Methane Emission Sensing Tools - How To Guide

Nicholas Kinsella

Ulster University

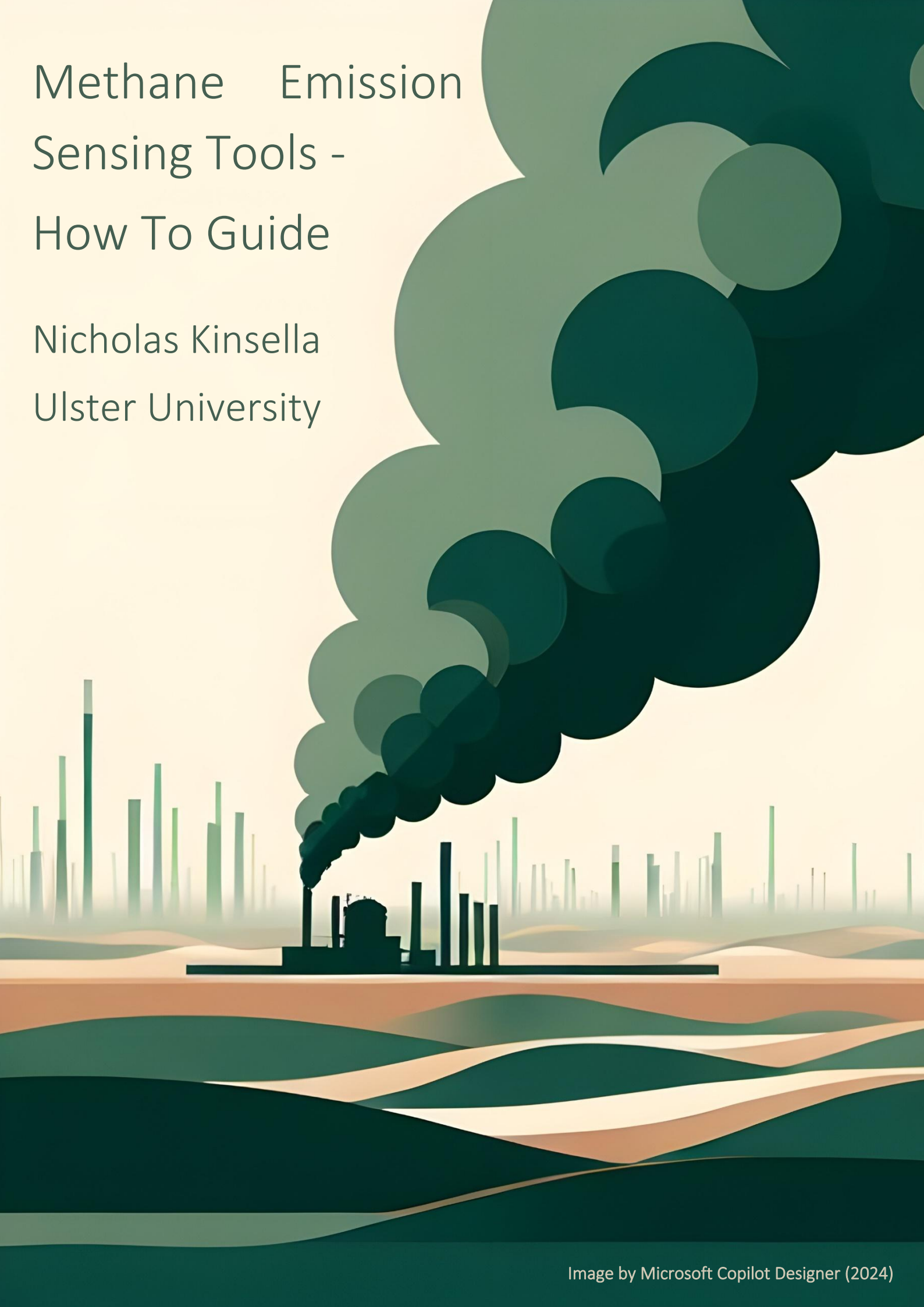


Image by Microsoft Copilot Designer (2024)

Table of Contents

1. Introduction

Methane (CH_4), a greenhouse gas with a warming potential 28 times greater than carbon dioxide over a 100-year period, has seen its atmospheric concentration increase over 250% since the industrial revolution. Despite CH_4's shorter atmospheric lifespan, it still has contributed to at least a quarter of anthropogenic warming since the Industrial Revolution (Pandey et al., 2023; Vigano et al., 2008).

Mismanaged oil and gas fields can produce significant CH_4 emissions (Jackson et al., 2020; Pandey et al., 2023) the 2021 Global Methane Pledge (GMP), was created to reduce anthropogenic emission levels by 30% of 2020 levels by 2030, with 157 countries participating (GMP, 2024; International Energy Agency, 2022; Malley et al., 2023).

In 2022 Algeria made a pledge to reduce its GHG emissions by between 7% and 22% (United Nations Development Programme, 2022) and the European Union has plans to launch a pilot scheme to encourage the Algerian state petrochemical company Sonatrach, to capture CH_4 under the 'You collect, we buy' scheme by the end of 2024 (European Commission,2023). Despite this, as of 2024, Algeria has yet to join the GMP and currently does not have government led commitments to reduce CH_4 emissions specifically (International Energy Agency, 2024).

Managed by state-owned company, Sonatrach, there are well over a hundred identified oil and gas fields in Algeria, two of which are considered giant (Abada et al., 2018). Situated in northern Algeria, 550 km south of Algers (fig.1), the Hassi R'Mel gas field covers an area of around 2,900km^2 and is a key resource for Europe, producing around 45% of Algeria's national gas production (Abada and Bouharkat, 2018; Naus et al., 2023; Rosenthal., 2023; Talamali., 2016). A further 350 km further to the southeast lies the Hassi Messaoud oil field, which covers 2,250 km^2 and produces around 36% of Algeria's national oil production (fig.1) (Kamr Eddine Aissou., 2024; Naus et al., 2023). Despite being an oil field, Hassi Messaoud is by far the worse emitter of CH_4 emissions, with Sentinel 5P detecting 154 super emissions since 2019, compared to just 8 in Hassi R'Mel over the same period, (Kayrros, 2024; Naus et al., 2023). This is backed up by the study of Naus et al. (2023) which detected 1 super-emitter event in Hassi R-Mel and 9 in Hassi Messaoud for the year 2020. Another study by Varon et al. (2021) detected 101 plumes between the 9th of October 2019 and 9th of August 2020 at Hassi Messaoud. This implies that current management practices of CH_4 emissions at Hassi Messaoud could be improved.

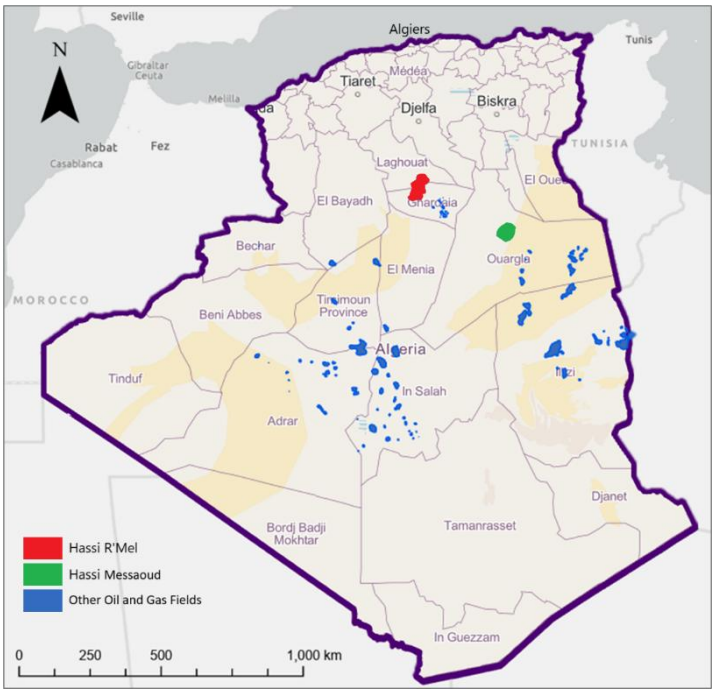


Figure 1: Hassi Messaoud oil field in Green (Lat 31.69, Long 6.03) and Hassi R'Mel gas field in Algeria in red (Lat 32.94, Long 3.27). (Field locations produced by manual survey of satellite images by author.)

Satellite missions have in recent years improved their CH_4 measurement capabilities, offering advantages over ground-based detectors (Parker et al., 2011). This guide outlines the use of two tools that can be used to monitor methane emissions.

- **The Integrated Methane Inversion (IMI):** An Amazon Web Services based tool for measuring methane emissions using the Sentinel-5P methane product and the GEOS-Chem model of atmospheric chemistry (Varon et al., 2022). Full details of the tool can be found here: https://imi.readthedocs.io/
- **Sentinel-2 Multi-Band-Multi-Pass CH_4 Mapper (S2MBMP):** This creates a high-resolution map that can show super emitting CH_4 point sources for selected Algerian petrochemical fields for a specific date. This tool can be downloaded or cloned from the following Github repository: https://github.com/zelcon01/Sentinel-2_Algeria_Methane/

This guide outlines their installation and use.

2. Integrated Methane Inversion (IMI)

2.1 Methodology

The Sentinel-5P Tropospheric Monitoring Instrument (TROPOMI) Methane product provides atmospheric CH4 concentrations in parts per billion (ppb). Its data has been shown to be in close agreement with ground-based measurements from the Total Carbon Column Observing Network (Liu et al., 2021; Lorente et al., 2021). Additional steps are required to estimate ground level CH4 emissions from these observed whole atmosphere concentrations, because CH4 can travel significant distances from its source and may be present in high altitudes over otherwise low-emission areas, and vice versa (Varon et al., 2022), resolving this requires an atmospheric transport model to be applied the data

The IMI (Integrated Methane Inversion) is a Python-based tool developed by Varon et al. (2022) on Amazon Web Services to estimate emissions in a selected area (fig. 2). Official bottom-up emission inventories of CH_4 are known to often be inaccurate (Dowd et al., 2023; Elkind et al., 2020; Karimi et al., 2021; Sherwin et al., 2024; Vaughn et al., 2018; Wang et al., 2024). The IMI seeks to improve on the bottom-up inventories by comparing them to top-down measurements by Sentinel -5P. Thereby providing a more accurate figure of emissions than official figures.

The inversion process starts with a "prior estimate of CH_4 emissions" based on official records such as the Global Fuel Exploitation Inventory (Scarpelli et al., 2022). Like all gasses CH_4 is affected by weather and other atmospheric influences, so next an atmospheric transport model is applied to the prior estimate data to create a model of predicted atmospheric concentrations, that should be seen by Sentenel-5P. The prediction is then compared to Sentenel-5P's real observations. The "inversion" involves adjusting the predicted concentrations so that they better match the observed concentrations. This results in a "posterior estimate" which should be a better reflection of reality.

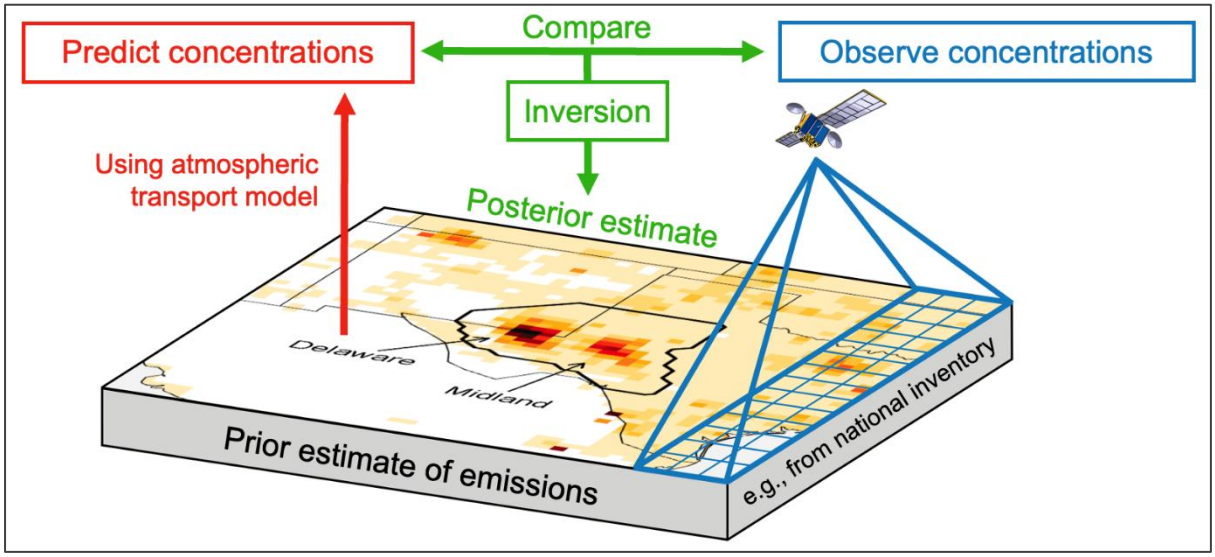


Figure 2: IMI processing step (Integrated Methane Inversion, 2024).

2.2 Setup

The following steps will show you how to setup a AWS account and run the IMI tool

2.2.1 Creating an AWS account

Creating an AWS account requires providing some personal and bank card information, as it is not a free service. The cost of running the IMI will be provided as an estimate prior to you running the full analysis. Open the following link in a browser http://aws.amazon.com/ and click on create an AWS account (fig. 3). Thereafter, follow the prompts.

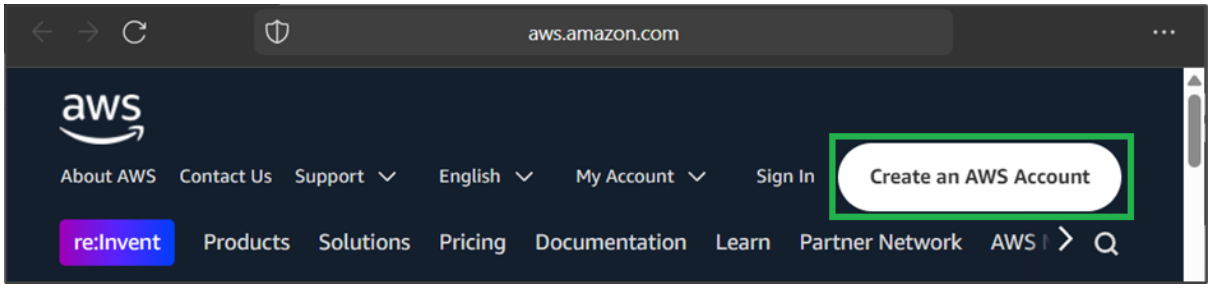


Figure 3: AWS sign in and create account landing page.

2.2.2 Add Amazon Simple Storage Service (S3) user permissions

This is an optional step if you wish to store your results long term. Firstly navigate to the AWS management console by clicking on the "Sign In to the Console" button or by using the "My Account" dropdown (fig. 4).

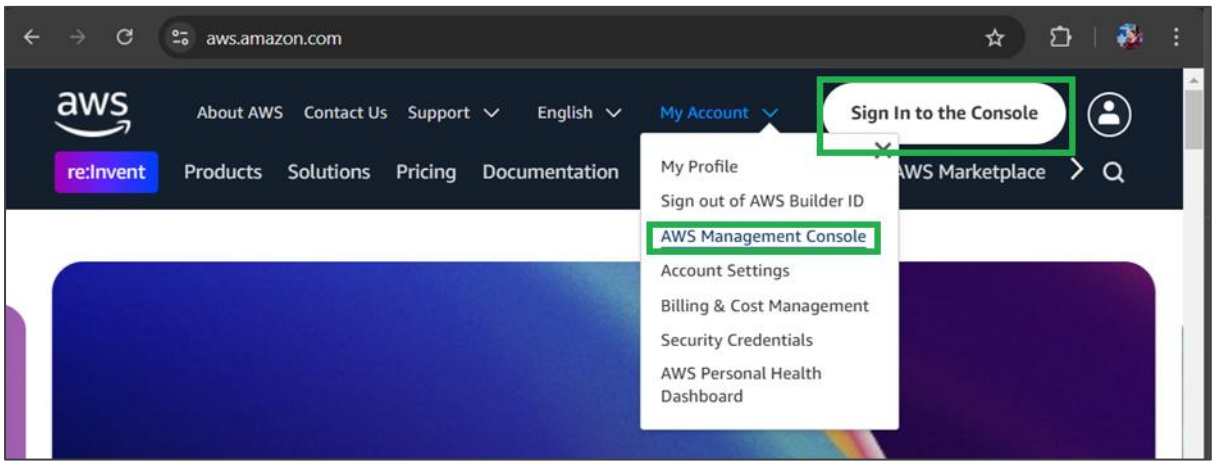


Figure 4: Sign in to my Console access on AWS landing page.

Once on the management console page, click the magnifying glass (fig. 5)

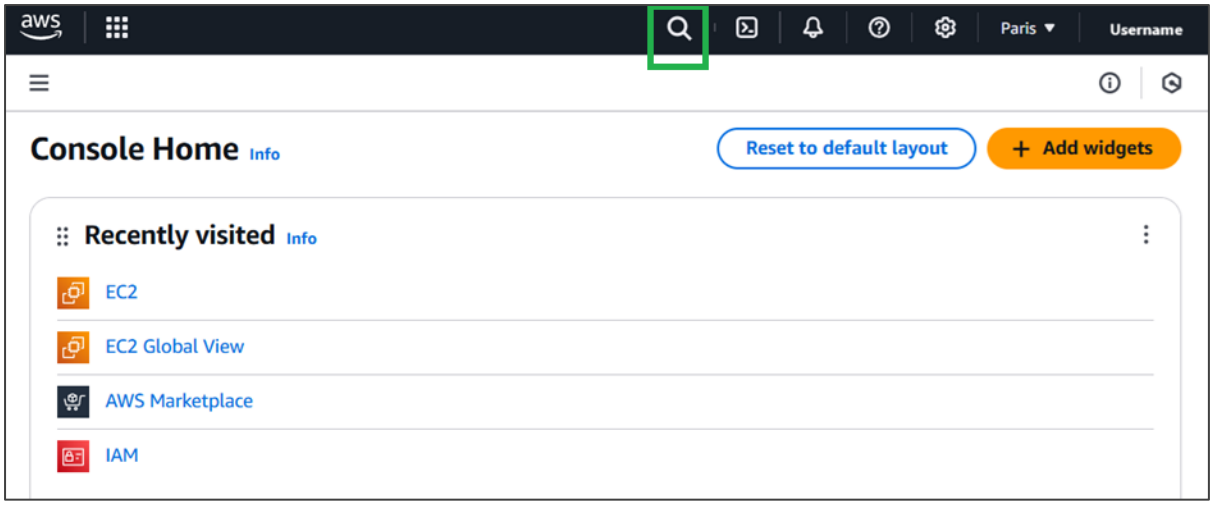


Figure 5: Location of search function in AWS console.

Then search for "IAM" (fig. 6).

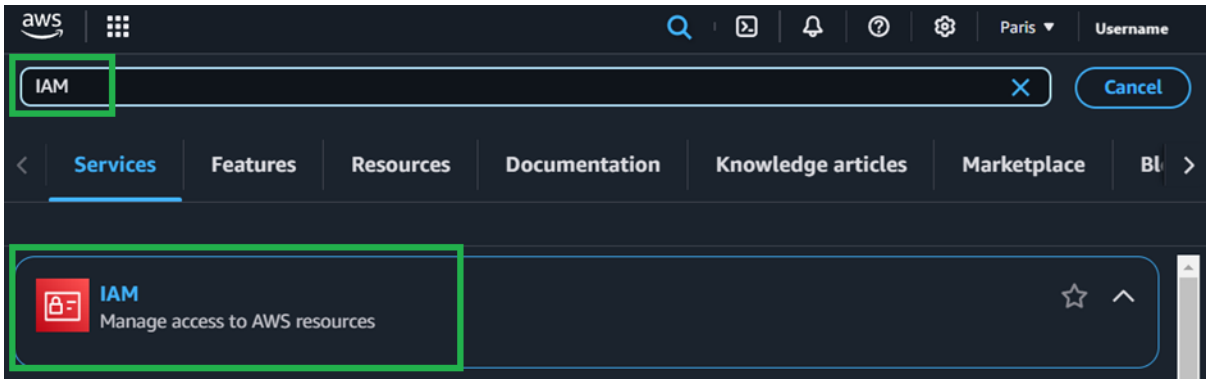


Figure 6: search function finding the AWS resource management access.

Under the "Access management" sidebar, click on Roles and then "Create role" (fig. 7)

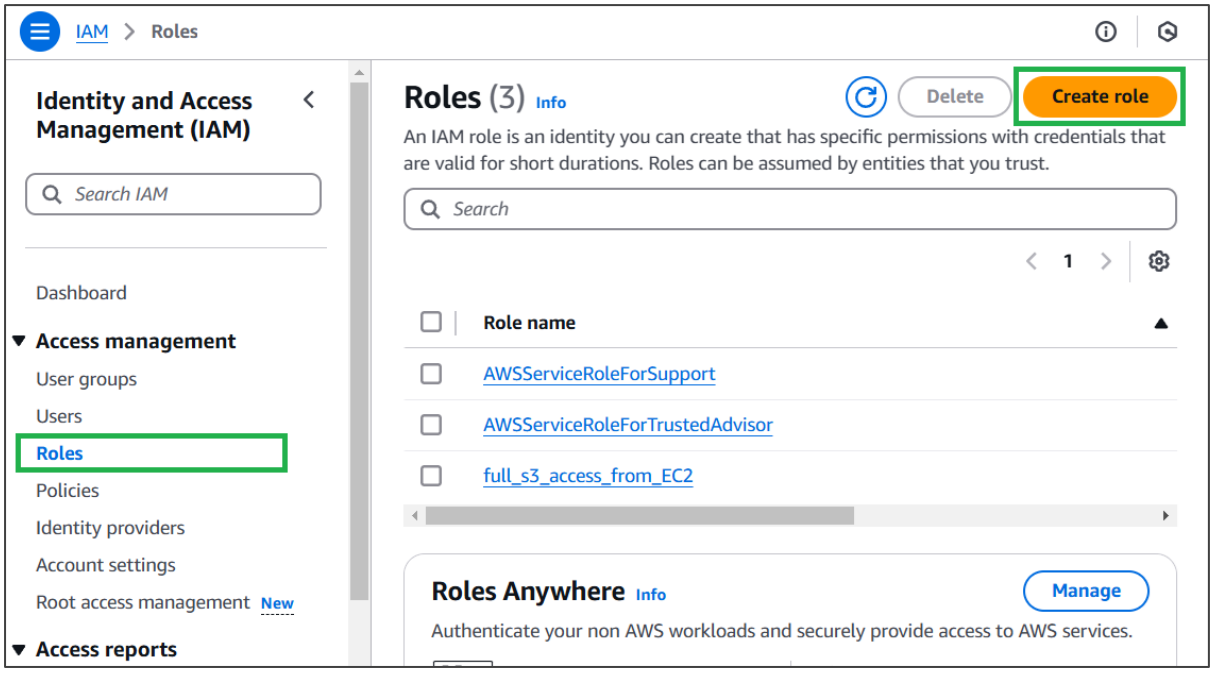


Figure 7: create role button location

In the “Use Case”, select EC2 and then click “next” (fig. 8).

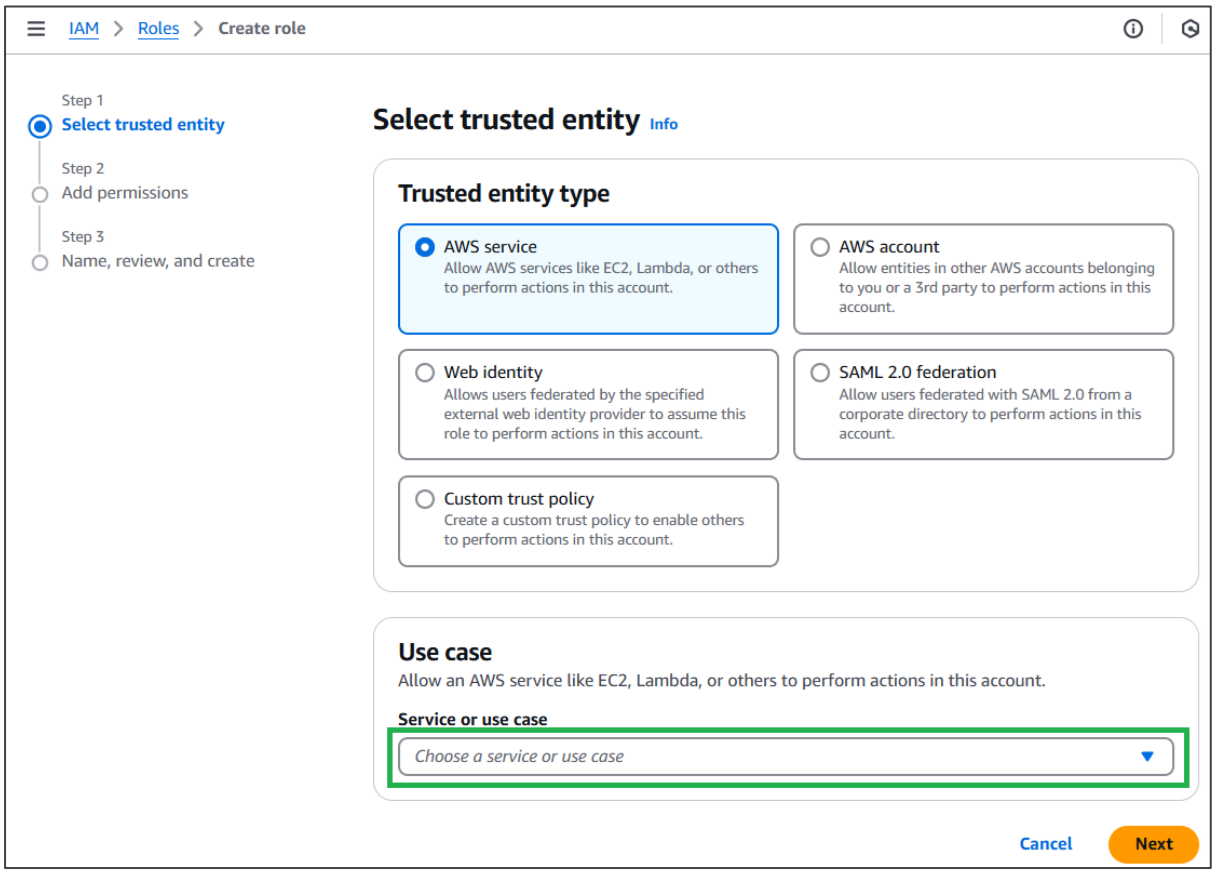


Figure 8: Use case selection location.

In the “Add permissions” screen type S3 in the search box and select AmazonS3FullAccess (fig. 9). Click next.

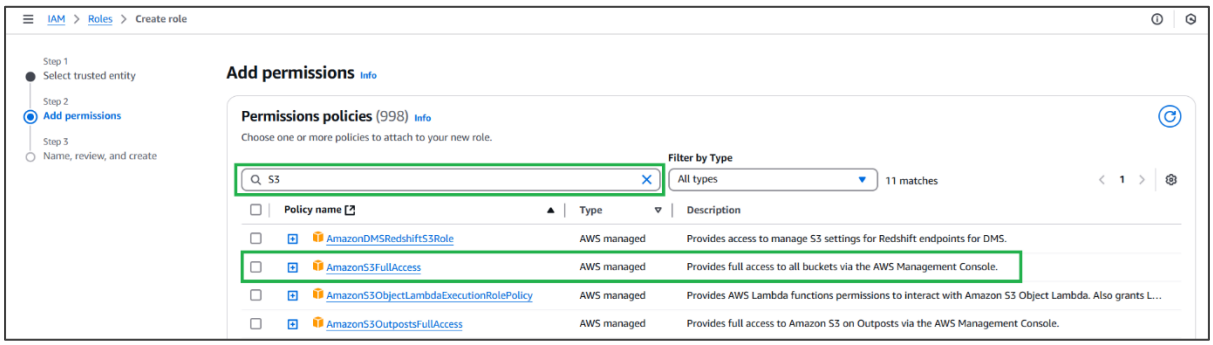


Figure 9: Use case selection location.

Lastly, choose a descriptive name for the role like “full_S3_access_from_EC2”. Click “create role” at the bottom of the page (fig. 10).

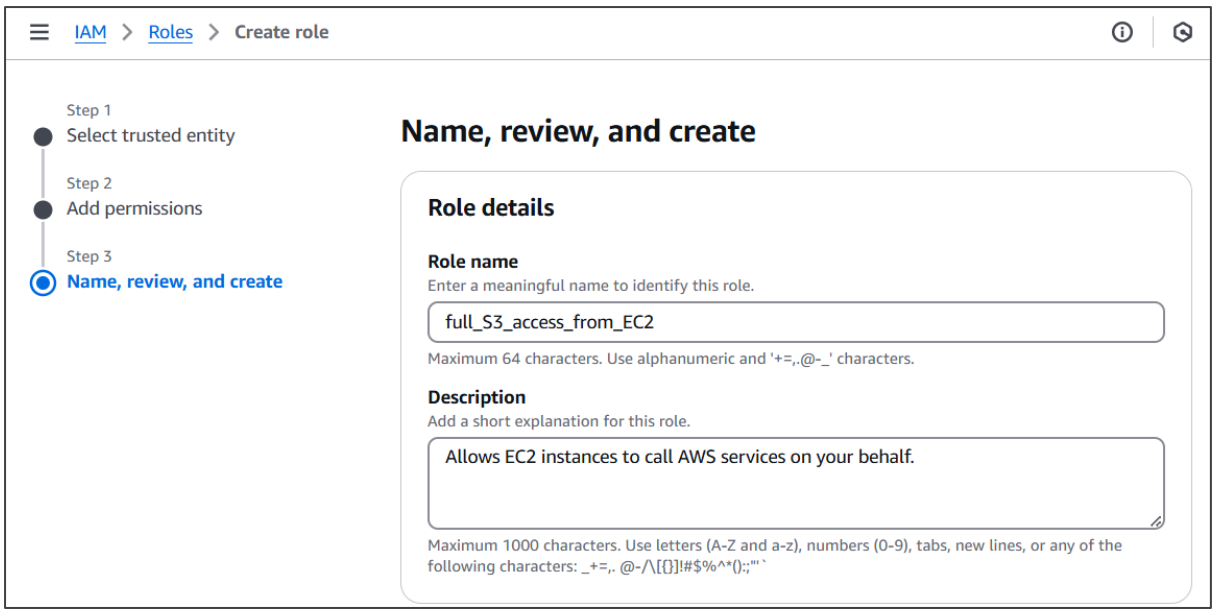


Figure 10: Create roll naming screen.

Now a new IAM role is created.

2.2.3 Launching an IMI instance

Navigate to https://aws.amazon.com/marketplace/pp/prodview-hkuxx4h2vpjba or search for "Integrated Methane Inversion". You should find the page show in figure 11. Click "view purchase options"

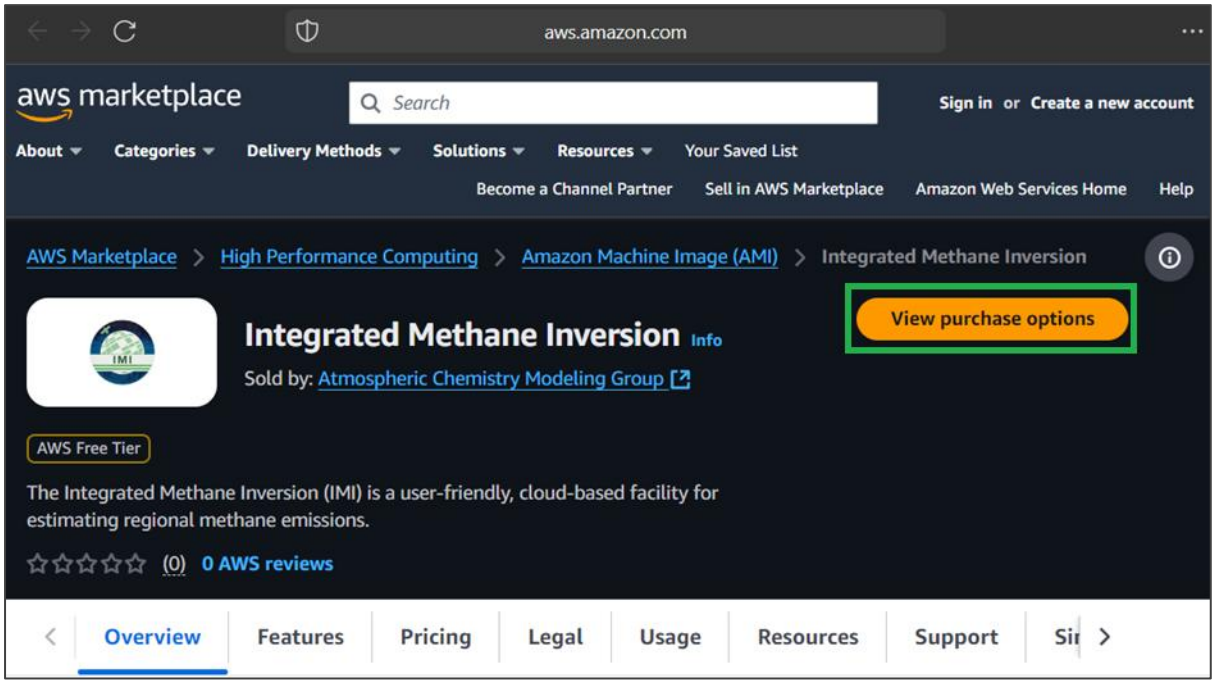


Figure 11: IMI landing page and purchase options button.

Next click "continue to configuration" (fig. 12).

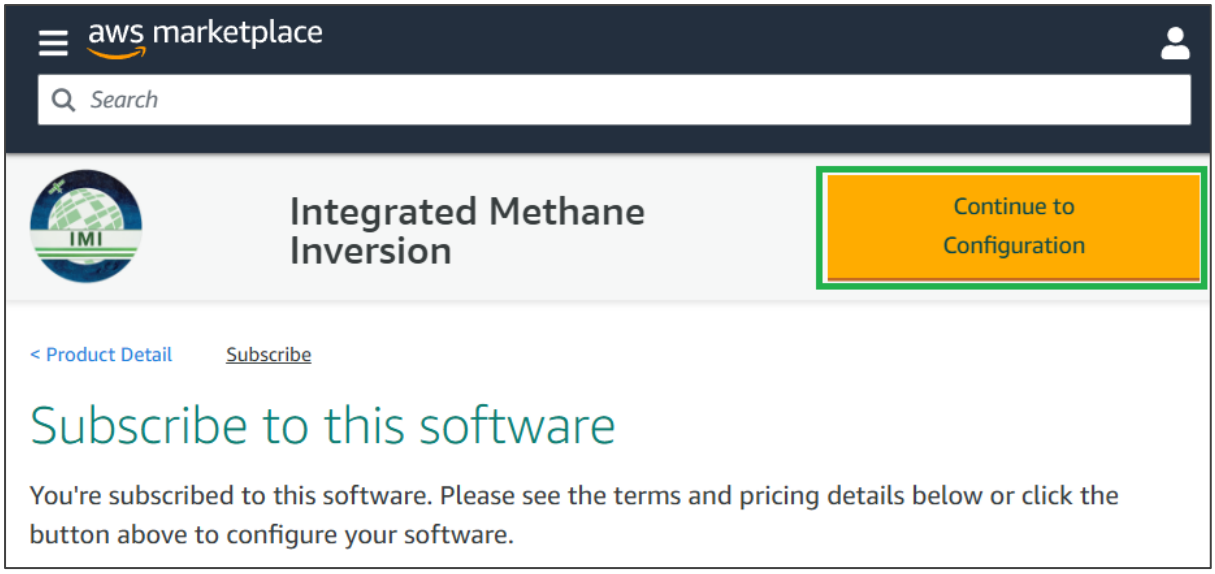


Figure 12: Subscription page with continue to configuration button.

Choose your preferred region and IMI version, then click *"Continue to Launch."* (fig. 13) Opting for a region closer to your physical location can enhance network connectivity but may lead to higher costs compared to selecting the region where GEOS-Chem data is hosted (us-east-1, N. Virginia).

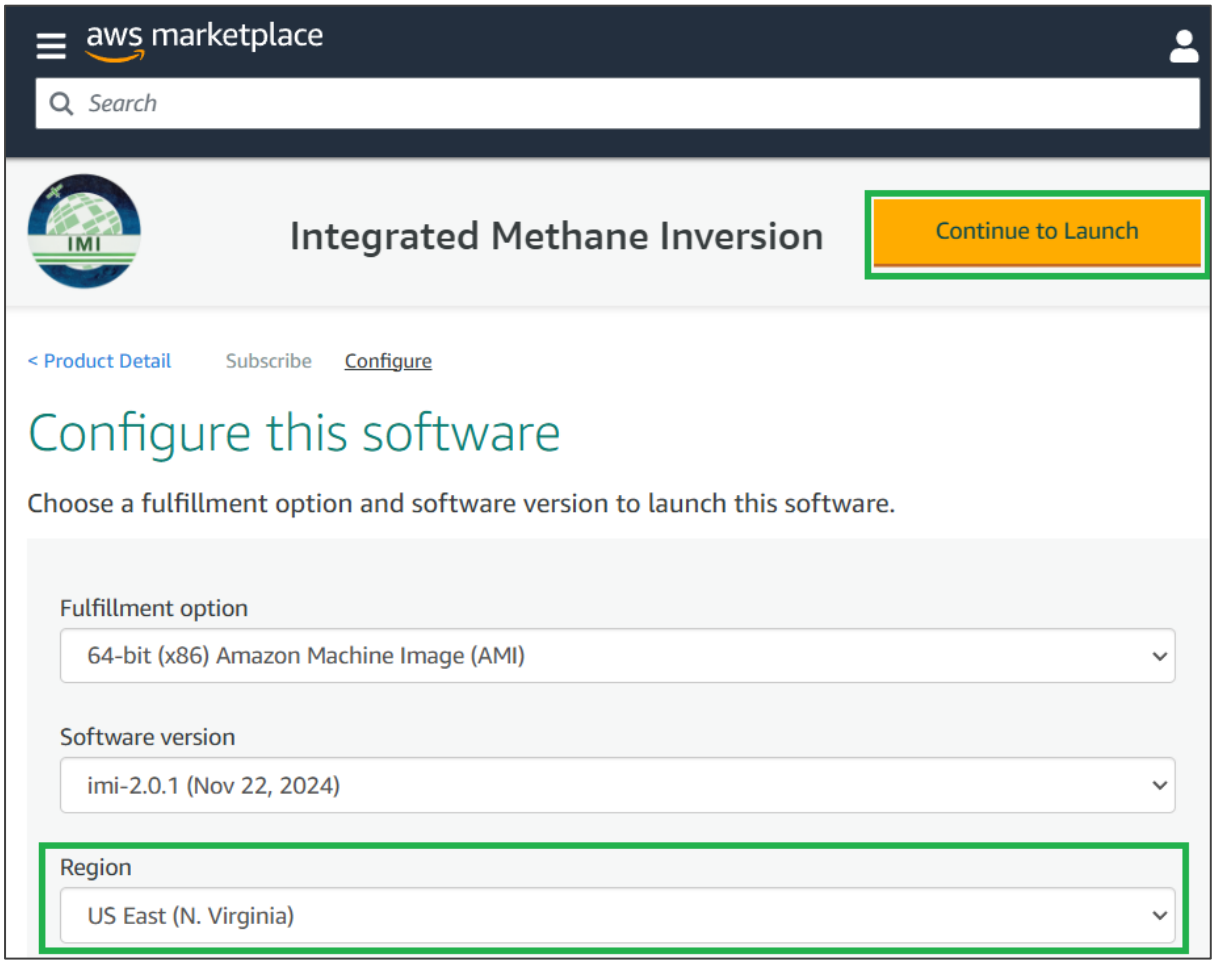


Figure 13: region configuration page and "continue to launch button".

In the "Launch this software" page, in the choose action prompt, select "Launch through EC2" and then click "Launch" (fig. 14).

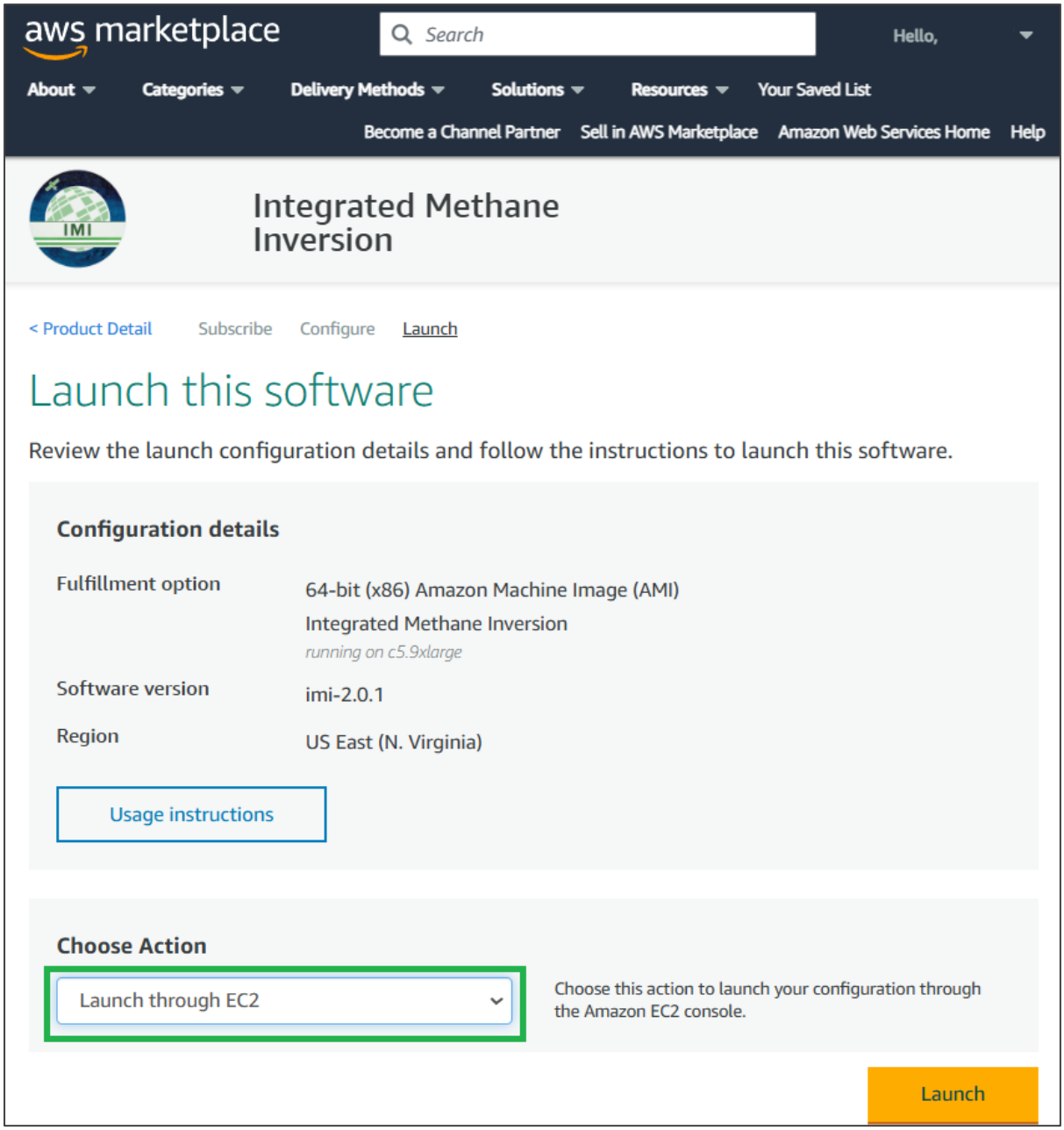


Figure 14: Launch software page.

Now it's time to determine the name of your instance and the hardware to run the IMI. The primary considerations are the number of CPUs and the amount of RAM. In the "Instance Type" section, you can choose from a wide range of options. The IMI will perform faster with a higher number of CPUs. The c5.9xlarge instance type is the recommended setting. It offers 36 CPU cores and 72GB of RAM. However, you can choose a different instance type with more or fewer cores and memory based on your specific needs (fig. 15).

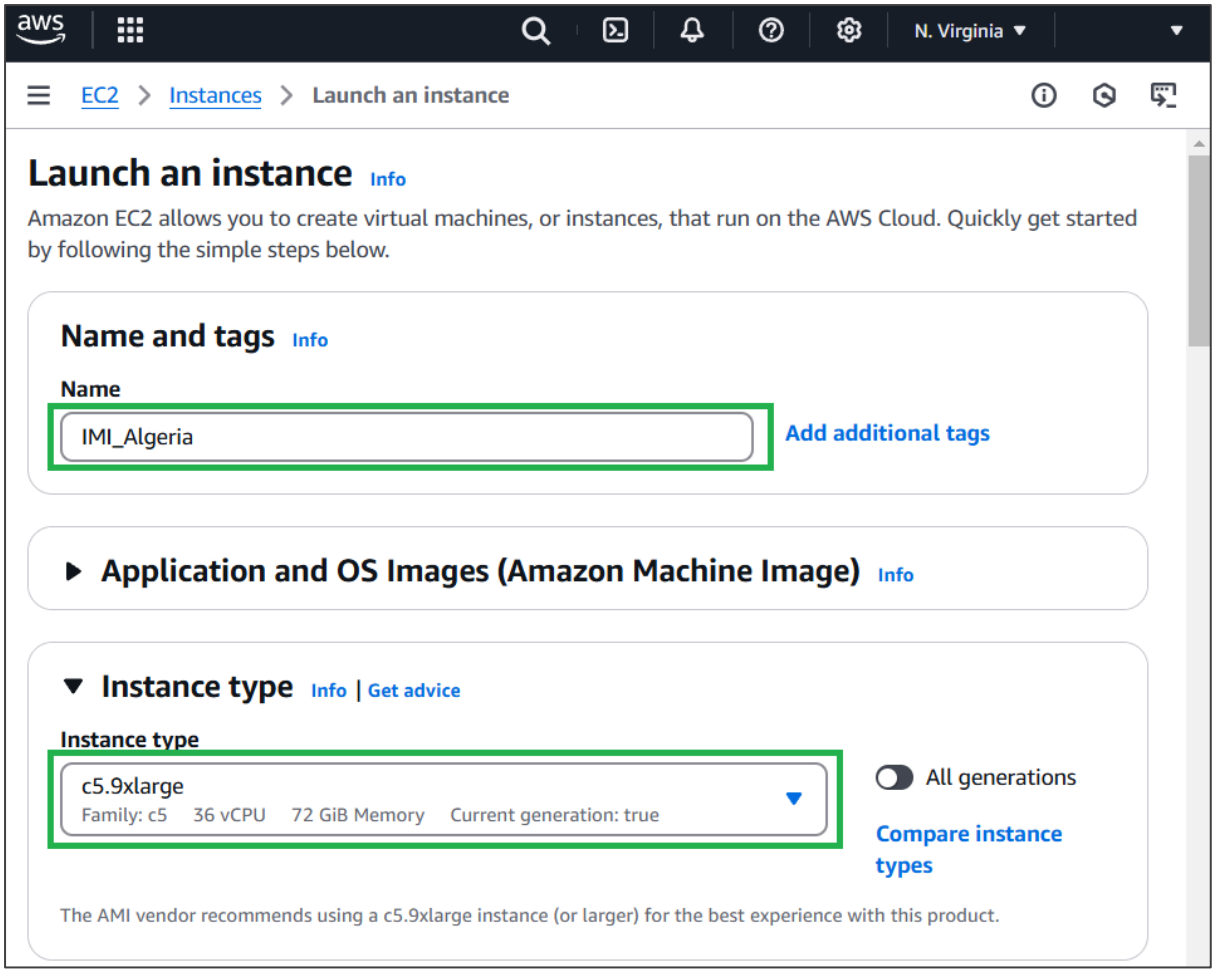


Figure 15: Hardware selection page.

In the next step, you'll create a new SSH key pair or select an existing one. This key pair acts as the password for accessing your server via SSH. To create a new key pair, click "Create new key pair". In the dialog box, assign a name to your key pair (e.g., imi_testing) and click "Create key pair". Once created, the key pair will be available for future use, and you can simply select it from the dropdown menu (fig. 16).

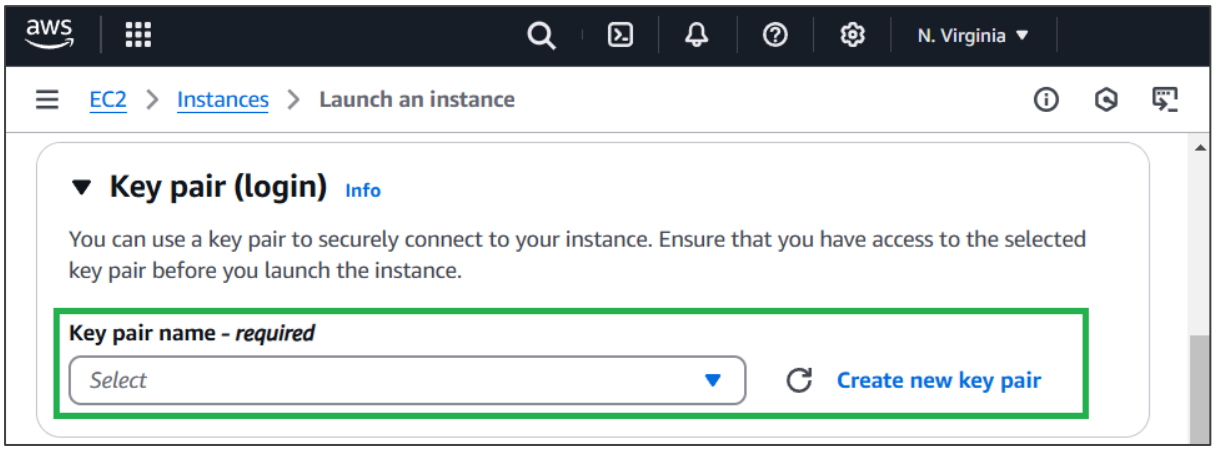


Figure 16: keypair selection.

If you are creating a key pair, use the settings shown in figure 17, choosing an appropriate name. Click "create key pair" once you have finished, saving it in a memorable directory that will be used later.

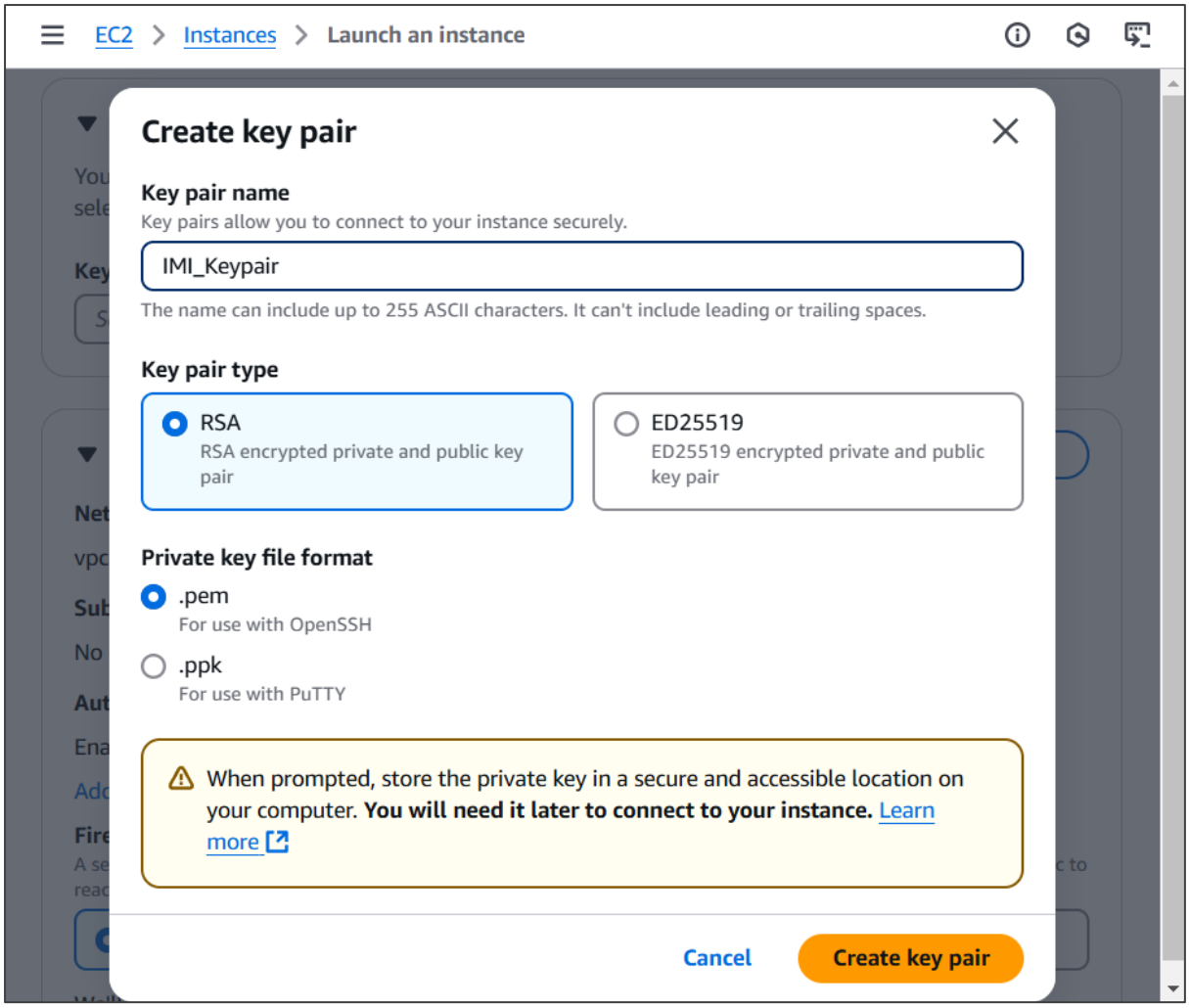


Figure 17: keypair setup.

Proceed to the "Configure Storage" section (fig.18), where you will select the size of the storage volume. Storage costs are around USD $100 per month per TB of space. The amount of storage you choose will depend on how long you intend to analyse an area. According to an example provided by the IMI team, 100 Gibibytes is sufficient for a 1-week inversion of an area of around 10,000km^2, whereas 5 terabytes is enough for 1 year. Also, 10,000km^2 is the minimum recommended area of analysis.

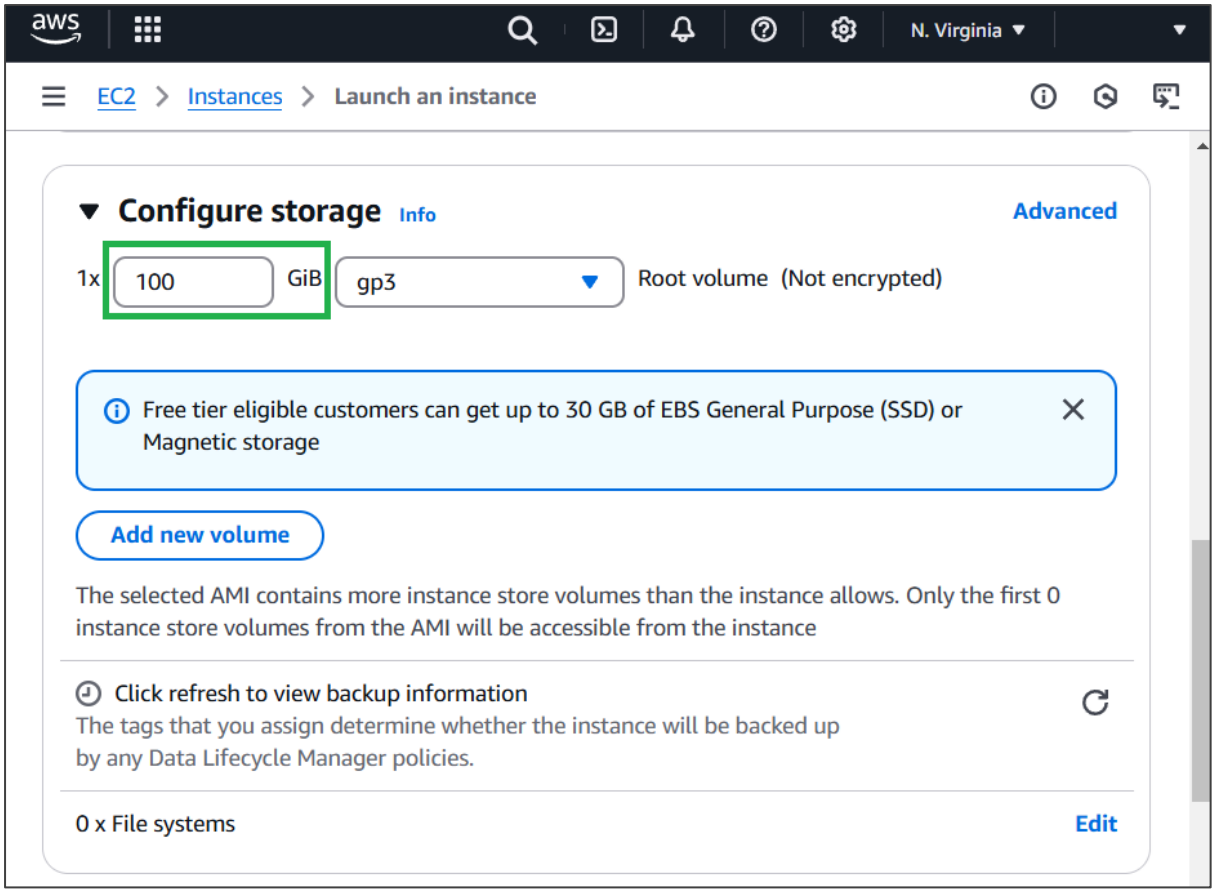


Figure 18: storage configuration section.

Next, open the "Advanced Details" section (fig. 19) and choose the IAM role you created in step 3 from the "IAM Instance Profile" dropdown. This will grant your EC2 instance access to S3 for uploading output data. You can leave all other configuration settings in the "Advanced Details" section as the default values.

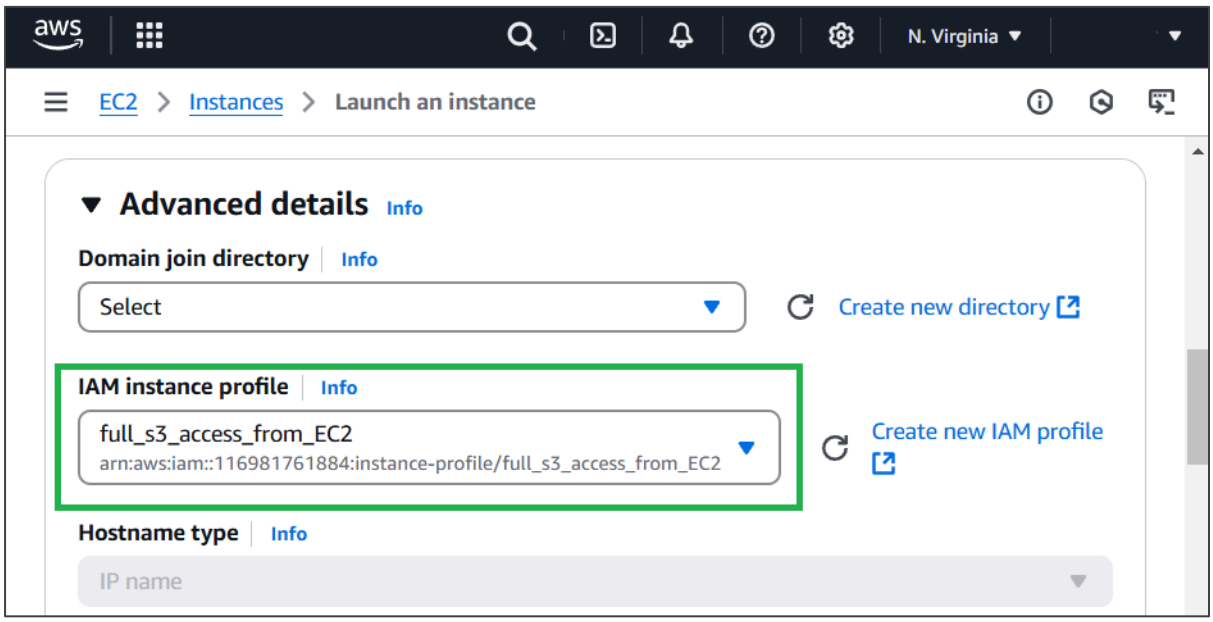


Figure 19: advanced details section.

Finally click the launch instance button in the section showed in figure 20.

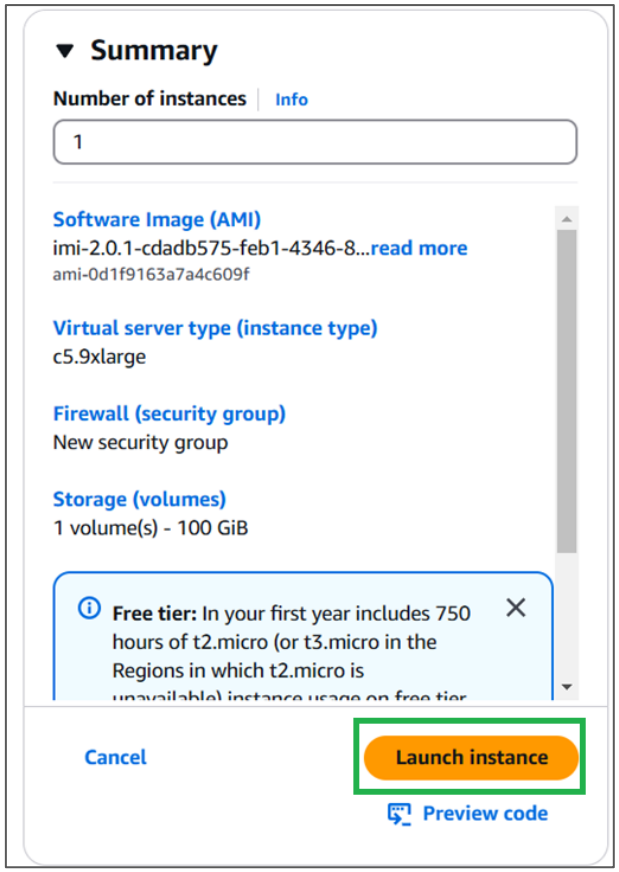


Figure 20: Summary section with "launch instance" button.

2.2.4 Connecting to instance

To connect to the instance and run the IMI, a program called Git will need to be downloaded and used. You can download Git from the following URL: https://git-scm.com/downloads/win, choosing the appropriate version of the software for your system (fig. 21).

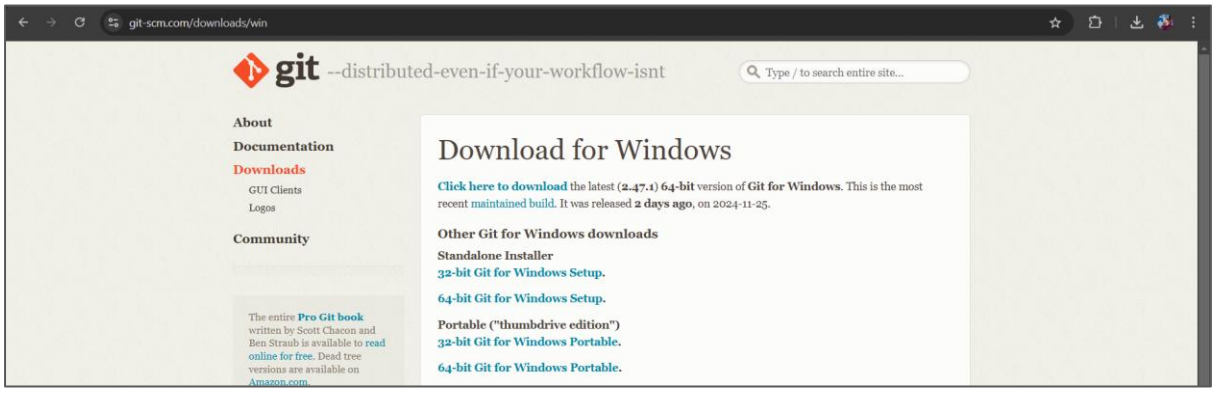


Figure 21: download page for Git.

Once downloaded, run and follow the instructions on the installation wizard (fig. 22).

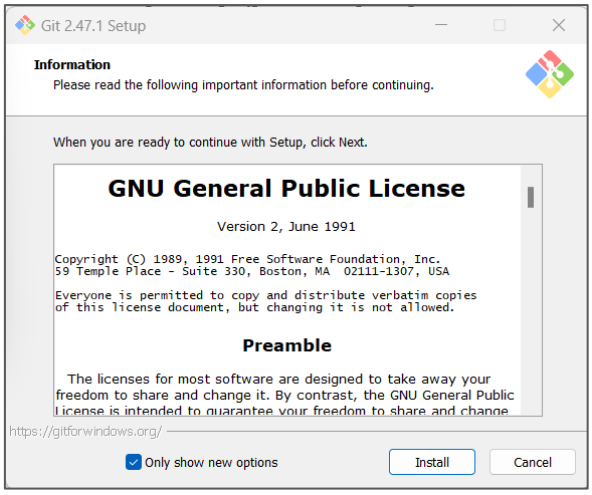


Figure 22: Git installation wizard.

Once completed, tick the box that says launch Git Bash (fig. 23). We will come back to it shortly.

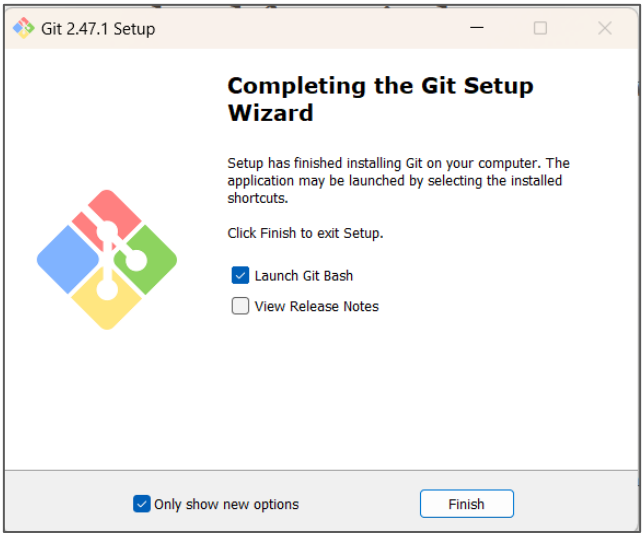


Figure 23: Git installation wizard.

Returning to the AWS management console, the first time you launch an instance it should be running already (fig. 24).

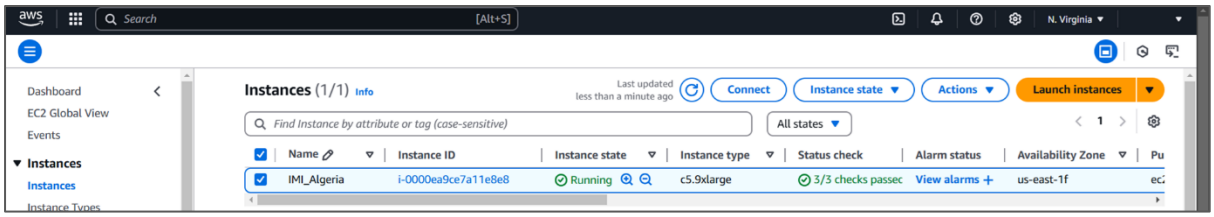


Figure 24: AWS management console Instance dashboard.

If it isn't running then select the instance using the tick box an then under "instance state" select "start instance" (fig. 25).

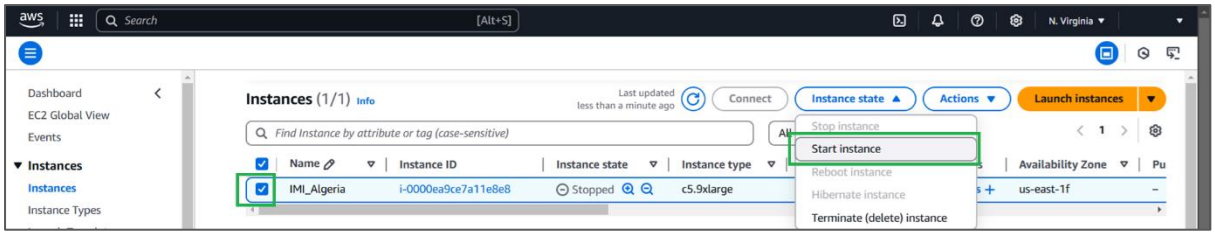


Figure 25: start instance button.

Once running, with the instance selected, select “actions” and then “connect” (fig. 26)

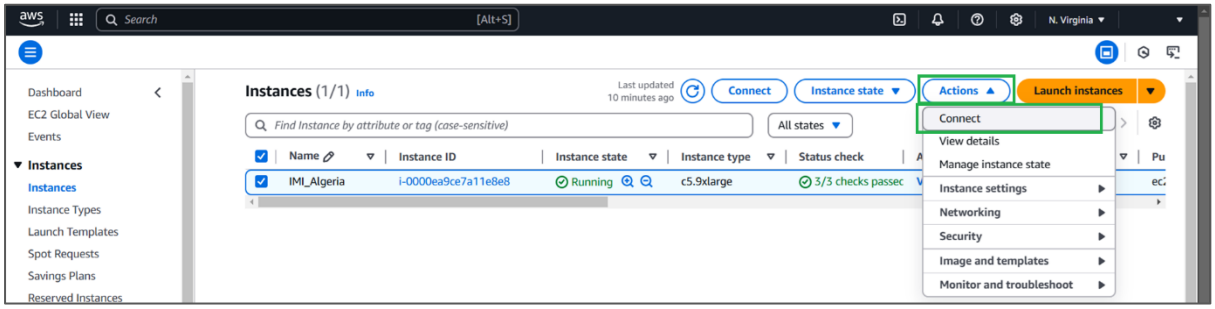


Figure 26: connect button.

In the connect to instance page, you have a list of instructions to follow using the Git Bash terminal you installed (fig. 27).

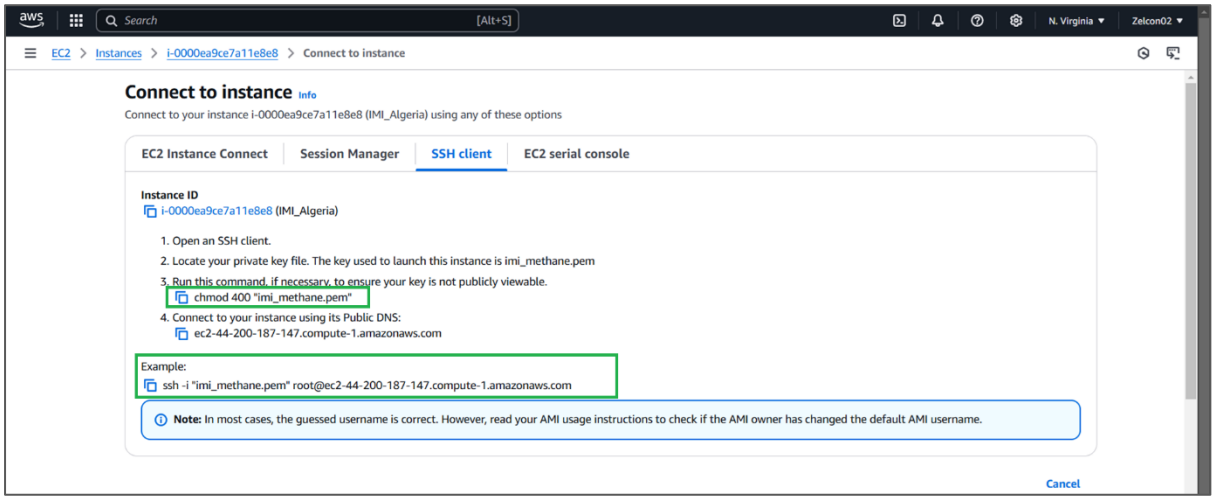


Figure 27: code to prevent keypair from being publicly visible and ssh connection command.

Firstly change directory to where you keep your keypair by typing cd <path to your keypair folder> (fig. 28)

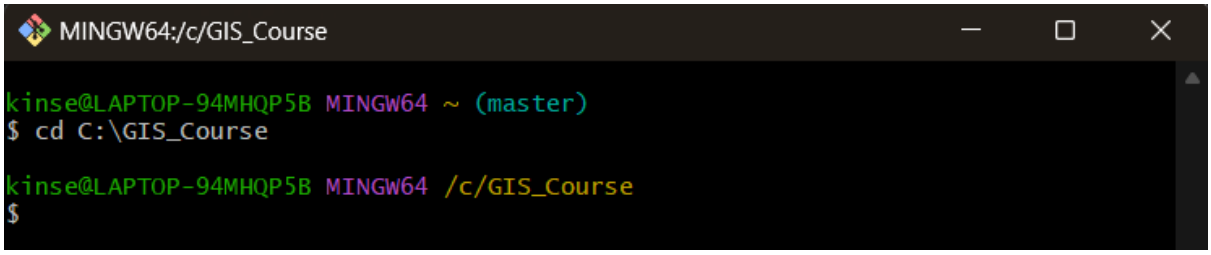


Figure 28: Gitbash console showing path to keypair folder.

Next you are going to enter in the command that ensures your keypair is not publicly viewable (fig. 29).

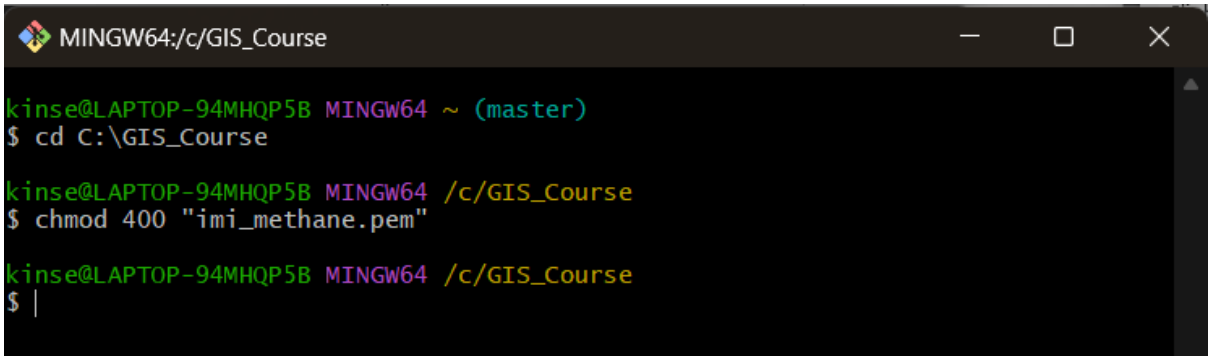


Figure 29: hiding your keypair.

Finally you are going to take the long "example" code (at the bottom of fig. 27) and change the word "root" to "ubuntu".

ssh -i "imi_methane.pem" root@ec2-44-200-187-147.compute-1.amazonaws.com

ssh -i "imi_methane.pem" ubuntu@ec2-44-200-187-147.compute-1.amazonaws.com

Then paste it into the Git Bash terminal using the right click function of the mouse. Run the code and answer "yes" to the prompt, "are you sure you want to continue connecting?" (fig. 30). Once this has run, you should receive a "welcome to ubuntu" message.

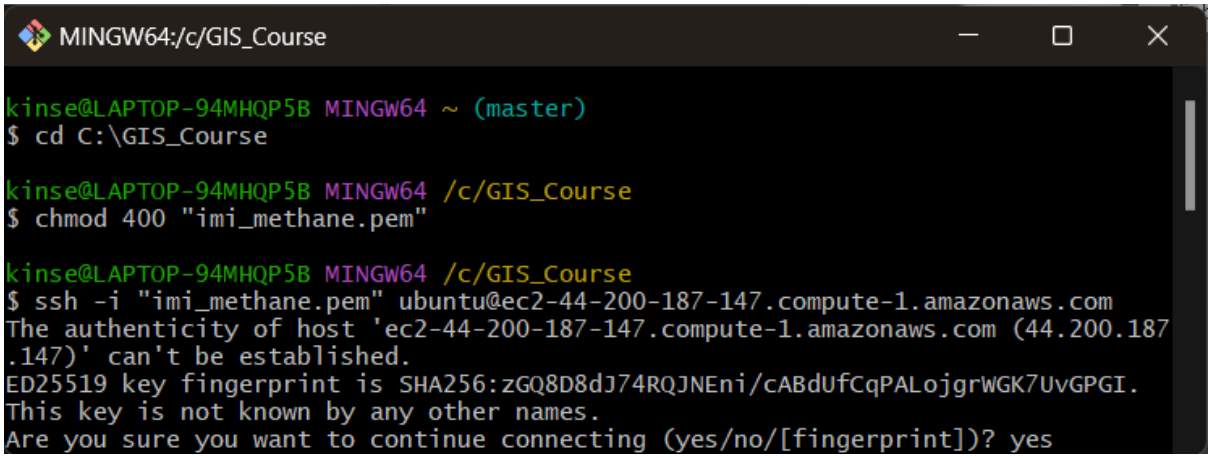


Figure 30: ssh connection to your instance.

2.2.5 Configuring and running the IMI Preview

In the Git Bash terminal, navigate to the IMI setup directory by typing:

cd ~/integrated_methane_inversion

Next open the inversion configuration file by typing the following:

emacs config.yml

The following screen should appear after a pause of a few moments (fig. 31)

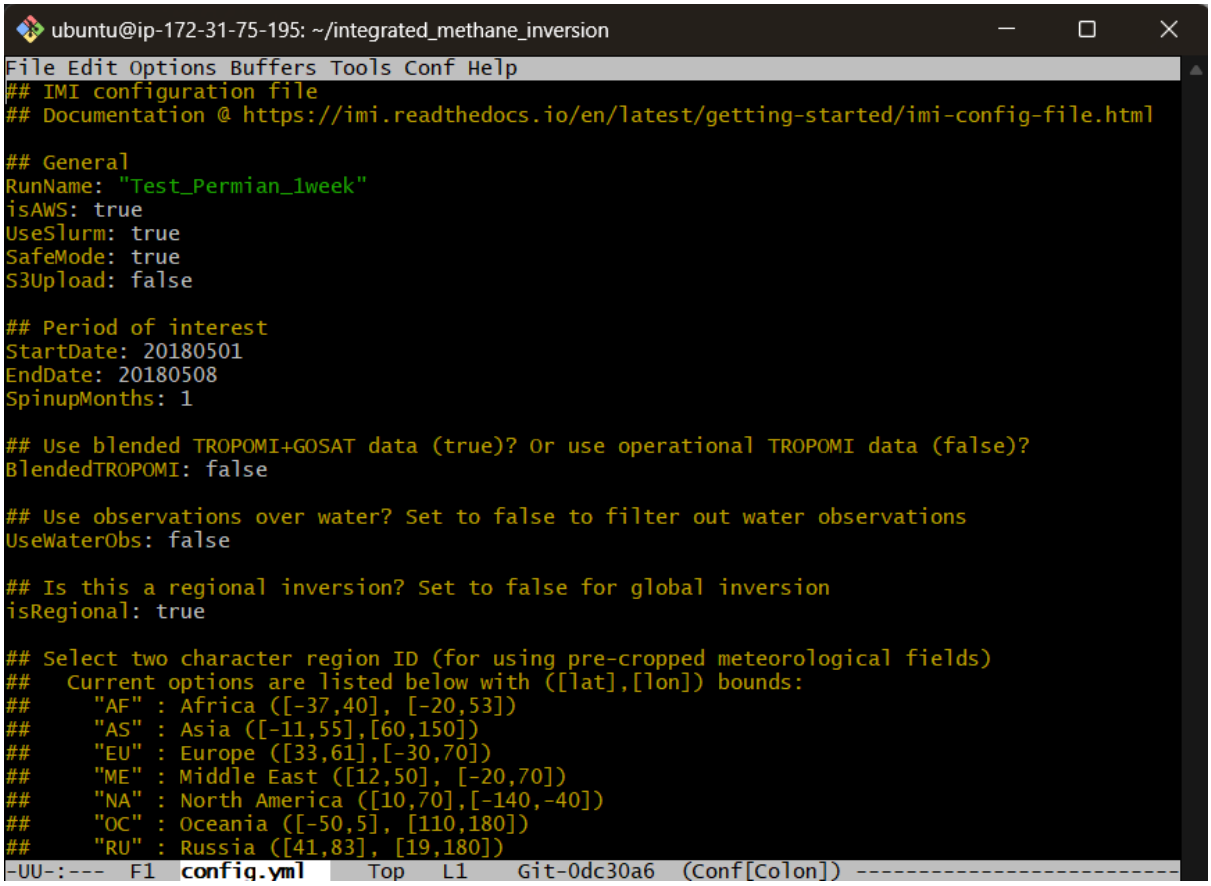


Figure 31: Beginning section of Config.yml

By default, this will run the IMI preview which will provide among other things, the expected cost of running the full inversion.

Make a note of the Runname. This must be unique for every inversion you run. Slightly further down, you can modify the date fields (format: YYYYMMDD) to choose the dates for the analysis (fig. 32).

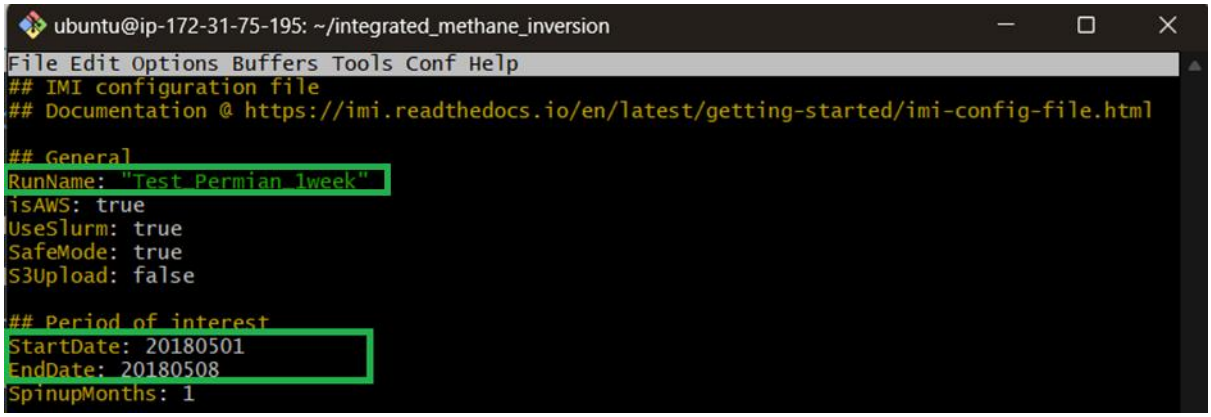


Figure 32: RunName field and start and end dates

Bounding box settings are found further down the page in the "region of interest" section (fig. 33). Specific boundings for each oil and gas field are given in figure 59 and table 1 on page 34 and 35 of this guide.

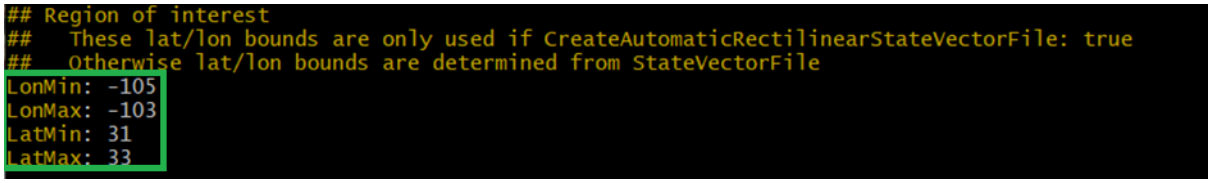


Figure 33: Analysis bounding box settings

Still further down is the IMI preview section which is by default set to "true". You will change this to "false" when deciding to go ahead with the inversion (fig. 34).

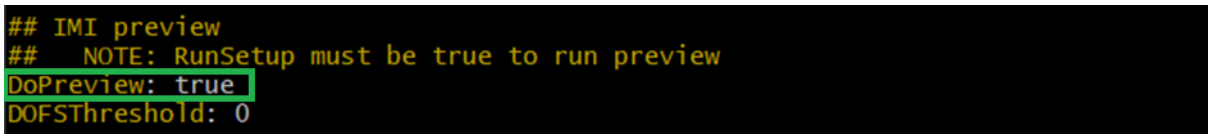


Figure 34: Config.yml DoPreview field set to true

Once you are satisfied with the settings, press F10 on your keyboard and using the arrow keys, navigate to "Exit", then press enter (fig. 35).

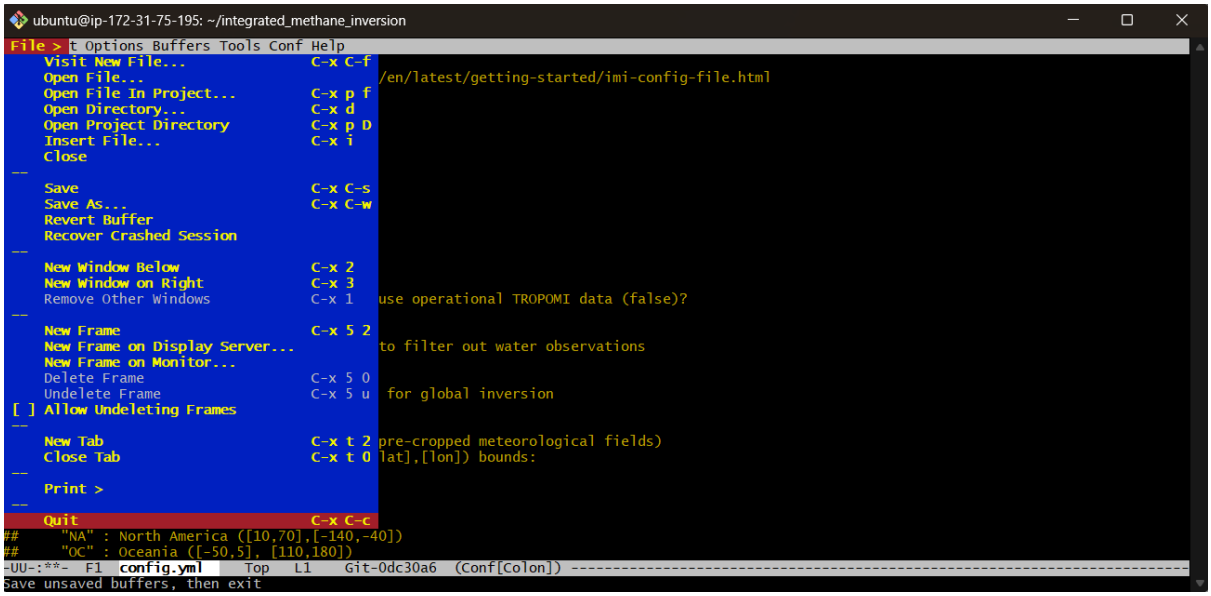


Figure 35: File menu in Config.yml

After exiting, run the following command to initiate the inversion preview:

```
sbatch run_imi.sh
```

and the following command to follow its progress:

```
tail --follow imi_output.log
```

Once the script has finished running it will look like figure 36 showing that the IMI preview for 1 week on the Permian Basin took around 15 minutes.

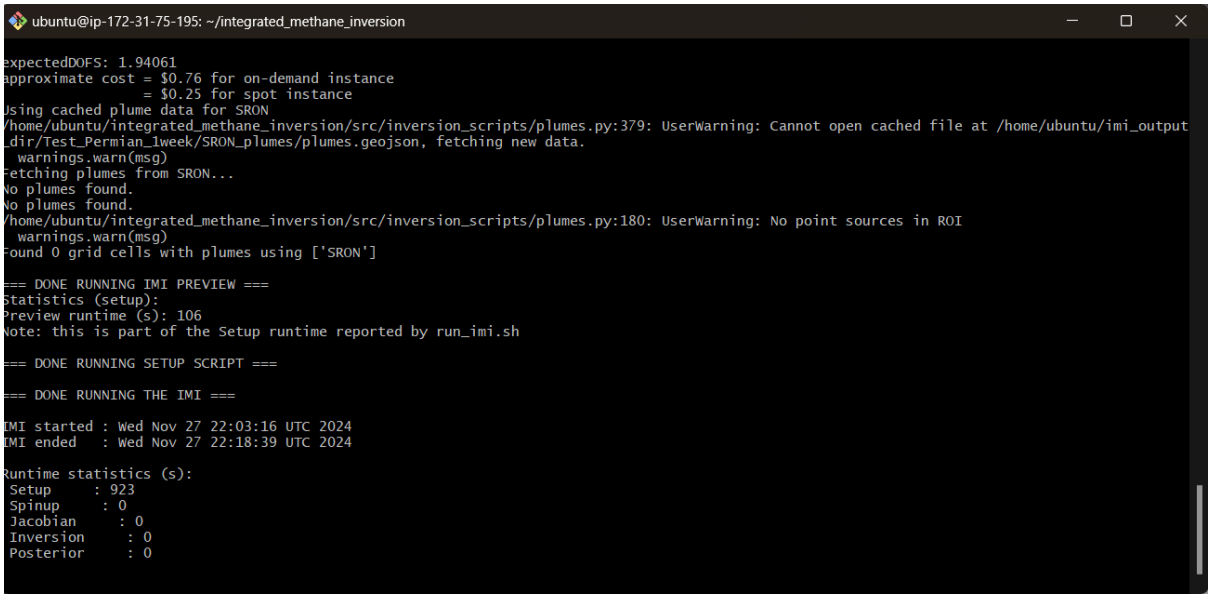


Figure 36: completed IMI Preview process.

2.2.6 Viewing the IMI Preview data

Open a new Git Bash terminal and log in as before (fig. 37)

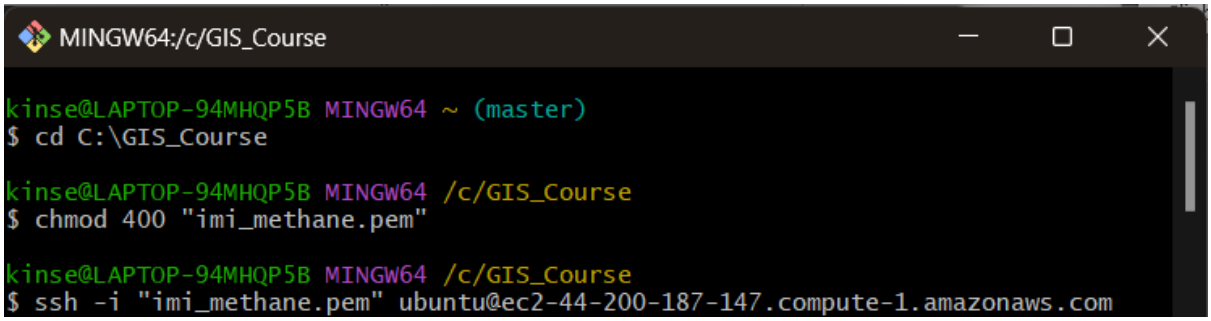


Figure 37: Login to instance

Once you have logged into the instance, enter the code:

jupyter notebook --no-browser --port 8080

This starts a Jupyter server on port 8080 and will print out two URLs with an authentication token. This as shown in figure 38.

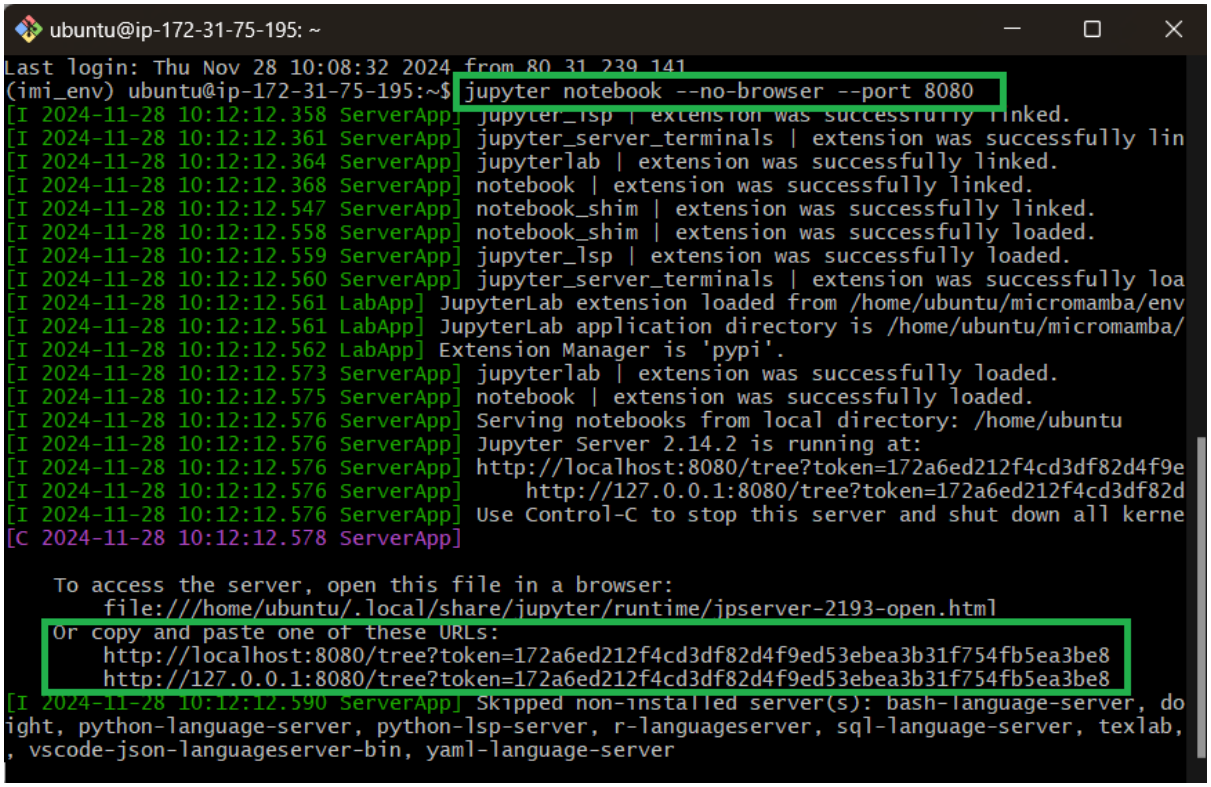


Figure 38: Jupyter server port on 8080 and URLs with authentication tokens.

Before you use these links you need to open an ssh tunnel from the ec2 instance to your local computer via port 8080. Open a new Git Bash terminal. Instead of logging into the instance as usual, you are going to use the following command to create the tunnel, changing the private key path to where your private key is stored on your local machine, and the host name to the same as the one shown in figure 39 on your connect to instance page.

Format: ssh -NL 8080:localhost:8080 -i /path/to/private_key ubuntu@<host-name>

Example: ssh -NL 8080:localhost:8080 -i "C:\GIS_Course\imi_methane.pem" ubuntu@ec2-3-238-122-254.compute-1.amazonaws.com

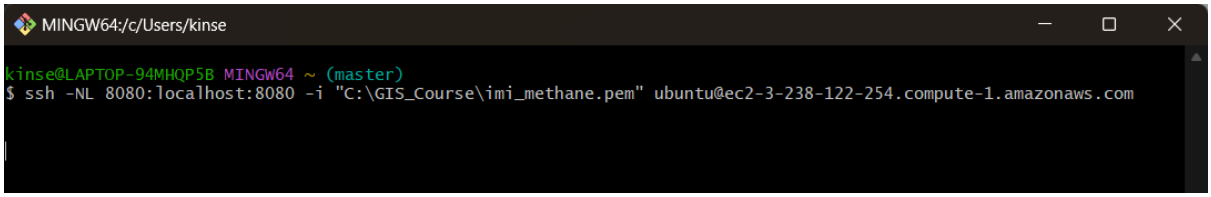


Figure 39: ssh tunnel command

Once you do this, nothing will seem to have happened in this Git Bash window but in the other window, commands will run to create the tunnel. You can now use one of the two URLs provided in figure 38 to access and view the IMI preview data (fig. 40).

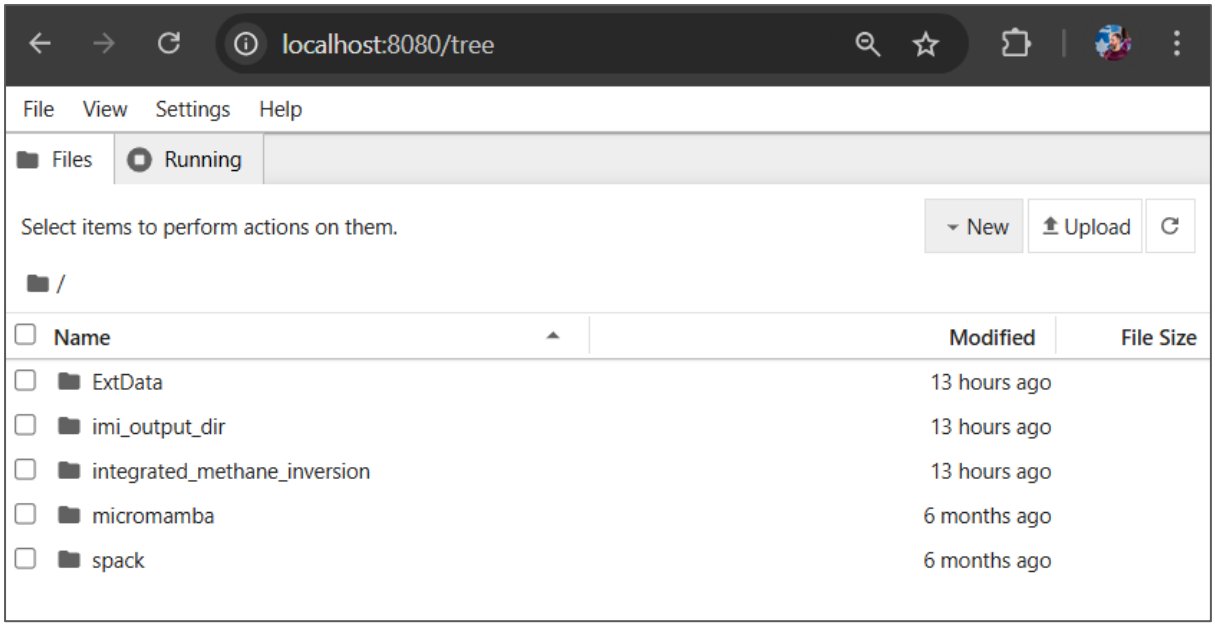


Figure 40: file directory of the IMI Preview

Once you have access to the file directory shown above, navigate to:

/imi_output_dir/Test_Permian_1week/preview/

From there you can download and view the IMI preview data (fig. 41)

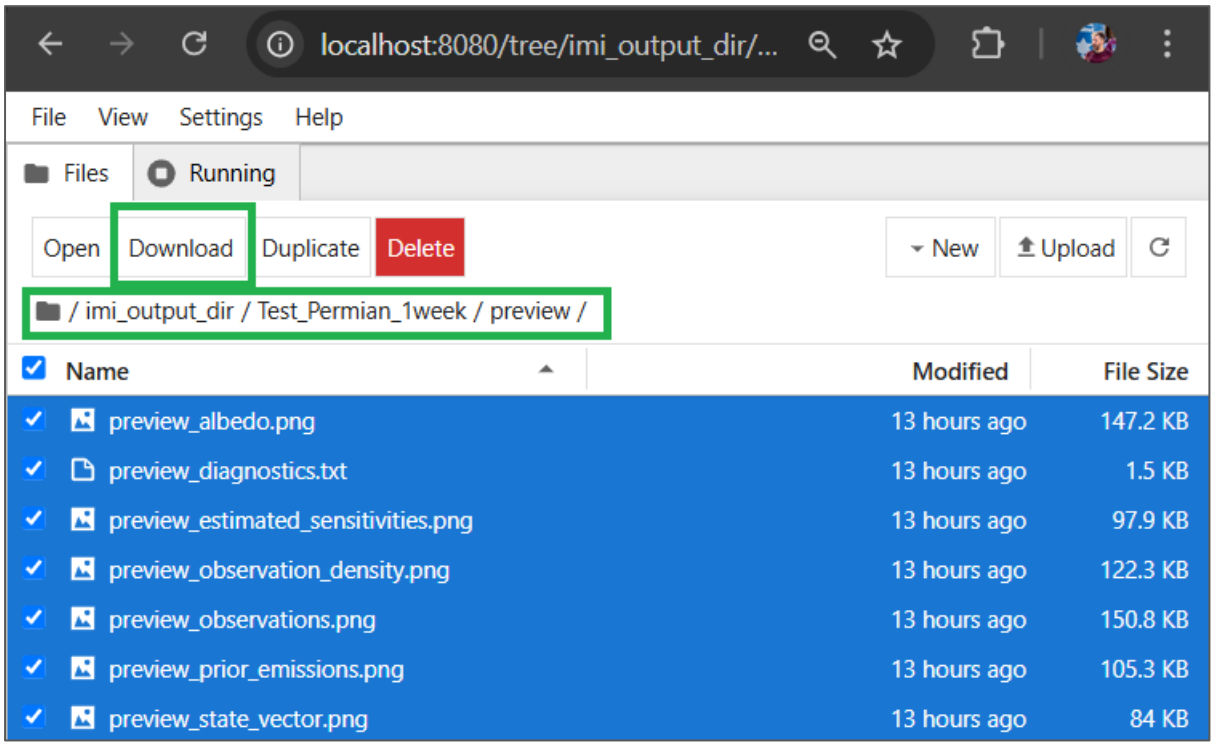


Figure 41: Location of IMI Preview data and download button

The IMI Preview will take approximately 15 minutes for a study size of 10,000km^2 of 1 week duration, with the total time scaling linearly.

2.2.7 Understanding the IMI Preview data

The IMI preview contains one .txt file and six maps, four of which are of interest. We will go through each of these in turn and explain which can help decide if you should proceed with the inversion. The

first two maps to examine are the visualizations of TROPOMI whole atmosphere data (preview_observations.png) and the official "prior" emission estimates (preview_prior_emissions.png). The spatial distribution between these two maps should become more corelated, the longer the period studied, as atmospheric concentrations average out across emission sources. A poor spatial correlation over longer periods may suggest that the "prior" inventory data does not accurately reflect the actual emissions (Fig. 42).

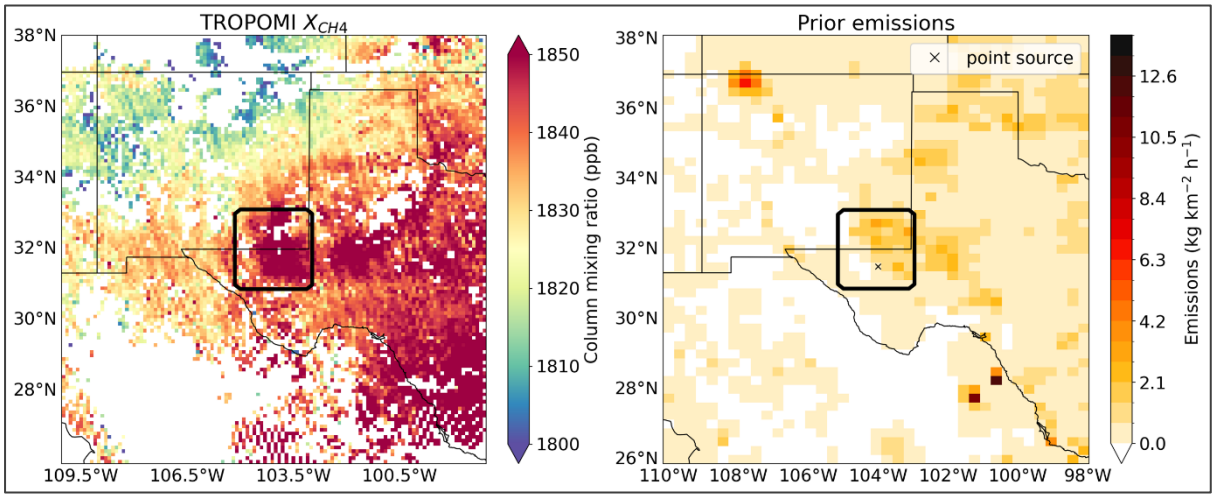


Figure 42: Mean TROPOMI Data for 1 week (left) and mapped "prior" inventory emissions (right)

The next map is of the observation density (preview_observation_density.png). In this example, the preview represents one week in May 2018, so the maximum observation density expected is 7 as Sentinel-5P has a daily return period (fig. 43). If there is significant cloud cover during the selected period, low observational density will be evident. In such cases, you may decide not to proceed with the inversion due to a lack of data.

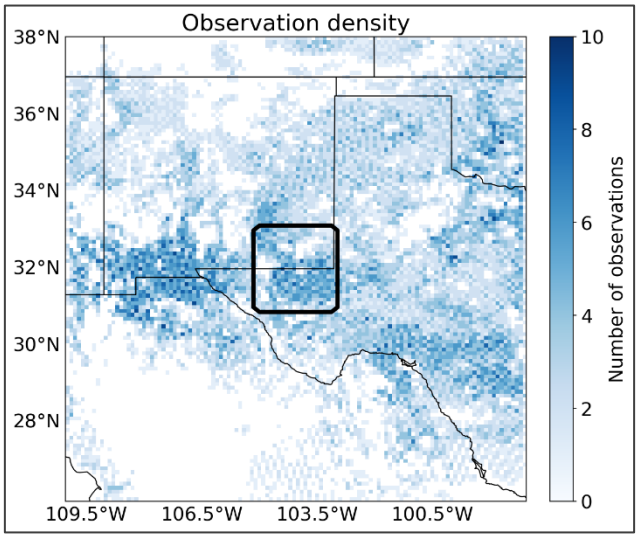


Figure 43: Observation density for 1 week

TROPOMI's measurements of surface albedo in the SWIR spectrum are visualized in the map labelled "preview_albedo.png" (fig. 44). Albedo is a critical factor in measuring gases like methane (CH_4) because variations in surface reflectivity can lead to inaccurate readings. The IMI method accounts for these differences, but areas with very low albedo may overestimate emissions, while areas with high albedo may underestimate them (Barré et al., 2021). This map does not say much about if one should go ahead

with an inversion or not, but examining it, users can identify potential false readings from the final IMI, particularly if the spatial patterns of albedo and methane retrievals appear similar.

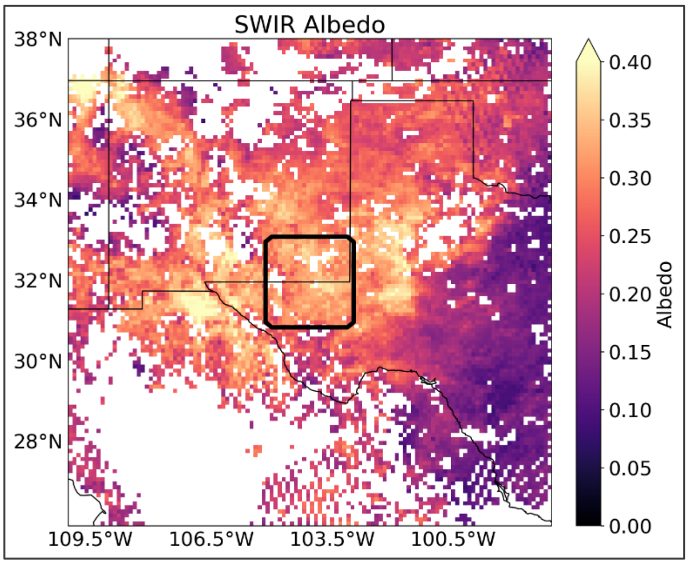
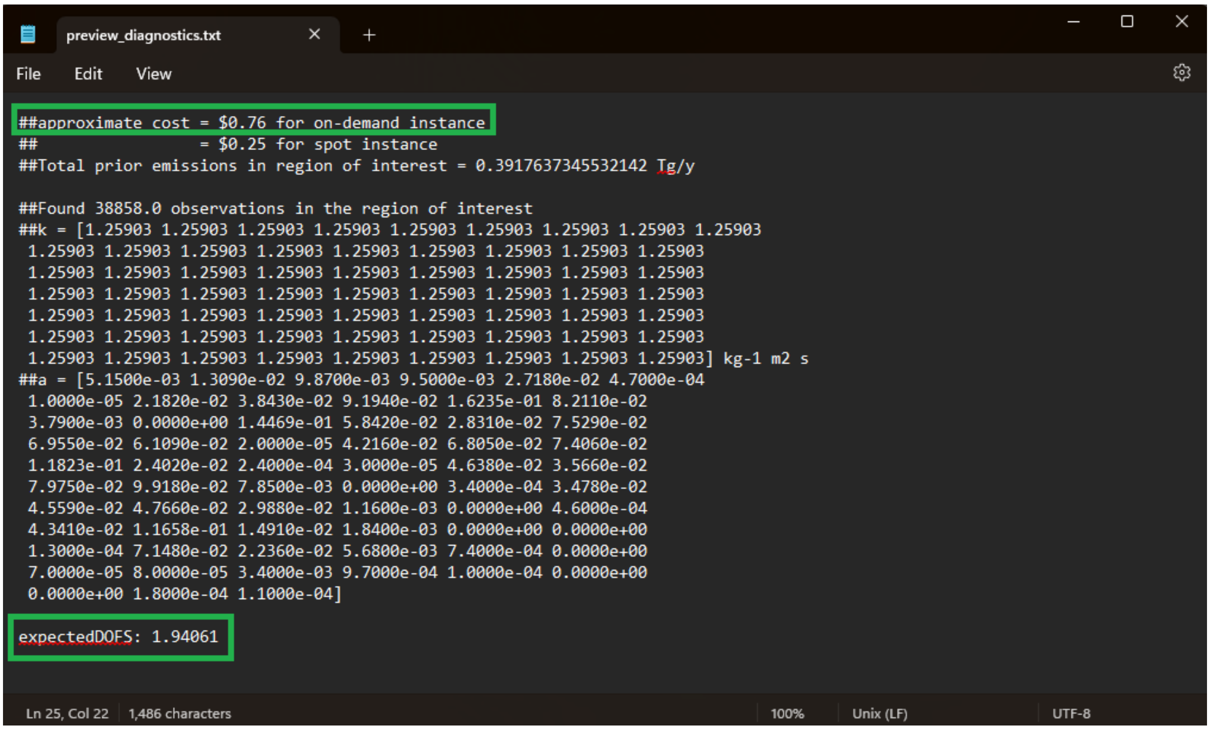


Figure 44: TROPOMI map of SWIR Albedo

The file "preview_diagnostics.txt" provides an estimate of the inversion's cost (shown in Figure 45) and a value called "expectedDOFS". DOFS, or Degrees of Freedom for Signal, measures how much the TROPOMI observations contribute to the final methane emission estimate compared to prior official inventory data. The DOFS value is influenced by the observation period; longer observation periods tend to increase DOFS, leading to more accurate results. A DOFS value of 1 is the minimum required to get meaningful emissions values, and higher values are preferred (Varon, et al. 2022). Shen et al. (2022) found that a DOFS greater than 2 was needed to reliably estimate emissions for oil and gas basins. If your DOFS is too low, you can extend the observation period and run the IMI preview again to improve the result.


preview_diagnostics.txt

File Edit View

```
##approximate cost = $0.76 for on-demand instance
##                  = $0.25 for spot instance
##Total prior emissions in region of interest = 0.3917637345532142 Tg/y

##Found 38858.0 observations in the region of interest
##k = [1.25903 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903
 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903
 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903
 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903
 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903
 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903
 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903 1.25903] kg-1 m2 s
##a = [5.1500e-03 1.3090e-02 9.8700e-03 9.5000e-03 2.7180e-02 4.7000e-04
 1.0000e-05 2.1820e-02 3.8430e-02 9.1940e-02 1.6235e-01 8.2110e-02
 3.7900e-03 0.0000e+00 1.4469e-01 5.8420e-02 2.8310e-02 7.5290e-02
 6.9550e-02 6.1090e-02 2.0000e-05 4.2160e-02 6.8050e-02 7.4060e-02
 1.1823e-01 2.4020e-02 2.4000e-04 3.0000e-05 4.6380e-02 3.5660e-02
 7.9750e-02 9.9180e-02 7.8500e-03 0.0000e+00 3.4000e-04 3.4780e-02
 4.5590e-02 4.7660e-02 2.9880e-02 1.1600e-03 0.0000e+00 4.6000e-04
 4.3410e-02 1.1658e-01 1.4910e-02 1.8400e-03 0.0000e+00 0.0000e+00
 1.3000e-04 7.1480e-02 2.2360e-02 5.6800e-03 7.4000e-04 0.0000e+00
 7.0000e-05 8.0000e-05 3.4000e-03 9.7000e-04 1.0000e-04 0.0000e+00
 0.0000e+00 1.8000e-04 1.1000e-04]

expectedDOFS: 1.94061
```

Ln 25, Col 22 | 1,486 characters | 100% | Unix (LF) | UTF-8

Figure 45: Preview_diagnostics.txt with estimated cost and DOES fields highlighted.

2.2.8 Configuring and running the IMI.

If the IMI preview looks correct and the cost is within your budget, you will now proceed with running the IMI. Open a new Git Bash terminal and connect to your instance as before (fig. 46).

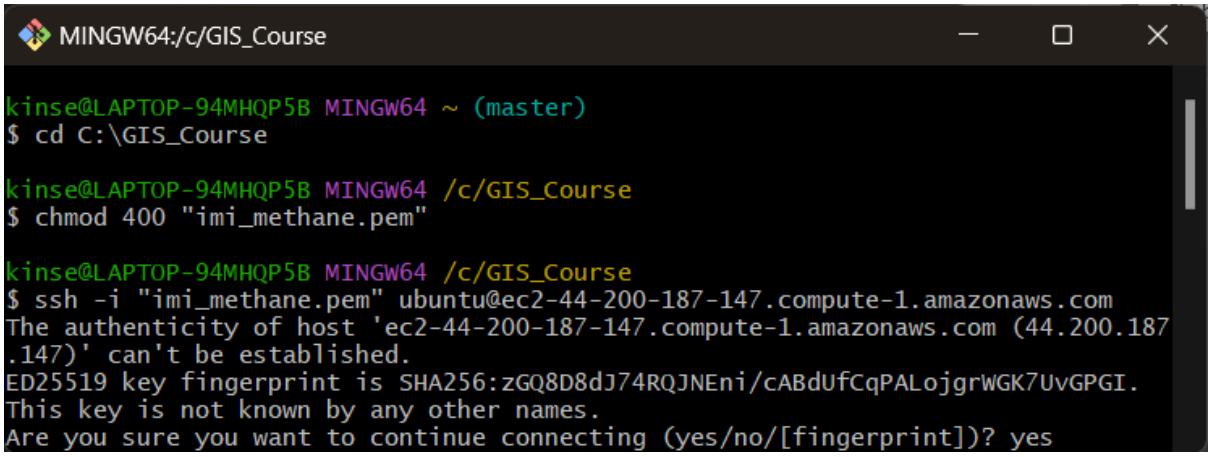


Figure 46: Connecting to instance.

In the Git Bash terminal, navigate to the IMI setup directory by typing:

cd ~/integrated_methane_inversion

Next open the inversion configuration file by typing the following:

emacs config.yml

The following screen should appear after a pause of a few moments (fig. 47)

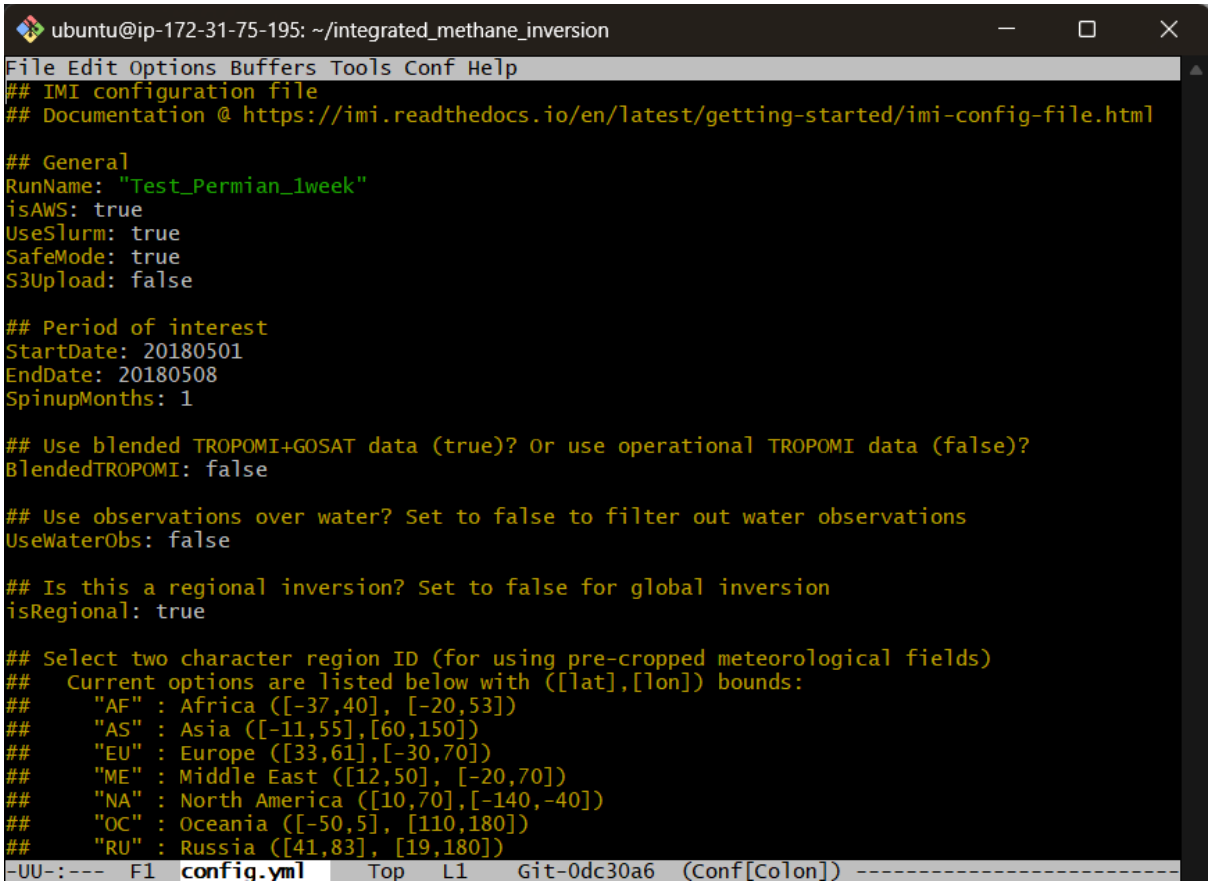


Figure 47: Config.yml section.

Before you changed the following fields to match the area and time you were interested in.

- StartDate
- EndDate
- LongMin
- LongMax
- LatMin
- LatMax

Now you are going to set the following fields to the values below to run the IMI instead of the IMI preview.

Setup modules

Turn on/off different steps in setting up the inversion

- RunSetup: true
- SetupTemplateRundir: false
- SetupSpinupRun: true
- SetupJacobianRuns: true
- SetupInversion: true
- SetupPosteriorRun: true

Run modules

Turn on/off different steps in performing the inversion

- DoHemcoPriorEmis: false
- DoSpinup: true
- DoJacobian: true
- ReDoJacobian: true
- DoInversion: true
- DoPosterior: true

IMI preview

NOTE: RunSetup must be true to run preview

- DoPreview: false

Once you have adjusted those settings, press F10 and save the changes (fig. 48), then press F10 again and Quit to return to the console.

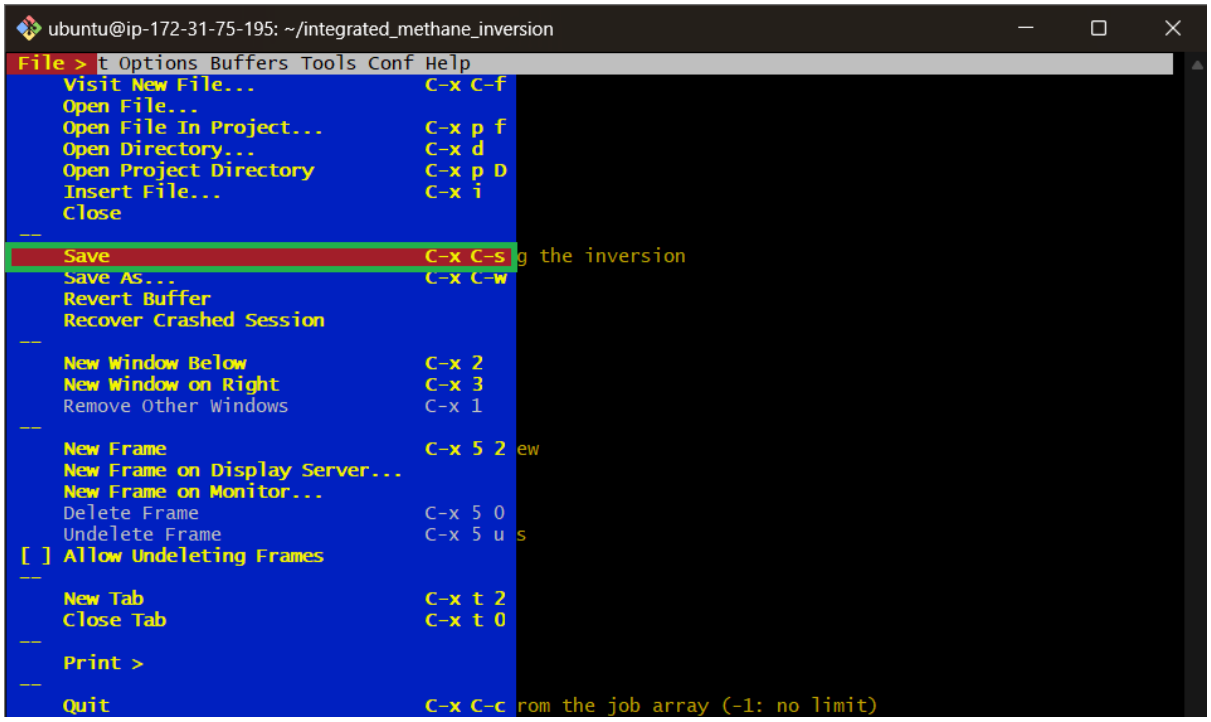


Figure 48: Location of the save button in the config.yml screen.

In the Git Bash terminal issue this command to run the IMI:

sbatch run_imi.sh

As before you can follow its progress by using the following command:

tail --follow imi_output.log

As shown in figure 49.

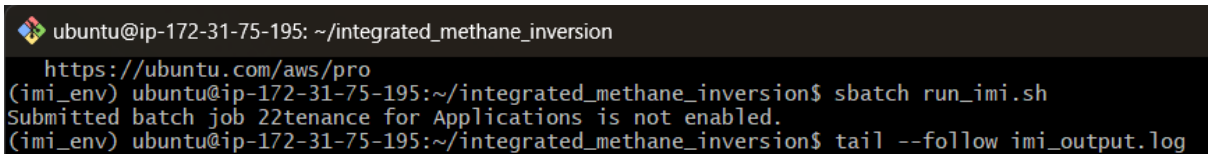


Figure 49: Commands running IMI and following progress.

At this point you can disconnect from your instance. The IMI will take approximately 2 and a half hours to run for a study size of 10,000km^2 of 1 week duration, with the total time scaling linearly.

2.2.9 Accessing the IMI data

Open a new Git Bash terminal and log in to your instance (fig. 50)

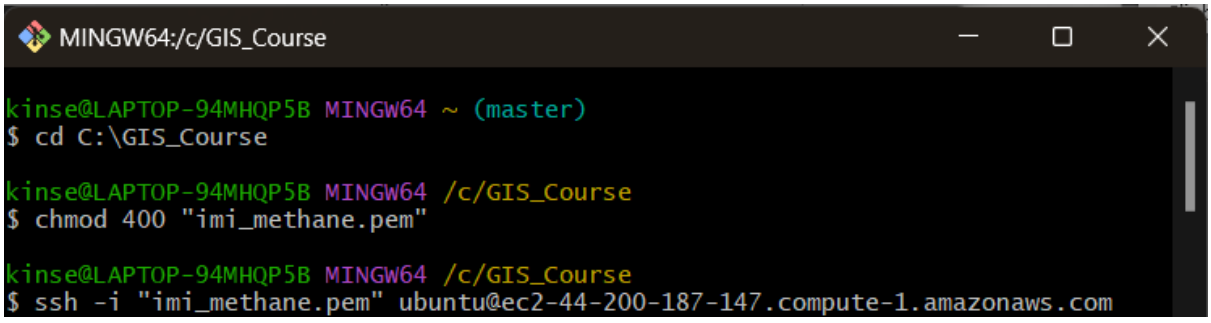


Figure 50: Connecting to instance.

Once you have logged into the instance, enter the code:

jupyter notebook --no-browser --port 8080

This starts a Jupyter server on port 8080 and will print out two URLs with an authentication token. This as shown in figure 51.

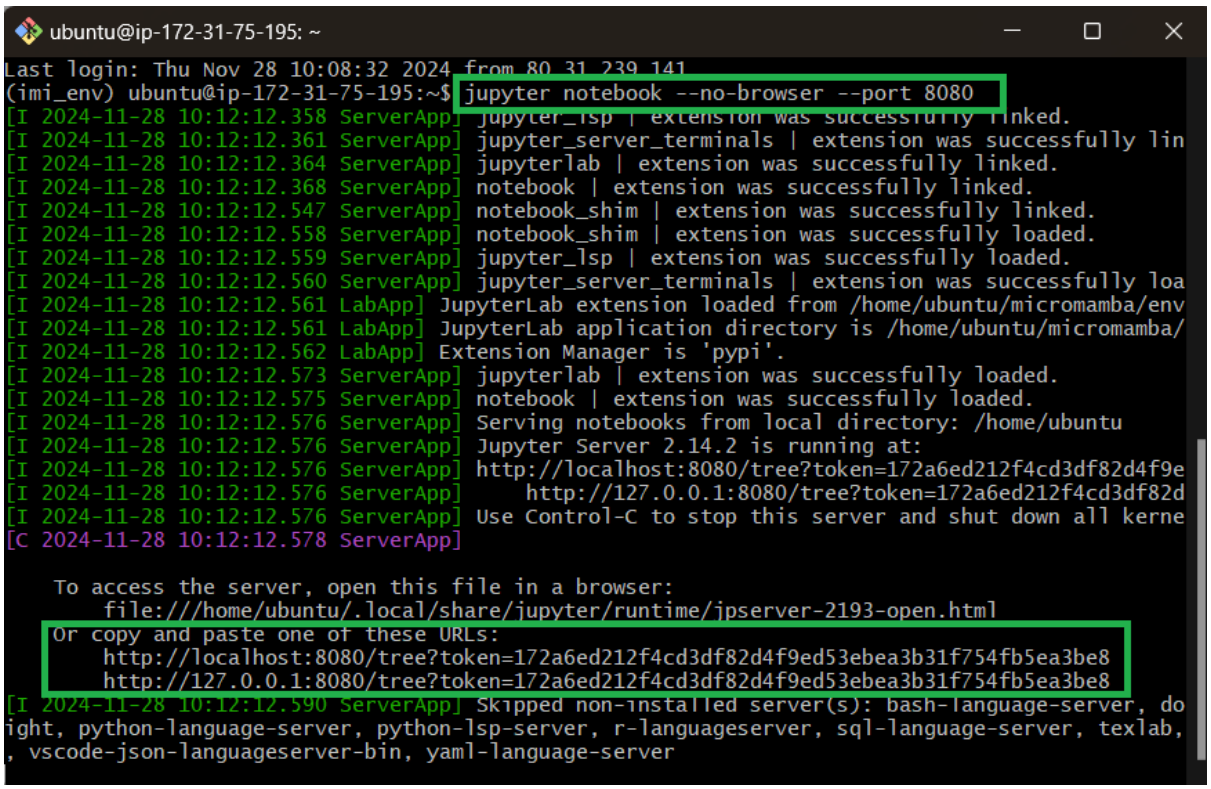


Figure 51: Jupyter server port on 8080 and URLs with authentication tokens.

Before you use these links you need to open an ssh tunnel from the ec2 instance to your local computer via port 8080. Open a new Git Bash terminal. Instead of logging into the instance as usual, you are going to use the following command to create the tunnel, changing the private key path to where your private key is stored on your local machine, and the host name to the same as the one shown in figure 27 on your connect to instance page (fig. 52).

Format: ssh -NL 8080:localhost:8080 -i /path/to/private_key ubuntu@<host-name>

Example: ssh -NL 8080:localhost:8080 -i "C:\GIS_Course\imi_methane.pem" ubuntu@ec2-3-238-122-254.compute-1.amazonaws.com

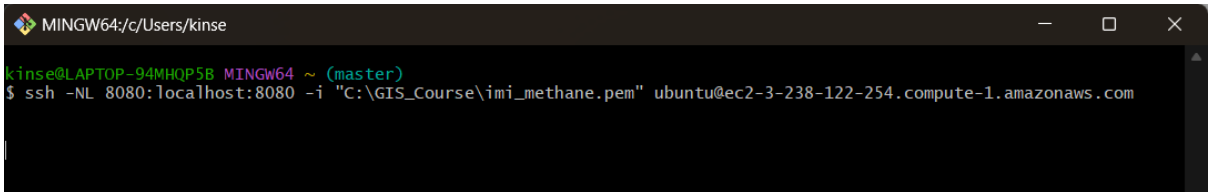


Figure 52: ssh tunnel connection

Once you do this, nothing will seem to have happened in this Git Bash window but there but in the background the tunnel will have been created. You can now use one of the two URLs provided in figure 61 to access and view the IMI preview data (fig. 53).

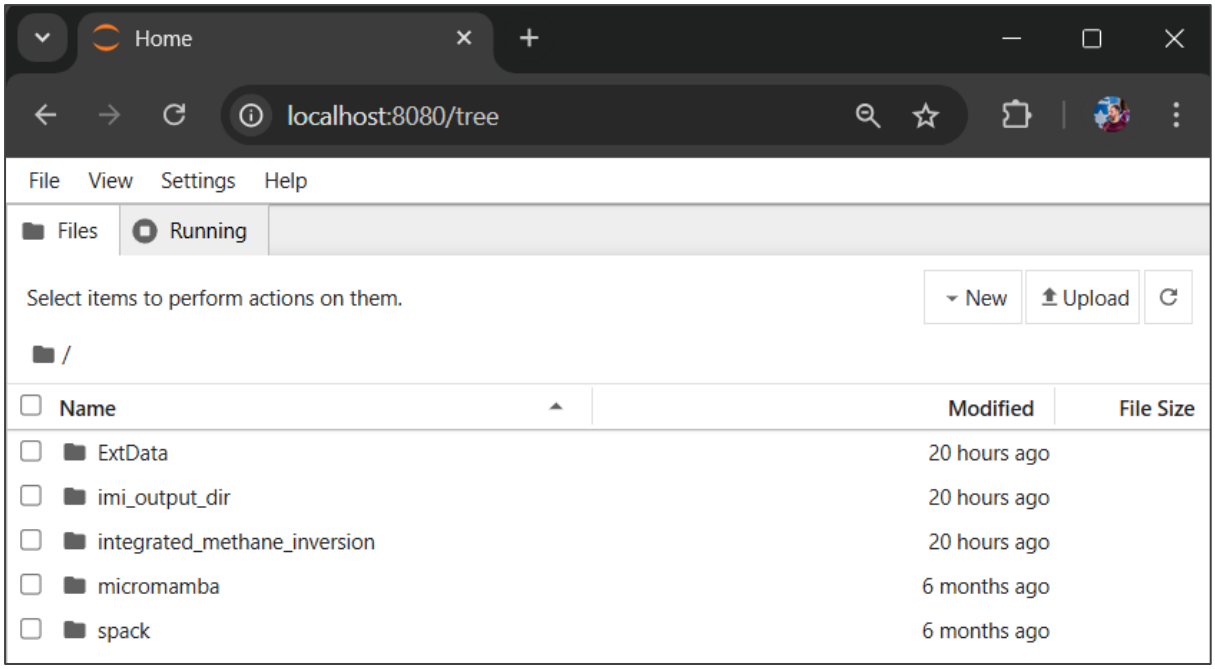


Figure 53: IMI file directory

Navigate to:

/imi_output_dir/Test_Permian_1week/inversion/

and look for "visualization_notebook.ipynb". Open this and a Jupyter notebook will appear. In this notebook, select from the menu "Run" and then "Run all cells" (fig. 54).

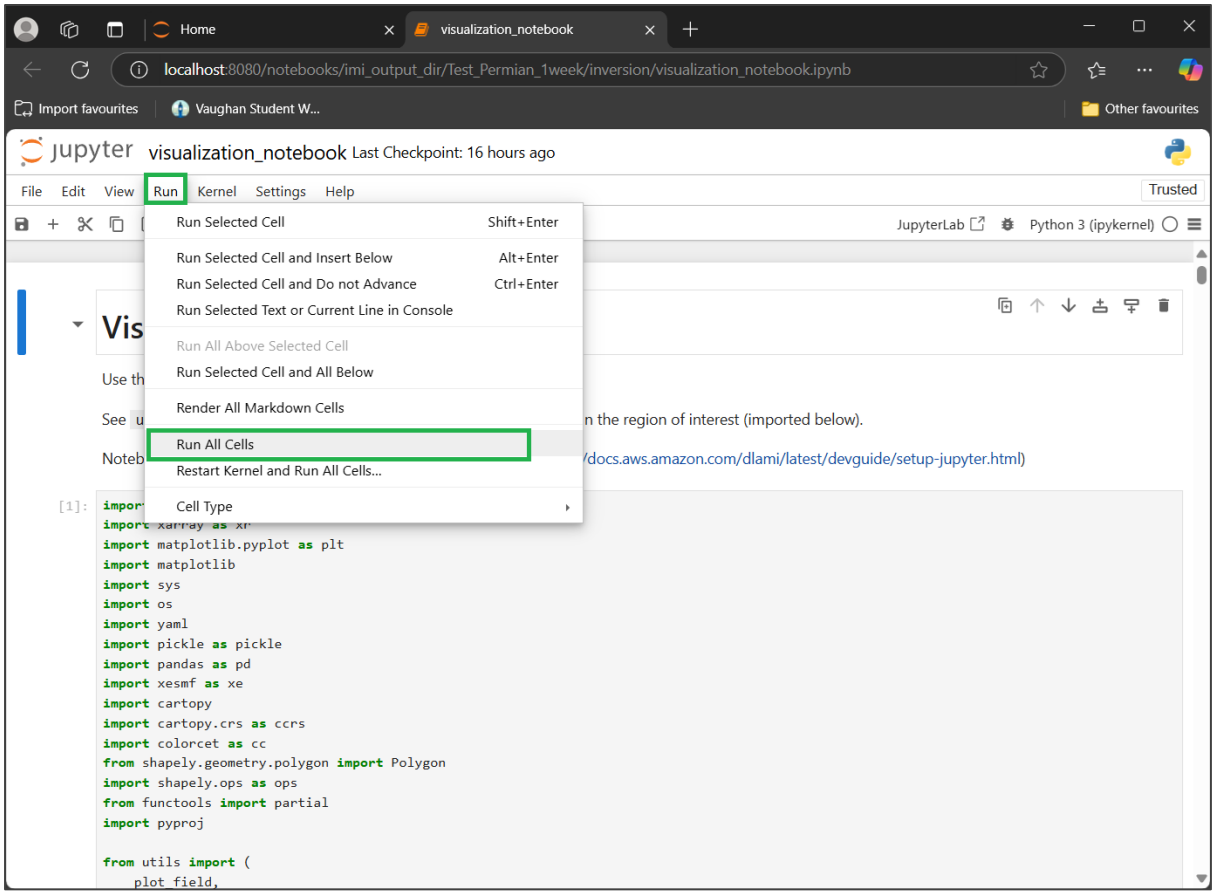


Figure 54: Run all cells command

2.2.10 Interpreting the IMI data

Once this is complete, scroll down to view the data. The first point of interest is code box 8 which shows the previous estimate for emissions based on official inventories (Prior) and the measured emissions based on Sentinel-5P and IMI (Posterior). As can be seen in figure 55, the official "prior" estimate is 0.079 Teragrams per year (Tg/y) below the IMI measured figure, which is 79,000 metric tonnes.

Figure 55: Official "prior" inventory emissions and Measured "posterior" emission totals.

Code box 9 further visualises this and shows the official "prior" estimates on a map, whereas code box 10 shows the mapped IMI measured "posterior" emissions (fig. 56).

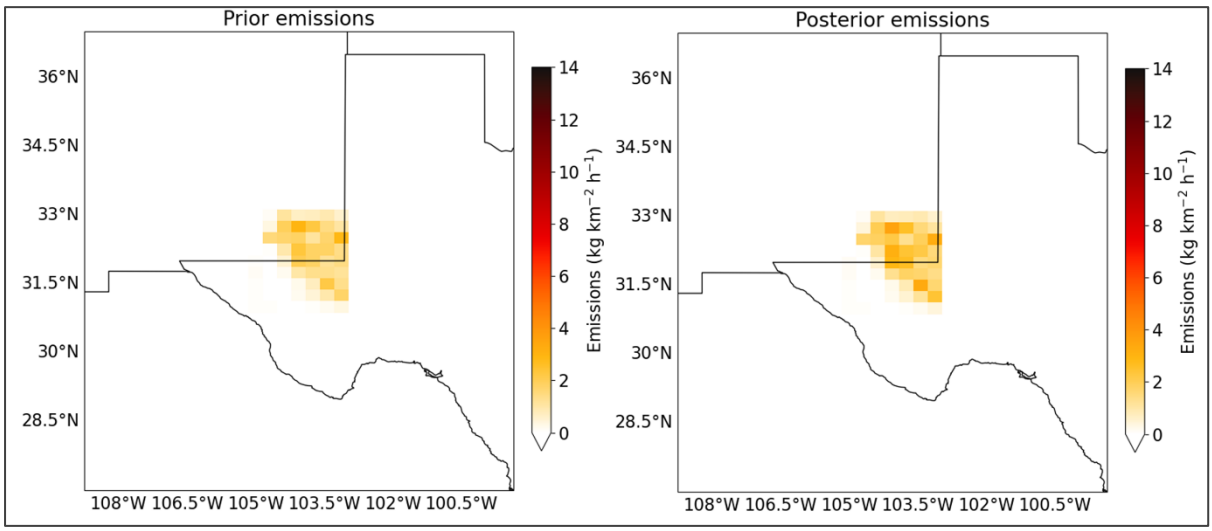


Figure 56: Mapped official "prior" inventory emissions and mapped "posterior" emissions

Code box 11 shows the sectoral emissions. This is determined by applying the ratios given by the official "prior" estimates to the IMI results (fig. 57).

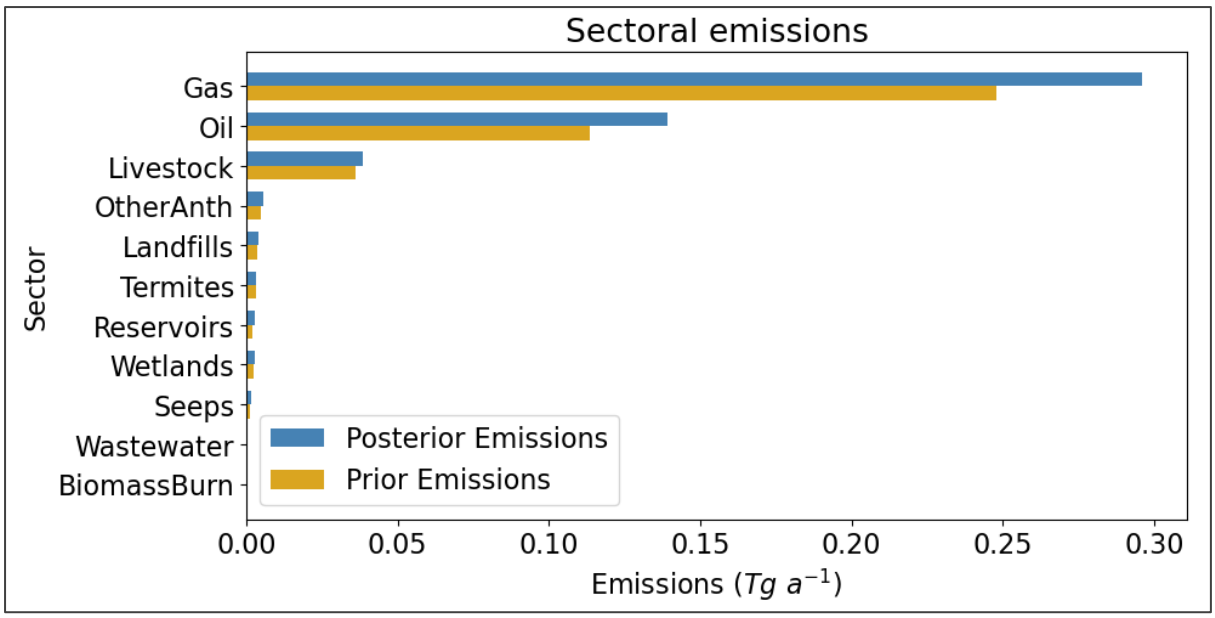


Figure 57: Emissions as divided by sector

If you wish to download any of the figures from the visualisation notebook, you can do so by navigating to:

/imi_output_dir/Test_Permian_1week/inversion/output/

Selecting the tick boxes for the figures and then clicking the download button (fig. 58)

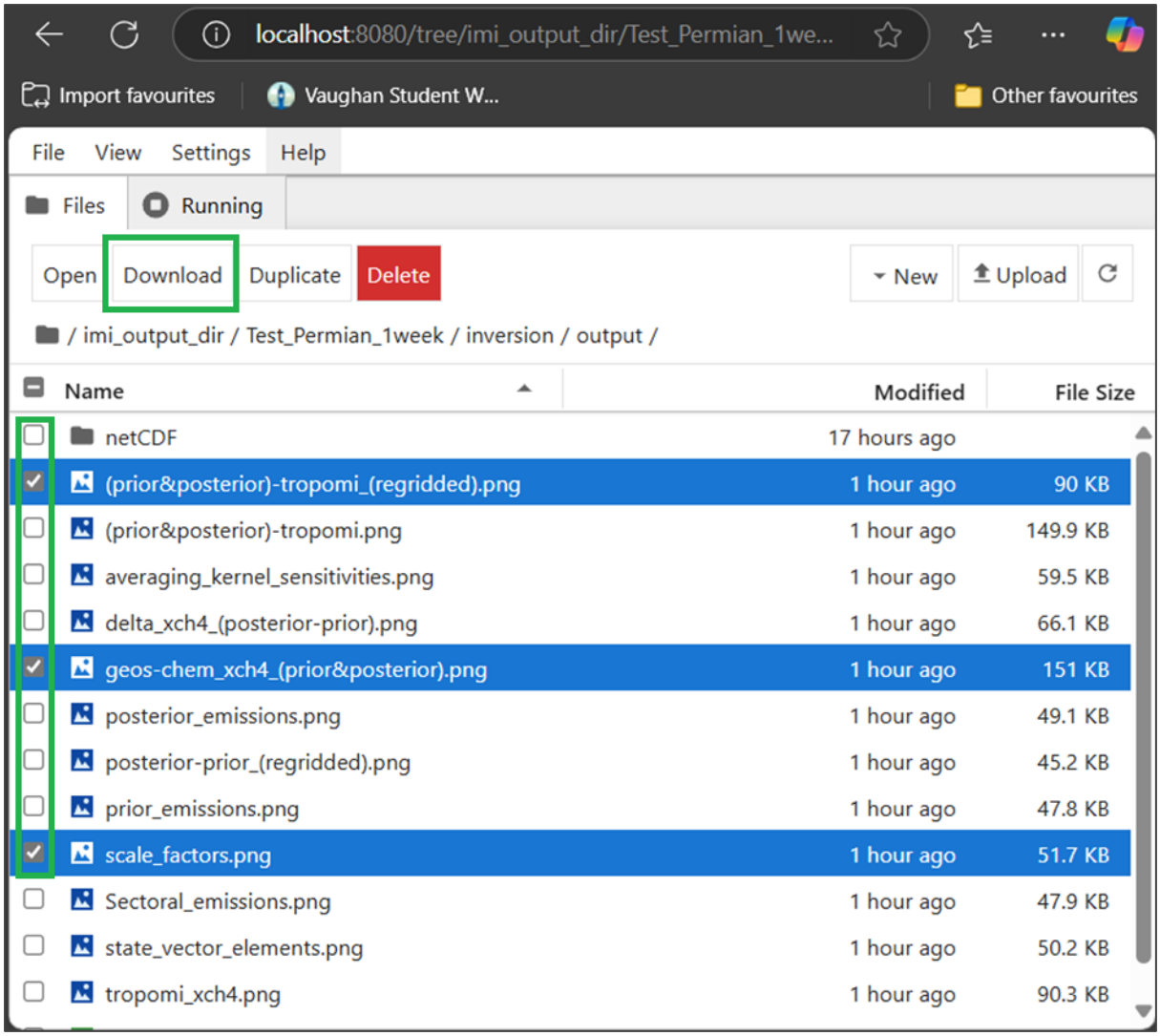


Figure 58:Location of check boxes and downloads.

2.2.11 Oil and gas field bounding parameters

101 fields were mapped in this project's survey. This was done by georeferencing the map in Abada and Bouharkat (2018) and then manually surveying the Imagery basemap in ArcGIS Pro. It is therefore, not perfect, and should ideally be replaced by official data when possible.

On the following page figure 59 shows a map of numbered fields which each correspond to a particular bounding box of around 10,000km around that field. Once you have found the field that you are interested in monitoring, please consult table 1 for its specific bounding coordinates, which are taken as 1 degree east, west, north and south of its centroid.

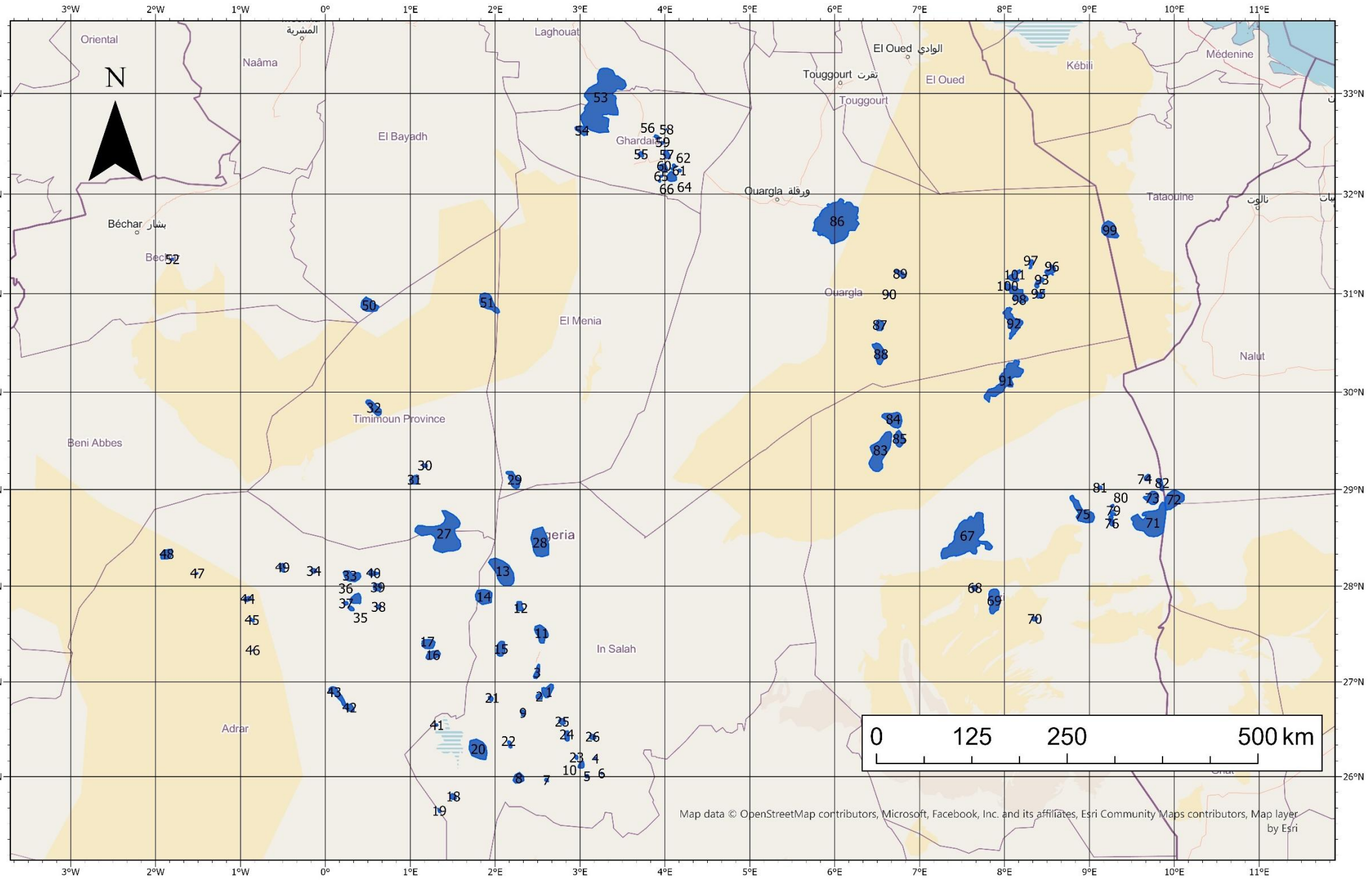


Figure 59: Numbered oil and gas field map

Table 1: Boundings for mapped Algerian fields

Nº	Field Name	LongMin	LongMax	LatMin	LatMax
1	Gour Mahmoud	1.618772	3.618772	25.90473	27.90473
2	Hassi Hassine	1.520698	3.520698	25.84444	27.84444
3	In Sallah	1.49459	3.49459	26.10151	28.10151
4	Mahbes Guenatir	2.182007	4.182007	25.19104	27.19104
5	Djebel Thara	2.082693	4.082693	25.00474	27.00474
6	Unknown	2.254464	4.254464	25.03616	27.03616
7	Krebb Ed Douro	1.606319	3.606319	24.96279	26.96279
8	Tibardine	1.276927	3.276927	24.98251	26.98251
9	Oued Djaret	1.329972	3.329972	25.67153	27.67153
10	Djebel Mouahdrine	2.016066	4.016066	25.12113	27.12113
11	Hassi Moumene	1.544895	3.544895	26.50759	28.50759
12	Garet El Befinat	1.286967	3.286967	26.79062	28.79062
13	Reg	1.090235	3.090235	27.14886	29.14886
14	Bouteraa	0.867103	2.867103	26.89035	28.89035
15	Garet El Guef	1.067896	3.067896	26.34577	28.34577
16	Oued Talha	0.269069	2.269069	26.28196	28.28196
17	Hassi M'Sari	0.213089	2.213089	26.39746	28.39746
18	Adrar Morrat NE	0.507646	2.507646	24.78721	26.78721
19	Adrar Morrat SW	0.342025	2.342025	24.6368	26.6368
20	Bahar El Hammar	0.799932	2.799932	25.28829	27.28829
21	Tit	0.949045	2.949045	25.82488	27.82488
22	Tirechoumine	1.173883	3.173883	25.34059	27.34059
23	Anasmit	1.958895	3.958895	25.20431	27.20431
24	In Bazzene	1.845299	3.845299	25.43049	27.43049
25	In Bazzene N	1.789646	3.789646	25.58308	27.58308
26	Djebel Belda	2.149968	4.149968	25.41968	27.41968
27	Hassi Barouda	0.39071	2.39071	27.54333	29.54333
28	Teguentour	1.529539	3.529539	27.45318	29.45318
29	Krechba	1.22075	3.22075	28.10089	30.10089
30	Erg Choueref	0.172232	2.172232	28.24736	30.24736
31	Bejouen	0.048506	2.048506	28.10105	30.10105
32	Ramedj	-0.4258	1.574196	28.8392	30.8392
33	Oued Zine	-0.68848	1.311522	27.10283	29.10283
34	Hassi Sbaa	-1.13753	0.862465	27.15955	29.15955
35	Hassi Llatoou	-0.69677	1.303225	26.76801	28.76801
36	Hassi Llatoou NE	-0.63814	1.361858	26.86594	28.86594
37	Hassi Llatoou Cambrien	-0.75645	1.243551	26.82124	28.82124
38	Qued Tourhar	-0.3727	1.627301	26.7855	28.7855
39	Azzene	-0.386	1.614004	26.9898	28.9898
40	Foukroun	-0.43969	1.560307	27.13475	29.13475
41	Mekerrane N	0.312063	2.312063	25.54544	27.54544
42	Azrafil SE	-0.71332	1.286684	25.72597	27.72597
43	Reggane	-0.85935	1.140647	25.86477	27.86477
44	Kahal Tabelbala N	-1.91849	0.081513	26.866	28.866
45	Djebel Heirane Kahal	-1.86152	0.138483	26.64626	28.64626
46	Djebel Heirane N	-1.8551	0.144902	26.33112	28.33112
47	Feidj El Had	-2.50793	-0.50793	27.13334	29.13334
48	Hassi M'Dakane	-2.869	-0.869	27.33109	29.33109
49	Touat / Decheira	-1.50686	0.49314	27.19443	29.19443
50	Hassi Tidjerane	-0.48716	1.512844	29.8808	31.8808

51	Hassi Ba Hammou	0.916395	2.916395	29.90914	31.90914
52	Meharez	-2.80463	-0.80463	30.34845	32.34845
53	Hassi R'Mel	2.257761	4.257761	31.92535	33.92535
54	Hassi R'Mel S	2.019565	4.019565	31.63556	33.63556
55	Macouda	2.715795	4.715795	31.3953	33.3953
56	Djorf	2.908041	4.908041	31.56769	33.56769
57	Oued Noumer	3.022115	5.022115	31.39291	33.39291
58	Unknown	3.018276	5.018276	31.64427	33.64427
59	Oued Noumer	2.968834	4.968834	31.5101	33.5101
60	Oued Noumer	2.982743	4.982743	31.26112	33.26112
61	Unknown	3.171823	5.171823	31.23443	33.23443
62	Unknown	3.110875	5.110875	31.28077	33.28077
63	Unknown	3.085245	5.085245	31.25313	33.25313
64	Ait Kheir	3.078018	5.078018	31.16998	33.16998
65	Unknown	2.956173	4.956173	31.17799	33.17799
66	Unknown	2.936542	4.936542	31.13219	33.13219
67	Fouye Tabankort	6.559587	8.559587	27.50264	29.50264
68	Adouhoum	6.650705	8.650705	26.9808	28.9808
69	Mezoratine	6.878882	8.878882	26.84833	28.84833
70	Remal	7.354091	9.354091	26.65971	28.65971
71	Alrar	8.732503	10.7325	27.66768	29.66768
72	Alar (Lybia)	8.993689	10.99369	27.89798	29.89798
73	Stah	8.741478	10.74148	27.91423	29.91423
74	Mereksene	8.683642	10.68364	28.12191	30.12191
75	Ohanet	7.918675	9.918675	27.76401	29.76401
76	Dimera	8.26935	10.26935	27.64778	29.64778
77	Dimera	8.254425	10.25443	27.70002	29.70002
78	Dimera	8.271056	10.27106	27.75194	29.75194
79	Dimera	8.282382	10.28238	27.78212	29.78212
80	Dimera	8.269205	10.2692	27.83156	29.83156
81	Dimera	8.128329	10.12833	28.02007	30.02007
82	Antar	8.84134	10.84134	28.05523	30.05523
83	Rhourde Adra S	5.535668	7.535668	28.38926	30.38926
84	Rhourde Nouss	5.692757	7.692757	28.72144	30.72144
85	Rhourde Adra	5.76142	7.76142	28.52019	30.52019
86	Hassi Messaoud	5.027957	7.027957	30.72952	32.72952
87	Nezia	5.529253	7.529253	29.6774	31.6774
88	Gassi Touil	5.528674	7.528674	29.39561	31.39561
89	Rhourde Akbar	5.777513	7.777513	30.19983	32.19983
90	Rhourde Sayah	5.641731	7.641731	29.99307	31.99307
91	El Merk	7.012159	9.012159	29.12799	31.12799
92	Ourhoud	7.0936	9.0936	29.71162	31.71162
93	Bir Sal Fatima	7.442603	9.442603	30.14124	32.14124
94	Bir Sal Fatima	7.388387	9.388387	30.09571	32.09571
95	Bir Sal Fatima	7.412735	9.412735	29.9916	31.9916
96	Bir Sal Fatima	7.54193	9.54193	30.22964	32.22964
97	Bir Sal Fatima	7.313152	9.313152	30.30008	32.30008
98	Qoubba N	7.165209	9.165209	29.98842	31.98842
99	El Borma	8.238376	10.23838	30.64019	32.64019
100	Hassi Berkine	7.035809	9.035809	30.07576	32.07576
101	Hassi Berkine	7.120797	9.120797	30.17533	32.17533

2.2.12 Shutting down the instance

Once you have run the inversion and collected the data you need, you must shut down the inversion to avoid any unnecessary costs. Open a browser and navigate to https://aws.amazon.com/, then login to your console using the user details you set up previously (fig. 60).

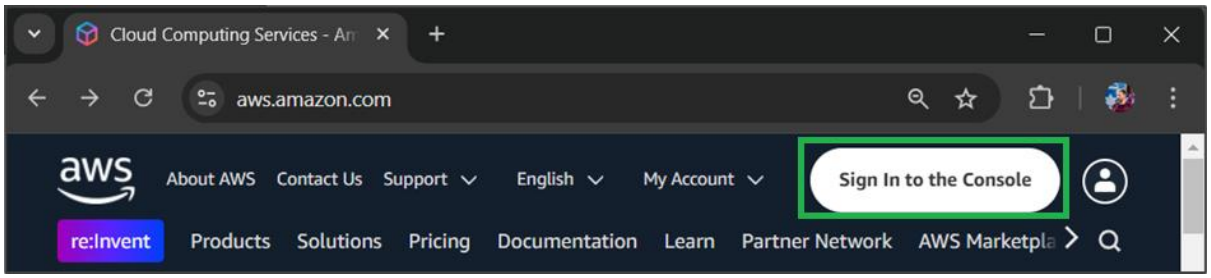


Figure 60: Sign into console button.

Once you have logged into your console, click on the EC2 Global view link (fig. 61).

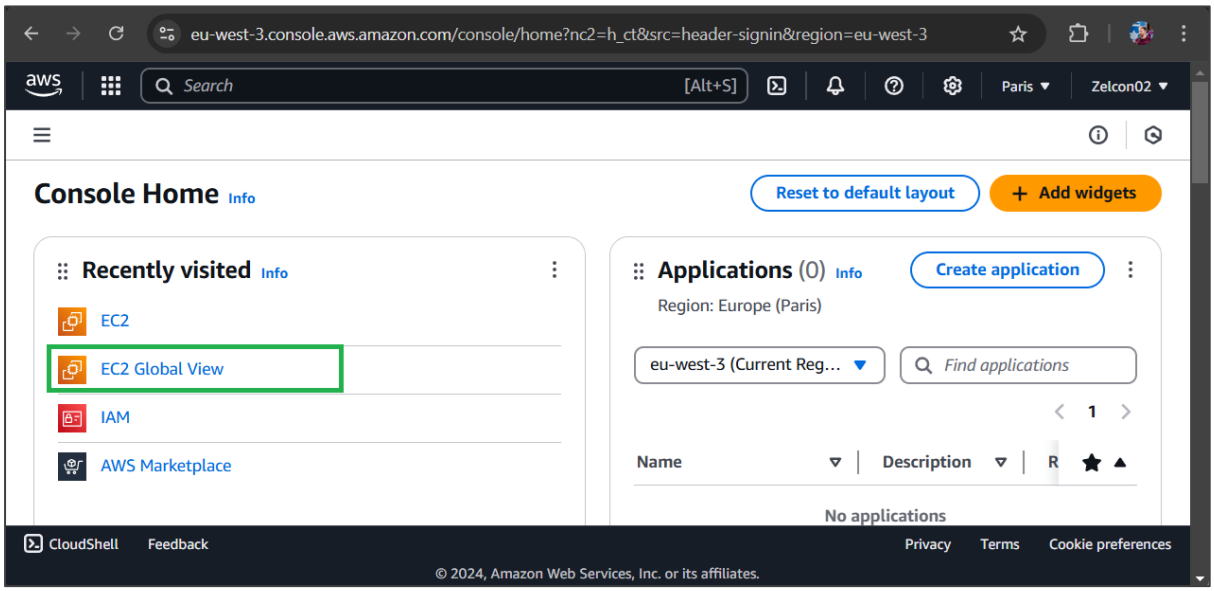


Figure 61: EC2 Global View button

You should see under “instances” 1 in 1 region (or however many instances you set up). Click on that link (fig. 62) and select “view all”.

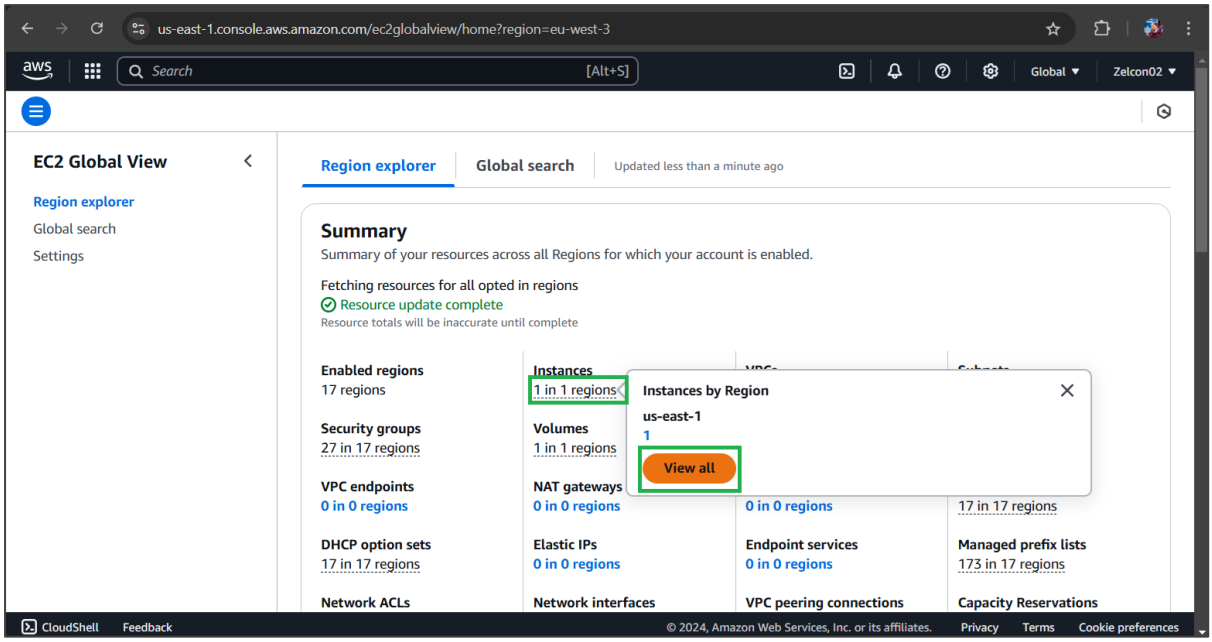


Figure 62: Instances field showing how many instances you have open.

Next click on the blue "resource ID" link (fig. 63)

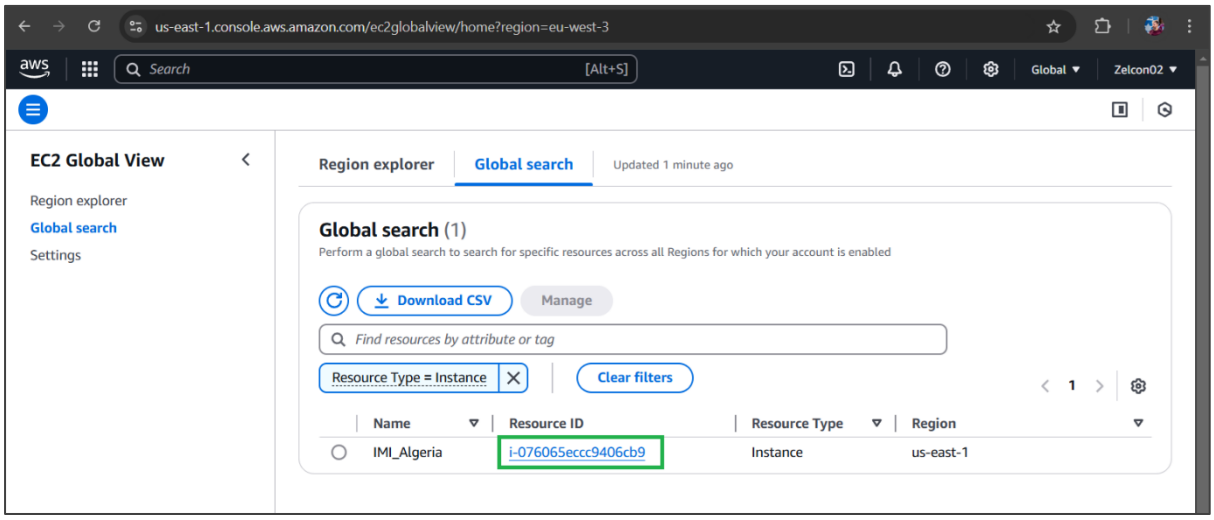


Figure 63: resource DI link location.

Finally open the "instance state" dropdown menu and select "stop instance" (fig. 64). Thereafter you should receive a message confirming the operation.

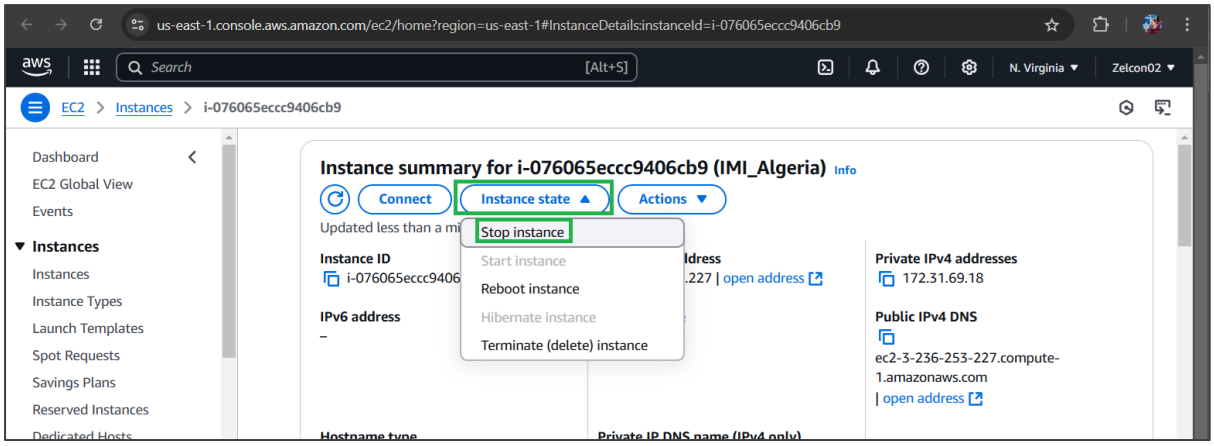


Figure 64: location of stop instance button.

2.3 Troubleshooting

If you are having problems running the IMI, please submit a ticket at the following Github page where the developers will attempt to help you out.

https://github.com/geoschem/integrated_methane_inversion/issues

3. Sentinel-2 Multi-Band-Multi-Pass CH_4 Mapper (S2MBMP)

3.1 Methodology

To help users locate emission sources in a field, Sentinel 2's MSI instrument with a $20m^2$ spatial resolution and return frequency of 3-5 days was employed.

Varon et al. (2021) demonstrated that methane columns from point sources can be measured by exploiting the SWIR-1 and SWIR-2 bands of Sentinel-2. A Python code has been developed accessing Sentinel 2's L1C data using the Copernicus OpenEO API. Scenes without obscuring cloud or smoke will be chosen as the non-emission scene and then these will be tested against other scenes with a maximum of 5% cloud cover. No emission and active emission scenes were each processed using the multi-band-single-pass equation as outlined in Varon et al. (2021):

$$MBSP = B11 - cB12$$

Where: ***B12*** is the Sentinel-2 SWIR-2 band, ***B11*** is the Sentinel-2 SWIR-1 band and ***c*** is calculated by least squares fitting B12 against B11.

Once the MBSP raster had been calculated, the following equation was then used to calculate the multi-band-multi-pass raster:

$$MBMP = ActiveMBSP - NoMBSP$$

Where **ActiveMBSP** is the multiband single pass for the active emission scene and **NoMBSP** is the multiband single pass for the no emission scene.

Figure 65 provides an overview of the main steps in processing.

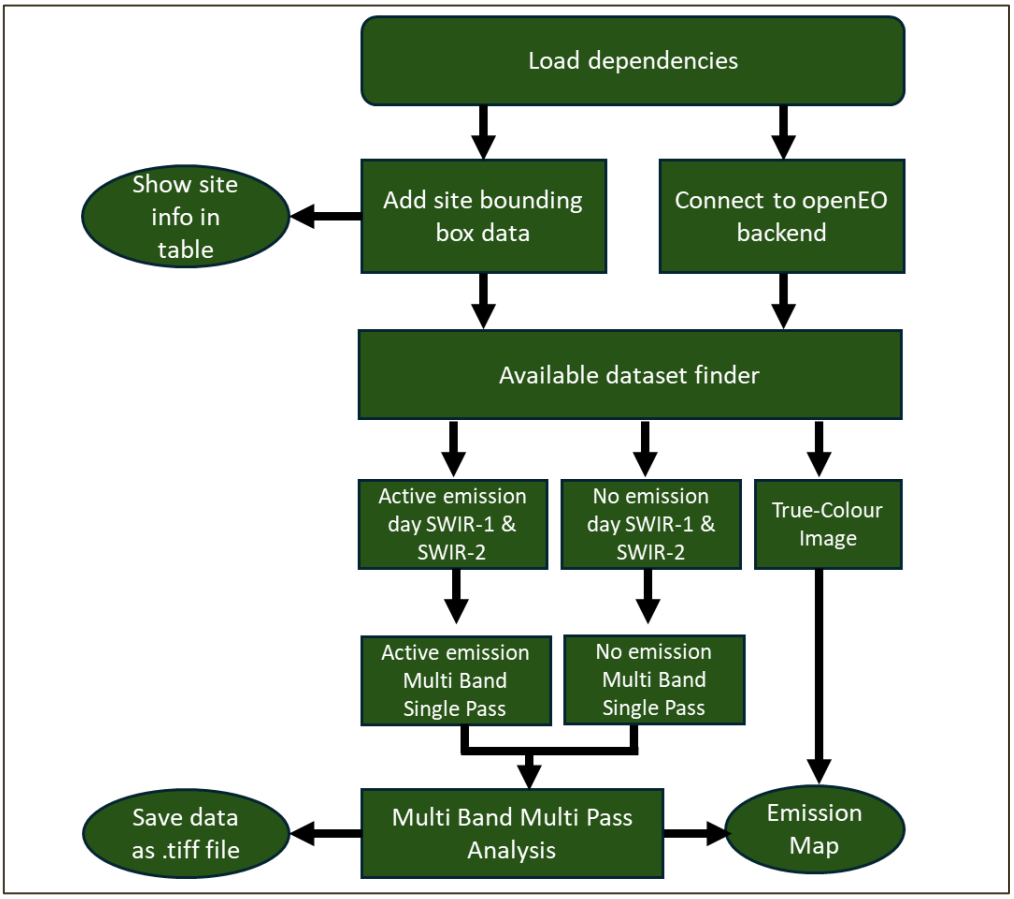


Figure 65: Flowchart of processes used for Sentinel 2 CH_4 Map

3.2 Data and Dependencies

The datasets used by these tools are outlined in table 2.

Table 2: Datasets used by tools

Description	Data	Source	For use in tool(s)
Oil and Gas Field Bounding	Long/Lat .csv file.	Manual surveying of arial images of Algeria by author	S2-MBMP
Sentinel 2 (MSI) bands 2, 3, 4, 11 and 12	Raster 10m2 for 2, 3 and 4 and 20m2 for 11 and 12	Copernicus Dataspace – OpenEO.	S2-MBMP

The dependencies contained in the environment.yml file are outlined in table 3.

Table3: dependencies required to use the tools.

Name	Version	Description
Python	3.12.2	Programming language to run the code
Jupyterlab	4.1.4	Computing environment for Python
Folium	0.16.0	Generates interactive maps with Leaflet.js.
Pandas	2.2.1	Data manipulation and analysis.
Geopandas	0.14.3	Geospatial data manipulation and analysis.
Matplotlib	3.8.3	Creates map visualizations
Rasterio	1.3.9	Reads and edits raster datasets.
Numpy	1.26.4	Performs numerical computations on arrays
Requests	2.31.0	Used fetch data from web services or APIs.
Rasterstats	0.19.0	Linear regression for brightness correction
OpenEO	0.28.0	API for Earth observation data processing
Shapely	2.0.3	Used to manage points, lines and polygon data

3.3 Setup

Anaconda Navigator, containing the Python programming language and 'Conda', a package manager for creating shareable environments that tools like this can run on will be installed. This includes Jupyter Lab which will be used to run this tool's code.

3.3.1 Anaconda Navigator

To download Anaconda, navigate to https://docs.anaconda.com/anaconda/install/ and follow the instructions of your operating system.

3.3.2 Creating a Conda Environment

In the Anaconda Navigator side bar, click the 'Environments' tab. You will see the installed packages (fig.66).

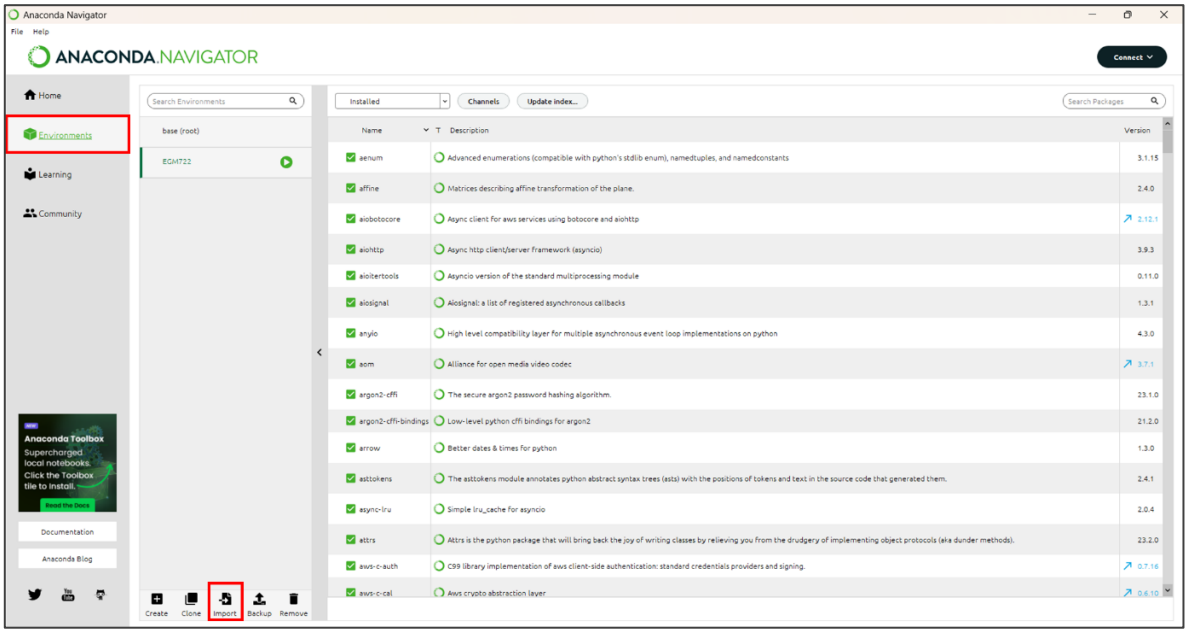


Figure 66: Environments tab of Anaconda Navigator with environments tab and import button highlighted in red.

Next click on the imports tab (fig.3) and select the file 'environment.yml' contained in the .zip file of the tool's download, choosing an appropriate name for the environment (fig. 67).

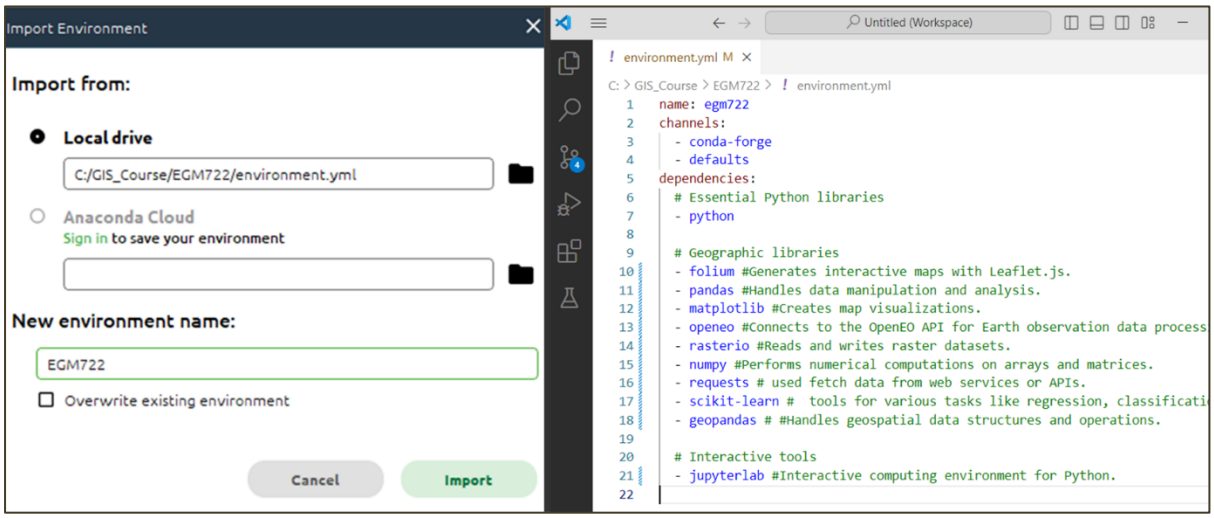


Figure 67: The import config box (left) and the contents of 'environment.yml' (right).

Click Import. The process of installing all the packages may take several minutes. Once finished you will be returned to the environments tab (fig.3)

Next click on the 'Home' tab in Anaconda Navigator's sidebar (fig. 68).

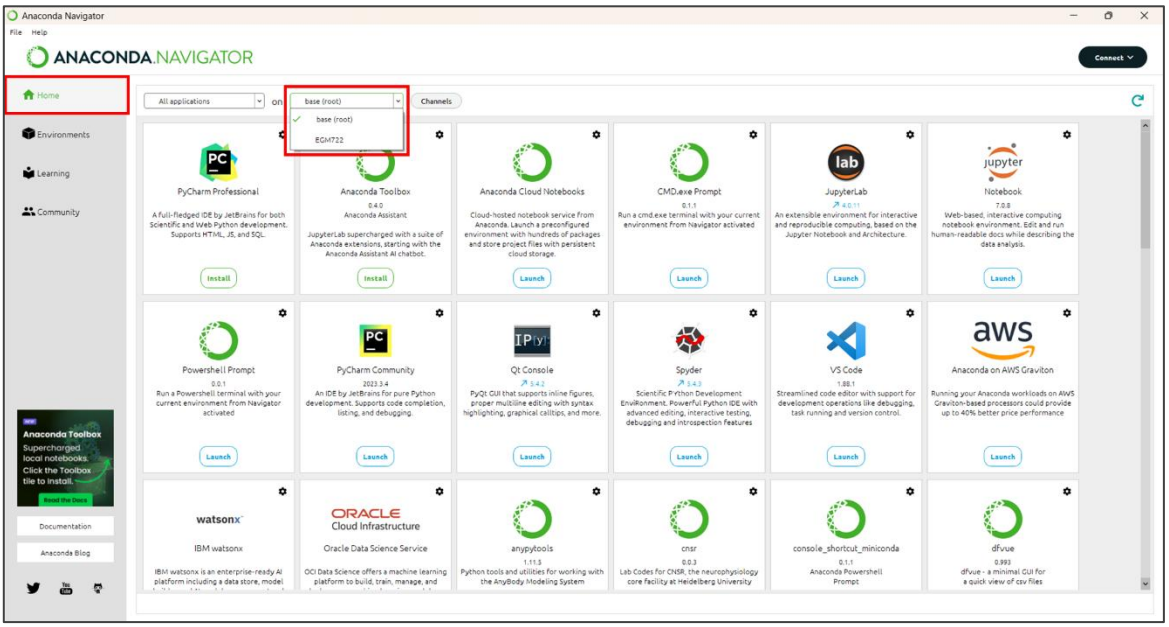


Figure 68: Anaconda Navigator with home tab and environment switching dropdown in red.

The dropdown highlighted in figure 68 should display two options, 'base (root)' and the name of your new environment (in figure 68 this is 'EGM722). **Ensure your environment name is always selected here or the dependencies installed earlier will not be available to it.**

3.3.3 Setting up Jupyter Lab

A configuration file ('.config') needs to be created to change the settings used by Jupyter Lab by default. Launch the CMD.exe prompt (fig.69)

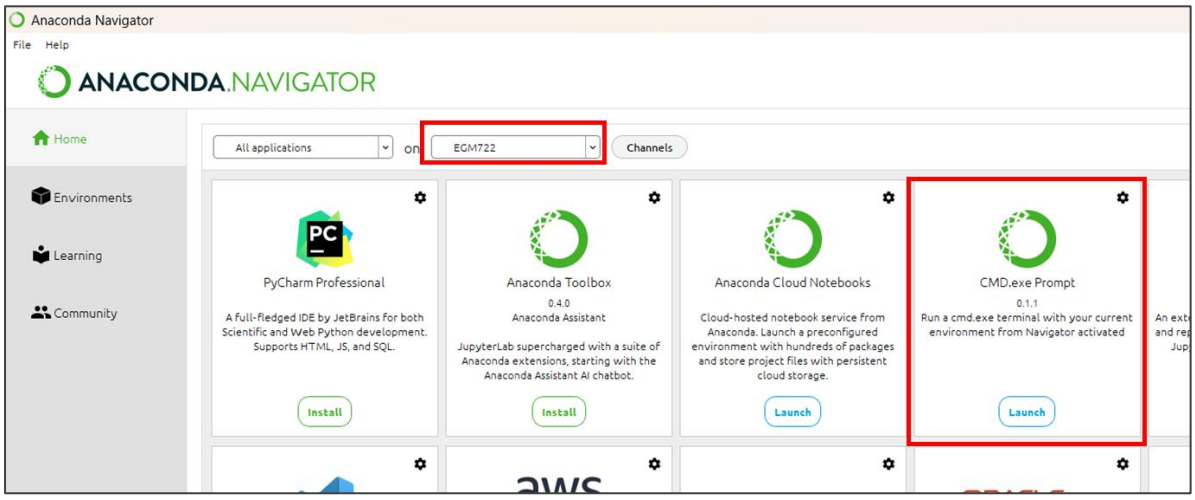


Figure 69: Highlighted locations of selected environment and CMD.exe Prompt

In the command prompt, enter the command:

```
jupyter lab --generate-config
```

This will create a new folder in your user directory called '.jupyter' containing a python script juptyer_lab_config.py. On Windows this is usually 'C:\Users\<your_username>'.

Jupyter lab will by default open in your user directory. Due to security restrictions, it is not possible to navigate to the parent directory of the launch location. So if Jupyter launches in 'C:\Users\RockyBalboa, it is not possible to move to 'C:\Users' or, 'C:\EGM722'. If the directory you are keeping your data in is outside your user directory, you will need to change the default opening folder to your data directory.

This location is also where you should store the following files and folder:

- Sentinel_5P_Atmospheric_Gas_Time_Series.ipynb
- Sentinel_5P_Atmospheric_Gas_Map.ipynb
- Sentinel_2_CH4_Multi-Band-Single-Pass.ipynb
- The folder "Data"

If your data directory is in your user directory, you should be able to click and navigate there using the interface of Jupyter Lab. If that is not the case, you will need to do the following:

Open an Anaconda Navigator CMD.exe prompt and type the following command:

```
jupyter --paths
```

This will show something like figure 70 although your file paths will be unique to you.

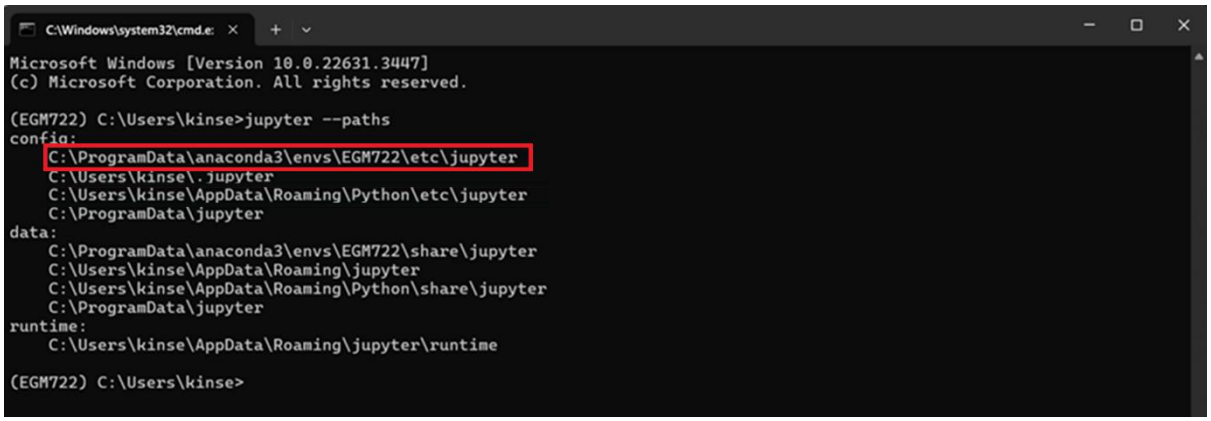


Figure 70: results of 'jupyter –paths' command showing path used by environment highlighted in red.

The 'jupyter_lab_config.py' file mentioned earlier needs to be copy pasted into that folder.

Once 'jupyter_lab_config.py' file has been moved, open it in Notepad++, Visual Studio Code or if you don't have those, Notepad. Using the shortcut 'CTRL + F' type in the following line: 'c.ServerApp.root_dir' (without quotes) and you should find the section highlighted in figure 71.

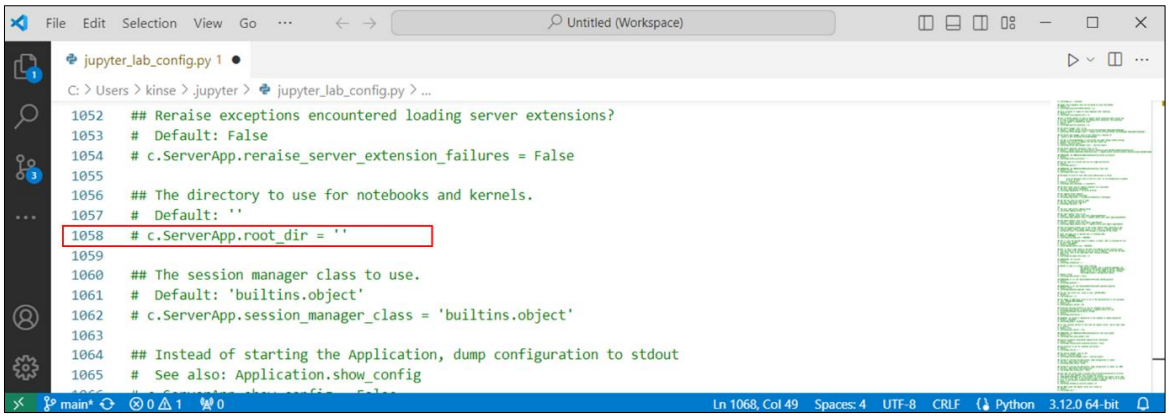


Figure 71: location of 'c.ServerApp.root_dir' in jupyter_lab_config.py

Remove the '#' and space from the start and add the path used by your environment between the quote marks, adding a 'r' beforehand (fig.72).

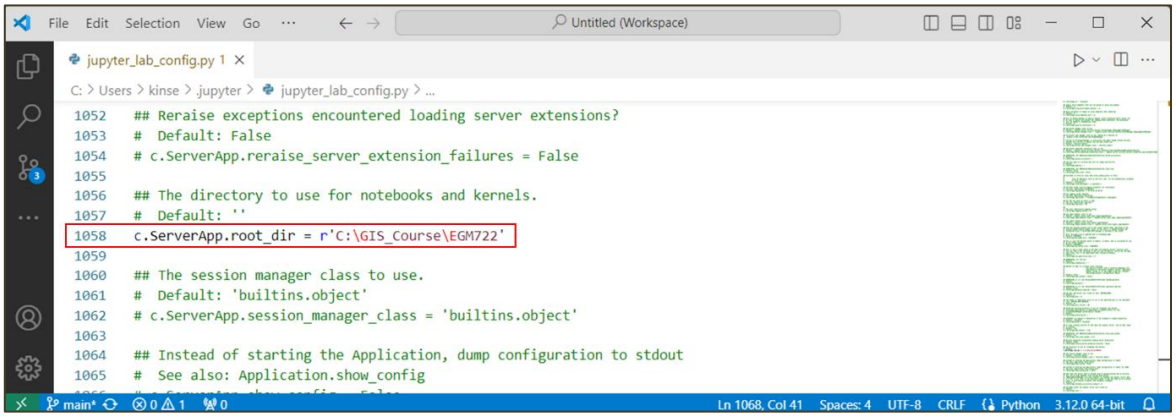


Figure 72: path to data directory added to jupyter_lab_config.py

Save and close this file and return to the Anaconda Navigator 'Home' tab. Launch Jupyter Lab and if you have followed the steps correctly, you should see that your data directory is automatically displayed (fig.73).

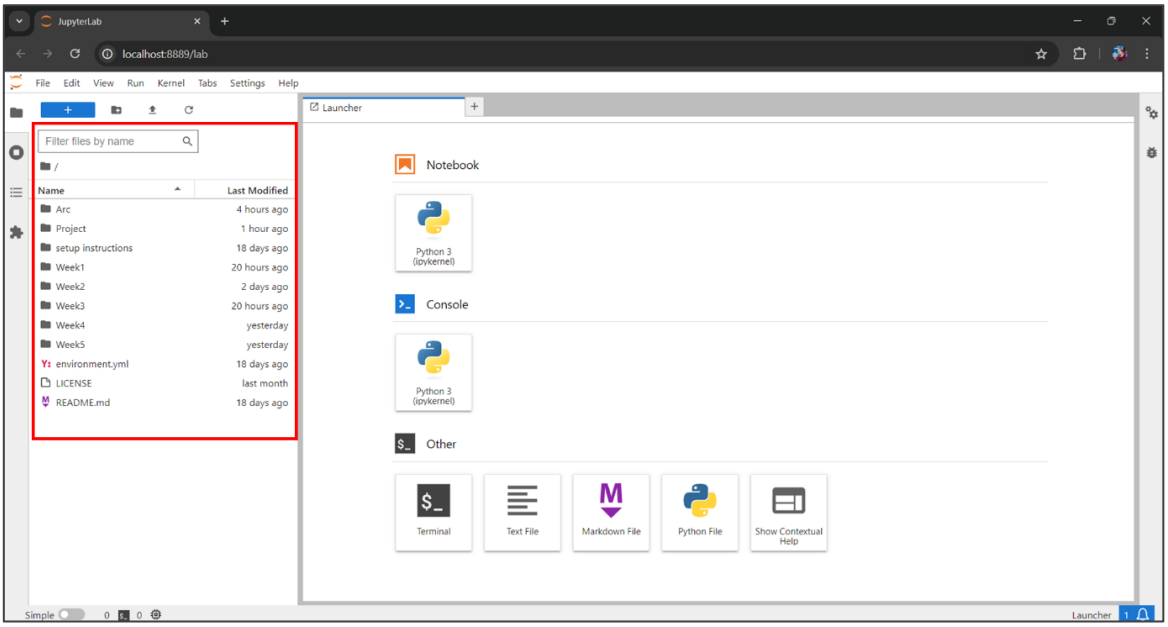


Figure 73: Jupyter Lab showing by default the data directory

3.3.4 OpenEO setup using Anaconda Navigator

OpenEO is an open-source API that allows access to the earth observation satellite missions run by the Copernicus program. These include the satellites used by this tool.

First search in the Anaconda Navigator environments tab for 'openeo'. Make sure that 'Not installed' is selected (fig.74). If the package appears here, click its tick box and select apply. If you can't see it here, please go to section 2.5.

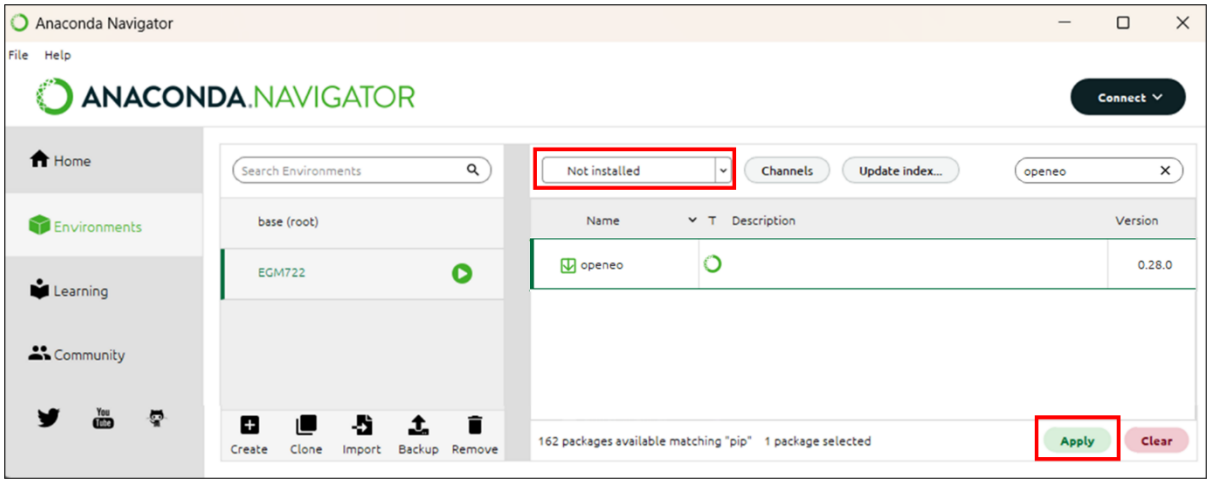


Figure 74: installing OpenEO from Anaconda Navigator.

Next you will be presented with the following screen (fig. 75). One this has finished processing the request. Simply click 'apply' to begin the installation.

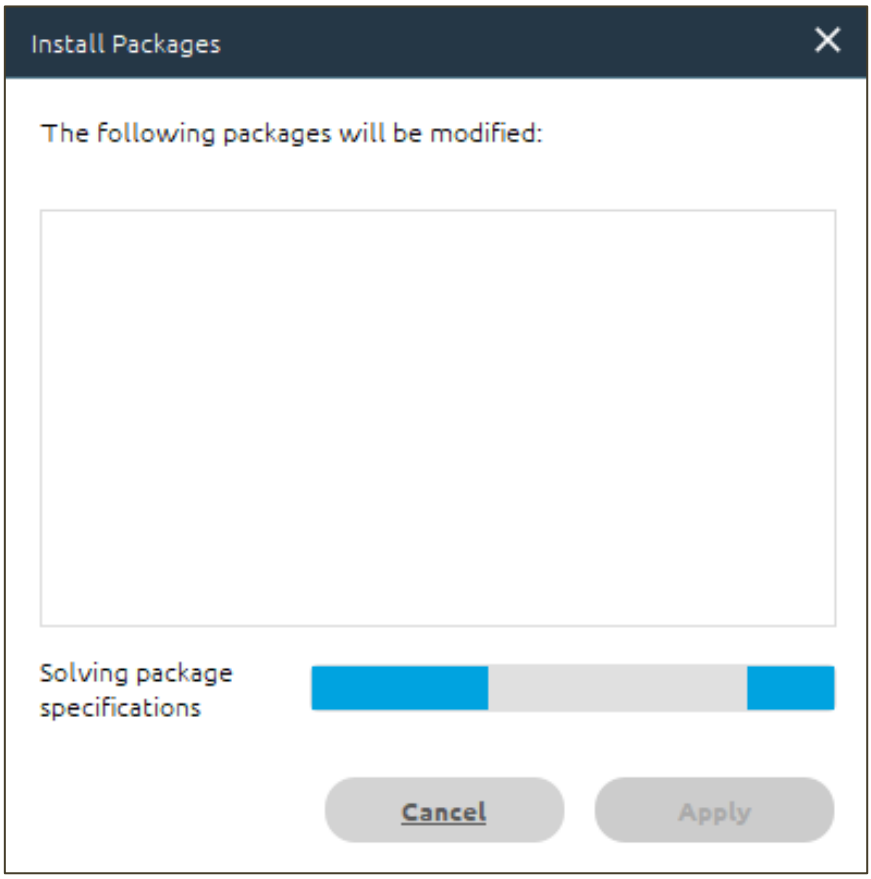


Figure 75: Anaconda Navigator package installer loading screen.

3.3.5 OpenEO setup using PyPi

If Anaconda Navigator cannot find OpenEO you can use PyPi, the official third-party software library for Python. Search for 'pip', selecting the appropriate tick-box and then click apply, then clicking apply once the install packages prompt has finished loading (fig.76).

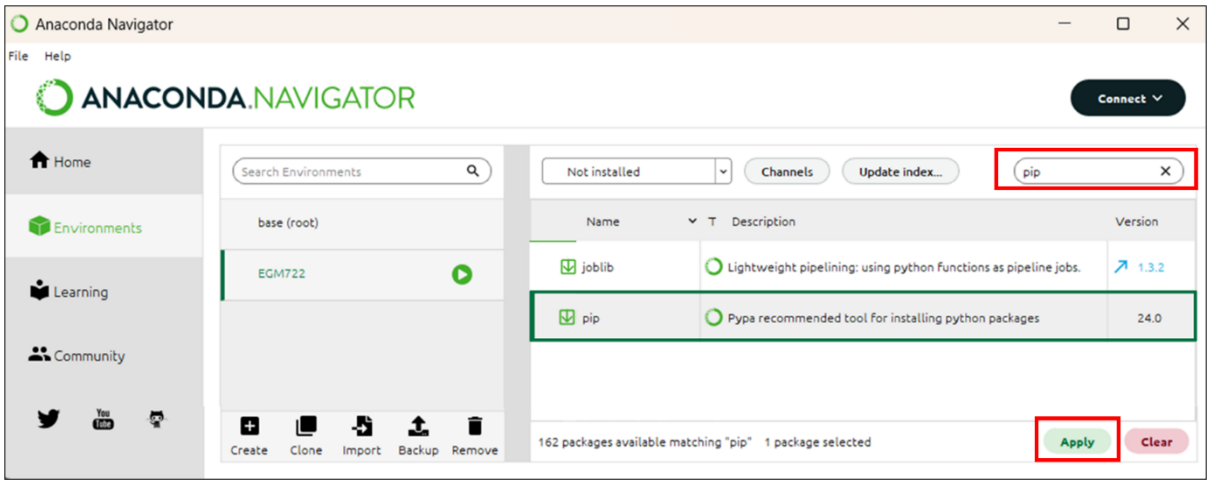


Figure 76: Installing pip via Anaconda Navigator

Open an Anaconda Navigator CMD.exe prompt and type the following command:

```
pip install openeo
```

Once the process has completed, you can close the CMD.exe prompt window.

2.3.6 Registering with Copernicus Data Space Ecosystem.

Accessing OpenEO requires an authentication. To do this, you need to complete a Copernicus Dataspace Registration. Go to https://dataspace.copernicus.eu/ and click the green login button (fig.77)

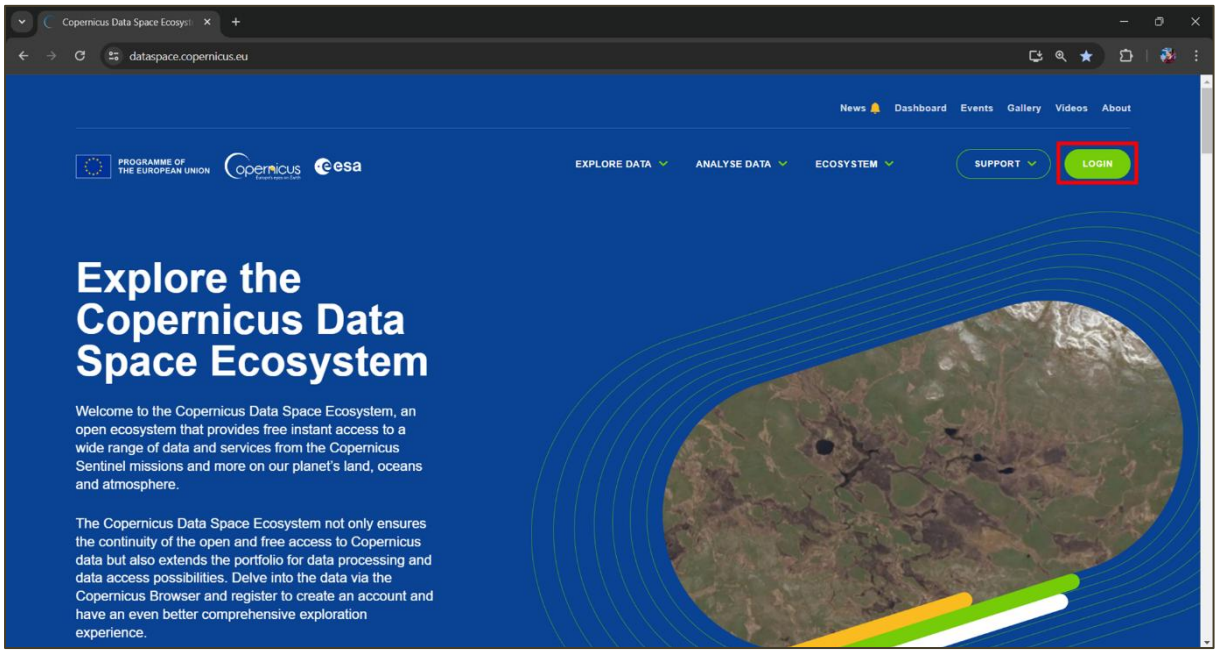


Figure 77: Copernicus Dataspace landing page with login button highlighted in red.

Next click the green 'register' button (fig.78):

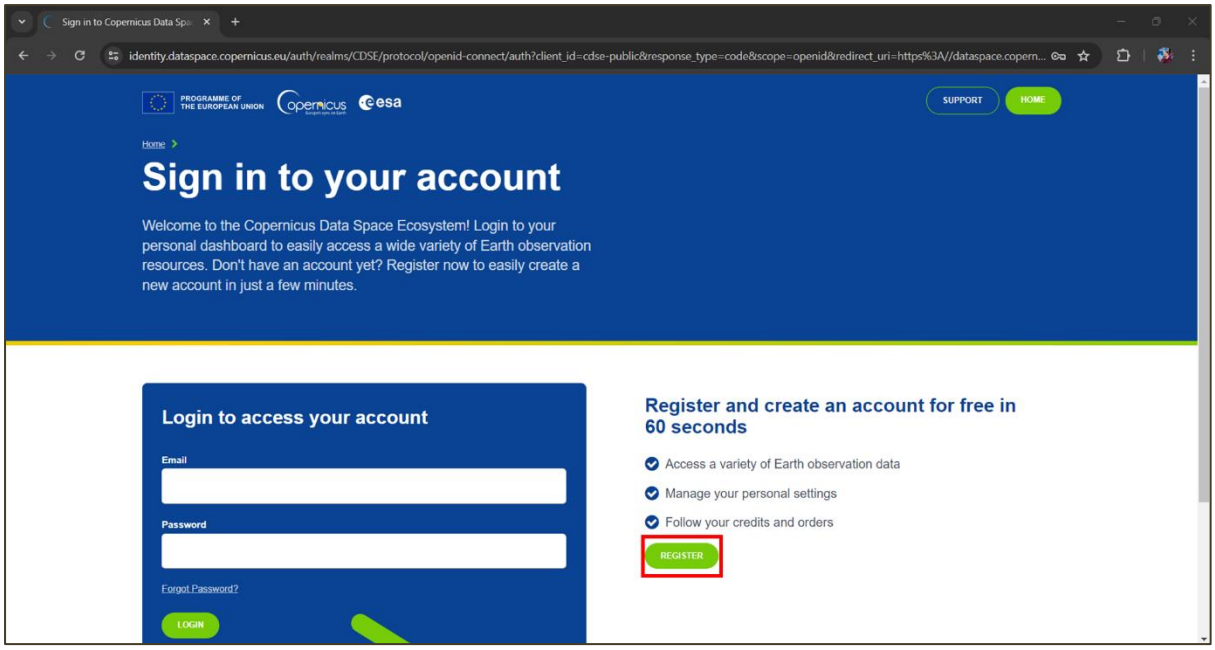


Figure 78: Copernicus Dataspace sign in page.

On the following page, fill out the application form and then at the bottom click the green 'register' button (fig.79).

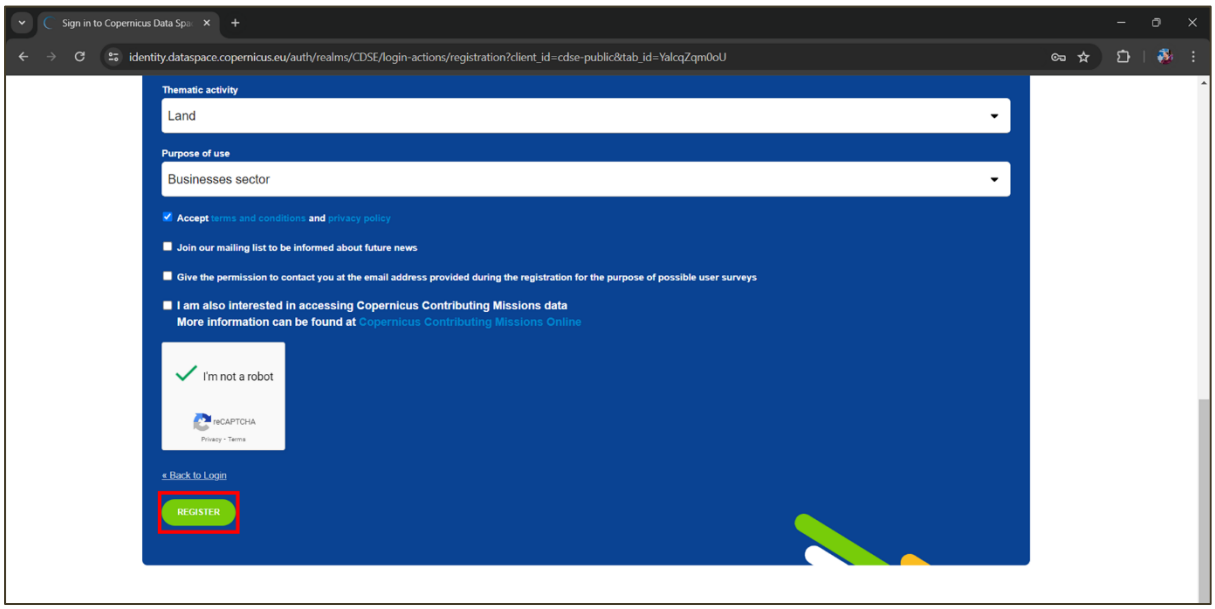


Figure 79: End of Copernicus registration page with register button highlighted in red.

Once registered, you will receive an email asking to verify your address. You can then log-in with your email and chosen password.

For any registration problems, email: help-login@dataspace.copernicus.eu

3.3.7 Running the tools in Jupyter-lab

Now that (almost) everything has been setup you can launch Jupyter Lab in the Anaconda Navigator (fig. 80). Remember as always that your project environment (here 'EGM722') should be selected and not 'base (root)'.

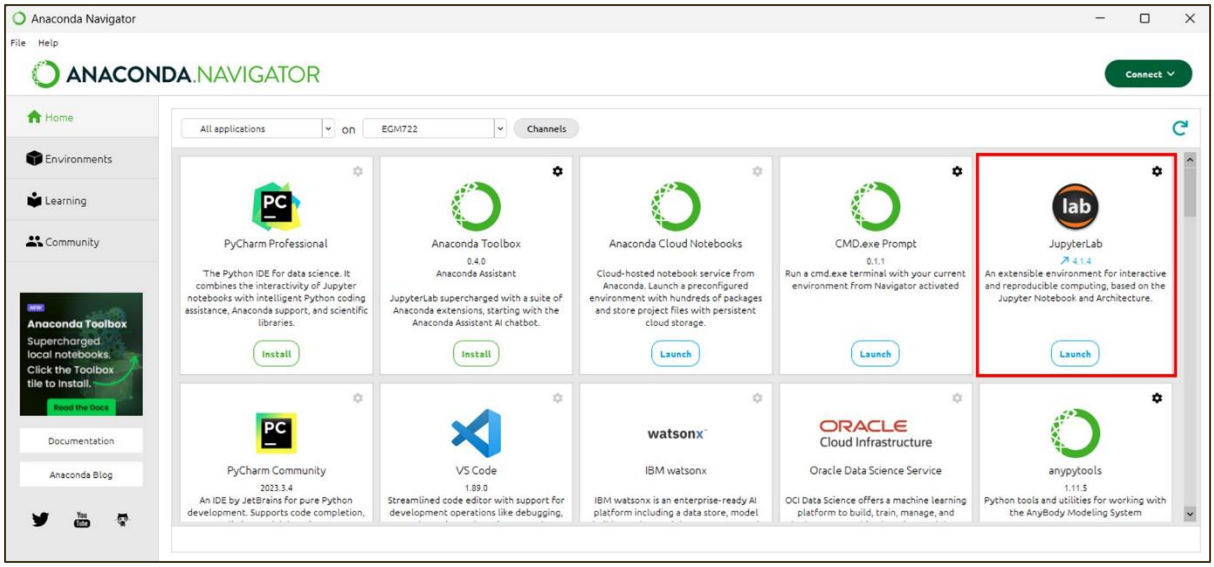


Figure 80: Location of Jupyter Lab in Anaconda Navigator highlighted in red.

Once Jupyter lab opens, you should see the three tools on the left (fig. 81).

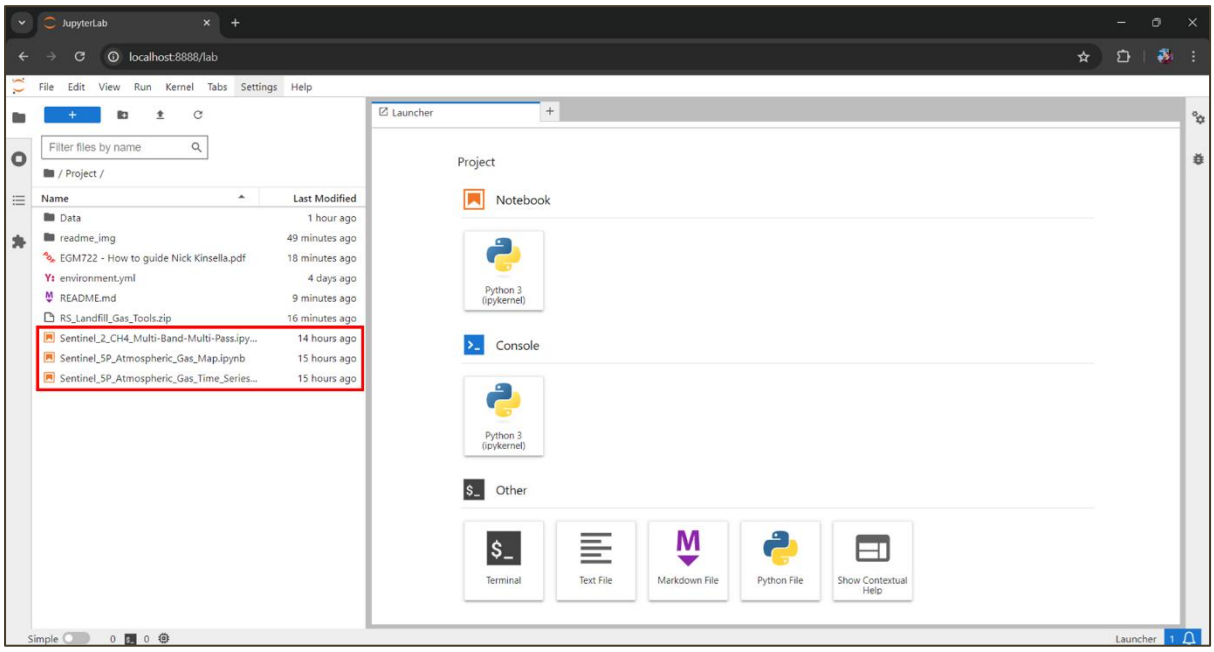


Figure 81: Location of tool scripts in Jupyter lab highlighted in red.

Click on one of the tools to open it. You can then follow the instructions, running the code by clicking the play button (fig. 82).

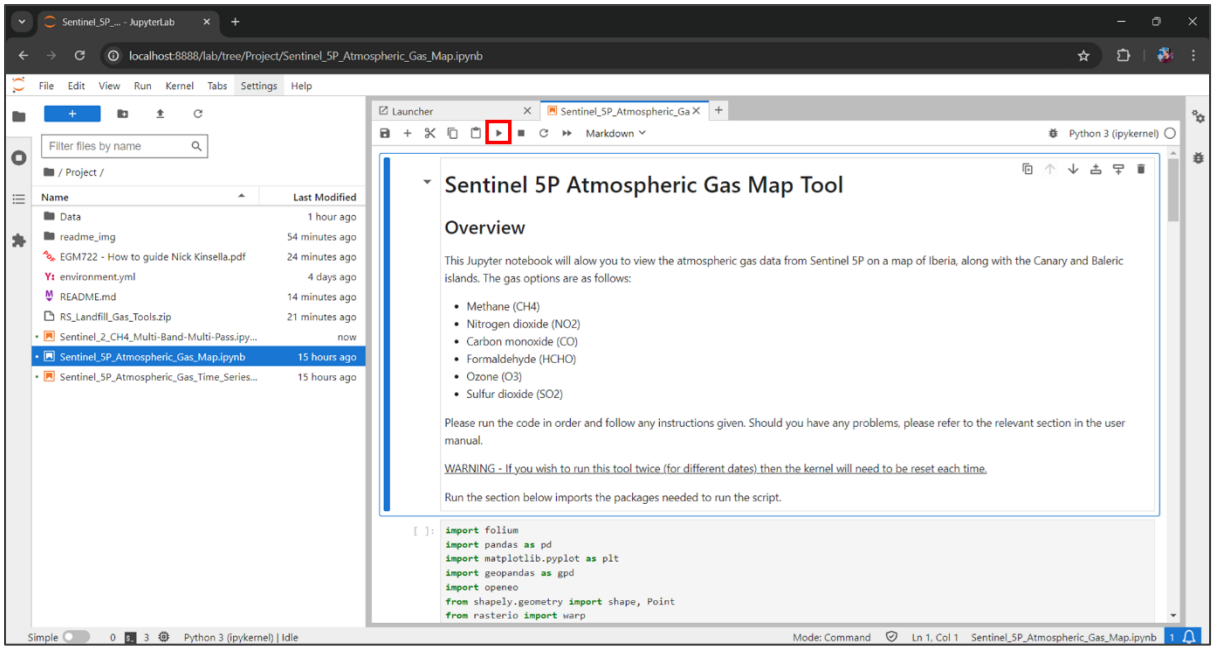


Figure 82: Location of the 'play' button which runs each of the code segments of the workbooks.

3.3.8 Authentication with OpenEO

The very first time one of the tools are run, the following section of code…

```
connection = openeo.connect(url="openeo.dataspace.copernicus.eu")
connection.authenticate_oidc()
```

… will provide you with a URL that will look something like this:

> Visit https://auth.example.com/device?user_code=EAXD-RQXV to authenticate.

Copy this into your web browser and login using the Copernicus Data Space. Once this is complete, run tool's Python script again and it will receive an authentication token, printing the message:

> Authorized successfully.

In future you may be prompted with a new URL to create a new authentication token, whereby you should repeat the steps of this section.

3.4 S2MBMP Code

Code 1 shows the loaded dependencies.

```
import folium
import pandas
import matplotlib.pyplot as plt
import openeo
import rasterio
from rasterio.transform import from_origin
import numpy as np
import requests
from folium import Map, LayerControl, LatLngPopup, GeoJson
from folium.raster_layers import ImageOverlay
from IPython.display import display
from skimage import exposure
import geopandas as gpd
```

Code 1: Loading of dependencies for the code to run.

Code 2 connects to the OpenEO backend.

```
connection = openeo.connect(url="openeo.dataspace.copernicus.eu")
connection.authenticate_oidc()
```

Code 2: Connecting to OpenEO

Code 3 reads a pre-processed file that contains the field location bounding boxes, and then displays it for the end user to see.

```
studysite_csv =
pandas.read_csv(r'C:\GIS_Course\Methane_Point_Detection\all_files\Data\A
lgerian_Oil_and_Gas_Fields.csv')
pandas.set_option('display.max_rows', None)
print(studysite_csv)
```

Code 3: Displaying the contents of the site file for easy reference.

Code 4 provides the available datasets for a date range and site selected by the user. It also has a cloud filter set at 15% as any cloud in the scene interferes with the tool. This section of code was provided by Sonneveld, E. (2024)

```
def get_spatial_extent(site_id):
    site = studysite_csv[studysite_csv['id'] == site_id].iloc[0]
    return {
        "west": site['west'],
        "south": site['south'],
        "east": site['east'],
        "north": site['north']
    }

def fetch_available_dates(site_id, temporal_extent):
    spatial_extent = get_spatial_extent(site_id)
```

```
                                        catalog_url                    =
f"https://catalogue.dataspace.copernicus.eu/resto/api/collections/Sentin
el2/search.json?box={spatial_extent['west']}%2C{spatial_extent['south']}
%2C{spatial_extent['east']}%2C{spatial_extent['north']}&sortParam=startD
ate&sortOrder=ascending&page=1&maxRecords=1000&status=ONLINE&dataset=ESA
-
DATASET&productType=L2A&startDate={temporal_extent[0]}T00%3A00%3A00Z&com
pletionDate={temporal_extent[1]}T00%3A00%3A00Z&cloudCover=%5B0%2C{cloud_
cover}%5D"
    response = requests.get(catalog_url)
    response.raise_for_status()
    catalog = response.json()
        dates   =   [date.split('T')[0]   for   date   in   map(lambda   x:
x['properties']['startDate'], catalog['features'])]
    return dates

# Please enter your perameters here.
site_id = 86 # Specify the site ID.
temporal_extent = ["2020-01-01", "2020-04-30"]  # Specify the the date
range you want to check for available data.
cloud_cover = 5

available_dates = fetch_available_dates(site_id, temporal_extent)
print("Available dates:", available_dates)
```

Code 4: Code for determining available dates for analysis.

Next the user will select the day to be investigated for emissions (code 5).

```
def active_emission(site_id, temporal_extent):
    site = studysite_csv[studysite_csv['id'] == site_id].iloc[0]

    active_emission = connection.load_collection(
        "SENTINEL2_L2A",
        temporal_extent=temporal_extent,
        spatial_extent={
            "west": site['west'],
            "south": site['south'],
            "east": site['east'],
            "north": site['north']
        },
        bands=["B11", "B12"],
    )
    active_emission.download("Sentinel-2_active_emissionMBMP.Tiff")

# Enter perameters for the active emission day
site_id = 86  # Specify the site ID
temporal_extent = ["2020-01-07", "2020-01-07"]

active_emission(site_id, temporal_extent)
```

Code 5: Code for downloading active emission dataset.

Then the user will choose a “no emission day” to compare the “active emission day” to. (code 6).

```
def no_emission(site_id, temporal_extent):
    site = studysite_csv[studysite_csv['id'] == site_id].iloc[0]

    no_emission = connection.load_collection(
        "SENTINEL2_L2A",
        temporal_extent=temporal_extent,
        spatial_extent={
            "west": site['west'],
            "south": site['south'],
            "east": site['east'],
            "north": site['north']
        },
        bands=["B11", "B12"],
    )
    no_emission.download("Sentinel-2_no_emissionMBMP.Tiff")

# Enter perameters for the active emission day
site_id = 86  # Specify the site ID
temporal_extent = ["2020-01-04", "2020-01-04"]

no_emission(site_id, temporal_extent)
```

Code 6: Code for downloading active emission dataset.

Code 7 downloads a true colour satellite image to aid in the visualisation of the data.

```
def truecolour_image(site_id, temporal_extent):
    site = studysite_csv[studysite_csv['id'] == site_id].iloc[0]

    truecolour_image = connection.load_collection(
        "SENTINEL2_L2A",
        temporal_extent=temporal_extent,
        spatial_extent={
            "west": site['west'],
            "south": site['south'],
            "east": site['east'],
            "north": site['north']
        },
        bands=["B02", "B03", "B04"],
    )
    truecolour_image.download("Sentinel-2_truecolourMBMP.Tiff")

# Enter parameters for the no emission day
site_id = 86  # Specify the site ID
temporal_extent = ["2020-01-07", "2020-01-07"]

truecolour_image(site_id, temporal_extent)
```

Code 7: Code for downloading true colour satellite image for data visualisation

Code 8 runs the analysis as outlined at the beginning of this section (4.3).

```
# Define file paths
Active_Multiband = "Sentinel-2_active_emissionMBMP.Tiff"
No_Multiband = "Sentinel-2_no_emissionMBMP.Tiff"

import numpy as np
import rasterio

# Define a function for least squares fitting
def least_squares_fit(x, y):
    # Remove NaNs (if any) for valid calculations
    mask = ~np.isnan(x) & ~np.isnan(y)
    x_valid = x[mask]
    y_valid = y[mask]

    # Calculate least squares fit parameters
    A = np.vstack([x_valid, np.ones_like(x_valid)]).T
    m, c = np.linalg.lstsq(A, y_valid, rcond=None)[0]
    return m, c

# Open datasets and perform least squares fitting
with rasterio.open(Active_Multiband) as Active_img,
rasterio.open(No_Multiband) as No_img:
    Active_B11 = Active_img.read(1)
    Active_B12 = Active_img.read(2)
    No_B11 = No_img.read(1)
    No_B12 = No_img.read(2)

    # Perform least squares fitting for Active_B11 vs Active_B12
       m_active, c_active = least_squares_fit(Active_B11.flatten(),
Active_B12.flatten())
    Corrected_Active_B12 = m_active * Active_B12 + c_active

    # Perform least squares fitting for No_B11 vs No_B12
    m_no, c_no = least_squares_fit(No_B11.flatten(), No_B12.flatten())
    Corrected_No_B12 = m_no * No_B12 + c_no

    # Calculate the fractional change
      SWIR_diff = (Active_B11 - Corrected_Active_B12) - (No_B11 -
Corrected_No_B12)


# Define the output file path
output_file = "SWIR_diff_output.tiff"

# Define the transform and metadata for the output file
transform = Active_img.transform
meta = Active_img.meta.copy()
meta.update({
```

```
    "driver": "GTiff",
    "height": SWIR_diff.shape[0],
    "width": SWIR_diff.shape[1],
    "count": 1,
    "dtype": SWIR_diff.dtype,
    "crs": Active_img.crs,
    "transform": transform
})

# Save the SWIR_diff array to the output file
with rasterio.open(output_file, "w", **meta) as dest:
    dest.write(SWIR_diff, 1)

print(f"SWIR_diff has been saved to {output_file}")
```

Code 8: Code running analysis

Code 9 displays the map. It firstly takes the true colour satellite image bands, stacks them and applies a brightness factor to make the image clearer. Then it adds the SWIR_diff data calculated in the previous code and sets it to only display data between 1.5 and 3 standard deviations above the mean. It adds scaled grid lines to help understand distances on the image. Finally it downloads the map as a .jpg file.

```
# Load the true color image
truecolour_sat = 'Sentinel-2_truecolourMBMP.Tiff'
img = rasterio.open(truecolour_sat)
blue = img.read(1)
green = img.read(2)
red = img.read(3)

# Adjust brightness
brightness_factor = 0.03  # Increase brightness factor
blue = np.clip(blue * brightness_factor, 0, 255)
green = np.clip(green * brightness_factor, 0, 255)
red = np.clip(red * brightness_factor, 0, 255)

# Stack bands to create RGB image
rgb = np.dstack((red, green, blue))
rgb = rgb / rgb.max()

# Apply logarithmic transformation to enhance contrast
rgb = np.log1p(rgb)

# Normalize the transformed image
rgb = rgb / rgb.max()

# Create folium map
m = Map(location=[31.7294, 6.0200], zoom_start=10, control_scale=True)

# Add true color image overlay
truecolour_overlay = ImageOverlay(
```

```
    image=rgb,
    bounds=[[31.485122001357873, 5.732553997080988], [31.973701421209405,
6.308794405143235]],
    opacity=1,
    interactive=True,
    cross_origin=False,
    zindex=1,
)
truecolour_overlay.add_to(m)

# Calculate mean and standard deviation of SWIR_diff
mean = np.nanmean(SWIR_diff)
std = np.nanstd(SWIR_diff)

# Standard deviation stretch
std_factor = 2  # Number of standard deviations
lower_bound = mean - std_factor * std
upper_bound = mean + std_factor * std

# Normalize SWIR_diff using standard deviation stretch
normalized_SWIR_diff = (SWIR_diff - lower_bound) / (upper_bound -
lower_bound)
normalized_SWIR_diff[normalized_SWIR_diff < 0] = 0
normalized_SWIR_diff[normalized_SWIR_diff > 1] = 1

# Apply colormap
cmap = plt.get_cmap('plasma')
SWIR_colored = cmap(normalized_SWIR_diff)

# Add SWIR_diff overlay
SWIR_overlay = ImageOverlay(
    image=SWIR_colored,
    bounds=[[31.485122001357873, 5.732553997080988], [31.973701421209405,
6.308794405143235]],
    opacity=1,
    interactive=True,
    cross_origin=False,
    zindex=2,
)
SWIR_overlay.add_to(m)

# Load GeoJSON file (already in EPSG:3857)
vector_point_path                                                    =
"C:/GIS_Course/Methane_Point_Detection/all_files/Data/known_point_source
s.geojson"
gdf = gpd.read_file(vector_point_path)

# Add vector point file overlay
```

```
GeoJson(gdf.to_json()).add_to(m)

# Add layer control
LayerControl().add_to(m)

# Add a click event to show latitude and longitude when clicking on the
map
m.add_child(LatLngPopup())

# Display the map
display(m)
```

Code 9: Code displaying the map

Code 10 allows the user to manually tag plumes for analysis using the latitude and longitude coordinates of plumes found by clicking on the map of code 9.

```
plume_coords = [
    (31.6887, 5.8102),  # Plume 1 (latitude, longitude)
    (31.7910, 5.8263),  # Plume 2 (latitude, longitude)
    (31.7978, 5.8341),  # Plume 4 (latitude, longitude)
    (31.9101, 6.0135),  # Plume 3 (latitude, longitude)
    (31.6389, 6.0025),  # Plume 4 (latitude, longitude)
]
```

Code 11 loads the reprojected SWIR_diff_4326 for the plume analysis. This is necessary as the original data did not support coordinates in latitude and longitude.

```
# Load TIFF file
tiff_file_path = r"C:\GIS_Course\Methane_Point_Detection\Sentinel-
2_Algeria_Methane\SWIR_diff_4326.tiff"
with rasterio.open(tiff_file_path) as tiff_file:
    raster_data = tiff_file.read(1)  # Read the first band
    nodata_value = tiff_file.nodata if tiff_file.nodata is not None else -
9999
    bounds = tiff_file.bounds
    transform = tiff_file.transform

# Mask nodata values
masked_data = np.ma.masked_equal(raster_data, nodata_value)
mean, std = np.nanmean(masked_data), np.nanstd(masked_data)
std_factor = 2
lower_bound, upper_bound = mean - std_factor * std, mean + std_factor * std
normalized_data = (masked_data - lower_bound) / (upper_bound - lower_bound)
normalized_data = np.clip(normalized_data, 0, 1)  # Clip to [0, 1]

# Apply colormap
cmap = plt.get_cmap('plasma')
rgb_data = (cmap(normalized_data)[:, :, :3] * 255).astype(np.uint8)

# Initialize the map
```

```
center_lat = (bounds.top + bounds.bottom) / 2
center_lon = (bounds.left + bounds.right) / 2
m = folium.Map(location=[center_lat, center_lon], zoom_start=11,
control_scale=True)

# Add SWIR_diff overlay
image_bounds = [[bounds.bottom, bounds.left], [bounds.top, bounds.right]]
swir_overlay = ImageOverlay(
    image=rgb_data,
    bounds=image_bounds,
    opacity=0.6,
    interactive=True,
    cross_origin=False,
    zindex=1,
)
swir_overlay.add_to(m)

# Mark plume locations
for i, (lat, lon) in enumerate(plume_coords):
    folium.CircleMarker(
        location=[lat, lon],
        radius=5,
        color="red",
        fill=True,
        fill_color="red",
        fill_opacity=1,
        popup=f"Plume {i + 1}",
    ).add_to(m)

# Calculate the mean value of the masked data
dataset_mean_value = masked_data.mean()

# Print the mean value
print(f"Mean value of the dataset: {dataset_mean_value}")
```

Finally the following code runs the plume analysis using the tagged location data as a reference. It draws blue polygons around each plume and provided their area, mean value and max value.

```
# Analyze plumes
def analyze_plume(masked_data, plume_coords, transform):
    plume_results = []
    labeled_array, _ = label(masked_data >
np.percentile(masked_data.compressed(), 65))  # Identify plumes
    for i, (lat, lon) in enumerate(plume_coords):
        try:
            row, col = rasterio.transform.rowcol(transform, lon, lat)
            row, col = int(row), int(col)
            plume_label = labeled_array[row, col]
            if plume_label == 0:
```

```
                plume_results.append({
                    "Plume": i + 1,
                    "Location (lat, lon)": (lat, lon),
                    "Status": "No plume detected",
                })
            else:
                plume_region = labeled_array == plume_label
                plume_values = masked_data[plume_region]
                plume_pixels = np.column_stack(np.where(plume_region))

                # Convert pixel coordinates to lat/lon
                plume_longitudes, plume_latitudes = rasterio.transform.xy(
                    transform, plume_pixels[:, 0], plume_pixels[:, 1]
                )
                points = np.array(list(zip(plume_latitudes,
plume_longitudes)))
                hull = ConvexHull(points)
                polygon_coordinates = [(points[vertex, 0], points[vertex,
1]) for vertex in hull.vertices]

                # Add polygon to the map
                folium.Polygon(
                    locations=polygon_coordinates,
                    color="blue",
                    weight=1,
                    fill=True,
                    fill_color="blue",
                    fill_opacity=0.4,
                    popup=f"Plume {i + 1} region",
                ).add_to(m)

                plume_results.append({
                    "Plume": i + 1,
                    "Location (lat, lon)": (lat, lon),
                    "Max Value": plume_values.max(),
                    "Mean Value": plume_values.mean(),
                    "Size (pixels)": plume_region.sum(),
                })
        except Exception as e:
            plume_results.append({
                "Plume": i + 1,
                "Location (lat, lon)": (lat, lon),
                "Status": f"Error: {e}",
            })
    return plume_results

# Perform analysis
plume_results = analyze_plume(masked_data, plume_coords, transform)
```

```
# Convert results to DataFrame
plume_df = pd.DataFrame([{
    "Plume": result["Plume"],
    "Location (lat, lon)": result["Location (lat, lon)"],
    "Max Value": result.get("Max Value", None),
    "Mean Value": result.get("Mean Value", None),
    "Size (pixels)": result.get("Size (pixels)", None),
    "Status": result.get("Status", "Plume detected"),
} for result in plume_results])

# Display DataFrame
print(plume_df)

# Add a layer control and click event
LayerControl().add_to(m)
m.add_child(LatLngPopup())

# Display the map
m
```

3.5 Expected Results and Demo

This section will illustrate the functionality of the tools by showcasing documented methane plumes and their corresponding depiction in the results. To ensure that this tool works as intended, a known emission in Algeria used by Varon et al. (2021) was imaged (fig.83).

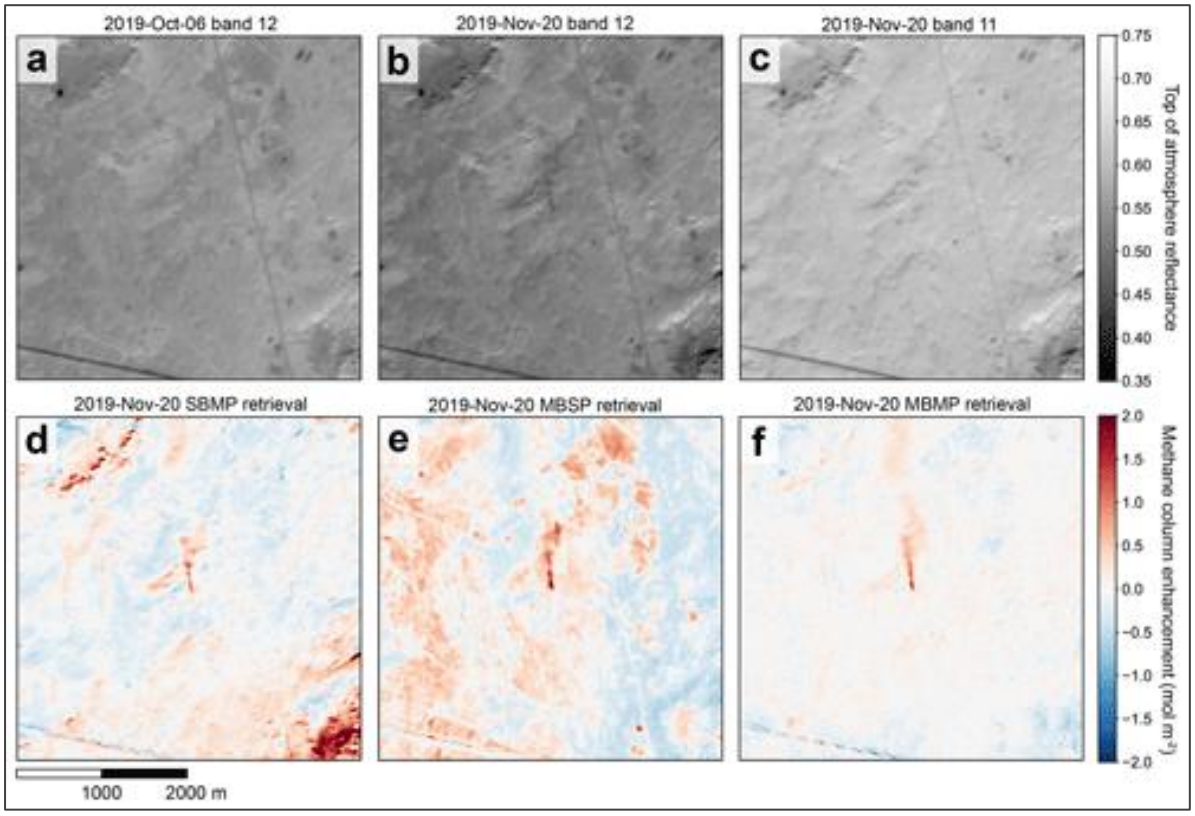


Figure 83: Sentinel 2 SWIR-2 (b), SWIR-1 (c) and Methane column retrieval for Multi Band Multi Pass method (e) from Varon et al. (2021)

This is the same plume shown in figure 84. Also shown are the 3 available layers. Clouds that appear in the SWIR_diff raster but are invisible in the normal colour satellite photo are methane plumes.

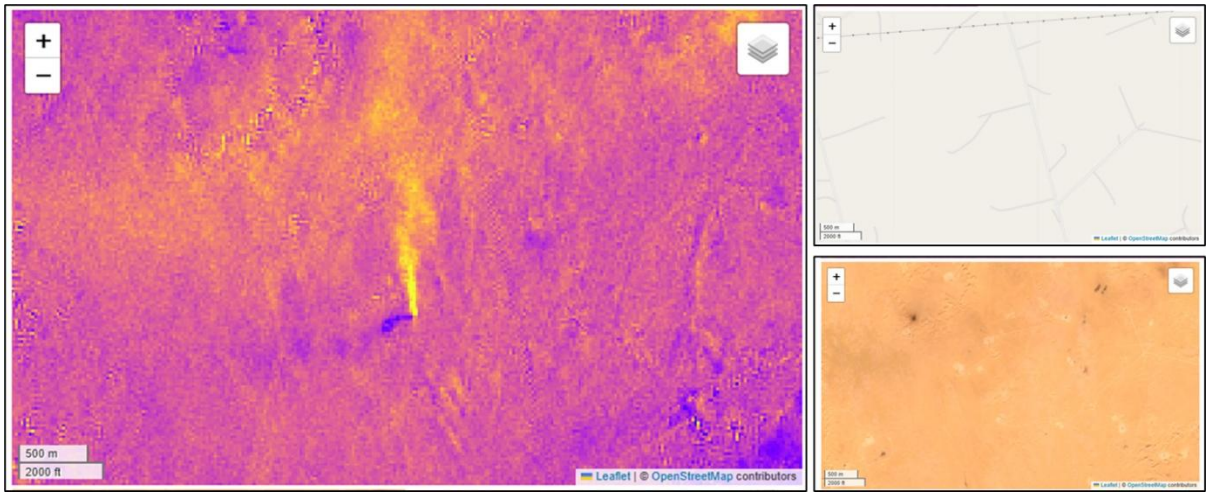


Figure 84: Methane column retrieval using S2-MBSP tool.

The 11th of January 2021 was chosen to demonstrate the tool's emission quantification abilities. Once plumes have been identified by manual tagging, the code draws polygons around the plumes and provides their area, mean value and max value. When compared to the mean dataset value, conclusions can be drawn about a given plume's magnitude.

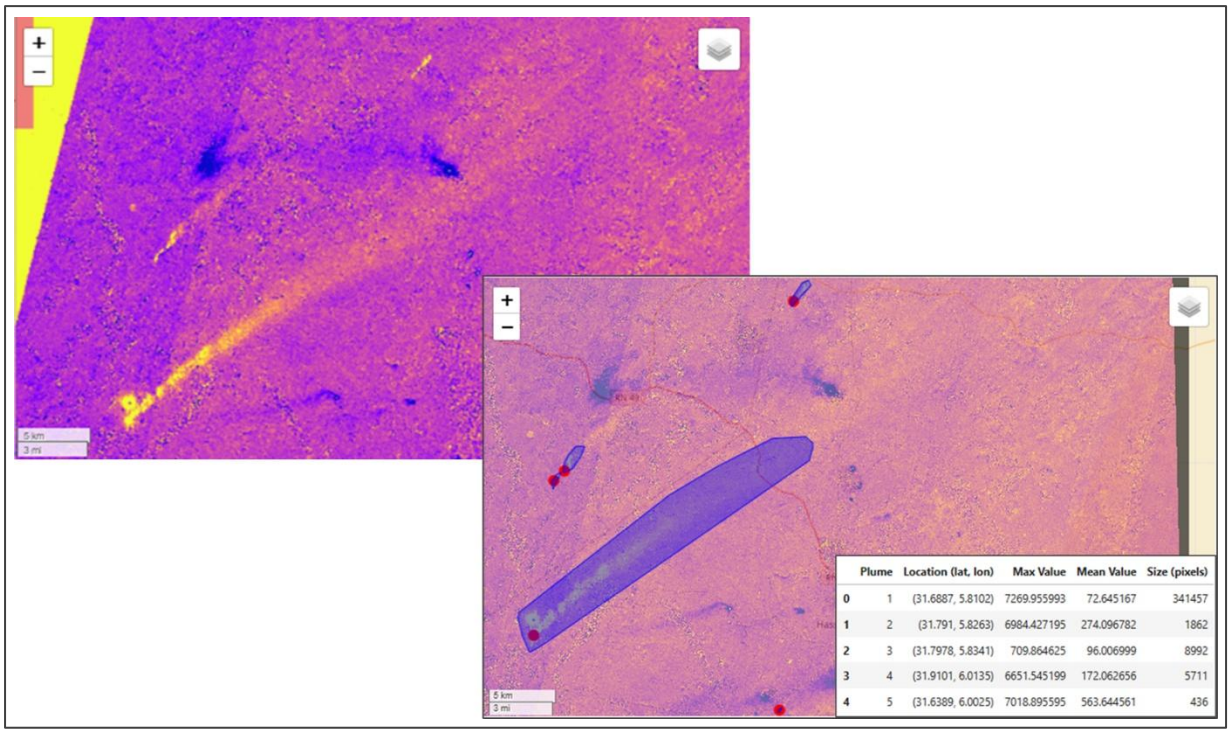


Figure 85: Plume images from the 11th of January 2021 showing plume measurement function.

3.6 Troubleshooting

Input errors have been covered in the tool notebooks, however there are errors unrelated to a user input which can cause problems. These are detailed below.

3.6.1 Remote disconnected error

```
Error: Remote disconnected
```

Or

```
OpenEoApiError: [500] Internal: Server error: KazooTimeoutError('Connection time-out') (ref: r-2405108830e742b59ef8ac2f28647fb5)
```

This can occur when there are issues with the Copernicus network. In the event that you see an error like this you can check page https://dataspace.copernicus.eu/news for any downtime messages and you can also contact the Copernicus dataspace team via the form at https://helpcenter.dataspace.copernicus.eu/hc/en-gb/requests/new

4. References

Abada, Z. and Bouharkat, M., 2018. Study of management strategy of energy resources in Algeria. *Energy Reports*, 4, pp.1-7. Available at: https://www.researchgate.net/publication/328663133_Study_of_management_strategy_of_energy_resources_in_Algeria [Accessed 18 Sep. 2024].

Dowd, E., Manning, A.J., Orth-Lashley, B., Girard, M., France, J., Fisher, R.E., Lowry, D., Lanoisellé, M., Pitt, J.R., Stanley, K.M. and O'Doherty, S., 2023. First validation of high-resolution satellite-derived methane emissions from an active gas leak in the UK. *EGUsphere*, 2023, pp.1-22.

Elkind, J., Blanton, E., Denier Van Der Gon, H., Kleinberg, R.L. and Leemhuis, A., 2020. Nowhere to hide: The implications of satellite-based methane detection for policy, industry and finance. *Columbia Center on Global Energy Policy*. Available at: https://energypolicy.columbia.edu

European Commission, 2023. EU announces €175m financial support to reduce methane emissions at COP28. *IP/23/6057*. Available at: https://ec.europa.eu/commission/presscorner/detail/en/IP_23_6057

Ferronato, N., Torretta, V., Ragazzi, M., & Rada, E.C. (2017). Waste mismanagement in developing countries: A case study of environmental contamination. UPB Sci. Bull, 79(2), 185-196.

Google Earth. (2024). Satellite view of Algerian petrochemical facility. Retrieved 10th of May 2024.

Integrated Methane Inversion (2024) *IMI processing steps*, Harvard University. Available at: https://imi.seas.harvard.edu/overview (Accessed: 27 November 2024).

International Energy Agency, 2022. Global Methane Tracker 2022. *IEA, Paris*. Available at: https://www.iea.org/reports/global-methane-tracker-2022

International Energy Agency, 2024. Global Methane Tracker 2024: Tracking pledges, targets, and action. Available at: https://www.iea.org/reports/global-methane-tracker-2024/tracking-pledges-targets-and-action [Accessed 17 Sep. 2024].

Jet Propulsion Laboratory. (2024). VISIONS: The EMIT Open Data Portal. Retrieved 10th of May 2024 from: https://tinyurl.com/2s4kkwe5

Jackson, R.B., Saunois, M., Bousquet, P., Canadell, J.G., Poulter, B., Stavert, A.R., Bergamaschi, P., Niwa, Y., Segers, A., and Tsuruta, A., 2020. Increasing anthropogenic methane emissions arise equally from agricultural and fossil fuel sources. *Environmental Research Letters*, 15(7), p.071002. Available at: https://doi.org/10.1088/1748-9326/ab9ed2 [Accessed 4 Jul. 2024].

Karimi, N., Ng, K.T.W. and Richter, A., 2021. Prediction of fugitive landfill gas hotspots using a random forest algorithm and Sentinel-2 data. *Sustainable Cities and Society*, 73, p.103097.

Liu, M., Van Der A, R., Van Weele, M., Eskes, H., Lu, X., Veefkind, P., De Laat, J., Kong, H., Wang, J., Sun, J. and Ding, J., 2021. A new divergence method to quantify methane emissions using observations of Sentinel-5P TROPOMI. *Geophysical Research Letters*, *48*(18), p.e2021GL094151.

Lorente, A., Borsdorff, T., Butz, A., Hasekamp, O., Aan De Brugh, J., Schneider, A., Wu, L., Hase, F., Kivi, R., Wunch, D. and Pollard, D.F., 2021. Methane retrieved from TROPOMI: Improvement of the data product and validation of the first 2 years of measurements. *Atmospheric Measurement Techniques*, *14*(1), pp.665-684.

Naus, S., Maasakkers, J.D., Gautam, R., Omara, M., Stikker, R., Veenstra, A.K., Nathan, B., Irakulis-Loitxate, I., Guanter, L., Pandey, S. and Girard, M., 2023. Assessing the relative importance of satellite-

detected methane superemitters in quantifying total emissions for oil and gas production areas in Algeria. *Environmental Science & Technology*, 57(48), pp.19545-19556.

Malley, C.S., Borgford-Parnell, N., Haeussling, S., Howard, I.C., Lefèvre, E.N. and Kuylenstierna, J.C., 2023. A roadmap to achieve the global methane pledge. *Environmental Research: Climate*, 2(1), p.011003.

McNabb, R. (2024). Reprojecting Rasters Using Rasterio [Blog post]. Retrieved 24th of April 2024 from https://iamdonovan.github.io/teaching/egm722/practicals/raster.html

Microsoft Copilot Designer. (2024). Cover artwork. Generated on 25th of November 2024

OpenEO. (2024). NDVI Timeseries. [Online]. Retrieved 22nd of April 2024, from: https://documentation.dataspace.copernicus.eu/notebook-samples/openeo/NDVI_Timeseries.html

Pandey, S., van Nistelrooij, M., Maasakkers, J.D., Sutar, P., Houweling, S., Varon, D.J., Tol, P., Gains, D., Worden, J., & Aben, I. (2023). Daily detection and quantification of methane leaks using Sentinel-3: a tiered satellite observation approach with Sentinel-2 and Sentinel-5p. Remote Sensing of Environment, 296, 113716.

Parker, R., Boesch, H., Cogan, A., Fraser, A., Feng, L., Palmer, P.I., Messerschmidt, J., Deutscher, N., Griffith, D.W., Notholt, J., & Wennberg, P.O. (2011). Methane observations from the Greenhouse Gases Observing SATellite: Comparison to ground based TCCON data and model calculations. Geophysical Research Letters, 38(15).

Rosenthal, J., 2023. In Reverse: Natural Gas and Politics in the Maghreb and Europe. *FPRI: Foreign Policy Research Institute*. United States of America. Available at: https://coilink.org/20.500.12592/bf2q0b [Accessed 29 September 2024].

Sentinel Hub. (2024) "About Sentinel-2 Data." Retrieved 5th of May 2024, from: https://docs.sentinel-hub.com/api/latest/data/sentinel-2-l2a/

Sentinel Hub. (2024) "About Sentinel-5P Data." Retrieved 23rd April 2024, from: https://docs.sentinel-hub.com/api/latest/data/sentinel-5p-l2/

Shen, L., Gautam, R., Omara, M., Zavala-Araiza, D., Maasakkers, J.D., Scarpelli, T.R., Lorente, A., Lyon, D., Sheng, J., Varon, D.J. and Nesser, H., 2022. Satellite quantification of oil and natural gas methane emissions in the US and Canada including contributions from individual basins. Atmospheric Chemistry and Physics, 22(17), pp.11203-11215.

Sherwin, E.D., Rutherford, J.S., Zhang, Z., Chen, Y., Wetherley, E.B., Yakovlev, P.V., Berman, E.S., Jones, B.B., Cusworth, D.H., Thorpe, A.K. and Ayasse, A.K., 2024. US oil and gas system emissions from nearly one million aerial site measurements. *Nature*, *627*(8003), pp.328-334.

Sonneveld, E. (2024). Obtaining a list of available Sentinel 2 datasets for a location. Copernicus Dataspace Forum. Retrieved 25th of April 2024 from https://tinyurl.com/4t2trevv

Talamali, S., Chaouchi, R. and Benayad, S., 2016. Sedimentological evolution of the Lower Series formation in the southern area of the Hassi R'Mel field, Saharan Platform, Algeria. *Arabian Journal of Geosciences*, 9, p.481. DOI: 10.1007/s12517-016-2506-7.

United Nations Development Programme, 2022. Country Programme Document for Algeria (2023-2027). DP/DCP/DZA/4. Available at: https://www.undp.org/sites/g/files/zskgke326/files/2023-03/CPD%20Algeria%202023-2027.pdf [Accessed 17 September 2024].

Varon, D.J., Jacob, D.J., McKeever, J., Jervis, D., Durak, B.O., Xia, Y. and Huang, Y., 2018. Quantifying methane point sources from fine-scale satellite observations of atmospheric methane plumes. *Atmospheric Measurement Techniques*, 11(10), pp.5673-5686.

Varon, D. J., Jervis, D., McKeever, J., Spence, I., Gains, D., & Jacob, D. J. (2021). High-frequency monitoring of anomalous methane point sources with multispectral Sentinel-2 satellite observations. Atmospheric Measurement Techniques, 14, 2771–2785.

Varon, D.J., Jacob, D.J., Sulprizio, M., Estrada, L.A., Downs, W.B., Shen, L., Hancock, S.E., Nesser, H., Qu, Z., Penn, E., Chen, Z., Lu, X., Lorente, A., Tewari, A. and Randles, C.A., 2022. Integrated Methane Inversion (IMI 1.0): a user-friendly, cloud-based facility for inferring high-resolution methane emissions from TROPOMI satellite observations. *Geoscientific Model Development*, 15, pp.5787–5805. https://doi.org/10.5194/gmd-15-5787-2022.

Varon, D.J., Jacob, D.J., Sulprizio, M., Estrada, L.A., Downs, W.B., Shen, L., Hancock, S.E., Nesser, H., Qu, Z., Penn, E., Chen, Z., Lu, X., Lorente, A., Tewari, A. and Randles, C.A., 2022. Integrated Methane Inversion (IMI 1.0): a user-friendly, cloud-based facility for inferring high-resolution methane emissions from TROPOMI satellite observations. *Geoscientific Model Development*, 15, pp.5787–5805. https://doi.org/10.5194/gmd-15-5787-2022.

Vigano, I., Van Weelden, H., Holzinger, R., Keppler, F., McLeod, A., & Röckmann, T. (2008). Effect of UV radiation and temperature on the emission of methane from plant biomass and structural components. Biogeosciences, 5(3), 937-947.

Wang, H., Fan, X., Jian, H. and Yan, F., 2024. Exploiting the Matched Filter to Improve the Detection of Methane Plumes with Sentinel-2 Data. *Remote Sensing*, *16*(6), p.1023.