Visualizing the NOAA CarbonTracker-CH4 3-D Mole Fraction (Pressure Corrected) Dataset

Matthew Ziminski, Graduate CS Student, University of Illinois Chicago
UIN: 656047645

Project Client: Dr. Youmi Oh, NOAA Global Monitoring Laboratory

Project Advisor: Dr. G. Elisabeta Marai, University of Illinois Chicago
2nd Committee Member: Dr. Stefany Sit, University of Illinois Chicago

Abstract

Methane the main component of natural gas is also a potent greenhouse gas that requires robust monitoring, so that scientists, educators, and the general public can get better informed of its effect(s) on local/regional air quality and current short term mitigations for climate change. However, current data visualizations for methane, and other scientific data, are either static 2-D graphs/plots or animated through using videos/gifs. These data visualization approaches can be limiting in that they lack interactiveness for an end user to gain exploratory insights by comparing and contrasting different areas of a dataset. This report thereby showcases how to build a functional and interactive visualization as the solution to the previously mentioned limitation.

Keywords: Interactive Scientific Data Visualization, Climate Data Visualization, Methane Visualization.

Table of Contents

1 Introduction

Being able to visualize and analyze the spatial and temporal distribution of methane (CH_4), the main component of natural gas, and the sources it comes from is important in understanding how to reduce the effects of climate change. According to the recent IPCC AR6 report, methane has a 28-34X heating capacity than carbon dioxide over a 100-year time horizon (Masson-Delmotte et al. 2021). Due to its short lifetime (~10 years), CH_4 is a good target for climate change mitigation. With a high heating capacity and a short lifetime, CH_4 is a potent greenhouse gas. To make matters worse, CH_4 ultimately oxidized to CO_2, which leads to the formation of tropospheric ozone, an air quality hazard, in polluted areas, (UNEP Global methane assessment, 2021). Therefore, these characteristics of CH_4 can lead to lasting impacts on the world and lead to the growing need for an Interactive visualization tool to better understand and quantify the spatial and temporal changes in atmospheric CH_4.

CarbonTracker-CH4 is NOAA's global atmospheric inversion system to produce quantitative estimates of atmospheric CH_4 emissions that are consistent with observed patterns of CH_4 in the atmosphere (https://gml.noaa.gov/ccgg/carbontracker-ch4/index.html). The premier version of CarbonTracker-CH4 was published in 2014 to estimate emissions of CH_4 from 2000 to 2010 using atmospheric measurements of CH_4 (Bruhwiler et al. 2014). For the second release planned in Summer 2023, the inversion system is further improved by jointly assimilating measurements of CH_4 and its isotopes, optimizing fluxes at a grid-scale, incorporating spatially- and temporally- resolved source signature of CH_4 isotopes, and extending the emission estimation to 2021. The 3-D visualization that this project produces will be available under the visualization tab on the CarbonTracker-CH4 website (https://gml.noaa.gov/ccgg/carbontracker-ch4/assimilated_ch4.html), and the interactive visualization will be a very useful tool for scientists, educators, and the general public to understand global atmospheric CH_4.

2 Related Work

There doesn't exist an interactive visualization tool similar to the one this report will be presenting. Currently, scientific visualizations of CH_4 consist of primarily videos/gifs and/or static plots (i.e., histograms, bar charts, line plots, etc.). These visualizations lack any interactiveness, or user input. In addition, users can learn insights about a dataset but are restricted to only viewing the dataset as it was intended to be viewed within the video/gif. Below are some examples that show support for the need of creating an interactive visualization tool for the CH_4 dataset.

(1) CarbonTracker CT2022 [4]

This resource serves as a support for the need for interactive visualizations for NOAA. The primary visualizations for this dataset are predominantly static plots (maps and/or tables). A gif is pictured showing an animation of the data in the shape of a sphere, but it's not interactive, so users have to download the data themselves to create their visualizations.

(2) CarbonTracker-CH4 [6]

This resource serves as additional support for the need for interactive visualizations for NOAA. There is an animated gif on the homepage and further static plots on subsequent sub-pages, but none of these are interactive. As mentioned in point 1, users are required to download the data themselves before they can do any visualization and/or analysis.

(3) Science on a Sphere (SOS) catalog [3]

SOS is a 6-foot diameter sphere that can project earth-based data onto it. The CH_4 dataset currently isn't in the catalog, but work is being done to add it. However, this sphere is constrained to being in predetermined locations (i.e., museums). It's also limited with what type of data it can display, usually 2-D datasets that can get mapped onto its surface. 3-D datasets are therefore unusable on SOS. So, a web-based interactive visualization would allow anyone to interact with the data from anywhere.

(4) Sources of Methane [1]

This resource has some good visualizations, but they are in the form of videos/gifs. This does not give the user liberty to navigate through the visualization, thereby severely limiting the way it is interpreted. This cannot be used to address the client's requirements.

(5) New 3-D View of Methane Tracks Sources and Movement around the Globe [7]

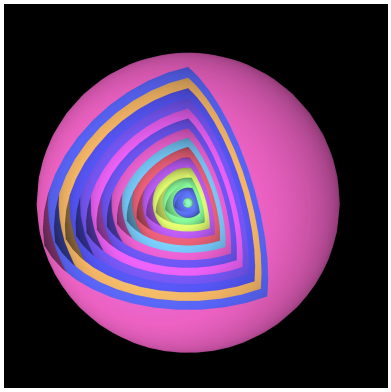
The visualizations found here are identical to some visualizations described in point (4). These visualizations, however, have been updated with new data, but they still face the same issues mentioned in the previous points, i.e. visualizations are in the form of videos/gifs.

3 Methods

3.1 Architecture

While doing initial research on how to create this interactive visualization tool I had a hard time wrapping my head around how longitude/latitude points in a 3-D cube could be used to plot data. I jumped from thinking of using voxels to using multi-nested shells in Three.js, but these initial prototypes were plagued with issues.

The ThreeJS Globe Visualization library allowed the developer to use multiple different layers together but didn't allow for the same layer to be used multiple times. Therefore, the tile layer looked promising, as there was an altitude attribute, but was hindered by the aforementioned limitation. Due to these limitations, I decided to look into Three.js, but even Three.js had its limitations when combined with Angular, a framework used to design/develop websites modularly, that I wasn't able to solve. Therefore I did more research into other ideas that allowed for quick prototyping, and eventually encountered a Three.js example of nested spherical shells (image on the next page,



https://threejs.org/examples/#webgl_clipping_intersection). I wanted to integrate this example within Angular but ran into Three.js Typescript issues, so I dropped using Angular and Three.js in favor of using Plotly's volume plot.

The reason for using Plotly was that it allows for faster prototyping and the documentation/community forums were very helpful. One user in particular @empet, created a 3d sphere visualization that I was able to reverse engineer using Plotly Python (https://chart-studio.plotly.com/~empet/14795/wind-speed-forecast/#/). Extracting the Python/Javascript code wasn't too helpful, as the majority of the data traces and the layout were hardcoded into JSON. So I had to figure out how she created the data from scratch. I eventually found @crgnam from the Plotly community who was able to successfully plot a texture onto a sphere surface (https://community.plotly.com/t/applying-full-color-image-texture-to-create-an-interactive-earth-globe/60166). Thereby allowing me to produce the sphere plot; more on the actual implementation later.

This project was written primarily in Python 3.11 and used PipEnv for the virtual Python environment. Some HTML/CSS/Javascript code was written as well, but minimal. The project started with only using plotly python to make the plots, but later morphed into a Flask app, and then into a Dask app. All three subprojects are present in the GitHub repository linked below in the documentation section.

3.1.1 Py Plotly to HTML

This subproject's goal was to create a Py Plotly Script that can easily and quickly generate Plotly plots using a dataset. Py Plotly can be run in a Jupyter Notebook or the browser, and its HTML file can be saved. A downside to using Py Plotly is that when using a large enough dataset (i.e., 5+ MB) the whole dataset gets hardcoded into the HTML file when saving. To prevent this, a backend will be necessary to serve the data to the Plotly plots.

3.1.2 Flask+Plotly

Plotly can be combined with Python Flask (a web server library) to create more dynamic plots or create plots using chunks from a larger dataset. The caveat to this approach is that you need to understand both Python and HTML/CSS/Javascript. Flask also requires static files to be saved inside a folder named "static", then HTML templates are to be stored inside a folder named "template." However, Flask+Plotly allows more flexibility and customizability in full-stack development, than Plotly Dash.

3.1.3 Plotly Dash

Plotly Dash is a great all-in-one solution for anyone who wants to code in one language. HTML/CSS/JS code can still be added to the project, but they have to be included inside of a folder named "assets" as that's the folder name Dash expects. Plotly Dash uses Flask as the backend and Python as a wrapper for ReactJS-written HTML objects/elements, including Plotly Objects. It also makes communication between server/client simple with callbacks which are essentially Python function decorators. Callbacks can be powerful tools for creating interactions between plots. Dash makes prototyping easier in some cases, but when dealing with large

datasets constant client/server communications become unfavorable. So client-side callbacks were created, but these too have their limitations. Therefore, for some cases (mostly dealing with large datasets) a flask+Plotly approach would be recommended.

3.2 Data Sources/Preprocessing

This project used two datasets, the NOAA-derived CH_4 Mole Fraction Pressure Corrected Dataset and the Global Self-consistent, Hierarchical, High-resolution Geography Database (GSHHG) for the coastlines (base map for each plot). No data preprocessing was necessary for the NOAA dataset, but heavy data preprocessing was necessary for the GSHHG dataset. The GSHHG dataset is a shapefile (a collection of geometries: points, lines, polygons, multi polygon, etc.) and all the (longitude, latitude) coordinates of all the polygons/multi polygons had to be extracted to be used by Plotly. Lastly, the Python Library Xarray was used to load/filter the NOAA dataset for this project, GeoPandas was used to load the GSHHG dataset, and NumPy arrays/vectorized functions were very beneficial in quickly transforming the data from one format to another.

3.3 Visual Encodings

I chose Volumetric and Spherical Visual Encodings for this project. The volumetric encoding not only allows for the 3-D CH_4 dataset to be rendered in 4-D (3-D + Color) but also allows an end user to see the distribution of CH_4 within a volumetric container. The spheric encoding was used to allow different (longitude, latitude) slices from the volumetric encoding to be viewed onto a spheric surface. Thereby allowing the end user to see the same data but in a different median. I wanted to create a 5-D sphere encoding, but limitations of Plotly and/or computer power prevented me from pursuing this any further than an initial test. The idea for this 5-D encoding was to have a 3-D sphere, with the 3-D Mole Fraction data mapped onto the sphere so that the height would extrude away from the sphere's surface, and color produced the 5th dimension.

3.3.1 Volumetric Plot

This plot consists of 3 parts: the coastline base map trace, the 4-D volume trace, and the longitude/latitude plane which was implemented using the 3-D line Mesh trace. The data for the volume trace had to be converted from a Xarray DataArray to a 3-D XYZ Numpy mgrid. I used the turbo color scale because it was one of the color scales that looked the best on the Plotly-Dark theme background. The labeling for these plots was a bit tricky, as the pressure attribute is indexed from 0…101, but the hPa goes from 1000 at the surface to 0 at the top of the atmosphere. Therefore I had to first flip the Numpy Z array then multiply everything by 10 and lastly use Numpy flatten using the 'F' key. 'F' stands for the Fortran method of column-major sort. For the layout, I disabled everything for the x/y/z axis (background, lines, grids, etc.), and modified the camera position.

3.3.2 Spherical Plot

Both the surface data and the coastline data had to be converted from cartesian coordinates to spherical coordinates. They then had to be converted back from spherical to cartesian for the surface trace's hover label. The layout follows the same as the volumetric plot,

however, the volumetric trace was replaced with a surface trace, and the camera's point of view wasn't modified.

3.4 Interactions

The main interactions for this project consisted of using a date picker to select an available date (month/day/year) in the dataset. Another interaction was to use a slider at intervals from 0 to 100 by steps of 10. Having 100 was too crowded, so 10 was a good median. No other interactions were able to be implemented due to time.

Colorblindness is a factor that affects how some people can view the world, so having an interaction that allows the user to pick a color scale best suited for them would make this project more accessible.

Another planned implemented interaction was to use a range slider to select a fixed range, then have a slider of that range to fine-tune the desired hPa level, allowing for more levels to be represented. However, I did not have enough time to implement this feature.

3.5 Documentation

Currently unfinished but will be finalized by the end of the semester.

Github: https://github.com/mziminski/NOAA-3D-CH4-IVT

4 Results

The outcome can be seen in the case study below. To add, this project has been completed and is functional, though if I had more time more functionality/features could've been added. I'll describe what other features I could've added in the discussion section.

4.1 Case Study

For this case study, I looked at the 07/02/2021 mole fraction pressure corrected dataset at 1000 hPa, 700 hPa, 400 hPa, and 200 hPa slices. I found this date interesting due to the unique shape that's found within Asia. The 1000 hPa slice surprised me, because of the structure's wide base covering most of Asia. Yet, its sphere plot counterpart doesn't show this structure too well. Therefore, if the sphere plot were to be on its own a user might not be able to see the full picture of what's going on in the region. To add, this analysis is a bit lacking because I'm not a climate scientist, so I can't say for certain if this structure has significance. All I can assume is that the structure conveys the flow of methane for this period of 1 day.

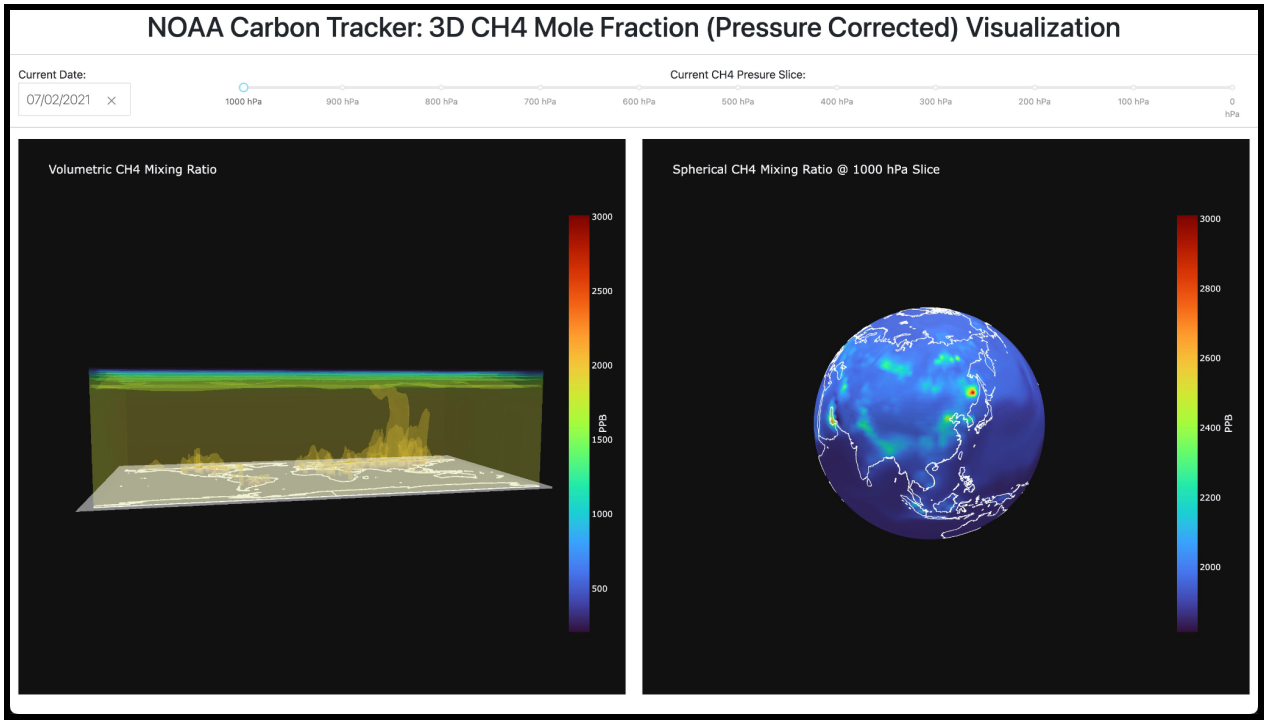


Figure 1: This figure shows the respective volumetric/spheric plots for 07/02/2021 at 1000 hPa.

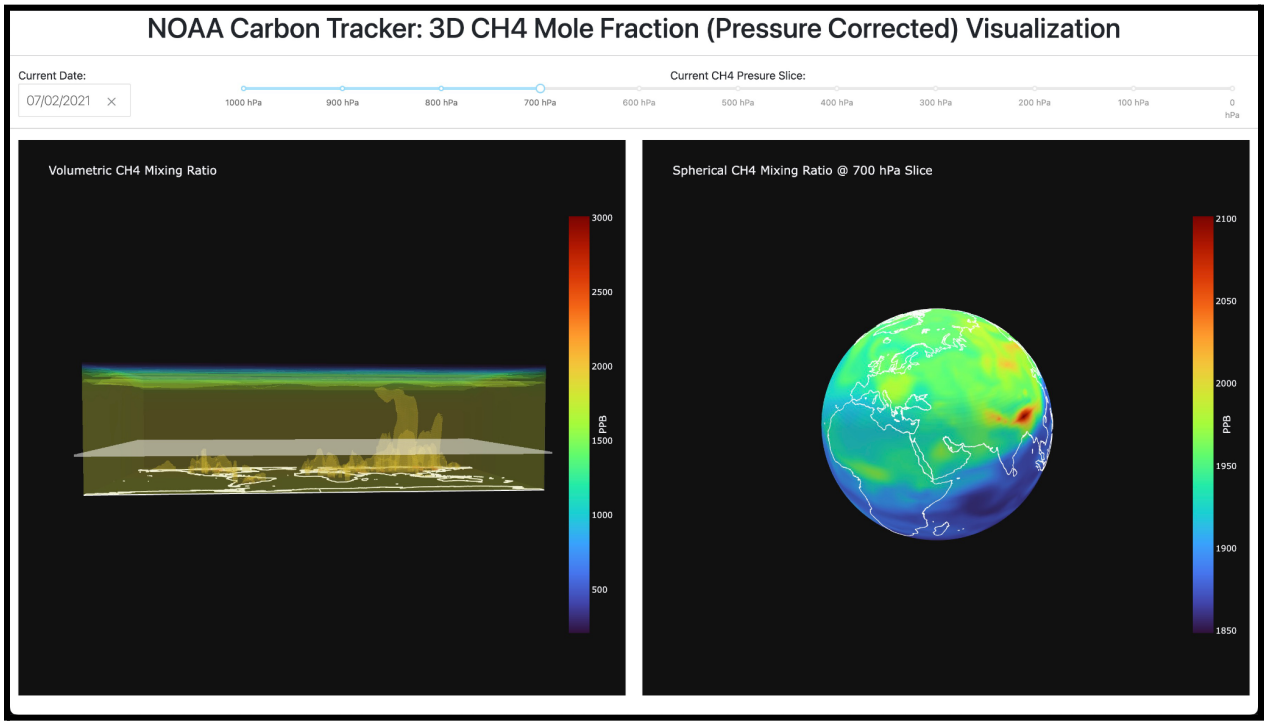


Figure 2: This figure shows the respective volumetric/spheric plots for 07/02/2021 at 700 hPa.

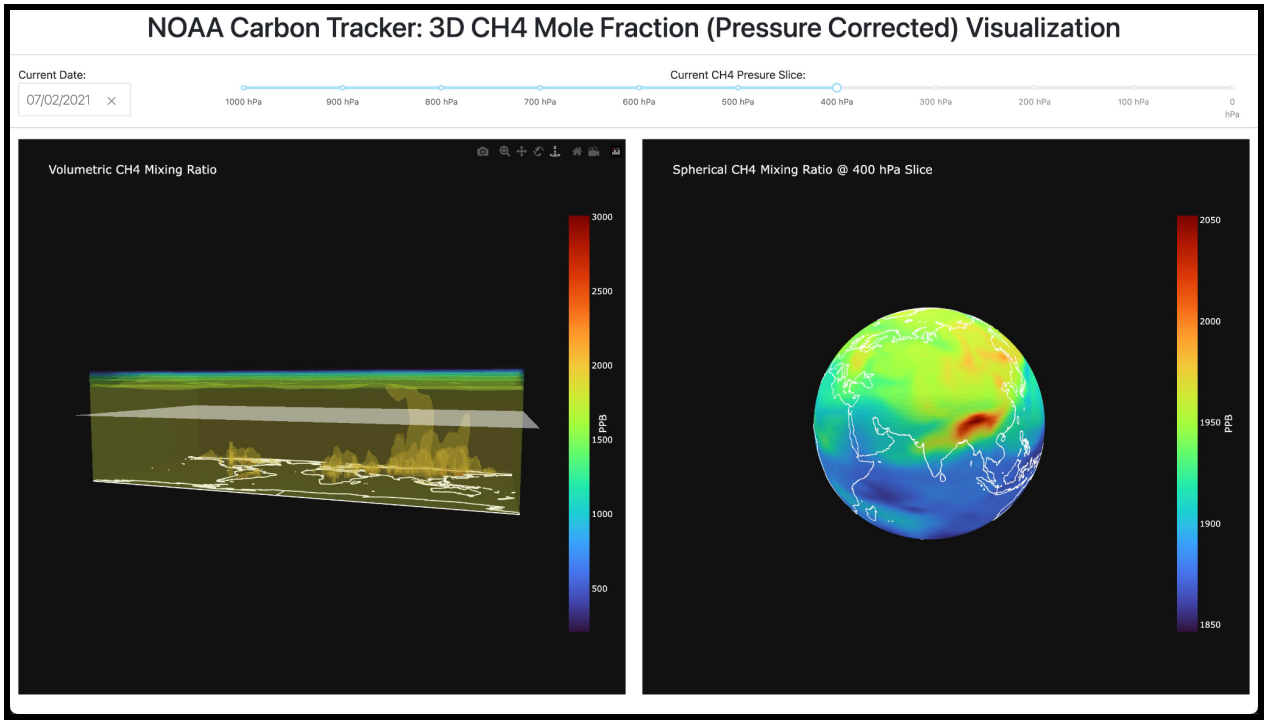


Figure 3: This figure shows the respective volumetric/spheric plots for 07/02/2021 at 400 hPa.

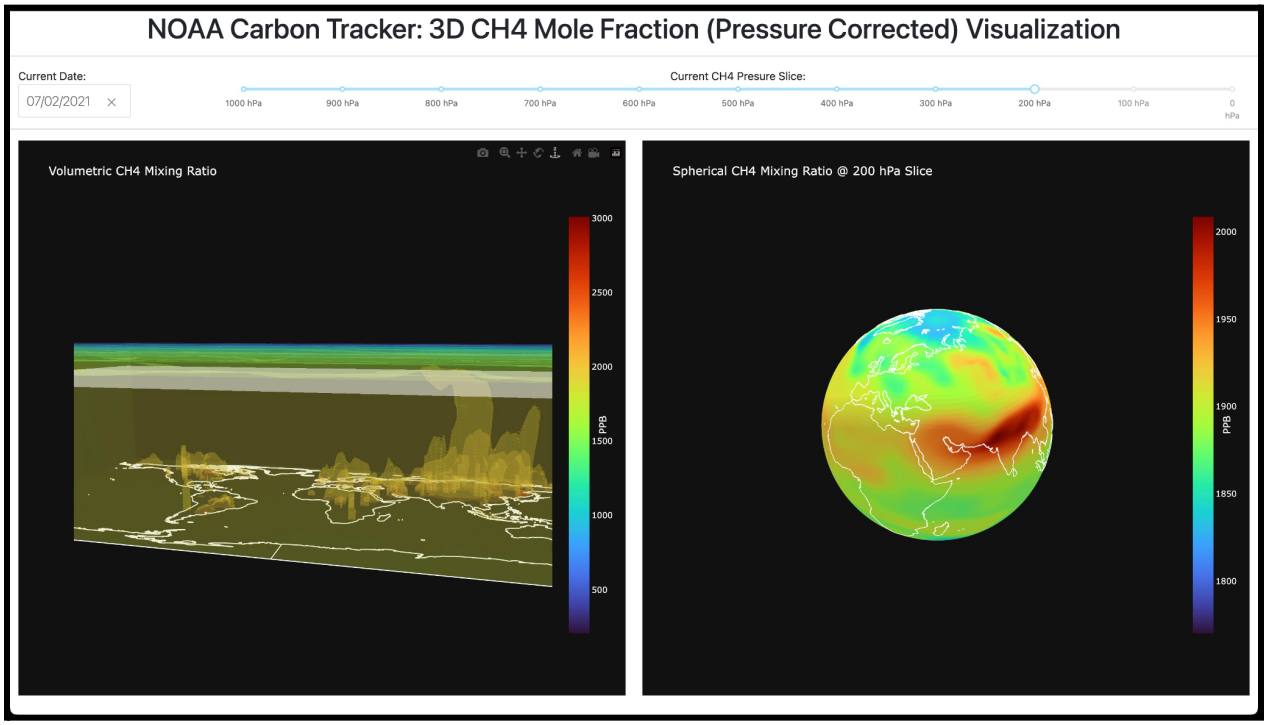


Figure 4: This figure shows the respective volumetric/spheric plots for 07/02/2021 at 200 hPa.

4.2 Client Feedback

"As a client of the 4-D visualization of CarbonTracker-CH4, I am very satisfied with Matt's results. He successfully converted the 4-dimensional CH4 mole fraction dataset into sphere and volume plots. We have implemented Matt's plots into the newly released CarbonTracker-CH4 website, which will be available in the near future at https://gml.noaa.gov/ccgg/carbontracker-ch4/. Matt's project will be referenced, and I appreciate his effort over the past several months in working hard on this project." –Dr. Youmi Oh

5 Discussion

The best feature of my solution is that it's interactive. Users can see both a volumetric and spherical representation of the dataset. This approach is far better than the voxel/shell idea I initially proposed since the user can see any underlying structures within the volume plot. This might've been harder to accomplish using my initial voxel/shell idea.

This project also has the potential to scale and have a proper backend serving the front end, but more work on the front end is needed. Currently, due to time constraints, I wasn't able to optimize/reduce the rendering times of the plots between interactions.

The total loading/rendering time from a page refresh to all plots being fully rendered is ~28 seconds. The breakdown is as follows: 7 seconds for the client to receive the sphere data, then 1-2 seconds for the data to be rendered; 16 seconds for the volumetric data to be received, then another 12 seconds for the plot to be fully rendered. To calculate these times a stopwatch was used alongside the network tab in the Chome Developer tools window pane. The stopwatch was used for an overall time, while the network tab was used for data load times.

The solution would be to have Dash or Flask store/cache the data to the browser, then have the browser handle the data processing/transformation for the respective plots. Rendering can be eliminated by using Plotly update, a function that can update specific traces without rendering the whole figure.

This project can also be generalized in the sense that any 3d datacube can be uploaded, then plotted and interacted with. But slight modifications to the codebase will have to be made to allow for this generalization.

However, due to data security concerns, my dashboard might not be able to be used as a dynamic means of visualizing data. My GitHub could be linked so users could download the dashboard themselves and interact with the data. But a very limited dashboard with hard-coded values can be possible to be published for the NOAA Carbontracker-CH4 Website.

6 Conclusion

In conclusion, I was able to successfully create an interactive and functional visualization tool for the NOAA CarbonTracker-CH4 3-D Mole Fraction (Pressure Corrected) Dataset.

Acknowledgments

Thank you Dr. Youmi Oh for being an amazing Mentor throughout this project, for allowing me to participate in such an opportunity, and for your very useful feedback and suggestions.

Thank you Dr. Stefany Sit for Introducing me to Dr. Youmi Oh in the fall 2022 semester, without this introduction I wouldn't have been given this unique opportunity, and thank you for your helpful suggestions in editing this report.

Thank you Dr. G. Elisabeta Marai for being my advisor throughout this project, for lecturing CS 529 as I used the knowledge gained from this course within this project, and for your helpful editorial suggestions.

Thank you Dr. Fabio Miranda for lecturing CS 524 Big Data Analysis and Visualization this semester (Spring 2023) as the material I learned from this course was integral in advancing this project.

Thank you Kenneth Schuldt for implementing my visualizations into the NOAA CarbonTracker-CH4 Website.

Thank you, Neil Chawla and Shreya Raj Kati for being my teammates in CS 529, and for contributing to our CS 529 project's workbook, as it was an integral part of my brainstorming process and writing process for this report.

Thank you Electronic Visualization Laboratory (EVL) at the University of Illinois Chicago (UIC) for allowing me to attend the Weekly Tech Meetings, and for allowing me to present this completed project during the last Tech Meeting of the Spring 2023 Semester.

Thank you Dr. Gavin McNicol for your feedback on this report.

Thank you to all the contributors responsible for creating Angular CLI, Plotly (Python, Js, etc.), D3.js, Three.js, and the ThreeJS Globe Visualization library.

References

[1] Blumberg, S. (2020) *New 3D view of methane tracks sources and movement around the Globe*, *NASA*. NASA's Earth Science News Team. Available at: https://www.nasa.gov/feature/goddard/2020/new-3d-view-of-methane-tracks-sources-and-movement-around-the-globe/.

[2] Bruhwiler, L., Dlugokencky, E., Masarie, K., Ishizawa, M., Andrews, A., Miller, J., Sweeney, C., Tans, P. and Worthy, D., 2014. CarbonTracker-CH 4: an assimilation system for estimating emissions of atmospheric methane. Atmospheric Chemistry and Physics, 14(16), pp.8269-8293.

[3] *Dataset catalog* (no date) *Science On a Sphere*. National Oceanic Atmospheric Administration (NOAA). Available at: https://sos.noaa.gov/catalog/datasets/.

[4] Jacobson, A. R. et al. (2023) "CarbonTracker CT2022." NOAA Global Monitoring Laboratory. doi: 10.25925/Z1GJ-3254.

[5] Masson-Delmotte, V., Zhai, P., Pirani, A., Connors, S.L., Péan, C., Berger, S., Caud, N., Chen, Y., Goldfarb, L., Gomis, M.I. and Huang, M., 2021. Climate change 2021: the physical science basis. Contribution of working group I to the sixth assessment report of the intergovernmental panel on climate change, 2.

[6] Peters et al, 2007, "An atmospheric perspective on North American carbon dioxide exchange: CarbonTracker", PNAS, November 27, 2007, vol. 104, no. 48, 18925-18930

[7] Starr, C. (2020) *NASA Scientific Visualization Studio - Sources of Methane*, *NASA's Scientific Visualization Studio*. NASA. Available at: https://svs.gsfc.nasa.gov/4799#section_credits.

[8] United Nations Environment Programme and Climate and Clean Air Coalition (2021). Global Methane Assessment: Benefits and Costs of Mitigating Methane Emissions. Nairobi: United Nations Environment Programme.

[9] Wessel, P. and Smith, W.H.F. (2016) *Shoreline/coastline databases: NCEI*, *Shoreline/Coastline Databases* | *NCEI*. U.S. Department of Commerce. Available at: https://www.ngdc.noaa.gov/mgg/shorelines/gshhs.html (Accessed: April 27, 2023).