversa. Eso es lo que a grandes rasgos permite corregir esta herramienta, elevando zonas que en la actualidad forman parte del mar o hundiendo zonas que solían estar por debajo del nivel del mar. Aparte del archivo ráster inicial y la capa vectorial que define las paleocostas, este algoritmo también recibe algunos otros parámetros que controlan cómo se determinarán los valores de elevación y/o profundidad de las zonas sumergidas o emergidas por el algoritmo. Tenemos el parámetro “Modification Mode” que define cómo se van a tratar los valores de estas zonas. Si se elige “Interpolation” como valor para este parámetro, entonces los valores dentro de zonas emergidas/sumergidas serán eliminados y se calcularán nuevos valores por medio de interpolación lineal entre los valores más cercanos. Si se elige “Rescaling”, entonces se conserva el relieve inicial de la zona modificada, pero reescalado a la nueva profundidad o elevación que se debe definir manualmente. Al finalizar el algoritmo, el resultado es añadido al proyecto como una nueva capa llamada “PaleoDEM_Paleoshorelines_set” aunque este nombre puede ser cambiado por el usuario, tanto en esta funcionalidad como en las otras.

3.1.3. Modify Topo/Bathymetry

Esta funcionalidad permite modificar los valores de elevación de un PaleoDEM dentro de un área delimitada. El área cuyos valores serán modificados debe ser definida con una máscara contenida en un archivo vectorial. Los inputs de este algoritmo son la imagen ráster que se va a modificar, el archivo o capa vectorial que define la zona del mapa cuya elevación va a ser modificada y el modo de modificación. Existen dos modos disponibles. En el primero se define un valor máximo y mínimo de elevación y los valores ya existentes dentro del área en cuestión serán reescalados linealmente para alcanzar dichos valores. La otra opción es definir una fórmula manualmente que indique cómo modificar los valores de altura iniciales. En caso de usar una fórmula también se pueden definir un máximo y un mínimo de elevación.

3.1.4. Create Topo/Bathymetry

Similar a la funcionalidad anterior, pero esta permite crear topografía o batimetría desde cero en lugar de modificar la ya presente. Al igual que en el caso de modificación de topo/-batimetría, en este caso también se requiere como archivos de entrada la imagen ráster que se va a modificar y una capa vectorial que define el polígono dentro del cual se realizará la modificación. En este caso también tenemos la opción de definir un valor máximo y mínimo de elevación, pero tenemos además parámetros específicos que dependen de la característica geográfica que se busca crear, ya sea montaña o mar. En el caso de las montañas, tenemos parámetros como el porcentaje de rugosidad y el ancho de su pendiente, y para los mares tenemos los atributos “Shelf Width” que corresponde al ancho de la zona menos profunda alrededor del mar que se va a crear y “Width of the continental slope” que se refiere al ancho de la pendiente de la corteza continental circundante.

3.1.5. Remove Artifacts

Esta funcionalidad permite eliminar artefactos o errores producidos por la utilización de las otras herramientas del plugin. Al hacer clic en ella, aparece un mensaje explicando cómo se usa. Al presionar el botón de aceptar en este mensaje, el cursor cambiará de forma a una cruz y se podrá dibujar un polígono en el mapa usando el mouse. El algoritmo borrará los artefactos dentro del área delimitada.

3.1.6. Prepare Masks

Similar a la funcionalidad de compilación de topo/batimetría esta permite combinar varias capas vectoriales en un solo Shapefile para que sea más fácil trabajar con él. Sus parámetros de entrada son las capas que se van a unir. Para cada una de las capas que se unirán, se puede definir un atributo llamado categoría que puede tomar los valores de “Coastline”, “Continental Shelf” o “Shallow Sea”, aunque también se puede especificar un valor personalizado. Este atributo tiene como propósito identificar el origen del polígono correspondiente. Al igual que en el caso de la compilación de topo/batimetría, el orden en que se seleccionan las capas es importante porque las que estén más arriba en la lista van a tomar preferencia y, si existe solapamiento, la geometría superpuesta será eliminada en las capas posteriores.

3.1.7. Standard Processing

Esta última herramienta contiene un menú con una serie de funcionalidades misceláneas de menor importancia, dentro de las cuales tenemos:

1. **Fill gaps:** Permite llenar espacios vacíos en un archivo ráster. Se puede usar interpolación o rellenar con un valor fijo.
2. **Copy/Paste raster:** Esta herramienta simplemente copia los valores de un archivo ráster a otro. Se puede usar una capa vectorial para definir la zona que se va a copiar en lugar de copiar todo el ráster.
3. **Smooth raster:** Suaviza una imagen ráster reduciendo cambios abruptos de elevación. Solo se puede usar en una imagen sin espacios vacíos, por lo que se recomienda usarlo después de la herramienta “Fill gaps”. Contiene dos tipos de suavizado que se pueden aplicar: filtro Gaussiano y filtro uniforme.
4. **Isostatic compensation:** Al trabajar con archivos ráster de topografía, en general, estos se presentan en dos tipos: Ice Topography y Bedrock Topography. Uno de ellos incluye la elevación debida al hielo en los casquetes polares y el otro no. Esta herramienta compara un archivo ráster de Bedrock Topography con otro de Ice Topography para obtener la diferencia de elevación entre ambos que corresponde a la cantidad de hielo en las regiones polares. Esto permite eliminar un cierto porcentaje de esa cantidad de hielo detectada.
5. **Set new sea level:** Modifica el nivel del mar desplazando los valores de elevación de todo el mapa verticalmente por un mismo valor.
6. **Calculate bathymetry:** Permite convertir una Agegrid, es decir, una imagen ráster con valores de edad de la corteza, en batimetría, es decir, valores de profundidad de la corteza. Para esto se ocupa la fórmula mencionada en la sección 2.3.

7. **Change map symbology:** Cambia la forma en que se muestra un archivo ráster dentro del mapa, permitiendo elegir entre una lista de paletas de colores predeterminadas.

3.2. Diseño y arquitectura del software

El código de Terra Antiqua está escrito en Python, al igual que la mayoría de plugins de QGIS, y se encuentra dividido en dos carpetas principales, `core` y `gui` que contienen, por así decirlo, el backend y el frontend de la aplicación.

3.2.1. Algoritmos

Dentro de la carpeta `core` encontramos la implementación de los algoritmos asociados a cada una de las funcionalidades descritas en la sección anterior. Todos ellos se encuentran en archivos separados y heredan de una clase en común llamada `TaBaseAlgorithm`, dentro del archivo `base_algorithm.py`. En general, estos objetos solo modifican el método `run` conservando todo el resto de funciones de la clase padre. También dentro de esta carpeta encontramos el archivo `algorithm_provider.py` que define la clase `TaAlgorithmProvider` que funciona como un wrapper alrededor de un algoritmo para manejar fácilmente su funcionamiento desde fuera y contiene funciones como `load()`, `start()`, `stop()`, entre otras, que permiten controlar el ciclo de vida de un algoritmo.

Adicionalmente, dentro de la carpeta `core` se encuentra el archivo `logger` que define la clase `TaFeedback` encargada de imprimir los mensajes informativos o de error que se van generando a medida que se ejecuta un algoritmo. Estas funciones son accesibles dentro de los algoritmos por medio de su atributo `self.feedback`. Por último, también tenemos un archivo llamado `utils.py` que contiene funciones de propósito variado que son utilizadas por algunos de los algoritmos.

3.2.2. Interfaz

Por otro lado, tenemos la carpeta `gui` en la que se encuentra todo lo relacionado con la interfaz de usuario de la aplicación. Similar a lo que ocurre con la carpeta `core` tenemos un archivo por cada algoritmo, cada uno de los cuales define el cuadro de diálogo que se mostrará al presionar el botón correspondiente. Todos ellos utilizan widgets provenientes de la librería QT para generar cada uno de los elementos de la interfaz y heredan de una misma clase llamada `TaBaseDialog`, definida dentro del archivo `base_dialog.py`. Esta clase contiene un método llamado `addParameter()` el cual recibe un widget (clase `QWidget`) y un título o label y es el que se usa para agregar cada uno de los parámetros de entrada de los algoritmos. Existen otras tres variantes de este método que tienen funcionalidades más específicas, a saber, `addAdvancedParameter()`, que permite añadir un parámetro que aparecerá dentro del menú de parámetros avanzados que está oculto por defecto, `addVariantParameter()`, que añade parámetros que se esconden dependiendo del valor de otro parámetro y `addMandatoryParameter()`, que permite añadir un parámetro cuyo valor es esencial para la ejecución del algoritmo, por lo que mostrará un mensaje de error si no está definido. También dentro de esta carpeta existe un archivo llamado `widgets.py` en donde se definen una serie de widgets personalizados que heredan de los ya presentes en la librería QT.

Como se puede ver, todas las funcionalidades presentes en Terra Antiqua tienen una representación dentro de la carpeta **core** en forma de un algoritmo y una representación dentro de la carpeta **gui** en forma de un diálogo. Cada algoritmo guarda una referencia al diálogo correspondiente en su atributo **self.dlg** lo que permite acceder a los valores de los parámetros de entrada ingresados por el usuario dentro de un algoritmo dado. Esta estructura se puede visualizar en la figura 3.3.

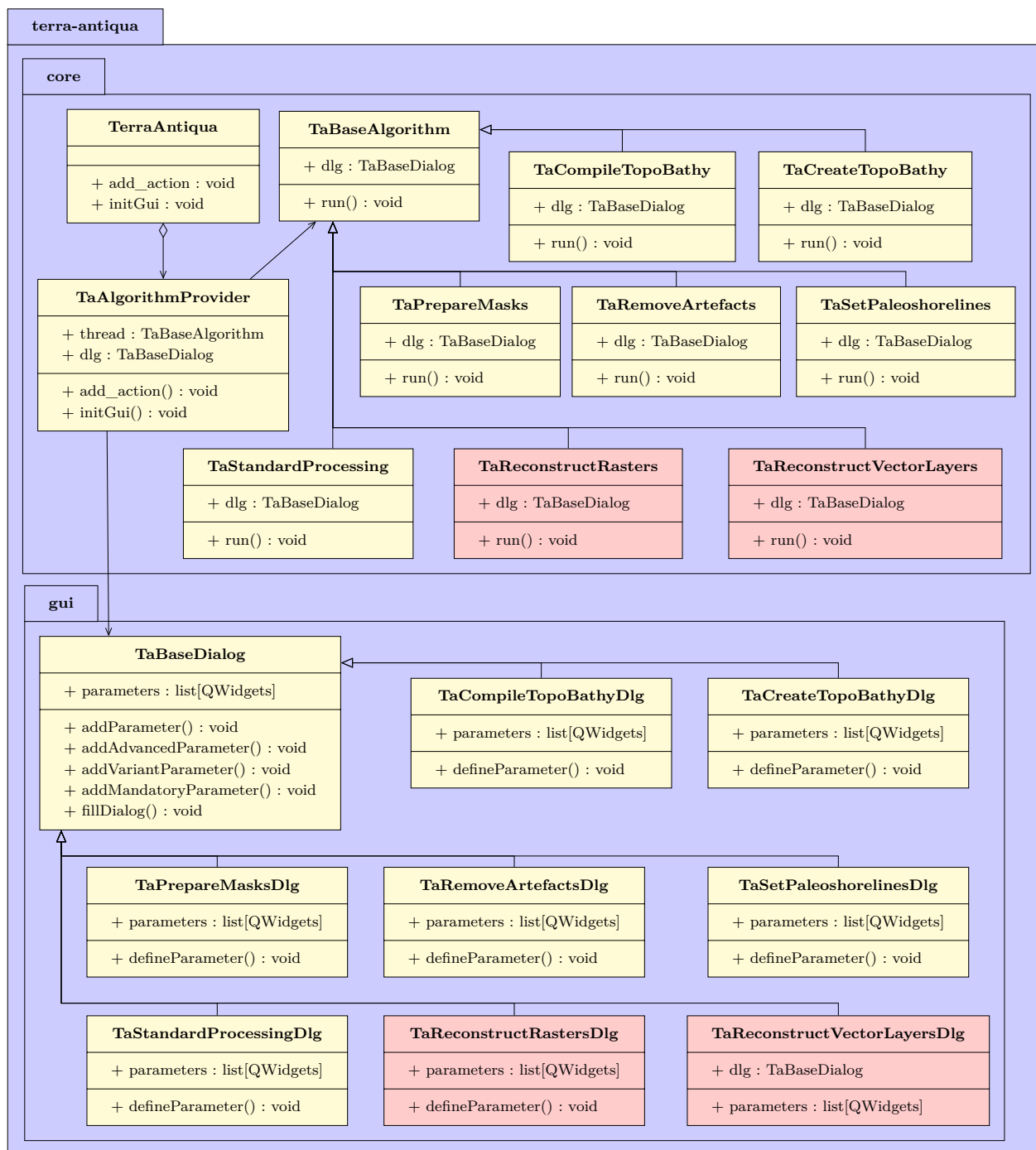


Figura 3.3: Diagrama de clases de Terra Antiqua. Las clases agregadas durante este trabajo se representan en color rojo.

Otros componentes no tan importantes de la aplicación son la carpeta `help_text` que contiene los textos de ayuda que se muestran en el lado derecho de la ventana en cada una de las herramientas, los cuales están escritos en HTML, y la carpeta llamada `resources` en donde se almacenan los archivos binarios como los íconos que forman parte de la interfaz. Estos últimos son compilados en un archivo `resources.py` que contiene la información de todos ellos en un solo lugar y permite su fácil importación en otros archivos.

Finalmente, tenemos el archivo `terra_antiqua.py` que es el archivo principal que une todos los componentes antes mencionados. Utiliza una función llamada `add_action()` para unir cada algoritmo con su diálogo correspondiente y añadirlos uno por uno a la interfaz. Todo esto ocurre dentro de la función llamada `initGui()`.

4. Diseño e implementación de las nuevas funcionalidades

4.1. Nuevas funcionalidades

Las nuevas funcionalidades que se agregaron a Terra Antiqua durante este trabajo son tres:

- **Reconstrucción de capas vectoriales:** Esta funcionalidad permite rotar un archivo vectorial (como Coastlines, Static Polygons, etc.) usando un modelo de tectónico específico, aplicando las funciones de pyGPlates.
- **Reconstrucción de topografía:** Esta funcionalidad tiene la finalidad de rotar un archivo ráster de topografía usando un modelo tectónico de rotación de manera similar a la funcionalidad de reconstrucción de capas vectoriales, pero aplicando las funciones del módulo `Raster` de `GPlately`.
- **Reconstrucción de batimetría:** Por medio de esta funcionalidad se puede generar la batimetría de una época específica aplicando el algoritmo de Williams [2]. Las dos últimas de estas funcionalidades se decidió unir en una sola llamada “Reconstruct Rasters”.

4.2. Metodología de desarrollo

Para la implementación de las nuevas funcionalidades se procedió de la siguiente manera. Primero se creó un fork del repositorio oficial del plugin, en donde se realizarían todos los cambios (<https://github.com/LucasAmion/terra-antiqua>). El trabajo realizado se dividió en dos partes principales, la reconstrucción de capas vectoriales y la reconstrucción de rásters.

Primero se trabajó en la herramienta de reconstrucción de capas vectoriales, ya que es la más simple de las dos y se puede utilizar como base para construir la otra. En segunda instancia, se trabajó en la implementación de la herramienta de reconstrucción de capas tipo ráster, la cual se puede dividir en la reconstrucción de topografía y la reconstrucción de batimetría, que utilizarían algoritmos totalmente distintos.

Se decidió mantener la arquitectura de software ya presente en el proyecto, porque su diseño ofrece una buena separación de responsabilidades entre las partes asociadas al backend y el frontend de la aplicación. Esto significa que para añadir cualquiera de las nuevas funcionalidades sería necesaria la creación de un nuevo archivo dentro de la carpeta `core` que contenga

un objeto de clase algoritmo que herede de `TaBaseAlgorithm`, un archivo de diálogo dentro de la carpeta `gui`, que herede de la clase `TaBaseDialog`, así como el correspondiente texto de ayuda escrito en HTML dentro de la carpeta `help_text`.

4.2.1. Reconstrucción de capas vectoriales

La primera funcionalidad que se desarrolló se trata del algoritmo de reconstrucción de capas vectoriales, que permite al usuario tomar una capa vectorial actual, ya sea una obtenida desde un archivo local o una de las ofrecidas por la librería `PlateModelManager`, y rotarla hasta una época específica usando uno de los modelos tectónicos disponibles.

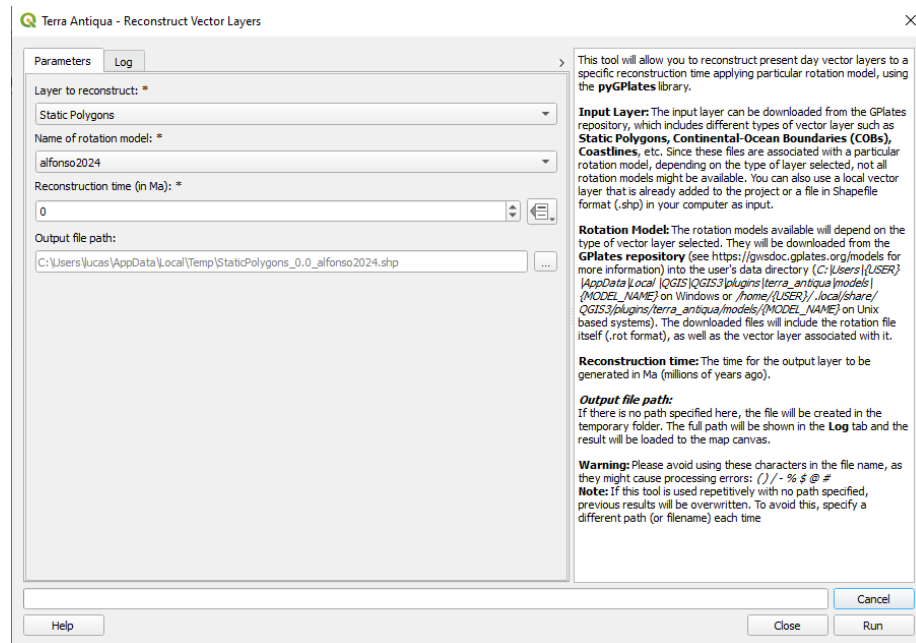


Figura 4.1: Interfaz del algoritmo de reconstrucción de capas vectoriales.

La interfaz asociada a este algoritmo se definió en el archivo `reconstruct_vector_layers_dlg.py` al interior de la carpeta `gui`. Dentro de la función `defineParameters()` se utiliza el método `addMandatoryParameter()` para agregar cada uno de los parámetros asociados con este algoritmo, estos son:

- **Layer to reconstruct:** Corresponde al tipo de capa vectorial que se usará como input para el algoritmo. Las opciones disponibles son: “Static Polygons”, “Continental Polygons”, “Coastlines”, “COBs”, “Cratons” y “Terranes”. Estas capas serán descargadas desde el repositorio oficial de EarthByte usando la librería `PlateModelManager`. También está la opción “Local Layer” que permite utilizar una capa vectorial propia como archivo de entrada. Esta puede ser seleccionada de entre las capas vectoriales ya añadidas al proyecto o se puede cargar desde un archivo en formato Shapefile dentro del computador del usuario.
- **Name of rotation model:** Se refiere al nombre del modelo tectónico que se usará para llevar a cabo la reconstrucción. Las opciones disponibles dependerán del tipo de capa vectorial elegida debido a que están íntimamente relacionados y no todos los modelos contienen todas las capas. Estos modelos también se descargarán usando la librería Pla-

teModelManager. Tanto los modelos como sus capas vectoriales asociadas serán almacenados en la carpeta `C:\User \{USER}\AppData\Local\QGIS\QGIS3\plugins\terra_antiqua\models\{MODEL_NAME}` en Windows o `/home/{USER}/.local/share/QGIS3/plugins/terra_antiqua/models/{MODEL_NAME}` en Linux. Esto permite que puedan ser reusados sin tener que volver a descargarse cada vez.

- **Reconstruction time:** Tiempo de la capa vectorial que se va a generar, medido en Ma (Millones de años atrás).

El algoritmo en sí está implementado dentro del archivo `reconstruct_raster_layers.py` dentro de la carpeta `core` y funciona de la siguiente manera:

- Primero se extraen los valores de los parámetros de entrada desde el objeto de diálogo asociado al algoritmo (`self.dlg`).
- Luego, se realiza la descarga de los archivos del modelo tectónico seleccionado. Esto incluye la descarga del archivo (o los archivos) `.rot` que contiene las rotaciones relativas de cada placa, así como también la capa vectorial que se va a reconstruir atrás en el tiempo, a menos que se haya seleccionado un archivo local. En cualquier caso, la capa obtenida queda guardada en la variable llamada `layer` y el modelo queda guardado en la variable `rotation_model`.
- Más adelante, se comprueba si es que ya existe un archivo con el mismo nombre en la ruta de salida indicada por el usuario y, de ser así, se intenta eliminar dicho archivo. Si por alguna razón el archivo no puede eliminarse, por ejemplo, al estar siendo usado en ese momento, el algoritmo se detiene con un mensaje de error.
- El paso siguiente es la reconstrucción como tal. Aquí se utiliza la función `reconstruct()` de `pyGPlates`, pasando como argumentos la capa vectorial, el modelo de rotación, la ruta de salida y el tiempo de la reconstrucción.
- Finalmente, el resultado es añadido como una nueva capa al proyecto utilizando el constructor de la clase `QgsVectorLayer` y pasando como parámetro la ruta del archivo resultante que queda en formato Shapefile.

En pseudocódigo:

Algorithm 1 Algoritmo de reconstrucción de capas vectoriales

```
model_name = self.dlg.modelName.currentText()
layer_type = self.dlg.layerType.currentText()
... # Así se obtienen todos los demás parámetros desde el objeto self.dlg
rotation_model = cache_manager.download_model(model_name)
if layer_type == "Local Layer" then
|   local_layer = self.dlg.localLayer.currentLayer()
else
|   layer = cache_manager.download_layer(model_name, layer_type)
end if
pygplates.reconstruct(layer, rotation_model, output_path, reconstruction_time)
```

Entre cada uno de los pasos ya listados se imprimen mensajes informativos y se incrementa el porcentaje de completación que será visible en la barra de progreso de la interfaz. Además, se revisa si el valor de la variable `self.killed` es verdadero para saber el algoritmo ha sido cancelado o se encontró con un error, en cuyo caso se detiene la ejecución, no sin antes imprimir el mensaje de error correspondiente.

4.2.2. Reconstrucción de topografía

El segundo algoritmo en ser implementado fue el de reconstrucción de archivos ráster de topografía. Es similar en su funcionamiento al algoritmo de reconstrucción de capas vectoriales, aunque bastante más complejo. Este algoritmo toma una imagen de la topografía actual del planeta (puede ser una imagen de topografía y batimetría combinadas, pero la batimetría será eliminada al realizar la reconstrucción) y lo reconstruye hacia atrás en el tiempo usando un modelo de rotación determinado. También se da la opción de cambiar la resolución del archivo antes de realizar la reconstrucción o de cortar la imagen resultante, definiendo valores máximos y mínimos de latitud y longitud. Como archivo inicial para las reconstrucciones se decidió utilizar el ETOPO Global Relief Model 2022, ya que contiene la información más actualizada y detallada del relieve terrestre. A diferencia de la herramienta anterior, en este caso se utiliza la librería GPlately en lugar de pyGPlates.

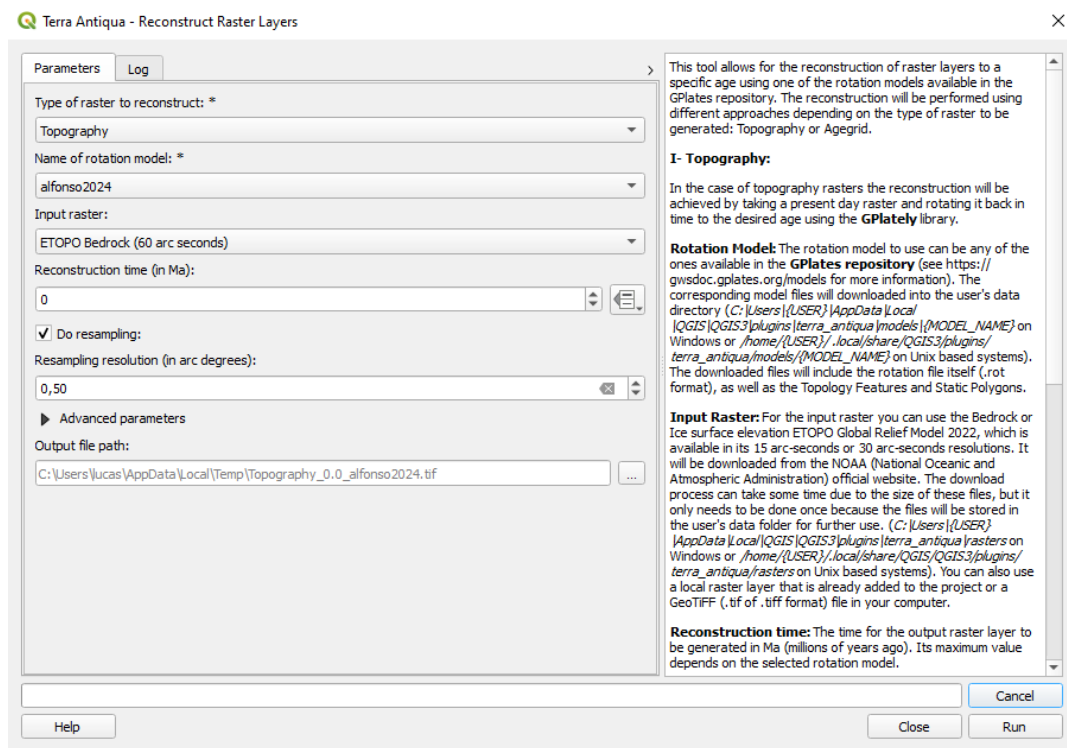


Figura 4.2: Interfaz del algoritmo de reconstrucción de topografía.

La interfaz de este algoritmo se encuentra definida en el archivo `reconstruct_rasters_dlg.py`, dentro del cual se definen los siguientes parámetros:

- **Name of rotation model:** Al igual que en el caso de la reconstrucción de capas vectoriales, corresponde al modelo tectónico de rotación que se usará en la reconstrucción, el cual se descargará por medio de la librería PlateModelManager.

- **Input raster:** Análogo al parámetro “Layer to reconstruct” del algoritmo para capas vectoriales. Permite elegir entre los distintos archivos del ETOPO Global Relief Model 2022. Tenemos las variantes Bedrock y Ice Topography, ambas disponibles en resoluciones de 60 o de 30 segundos de arco. Estos archivos serán descargados desde la página oficial del NCEI (National Centers for Environmental Information). Alternativamente, también se puede utilizar un archivo local en formato GeoTiff o una capa tipo ráster ya añadida al proyecto.
- **Reconstruction time:** Tiempo del archivo ráster de salida del algoritmo en Ma (Millones de años atrás).
- **Do resampling:** Si se activa esta opción se podrá disminuir la resolución de la imagen antes de comenzar la reconstrucción. Esto puede ser útil cuando no se requiere de una precisión tan elevada, ya que permite disminuir el tiempo que tomará la reconstrucción, además del tamaño del archivo resultante.
- **Resampling resolution:** Este parámetro determina la nueva resolución, medida en grados, que tendrá la imagen si es que la opción “Do resampling” está marcada. Si bien se puede seleccionar valores mayores a la resolución del archivo de entrada, esto no tendría mucho sentido ya que aumentaría el tamaño el archivo de salida sin agregar ninguna información nueva.
- **Interpolation method:** Método a utilizar para calcular cada uno de los puntos nuevos generados durante el proceso de resampling. Se puede elegir entre interpolación de 1° (lineal), 2° (cuadrada), 3° (cúbica), 4° o 5° grado.
- **Minimum/Maximum longitude:** Modificando estos valores se puede acotar la extensión de la imagen a una zona específica del planeta.
- **Minimum/Maximum latitude:** Análogo al anterior pero para la latitud.
- **Number of threads:** Esto controla la cantidad de procesos que se ejecutarán en paralelo durante el proceso de reconstrucción. Un mayor número de threads significará un menor tiempo de procesamiento, aunque esto está limitado por la cantidad de núcleos del procesador.

El algoritmo en sí se encuentra implementado en el archivo `reconstruct_rasters.py`. Su funcionamiento grandes rasgos se describe a continuación:

- El primer paso es leer el valor de los parámetros de entrada.
- Luego se descarga el modelo de rotación, además de las capas vectoriales de Static Polygons o COBs (Continent Ocean Boundaries) dependiendo de cuál está disponible para el modelo seleccionado. Esta capa vectorial se usará para cortar el raster y poder extraer solo la topografía del mismo. Este paso solo se ejecuta si es que el tiempo de reconstrucción seleccionado es mayor que cero.
- A continuación, se descarga el ráster de entrada o se lee el archivo local, dependiendo de la opción seleccionada por el usuario. De cualquier modo, el ráster inicial queda guardado como un objeto de la clase `Raster` de `GPlately`.

- Ahora se realiza el proceso de resampling que se describió anteriormente, en el caso de que el usuario haya marcado esa opción. Para esto se utiliza el método `resample()` propio de la clase `Raster` de `GPately` que recibe como argumentos la resolución de resampling y el método de resampling elegidos.
- En el siguiente paso se realiza la reconstrucción en sí (solo si el tiempo de reconstrucción es mayor a cero). Aquí se llama a la función `reconstruct()` de la clase `Raster`, pasándole el tiempo de reconstrucción, el número de threads y la capa vectorial de particionamiento como argumentos.
- Luego se corta el ráster según los límites de longitud y latitud definidos.
- Finalmente, se exporta el resultado a un archivo en formato GeoTiff en la ruta indicada y se agrega dicho archivo como una nueva capa llamando al constructor de la clase `QgsRasterLayer` de la API de QGIS.

4.2.3. Reconstrucción de batimetría

Para la reconstrucción de batimetría se utilizó el algoritmo descrito en la sección 2.2. Se creó un fork del repositorio de Simon Williams que alberga el código del algoritmo (<https://github.com/LucasAmion/agegrid/>), para poder realizar algunos pequeños cambios que permitan utilizar el algoritmo dentro de Terra Antiqua. El principal cambio realizado fue el modo en el que se obtienen los parámetros de entrada. Originalmente, estos eran definidos dentro de un archivo de configuración en formato YAML, y se modificó para que estos sean obtenidos en forma de argumentos de función, para que así sea más fácil utilizar los valores ingresados interactivamente por el usuario. También se cambió ligeramente la forma en que se realiza la paralelización en las etapas de procesamiento más intensivo del algoritmo. En el código original se usaban procesos y se pasó a utilizar threads, debido a que el código no funcionaba correctamente en el sistema operativo Windows. El problema es que al usar procesos se abrían múltiples ventanas de QGIS por cada nuevo proceso creado, lo cual claramente no era el resultado esperado. Para utilizar este código dentro de Terra Antiqua simplemente se importa la función `run_paleo_age_grids()` dentro del archivo `reconstruct_rasters.py`.

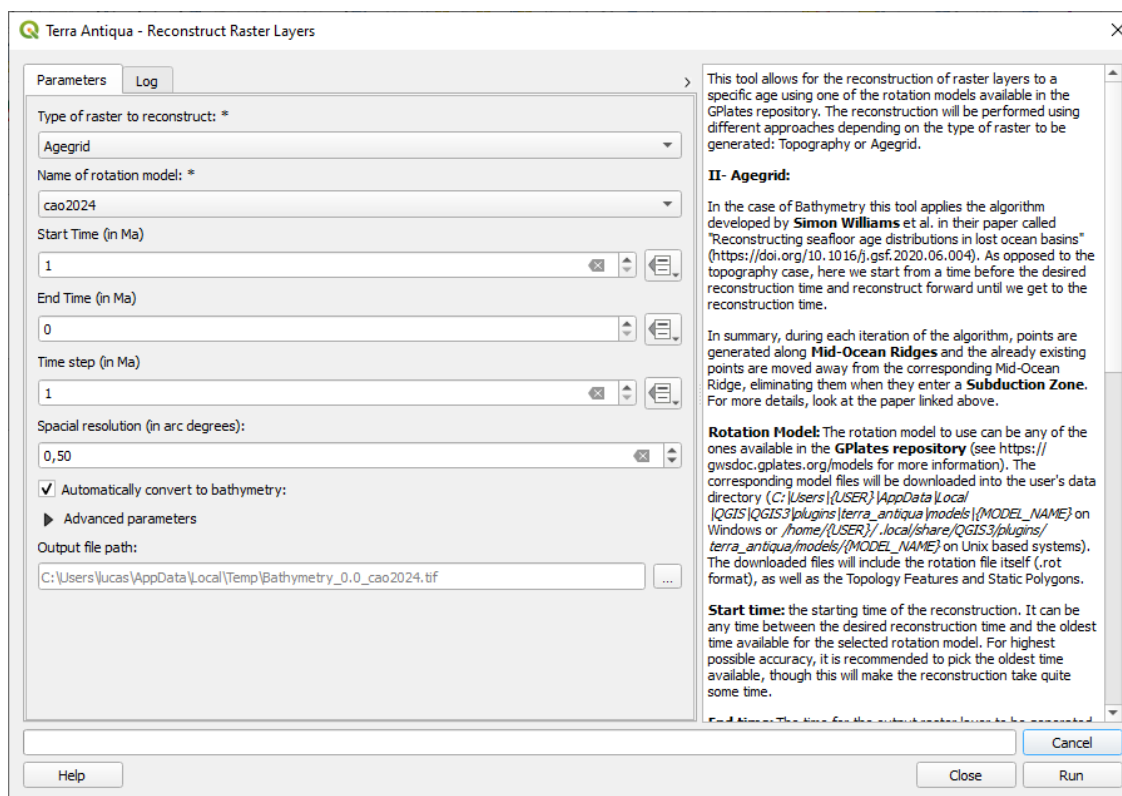


Figura 4.3: Interfaz del algoritmo de reconstrucción de batimetría.

Se decidió agregar esta funcionalidad como una nueva opción dentro del algoritmo de reconstrucción de rasters ya presente en lugar de definir una herramienta separada. Para esto se agregó un nuevo parámetro llamado “Type of raster to reconstruct” cuyo valor puede ser “Topography” o “Agegrid”. Se utilizó la función `addVariantParameter()` para agregar los parámetros que son específicos para cada una de estas dos opciones, de manera que estos estén ocultos si no son necesarios. Parámetros como el modelo de rotación, longitud/latitud máxima y mínima y número de threads son compartidos para ambos algoritmos. Los parámetros específicos del algoritmo de reconstrucción de batimetría son los siguientes:

- **Start time:** El tiempo de inicio de la reconstrucción. Puede ser cualquier valor entre el tiempo de reconstrucción deseado y el tiempo más antiguo disponible para el modelo de rotación seleccionado. Para lograr una mayor precisión, es recomendable elegir el tiempo más antiguo posible, aunque esto hará que la reconstrucción tome bastante tiempo.
- **End time:** Tiempo de destino de la reconstrucción en Ma. Debe ser al menos un Ma más reciente que el tiempo de inicio.
- **Time step:** Tiempo transcurrido entre cada una de las iteraciones del algoritmo. Seleccionar un valor más alto acelerará el proceso, pero reducirá la precisión del resultado.
- **Spatial resolution:** Tiene un efecto similar a la resolución de resampling del algoritmo de topografía. Se refiere a la distancia entre los puntos generados a lo largo de las dorsales oceánicas en cada iteración del algoritmo y entre los puntos generados en el último paso, donde el resultado se convierte a un archivo en formato ráster.

- **Convert to bathymetry:** Este parámetro está habilitado de forma predeterminada. El ráster de edad generado (Agegrid), se convertirá automáticamente en batimetría utilizando la fórmula presentada en la sección 2.3.
- **Spreading rate:** Corresponde al spreading rate utilizado en el primer paso del algoritmo en donde se rellenan los valores del mapa en el tiempo inicial de la reconstrucción. Se aconseja no modificar el valor por defecto, ya que es el recomendado por los autores.

Como se mencionó anteriormente, esta funcionalidad fue integrada junto con la de reconstrucción de topografía dentro de un solo algoritmo que se resume a continuación en forma de pseudocódigo:

Algorithm 2 Algoritmo de reconstrucción de capas tipo ráster

```

model_name = self.dlg.modelName.currentText()
raster_type = self.dlg.rasterType.currentText()
... # Así se obtienen todos los demás parámetros desde el objeto self.dlg
if raster_type == "Topography" then
    rotation_model = cache_manager.download_model(model_name)
    if local then # Si el archivo raster inicial seleccionado es local o no
        | topo_raster = gdal.Open(local_layer)
    else
        | topo_raster = cache_manager.download_raster(rasterIdx)
    end if
    if resampling then
        | topo_raster.resample(resampling_resolution, interpolationMethod)
    end if
    topo_raster.reconstruct(reconstruction_time, n_threads)
    extent = (minlon, maxlon, minlat, maxlat)
    if extent != (-180, 180, -90, 90) then
        | clipArrayToExtent(topo_raster, extent) # Cortar la imagen según valores máximos
        | y mínimos de latitud y longitud
    end if
    exportArrayToGeoTIFF(output_path, topo_raster)
end if
if raster_type == "Agegrid" then
    cache_manager.download_model(model_name)
    agegrid = run_paleo_age_grids(model_name, start_time, end_time, time_step, re-
    solution, minlon, maxlon, minlat, maxlat, n_threads, spreading_rate)
    if convert then # Si la opción de convertir a batimetría fue seleccionada por el usuario
        | agegrid = convertAgeToDepth(agegrid)
    end if
    exportArrayToGeoTIFF(output_path, agegrid)
end if

```

4.2.4. Descarga de archivos

En un principio, la descarga de archivos, ya sea modelos de rotación, capas vectoriales o imágenes ráster, estaba implementada dentro de los propios algoritmos, pero se decidió mover el código asociado a esta funcionalidad a una clase separada llamada `TaCacheManager` dentro del archivo `cache_manager.py`. De esta manera, se puede reutilizar código, usando las mismas funciones tanto en el algoritmo de reconstrucción de capas vectoriales como en el de reconstrucción de capas tipo ráster. Se utilizan las clases para `PlateModelManager` y `PresentDayRasterManager` de la librería `plate_model_manager` para implementar los distintos métodos de esta clase, algunos de los cuales son:

- **`get_available_models`**: Recibe como argumento una lista de capas requeridas (que puede ser vacía) y entrega un listado de modelos de rotación que contienen las capas solicitadas. Esta función es utilizada para obtener las opciones que se mostrarán en la interfaz de los algoritmos.
- **`is_layer_available`**: Este método es solo usado internamente por la función `get_available_models()` para saber si una capa está disponible para un modelo específico.
- **`get_model_bigtime`**: Permite saber el tiempo máximo de un modelo de tectónico particular. Se utiliza para definir el valor máximo del parámetro “Reconstruction Time”.
- **`download_model`**: Descarga un modelo tectónico de rotación.
- **`download_layer`**: Descarga una capa vectorial asociada a un modelo de rotación específico.

4.2.5. Documentación

Las últimas tareas realizadas fueron escribir los mensajes de ayuda en formato HTML que se muestran en la parte derecha de los cuadros de diálogo de los algoritmos, explicando el propósito de cada uno de los parámetros de entrada, y actualizar el archivo README del proyecto, agregando instrucciones de instalación del plugin escritas en inglés, las que se pueden encontrar en el Anexo A.

5. Validación y prueba de la herramienta

5.1. Prueba de funcionalidades

Para evaluar los resultados obtenidos en esta memoria, se llevó a cabo un ejemplo de ejecución que combina todas las nuevas características desarrolladas, además de algunas de las ya presentes en el plugin. Este ejemplo se trata de una reconstrucción enfocada en la época de 30 millones de años atrás. El procedimiento fue el siguiente: Primero se generó la topografía y la batimetría por separado, usando las herramientas correspondientes. Luego se unieron ambas partes en una sola imagen ráster usando la herramienta de compilación de topo/batimetría, descrita en la sección 3.1.1. Se utilizó el modelo de Müller 2022, ya que es uno de los más recientes y completos. A continuación, se realizó la reconstrucción de los COBs asociados a este modelo a la misma época usando la herramienta de reconstrucción de capas vectoriales. Finalmente, usando la herramienta de modificación de topo/batimetría (ver sección 3.1.3) se alteró el relieve de la zona noroeste de los Andes, región que, según información presente en la literatura, solía tener una altura menor a la actual durante la época de estudio [8].

5.1.1. Reconstruyendo la topografía

Para la creación de la topografía de la época en cuestión se utilizó la herramienta de reconstrucción de capas de tipo ráster haciendo clic en el primer botón de la barra de tareas de Terra Antiqua y seleccionando la opción de reconstruir topografía en el primer parámetro, llamado “Type of raster to reconstruct”. Luego, se seleccionó el modelo de rotación a utilizar durante todo este ejemplo, que como ya se dijo se trata del modelo Müller 2022. Como archivo de entrada se utilizó el ETOPO 2022 en su versión de Bedrock Topography, ya que la elevación debida al hielo de los polos no será relevante para este ejemplo. Se eligió la variante de 60 arcosegundos de resolución porque no se necesita una precisión tan alta y la descarga tomará menos tiempo al ser un archivo de menor tamaño. Además, se usó la función de resampling para disminuir la resolución del ráster a 0.5 grados y así acelerar el proceso de reconstrucción. En este caso no se modificaron ninguno de los parámetros avanzados de este algoritmo.

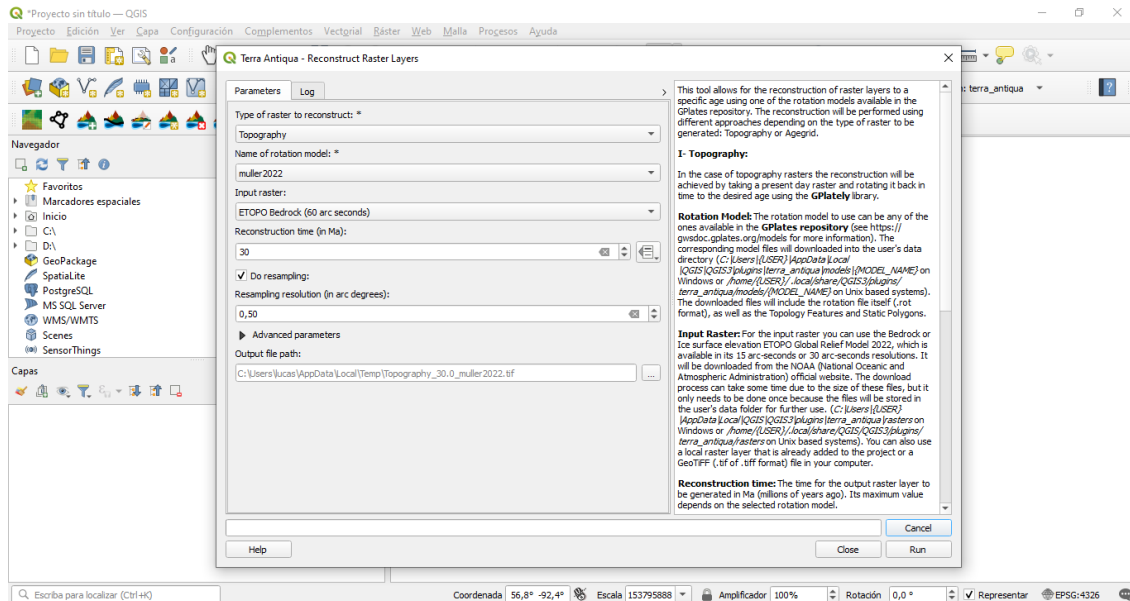


Figura 5.1: Parámetros usados en la reconstrucción de topografía de la época de 30 Ma.

Luego de hacer clic en el botón “Run” y esperar que finalice la ejecución del algoritmo, el resultado fue agregado al proyecto como una nueva capa llamada “Topography_30.0_muller2022”, la cual es inmediatamente visible en el mapa y en el panel de capas de la parte izquierda de la ventana. Esto se puede ver en la figura 5.2.

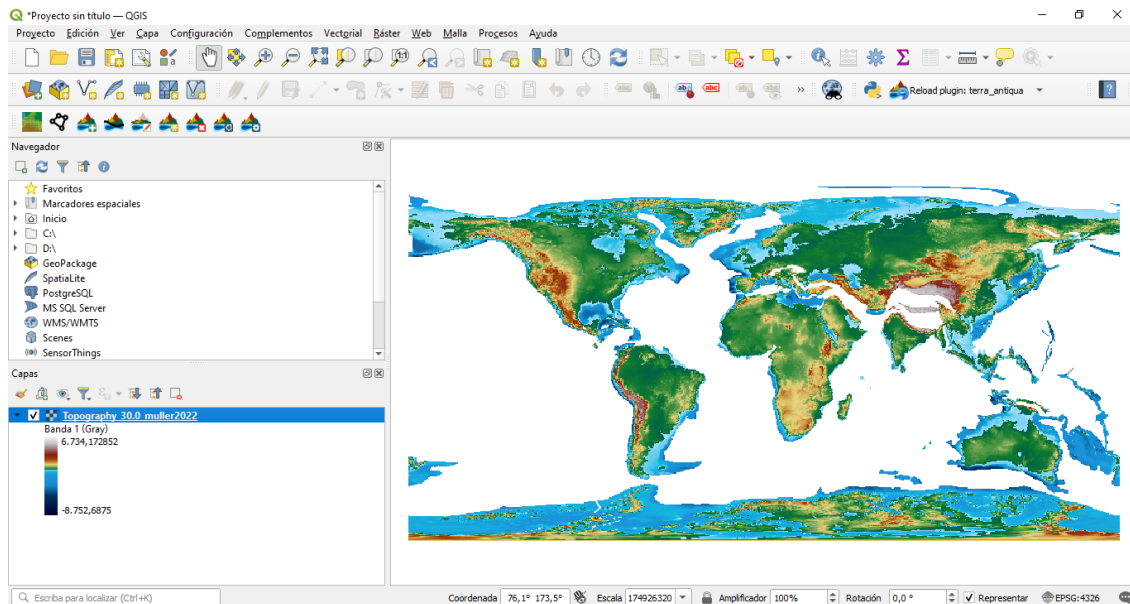


Figura 5.2: Resultado de la reconstrucción de topografía para 30 Ma.

5.1.2. Reconstruyendo la batimetría

Una vez obtenida la topografía de la época de estudio se continuó con la reconstrucción de la batimetría. Para esta etapa se vuelve a utilizar la herramienta de reconstrucción archivos ráster, pero esta vez seleccionando la opción “Agegrid” en el parámetro “Type of raster to reconstruct”, lo que cambia los parámetros de entrada disponibles para el algoritmo revelando

los parámetros específicos de la reconstrucción de una Agegrid. En este diálogo se seleccionó el mismo modelo de rotación, Müller 2022. Es importante que el modelo utilizado para la generación de la topografía sea el mismo usado para la generación de batimetría para que los resultados puedan ser unidos correctamente. Para el tiempo de inicio de la reconstrucción se seleccionó la época de 100 millones de años atrás. Como se mencionó en la sección 2.2 es recomendable usar un valor de por lo menos 50 millones de años previo a la edad de reconstrucción, para mitigar los efectos de las suposiciones que se hacen para generar el fondo oceánico inicial, por lo que este valor debería ser más que suficiente. También se aumentó el intervalo de tiempo a 2 Ma para acortar el tiempo de procesamiento y se ingresó el tiempo de reconstrucción, que como ya se dijo equivale a 30 Ma. Se usó una resolución espacial de 0.5 grados, la misma usada para el ráster de topografía. Esto es relevante debido a que, para que la herramienta de compilación funcione correctamente, todas las capas que se van a unir deben tener la misma resolución. Finalmente, se usó la opción de convertir la Agegrid a batimetría, la cual se encuentra marcada por defecto. Como se puede observar en la figura 5.5, el ráster de batimetría generado encaja perfectamente con el de topografía ya generado, gracias a haber sido generados con el mismo modelo de rotación.

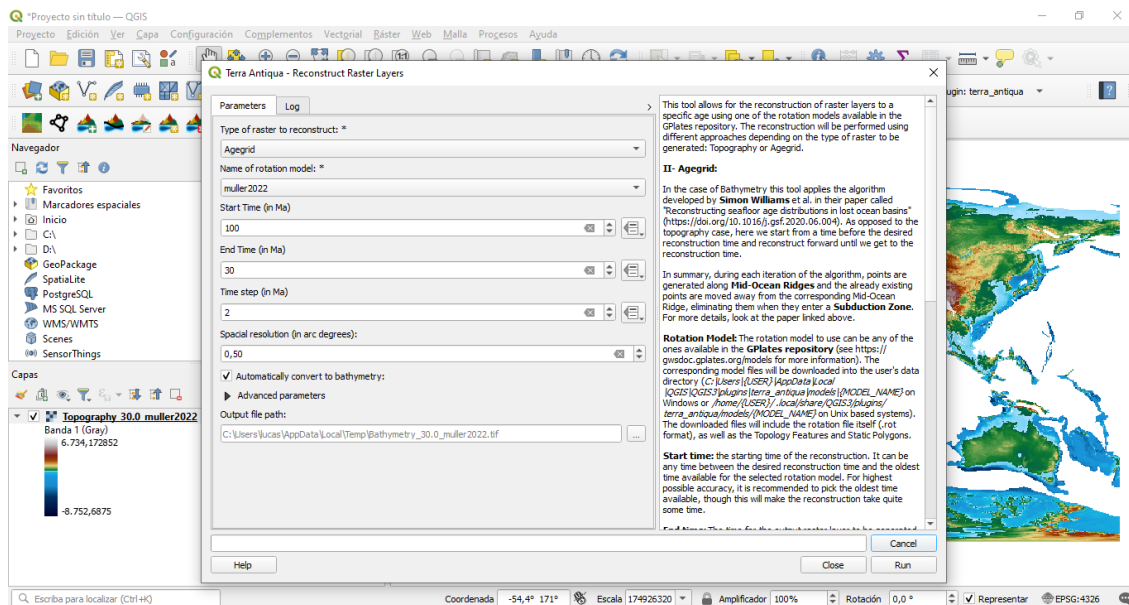


Figura 5.3: Parámetros usados en la reconstrucción de batimetría de la época de 30 Ma.

5.1.3. Integrando las imágenes

Al tener ya ambos archivos ráster, se procedió a unirlos utilizando la herramienta de compilación de topo/batimetría, que corresponde al tercer botón contando desde la izquierda en la barra de tareas de Terra Antiqua. Aquí, simplemente se seleccionó la capa de batimetría, se presionó el botón con un signo “+” para añadir otra capa y allí se agregó la de topografía. Al presionar “Run”, el proceso estuvo listo de manera casi inmediata, obteniendo así una nueva capa llamada “Compiled_DEM_Topo+Bathy”. Como esta capa contiene toda la información de las dos anteriores, éstas fueron eliminadas.

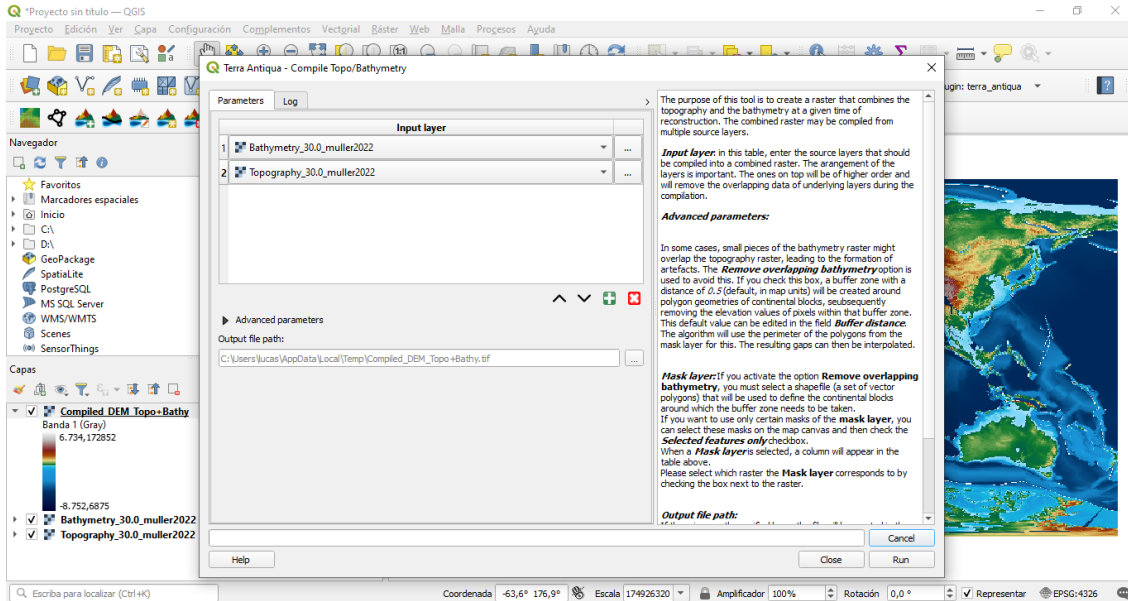


Figura 5.4: Parámetros usados para la compilación de topografía y batimetría de 30 Ma.

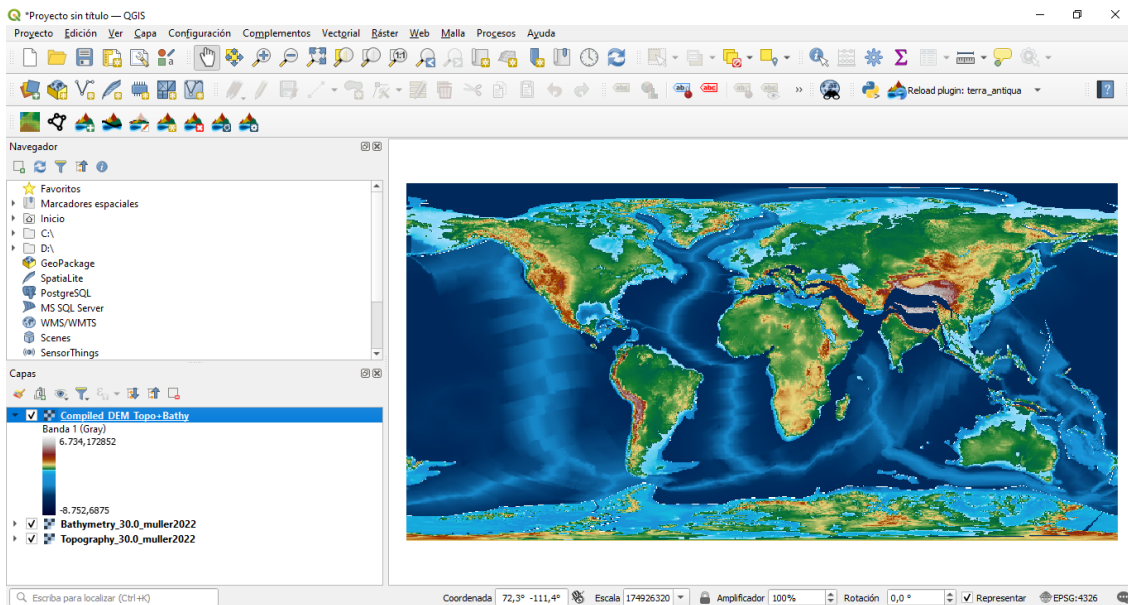


Figura 5.5: Ráster compilado de topografía y batimetría de 30 Ma.

5.1.4. Reconstruyendo COBs

En el siguiente paso se realizó la reconstrucción de los COBs (Continental Ocean Boundaries) asociados al modelo Müller 2022. Esto se llevó a cabo para poder luego usar los polígonos definidos en esta capa vectorial dentro de la herramienta de modificación de topo/batimetría, que requiere no solo el archivo ráster que se va a modificar, sino también una capa vectorial que delimitará la zona del mapa donde la modificación tendrá lugar. Se utilizó la herramienta de reconstrucción de capas vectoriales (segundo botón en la barra de tareas), ingresando el tipo de capa vectorial que se va a reconstruir, que en este caso son los COBs, el modelo tectónico de rotación, que tiene que ser el mismo utilizado en los pasos anteriores, es decir,

Müller 2022, y la edad de la reconstrucción (30 Ma). Al finalizar el proceso, que es bastante más rápido que la reconstrucción de una imagen ráster, se obtuvo una nueva capa vectorial añadida al mapa que, como se puede contemplar en la figura 5.7, encaja a la perfección con el ráster ya generado, debido a que fue justamente esta capa la que se utilizó para cortar las imágenes en los pasos anteriores.

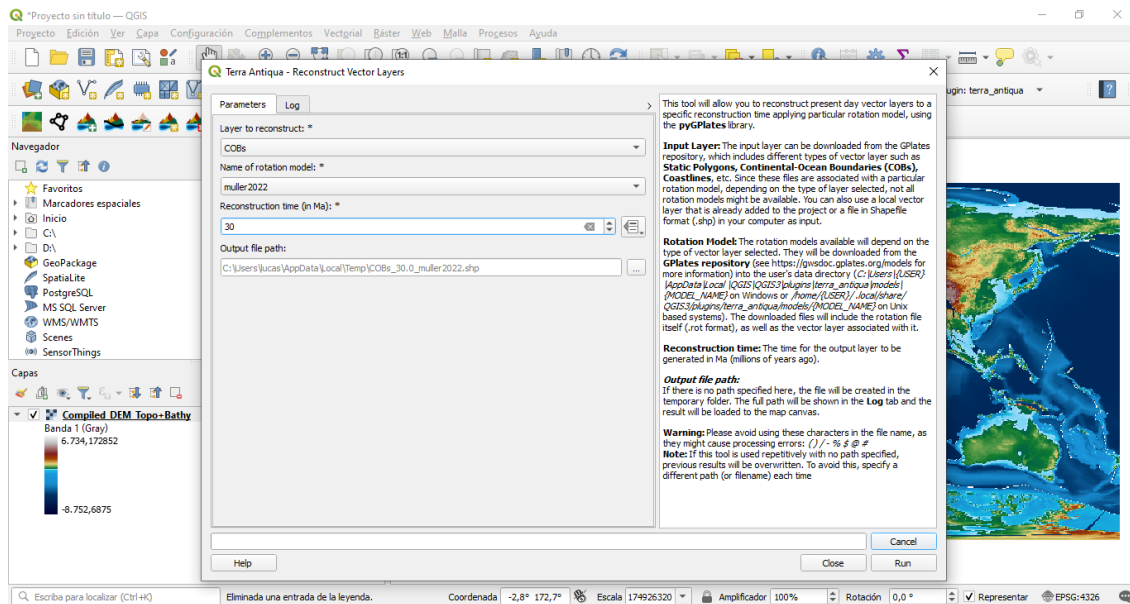


Figura 5.6: Parámetros usados para la reconstrucción de los COBs del modelo Müller 2022 a la época de 30 Ma.

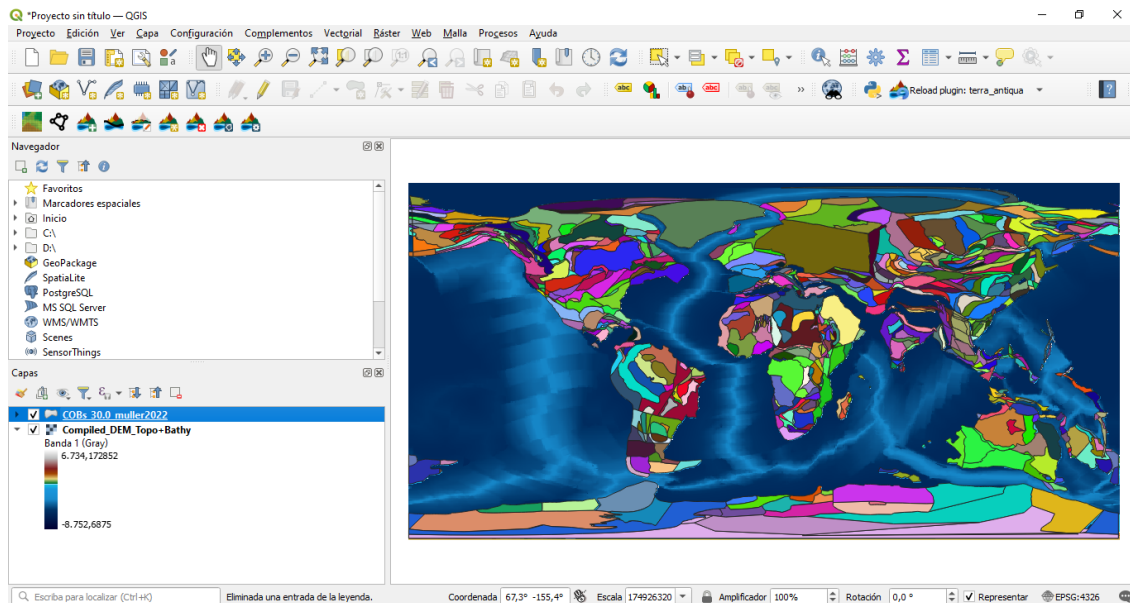
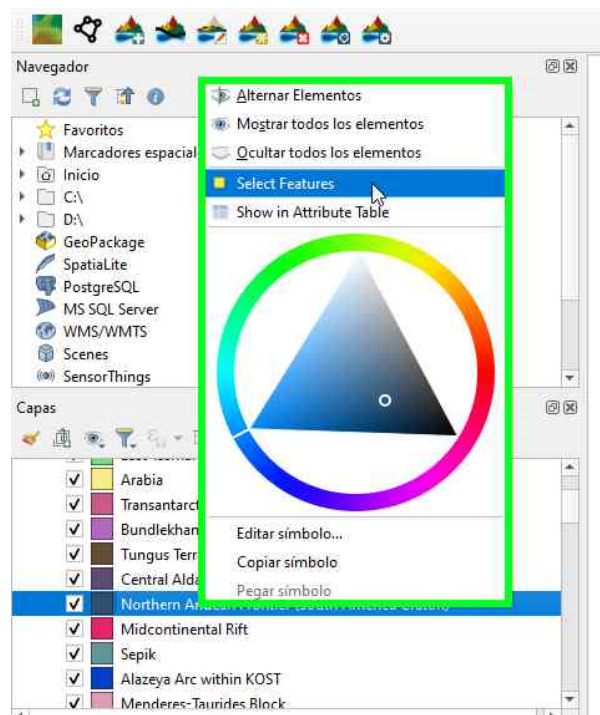
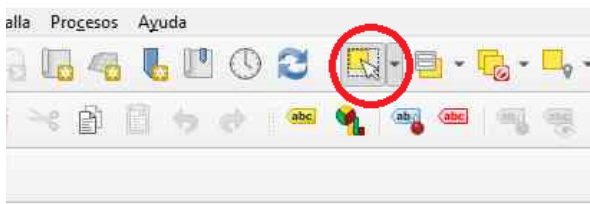


Figura 5.7: Resultado de la reconstrucción de los COBs del modelo Müller 2022 a la época de 30 Ma.

5.1.5. Modificando topografía

Finalmente, se utilizó la herramienta de modificación de topo/batimetría en donde se requirió el uso de las dos capas ya generadas. Antes que eso, sin embargo, fue necesario seleccionar el polígono específico de interés para la modificación a realizar, de entre todos los que forman parte de la capa vectorial en uso. Esto se puede lograr haciendo clic en el botón de “Seleccionar objetos por área” que está en la parte superior de la ventana de QGIS. También se puede expandir la capa de los COBs dentro del panel de capas en la parte izquierda y buscar el polígono deseado por nombre, en este caso se trata de “Northern Andean Frontier (South America Craton)”, hacer clic derecho en él y escoger la opción llamada “Select features”. Con el polígono ya seleccionado, se utilizó el menú de modificación de topo/batimetría seleccionando la capa de topografía y batimetría compiladas como archivo de entrada y la capa vectorial de los COBs para delimitar el área de la modificación. Se marcó la casilla “Select features only” para que solo el polígono seleccionado sea tomado en cuenta. El modo de modificación aplicado fue el de reescalar con valores máximos y mínimos. Se utilizaron los valores de 4000 km y 2000 km de altura, respectivamente. El resultado obtenido se puede observar en la figura 5.10, en donde se aprecia cómo el relieve de la zona efectivamente cambió con respecto a la imagen original.



(a) Usando la herramienta de “Seleccionar objetos por área”.

(b) Buscando el polígono dentro del panel de capas.

Figura 5.8: Selección del polígono de interés para la modificación de topografía.

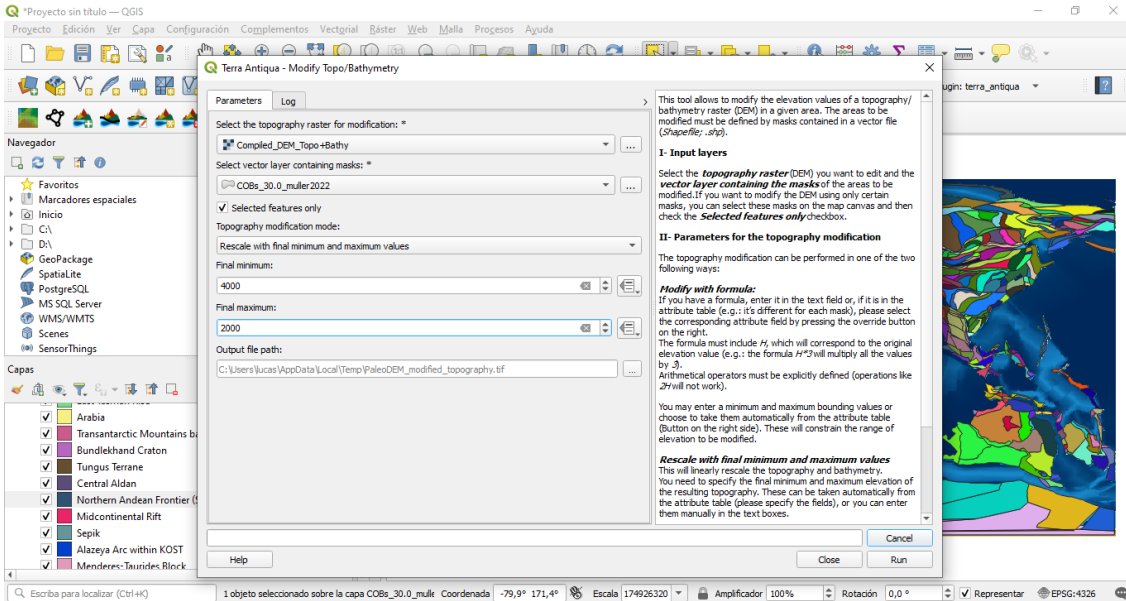


Figura 5.9: Parámetros para la modificación de la topografía de los Andes a los 30 Ma.

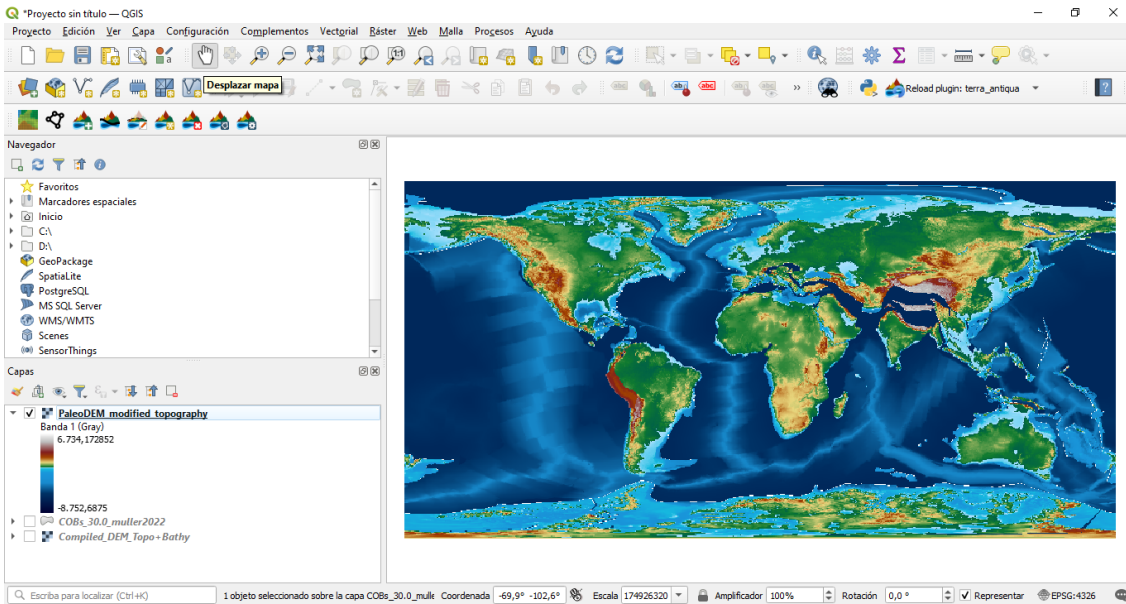


Figura 5.10: Resultado de la modificación de la topografía de los Andes a los 30 Ma.

La reconstrucción aquí desarrollada no pretende ser científicamente precisa, sino que se realiza con propósitos ilustrativos. Aun así, permite comprobar cómo las nuevas funcionalidades añadidas a Terra Antiqua lo han convertido en una herramienta muy completa que nos permitió ejecutar el proceso completo de construcción de un paleoDEM, desde la descarga de archivos de entrada, pasando por la reconstrucción basada en modelos de rotación, hasta la modificación manual posterior, todo sin tener que salir de QGIS. En el pasado, para llevar a cabo este mismo tipo de reconstrucción, habría sido necesario tener los archivos de entrada de antemano. Además, toda la primera parte del proceso habría tenido que ser llevada a cabo en GPlates para luego exportar y cargar los resultados dentro de QGIS, lo que resulta bastante menos conveniente.

5.2. Benchmark del algoritmo de batimetría

También se realizó una prueba más exhaustiva con el algoritmo de generación de batimetría ya que este es el más complejo y el que usa más recursos computacionales. El objetivo de esta prueba fue comprobar el desempeño del algoritmo en los aspectos de uso de memoria, tiempo transcurrido y la correctitud de los resultados, comparándolo con el código original de Simon Williams [2]. Para demostrar que el algoritmo efectivamente permite la generación de batimetría de épocas muy remotas para las cuales toda la corteza terrestre ha desaparecido, se decidió basar la prueba en la generación de la batimetría para la época de 500 Ma. Se usó un tiempo inicial de 600 Ma y un intervalo de 1 Ma. El modelo de rotación utilizado fue el mismo de la parte anterior (Müller 2022). Se repitió la prueba con dos resoluciones diferentes (1 y 0.1 grados) y usando 1 o 4 núcleos para determinar el impacto que tienen estos parámetros sobre el tiempo transcurrido y la memoria utilizada. Las pruebas fueron realizadas en un computador con sistema operativo Windows 10 de 64 bits con 8 GB de RAM y un procesador Intel Core i3 de 11° generación de 4 núcleos.

5.2.1. Desempeño en memoria y tiempo

Los resultados obtenidos en cuanto a tiempo transcurrido y memoria utilizada para los distintos escenarios de prueba se pueden observar en la tabla 5.1. Lo primero que se puede notar es que tanto los tiempos como los valores de memoria obtenidos son bastante similares entre el código original y el algoritmo de Terra Antiqua. Esto demuestra que, en términos de eficiencia, ambos son comparables. La pequeña demora observada podría atribuirse a la interfaz de usuario de QGIS, que consume recursos adicionales de la CPU, algo que no ocurre en el caso del código original.

En cuanto al uso de memoria, se puede notar que el código de Terra Antiqua tiende a ocupar menos memoria RAM que el código original. Esto se puede atribuir al cambio realizado en el backend de paralelización del algoritmo, mencionado en la sección 4.2.3, ya que los threads son una alternativa más liviana frente a los procesos. También se puede observar que, si bien el uso de un mayor número de núcleos disminuye el tiempo de procesamiento necesario, esta mejora no es tan significativa como se habría esperado, siendo solamente del orden de entre 20% y 30%. Esto se puede deber a que solo algunas partes del algoritmo pueden paralelizarse, mientras que la mayoría de ellas se deben realizar de manera secuencial.

Por último, podemos comprobar que la resolución utilizada tiene un impacto considerable, tanto en el tiempo transcurrido como en la cantidad de memoria utilizada. Si se desea obtener resultados con resoluciones mayores a las usadas aquí, sería necesario un computador con mucha más memoria RAM.

Tabla 5.1: Resultados de benchmark para el algoritmo de generación de batimetría

Software utilizado	Resolución	Núcleos de CPU	Tiempo transcurrido	Máxima memoria utilizada
Algoritmo original de Williams	1.0°	1	17 m 30 s	1.112.836 K
		4	12 m 20 s	1.108.312 K
	0.1°	1	1 h 26 m 32 s	4.788.548 K
		4	1 h 12 m 22 s	4.616.084 K
Terra Antiqua	1.0°	1	17 m 47 s	1.313.012 K
		4	13 m 19 s	1.081.560 K
	0.1°	1	1 h 42 m 39 s	4.372.116 K
		4	1 h 14 m 27 s	3.910.508 K

5.2.2. Validación de los resultados

Para validar la exactitud de las imágenes ráster producidas mediante el algoritmo de generación de batimetría se escribió un pequeño script de Python que compara los archivos obtenidos mediante ambos métodos, calculando la diferencia porcentual promedio entre ambos. En este script se utiliza la librería `rasterio` para leer los archivos GeoTIFF y funciones de la librería `numpy` para realizar los cálculos necesarios. El código en cuestión se encuentra a continuación:

Código 5.1: Script de Python que compara los resultados de ambos métodos.

```

1 import rasterio
2 import numpy as np
3
4 def calcular_diferencia(archivo_tiff1, archivo_tiff2):
5     # Leer las matrices de los archivos GeoTIFF
6     with rasterio.open(archivo_tiff1) as src1, rasterio.open(archivo_tiff2) as src2:
7         matriz1 = src1.read(1) # Leer la primera banda como matriz
8         matriz2 = src2.read(1) # Leer la primera banda como matriz
9
10    # Eliminar valores NaN
11    matriz1 = np.nan_to_num(matriz1, nan=0)
12    matriz2 = np.nan_to_num(matriz2, nan=0)
13
14    # Verificar si las dimensiones coinciden
15    if matriz1.shape != matriz2.shape:
16        raise ValueError("Las dimensiones de las matrices no coinciden.")
17
18    # Calcular la diferencia porcentual entre ambas matrices
19    epsilon = 1e-10 # Se añade un pequeño valor para evitar división por 0
20    diferencia_porcentual = (np.abs(matriz1 - matriz2) / (np.abs(matriz1) + epsilon)) * 100
21
22    # Calcular el promedio de la diferencia porcentual
23    diferencia_porcentual_promedio = np.nanmean(diferencia_porcentual)
24
25    print("Promedio de la diferencia porcentual:", np.nanmean(

```

```

    ↪ diferencia_porcentual_promedio))
26
27 print("Resolución 0.1°")
28 archivo_tiff1 = "tiff_files/williams-0.1.tif"
29 archivo_tiff2 = "tiff_files/terrantiqua-0.1.tif"
30 calcular_diferencia(archivo_tiff1, archivo_tiff2)
31
32 print("Resolución 1°")
33 archivo_tiff1 = "tiff_files/williams-1.0.tif"
34 archivo_tiff2 = "tiff_files/terrantiqua-1.0.tif"
35 calcular_diferencia(archivo_tiff1, archivo_tiff2)

```

Las dimensiones de las matrices son correctas, pero se obtuvo una diferencia porcentual de un 1.8099% para las imágenes de 0.1° y de 1.8997617% para las imágenes de 1° de resolución. Si bien este es un valor bastante bajo, no se esperaba que hubiera diferencia alguna entre ambas y más trabajo tendrá que ser realizado a futuro para determinar el origen de estas discrepancias. Es posible que estas diferencias sean producto del cambio realizado en el funcionamiento de la paralelización del algoritmo, mencionado anteriormente.

6. Conclusiones

En este trabajo de memoria se extendió la funcionalidad del plugin de QGIS llamado Terra Antiqua, añadiendo la posibilidad de realizar reconstrucciones de mapas en formato ráster o vectorial hacia atrás en el tiempo, aprovechando las características de la API de GPLates. Esto cumple el objetivo de brindar a los investigadores en la materia una herramienta autocontenida con la cual puedan realizar todas las etapas de una reconstrucción paleogeográfica dentro de un mismo programa sin tener que recurrir a herramientas externas.

Cabe destacar que, para el caso particular de la reconstrucción de batimetría, se hace uso de un método relativamente nuevo en la literatura que hasta el momento no había sido incorporado dentro de un programa de escritorio. Integrar este procedimiento dentro del flujo de trabajo de Terra Antiqua le otorga a este algoritmo una interfaz amigable que facilita enormemente su utilización para usuarios no expertos en programación.

En cuanto a posibles trabajos futuros que se podrían realizar para extender aún más las posibilidades ofrecidas por este software, está la idea de poder cargar modelos de rotación propios desde archivos locales, en lugar de tener acceso únicamente a los que se pueden descargar desde el sitio web de GPLates, publicados por otros autores. Esto se había contemplado como posible tarea a realizar dentro de esta memoria, pero en última instancia no fue posible por falta de tiempo y por haber priorizado otros aspectos.

Adicionalmente, en un futuro se podría agregar la capacidad de guardar en caché los archivos intermedios obtenidos durante el proceso de generación de batimetría para acelerar el proceso la próxima vez que se ejecute. Además, será necesario investigar el origen de las pequeñas discrepancias encontradas en los resultados de este algoritmo en comparación con los obtenidos mediante el código original de Simon Williams [2].

Por otro lado, también está la posibilidad de incluir otros métodos y flujos de trabajo presentes en la literatura que permiten realizar reconstrucciones paleogeográficas por otros medios. Por ejemplo, tenemos el paper de Gurnis (2018) en donde se discute un nuevo método de reconstrucción de topografía que utiliza placas que se deforman a lo largo del tiempo en lugar de polígonos estáticos [9] o la librería pyBacktrack que contiene algoritmos para reconstrucción de batimetría similares al utilizado aquí, pero que usan otro tipo de evidencias geológicas como base [10].

Bibliografía

- [1] Torsvik, T. H., Steinberger, B., Gurnis, M., y Gaina, C., “Plate tectonics and net lithosphere rotation over the past 150my”, *Earth and Planetary Science Letters*, vol. 291, no. 1, pp. 106–112, 2010, doi:<https://doi.org/10.1016/j.epsl.2009.12.055>.
- [2] Williams, S., Wright, N. M., Cannon, J., Flament, N., y Müller, R. D., “Reconstructing seafloor age distributions in lost ocean basins”, *Geoscience Frontiers*, vol. 12, no. 2, pp. 769–780, 2021, doi:<https://doi.org/10.1016/j.gsf.2020.06.004>.
- [3] Baatsen, M., van Hinsbergen, D. J. J., von der Heydt, A. S., Dijkstra, H. A., Sluijs, A., Abels, H. A., y Bijl, P. K., “Reconstructing geographical boundary conditions for palaeoclimate modelling during the cenozoic”, *Climate of the Past*, vol. 12, no. 8, pp. 1635–1644, 2016, doi:[10.5194/cp-12-1635-2016](https://doi.org/10.5194/cp-12-1635-2016).
- [4] Fowler, C. M. R., *The Solid Earth: An Introduction to Global Geophysics*. Cambridge University Press, 2 ed., 2004.
- [5] Müller, R. D., Flament, N., Cannon, J., Tetley, M. G., Williams, S. E., Cao, X., Bodur, O. F., Zahirovic, S., y Merdith, A., “A tectonic-rules-based mantle reference frame since 1 billion years ago – implications for supercontinent cycles and plate–mantle system evolution”, *Solid Earth*, vol. 13, no. 7, pp. 1127–1159, 2022, doi:[10.5194/se-13-1127-2022](https://doi.org/10.5194/se-13-1127-2022).
- [6] Müller, R. D., Cannon, J., Qin, X., Watson, R. J., Gurnis, M., Williams, S., Pfaffelmoser, T., Seton, M., Russell, S. H. J., y S., Z., “Gplates: Building a virtual earth through deep time.”, *Geochemistry, Geophysics, Geosystems*, vol. 19, pp. 2243–2261, 2018, doi:[10.1029/2018GC007584](https://doi.org/10.1029/2018GC007584). Disponible en <https://www.gplates.org/>.
- [7] QGIS-Development-Team, “Qgis geographic information system”, QGIS Association, 2024, <https://www.qgis.org>.
- [8] Poblete, F., Dupont-Nivet, G., Licht, A., van Hinsbergen, D., Roperch, P., Mihalynuk, M., Johnston, S., Guillocheau, F., Baby, G., Fluteau, F., Robin, C., van der Linden, T., Ruiz, D., y Baatsen, M., “Towards interactive global paleogeographic maps, new reconstructions at 60, 40 and 20ma”, *Earth-Science Reviews*, vol. 214, p. 103508, 2021, doi:<https://doi.org/10.1016/j.earscirev.2021.103508>.
- [9] Gurnis, M., Yang, T., Cannon, J., Turner, M., Williams, S., Flament, N., y Müller, R. D., “Global tectonic reconstructions with continuously deforming and evolving rigid plates”, *Computers Geosciences*, vol. 116, pp. 32–41, 2018, doi:<https://doi.org/10.1016/j.cageo.2018.04.007>.
- [10] Müller, R. D., Cannon, J., Williams, S., y Dutkiewicz, A., “Pybacktrack 1.0: A tool

for reconstructing paleobathymetry on oceanic and continental crust”, *Geochemistry, Geophysics, Geosystems*, vol. 19, pp. 1898–1909, 2018, doi:10.1029/2017GC007313. Disponible en <https://github.com/EarthByte/pyBacktrack/>.

Anexo

A. Instrucciones de instalación

Código A.1: Instrucciones de instalación dentro del archivo README.md

```
1  Installation
2  =====
3  Some of the dependencies needed to run this plugin are only available through conda.
4  This means that, if you have QGIS installed by other means, you are going to have
5  to reinstall it through conda.
6
7  The process is as follows:
8
9  - First, you need to have conda installed in your system. It is recommended to install it
10 from the [conda-forge distribution](https://conda-forge.org/download/), to ensure the conda-forge
11 channel is available by default. If you installed conda via the Anaconda
12 distribution instead, you need to enable the conda-forge channel manually by
13 running the following command in your terminal:
14
15 ```sh
16 conda config --add channels conda-forge
17 ```
18
19 or adding the following lines to your `.condarc` file (commonly located in the user's home
20 directory):
21
22 ```yaml
23 channels:
24 - conda-forge
25 ```
26
27 - Secondly, you need to clone this repository into the directory where QGIS expects to
28 find its plugins, namely ~C:\Users\{USER}\AppData\Roaming\QGIS\QGIS3\profiles\
29 default\python\plugins` in Windows and ~/home/{USER}/.local/share/QGIS/QGIS3/
30 profiles/default/python/plugins` in Linux.
31
32 ```sh
33 cd /home/{USER}/.local/share/QGIS/QGIS3/profiles/default/python/plugins
```

```
24 git clone https://github.com/LucasAmion/terra-antiqua.git
25 ```
26
27 > Note: For this you git installed in your system. You can download it from here: https
    ://git-scm.com/downloads. Alternatively, you can simply download the code of this
    repository as a ZIP file and extract it in the aforementioned directory.
28
29 - Finally, install all the necessary conda dependencies (listed in the `environment.yml` file)
    by running the following command while inside the root folder of the project:
30
31 ```sh
32 cd terra-antiqua
33 conda env create -n terra-antiqua
34 ```
35
36 This will create a new conda environment called `terra-antiqua` with all the necessary
    dependencies including GPLates and QGIS itself.
37
38 Now, to use the software you just need to activate the environment and run QGIS like so
    :
39
40 ```sh
41 conda activate terra-antiqua
42 qgis
43 ```
44
45 If instead you wish to install the dependencies in an already existing conda environment you
    need to activate said environment and run this command:
46
47 ```sh
48 conda env update --file environment.yml
49 ```
50
51
```