TESIS TESIS TESIS TESIS TESIS

UNIVERSIDAD AUTONOMA
DE AGUASCALIENTES

CENTRO DE CIENCIAS BÁSICAS
DEPARTAMENTO DE SISTEMAS DE INFORMACIÓN

TITULO:

ANÁLISIS DE IMÁGENES RÁSTER DE MÉXICO DERIVADAS DE GEOMEDIANAS UTILIZANDO PYTHON EN CUADERNOS DE JUPYTER

TRABAJO PRÁCTICO QUE PRESENTA LA
L.I. SHEILA JENNY DEL ROCÍO MENDOZA GONZÁLEZ
PARA OPTAR POR EL GRADO DE: MAESTRÍA EN
INFORMÁTICA Y TECNOLOGÍAS COMPUTACIONALES

COMITÉ TUTORAL:
TUTOR: Dr. Juan Muñoz López
CO-TUTOR: M.C. Abel Alejandro Coronado Iruegas
ASESOR: Dr. Luis Eduardo Bautista Villalpando

AGUASCALIENTES, AGS. JUNIO DE 2022.

TESIS TESIS TESIS TESIS TESIS

TESIS TESIS TESIS TESIS TESIS

AUTORIZACIONES

M. EN C. JORGE MARTÍN ALFÉREZ CHÁVEZ
DECANO DEL CENTRO DE CIENCIAS BÁSICAS

_____
P R E S E N T E

Por medio del presente como **Miembros del Comité Tutoral** designado del estudiante **SHEILA JENNY DEL ROCO MENDOZA GONZÁLEZ** con ID **114350** quien realizó el trabajo práctico titulado: **ANÁLISIS DE IMÁGENES RÁSTER DE MÉXICO DERIVADAS DE GEOMEDIANAS UTILIZANDO PYTHON EN CUADERNOS DE JUPYTER**, un trabajo propio, innovador, relevante e inédito y con fundamento en el Artículo 175, Apartado II del Reglamento General de Docencia damos nuestro consentimiento de que la versión final del documento ha sido revisada y las correcciones se han incorporado apropiadamente, por lo que nos permitimos emitir el **VOTO APROBATORIO**, para que ella pueda proceder a imprimirlo así como continuar con el procedimiento administrativo para la obtención del grado.

Ponemos lo anterior a su digna consideración y sin otro particular por el momento, le enviamos un cordial saludo.

A T E N T A M E N T E
"Se Lumen Proferre"
Aguascalientes, Ags., a 15 de junio de 2022.

JUAN MUÑOZ LÓPEZ
Tutor de trabajo práctico

ABEL ALEJANDRO CORONADO IRUEGAS
Co-Tutor de trabajo práctico

LUIS EDUARDO BAUTISTA VILLALPANDO
Asesor de trabajo práctico

c.c.p.- Interesado
c.c.p.- Secretaría Técnica del Programa de Posgrado

TESIS TESIS TESIS TESIS TESIS

TESIS TESIS TESIS TESIS TESIS TESIS

AUTORIZACIONES

universidad autónoma de aguascalientes

DICTAMEN DE LIBERACION ACADEMICA PARA INICIAR LOS TRAMITES DEL EXAMEN DE GRADO

Fecha de dictaminación dd/mm/aaaa: 15/06/22

NOMBRE: SHEILA JENNY DEL ROCIO MENDOZA GONZALEZ **ID** 114350

PROGRAMA: Maestría en Informática y Tecnologías Computacionales **LGAC (del posgrado):** Gestión de sistemas y tecnologías de información para mejorar competitividad, innovación y cambio organizacional

TIPO DE TRABAJO: ( ) Tesis (X) Trabajo Práctico

TITULO: ANALISIS DE IMAGENES RASTER DE MEXICO DERIVADAS DE GEOMEDIANAS UTILIZANDO PYTHON EN CUADERNOS DE JUPYTER

IMPACTO SOCIAL (señalar el impacto logrado): DARA CONOCER AL PUBLICO EN GENERAL EL CUBO DE DATOS GEOESPACIALES DE MEXICO DESARROLLADO EN EL INEGI, PARA SU USO Y EXPLOTACIÓNDE INFORMACIÓN CON EL FIN DE AYUDAR A UNA MEJOR TOMA DEDECISIONES O ADQUISICIÓN DE CONOCIMIENTO DEL MISMO DE QUIEN LO REQUIERA CON AYUDADE LOS CUADERNOS DE JUPYTER

INDICAR SI NO N.A. (NO APLICA) SEGÚN CORRESPONDA:	
Elementos para la revisión académica del trabajo de tesis o trabajo práctico:	
SI	El trabajo es congruente con las LGAC del programa de posgrado
SI	La problemática fue abordada desde un enfoque multidisciplinario
SI	Existe coherencia, continuidad y orden lógico del tema central con cada apartado
SI	Los resultados del trabajo dan respuesta a las preguntas de investigación o a la problemática que aborda
SI	Los resultados presentados en el trabajo son de gran relevancia científica, tecnológica o profesional según el área
SI	El trabajo demuestra más de una aportación original al conocimiento de su área
SI	Las aportaciones responden a los problemas prioritarios del país
SI	Generó transferecia del conocimiento o tecnológica
SI	Cumple con la ética para la investigación (reporte de la herramienta antiplagio)
El egresado cumple con lo siguiente:	
SI	Cumple con lo señalado por el Reglamento General de Docencia
SI	Cumple con los requisitos señalados en el plan de estudios (créditos curriculares, optativos, actividades complementarias,estancia, predoctoral, etc)
SI	Cuenta con los votos aprobatorios del comité tutoral, en caso de los posgrados profesionales si tiene solo tutorpodrá liberar solo el tutor
SI	Cuenta con la carta de satisfacción del Usuario
SI	Coincide con el titulo y objetivo registrado
SI	Tiene congruencia con cuerpos académicos
SI	Tiene el CVU del Conacyt actualizado
N.A.	Tiene el artículo aceptado o publicado y cumple con los requisitos institucionales (en caso que proceda)
En caso de Tesis por artículos científicos publicados	
N.A.	Aceptación o Publicación de los artículos según el nivel del programa
N.A.	El estudiante es el primer autor
N.A.	El autor de correspondencia es el Tutor del Núcleo Académico Básico
N.A.	En los artículos se ven reflejados los objetivos de la tesis, ya que son producto de este trabajo de investigación.
N.A.	Los artículos integran los capítulos de la tesis y se presentan en el idioma en que fueron publicados
N.A.	La aceptación o publicación de los artículos en revistas indexadas de alto impacto

Con base a estos criterios, se autoriza se continúen con los trámites de titulación y programación del examen de grado: Sí X / No

FIRMAS

Elaboró:

* NOMBRE Y FIRMA DEL CONSEJERO SEGÚN LA LGAC DE ADSCRIPCION: Dr.Cesar Eduardo Velázquez Amador

NOMBRE Y FIRMA DEL SECRETARIO TÉCNICO: MTIC Jorge Eduardo Macías Luévano

* En caso de conflicto de intereses, firmará un revisor miembro del NAB de la LGAC correspondiente distinto al tutor o miembro del comité tutoral, asignado por el Decano.

Revisó:

NOMBRE Y FIRMA DEL SECRETARIO DE INVESTIGACIÓN Y POSGRADO: Dra. Hayde Martínez Ruvalcaba

Autorizó:

NOMBRE Y FIRMA DEL DECANO: M. en C. Jorge Martín Alférez Chávez

Nota: procede el trámite para el Depto. de Apoyo al Posgrado

En cumplimiento con el Art. 105C del Reglamento General de Docencia que a la letra señala entre las funciones del Consejo Académico: ... Cuidar la eficiencia terminal del programa de posgrado y el Art. 105F las funciones del Secretario Técnico, llevar el seguimiento de los alumnos.

Elaborado por: D. Apoyo al Posg.
Revisado por: D. Control Escolar/D. Gestión de Calidad.
Aprobado por: D. Control Escolar/ D. Apoyo al Posg.

Código: DO-SEE-FO-15
Actualización: 01
Emisión: 28/04/20

TESIS TESIS TESIS TESIS TESIS TESIS

TESIS TESIS TESIS TESIS TESIS

AGRADECIMIENTOS

Agradezco profundamente al Dr. Juan Muñoz López por su confianza en mi persona y por haberme adoptado en su área de trabajo para llevar a cabo un proyecto tan importante para el INEGI como es la explotación de la información de las imágenes ráster contenidas en el Cubo de Datos Geoespacial de México y por gestionar todos los recursos necesarios para llevarlo a cabo.

De igual manera agradezco al equipo de trabajo del Dr. Juan Muñoz, entre ellos a Silvia Fraustro, Abel Coronado y a Armando Soto por el apoyo recibido para llevar a cabo tan importante proyecto.

También agradezco al Ing. José Luis Mondragón por su apoyo y por permitir mi participación en dicho proyecto.

A todos ustedes.. ¡¡ Muchas gracias !!

TESIS TESIS TESIS TESIS TESIS

TESIS TESIS TESIS TESIS TESIS

DEDICATORIA

A mis hijos, Aimée y Darien, quienes son mi motivo para mejorar cada día y son quienes me impulsan a realizar mis sueños…

Para mí, porque una vez más, me he demostrado que puedo alcanzar mis sueños a pesar de las críticas negativas de las demás personas y de mis limitaciones, entre ellas la edad y la salud.

TESIS TESIS TESIS TESIS TESIS

ÍNDICE GENERAL

TESIS TESIS TESIS TESIS TESIS

ÍNDICE DE TABLAS

ÍNDICE DE GRAFICAS O FIGURAS

FIGURA ***Pág.***

TESIS TESIS TESIS TESIS TESIS

RESUMEN EN ESPAÑOL

Un Cubo de Datos Geoespaciales es un arreglo sistemático que facilita a usuarios el acceso a las imágenes satelitales de interés, por lugar y fecha de captación específicos, para su uso y/o procesamiento directo. El cubo de Datos Geoespacial de México, producto realizado por el INEGI, contiene las imágenes ráster, las cuales son fotografías satelitales de terrenos con dimensiones de kilómetros cuadrados, de toda la República Mexicana desde los años 1993 hasta el año 2019. El presente trabajo trata de cómo realizar análisis ráster de dichas imágenes por medio de implementación de cuadernos de jupyter programados en lenguaje Python. Dichos cuadernos han sido elaborados de una manera sencilla de entender, con el fin de que el usuario pueda interpretar dicha información contenida en el cuaderno y se interese no sólo por aprender a programar en Python sino a analizar las imágenes satelitales de nuestro país.

TESIS TESIS TESIS TESIS TESIS

TESIS TESIS TESIS TESIS TESIS

ABSTRACT

A Geospatial Data Cube is a systematic arrangement that provides users with access to the satellite images of interest, by specific place and date of capture, for their use and / or direct processing. The Geospatial Data cube of Mexico, a product made by INEGI, contains raster images, which are satellite photographs of land with dimensions of square kilometers, of the entire Mexican Republic from 1993 to 2019. This work deals with of how to perform raster analysis of said images by means of the implementation of jupyter notebooks programmed in Python language. Said notebooks have been developed in a way that is easy to understand, so that the user can interpret the information contained in the notebook and become interested not only in learning to program in Python but also in analyzing the satellite images of our country.

TESIS TESIS TESIS TESIS TESIS

TESIS TESIS TESIS TESIS TESIS

INTRODUCCIÓN

Las necesidades de las organizaciones para cumplir sus metas y llegar a tomar las mejores decisiones han planteado retos en cuanto al análisis de la información, ya sea por competitividad, valor agregado, costos o ganancia; tener una base de datos de todos los movimientos transaccionales realizados en un histórico de tiempo se volvió algo muy importante. Sumado a esto, surgió un factor que permitiría tener control sobre el lugar de los sucesos y así situar los esfuerzos donde en realidad se necesitan, este factor se identifica con el espacio geográfico almacenado en las bases de datos como tipo de dato geométrico o espacial. A partir de este tipo se crean nuevas formas de análisis y manejo de los datos como la inteligencia de negocios, los cubos de datos, las consultas, modelado y minería de datos enfocados en el descubrimiento de conocimiento espacial.

Un Cubo de Datos Geoespaciales es un arreglo sistemático que facilita a usuarios el acceso a las imágenes satelitales de interés, por lugar y fecha de captación específicos, para su uso y/o procesamiento directo.

México, está a la vanguardia en el uso de tecnologías y sistemas para el levantamiento de información geoespacial, pero hace falta “mayor cohesión” y una “visión integradora” entre las distintas dependencias para aprovecharlas mejor y solucionar múltiples problemas que van desde el registro del catastro a la ubicación de áreas de desarrollo territorial. (La Jornada, 2019).

El INEGI cuenta con el Marco de integración de información geoespacial y el Cubo de datos geoespaciales de México, en el cual se tiene Imágenes LANDSAT proporcionadas por la NASA, que van del período de 1985 a 2018 para ciertas áreas de Interés. Este volumen de imágenes ráster es de aprox. 3.2 TB.

Ahora bien, sabiendo todo lo anterior, en el presente trabajo práctico, se va a plantear cómo poder explotar toda la información valiosa contenida en el Cubo de Datos Geoespaciales de México para el beneficio de la sociedad mexicana.

TESIS TESIS TESIS TESIS TESIS

1. PLANTEAMIENTO DE LA PROBLEMÁTICA A ATENDER

Actualmente el INEGI cuenta con un producto llamado Cubo de Datos Geoespacial de México, el cual contiene imágenes ráster de toda la República Mexicana, las cuales fueron capturadas desde el año de 1993 hasta el año 2019 por los satélites LandSat propiedad de la NASA.

Dichas imágenes ráster ya mencionadas, son de gran importancia ya que por medio de ellas se pueden hacer análisis ráster de cierto territorio dentro de nuestro país, dichos análisis pudiendo ser los cambio que se ha tenido durante los años en dicho terreno, por lo que el INEGI busca que esta información pueda ser consultada y analizada no sólo por especialistas con cierto expertiz en el área de la geografía, sino también se desea que sea consultada y analizada por el público en general.

2. JUSTIFICACIÓN PARA REALIZAR EL TRABAJO PRÁCTICO

Con el desarrollo e implementación de cuadernos de jupyter por parte del INEGI, cualquier persona podrá obtener y analizar información, de manera sencilla información de datos ráster obtenidas del cubo de datos geoespacial con imágenes satelitales de México, con las siguientes ventajas:

- Minimizar el tiempo y el conocimiento especializado requerido para aprovechar las imágenes de satélite de México.
- Menos procesamiento (datos Listos para el análisis)
- Menos almacenamiento para el usuario (descargas específicas de área)
- Organización automática (ingesta/indexado)
- Acceso sencillo y eficiente a la información

3. OBJETIVOS DE LA INTERVENCIÓN

7.1. OBJETIVO GENERAL:

Implementar una herramienta sencilla y de fácil entendimiento que permita a los usuarios la explotación de la información de imágenes ráster de México, las cuales son derivadas de Geomedianas contenidas en el cubo de datos de INEGI.

7.2. OBJETIVOS ESPECÍFICOS:

1. Desarrollar procedimientos sencillos por medio de scripts desarrollados con código abierto y gratuito (Python y herramientas de código abierto para procesar datos espaciales) para obtener información de datos ráster del cubo de datos geoespacial de México, para que cualquier persona pueda realizar procesamiento y análisis masivo en:
    - El espacio: regiones, países enteros.
    - En el tiempo: días, semanas, meses, años, décadas.

2. Implementar una plataforma de análisis eficiente de series de tiempo para soportar aplicaciones de cambio en la tierra a nivel local, regional y nacional.

3. Para el análisis de series de tiempo, obtener la información de los datos ráster “un píxel a la vez”.

4. Capacitar a instituciones de gobiernos, instituciones educativas del país y al personal de INEGI que así lo requiera, en el uso de esta plataforma mencionada

TESIS TESIS TESIS TESIS TESIS

TESIS TESIS TESIS TESIS TESIS

4. FUNDAMENTACIÓN TEÓRICA

4.1. BASES DE DATOS GEOESPACIALES

4.1.1. Bases de datos geoespaciales

"El trabajo sobre la gestión global de la información geoespacial en los últimos años ha Confirmado que uno de los desafíos clave es una mejor integración de la información geoespacial y estadística como base para una toma de decisiones sólida y basada en la evidencia."(HonGbo, 2012)

Actualmente nos movemos en un mundo en el que los datos nos invaden. Tenemos multitud de datos referidos a cualquier tema que nos podamos imaginar. A la hora de trabajar con ellos, es muy importante saber cuáles son los datos interesantes, que características tienen, cómo se relacionan entre sí, es imprescindible saber manejarlos para poder convertirlos en INFORMACIÓN.

Una Base de datos es una colección de elementos o datos interrelacionados que se pueden procesar por uno o más sistemas de aplicación. (Cecilia et al., n.d.) Esta funcionalidad ayuda a evitar:

- Redundancia de los datos.- datos que deben aparecer una y solo una vez en el sistema.
- Pobre control de los datos.- el mismo elemento de los datos suele tener varios nombres, dependiendo del archivo en el que esté contenido.
- Capacidad inadecuada de manipulación de los datos.- Los archivos indexados, permiten tener un control sobre las consultas, existencia de identificadores únicos.
- Programación excesiva.- para hacer consultas y manipulación más rápida y eficiente.

Ventajas de diseñar y trabajar con una base de datos representa:

TESIS TESIS TESIS TESIS TESIS

- Tener datos estructurados.
- Tener datos y procesos de división, esto significa una mayor dependencia del dato y mayor flexibilidad de procesamiento.
- Integridad de los Datos, esto es una consistencia de los datos así como seguridad y protección de los mismos.
- Datos de larga vida y duración.

Toda base de datos requiere de un manejador de base de datos y en el mercado existen muchos que acorde con las necesidades de cada usuario ofrecen ventajas y desventajas sobre el control de los datos; éstos manejadores se conforman de una base de datos, un sistema computacional que manipula las bases de datos y el hardware.

Toda base de datos requiere un diseño que incluya a los usuarios, sus necesidades, los datos, así como la estructura de la organización para la cual estará diseñada. En el contexto de los Sistemas de Información Geográfica una base de datos es un componente primordial ya que permite transformar los datos en información.

Las bases de datos espaciales se integran completamente con la base de datos relacional de objetos. (Cecilia et al., n.d.) Bajo estas bases surge el concepto de GEODATABASE que es un modelo genérico para el tratamiento de información geoespacial, en el cual se almacena objetos geográficos, sus atributos, sus relaciones (espaciales o no) y el comportamiento de cada uno de sus elementos.

En este esquema se migra de capas temáticas (colección de elementos geográficos) a entidades reales como red de carreteras, red de alcantarillado, energía eléctrica, etc. y se incorporan el concepto de entidad cuando nos referimos a transformadores, carreteras o lagos. La utilidad de este concepto refleja el tratamiento de nuestros datos para realizar análisis espacial.

Una base de datos espaciales nos permite el almacenamiento de las geometrías de un archivo cartográfico dentro de una base de datos, de modo que podamos almacenar y

TESIS TESIS TESIS TESIS TESIS

analizar estos datos. *(Cecilia et al., n.d.)* Permitiendo manipular toda nuestra información geográfica de una manera mucho más rápida y eficaz que una base de datos común, y además, nos permite realizar muchos análisis como un software SIG de escritorio. *(Asociación Geoinnova, n.d.)*

El Geodatabase es un modelo con mucha flexibilidad que permite modelar la realidad, a través de la cual la integridad de la información tanto en una base de datos alfanumérica como espacial es mucho más sencilla debido a las características de centralización, la implementación del modo comportamiento y la forma multiusuario de trabajo.

4.1.2. ¿Qué es lo más reciente en base de datos geoespaciales?

Partiendo de la importancia de los datos, su gestión tanto de la parte alfanumérica como espacial requiere especialización. Una base de datos espaciales nos permite manipular toda nuestra información geográfica de una manera mucho más rápida y eficaz que una base de datos común, y además, nos permite realizar muchos análisis como un software SIG de escritorio.

Un Sistema de Información Geográfica, SIG, son herramientas que permiten also usuarios crear consultas itneractivas, analizar la información espacial, editar datos, mapas y presentar los reusltados de todas estas operaciones. En el sentido más estricto, es cualquier sistema de información capaz de integrar, almacenar, editar, analizar, compartir y mostrar la inforamción geográficametne referenciada.(Jordi Vivancos Martí, Albert Llastarri, Mònica Grau, n.d.)

Actualmente tenemos muchas herramientas que nos ayudan a gestionar todos estos datos, pero concretamente las bases de datos PostGIS (figura 1) son una gran ayuda a la hora de tratar datos espaciales.

Podríamos tener infinidad de ejemplos en este campo:

1. Consultas geográficas múltiples

2. Qué espacios naturales atraviesan a una determinada distancia de una carretera
3. Cuánta población hay en una determinada zona de influencia
4. Cuántos lugares de interés hay en una determinada zona
5. Cuál es la zona con menor impacto para situar una determinada fábrica

PostGIS nos ofrece herramientas, no sólo para trabajar de forma eficiente con grandes tablas alfanuméricas ya que hereda las características de PostgreSQL, sino para hacer tratamiento y análisis de datos espaciales.

Existe un gran número de herramientas GIS para escritorio que permite visualizar los datos PostGIS como, por ejemplo, QGIS, ArcGIS, o gvSIG así como servidores de mapas web, como MapServer o ArcGIS Server. (Asociación Geoinnova, n.d.)

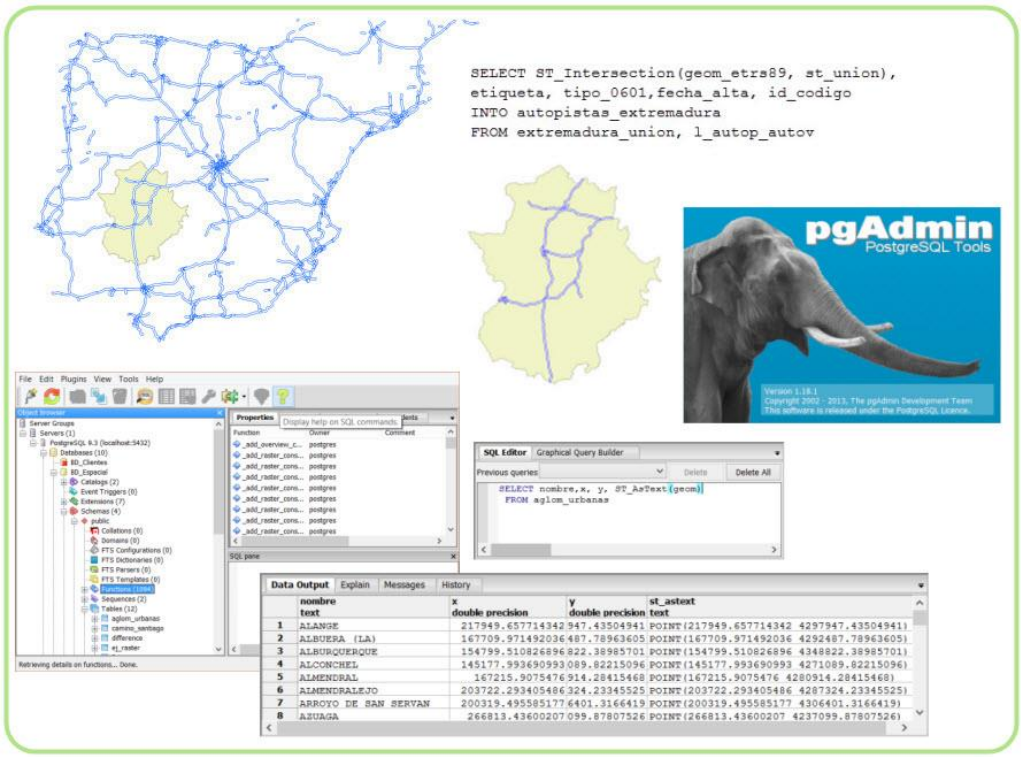


Figura 1. Bases de datos PostGIS desde la herramienta pgAdmin.

4.1.3. Tipos de Datos (Geodatos) en bases de datos geoespaciales

Antes de traer Geodato a cualquiera de estos programas de procesamiento (SIG), es importante mencionar cómo se extrae la realidad percibida al ambiente dentro de estos sistemas. Entre las maneras de codificar o representar la geografía (reducir la realidad percibida de los elementos geográficos a cierto nivel de abstracción) están: (*QGis Documentation*, n.d.)

1. **Vectorial**: Proporciona una manera de representar "objetos espaciales" del mundo real dentro de un ambiente SIG. Un objeto espacial es algo que puede ver en el paisaje. Conformado por tres niveles geométricos:

   - **Punto**: consiste en un solo vértice, representando objetos puntuales como aeropuerto, pozo, escuela, etc. Figura 2.

   - **Línea o miltilínea (polyline)**: consiste en dos o más vértices, el primer y último vértice no son iguales. Ejemplo: corrientes de agua, carreteras, caminos, línea de transmisión eléctrica, etc. Figura 3.

   - **Área**: se forma cuando tres vértices se encuentran presentes, el último vértice es igual al primero. Son utilizados para representar superficies, por ejemplo, áreas de cultivo, delimitación de territorio, edificaciones, etc. Figura 4.

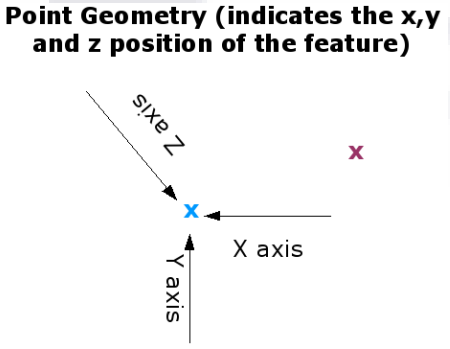


Figura 2. Dato vectorial tipo punto.

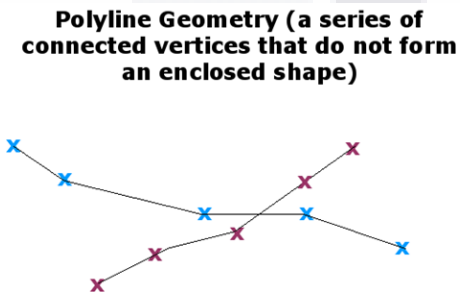


Figura 3. Dato vectorial tipo línea.

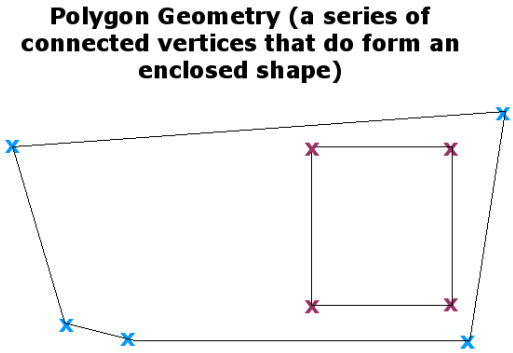


Figura 4. Vector tipo área.

- **Ráster**: Los datos ráster se componen de una matriz de píxeles (también llamadas celdas), cada uno con un valor que representa las condiciones de la zona cubierta por dicha celda. Un conjunto de datos ráster está compuesto de filas (corriendo de un lado a otro) y columnas (corriendo hacia abajo) de píxeles (también conocidos como celdas). Cada píxel representa una región geográfica, y el valor en ese píxel representa alguna característica de dicha región. El ejemplo más común del uso de ráster para representar geografía es la fotografía aérea. Figura 5. Dado que la matriz es regular (el tamaño del píxel es constante) y que conocemos la posición en coordenadas del centro de una de las celdas, se puede decir que todos los píxeles están georreferenciados. (Jordi Vivancos Martí, Albert Llastarri, Mònica Grau, n.d.)

Figura 5. Dato ráster.

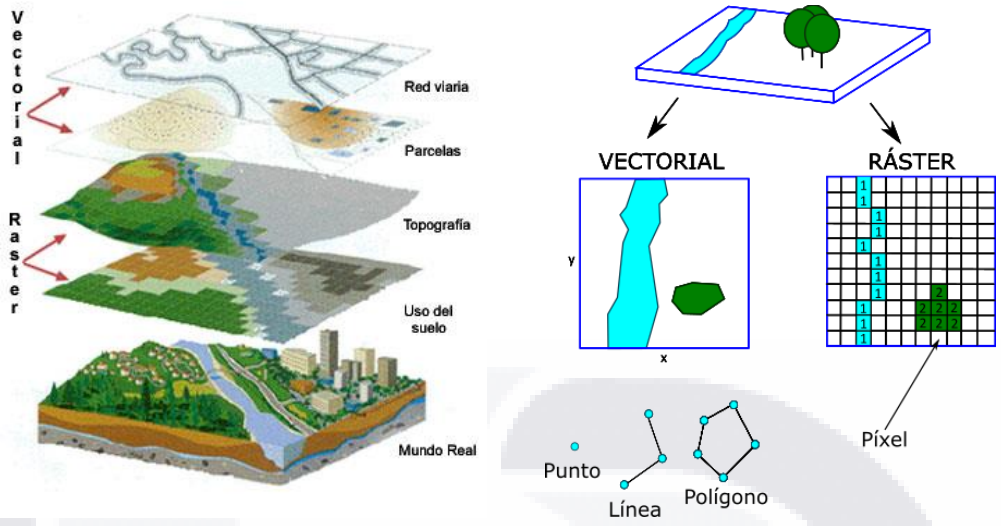


Figura 6. Ejemplo de capas temáticas de un SIG.

4.1.4. Aplicaciones de los SIG

La mayor utilidad de un sistema de información geográfico está íntimamente relacionada con la capacidad de visualizar datos de forma gráfica y de construir modelos o representaciones del mundo real a partir de integrar y combinar datos de diversa naturaleza dentro de un marco territorial. Figura 6.

Estos modelos, son muy útiles para la simulación de los efectos que produce sobre un determinado territorio, un proceso natural o una acción humana.

Los SIG contribuyen al análisis y aportan soluciones para un amplio rango de necesidades como, por ejemplo:

- Producción y actualización de la cartografía básica
- Administración de servicios públicos (suministro de agua, energía, comunicaciones, saneamiento, entre otros)
- Regulación del uso del suelo
- Catastro
- Atención de emergencias: incendios, terremotos, accidentes de tránsito, etc.
- Estratificación socioeconómica

TESIS TESIS TESIS TESIS TESIS

- Gestión medioambiental: saneamiento básico ambiental y mejora de las condiciones ambientales
- Evaluación de áreas de riesgos (prevención y atención de desastres)
- Localización óptima de las infraestructuras y equipamientos sociales
- Diseño y mantenimiento de la red viaria
- Formulación y evaluación de planes de desarrollo social y económico.

4.1.5. Cartografía y Teledetección

Actualmente, la Teledetección se considera una fuente de información más a un SIG. Como se mencionó anteriormente, los SIG poseen la capacidad de combinar imágenes de Teledetección (formato ráster) con datos vectoriales (Figura 6). Además, los modelos digitales de elevación del terreno (MDE), mejoran notablemente la capacidad de visualización del territorio. Los satélites de teledetección en alta resolución, y los sistemas GPS, aportan datos de gran precisión y actualizados para mejorar la cartografía convencional. Un ejemplo de las aplicaciones prácticas para el gran público de todas estas tecnologías, es la reciente aparición de Google Maps y Google Earth. Sistemas de Información Geográfica de manejo muy sencillo, que combinan mapas e imágenes satélite.

Los SIG con información ráster, aunque también puede abordarse de forma independiente, dentro de proyectos llamados cubo de datos geoespaciales, los cualés pocos países ya tienen bien establecido bien este concepto y lo están levando a cabo en sus naciones. Un ejemplo de ello, Water Observations from Space (WOfS) en Austgralia.

4.2. CUBO DE DATOS GEOESPACIALES ACTUALES

Un Cubo de Datos Geoespaciales es un arreglo sistemático que facilita a usuarios el acceso a las imágenes satelitales de interés, por lugar y fecha de captación específicos, para su uso y/o procesamiento directo.

Water Observations from Space (WOfS) es un servicio web que muestra observaciones históricas de aguas superficiales derivadas de imágenes satelitales para toda Australia desde 1987 hasta la actualidad, todo esto por medio de cubos de datos geoespaciales. El objetivo de WOfS es permitir una mejor comprensión de dónde suele estar presente el agua; donde rara vez se observa; y donde la inundación de la superficie ha sido observada ocasionalmente por satélite. WOfS muestra el agua superficial detectada del archivo de imágenes satelitales Landsat 5 y Landsat 7 en toda Australia. (Geoscience Australia, n.d.)

Para cada celda de la cuadrícula dentro del mapa, WOfS muestra:

- El número de observaciones satelitales claras durante el período (1987 hasta el presente)
- El número de ocasiones en que se detectó agua
- El porcentaje de observaciones claras sobre las cuales se detectó agua, y
- La confianza (o probabilidad) de que una observación de agua en este lugar es correcta. Este es un porcentaje, basado en una serie de factores que incluyen la pendiente de la tierra y la existencia de otra evidencia corroborativa.

El proyecto WOfS comenzó en 2011 e incluyó lanzamientos de información por etapas y un producto de prueba. El desarrollo de este producto ahora está completo. Los datos se seguirán actualizando cada tres meses.

4.2.1. Imágenes satelitales: Colección Landsat para México

LANDSAT: Misión satelital de observación de la Tierra más antigua en operación, generando imágenes desde 1972 hasta la actualidad, capturando imágenes en 12 bandas distintas con una resolución máxima de 15m.

La constelación de satélites LANDSAT (LAND=tierra y SAT=satélite), que inicialmente se llamaron ERTS (Earth Resources Technology Satellites), fue la primera misión de los Estados Unidos para el monitoreo de los recursos terrestres. La forman 8 satélites de los cuales sólo se encuentran activos el 5 y el 8. Su mantenimiento y operación está a cargo de la Administración Nacional de la Aeronáutica y del Espacio (NASA) en tanto que la

TESIS TESIS TESIS TESIS TESIS

producción y comercialización de las imágenes depende del Servicio Geológico de los Estados Unidos (USGS).

Los satélites LANDSAT llevan a bordo diferentes instrumentos. Su evolución buscó siempre captar más información de la superficie terrestre, con mayor precisión y detalle, de ahí las mejoras radiométricas, geométricas y espaciales que se incorporaron a los sensores pasivos; el primero, conocido como Multispectral Scanner Sensor (MSS), seguido de Thematic Mapper (TM) que tiene mayor sensibilidad radiométrica que su antecesor, Enhanced Thematic Mapper Plus (ETM+) que entre sus mejoras técnicas destaca una banda espectral (pancromática) con resolución de 15 metros.

El último satélite Landsat, Landsat-8 lleva a bordo dos sensores de imágenes Operational Land Imager (OLI) y Thermal Infrared Sensors /TIIRS).

Las imágenes Landsat, de acuerdo al sensor pueden estar compuestas de 4 (Landsat-1,2,3) hasta 11 bandas espectrales (Landsat-8). Dependiendo del satélite y sensor, incluyen un canal pancromático, rango visible, una o más bandas infrarrojas, y una o dos bandas térmicas. Las resoluciones espaciales varían entre 15, 30, 60 y 120 según el sensor y banda. (Emilio Chuvieco, n.d.)

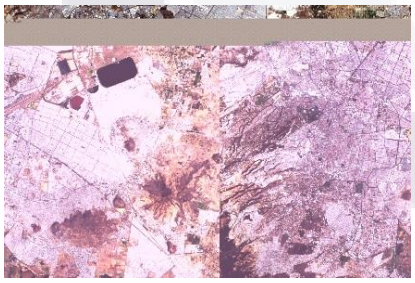


Figura 7. Imagen LANDSAT7 formada por las bandas 1,2 y 3, con 30 m. de resolución espacial, que muestra la Ciudad de México. Fuente: INEGI

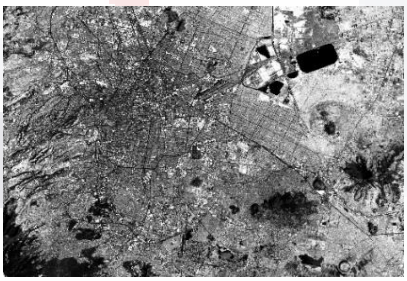


Figura 8. Imagen LANDSAT7 pancromática con 15 m. de resolución espacial, que muestra la Ciudad de México. Fuente: INEGI

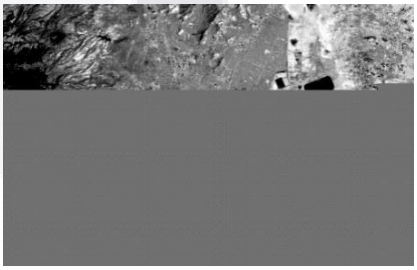


Figura 9. Imagen LANDSAT7 térmica con 60 m. de resolución espacial, que muestra la Ciudad de México. Fuente: INEGI.

Son de utilidad para el monitoreo de la vegetación, aplicaciones geológicas, en el estudio de los recursos naturales y de cultivo.

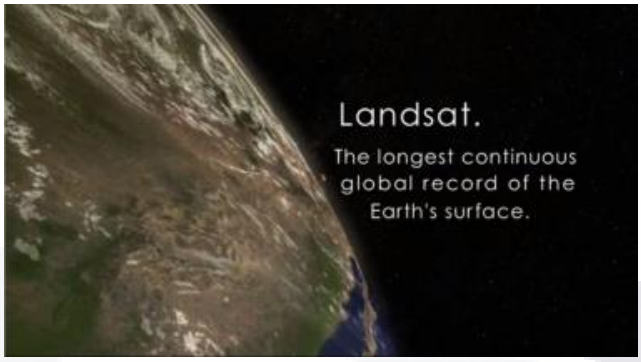


Figura 10: Imagen promocional del Satélite LANDSAT, autor NASA.

Disponibilidad: El acervo de imágenes Landsat en su totalidad, desde 1976 a la fecha, es de acceso gratuito a cualquier usuario a través de la página de USGS https://earthexplorer.usgs.gov/. *(https://earthexplorer.usgs.gov/, n.d.)*

4.2.2. Cubo de Datos Geoespaciales en México

Actualmente, nuevos usuarios encuentran en la información geoespacial oportunidades para futuras estrategias de sus negocios, México está a la vanguardia en el uso de tecnologías y sistemas para el levantamiento de información geoespacial, pero hace falta “mayor cohesión” y una “visión integradora” entre las distintas dependencias para aprovecharlas mejor y solucionar múltiples problemas que van desde el registro del catastro a la ubicación de áreas de desarrollo territorial, señaló el presidente del INEGI Santaella. (La Jornada, 2019).

¿Qué es el INEGI?

Es el Instituto Nacional de Estadística y Geografía (INEGI), institución pública autónoma y responsable de normar y coordinar el Sistema Nacional de Información Estadística y Geográfica, así como de captar y difundir información de México en cuanto al territorio, los

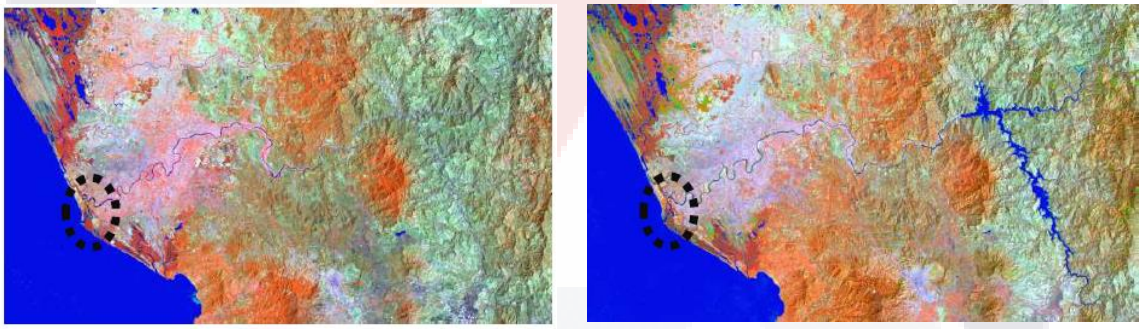
recursos, la población y economía, que permita dar a conocer las características de nuestro país y ayudar a la toma de decisiones.

El 25 de enero de 1983 se creó, por decreto presidencial y con su su creación, se modernizó la valiosa tradición que tenía nuestro país en materia de captación, procesamiento y difusión de información acerca del territorio, la población y la economía. Conjuntó en una sola institución la responsabilidad de generar la información estadística y geográfica del país.

Área de oportunidad

México, a través del Instituto Nacional de Geografía (El INEGI) cuenta con el Marco de integración de información geoespacial y el Cubo de datos geoespaciales de México con el que pretende visualizar imágenes de satélite de manera sencilla y que facilita el análisis con series de tiempo, encontrándose aquí una gran área de oportunidad por medio de la implementación de un Cubo de Datos Geoespacial.

Un ejemplo de lo que se busca obtener, son las series de tiempo en la erosión costera en la boca del Río Santiago. Figura 11 (Fuente INEGI).



1986 1995

Figura 11. Erosión costera en la boca del Río Santiago en 1986 y en 1995.

Para implementar la herramienta del Cubo de Datos Geoespacial de imágenes satelitales de México, el Instituto cuenta con los siguientes insumos: infraestructura computacional, arquitectura específica y perspectiva de uso transversal en el Instituto (Francisco Javier Nava, 2019), y además de contar con un catálogo de 140 imágenes satelitales (Figura12)

TESIS TESIS TESIS TESIS TESIS

proporcionadas por la NASA las cuales fueron capturadas por el satélite Landsat, el cual se explicó anteriormente.

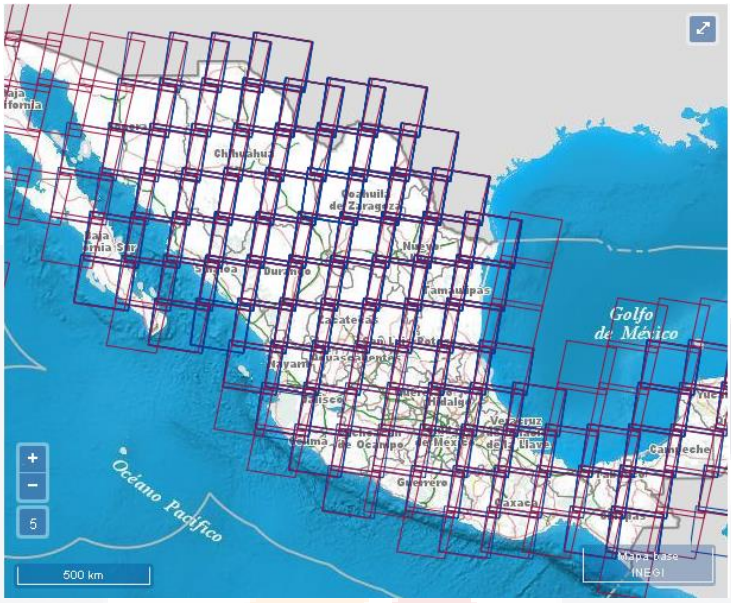


Figura 12. El Territorio mexicano de casi 2 millones de km2 es cubierto por 140 imágenes satelitales..(Francisco Javier Nava, 2019)

Figura 13. Cubo de Datos Geoespacial INEGI (Francisco Javier Nava, 2019)

La implementación de la herramienta del Cubo de Datos Geoespacial de imágenes satelitales de México, representa un gran reto echarlo a andar para el beneficio de cualquier tipo de usuario pues le será de gran utilidad en toma de decisiones.

TESIS TESIS TESIS TESIS TESIS

4.3. JUPYTER NOTEBOOK

4.3.1. Jupyter Notebook

Los cuadernos de Jupyter o mejor conocidos como Jupyter Notebook, es una herramienta (interfaz web) bastante buena la cual está compuesta de código abierto, la cual permite la ejecución de código por medio del navegador y además permite incluir no solamente texto, sino también audio e imágenes.. Estos cuadernos pueden ser desarrollados o programados en diferentes lenguajes, siendo el más común el lenguaje Python. Los cuadernos de Jupyter se ejecutan utilizando un Kernel, es decir, se comunican con un núcleo de cálculo, principalmente de Python. Cabe mencionar que el aspecto de código abierto ha permitido que se pueda disponer de otros tipos de núcleos como lo son los Java, Ruby, Oracle, C, C++, R, Julia, Octave, Fortran, , R, Haskell, SageMath, Scala, Matlab y Mathematica.

La facilidad en el uso de los cuadernos de Jupyter ha permitido que sea cada vez más popular su uso entre los investigadores, los docentes y en especial entre los alumnos, ya que con los cuadernos ya mencionados se pueden construir documentos interactivos facilitándoles el aprendizaje y la exploración de temas científicos. En resumen, los cuadernos de Jupyter son una herramienta de gran utilidad y de apoyo en trabajo individual de los alumnos y una gran herramienta de apoyo docente para los profesores para la explicación de sus temas.

4.3.2. Jupyter Notebooks, sus orígenes

El origen de Jupyter Notebook se debe a que es una evolución de la interfaz Ipython Notebook, contiene las mismas funcionalidades, pero aún mejor ya que permite ejecutar código en múltiples lenguajes. Esta nueva interfaz web, es el desarrollo de un cuaderno computacional pero en software libre, teniendo el mismo estilo que presentan los cuadernos que se utilizan en Maple o en el programa Mathematica,

Cabe mencionar que el primer intento de cuaderno fue desarrollado por William Stein, y fue para el programa de cálculo Sage (Sage Notebooks). Dicho programa a cargo de Fernando Pérez y Robert Kern, en la Universidad de Berkeley en California, Estados Unidos. Para

crear el cuaderno se basaron en el lenguaje de programación Python, debido a su gran versatilidad y su carácter libre, siendo esta característica que la comunidad científica lo adoptara.

En un inicio, la intensión era sólo en crear un intérprete de comandos interactivo en lenguaje Python (a quien antes se le llamaba IPython) y no en crear una interfaz web. Fue en el año de 2010, que el equipo de desarrollo de IPython desarrolla la interfaz web de Jupyter Notebook o cuadernos de Jupyter. A partir de entonces, tanto el desarrollo como el uso de estos cuadernos de Jupyter ha ido en ascenso, siento tal que GitHub (la plataforma de desarrollo de proyectos libres más importante actualmente) permite la visualización de Jupyter Notebooks directamente en su página web, teniendo en su alojamiento más de un millón de estos cuadernos de Jupyter, permitiendo el acceso en sus servidores.

4.3.3. Descripción de la interfaz de Jupyter Notebook

Como primer paso, al ejecutar un cuadernos de Jupyter o Jupyter Notebook en el equipo de cómputo, en el navegador web se abrirá una página llamada Jupyter Dashboard, en la cual se mostrará una lista de archivos disponibles en la carpeta en donde se haya ejecutado programa, se mostrará las extensiones instaladas en el Jupyter Notebook, como lo son el gestor de cursos y corrección automática Nbgrader. En la página ya mencionada, se muestra un botón en la esquina superior derecha, el cual sirve para subir archivos al servidor, pero cuando se ejecuta Jupyter Notebook de manera local, este botón no presenta una gran funcionalidad, pero sí cuando se accede a estos documentos desde otro equipo de cómputo y a través del navegador.

¿Qué se hace para generar el primer Jupyter Notebook?, se da click en el botón New y se elegirá el núcleo de cálculo de la lista de núcleos que se muestren que estén instalados. Por defecto, sólo se mostrará el núcleo de cálculo de Python. En la figura 14 se muestra el ejemplo de un documento tipo Jupyter Notebook. Figura 14.

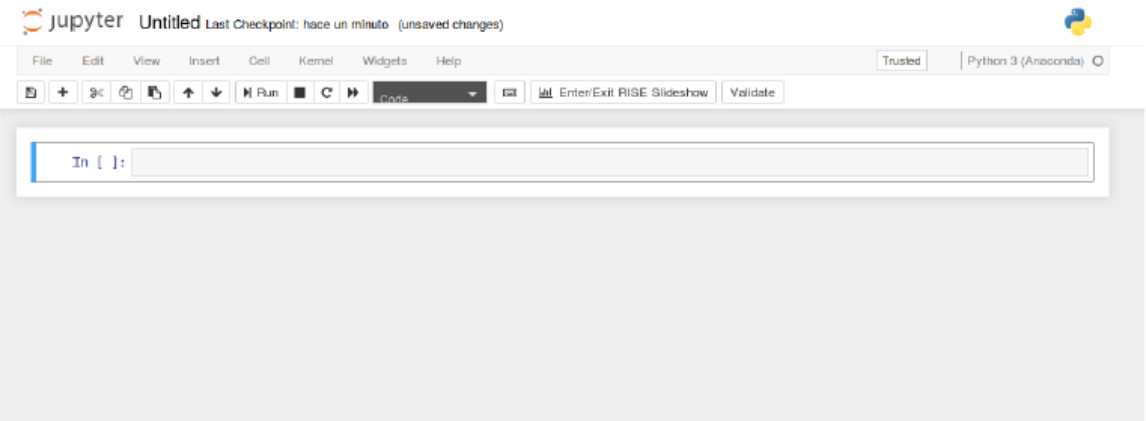


Figura 14. Ejemplo de un documento Jupyter Notebook.

En el documento o cuaderno de Jupyter, en la parte superior se muestra los diferentes tipos de menús y de botones, como son para edición, para descargar el documento en varios formatos como el formato tipo pdf, para insertar o borrar partes del documento, elección y el control del núcleo de cálculo (botón para parar la ejecución del núcleo), un botón de ayuda con la documentación y descripción necesaria de Jupyter Notebook. El documento se divide en diferentes celdas, cada celda puede ser de diferente tipo, ya sea de tipo texto o de tipo código. Para las celdas de tipo texto, se tiene la selección de diferentes opciones como lo son: título, el cual ofrece diferentes tamaños de letra para los títulos, texto ya sea plano o enriquecido utilizando el lenguaje Markdown. Para las celdas de tipo código, aquí es donde se puede incrustar el código de programación, en este caso en lenguaje Python. El documento también contiene librerías específicas para importar videos o imágenes para poder ser incluidos dentro de las celdas de texto, también se pueden incluir enlaces a recursos web o incluso incrustar una página web completa dentro del documento. Para añadir texto enriquecido es por medio del uso del lenguaje Markdown, permitiendo incluir listas, texto en negrita, texto en cursiva, tablas o imágenes.

A continuación, sólo se describirán las opciones de formato más utilizadas en los cuadernos de Jupyter, ya que el presente documento no es un manual exclusivo sobre el manejo del lenguaje Python y/o Markdown, ya que existe una gran cantidad de tutoriales en la red.

• *Texto en negrita:* Para mostrar texto en negrita, la palabra en cuestión se escribe entre dos pares de asteriscos. Por ejemplo: **hola** aparecerá como **hola**.
• *Texto en cursiva*: Para mostrar texto en cursiva, la palabra en cuestión se escribe entre asteriscos simples. Por ejemplo: *hola* aparecerá como *hola.*
• *Listas*: Para indicar una lista, en el lenguaje Markdown, se utiliza un asterisco o para listas enumeradas se utiliza un número seguido de un punto. Automáticamente se organiza los items asignándoles el número correcto.
• *Insertar imágenes*: En Markdown para incluir imágenes se utiliza la sintaxis siguiente: ![nombre alternativo](dirección de la imagen), en donde el nombre alternativo aparecerá en caso de que no se pueda cargar la imagen, la dirección hace referencia a una imagen local o a un enlace en Internet.
• *Incluir código HTML*: Se puede incluir directamente el lenguaje HTML ya que el lenguaje Markdown es un subconjunto del lenguaje HTML
• *Enlaces*: Para incluir enlaces ya sea a otras partes del documento o a otras páginas en internet o enlaces a archivos locales, es por medio de la siguiente sintaxis:
[texto](direcci_on del enlace).
• *Fórmulas matemáticas*: Para incluir código en LaTeX y mostrar todo tipo de fórmulas y expresiones matemáticas es por medio del uso de MathJax. Para incluir las fórmulas dentro de una línea de texto, se escriben entre símbolos de dólar: $ fórmula $ Para las expresiones separadas del texto se hace uso del caracter de dólar dobles: $$expresión separada$$.

Las opciones anteriores descritas ayudan a utilizar las celdas de texto de manera útil y versátil, ayudando a documentar o hacer anotaciones en el código de otras celdas, como por ejemplo, incluir la explicación a detalle o en breve de concepto teóricos o de notas sobre las secciones de código que esté mostrando el cuaderno de Jupyter.

Cuando se crea una nueva celda, en automático se una celda de código, para cambiar a celda de texto es necesario ir al menú donde se encuentra las opciones de tipo de celda y hacer el cambio correspondiente. Les recuerdo que el lenguaje que se utiliza en las celdas de código ya viene determinado por la elección del núcleo para todo el documento. Hasta el momento de escribir este documento, no está soportado el que se mezcle dos lenguajes de programación dentro del cuaderno de Jupyter.

4.3.3. Instalación para Jupyter Notebook

El requisito principal para la instalación de Jupyter Notebook es contar con Python, ya que independientemente de que en los cuadernos se permita la ejecución en otros lenguajes de programación, la mayor parte de las funciones de Jupyter Notebook están implementadas en el lenguaje mencionado que es Python.

Los cuadernos de Jupyter ya cuentan con suites de cálculo científico ofreciendo gran cantidad de módulos de Python, esto es debido a que ha crecido la popularidad de este lenguaje en el ambiente científico, siendo la más conocida Anaconda, de la empresa Continuum Analytics. También se cuenta con un gestor de paquetes lo que ha facilitado la instalación y la gestión de librerías de Python relacionadas con aplicaciones científicas.

Anaconda instala el módulo de Jupyter Notebook por defecto, por lo que independientemente del sistema operativo que se esté usando, se recomienda la instalación de la distribución de Python científico. En la página de descarga de Anaconda se puede descargar gratuitamente este producto, basta con seguir los pasos ahí descritos para su instalación para poder tener disponible Jupyter Notebook, y para ejecutarlo basta con escribir en una consola el comando *jupyter notebook.*

5. DISEÑO DE LA INTERVENCIÓN

5.1. DESCRIPCIÓN BREVE DEL DISEÑO DE INTERVENCIÓN

Una vez recogido el marco teórico se llevó a cabo el procedimiento del diseño de la intervención o aplicación, para el presente caso práctico sería el diseño de la aplicación. Para ello se tuvieron que responden las siguientes cuestiones (UAA, 2016):

- Qué se hizo
- Cómo se llevó a cabo,
- Cuándo se realizó,
- Qué etapas comprendió,
- Cuáles fueron las actividades,
- Con qué recursos se contó
- Cuáles son los beneficios esperados

Para responder las cuestionantes anteriores y llevar a cabo el desarrollo e implementación de cuadernos de jupyter en Python para acceder a la información del cubo de Cubo de Datos Geoespaciales de México y proceder a realizar análisis ráster a dicha información, se procedió a poner en marcha la metodología de análisis, diseño, desarrollo e implementación de sistemas informáticos propuesta por los autores Kenneth E. Kendall y Julie E. Kendall. Para conocer un poco más de esta metodología, se describe a continuación de una manera breve:

Primero definiremos lo que es sistema de información: Según Kendall & Kendall (Kendall, 2005). “Un sistema de información es un trabajo que se realiza debido a la interacción resuelta entre los usuarios y la computadora, en donde se requiere que el software y hardware trabajen al unísono para el beneficio de una empresa u organización o entorno que lo requiera”.
Dicha metodología se basa en las siguientes siete fases descritas a continuación y observadas en la Figura15:

5.1.1. Fases de la Metodología de Análisis y Desarrollo de Sistemas propuesta por los Autores Kendall

I Identificación de problemas, oportunidades y objeticos

II Determinación de los requerimientos de información

III Análisis de las necesidades del sistema

IV Diseño del sistema

V Desarrollo y documentación del software

VI Pruebas y mantenimiento del sistema

VII Implementación y evaluación del sistema

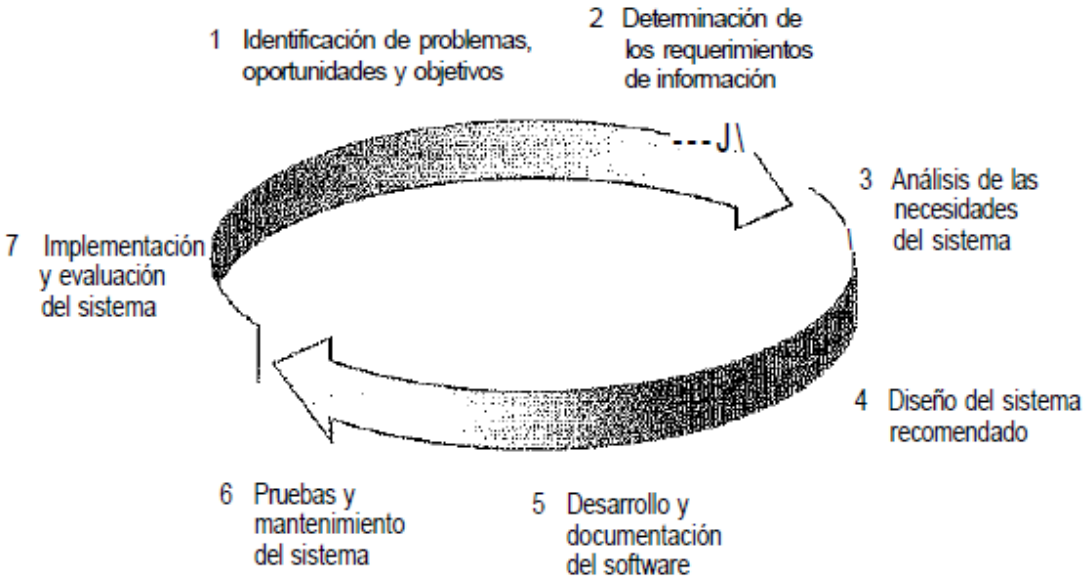


Figura 15. Fases de la metodología de análisis y desarrollo de sistemas propuesta por los autores Kendall.

Fase I. Identificación de problemas, oportunidades y objetivo.

En esta etapa se determina lo que la organización intenta realizar, llevando a cabo las siguientes actividades:

- Observación directa del entorno.
- Identificar los problemas existentes.
- Identificar los objetivos para alcanzar sus metas.

FASE II. Determinar los requerimientos de información.

Pueden utilizarse diversos instrumentos:

- El estudio de los datos
- Formas usadas para la organización
- La entrevista
- Los cuestionarios
- la observación de la conducta de quién tomó la decisión, así como de su ambiente

FASE III. Análisis del sistema.

- Se analizan las necesidades propias del sistema.
- Representar en forma gráfica la entrada de datos a la organización, de los procesos y la salida de información.
- Se deben contar con herramientas y técnicas estructuradas diseñadas para análisis de sistemas como son los diagramas de flujo de datos.

FASE IV. Diseño del sistema recomendado.

- Se usa la información recolectada con anterioridad.
- Se elabora el diseño lógico de sistemas de información
- Los procedimientos deben estar precisos para la captura de datos.
- Tener un diseño de los archivos o la base de datos.
- Tener técnicas estructuradas para un buen diseño de formularios.

FASE V. Desarrollo y documentación del software.

- El usuario trabaja de Manera conjunta con los diseñadores para poder elaborar un Software
- Los diagramas de flujo, en donde puede el diseñador puede requerir con su programación.
- El método HIPO.

FASE VI. Prueba y mantenimiento del sistema,

- Todo sistema de información debe probarse antes de ser utilizado.

TESIS TESIS TESIS TESIS TESIS

- Se identifica las posibles fallas del sistema.
- Todos los programas y sistemas deben verificarse regularmente.
- Se realiza mantenimiento continuo para verificar que los equipos trabajen en perfectas condiciones.
- Utilizar los datos del sistema real durante las pruebas.

FASE VII. Implementación y evaluación del sistema.

- Última fase del desarrollo de sistemas.
- El analista participa en la implementación del sistema de información.
- Se lleva a cabo la conversión del sistema anterior al nuevo, lo que implica la conversión de archivos en formatos anteriores a formatos actuales, la preparación de las bases de datos, la instalación de equipo de así requerirlo, y la puesta en producción del nuevo sistema.
- Los usuarios son capacitados en el manejo del sistema.

Siguiendo la metodología anterior descrita, fue como se llevó a cabo el desarrollo de los cuadernos de Jupyter durante el período comprendido entre enero a julio de 2021.

5.2. PUESTA EN MARCHA

Una vez hecho el diseño para implementar los 3 primeros cuadernos de Jupyter para el análisis ráster de las imágenes satelitales, primeramente, se contempló utilizar el software siguiente:

- Lenguaje de programación en Python
- Anaconda
- Se utilizaron los sistemas operativos Ubuntu 18.0 y Windows 10 Pro, para probar el funcionamiento correcto de los cuadernos tanto en Windows como en Linux.
- Docker

TESIS TESIS TESIS TESIS TESIS

Además, los recursos con lo que se contaron, los cuales fueron proporcionados por el Instituto fueron los siguientes:

- Cubito de datos (Geocubo Portable): Versión compacta del Cubo de Datos Geoespacial de México. Dicha versión la llamamos “Cubito de Datos“ o “Geocubo Portable”, la cual se puede instalar en un equipo personal de cómputo (PC o laptop), ya que es una base de datos que hace referencia a los archivos de Geomedianas de los años de 1993 a 2019, por lo que todas las consultas que se hagan desde este cubito serán resueltas a partir de las Geomedianas indexadas en él.
- Equipo de cómputo personal, siendo una laptop Lenovo Thinkpad, con procesador AMD Ryzen 5 3500U de 2.10 GHz, 16 Gigabytes en memoria RAM, y un 1 Terabyte en disco duro.
- 100 Gigabytes de almacenamiento disponible en disco duro en el equipo de cómputo personal proporcionado por el Instituto, necesario para almacenar la cantidad de imágenes que se analizaron, ya que cada Geomediana nacional pesa alrededor de 35 Gigabytes y el total de espacio que ocupan todas las Geomedianas es de 1 Terabyte .

Después de tener los recursos necesarios con el software requerido y mencionado anteriormente, se procedió a realizar lo siguiente:

- Imagen de Docker con el Cubito de Datos: Imagen con formato TAR con un tamaño de 4 Gigabytes, la cual se instaló en el equipo personal de cómputo proporcionado por el mismo. Esta imagen contiene el software necesario para el correcto funcionamiento del cubito de datos. Dicha imagen se almacenó en la carpeta de “Documentos”.
- Estructura de archivos: Para el correcto funcionamiento del cubito de datos se diseñó una estructura de archivos, la cual es una carpeta llamada “Almacenamiento” la cual contiene los archivos tipo TIF (formato de archivo de imagen que se utiliza generalmente para trabajar con datos de imagen sin procesar) de las Geomedianas anuales y los scripts en Python. Esta carpeta se compactó en un archivo de tipo zip

al cual se denominó “Almacenamiento.zip” para ser descompactado en la raíz de alguna unidad de almacenamiento (C:/, D:/, para el caso de Windows; /home/user/, para el caso de Ubuntu

En la estructura de la carpeta “Almacenamiento” contiene dos subcarpetas y a su vez éstas contienen más subcarpetas, a continuación, se describe de una manera sencilla:

Subcarpetas de la carpeta “Almacenamiento”:

- datos_term: Carpeta donde se almacena todo lo referente a las Geomedianas, las cuales se dividen en anuales, subanuales y multianuales. De momento en el cubito de datos solo están disponibles las Geomedianas anuales, y cada año está separado por carpetas y dentro de cada año se tiene la estructura siguiente:
- Carpeta GM: Contiene el archivo TIF de la Geomediana.
- Carpeta dcFiles: Contiene los archivos necesarios para el correcto funcionamiento del cubito de datos.
- Fuentes: Carpeta que contiene los scripts de Python con los cuales se lleva a cabo la generación de productos del cubo de datos. Cabe señalar que estos programitas pueden ser modificados por usuarios avanzados en programación para que a partir de ellos pueda crear sus propias herramientas de análisis ráster.

Para finalmente, proceder con la programación de los cuadernos de Jupyter con Python.

5.3. BENEFICIOS ESPERADOS

Los beneficios que se esperan obtener con el desarrollo de los cuadernos de Jupyter con acceso al Cubo de Datos Geoespacial de México, es dar a conocer al público en general la información contenida en dicho cubo, es decir, sus imágenes satelitales ya generadas por medio de las Geomedianas para ser consultadas y analizadas por medio de nuestros cuadernos, y además de que, los usuarios puedan realizar su propio análisis modificando el código de programación conformado por dichos cuadernos.

TESIS TESIS TESIS TESIS TESIS

6. RESULTADO DE LA INTERVENCION

Como resultado se obtuvo un producto final, el cual lo nombramos "*Geocubo Portable*", o mejor conocido por nosotros el Cubito de Datos, en donde se lleva de la mano a cualquier usuario a generar análisis geoestadísticos en diferentes zonas de estudio dadas por Cubo de Datos Geoestadístico de México.

6.1 NUESTRA HERRAMIENTA EN SU CONJUNTO

La herramienta resultante como producto final está compuesta de los siguientes elementos:

- Cuadernos de Jupyter programados en Python.
- Estructura de archivos necesarios para el cubito de datos (*Geocubo Portable*).
- Imagen de Docker con el cubito de datos (*Geocubo Portable*) con el software necesario para el correcto funcionamiento del mismo.
- Manual de usuario en archivo con formato tipo pdf y videos-tutoriales que llevan al usuario paso a paso en la configuración e instalación necesaria para el buen funcionamiento de los cuadernos ya mencionados, como lo es la instalación y configuración tanto de Docker como del Cubito de Datos.

Todo lo anteriormente enlistado, disponible al público en general, junto con manuales en archivos con formato tipo pdf y videos-tutoriales explicando con voz paso a paso a seguir para la instalación de los componentes necesarios y el cómo usar cada cuaderno de Jupyter. Además de la instalación por parte del usuario de Docker ya sea para Windows o Linux Ubuntu.

Link de acceso: https://git.inegi.org.mx/cdgm/geocubo-portable

TESIS TESIS TESIS TESIS TESIS

6.1.2 REQUERIMIENTOS

Requerimientos mínimos recomendados para realizar nuestro cuaderno de Jupyter con el cubito de datos (Geocubo Portable):

1. Sistema operativo Windows 10 Pro o Ubuntu 18+.
2. Procesador de 4 núcleos.
3. 8 Gigabytes de memoria RAM.
4. 100 Gigabytes de almacenamiento disponible. Se hace énfasis en que la cantidad de almacenamiento dependerá de la cantidad de imágenes que se desee analizar, ya que el tamaño de cada Geomediana Nacional es aproximadamente de 35 Gigabytes, siendo de 1 Terabyte. el total de espacio que ocupan todas las Geomedianas.

6.2 RECURSOS A INSALAR

6.2.1. Docker

Tecnología para la creación y uso de contenedores de Linux; permite crear diferentes ambientes de trabajo para la ejecución de aplicaciones.

Para instalar Docker en un equipo tanto con Windows y/o Linux, basta con descargar el instalador de la versión estable más actual desde el siguiente enlace:

- https://hub.docker.com/

instalación de Docker se describe en el *Anexo 1* del presente documento.

6.2.2. Cubito de Datos (Geocubo Portable)

El cubito de datos o Geocubo Portable requiere de una estructura de archivos para su correcto funcionamiento la cual es descargada del servidor de INEGI.

TESIS TESIS TESIS TESIS TESIS

El archivo que se descarga se llama “Almacenamiento.zip” y se sugiere que sea descompactado en la raíz de alguna unidad de almacenamiento (C:/, D:/, para el caso de Windows; /home/user/, para el caso de Ubuntu)

La carpeta “Almacenamiento” contiene 2 subcarpetas:

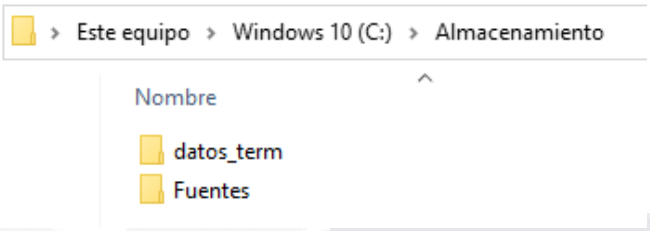


Figura 16. Estructura de carpetas del Cubo Portable / Cubito de datos

1. **datos_term**: En esta carpeta se almacena todo lo referente a las Geomedianas, se dividen en anuales, subanuales y multianuales. De momento en el cubito de datos solo están disponibles las Geomedianas anuales, cada año está separado por carpetas y dentro de cada año vamos a encontrar:
    a. La carpeta **GM**, que es donde se almacena el archivo TIF de la Geomediana.
    b. La carpeta **dcFiles**, archivos necesarios para el correcto funcionamiento del cubito de datos.

2. Fuentes: En esta carpeta vamos a encontrar los scripts de Python con los cuales se lleva a cabo la generación de productos del cubo de datos, se comparten con el fin de que usuarios avanzados en temas de programación y uso del cubo de datos puedan tomar estos scripts y analizarlos para crear a partir de ellos herramientas que les sean útiles para sus tareas.

Proyecto en Gitlab

Otro elemento necesario para instalar el Geocubo Portable es el proyecto de “geocubo-portable” publicado en el Gitlab institucional de INEGI. Se recomienda clonarlo con Git para tener las actualizaciones más recientes, sin embargo, basta con descargarlo.

Descargar el proyecto en formato **ZIP** y se sugiere sea almacenado y descompactarlo en la carpeta de “Documentos”.

Figura 17. Descarga del proyecto GitLab

6.2.3. Imagen de Docker con el Geocubo Portable

El último elemento que se necesita es la imagen de Docker que contiene todo el software necesario para el correcto funcionamiento del Geocubo Portable. Esta imagen está contenida en el cd anexo al presente documento y es un archivo llamado *CubitoDatos.tar*

Este archivo tiene un formato **TAR** y un tamaño mayor a los 4 GB, por lo que su descarga puede demorar varios minutos, se sugiere sea almacenado en la carpeta de “Documentos” para facilitar su acceso más adelante. Ver *Anexo 2*.

6.3 CUADERNOS DE JUPYTER NIVEL PRINCIPIANTE

Los cuadernos de Jupyter incluidos en el producto final y los cuales también vienen incluidos en el cd anexo al presente documento son los siguientes:

- ***01_Carga de datos y cálculo de índices normalizados.***

  Cuaderno donde se carga, visualiza, manipula e interpreta datos satelitales para realizar un análisis sobre cómo cambia el índice de vegetación con el tiempo en un área determinada de estudio. Ver *Anexo 3*.

- ***02_Análisis con múltiples polígonos***

  Cuaderno donde se realiza un análisis vectorial a través de múltiples polígonos en un archivo vectorial (ESRI Shapefile o GeoJSON) utilizando **Open Data Cube** para extraer datos satelitales dentro del área de estudio determinada. Ver *Anexo 4.*

- ***03_Series animadas de tiempo***

  Cuaderno donde se toma una serie de imágenes satelitales del cubo de datos geoespacial y los exporta a una animación de series de tiempo visualmente atractiva que muestre cómo ha cambiado la zona seleccionada como área de estudio en los años determinados por el usuario. Ver Anexo 5.

6.3.1. Iniciar Contenedor

Si el contenedor con el Cubito de Datos (Geocubo Portable) ya se encuentra disponible en su equipo, pero no se está ejecutando, es necesario que desde una terminal en Linux o una consola de Power Shell de Windows, se ubiquen en la raíz del proyecto de Gitlab y se ejecute el siguiente comando:

docker-compose up –d

*Nota***:** Similar a como se realizó en el capítulo anterior, al momento de iniciar con la creación de los contenedores.

6.3.2. Descargando imágenes

El Cubito de Datos (Geocubo Portable) requiere de imágenes para poder hacer las consultas de los cuadernos, estás imágenes se almacenen en la carpeta Almacenamiento que se presentó al principio de este manual, sin embargo, de inicio la carpeta no contiene ninguna imagen y será necesario ejecutar el cuaderno para descargas de imágenes.

Para poder iniciar el cuaderno de descargas habrá que acceder al siguiente link desde cualquier navegador web: *http://127.0.0.1:8888/?token=*

Al ingresar al sitio se solicita una contraseña, es la siguiente: ***datacubeOnDocker***

Una vez que ingresamos la contraseña, se muestra una pantalla como la siguiente:

TESIS TESIS TESIS TESIS TESIS

Figura 18. Descarga de imágenes 1

La carpeta *Descargas* contiene el cuaderno que se debe iniciar para la descarga de imágenes. El cuaderno está listo para descargar las imágenes que comprenden al estado de Aguascalientes, basta con abrirlo y seguir las instrucciones.

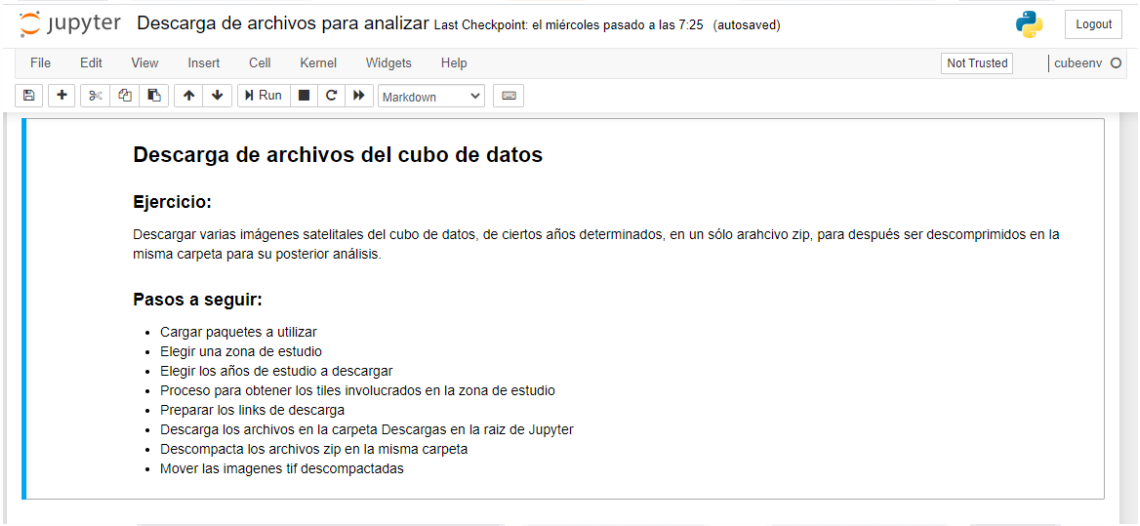


Figura 19. Descarga de imágenes 2

Nota: El cuaderno puede ser modificado para descargar otras regiones del país.

6.3.3. Ejecución de los cuadernos para principiantes de Jupyter

Jupyter Notebook es una herramienta de programación en Python que nos permitirá interactuar mediante código con el cubo de datos. Para acceder a esta herramienta debemos entrar al siguiente enlace: *http://127.0.0.1:8888/tree*

Nota: Se nos pedirá una contraseña, la cual es: ***datacubeOnDocker***

Una vez que accedamos con la contraseña, veremos una página web como se muestra a continuación:

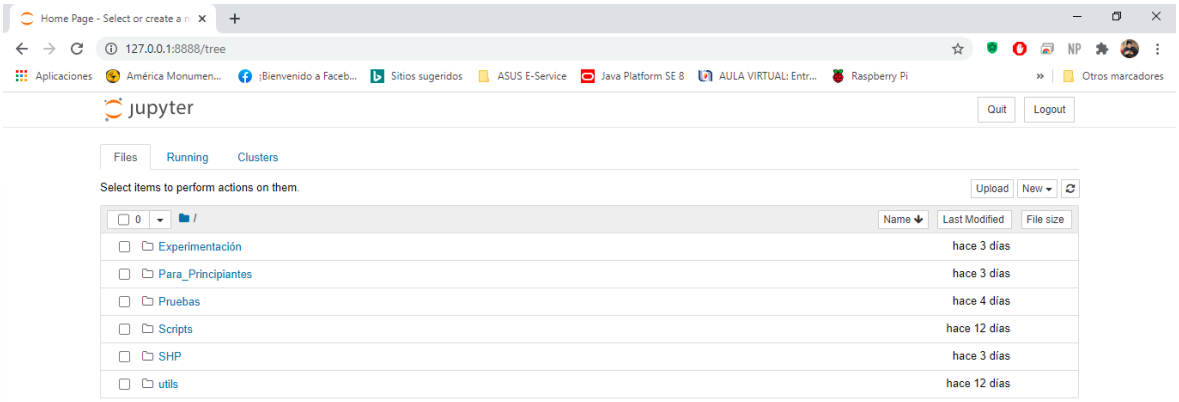


Figura 20. Cuadernos 1

En la ventana se muestra como un explorador de archivos, donde se pueden crear o consultar cuadernos de Jupyter con código que interactúe directamente con el cubito de datos. Para probar el primero de los cuadernos, hacemos doble clic en la carpeta *Para_Principiantes*, ahí veremos 3 cuadernos con ejercicios que resultarán de utilidad para quienes inician como usuarios del cubo de datos.

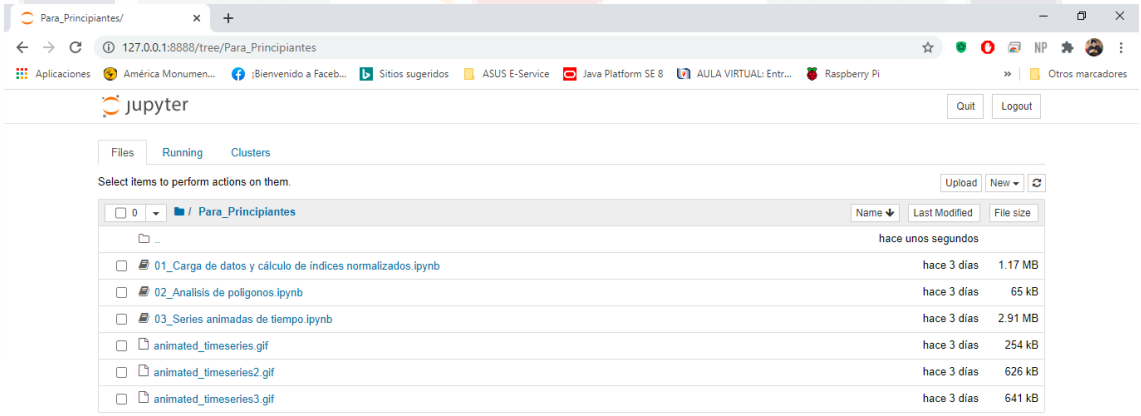


Figura 21. Cuadernos 2

Hacemos clic en el cuaderno llamado “01_Carga de datos y cálculo de índices normalizados.ipynb”. Se nos mostrará la siguiente pantalla, en una pestaña nueva, con el código y una explicación breve del cuaderno:

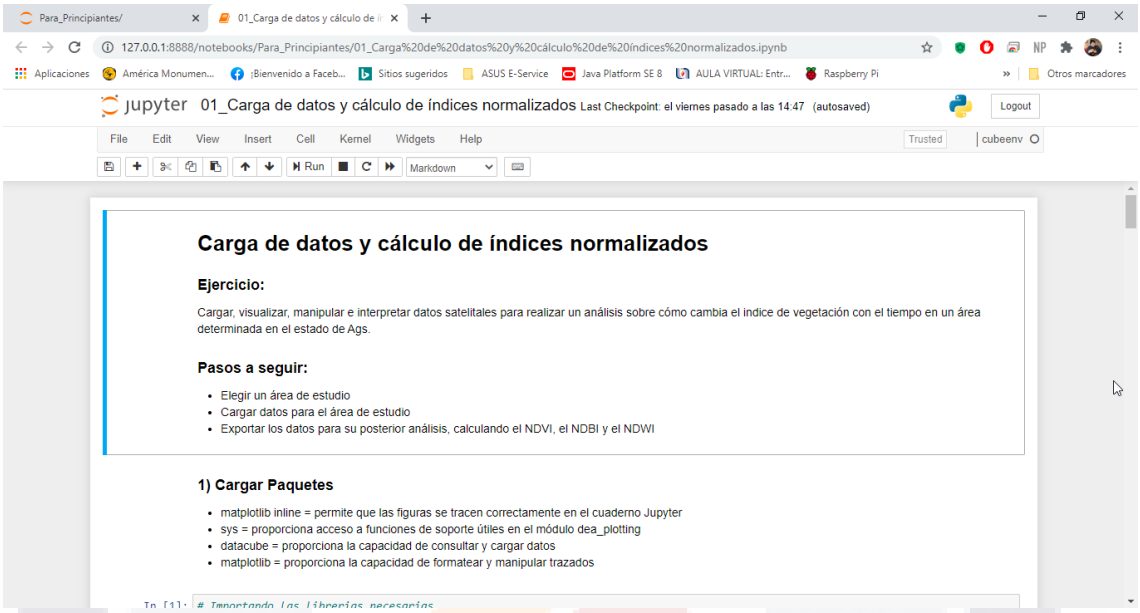


Figura 22. Cuadernos 3

Las divisiones que se muestran en color azul son llamadas celdas, para ejecutar el código de una celda es necesario seleccionarla haciendo clic sobre ella y luego presionar el botón Run en la parte superior de nuestra pantalla. Como se muestra en la siguiente imagen:

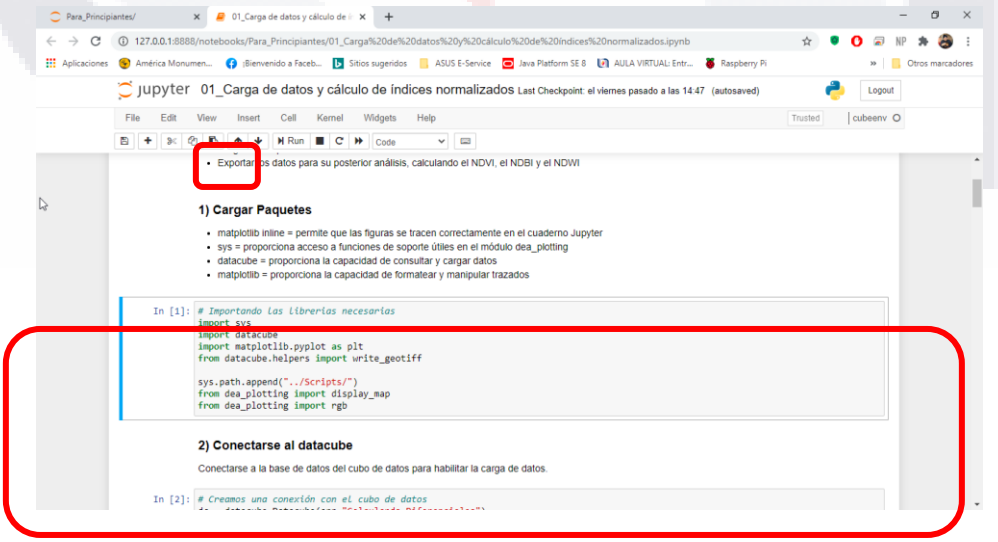


Figura 23. Cuadernos 4

Para la correcta ejecución del cuaderno deberá ejecutar todas las celdas con código y verificar que no tenga errores. Una vez que terminé con ello se puede constar que la instalación del Cubito de Datos fue un éxito.

Figura 24. Cuadernos 5

7. EVALUACIÓN DE LA INTERVENCION

7.1. RESULTADOS OBTENIDOS

La mejor manera que se tuvo para evaluar tanto la practicidad y la usabilidad de los cuadernos de Jupyter desarrollados en Python para la explotación del Geocubo Portable, fue ofrecer dichos cuadernos a través de la realización de un curso, el cual se llamó “Curso – taller Cubo de Datos Geoespaciales”.

El “Curso – taller Cubo de Datos Geoespaciales” fue organizado por el INEGI y la Universidad Autónoma del Estado de México en julio de 2021, con el objetivo de dar a conocer a los estudiantes de la facultad de Geografía de la Universidad Autónoma del Estado de México el Cubo de Datos Geoespaciales de México y su explotación de información por medio de los cuadernos de Jupyter en Phyton. La modalidad del mencionado curso-taller fue virtual.

INEGI fue quien impartió el curso, y facilitó el material para llevarlo a cabo, el cual se encuentra contenido dentro del disco compacto (CD) que acompaña al presente documento. Cabe mencionar que una servidora participó como instructora de curso.

Durante el curso-taller, los estudiantes aprendieron a utilizar los cuadernos para, además de aprender a utilizarlos y conocer el Cubo de Datos Geoespaciales de México, iniciar con sus propios análisis espaciales.

Lo que se observó durante el curso-taller, en cuanto a la aceptación y uso de los cuadernos con el cubito de datos fue lo siguiente:

Aspecto	Calificación			
	Muy bien (10)	Bien (8/9)	Regular (6-7)	Malo (0-5)
Facilidad de uso	X			
El flujo del cuaderno es intuitivo	X			
Se entiende la explicación dada dentro del cuaderno	X			
Facilidad para cambio de parámetros o de código para realizar análisis espacial	X			
Facilidad en la instalación de componentes		X		

Tabla 1. Aceptación de los cuadernos de Jupyter por parte de los estudiantes

Material utilizado durante el curso-taller:

- Manuales en archivos con formato pdf
- Videotutoriales con voz
- Cuadernos de Jupyter
- Estructura de archivos necesarios para el Cubito de Datos
- Imagen de Docker con el Cubito de Datos con el software necesario para el correcto funcionamiento del mismo
- Asesorías y apoyo por parte de los instructores

Objetivo general y específicos cumplidos:

- Alumnos tuvieron mayor entendimiento de lo que es el Cubo de Datos Geoespaci-laes de México
- Los materiales proporcionados les facilitó la comprensión del uso de los cuadernos de Jupyter en Python

TESIS TESIS TESIS TESIS TESIS

- Los cuadernos resultaron ser una herramienta novedosa para análisis ráster
- Aprendieron “jugando” al cambiar los parámetros lo que les ayudó a crear sus propios cuadernos a su gusto.

7.2. EVIDENCIAS DEL CURSO-TALLER

A continuación, se muestran algunas impresiones de pantalla que se tomaron durante las sesiones en el Curso – taller Cubo de Datos Geoespaciales.

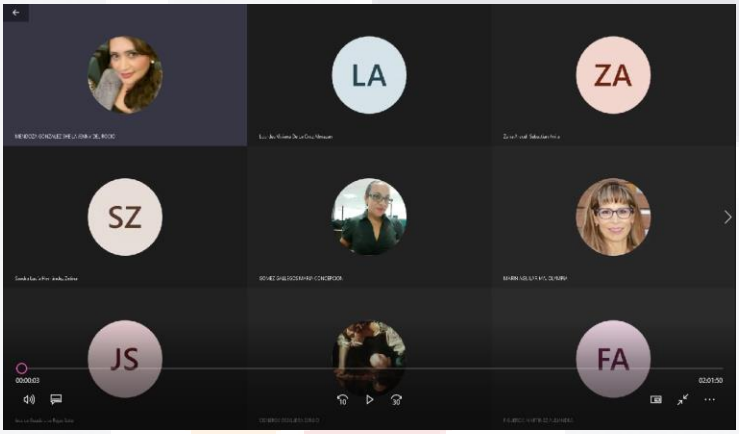


Figura 25. Sesión virtual del Curso – taller Cubo de Datos Geoespaciales

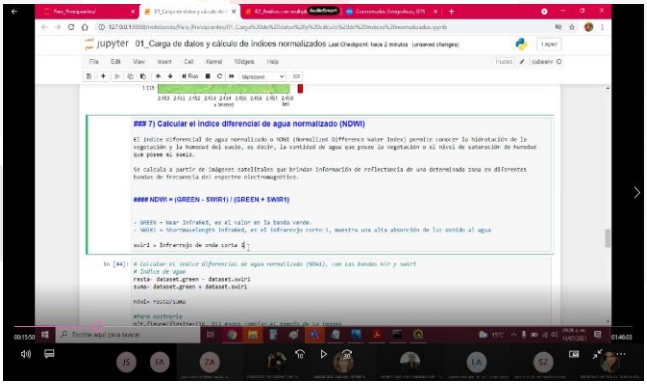


Figura 26. Explicación del cuaderno de Jupyter durante la sesión del curso-taller

Mi constancia como instructora durante el Curso – taller Cubo de Datos Geoespaciales

INEGI

Universidad Autónoma del Estado de México

El Instituto Nacional de Estadística y Geografía,
a través de la Dirección General de Integración, Análisis e Investigación
y la **Universidad Autónoma del Estado de México,**
a través de la Facultad de Geografía
Otorgan la presente

Constancia

a: **Sheila Jenny Del Rocío Mendoza González**

Por su participación como instructora frente a grupo en el **Curso-Taller "Cubo de datos geoespaciales"**, impartido del 28 de junio al 15 de julio de 2021 en la modalidad en línea, con una duración de 30 horas.

Toluca, México, 15 de julio de 2021.

Dr. Sergio Carrera Riva Palacio
DIRECTOR GENERAL DE INTEGRACIÓN, ANÁLISIS E INVESTIGACIÓN

Dr. Rodrigo Huitrón Rodríguez
DIRECTOR DE LA FACULTAD DE GEOGRAFÍA DE LA UAEM

Figura 27. Constancia de Instructora

7.3. TRABAJO A FUTURO

Uno de los propósitos con los que cuenta el Instituto, es seguir dando a conocer el Cubo de Datos Geoespaciales de México a usuarios en general, y sería a través de lo siguientes acciones englobadas:

- Dar continuidad con el proyecto de los cuadernos de Jupyter y el Geocubo Portable
- Desarrollo de cuadernos nivel intermedio y avanzado
- Mayor difusión del producto a instituciones educativas y al público en general

TESIS TESIS TESIS TESIS TESIS

CONCLUSIONES

El Cubo de Datos Geoespaciales de México, junto con los cuadernos de Jupyter desarrollados durante el presente trabajo práctico, son una herramienta de gran utilidad y de fácil uso y aprendizaje para los usuarios interesados en realizar análisis espacial en temas relacionados con el uso de suelo y vegetación, la erosión costera, la agricultura, la deforestación, los cuerpos de agua o los asentamientos humanos, además, que facilita el acceso a las imágenes contenidas en el cubo para llevar a cabo dichos análisis.

Además, durante el desarrollo y puesta en práctica del presente trabajo práctico, no sólo adquirí nuevo conocimiento y desarrollo de habilidades de investigación, sino que tuve la oportunidad de explorar nuevas tecnologías y profundizar en otras, y reforzar conocimientos sobre análisis espacial, y lo más valioso para mí fue las nuevas vivencias personales y profesionales ganadas.

¡GRACIAS!

TESIS TESIS TESIS TESIS TESIS

REFERENCIAS BIBLIOGRÁFICAS

- Asociación Geoinnova. (n.d.). *El secreto de las bases de datos espaciales: índices y funciones espaciales - Territorio Geoinnova - SIG y Medio Ambiente*.
- Cecilia, M. G., Nieto, G., Luis, M. G., & Fajardo, A. C. (n.d.). *¿ Qué son bases de datos geoespaciales ?*
- Emilio Chuvieco. (n.d.). *Fundamentos de-teledeteccion-emilio-chuvieco*. Retrieved April 5, 2020, from https://es.slideshare.net/noldinn/fundamentos-deteledeteccionemiliochuvieco
- Francisco Javier Nava, I. (2019). *Herramientas de información geoespacial para la agenda de desarrollo sostenible*.
- Geoscience Australia. (n.d.). *Water Observations from Space*. https://www.ga.gov.au/scientific-topics/community-safety/flood/wofs
- HonGbo, S. G. de la O. W. (2012). , 2012. *Consejo Económico y Social de Las Naciones Unidas*.
- Jordi Vivancos Martí, Albert Llastarri, Mònica Grau, D. V. (n.d.). *Teledetección y Sistemas de Información Geográfica (SIG)*. http://concurso.cnice.mec.es/cnice2006/material121/unidad2/td_sig.htm
- *QGis Documentation*. (n.d.). https://www.qgis.org/es/docs/index.html
- Susana González G. (2019). Hay carencias tecnológicas para explotar información en México: Inegi. *La Jornada*.
- *UAA. Universidad Autónoma de Aguascalientes. Manual para la Elaboración del Trabajo Recepcional en los Programas de Posgrado: Tesis o Trabajo Práctico. 2016*
- *Kendall Kennet, Kendall Julie. Análisis y Diseño de Sistemas. Sexta Edición. 2005.*

TESIS TESIS TESIS TESIS TESIS

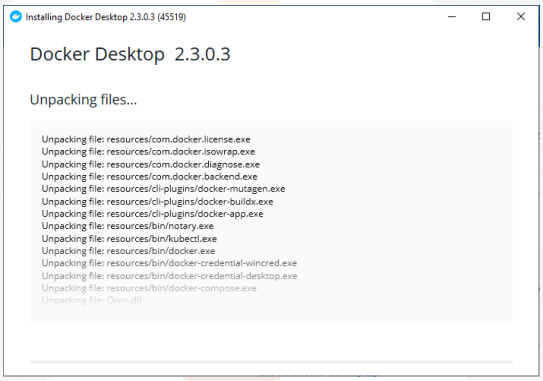
ANEXOS

ANEXO 1. INSTALACIÓN DE DOCKER

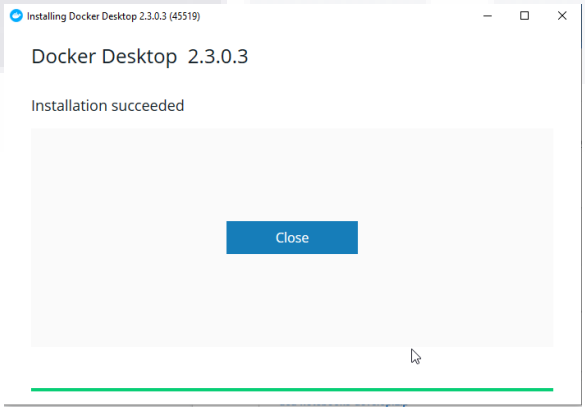
En Windows: Para instalar Docker en un equipo con Windows es bastante sencillo, basta con descargar el instalador de la versión estable más actual desde el siguiente enlace:

https://hub.docker.com/editions/community/docker-ce-desktop-windows

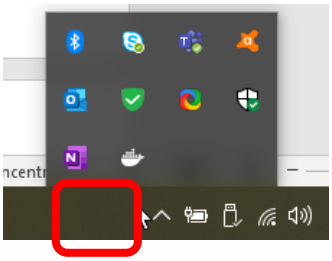
El archivo pesa alrededor de 400MB por lo que demora un poco en descargarse. Una vez que se descargue, procedemos a su instalación haciendo doble clic sobre el mismo para que inicie el asistente de instalación:



Establecemos todas las configuraciones por defecto o recomendadas por el asistente de instalación y permitimos el reinicio del equipo al finalizar.

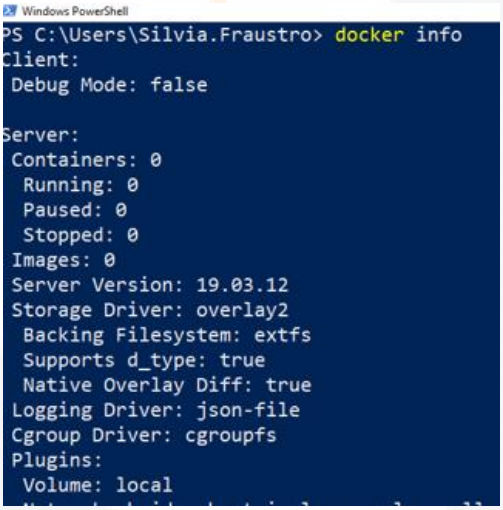


Una vez que se haya reiniciado el equipo, podemos ver en nuestra barra de tareas el icono de una ballena, lo que quiere decir que docker se está ejecutando en nuestra maquina:



Para probar Docker, abrimos una venta de *Power Shell* de Windows y ejecutamos el siguiente comando:

docker info



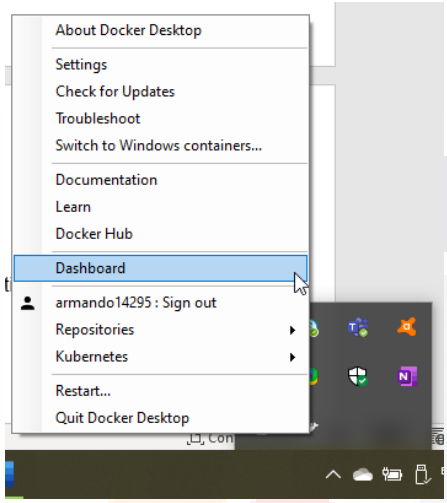
En respuesta a ese comando se deben mostrar las características de Docker en el equipo que se está ejecutando, en caso de una instalación sin éxito, se mostrará el mensaje de error correspondiente a que no se identifica el comando Docker.

Incrementar los recursos de Docker en el sistema:

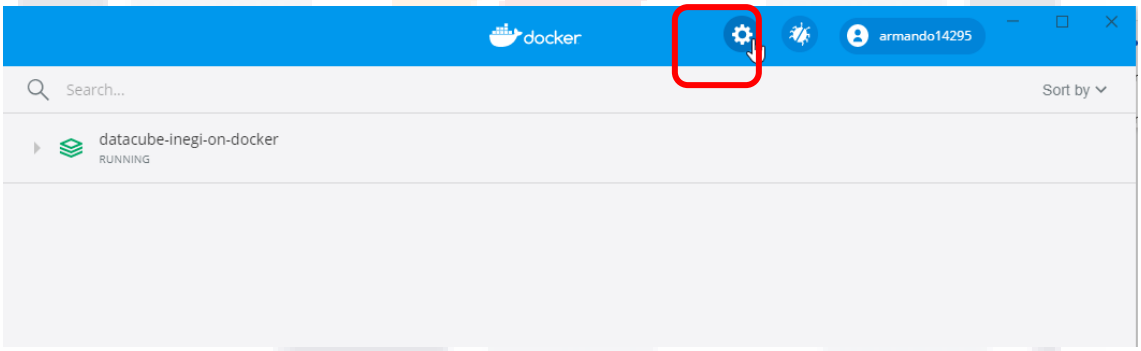
En Windows, Docker funciona como una máquina virtual con recursos limitados establecidos por el software, y esos recursos no son suficientes para procesar grandes volúmenes de datos, por lo que es necesario incrementarlos.

Para poder incrementar los recursos a nuestro Docker, debemos seguir los siguientes pasos:

1. Abrimos el *dashboard* de *Docker Desktop* haciendo clic derecho en la ballenita de la barra de tareas, como se muestra en la imagen:

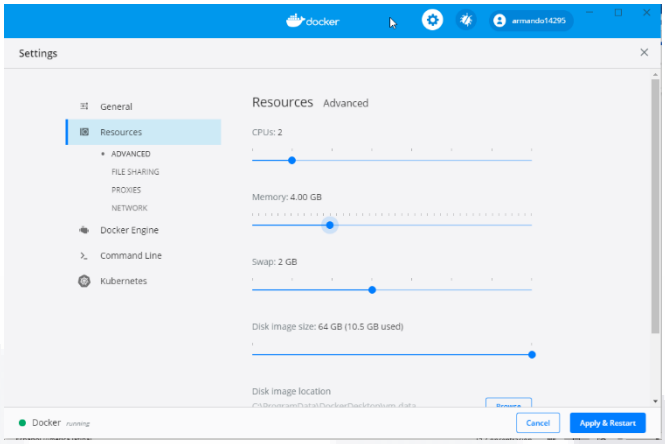


2. Se abrirá una ventana como la siguiente y debemos dar clic en el engrane de ajustes en la parte superior de la ventana:



3. Se mostrarán las configuraciones generales de Docker en nuestro sistema, daremos clic en la pestaña *Resources* en la barra de la izquierda de la ventana y ajustaremos los recursos como se muestra en la imagen:

Nota: Se sugiere que como mínimo de memoria *RAM* se establezcan 4GB y como memoria de *SWAP* 2GB.



4. Por último, presionamos el botón de *Apply & Restart* para que se apliquen los cambios, puede tardar algunos minutos.

En Linux Ubuntu: Para la instalación de docker en este sistema operativo hay que seguir los siguientes pasos:

1. Actualizar repositorios con el comando:

sudo apt update

```
armando@armando-INEGI:~$ sudo apt update
[sudo] contraseña para armando:
Obj:1 http://mx.archive.ubuntu.com/ubuntu focal InRelease
Obj:2 http://mx.archive.ubuntu.com/ubuntu focal-updates InRelease
Obj:3 http://security.ubuntu.com/ubuntu focal-security InRelease
Obj:4 http://mx.archive.ubuntu.com/ubuntu focal-backports InRelease
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se pueden actualizar 23 paquetes. Ejecute «apt list --upgradable» para verlos.
```

Nota: Si el comando le pide que se actualicen paquetes del sistema operativo, es necesarios ejecutar el siguiente comando: *sudo apt-get upgrade*

```
armando@armando-INEGI:~$ sudo apt upgrade
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Calculando la actualización... Hecho
Se actualizarán los siguientes paquetes:
  fonts-opensymbol gnome-shell gnome-shell-common libjuh-java libjurt-java
  libreoffice-ogltrans libreoffice-pdfimport libreoffice-style-breeze
  libreoffice-style-colibre libreoffice-style-elementary
  libreoffice-style-tango libridl-java libuno-cppu3 libuno-cppuhelpergcc3-3
  libuno-purpenvhelpergcc3-3 libuno-sal3 libuno-salhelpergcc3-3
  libunoloader-java python3-distupgrade ubuntu-release-upgrader-core
  ubuntu-release-upgrader-gtk uno-libs-private ure
23 actualizados, 0 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 0 B/15.9 MB de archivos.
Se utilizarán 52.2 kB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s
(Leyendo la base de datos ... 190813 ficheros o directorios instalados actualmen
```

Y en seguida, volver a ejecutar el comando de *update*.

2. A continuación, instale algunos paquetes que le permiten a *apt* usar paquetes mediante *HTTPS*:

   sudo apt install apt-transport-https ca-certificates curl software-properties-common

```
armando@armando-INEGI:~$ sudo apt install apt-transport-https ca-certificates curl software-properties-com
mon
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
software-properties-common ya está en su versión más reciente (0.98.9).
fijado software-properties-common como instalado manualmente.
ca-certificates ya está en su versión más reciente (20190110ubuntu1.1).
fijado ca-certificates como instalado manualmente.
Se instalarán los siguientes paquetes NUEVOS:
  apt-transport-https curl
0 actualizados, 2 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 163 kB de archivos.
Se utilizarán 570 kB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s
Des:1 http://mx.archive.ubuntu.com/ubuntu focal-updates/universe amd64 apt-transport-https all 2.0.2ubuntu
0.1 [1 708 B]
Des:2 http://mx.archive.ubuntu.com/ubuntu focal-updates/main amd64 curl amd64 7.68.0-1ubuntu2.1 [162 kB]
Descargados 163 kB en 1s (149 kB/s)
Seleccionando el paquete apt-transport-https previamente no seleccionado.
(Leyendo la base de datos ... 190813 ficheros o directorios instalados actualmen
te.)
Preparando para desempaquetar .../apt-transport-https_2.0.2ubuntu0.1_all.deb ...
Desempaquetando apt-transport-https (2.0.2ubuntu0.1) ...
Seleccionando el paquete curl previamente no seleccionado.
Preparando para desempaquetar .../curl_7.68.0-1ubuntu2.1_amd64.deb ...
Desempaquetando curl (7.68.0-1ubuntu2.1) ...
Configurando apt-transport-https (2.0.2ubuntu0.1) ...
Configurando curl (7.68.0-1ubuntu2.1) ...
Procesando disparadores para man-db (2.9.1-1) ...
```

3. Luego, agregue la clave GPG para el repositorio oficial de Docker a su sistema:

   curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add –

```
armando@armando-INEGI:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
OK
```

4. Agregue el repositorio de Docker a las fuentes de APT:

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu focal stable"

```
armando@armando-INEGI:~$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubunt
u focal stable"
Des:1 https://download.docker.com/linux/ubuntu focal InRelease [36.2 kB]
Obj:2 http://mx.archive.ubuntu.com/ubuntu focal InRelease
Obj:3 http://mx.archive.ubuntu.com/ubuntu focal-updates InRelease
Obj:4 http://security.ubuntu.com/ubuntu focal-security InRelease
Obj:5 http://mx.archive.ubuntu.com/ubuntu focal-backports InRelease
Des:6 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages [3 056 B]
Descargados 39.2 kB en 1s (30.5 kB/s)
Leyendo lista de paquetes... Hecho
```

5. Posteriormente, actualice la base de datos de paquetes usando los paquetes de Docker del repositorio que acaba de agregar:

sudo apt update

6. Por último, instale Docker:

sudo apt install docker-ce

```
armando@armando-INEGI:~$ sudo apt install docker-ce
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  aufs-tools cgroupfs-mount containerd.io docker-ce-cli git git-man
  liberror-perl pigz
Paquetes sugeridos:
  git-daemon-run | git-daemon-sysvinit git-doc git-el git-email git-gui gitk
  gitweb git-cvs git-mediawiki git-svn
Se instalarán los siguientes paquetes NUEVOS:
  aufs-tools cgroupfs-mount containerd.io docker-ce docker-ce-cli git git-man
  liberror-perl pigz
0 actualizados, 9 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 90.8 MB de archivos.
Se utilizarán 420 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s
```

7. Ahora debería tener Docker instalado, el demonio iniciado, y el proceso habilitado para iniciar durante el arranque. Verifique que se esté ejecutando:

sudo systemctl status docker

TESIS TESIS TESIS TESIS TESIS

```
armando@armando-INEGI:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset>
     Active: active (running) since Tue 2020-06-30 10:05:36 CDT; 50s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 11341 (dockerd)
      Tasks: 16
     Memory: 41.3M
     CGroup: /system.slice/docker.service
             └─11341 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/con>

jun 30 10:05:35 armando-INEGI dockerd[11341]: time="2020-06-30T10:05:35.7218620>
jun 30 10:05:35 armando-INEGI dockerd[11341]: time="2020-06-30T10:05:35.7218748>
jun 30 10:05:35 armando-INEGI dockerd[11341]: time="2020-06-30T10:05:35.7218949>
jun 30 10:05:35 armando-INEGI dockerd[11341]: time="2020-06-30T10:05:35.7222400>
jun 30 10:05:35 armando-INEGI dockerd[11341]: time="2020-06-30T10:05:35.8996046>
jun 30 10:05:36 armando-INEGI dockerd[11341]: time="2020-06-30T10:05:36.0019594>
jun 30 10:05:36 armando-INEGI dockerd[11341]: time="2020-06-30T10:05:36.0601894>
jun 30 10:05:36 armando-INEGI dockerd[11341]: time="2020-06-30T10:05:36.0605331>
jun 30 10:05:36 armando-INEGI dockerd[11341]: time="2020-06-30T10:05:36.1014673>
jun 30 10:05:36 armando-INEGI systemd[1]: Started Docker Application Container >
```

De forma predeterminada, el comando Docker solo se puede ejecutar por el usuario de root o por un usuario perteneciente al grupo *Docker*, el cual se creó durante la instalación de Docker. Si intenta ejecutar el comando docker sin el prefijo sudo o sin estar en el grupo docker, en la consola se mostrará un mensaje como el siguiente:

docker: Cannot connect to the Docker daemon. Is the docker daemon running on this host?.
See 'docker run --help'.

Para agregar su nombre de usuario al grupo *Docker* y así evitar escribir sudo antes ejecutar el comando Docker, ejecute lo siguiente:

sudo usermod -aG docker ${USER}

Para aplicar la nueva membresía de grupo, debe cerrar sesión en el servidor y volver a iniciarla, o puede ejecutar lo siguiente:

su - ${USER}

Nota: Se le pedirá que ingrese la contraseña de su usuario para poder continuar.

```
armando@armando-INEGI:~$ sudo usermod -aG docker armando
armando@armando-INEGI:~$ su - armando
Contraseña:
```

Para probar que todo funciona correctamente, puede ejecutar el siguiente comando para obtener información de Docker en su sistema:

docker info

```
armando@armando-INEGI:~$ docker info
Client:
 Debug Mode: false

Server:
 Containers: 0
  Running: 0
  Paused: 0
  Stopped: 0
 Images: 0
 Server Version: 19.03.12
 Storage Driver: overlay2
  Backing Filesystem: extfs
  Supports d_type: true
  Native Overlay Diff: true
 Logging Driver: json-file
 Cgroup Driver: cgroupfs
```

Por último, es necesario hacer la instalación de una utilería de Docker para hacer usar correctamente el cubito de datos, para ello ejecutamos el siguiente comando:

sudo apt install docker-compose

```
armando@armando-INEGI:~/Documentos/Docker/datacube-inegi-on-docker-master$ sudo apt install docker-compose
[sudo] contraseña para armando:
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  python3-attr python3-cached-property python3-distutils python3-docker python3-dockerpty python3-docopt
  python3-importlib-metadata python3-jsonschema python3-more-itertools python3-pyrsistent
  python3-setuptools python3-texttable python3-websocket python3-zipp
Paquetes sugeridos:
  python-attr-doc python-jsonschema-doc python-setuptools-doc
Paquetes recomendados:
  docker.io
Se instalarán los siguientes paquetes NUEVOS:
  docker-compose python3-attr python3-cached-property python3-distutils python3-docker python3-dockerpty
  python3-docopt python3-importlib-metadata python3-jsonschema python3-more-itertools python3-pyrsistent
  python3-setuptools python3-texttable python3-websocket python3-zipp
0 actualizados, 15 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 915 kB de archivos.
Se utilizarán 4 758 kB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s
```


TESIS TESIS TESIS TESIS TESIS

ANEXO 2. INSTALACIÓN DEL CUBITO DE DATOS O GEOCUBO PORTABLE

La imagen de Docker que se descargó desde el inicio de este manual, tiene un tamaño aproximado de 4GB y contiene la instalación de todas las librerías que se requieren para el correcto funcionamiento del cubito de datos. Una vez que se descargó en nuestro equipo, el siguiente paso es añadir la imagen al repositorio de nuestro Docker local, para ello abrimos una terminal en Linux o una consola de Windows Power Shell para ejecutar el siguiente comando:

Nota: Antes de ejecutar el comando para cargar la imagen al repositorio de Docker, hay que colocarse, desde la consola, en la ubicación de la imagen que recién descargamos. Para ello puede utilizar el comando: *cd /ruta/donde/se/encuentre/el/archivo*. Si usted almacenó la imagen en la carpeta de Documentos, como se sugirió al inicio, puede utilizar el siguiente comando:

cd .\Documents\

docker load -i .\datacube_prueba.tar

Este paso puede tardar algunos minutos, pero una vez que termine podemos ver que se agregó a nuestra lista de imágenes en docker a través del siguiente comando:

docker image ls

Y se nos mostrará un resultado como el que se muestra en la siguiente imagen:

```
PS C:\Users\asv14> docker image ls
REPOSITORY                                TAG       IMAGE ID       CREATED         SIZE
datacube-inegi-on-docker_datacube         latest    b6f246c3a0db   2 weeks ago     4.11GB
```

Para continuar con el proceso de instalación del cubito de datos, es necesario modificar el archivo *docker-compose.yml* que se encuentra en la raíz del proyecto que descargamos de gitlab previamente. Las líneas para modificar se encuentran señaladas en la siguiente imagen:

```
20      volumes:
21        - ./datacube-inegi/.datacube.conf:/root/.datacube.conf
22        - ./Jupyter-Notebook:/home/notebooks
23        - C:/Almacenamiento/datos_term:/datos_term/
24        - C:/Almacenamiento/Fuentes:/Fuentes/
25      depends_on:
```

Se tienen que ajustar las rutas de nuestro equipo hacia la carpeta *Almacenamiento*; la carpeta con la estructura de archivos que se descargó al inicio de este manual. Una vez que tenemos la ruta procedemos a crear el contenedor con el software para el cubo de datos. Para ello, desde la terminal en Linux o desde la consola de *Power Shell* de Windows, nos ubicamos en la raíz del proyecto (Comando “cd /ruta/donde/este/el/proyecto”) y se ejecutará el siguiente comando:

Nota: Por defecto la ruta donde se clona el proyecto es en la carpeta Documentos, el comando para moverse a esa ubicación es el siguiente:

cd '.\Documents\datacube-inegi-on-docker\'

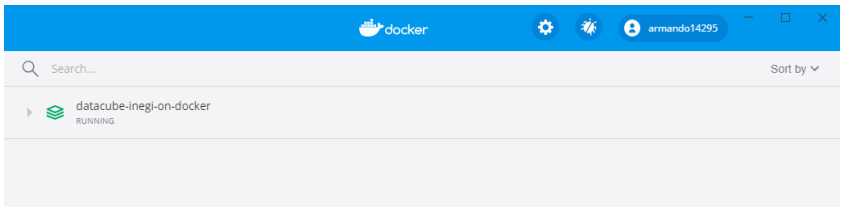
docker-compose up -d

Windows PowerShell

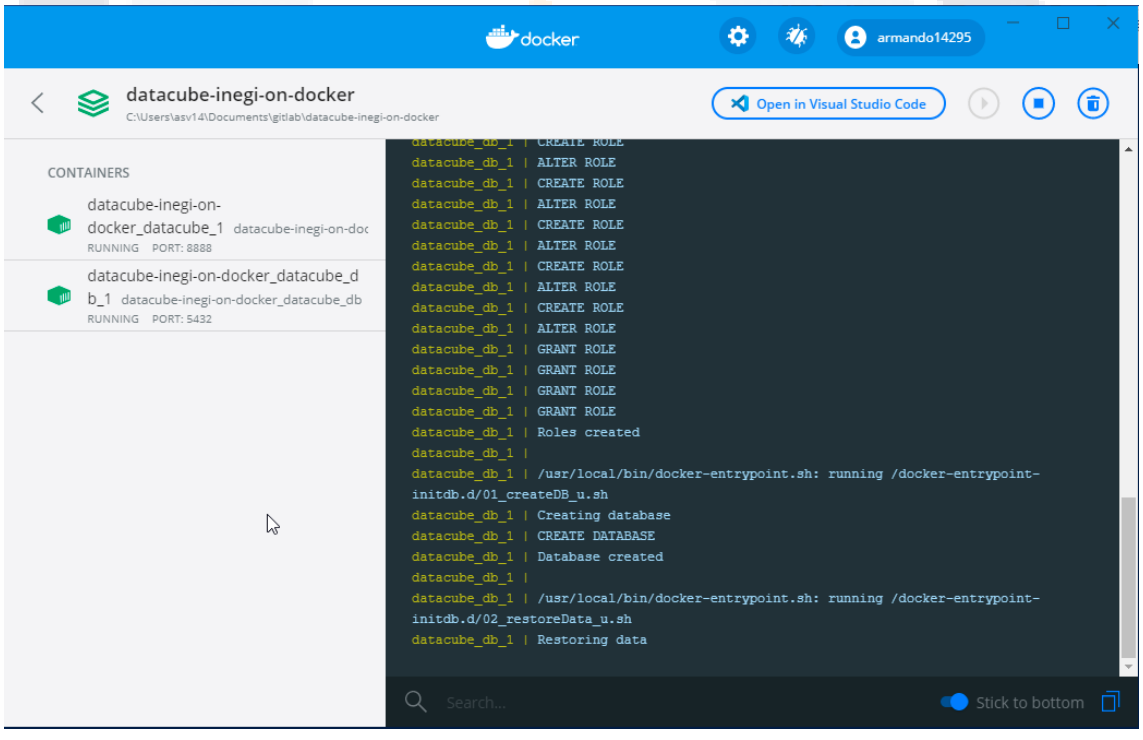
```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/pscore6

PS C:\Users\asv14> cd 'C:\Users\asv14\Documents\gitlab\datacube-inegi-on-docker\'
PS C:\Users\asv14\Documents\gitlab\datacube-inegi-on-docker> docker-compose up -d
Creating datacube-inegi-on-docker_datacube_db_1 ... done
Creating datacube-inegi-on-docker_datacube_1    ... done
PS C:\Users\asv14\Documents\gitlab\datacube-inegi-on-docker>
```

Una vez que se haya ejecutado el comando los contenedores de Docker para el Cubito de Datos estarán activos e iniciando sus tareas, podemos monitorearlos desde el *Dashboard*, al cual accedemos haciendo clic secundario sobre el icono de Docker en la barra de

tareas y seleccionando la opción *Dashboard*, se nos mostrará una pantalla como la siguiente:



En la ventana del *Dashboard* se nos muestran los contenedores que se están ejecutando en nuestro equipo, para este caso solo tenemos en ejecución los contenedores del proyecto del Cubito de Datos. Al hacer clic sobre el nombre del proyecto, se nos muestra una pantalla como la siguiente:



Es importante destacar, que en este momento aún no es posible hacer consultas al Cubito de Datos, pues su base de datos está en proceso de Restauración; como lo indica la ultima línea que podemos ver en la consola de monitoreo. Ese proceso puede tardar algunos minutos y una vez que concluya nos mostrará un mensaje como el siguiente:

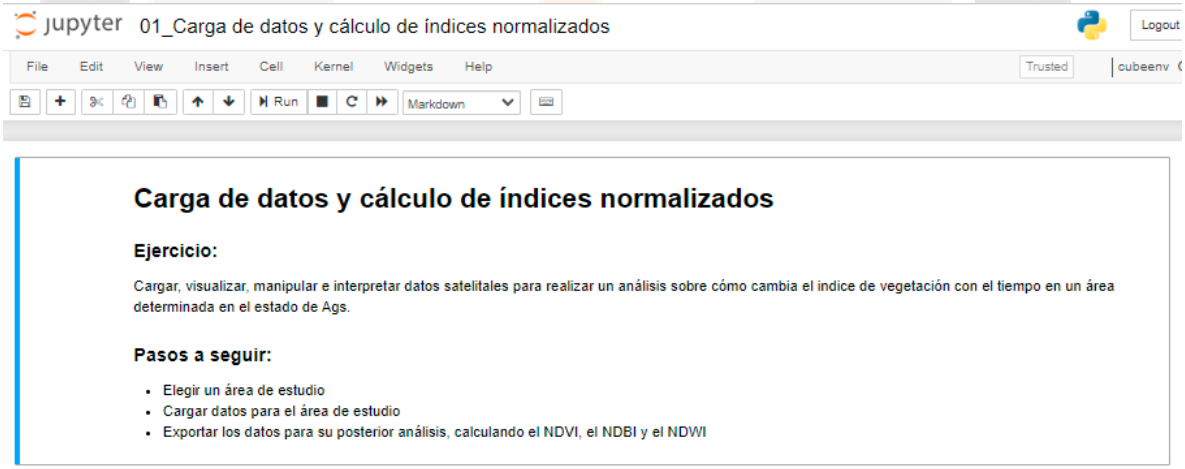
```
2020-10-23 14:22:42 UTC
datacube_db_1 | 2020-10-23 14:22:43.133 UTC [1] LOG: database system is ready to accept
connections
```

En el momento que veamos ese mensaje sobre la pantalla de monitoreo de los contenedores, entonces será el momento de poder iniciar con la ejecución de cuadernos de Jupyter, tema que se aborda en el siguiente capítulo.

Cuando se desee detener los contenedores, por ejemplo, para apagar su equipo de cómputo, debe ejecutar el siguiente comando para evitar una posible pérdida de datos:

docker-compose stop

Anexo 3. 01_Carga de datos y cálculo de índices normalizados



1) Cargar Paquetes

- matplotlib inline = permite que las figuras se tracen correctamente en el cuaderno Jupyter
- sys = proporciona acceso a funciones de soporte útiles en el módulo dea_plotting
- datacube = proporciona la capacidad de consultar y cargar datos
- matplotlib = proporciona la capacidad de formatear y manipular trazados

In [1]:
```
# Importando las librerias necesarias
import sys
import datacube
import matplotlib.pyplot as plt
from datacube.helpers import write_geotiff

sys.path.append("../Scripts/")
from dea_plotting import display_map
from dea_plotting import rgb
```

2) Conectarse al datacube

Conectarse a la base de datos del cubo de datos para habilitar la carga de datos.

In [2]:
```
# Creamos una conexión con el cubo de datos
dc = datacube.Datacube(app="Calculando_Diferenciales")
```

3) Establecer la zona de estudio

Delimitar la zona de estudio y mostrarla en un mapa. Es importante destacar que se deben obtener los puntos de latitud y longitud sobre los puntos superior derecho e inferior izquierdo del area que se desea estudiar. Se recomienda hacer uso del siguiente enlace para obtenerlos:

https://www.coordenadas-gps.com/

In [3]:
```
#Establecemos la zona de estudio
#latitud= (22.17824, 22.10644)
#longitud= (-102.39042, -102.49808)
latitud= (22.19443, 22.116693)
#la x viene de la coordenada de latitud del punto superior derecho
#la y viene de la coordenada de longitud del punto inferior izq
longitud= (-102.40785, -102.490565)
#la x viene de la coordenada de longitud del punto superior derecho
#la y viene de la coordenada de longitud del punto inferior izq


# Mostramos el area de estudio en maps
display_map(x=longitud, y=latitud)
```

Out[3]:

4) Definir un query de busqueda en el cubo de datos.

Una vez que ya tenemos definida nuestra area de estudio, creamos un query con una búsqueda que contendrá la lista de las variables que se usarán para cargar los datos satelitales desde el cubo de datos; las variables de interes que formarán la busqueda son: Longitud (X), Latitud (Y), Tiempo de interés y las mediciones de las bandas espectrales con las que se quieren trabajar.

In [4]:
```
# Creamos un query con una lista con las variables que buscamos
query = {'x': longitud,
         'y': latitud,
         'time': ('1987'), #2010 #1987
         'measurements': ['blue',
                          'green',
                          'nir',
                          'red',
                          'swir1',
                          'swir2']
         }
#time = 1987 = para def_geomedian_ls5, es decir para landSat 5
#nir = Infrarrojo cercano
#swir1 = Infrarrojo de onda corta 1
#swir2 = Infrarrojo de onda corta 2
```

5) Cargar los datos del query

Cargar los datos obtenidos en el query con la función dc.load y mostrarlos con la función RGB.

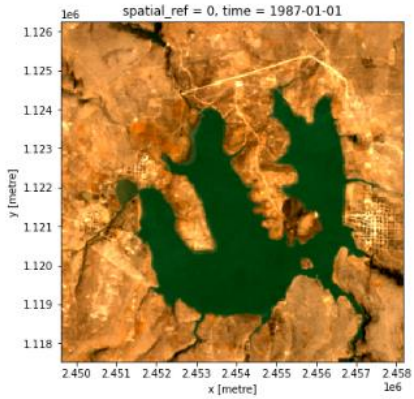
Parámetros de la función dc.load:

- Se especifica el satélite del cual se buscará la imagen; en este caso como el tiempo de interes es del año 2014, corresponde el satélite Landsat 8.
- El segundo parámetro es el apuntador al query de la busqueda que especificamos en la celda anterior.

Parámetros de la función RGB:

- El dataset que se obtuvo como resultado de la busqueda en la carga de datos.
- La(s) banda(s) que se quieren mostrar, puede ser desde una o hasta la combinación de 3 bandas. En este caso como se quiere mostrar en color verdadero son Red, Green, Blue.

In [5]:
```
# Cargamos los datos en una variable.
dataset = dc.load(product="def_geomedian_ls5", **query)
# ponemos para landSat5 ya que en time en el query pusimos el año 1987

#dataset = dc.load(product="def_geomedian_ls7", **query)

# Mostramos el area de estudio en verdadero color
rgb(dataset, bands=['red','green','blue'])
```



6) Calcular el índice diferencial normalizado de la vegetación NDVI

El Índice de Vegetación de Diferencia Normalizada, también conocido como NDVI por sus siglas en inglés, es un índice de vegetación que se utiliza para estimar la cantidad, calidad y desarrollo de la vegetación con base a la medición de la intensidad de la radiación de ciertas bandas del espectro electromagnético que la vegetación emite o refleja.

Para el cálculo de los índices de vegetación es necesaria la información que se encuentra en las bandas roja e infrarroja de ese espectro electromagnético.

El cálculo del NDVI se hace mediante la siguiente fórmula:

NDVI = (NIR – RED) / (NIR + RED)

NIR = infrarrojo cercano

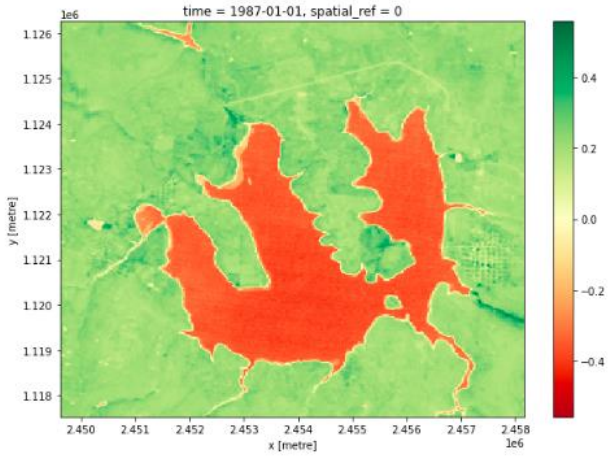
- El intervalo de valores posibles oscila entre -1 y 1.
- Los valores negativos están asociados a zonas de agua y nieve.
- Valores positivos próximos a 0 representan zonas rocosas y desnudas que pueden adquirir algo de vegetación hasta llegar a valores próximos a 0,3. A partir de este valor encontramos presencia de vegetación. Cuanto mayor sea el valor más frondosa será la vegetación hasta adquirir valores próximos a 1.

Tener en cuenta que, aspectos como la floración, la existencia de suelo expuesto, la presencia de masas de agua, o la pérdida de hojas pueden influir bruscamente en los valores del NDVI.

In [6]:
```
# Calcular el NDVI (índice diferencial normalizado de la vegetación) de las bandas nir y red
# Indice de vegetación
resta= dataset.nir - dataset.red
suma= dataset.nir + dataset.red

ndvi= resta/suma
```

Para mostrar el resultado del ndvi sobre la región de estudio ejecutamos la siguiente celda:

In [7]:
```
# Para mostrar el resultado
plt.figure(figsize=(10, 7))   # valores pequeños, área pequeña a mostrar, 13,10
ndvi.isel(time=0).plot(cmap="RdYlGn")
plt.show()
```



7) Calcular el índice diferencial de agua normalizado (NDWI)

El índice diferencial de agua normalizado o NDWI (Normalized Difference Water Index) permite conocer la hidratación de la vegetación y la humedad del suelo, es decir, la cantidad de agua que posee la vegetación o el nivel de saturación de humedad que posee el suelo.

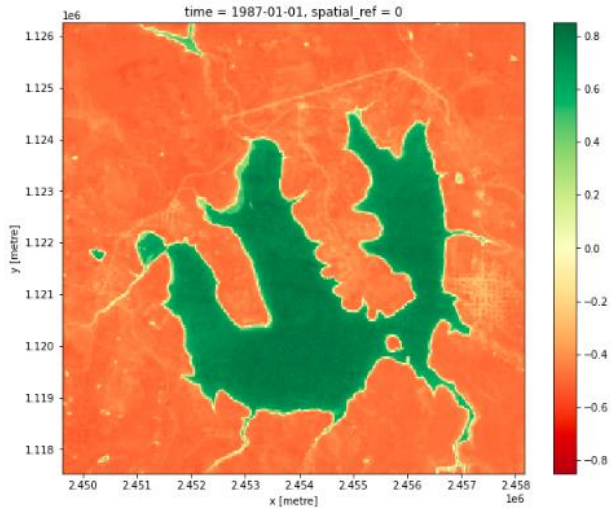
Se calcula a partir de imágenes satelitales que brindan información de reflectancia de una determinada zona en diferentes bandas de frecuencia del espectro electromagnético.

NDWI = (GREEN - SWIR1) / (GREEN + SWIR1)

- GREEN = Near InfraRed, es el valor en la banda verde.
- SWIR1 = ShortWavelength InfraRed, es el infrarrojo corto 1, muestra una alta absorción de luz debido al agua

swir1 = Infrarrojo de onda corta 1

In [8]:
```
# Calcular el indice diferencial de agua normalizado (NDWI), con las bandas nir y swir1
# Indice de agua
resta= dataset.green - dataset.swir1
suma= dataset.green + dataset.swir1

ndwi= resta/suma

#Para mostrarlo
plt.figure(figsize=(10, 8)) #para cambiar el tamaño de la imagen
ndwi.isel(time=0).plot(cmap="RdYlGn")
plt.show()
```



Nota: Es posible alterar el tamaño de la grafica de salida cambiando los valores de la función figsize, pero hay que tener cuidado de mantener una cierta proporción con el tamaño, ya que puede estirar la imagen.

8) Calcular el índice diferencial normalizado de edificación (NDBI)

El Índice de Diferencia Normalizada Edificada NDBI permite calcular la estimación de zonas con superficies edificadas o en desarrollo de construcción frente a las habituales zonas naturalizadas con vegetación o desnudas.

Es un índice de área construida, según la proporción de construcción en cada pixel. Sus valores van de -1 a 1, e indican, en forma creciente, el área construida.

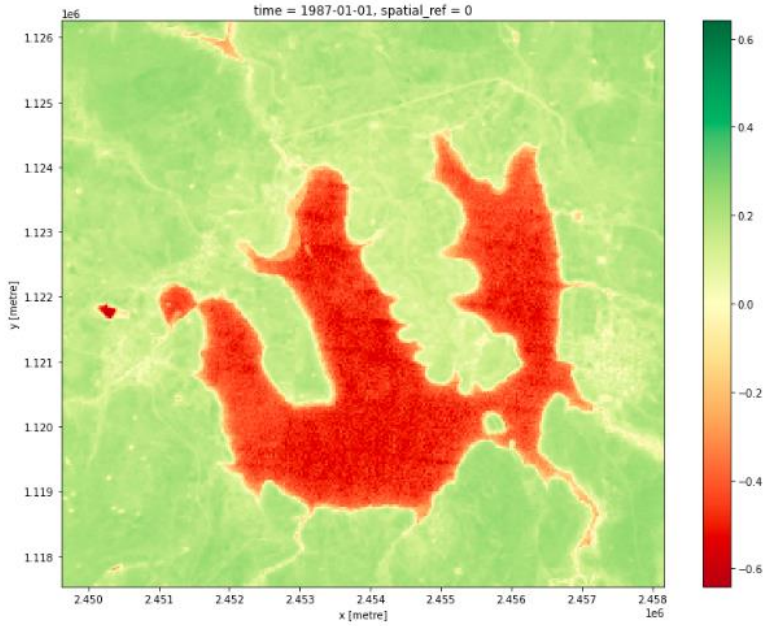
- Valores de tendencia negativa indican presencia de zonas con vegetación.
- Valores intermedios comienzan a determinar zonas desnudas, cultivos en crecimiento o zonas o en fase de construcción
- Valores de tendencia positivos elevados para indicar zonas territoriales con coberturas de suelo edificadas o infraestructuras antrópicas.

El cálculo del índice NDBI requiere de las bandas de análisis del infrarrojo a través de las bandas SWIR y NIR

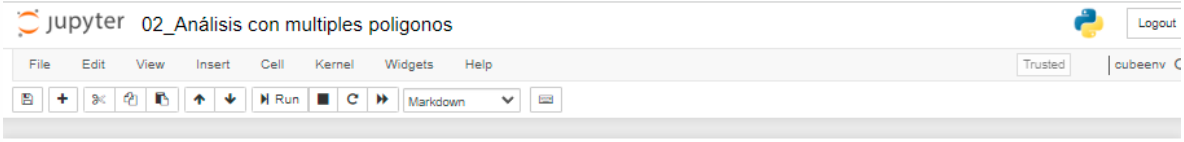
NDBI= (SWIR1 - NIR) / (SWIR1 + NIR)

swir1 = Infrarrojo de onda corta 1

In [9]:
```
# Calcular el diferencial normalizado de las bandas swir1 y nir (NDBI)
# Indice de edificación
resta= dataset.swir1 - dataset.nir
suma= dataset.swir1 + dataset.nir

ndbi= resta/suma

#Para mostrarlo
plt.figure(figsize=(13, 10))
ndbi.isel(time=0).plot(cmap="RdYlGn")
plt.show()
```



Anexo 4. 02_Análisis con multiples poligonos



Análisis con múltiples polígonos

Ejercicio:

Realizar un análisis vectorial a través de múltiples polígonos en un archivo vectorial (ESRI Shapefile o GeoJSON) utilizando **Open Data Cube** para extraer datos satelitales del estado de Ags. y sus municipios.

Pasos a seguir:

- Usar el **paquete de python geopandas** para abrirlo como un **GeoDataFrame**
- Luego, iterar a través de cada geometría y extraer datos de satélite correspondientes a la extensión de cada geometría.
- Realizar un análisis vectorial por cada conjunto de datos resultante.

Para el análisis vectorial: Calcular NDVI y trazar los datos.

Podemos recuperar datos para cada polígono y realizar lo siguiente:

```
1) Primero abrimos el polígono usando geopandas
2) Iterar a través de un geodatframe, extrayendo datos satelitales del estado de Ags y sus municipios.
3) Calcule NDVI.
4) Trazar NDVI para la extensión del polígono
```

1) Cargar paquetes

Uso de la geometría del **paquete datacube.utils**: es importante para guardar el sistema de referencia de coordenadas del archivo de forma entrante en un formato que la consulta pueda entender.

In [1]:
```
# 1) Importando las librerias necesarias
# ------------------------------------------------
import datacube
import rasterio.crs
import geopandas as gpd
import matplotlib.pyplot as plt
from datacube.utils import geometry
import sys
sys.path.append('../Scripts/')  #ojo con la ruta de Scripts
from dea_plotting import rgb, map_shapefile
from dea_bandindices import calculate_indices
from dea_spatialtools import xr_rasterize
from dea_temporaltools import time_buffer
from dea_datahandling import load_ard
```

```
/opt/conda/envs/cubeenv/lib/python3.6/site-packages/datacube/storage/masking.py:4: DeprecationWarning: datacube.storage.masking
has moved to datacube.utils.masking
  category=DeprecationWarning)
```

2) Conectarse al datacube

Conéctarse a la base de datos del cubo de datos para habilitar la carga de datos.

In [2]:
```
# Creamos una conexión con el cubo de datos
dc = datacube.Datacube(app="Analizando_un_poligono_municipal_prueba")
```

3) Declarar los parámetros de análisis

time_of_interest: Ingrese fecha, en unidades AAAA-MM-DD, alrededor de la cual cargar datos satelitales, p. '2019-01-01'

time_buff: un búfer de una duración determinada (por ejemplo, días) alrededor del parámetro time_of_interest, por ejemplo '30 dias'

vector_file: una ruta a un archivo vectorial (ESRI Shapefile o GeoJSON)

attribute_col: una columna en el archivo vectorial utilizada para etiquetar los conjuntos de datos de la matriz de salida que contienen imágenes de satélite. Cada fila de esta columna debe tener un identificador único

products: una lista de nombres de productos para cargar desde el cubo de datos, ej. ['ga_ls7e_ard_3', 'ga_ls8c_ard_3']

measurements (mediciones): una lista de nombres de bandas para cargar desde el producto satelital, ej: ['nbart_red', 'nbart_green']

resolution: la resolución espacial de los datos de satélite cargados, p. para Landsat, esto es (-30, 30)

output_crs: el sistema de referencia de coordenadas / proyección del mapa para cargar datos, p. 'EPSG: 3577' para cargar datos en la proyección Albers Equal Area

align (alinear): Cómo alinear las coordenadas x, y con respecto a cada píxel. Landsat Collection 3 debe alinearse al centro alinear = (15, 15) si los datos se cargan en su proyección de zona UTM nativa, p. 'EPSG: 32756'

In [3]:
```
# Parametros de análisis
#-----------------------------------------------------------------------

# Periodo de años que nos intera analizar

# entre [] = toma cada elemento declarado dentro de los []
time_of_interest = ['1993', '1994', '1995', '1996', '1997', '1998', '1999',
                    '2000', '2001', '2002', '2003', '2004', '2005', '2006',
                    '2007', '2008', '2009', '2010', '2011', '2012', '2013',
                    '2014', '2015', '2016', '2017', '2018', '2019']



# 2000 al 2019

# entre () toma del año inicial al año final
#time_of_interest = ('2010','2012')

vector_file = '../SHP/AGS_mun_AEA.shp' # Archivo shp o Geojson con los poligonos
attribute_col = 'CVEGEO' # Columna identificador de archivo = contiene la clave del municipio cve_edo + cve_edo_mun
products = ['def_geomedian_ls5','def_geomedian_ls7','def_geomedian_ls8'] # Producto del cubo en la base de datos correspondiente
#measurements = ['blue','green','nir','red','swir1','swir2'] # Bandas en el orden que se encuentran en la Geomediana
measurements = ['nir','red'] # Bandas para cálculo de NDVI
```

4) Mostrar la estructura del archivo vectorial

Para checar cómo está estructurado el archivo para entender qué estamos iterando.

In [4]:
```
# Mostrar la estructura del archivo vectorial
gdf = gpd.read_file(vector_file)
#print(gdf)    # muestra la info de todos los mpios.
gdf.head()    # muestra la info de los primeros 5 registros, en formato de tabla
```

Out[4]:

	CVEGEO	CVE_ENT	CVE_MUN	NOMGEO	geometry
0	01001	01	001	Aguascalientes	POLYGON ((2489072.503 1115771.584, 2489352.774...
1	01002	01	002	Asientos	POLYGON ((2494680.261 1141224.505, 2494749.948...
2	01003	01	003	Calvillo	POLYGON ((2429607.454 1120262.254, 2429071.902...
3	01004	01	004	Cosío	POLYGON ((2470517.824 1155028.588, 2470552.248...
4	01005	01	005	Jesús María	POLYGON ((2465526.729 1114740.466, 2465752.546...

5) Crear un objeto de consulta de cubo de datos

Luego creamos un diccionario (query con una búsqueda) que contendrá los parámetros que se usarán para cargar datos desde el cubo de datos.

In [5]:
```
# Creamos una busqueda
query = {'time': time_of_interest,
         'measurements': measurements
         }
```

6) Cargar los datos satelitales

Aquí iteraremos por cada fila de **geopandas.GeoDataFrame** y cargaremos los datos satelitales. Los resultados se agregarán a un objeto de diccionario para analizar cada conjunto de datos.

In [6]:
```
# Diccionario (query) se guarda en varable results
results = {}

# Recorrer los polígonos en el geodataframe y extraer datos satelitales
for index, row in gdf.iterrows():

    print(f'Feature: {index + 1}/{len(gdf)}')

    # Extraer la geometría por cada registro (renglón) como un objeto datacube geometry
    geom = geometry.Geometry(geom=row.geometry, crs=gdf.crs)

    # Actualizar the query para incluirlo en el geopolygon
    query.update({'geopolygon': geom})

    # Load landsat
    ds = load_ard(dc=dc,
                  products=products,
                  ls7_slc_off = False,
                  mask_pixel_quality=False,
                  mask_contiguity=False,
                  group_by='solar_day',
                  **query)
```

```
                mask_contiguity=False,
                group_by='solar_day',
                **query)

    # Generar un polygon mask para mantener los datos dentro del polígono
    mask = xr_rasterize(gdf.iloc[[index]], ds)

    # Mask dataset to set pixels outside the polygon to `NaN`
    # Enmascara el dataset (conjunto de datos) para establecer los píxeles fuera del polígono con `NaN`
    ds = ds.where(mask)

    # Append results to a dictionary using the attribute
    # Agregar los resultados a un diccionario usando el atributo
    # column as an key
    results.update({str(row[attribute_col]): ds})
    print("-------------------------------------------------------")
```

```
Feature: 1/11
Loading def_geomedian_ls5 data
    Applying invalid data mask
Loading def_geomedian_ls7 data
    Applying invalid data mask
Loading def_geomedian_ls8 data
    Applying invalid data mask
Combining and sorting data
    Returning 27 observations
Rasterizing to match xarray.DataArray dimensions (1680, 1780)
-------------------------------------------------------
Feature: 2/11
Loading def_geomedian_ls5 data
    Applying invalid data mask
Loading def_geomedian_ls7 data
    Applying invalid data mask
Loading def_geomedian_ls8 data
    Applying invalid data mask
Combining and sorting data
```

7) Obtener el índice normalizado diferencial de la vegetación NDVI

El Índice de Vegetación de Diferencia Normalizada, también conocido como NDVI por sus siglas en inglés, es un índice de vegetación que se utiliza para estimar la cantidad, calidad y desarrollo de la vegetación con base a la medición de la intensidad de la radiación de ciertas bandas del espectro electromagnético que la vegetación emite o refleja.

Para el cálculo de los índices de vegetación es necesaria la información que se encuentra en las bandas roja e infrarroja de ese espectro electromagnético.

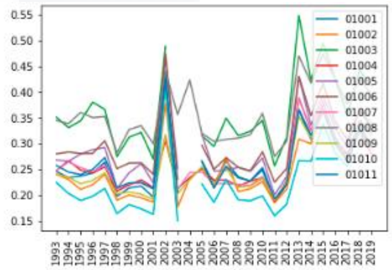
El cálculo del NDVI se hace mediante la siguiente fórmula:

NDVI = (NIR – RED) / (NIR + RED)

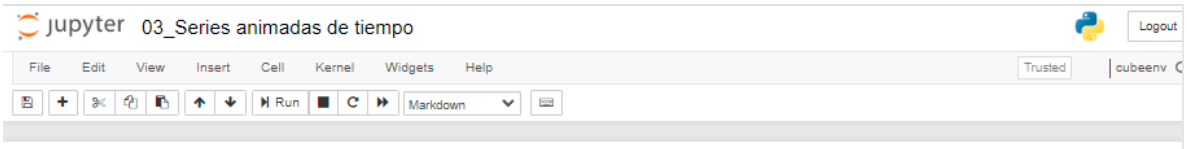
NDVI = (Banda infrarroja cercana – Banda roja) / (Banda infrarroja cercana + Banda roja)

```
In [7]: # 7) Obtener el NDVI
        # --------------------------------------------------------------

        #nos apoyó con su experiencia en la graficacion del resultado

        #valores de cada año y cada mpio
        ndvis= []
        for cve, data in results.items():

            resta= results[cve].data_vars['nir'] - results[cve].data_vars['red']
            suma= results[cve].data_vars['nir'] + results[cve].data_vars['red']

            ndvi= resta/suma
            mean = ndvi.mean(dim=['x', 'y'])
            ndvis.append(mean)

            # 8) Graficar el NDVI
            # ------------------------------------------------------
            plt.plot(mean.data, label = cve)
            plt.legend(loc='upper right')
            plt.xticks(range(len(time_of_interest)), time_of_interest, rotation='vertical')
```

```
/opt/conda/envs/cubeenv/lib/python3.6/site-packages/xarray/core/nanops.py:142: RuntimeWarning: Mean of empty slice
  return np.nanmean(a, axis=axis, dtype=dtype)
/opt/conda/envs/cubeenv/lib/python3.6/site-packages/xarray/core/nanops.py:142: RuntimeWarning: Mean of empty slice
  return np.nanmean(a, axis=axis, dtype=dtype)
/opt/conda/envs/cubeenv/lib/python3.6/site-packages/xarray/core/nanops.py:142: RuntimeWarning: Mean of empty slice
  return np.nanmean(a, axis=axis, dtype=dtype)
/opt/conda/envs/cubeenv/lib/python3.6/site-packages/xarray/core/nanops.py:142: RuntimeWarning: Mean of empty slice
  return np.nanmean(a, axis=axis, dtype=dtype)
/opt/conda/envs/cubeenv/lib/python3.6/site-packages/xarray/core/nanops.py:142: RuntimeWarning: Mean of empty slice
  return np.nanmean(a, axis=axis, dtype=dtype)
/opt/conda/envs/cubeenv/lib/python3.6/site-packages/xarray/core/nanops.py:142: RuntimeWarning: Mean of empty slice
  return np.nanmean(a, axis=axis, dtype=dtype)
```



Anexo 5. 03_Series animadas de tiempo



Creación de series animadas de tiempo = ejemplo a mostrar

Las animaciones pueden ser un método poderoso para visualizar cambios en el paisaje a través del tiempo usando imágenes satelitales. Los datos satelitales obtenidos a partir del cubo de datos geoespacial de INEGI son ideales para realizar estas animaciones, ya que han sido georreferenciados, se procesó la reflectancia de superficie y se tienen en una base de datos espacio-temporal, permitiendo que las condiciones del paisaje se extraigan y visualicen de manera consistente a lo largo del tiempo.

Usando la función `xr_animation` del script `dea_plotting`, ubicado en la carpeta "Scripts" de la raíz de este Jupyter, es posible tomar una serie de imágenes satelitales del cubo de datos geoespacial y exportar una animación de series de tiempo visualmente atractiva que muestre cómo ha cambiado cualquier ubicación en México en los últimos años; de 1993 a 2019.

Descripción

Este cuaderno muestra cómo:

1. Importar una serie temporal de imágenes satelitales libres de nubes desde múltiples satélites.
2. Mostrar los datos como una animación de series de tiempo de tres bandas
3. Mostrar los datos como una animación de series de tiempo de una banda
4. Exporte las animaciones resultantes como un archivo GIF.
5. Aplique funciones de procesamiento de imágenes personalizadas a cada cuadro de animación

Para empezar...

Para ejecutar este análisis, ejecute todas las celdas en el cuaderno, comenzando con la celda "Cargar paquetes"

1) Cargar paquetes

Se importan los elementos necesarios para el correcto funcionamiento de este cuaderno.

In [19]:
```
import sys
import datacube
import skimage.exposure
import geopandas as gpd
import matplotlib.pyplot as plt
from IPython.display import Image
sys.path.append('../Scripts/')
from dea_bandindices import calculate_indices
from dea_datahandling import load_ard
from dea_plotting import rgb
from dea_plotting import xr_animation
```

2) Crear conexión con el cubo de datos geoespacial

Conéctarse a la base de datos del cubo de datos para habilitar la carga de datos.

In [28]:
```
dc = datacube.Datacube(app='Serie Animada de Tiempo')
```

3) Cargar datos de satélite del cubo de datos

Podemos usar la función `load` para cargar datos de múltiples satélites y devolver una única matriz de datos que contiene solo observaciones con un porcentaje mínimo de píxeles de buena calidad. Esto nos permitirá crear una animación de series de tiempo visualmente atractiva de observaciones que no se ven afectadas por nubes.

En el siguiente ejemplo, podemos ver como se establece una busqueda (query); en la cual se está buscando en los satelites landsat 7 y 8, imágenes correspondientes al perido de los años de 2011 a 2014 de la presa Calles en el estado de Aguascalientes.

In [29]:
```
# Set up a datacube query to load data for
query = {'product': ['def_geomedian_ls8','def_geomedian_ls7'],
         #definir area de estudio
         'x': (-102.39042, -102.49808),
         'y': (22.17824, 22.10644),

         'time': ('2011-01-01', '2014-01-01'),
         'measurements': ['blue',
                          'green',
                          'nir',
                          'red',
                          'swir1',
                          'swir2']
         }

ds = dc.load(**query)
```

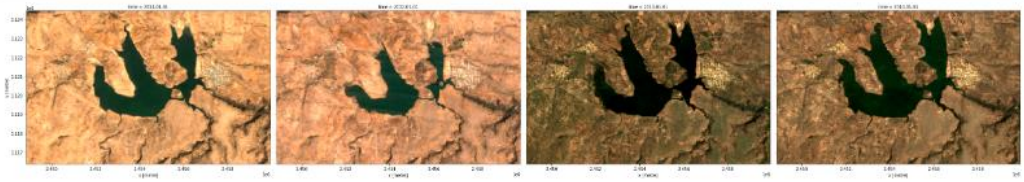
4) Mostrar los datos con la función RGB

Para tener una idea rápida de cómo se ven los datos, podemos mostrarlos en color verdadero usando la función `rgb` .

La función rgb puede recibir los siguientes parámetros:

- ds [x_array dataset]: Es un array de 2 o más imágenes para mostrar en RGB, si es de más de 2 imágenes será necesario agregar el parametro "index" para especificar cuales imágenes son las que se quieren gráficar.
- bands [String]: Es una lista de maximo 3 Strings que contengan el nombre de las bandas a analizar. Por default son los valores: 'nbart_red', 'nbart_green' y 'nbart_blue'
- index [integer]: Es un entero o lista de enteros que puede usarse para seleccionar uno o varios elementos de la lista del ds.
- size [integer]: Es un entero para especificar el tamaño en pulgadas de cada gráfica. Por default es 6.
- savefig_path [String]: Es un string con la ruta donde se desea guardar la gráfica de salida. Por default es "None", por lo que no se guardará la imagen a menos que asi se desee.

In [30]:
```
rgb(ds, bands=['red','green','blue'], index=[0,1,2,3])
```



5) Generar un gif animado con la serie de tiempo.

La función `xr_animation` se basa en la función dentro de matplotlib.animation. Toma una matriz de datos y exporta una animación GIF o MP4 de una o tres bandas que muestra cambios en el paisaje a lo largo del tiempo.

Aquí, graficamos el conjunto de datos que cargamos arriba como un GIF animado, usando las bandas satelitales ['red', 'green', 'blue'] para generar una animación RGB de color verdadero. El intervalo entre los cuadros de animación se establece en 100 milisegundos con intervalo, y el ancho de la animación en 300 píxeles con width_pixels:

In [32]:
```
xr_animation(ds=ds,
             bands=['red', 'green', 'blue'],
             output_path='animated_timeseries.gif',
             interval=200, #500 velocidad entre cada imagen
             width_pixels=700)  #500

# Plot animated gif
plt.close()
Image(filename='animated_timeseries.gif')
```

```
Exporting animation to animated_timeseries.gif
```

Out[32]:

- ds [x_array dataset]: Es un array de 2 o más imágenes para mostrar.
- bands [String]: Es una lista de maximo 3 Strings que contengan el nombre de las bandas a analizar, importante destacar, que esas bandas deben estar disponibles en el dataset (ds).
- output_path [String]: Un string con la ruta donde se almacenará el archivo resultante, ya sea *.gif* o *.mp4*.
- width_pixels [Integer]: Es el ancho en pixeles de la figura resultante.
- interval [Integer]: Es un entero que establece el intervalo en milisegundos entre los cuales se debe de cambiar de imagen. El valor por default es de 100.
- show_date [bool o String]: (Opcional) Define como publicar la fecha en la animación, por default es: "%d %b %y", pero se puede adaptar a algun formato especifico. Tambien puede tomar el valor de *False* para no mostrar la fecha en la imagen.
- show_text [String]: (Opcional) Es un string que puede ser mostrado durante la animación que se generé.

In [33]:

```
# Produce time series animation of red, green and blue bands
xr_animation(ds=ds,
             bands=['red', 'green', 'blue'],
             output_path='animated_timeseries2.gif',
             width_pixels=300,
             show_text='**** Presa Calles ****',
             show_date='20%y',
             interval=700)

# Plot animated gif
plt.close()
Image(filename='animated_timeseries2.gif')
```

```
Exporting animation to animated_timeseries2.gif
```

Out[33]:

El siguiente ejemplo es solo para mostrar como sería crear una animación con una sola banda; en este caso la banda "red".

In [34]:

```
xr_animation(ds=ds,
             bands=['red'],
             output_path='animated_timeseries3.gif',
             width_pixels=500,
             show_text='Presa Calles',
             show_date='año='+'20%y',
             interval=300)

# Plot animated gif
plt.close()
Image(filename='animated_timeseries3.gif')
```

```
Exporting animation to animated_timeseries3.gif
```

Out[34]: