EXPLORANDO BIBLIOTECAS PYTHON PARA VISUALIZAÇÃO DE CAMPOS METEOROLÓGICOS DO THE BRAZILIAN DEVELOPMENTS ON THE REGIONAL ATMOSPHERIC MODELING SYSTEM (BRAMS)

Lucas Adati de Paula

Relatório de Iniciação Científica do programa PIBIC, orientado pelo Dr. Eugenio Sper de Almeida.

INPE

São José dos Campos - SP

2023

AGRADECIMENTOS

Ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), pelo auxílio financeiro com uma bolsa de Iniciação Científica dentro do Programa PIBIC/CNPq/INPE (Processo 163139/2022-9).

Ao Instituto Nacional de Pesquisas Espaciais (INPE) pela oportunidade de estudos e disponibilização dos dados georreferenciados.

Ao meu orientador, que durante o processo da iniciação científica ter me guiado, auxiliado e oferecido diversas oportunidades de aprendizado.

RESUMO

O resultado de uma previsão numérica de tempo e clima, gerado por um modelo atmosférico, é um conjunto de matrizes multidimensionais. O processo de geração de previsão de tempo consiste na execução de um *workflow* que coleta informações meteorológicas, altera sua resolução espacial e executa os modelos de previsão numérica. O *The Brazilian developments on the Regional Atmospheric Modeling System* (BRAMS) é um modelo numérico de previsão do tempo projetado para simular circulações atmosféricas, sendo suas saídas disponibilizadas no formato GrADS. Atualmente a geração de figuras e gráficos com informações meteorológicas utiliza o GrADS e *shell scripts*. O objetivo deste trabalho é simplificar a visualização dos campos meteorológicos do modelo atmosférico BRAMS em seus diferentes níveis atmosféricos. Utilizou-se a linguagem *Python,* e as bibliotecas *Xgrads,* Xarray*, Numpy* e *Metpy* como base. Foram analisadas, definidas e adaptadas bibliotecas para manipulação e visualização de dados atmosféricos. O ambiente de trabalho para a leitura desses dados foi preparado e automatizado. Para a geração de mapas com os dados dos campos meteorologicos foram exploradas as seguintes bibliotecas Python: *Cartopy /Matplotlib*, *Plotly, Bokeh e Folium /Matplotlib/Geojsoncontour.* A biblioteca *Folium* em conjunto com o *Matplotlib* (v.3.3.2) e *Geojsoncontour* foi a única que possibilitou gerar mapas com os dados dos campos meteorológicos. O framework Streamlit (v.1.16.0) permitiu a criação de uma interface web simples para disponibilizar os mapas gerados. Como resultados, foi possível disponibilizar as plotagens dos mapas em uma aplicação web interativa e simplificada para o usuário. Desta forma, a manipulação e visualização dos mapas foi simplificada ao usuário final, sem a necessidade de ter prévio conhecimento de alguma linguagem de programação.

Palavras-chave: Modelos atmosféricos. BRAMS. Visualização. Linguagem Python. Bibliotecas gráficas

ABSTRACT

The result of a numerical weather and climate forecast, generated by an atmospheric model, is a set of multidimensional matrices. The weather forecast generation process consists of executing a workflow that collects meteorological information, changes its spatial resolution and runs numerical forecast models. The Brazilian developments on the Regional Atmospheric Modeling System (BRAMS) is a numerical weather forecast model designed to simulate atmospheric circulations, with its outputs available in GrADS format. Currently, the generation of figures and graphs with meteorological information uses GrADS and shell scripts. The objective of this work is to simplify the visualization of the meteorological fields of the BRAMS atmospheric model at its different atmospheric levels. We used Python language, and the libraries Xgrads, Xarray, Numpy and Metpy as a base. Libraries for manipulating and visualizing atmospheric data were analyzed, defined and adapted. The work environment for reading this data was prepared and automated. To generate maps with data from meteorological fields, the following Python libraries were explored: Cartopy/Matplotlib, Plotly, Bokeh and Folium /Matplotlib/Geojsoncontour. The Folium library together with Matplotlib and Geojsoncontour was the only one that made it possible to generate maps with data from meteorological fields. The Streamlit framework allowed the creation of a simple web interface to make the generated maps available. As a result, it was possible to make map plots available in an interactive and simplified web application for the user. In this way, the manipulation and visualization of maps was simplified for the end user, without the need to have prior knowledge of any programming language.

Keywords: Atmospheric models. BRAMS. Visualization. Python language. Graphic libraries

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

BRAMS – Brazilian developments on the Regional Atmospheric Modelling System

CF – Climate and Forecasting

CPTEC – Centro de Previsão de Tempo e Estudos Climáticos

CRS – Coordinate Reference System

GrADS – Grid Analysis and Display System

INPE – Instituto Nacional de Pesquisas Espaciais

NetCDF – Network Common Data Form

SUMÁRIO

1. INTRODUÇÃO

A previsão de tempo consiste, segundo Corte-Real (2015), na determinação dos estados futuros da atmosfera, partindo de um estado já conhecido em um determinado momento. Desta forma, a previsão do tempo é um conjunto de informações que expõem o estado da atmosfera em um determinado período. O resultado da saída de modelos numéricos constitui-se de um conjunto de matrizes multidimensionais. Essas matrizes podem ser vistas como colunas verticais sobre pontos de grade (Lynch, 2008).

O procedimento de geração da previsão do tempo constitui-se em realizar a coleta das informações meteorológicas, alteração da resolução espacial e execução dos modelos de previsão numérica (Almeida e Bauer, 2012). Esse *workflow* é executado por supercomputadores ou *Clusters*. O resultado contém mapas e gráficos meteorológicos, gerado a partir das saídas dos modelos para análise por meteorologistas e visualização em páginas Web.

Um dos componentes de um workflow meteorológico gera a saída do *The Brazilian developments on the Regional Atmospheric Modeling System* (BRAMS), no formato Grid Analysis and Display System (GrADS). A manipulação dessa saída possibilita a geração de mapas e gráficos. No decorrer do *workflow*, a interação entre os componentes e arquivos do workflow impacta no desempenho.

A manipulação dos componentes do workflow requer conhecimento programação prévio de *shell script* e GrADS. Neste contexto faz-se necessário a busca por alternativas que permitam a execução do *workflow* meteorológico de maneira simples e com desempenho.

O objetivo geral deste projeto consiste na visualização dos campos de atmosféricos do modelo atmosférico BRAMS em seus diferentes níveis atmosféricos utilizando a linguagem *Python* e suas bibliotecas. Os objetivos específicos são realizar a análise e avaliação das bibliotecas para a manipulação e visualização de dados atmosféricos, definição/adaptação da biblioteca adequada e a preparação/automatização do ambiente de trabalho para a leitura destes dados.

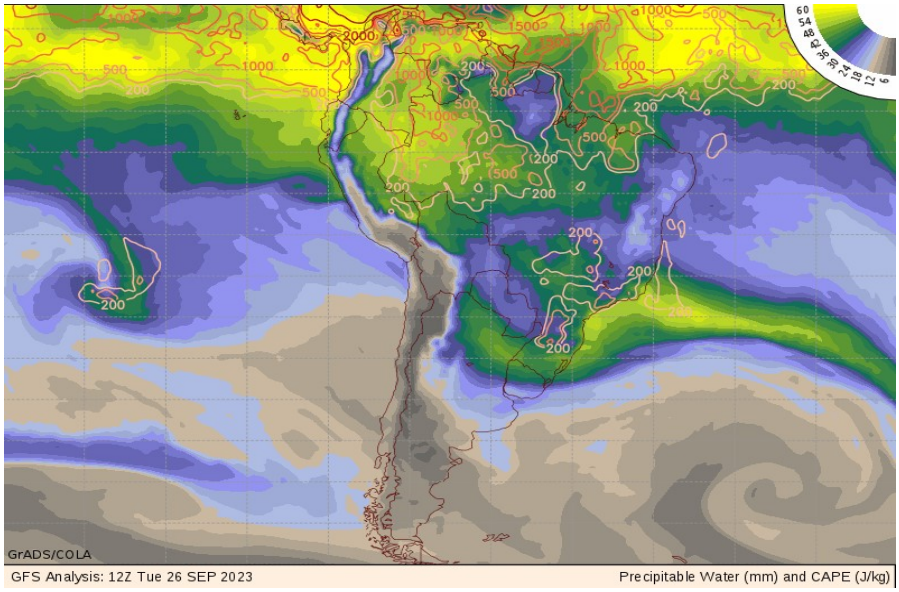
2. REVISÃO DE LITERATURA

The Brazilian developments on the Regional Atmospheric Modeling System (BRAMS) é um modelo numérico de previsão do tempo multifuncional de escala regional. Foi projetado para simular circulações atmosféricas com abrangência desde escalas hemisféricas até grandes simulações de turbilhões (LES) da camada limite planetária (Centro de Previsão de Tempo e Estudos Climáticos (CPTEC/INPE), 2023).

A saídas de modelos numéricos de tempo podem ser acessadas, manipuladas e visualizadas pelo Grid Analysis and Display System (GrADS) de forma interativa e simplificada (Doty and Kinter, 2023). Além de ser implementado mundialmente em diversos sistemas operacionais normalmente utilizados, um aspecto importante sobre o GrADS é ser distribuído gratuitamente pela internet (COLA, 2023).

A manipulação e visualização no GrADS são realizadas por uma linguagem específica. A figura 2.1 apresenta um dos campos meteorológicos (*Preciptable Water and CAPE*) do modelo GFS em um determinado nível atmosférico utilizando o software de visualização GrADS.

Figura 2.1 - Visualização do modelo GFS



Fonte: Cola (2023).

Observa-se a crescente utilização da linguagem *Python* em diversas áreas, inclusive em Ciência da Terra. Suas vantagens incluem: linguagem interpretada, orientada a objetos, interativa, disponibilidade de bibliotecas gráficas e numéricas e uma

estrutura de dados moderna. Sua aplicação inclui desde a análise de dados a computação distribuída, e interface gráficas de usuários a sistema de informação geográficas (Lin, 2012b; Siegel, 2018).

Há uma predisposição natural ao uso do *Python* (Lin, 2012a), como uma opção para substituir a programação *shell script* e GrADS. Dessa forma, optou-se em utilizá-la para manipular e visualizar o BRAMS em um *workflow* meteorológico de uma maneira mais simples.

O pacote *Python* denominado *Xgrads* foi desenvolvido para analisar e ler o arquivo *.ctl* utilizado pelo GrADS. O *Xgrads* realiza a interpretação do arquivo descritor de sufixo "**.ctl*" que contém apenas o conjunto de dados 4D binário bruto (MiniUFO, 2023). Este arquivo se assemelha ao arquivo NetCDF.

A biblioteca do NetCDF possui funções de manipulação de dados armazenados em matrizes que possuem dimensões, variáveis e metadados (Oliveira, Nakai,2014). As informações do cabeçalho contêm todos os dados sobre dimensões, atributos, variáveis, entre outros. Dessa forma, com o auxílio do pacote *Xgrads*, este conjunto de dados 4D binário retorna um *xarray.Dataset*.

Outra biblioteca Python importante nesse processo de interpretação dos dados é o Xarray. Esta biblioteca oferece um conjunto de ferramentas e estrutura de dados para matrizes rotuladas N-dimensionais, o qual é o alicerce da análise de dados científicos moderna.

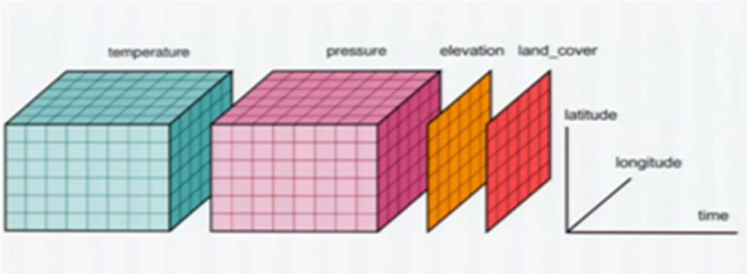
Os recursos primordiais desta biblioteca englobam a indexação com base em rótulos e aritmética, manipulação de dados multidimensionais e interoperabilidade com outros pacotes científicos fundamentais do *Python*. Além disso, a biblioteca apresenta diversas opções de entrada e saída e execução de tarefas paralelamente caso a memória não tenha capacidade de armazenar os dados analisados (Hoyer e Hamman, 2017).

O *Xarray* apresenta rótulos para dimensões, coordenadas e atributos, tornando a experiência do usuário mais intuitiva e menos suscetível a erros (Developers Xarray, 2023). A biblioteca *Numpy* é uma das estruturas de dados central do *Xarray*, sendo evidenciada como um pacote fundamental para computação científica em *Python*, oferecendo um objeto de *array* multidimensional (Numpy Developers, 2023). Dessa

forma, o *Xarray* junto com o *Numpy* carrega, analisa e processa os dados em memória principal.

Ao realizar a implementação do *Xarray*, o *array* multidimensional rotulado é nomeado de *DataArray*. Este possui as seguintes propriedades: valor, dimensão, coordenadas e atributos (Developers Xarray, 2023). O conjunto de *DataArray's* compõem um *Dataset*, onde dentro deste *Dataset* os *DataArray's* compartilham das mesmas dimensões e coordenadas. A representação de um *Dataset* pode ser observada na figura 2.2.

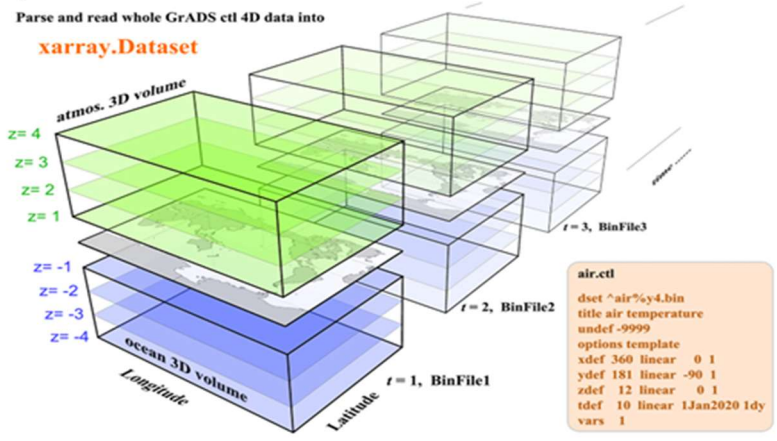
Figura 2.2 - Representação de um Dataset.



Fonte: Developers Xarray (2023).

É possível representar um *xarray.Dataset* como um banco de dados *array* multidimensional, com um grande volume de dados em várias dimensões na memória volátil. Por exemplo, esses dados representam as camadas da atmosfera, terrestre e oceano, as quais compartilham coordenadas, atributos e variáveis (figura 2.3).

Figura 2.3 - Analisar e ler toda a informação dos dados do GrADS ctl 4D.



Fonte: Xgrads (2023).

A biblioteca *Metpy* fornece um conjunto de ferramentas em *Python* para ler, visualizar e realizar cálculos utilizando dados meteorológicos (Developers Metpy, 2023). Segundo May e Bruick (2019) a funcionalidade geral do *Metpy* é dividida em leitura de dados, cálculos e plotagem específica da meteorologia.

Esta biblioteca possui uma coleção de métodos e propriedades anexadas ao *.metpy* para operações com coordenadas/CRS (Coordinate Reference System) (Developers Metpy, 2023). O *Metpy* segue a convenção *Climate and Forecasting (*CF) para suas definições de CRS. Ela as armazena em cache na coordenada *metpy_crs* para que persista através dos cálculos e outras operações de matriz.

Incluído no conjunto de ferramentas, o *Metpy* disponibiliza a identificação dos tipos de coordenadas, pois as mesmas podem possuir nomes diferentes referindo-se ao mesmo tipo. Sendo assim, o *Metpy* identifica de forma sistêmica as coordenadas dos seguintes tipos: latitude, y, longitude, x (Developers Metpy, 2023). Dessa forma, a manipulação dos dados é facilitada pelo *Metpy*, pois este possibilita consultar os dados sem a necessidade de saber especificamente o nome de uma determinada coordenada como apresentado anteriormente.

O *framework Streamlit* proporciona a criação e o gerenciamento de aplicações web simples sem a necessidade de prévia experiência com *front-end*. Trata-se de uma biblioteca *Python* de código aberto que facilita a criação e o compartilhamento de aplicações *web* personalizadas (Snowflake Inc., 2023).

Existem diversas bibliotecas disponíveis para a plotagem de gráficos em *Python*. Entre elas estão a biblioteca *Cartopy*, *Plotly*, *Bokeh* e o *Folium.* O *Cartopy* é um pacote *Python* que foi projetado para o processamento de dados geoespaciais para produzir mapas (Cartopy contributors., 2023).

Para integrar os dados antes coletados ao *Cartopy*, é utilizado o *Matplotlib* que é uma biblioteca abrangente para criação de visualizações estáticas, animadas e interativas (The Matplotlib development team, 2023). Há diversas opções de projeções para cada objetivo específico (Cartopy contributors, 2023).

A biblioteca gráfica *Python* do *Plotly* cria gráficos interativos. Esta biblioteca fornece diversos recursos tais como gráficos de linhas, gráficos de dispersão, gráficos de área, gráficos de barras, barras de erro, gráficos de caixa, histogramas, mapas de calor, subtramas, eixos múltiplos, gráficos polares e gráficos de bolhas (Plotly, 2023). Os gráficos de dispersão nos mapas e o mapa de contorno, por exemplo, destacam áreas geográficas.

O *Bokeh* é uma biblioteca *Python* para criar visualizações interativas para navegadores modernos. Com essa biblioteca é possível criar desde gráficos simples até painéis complexos com conjuntos de dados de streaming (Bokeh Contributors, 2023).

A biblioteca *Folium* simplifica a visualização de dados que foram manipulados em *Python* em um mapa de folheto interativo. A criação dessa biblioteca se baseou na organização de dados da linguagem *Python* e da estrutura da biblioteca *Leaflet.js.* Também permite a transmissão de vetores como marcadores de mapa. Outra funcionalidade é oferecer suporte a sobreposição de imagem e *GeoJSON*, entre outros formatos, e dispõe diversas camadas vetoriais integradas (Rob Story, 2023).

O geojsoncontour é um módulo *Python* para converter gráficos de contorno Matplotlib em formato *GeoJSON*. O objetivo desse módulo é apresentar gráficos de contorno geográfico, criados com matplotlib, como camada vetorial em mapas interativos como OpenLayers e Leaflet (Bart Römgens, 2023).

Como proposta deste trabalho temos a implementação de um workflow meteorológico fazendo uso de bibliotecas da linguagem Python. Dessa forma, possibilitando a análise e avaliação das bibliotecas como alternativa ao sistema GrADS na visualização e manipulação dos dados produzidos pelo modelo BRAMS.

3. MATERIAL E MÉTODOS

Este projeto explorou bibliotecas Python com o objetivo de visualizar os campos meteorológicos do BRAMS de uma maneira mais simplificada para o usuário. Para isto, como parte do desenvolvimento foi necessário estudar o processo atual e em seguida avaliar as possibilidades que a linguagem Python proporciona. Sendo assim, inicialmente houve a familiarização da saída do modelo BRAMS e com a ferramenta GrADS.

Posteriormente, um estudo da literatura foi realizado com o objetivo de identificar quais bibliotecas da linguagem Python seriam capazes de visualizar os dados da saída do modelo numérico BRAMS de uma maneira mais simples para o usuário em comparação à ferramenta atual GrADS. Entretanto, para visualizar esses dados foi necessário utilizar diversas bibliotecas da linguagem Python para acessar, analisar e extrair dados das saídas do modelo BRAMS.

A geração dos dados a partir da leitura e interpretação da saída do modelo numérico BRAMS foi feita utilizando o pacote Python *Xgrads (*v.0.2.3). Para a leitura deste conjunto de dados binário foi feita a conversão do arquivo .ctl para um *xarray.Dataset* utilizando a funcionalidade "*open_CtlDataset*" do pacote *Xgrads*.

O *Xarray* (v.0.20.2) em conjunto com o *Numpy* (v.1.21.6), permite carregar, corrigir, analisar e processar os dados em memória. Dessa forma, após a interpretação teremos os dados lidos pelo *Xgrads*, convertidos para *Xarray*, tornando dessa maneira possível a análise dos dados.

Esta biblioteca foi responsável pelo tratamento de dados que permitiu selecioná-los e carregá-los na memória da máquina. Esses dados foram armazenados na forma de um *array* multidimensional por meio da conversão dos dados do arquivo "**.ctl*" que até esse ponto está como "*xgrads.core.CtlDescriptor*", para o "*xarray.core.dataset.Dataset*".

Após a conversão para *Xarray* emprega-se o *Metpy (*v.1.2.0), que permitiu o *parse* do *array* multidimensional utilizando o método *parse_cf()*. Dessa forma, foi possível manipular os dados do *Dataset* de maneira simplificada.

Para disponibilizar a manipulação dos dados para o usuário foi adotado o *framework Streamlit (*v.1.16.0). Com o auxílio desse *framework* é possível apresentar o

projeto em uma aplicação *web* interativa. Já a biblioteca *Matplotlib* (v.3.3.2) foi utilizada para gerar os gráficos de contorno dos dados.

Para a geração de mapas com os dados dos campos meteorológicos, apenas quatro bibliotecas serão implementadas e avaliadas para se apresentarem como possíveis alternativas para a ferramenta GrADS na manipulação e visualização do BRAMS. As quatro bibliotecas *Python*: *Cartopy* (v.0.20.3), *Plotly* (v.5.8.0), *Bokeh* (v.2.4.3) e *Folium* (0.14.0).

As seguintes bibliotecas Python foram exploradas neste projeto de pesquisa para visualizar os campos de atmosféricos do modelo atmosféricos BRAMS em seus diferentes níveis atmosféricos:

- *Cartopy* é um pacote projetado para o processamento de dados geoespaciais para produzir mapas. Foi utilizado a projeção *PlateCarreeo* para a geração do mapa;
- *Matplotlib* é uma biblioteca para criação de visualizações, foi utilizada para criação dos gráficos de contorno. *Plotly* é uma biblioteca para criar gráficos e mapas interativos. Para a geração do mapa foi explorada projeção *Scattergeo* em conjunto com a função *Contour*() para geração do gráfico de contorno;
- *Bokeh* é uma biblioteca que apresenta a funcionalidade de criar visualizações interativas. Foi utilizado a projeção *CartoDB Positron* para a geração do mapa;
- Folium é uma biblioteca que simplifica a visualização de dados em um mapa interativo e oferece suporte a sobreposição de imagem e compatibilidade com o formato GeoJSON. A projeção utilizada para geração do mapa no Folium foi a *OpenStreetMap;*
- O geojsoncontour é um módulo que faz a conversão do resultado do *Matplotlib* em formato GeoJSON que fica sobreposto no mapa gerado pelo *Folium;*
- O *framework Streamlit* foi utilizado para disponibilizar a visualização dos mapas gerados de forma simples em uma aplicação web personalizada.

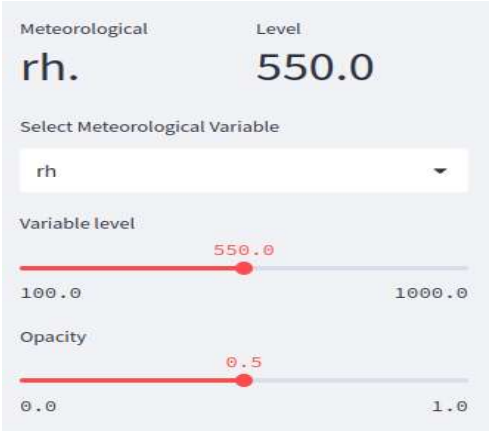
4. RESULTADOS E DISCUSSÕES

O BRAMS é um modelo atmosférico de previsão do tempo e gera a saída com as informações meteorológicas que são utilizadas atualmente pelo GrADS na geração de mapas e gráficos. Contudo, o manuseio do GrADS para manipular e visualizar dados é realizada por meio de *shell script* devido essa ferramenta não apresentar interface gráfica.

A utilização da linguagem Python como alternativa para visualizar os campos meteorológicos do modelo numérico BRAMS é devido a fácil aprendizagem e muito aplicado em projetos relacionados à ciência de dados por apresentar diversas bibliotecas. Durante o desenvolvimento do projeto foram analisadas bibliotecas para manipulação e visualização de dados atmosféricos, definição e adaptação da biblioteca adequada e a preparação e automação do ambiente de trabalho para a leitura desses dados.

O *framework* Streamlit permitiu a manipulação dos dados e visualização dos campos meteorológicos (em seus diferentes níveis atmosféricos) em um servidor *web*, disponibilizados na forma simples e rápida de um portal *web*. A figura 4.1 apresenta uma interface de fácil utilização pelo usuário implementada com o *framework Streamlit* que contém o nome da variável, o nível, um seletor para a variável desejadae um *slider* para definir o nível ("*Variable level*") e um *slider* para definir a opacidade.

Figura 4.1 - Menu de controle dos dados.



Fonte: Autoria própria (2023).

A figura 4.1 apresenta a variável meteorológica selecionada ("rh"), que significa Umidade Relativa. As camadas dessa variável estão dispostas de 100 a 1000, sendo possível serem visualizadas em intervalos de 50. A figura 4.2 mostra o resultado da interface da linha de comando quando a variável com seu *level* são selecionados.

Figura 4.2 – Resultado da interface da linha de comando.

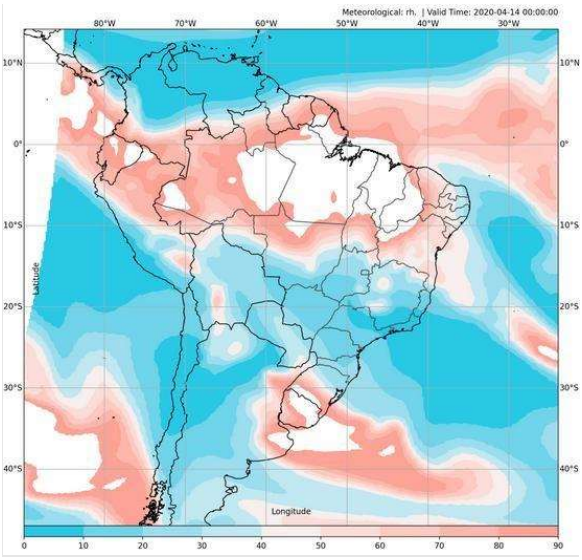
```
########################################################################
***                  TIME: 2023-09-28 16:45:00                       ***
########################################################################


>>> Selection:  rh 550.0
>>>  <class 'xarray.core.dataarray.DataArray'>
----------------------------------------
>>> var_interval:  [ 0 10 20 30 40 50 60 70 80 90]
>>> var_selected:  rh
<class 'cartopy.mpl.geoaxes.GeoAxes'>
```

Fonte: Autoria própria (2023).

Para a geração de mapas com os dados dos campos atmosféricos foram exploradas quatro bibliotecas Python: Cartopy, Plotly, Bokeh e Folium. Inicialmente, optou-se pela biblioteca Cartopy em conjunto com o Matplotlib para realizar a integração dos dados coletados.

A apresentação do mapa com o gráfico de contorno dos dados gerados pelo Matplotlib foi disponibilizada ao usuário, entretanto, não foi possível a interação do usuário com o mapa. A figura 4.3 mostra a imagem criada com o Cartopy e Matplotlib disponibilizada ao usuário.

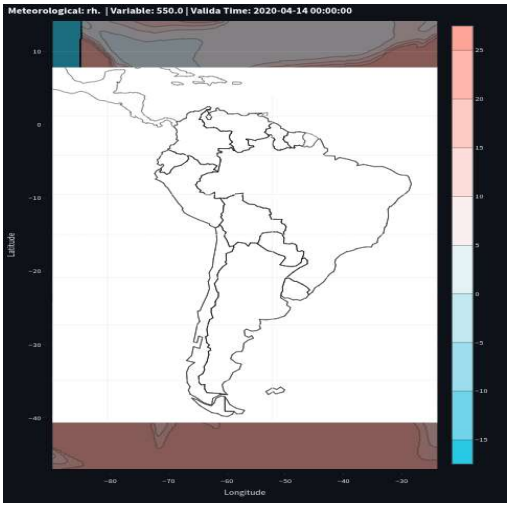
Figura 4.3 – Resultado Cartopy e Matplotlib.



Fonte: Autoria própria (2023).

A biblioteca Plotly permitiu apresentar mapas interativos e o contorno dos mapas. No entanto, não foi possível realizar a integração entre o gráfico de contorno e o mapa gerado, ficando o mapa sobreposto ao gráfico. Isso pode ser observado na figura 4.4.

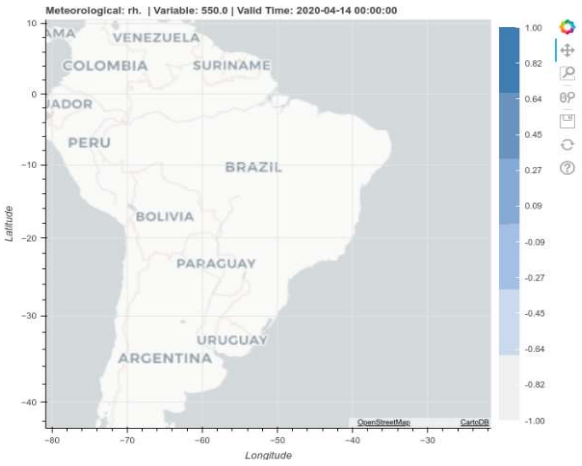
Figura 4.4 – Resultado Plotly.



Fonte: Autoria própria (2023).

A biblioteca Bokeh possui apenas o recurso necessário para realizar a plotagem do mapa. Na versão utilizada não possuía nenhuma função para criação de gráficos de contorno. Dessa forma, o resultado obtido foi apenas o mapa que pode ser observado na figura 4.5.

Figura 4.5 – Resultado Bokeh.

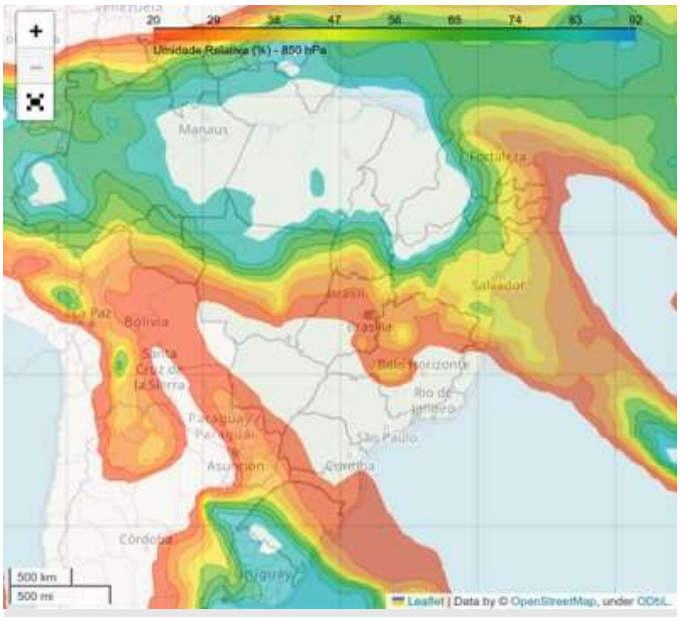


Fonte: Autoria própria (2023).

O Folium foi utilizado para geração dos mapas, sendo o *OpenStreetMap* utilizado como mapa. Foi implementada no mapa a funcionalidade de *zoom in e zoom out,* e a possibilidade de colocar em tela cheia para melhor visualização dos dados. Também foi criado um *grid map* que foi integrado ao mapa.

O *Matplotlib* e o *geojsoncontour* foram utilizados na geração dos gráficos de contornos, onde o resultado gerado pelo *Matplotlib* é convertido pelo *geojsoncontour* em um GeoJSON. Esse GeoJSON foi adicionado e disponibilizado sobre o mapa gerado. Sendo assim, foi possível criar um mapa interativo com o gráfico de contorno (figura 4.6).

Figura 4.6 – Resultado com Folium, geojsoncontour e Matplotlib.



Fonte: Autoria própria (2023).

5. CONCLUSÃO

Este projeto possibilitou adquirir conhecimentos sobre modelos atmosféricos, linguagem Python, suas bibliotecas Folium, Geojsoncontour, Matplotlib, Metpy, Numpy, Xarray, Xgrads e o *framework* Streamlit. Desta forma, foi possível utilizar estes conhecimentos na manipulação e visualização de dados atmosféricos, definição e adaptação da biblioteca adequada, e a preparação e automação do ambiente de trabalho para a leitura desses dados.

Observou que apesar das bibliotecas Cartopy, Plotly e Bokeh terem finalidade de geração de mapas, elas não atenderam alguns dos objetivos do projeto na versão utilizada. O Cartopy, por exemplo, não disponibiliza mapas interativos. Já o Plotly e o Bokeh não disponibilizaram os dados em conjunto com o mapa.

Por outro lado, a biblioteca Folium em conjunto com o Matplotlib e Geojsoncontour foi capaz de gerar mapas com os dados dos campos meteorológicos. O *framework* Streamlit permitiu a criação de uma interface web simples para disponibilizar os mapas gerados.

Como resultados, foi possível disponibilizar as plotagens dos mapas em uma aplicação *web* interativa e simplificada para o usuário. Desta forma, a manipulação e visualização dos mapas foi facilitado ao usuário final, pois não há a necessidade de ter prévio conhecimento de alguma linguagem de programação.

6. REFERÊNCIAS

Almeida, E.S.; Bauer, M. (2012). Reducing Time Delays in Computing Numerical Weather Models at Regional and Local Levels: A Grid-Based Approach. International Journal of Grid Computing & Applications, 3(4), 1.

Bart Römgens. GitHub – bartromgens/geojsoncontour: Convert matplotlib contour plots to geojson. Disponível em: <https://github.com/bartromgens/geojsoncontour/tree/master>. Acesso em: 10 de maio de 2023.

Bokeh contributors. Bokeh. Disponível em: < http://bokeh.org/>. Acesso em: 29 de agosto de 2023.

Cartopy contributors. Introduction – Cartopy 0.22.0 documentation. Disponível em: < https://scitools.org.uk/cartopy/docs/latest/ > Acesso em: 27 de agosto de 2023.

Centro de Previsão de Tempo e Estudos Climáticos (CPTEC/INPE). BRAMS, 2015. About BRAMS. Disponível em: < http://brams.cptec.inpe.br/about/ >. Acesso em: 21 de janeiro de 2023.

COLA. Grid Analysis and Display System (GrADS). 2021. GrADS Home Page. Disponível em: < http://cola.gmu.edu/grads/ > Acesso em: 22 de janeiro de 2023.

Corte-Real, João. 2015. A importância da previsão do tempo na prevenção de riscos meteorológicos. Disponível em: <https://revistas.rcaap.pt/finisterra/article/view/7867 > Acesso em: 29 de junho de 2023.

Doty, B.; Kinter III, J.L. (1993) The Grid Analysis and Display System (GrADS): An update. Ninth International Conference on Interactive Information and Processing Systems, Anaheim, CA, 17-22.

Hoyer, S., Hamman, J. (2017). xarray: ND labeled arrays and datasets in Python. Journal of Open Research Software, 5(1).

Kinter III, J.L.; B. Doty, B. (1993). The Grid Analysis and Display System: A practical desktop tool for analyzing geophysical data. Information Systems Newsletter, 27, NASA, OSSA, JPL, Pasadena, CA.

Lin, J.W.B. (2012a). A Hands-On Introduction to Using Python in the Atmospheric and Oceanic Sciences. Lulu. com.

Lin, J. W. B. (2012b). Why Python is the next wave in earth sciences computing. Bulletin of the American Meteorological Society, 93(12), 1823-1824.

Lynch, P. (2008) 'The origins of computer weather prediction and climate modeling', in Journal of Computational Physics, Vol. 227, No. 7, pp.3431–3444.

May, R.; Bruick, Z. (2019) MetPy: A Community-Driven, Open-Source Python Toolkit for Meteorology. AGUFM, 2019, NS21A-16.

Metpy Developers. MetPy – MetPy 1.5; Disponível em: < https://unidata.github.io/MetPy/latest/index.html>. Acesso em: 19 de agosto de 2023.

MiniUFO; Welcome to xgrads's documentation!. Disponível em: < https://xgrads.readthedocs.io/en/latest/>. Acesso em: 10 de julho de 2023.

Numpy Developers. NumPy. Disponível em:< https://numpy.org/>. Acesso 19 de agosto de 2023.

Oliveira, Graciela; Nakai, Alan Massaru. 2014.Utilizando R para manipular dados de projeção climática. Disponivel em: < https://www.alice.cnptia.embrapa.br/alice/bitstream/doc/1009820/1/06814.pdf > Acesso em: 15 de julho de 2023.

Plotly. Plotly: Low-Code Data App Development. Disponível em: <https://plotly.com/>. Acesso em 29 de agosto de 2023.

Rob Story. Folium – Folium 0.1.dev1+g57e8eae documentation. Disponível em: <https://python-visualization.github.io/folium/latest/>. Acesso em: 28 de agosto de 2023.

Sampaio, Gilvan; Dias, Pedro Leite da Silva. Evolução dos modelos climáticos e de previsão de tempo e clima. REVISTA USP, São Paulo, n. 103, p. 41-54, 2014. Disponível em: < https://www.revistas.usp.br/revusp/article/view/99179/97655 > Acesso em: 5 de junho de 2023.

Siegel, Idaltchion Fabricio. 2018. LINGUAGEM PYTHON E SUAS APLICAÇÕES EM CIÊNCIA DE DADOS. Disponível em: < https://app.uff.br/riuff/bitstream/handle/1/8946/TCC_IDALTCHION_FABRICIO_SIEGEL.pdf?sequence=1&isAllowed=y >Acesso em: 13 de julho de 2023.

Snowflake Inc. Streamlit – A faster way to build and share data apps. Disponível em: < https://streamlit.io/>. Acesso em: 10 de agosto de 2023.

The Matplotlib development team. Matplotlib – Visualization with Python. Disponível em: < https://matplotlib.org/>. Acesso em: 10 de agosto de 2023.

Xarray Developers; Xarray documentation. Disponível em: <https://docs.xarray.dev/en/stable/>. Acesso em: 10 de julho de 2023.