



とAIで作ったツール

各種AIで生成したpythonスクリプトを基にツールを作成してみました。
JvqBle、JvqBleIosで保存したcsv, kmlからチャート、マップを作成します。

・使用したハード・ソフトウェアとAI先生

Mac
Windows11
iPad

Visual Studio Code
python3
jupyter notebook
Google Colab

ChatGPT
Copilot
Google検索

■ Pythonについて

Pythonは、高水準で読みやすく、簡潔な構文を持つプログラミング言語です。データ分析、ウェブ開発、機械学習など、幅広い用途に利用されており、豊富なライブラリと大規模なコミュニティが特徴です。(ChatGPTより)

・ AIに聞いてみました。

pythonは得意ですか？

・ ChatGPTの答え

はい、Pythonについてはかなり得意です！データ分析、ウェブ開発、機械学習、アルゴリズムの実装など、Pythonを使ったさまざまな課題に対応できます。質問があれば何でも聞いてください！

・ Copilotの答え

Pythonは得意です！プログラミングの基本から、データ分析、ウェブ開発、機械学習など、幅広いトピックについて対応できます。具体的に何かお手伝いできることがありますか？例えば、コードの書き方やデバッグのお手伝いなどがあれば、ぜひ教えてくださいね。

■ JVQファイルのフォーマット

・ CSVファイル(.csv)

No: 通し番号, Date: 年月日時分秒ミリ秒, X, Y, Z: 加速度, T: 温度, H: 湿度, A: 気圧

```
No, Date, X, Y, Z, T, H, A
0, 2025/02/18 13:04:00.000, -0.01, -1.02, 0.00, 15.7, 44, 914
1, 2025/02/18 13:06:00.000, -0.01, -1.02, 0.01, 15.5, 44, 914
2, 2025/02/18 13:10:00.000, -0.01, -1.02, -0.01, 15.5, 44, 914
3, 2025/02/18 13:12:00.000, -0.01, -1.02, 0.00, 15.5, 40, 914
```

・ GPSウェイポイントファイル(.kml)

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
  <Document>
    <Placemark>
      <name>24/11/27 16:38:45</name>
      <description>0001 (35.99310, 138.15847, 820)</description>
      <Point>
        <coordinates>138.15847, 35.99310, 820</coordinates>
      </Point>
      <TimeStamp>
        <when>2024-11-27T16:38:45Z</when>
      </TimeStamp>
    </Placemark>
  </Document>
</kml>
```

・ GPSトラックファイル(.kml)

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
  <Document>
    <Placemark>
      <name>Tracks</name>
      <Style>
        <LineStyle>
          <color>ffd18802</color>
          <width>5</width>
        </LineStyle>
      </Style>
      <LineString>
        <coordinates>
          138.15847, 35.99310, 820
          138.15853, 35.99295, 819
          138.15848, 35.99313, 817
        </coordinates>
      </LineString>
    </Placemark>
  </Document>
</kml>
```

※実際のkmlファイルにインデントは付けてありません。

■ チャート描画の基本形

・ ChatGPTへの指示

```
python
CSVファイルを開く。
X軸を'Date', Y軸を'T' としてグラフを描く。
```

・ 出力されたコード

```
import pandas as pd
import matplotlib.pyplot as plt

# CSVファイルを読み込む (適切なファイルパスに変更してください)
file_path = "data.csv" # 例: 'C:/path/to/data.csv'
df = pd.read_csv(file_path)

# 'Date'列を日付型に変換 (必要に応じて)
df['Date'] = pd.to_datetime(df['Date'])

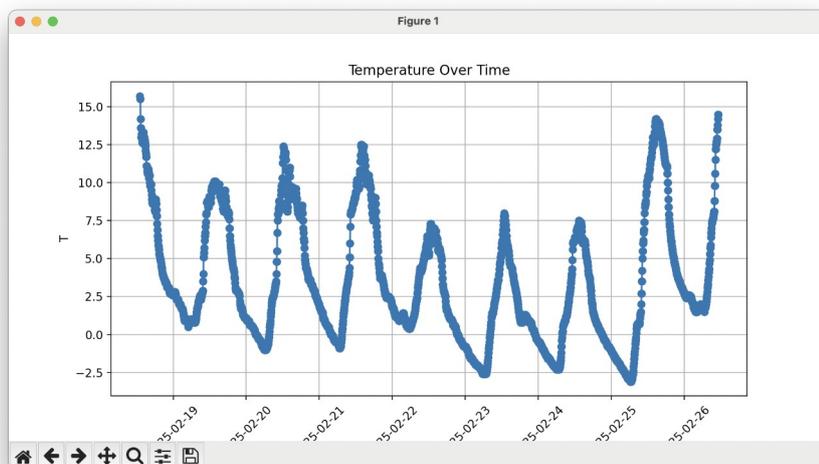
# グラフを描画
plt.figure(figsize=(10, 5))
plt.plot(df['Date'], df['T'], marker='o', linestyle='-')

# タイトルとラベル
plt.title("Temperature Over Time")
plt.xlabel("Date")
plt.ylabel("T")
plt.xticks(rotation=45) # 日付ラベルを見やすく回転

# グリッド表示
plt.grid()

# グラフを表示
plt.show()
```

・ そのまま実行した結果



・前ページから、繰り返しチャット&手直しをして各データをウィンド別に描画するようにしました。

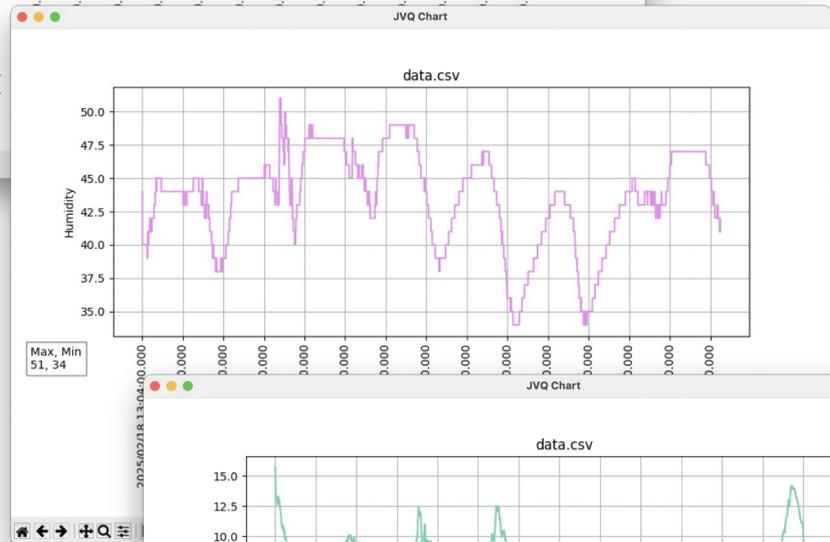
データ選択

- X
- Y
- Z
- 温度
- 湿度
- 気圧

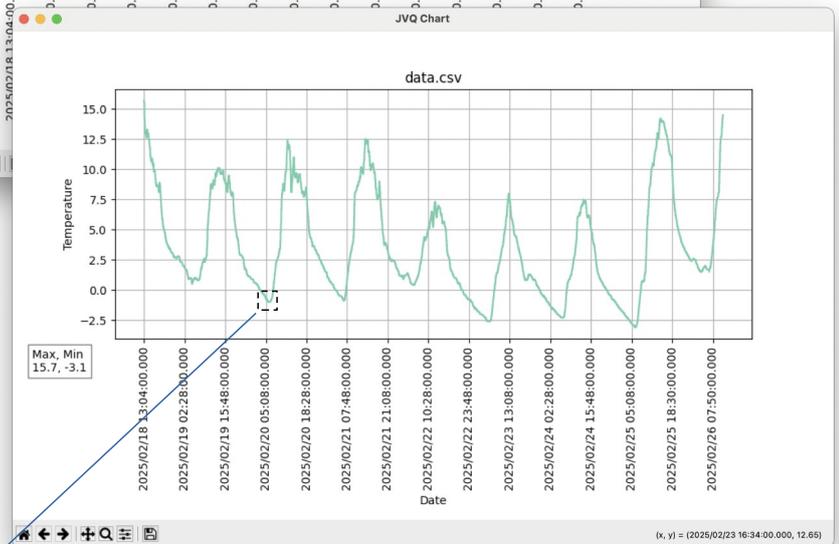
チャート

終了

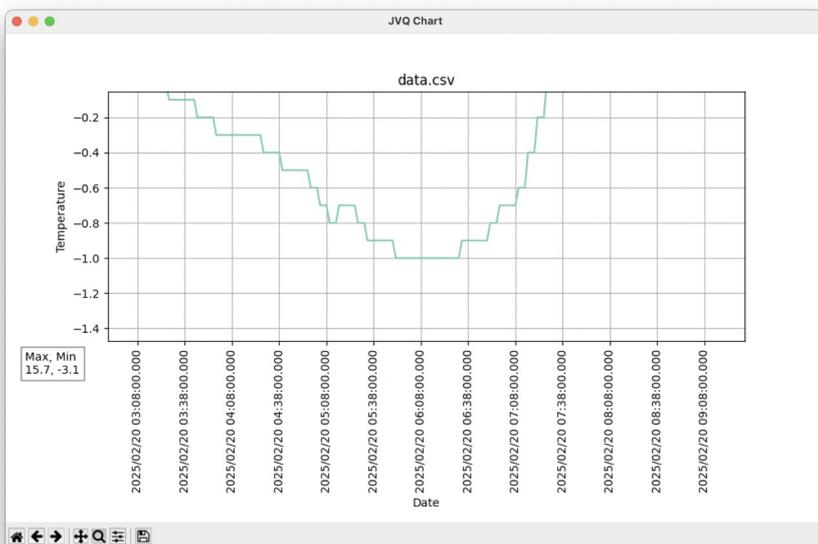
データを選択



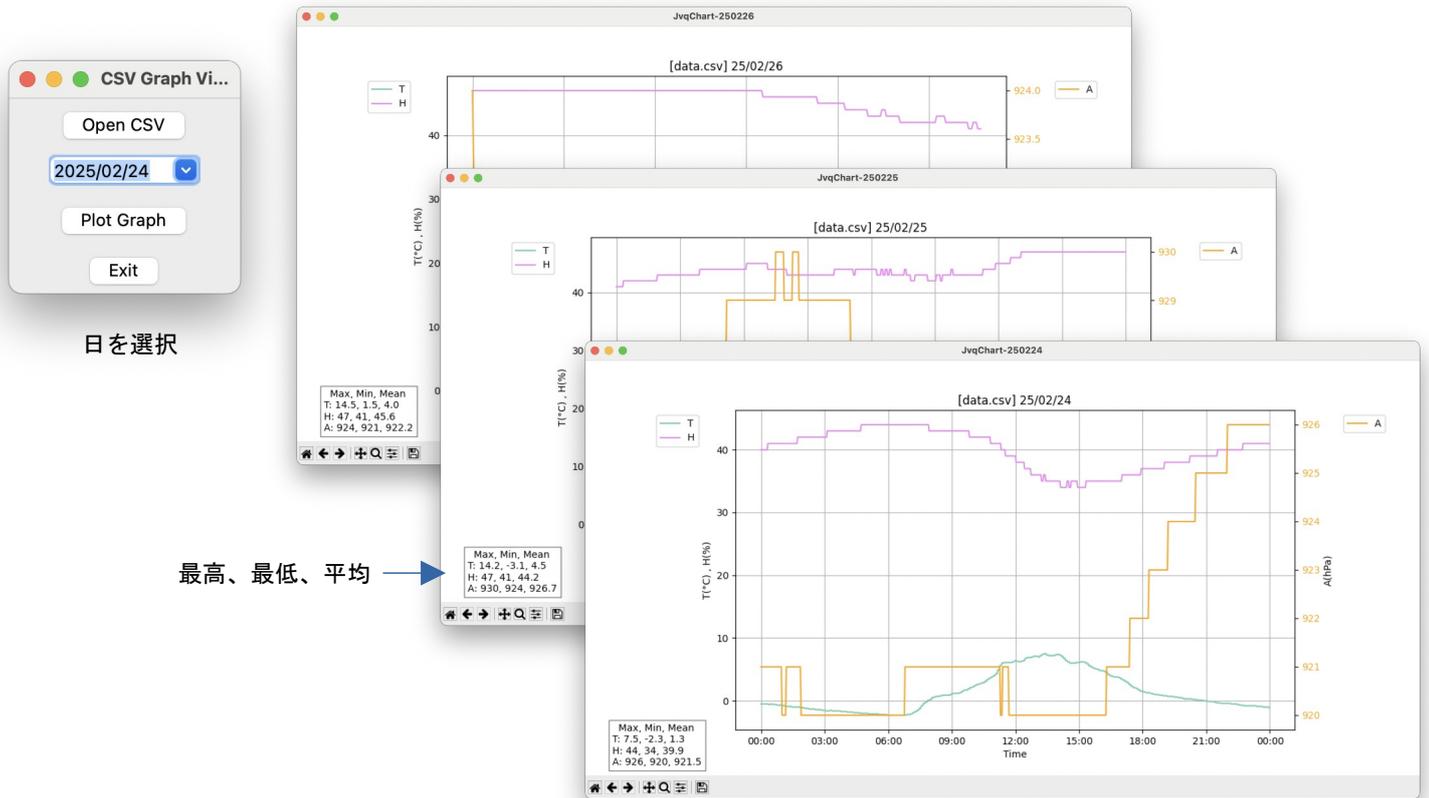
複数のウィンドを表示できます。



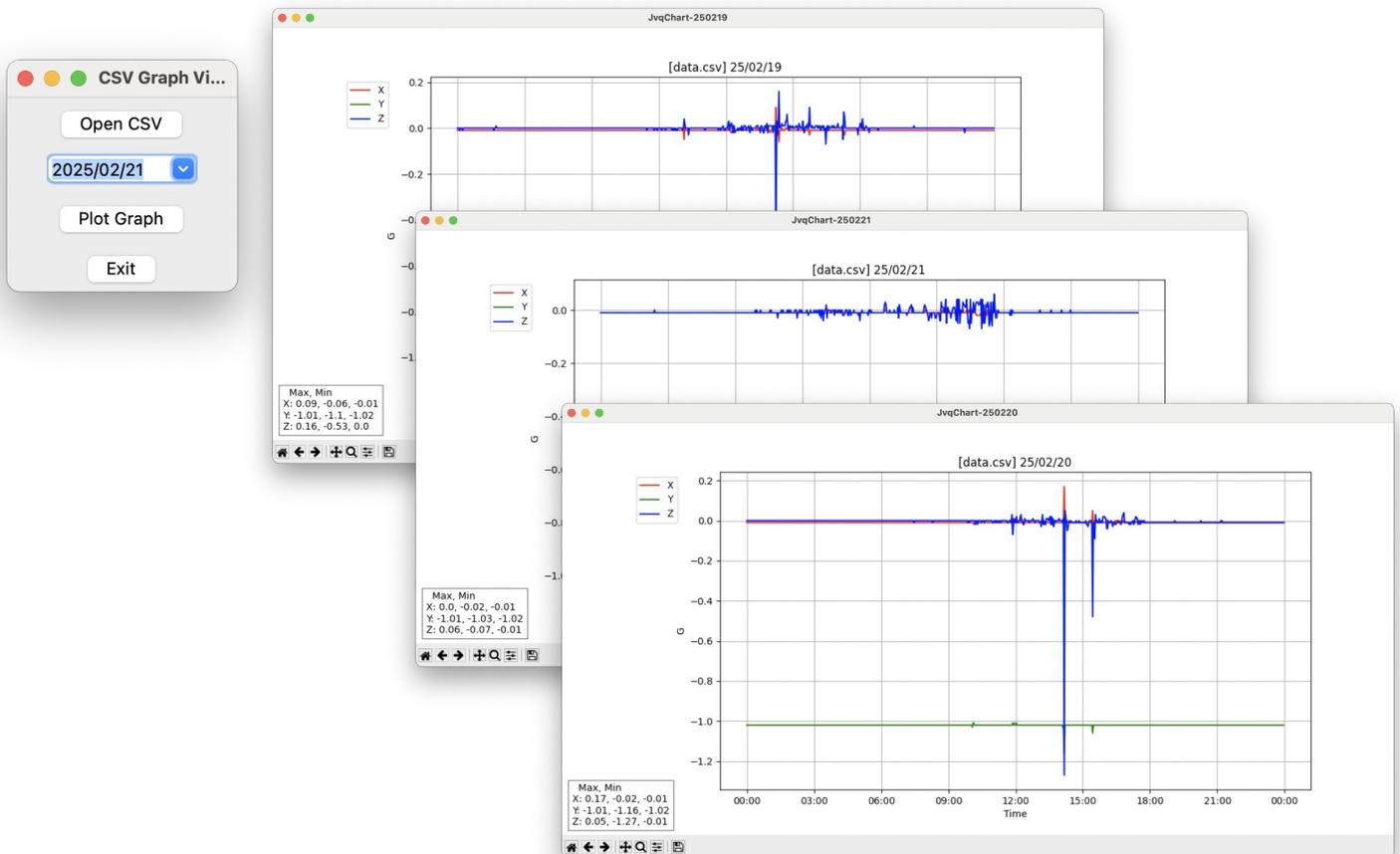
拡大、png保存などができます。



・1日単位のチャートを描かせてみました。左Y軸は温度と湿度、右Y軸は気圧のスケール。



・1日単位のチャート。重力加速度X,Y,Zバージョン。



■ 表作成の基本形

- ・ ChatGPTへの指示

```
python3
data.csvを開きウインドに表を作成する。
```

- ・ 出力されたコード

```
import pandas as pd
import tkinter as tk
from tkinter import ttk

# CSVファイルを読み込む
file_path = "data.csv" # 適切なパスに変更
df = pd.read_csv(file_path)

# tkinterウインドウを作成
root = tk.Tk()
root.title("CSV Data Table")

# 表の作成
tree = ttk.Treeview(root)

# DataFrameのカラムをウインドウにセット
tree["columns"] = list(df.columns)
tree["show"] = "headings" # インデックス列を表示しない

# 各列のヘッダー設定
for col in df.columns:
    tree.heading(col, text=col)

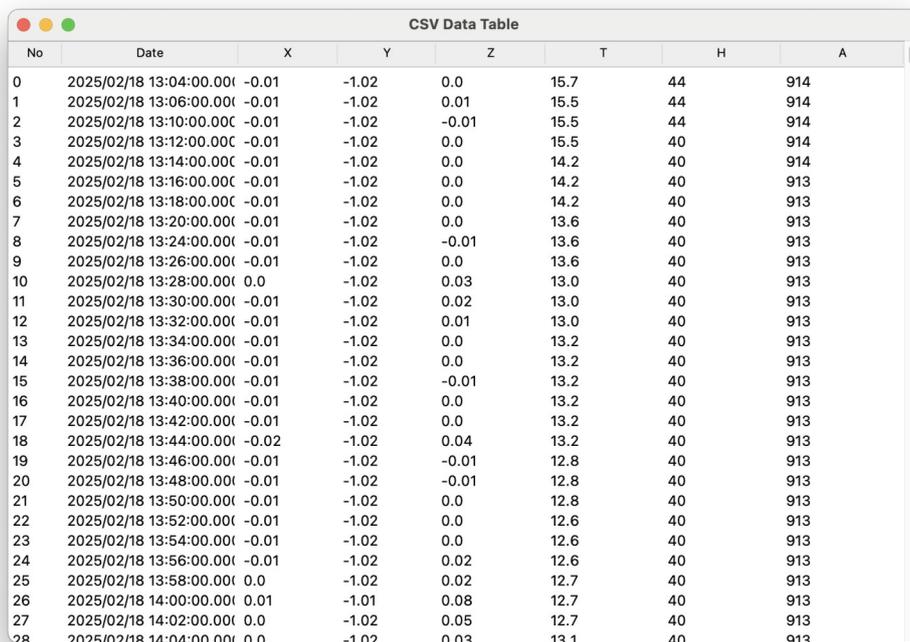
# データをウインドウに挿入
for _, row in df.iterrows():
    tree.insert("", "end", values=list(row))

# スクロールバーを追加
scrollbar = tk.Scrollbar(root, orient="vertical", command=tree.yview)
tree.config(yscrollcommand=scrollbar.set)
scrollbar.pack(side="right", fill="y")

# Treeviewを表示
tree.pack(expand=True, fill="both")

# GUIウインドウを開始
root.mainloop()
```

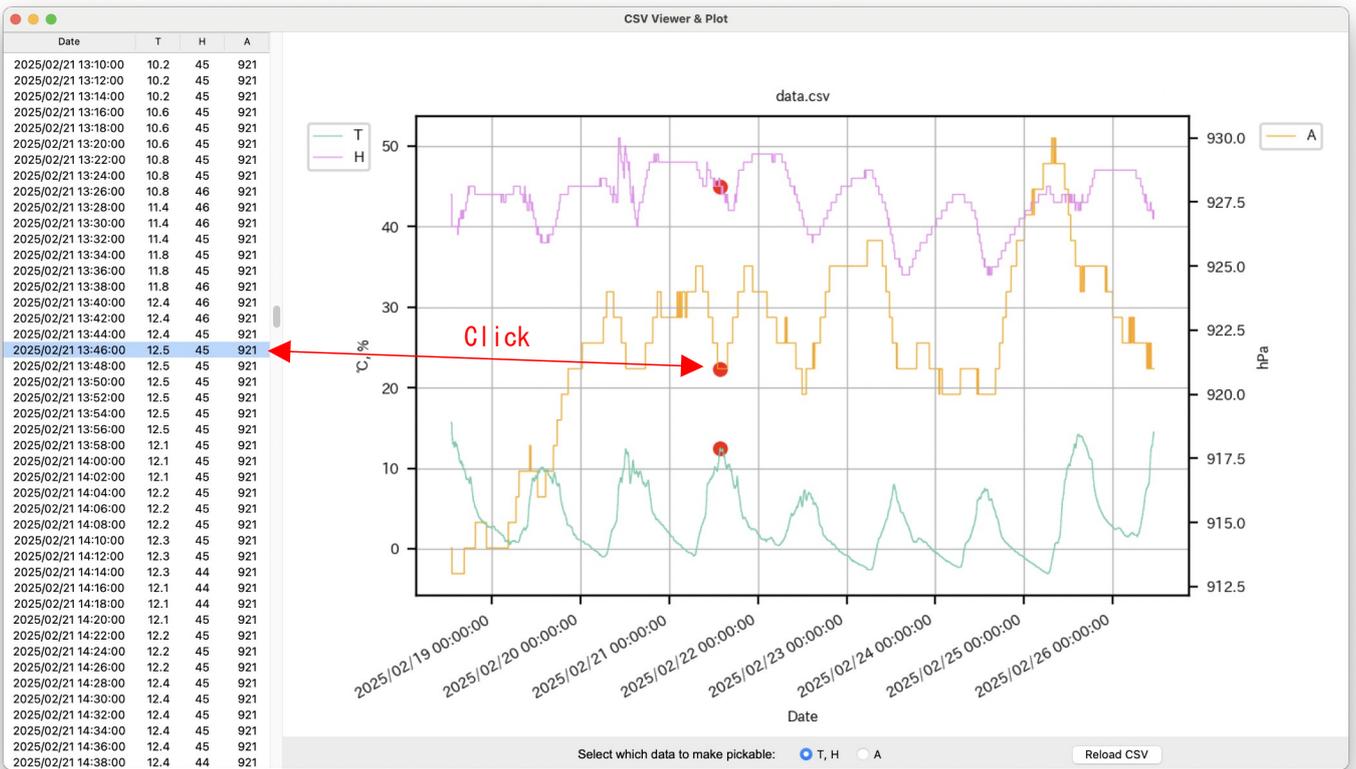
- ・ そのまま実行した結果



No	Date	X	Y	Z	T	H	A
0	2025/02/18 13:04:00.000	-0.01	-1.02	0.0	15.7	44	914
1	2025/02/18 13:06:00.000	-0.01	-1.02	0.01	15.5	44	914
2	2025/02/18 13:10:00.000	-0.01	-1.02	-0.01	15.5	44	914
3	2025/02/18 13:12:00.000	-0.01	-1.02	0.0	15.5	40	914
4	2025/02/18 13:14:00.000	-0.01	-1.02	0.0	14.2	40	914
5	2025/02/18 13:16:00.000	-0.01	-1.02	0.0	14.2	40	913
6	2025/02/18 13:18:00.000	-0.01	-1.02	0.0	14.2	40	913
7	2025/02/18 13:20:00.000	-0.01	-1.02	0.0	13.6	40	913
8	2025/02/18 13:24:00.000	-0.01	-1.02	-0.01	13.6	40	913
9	2025/02/18 13:26:00.000	-0.01	-1.02	0.0	13.6	40	913
10	2025/02/18 13:28:00.000	0.0	-1.02	0.03	13.0	40	913
11	2025/02/18 13:30:00.000	-0.01	-1.02	0.02	13.0	40	913
12	2025/02/18 13:32:00.000	-0.01	-1.02	0.01	13.0	40	913
13	2025/02/18 13:34:00.000	-0.01	-1.02	0.0	13.2	40	913
14	2025/02/18 13:36:00.000	-0.01	-1.02	0.0	13.2	40	913
15	2025/02/18 13:38:00.000	-0.01	-1.02	-0.01	13.2	40	913
16	2025/02/18 13:40:00.000	-0.01	-1.02	0.0	13.2	40	913
17	2025/02/18 13:42:00.000	-0.01	-1.02	0.0	13.2	40	913
18	2025/02/18 13:44:00.000	-0.02	-1.02	0.04	13.2	40	913
19	2025/02/18 13:46:00.000	-0.01	-1.02	-0.01	12.8	40	913
20	2025/02/18 13:48:00.000	-0.01	-1.02	-0.01	12.8	40	913
21	2025/02/18 13:50:00.000	-0.01	-1.02	0.0	12.8	40	913
22	2025/02/18 13:52:00.000	-0.01	-1.02	0.0	12.6	40	913
23	2025/02/18 13:54:00.000	-0.01	-1.02	0.0	12.6	40	913
24	2025/02/18 13:56:00.000	-0.01	-1.02	0.02	12.6	40	913
25	2025/02/18 13:58:00.000	0.0	-1.02	0.02	12.7	40	913
26	2025/02/18 14:00:00.000	0.01	-1.01	0.08	12.7	40	913
27	2025/02/18 14:02:00.000	0.0	-1.02	0.05	12.7	40	913
28	2025/02/18 14:04:00.000	0.0	-1.02	0.03	13.1	40	913

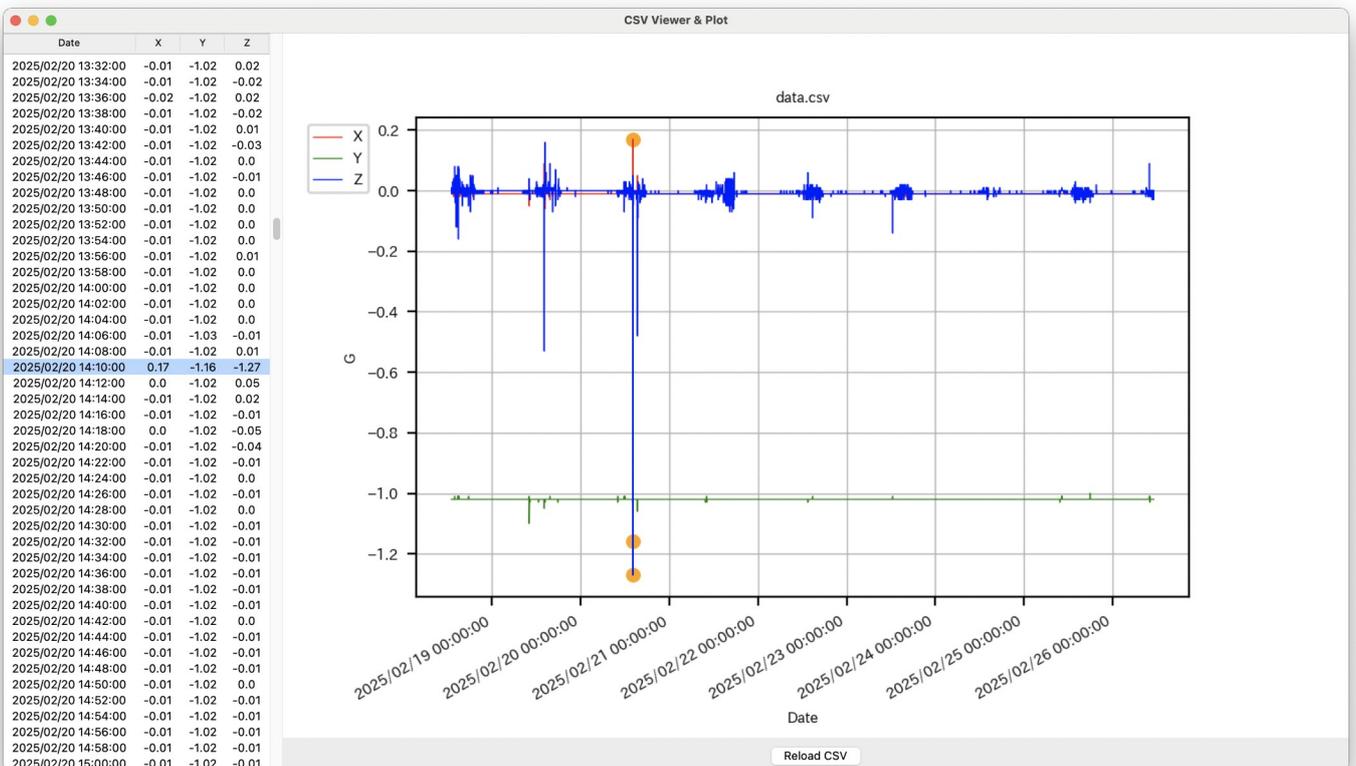
・表とチャートを組み合わせました。クリックでマーカを表示、相互に連動します。

温度、湿度、気圧 バージョン



※ matplotlib twinx() で右側のY軸を作ると左Y軸のpick_eventが発火しなくなるらしい。対策としてRadiobuttonで左右のzorderの順位を切り替えるようにした。

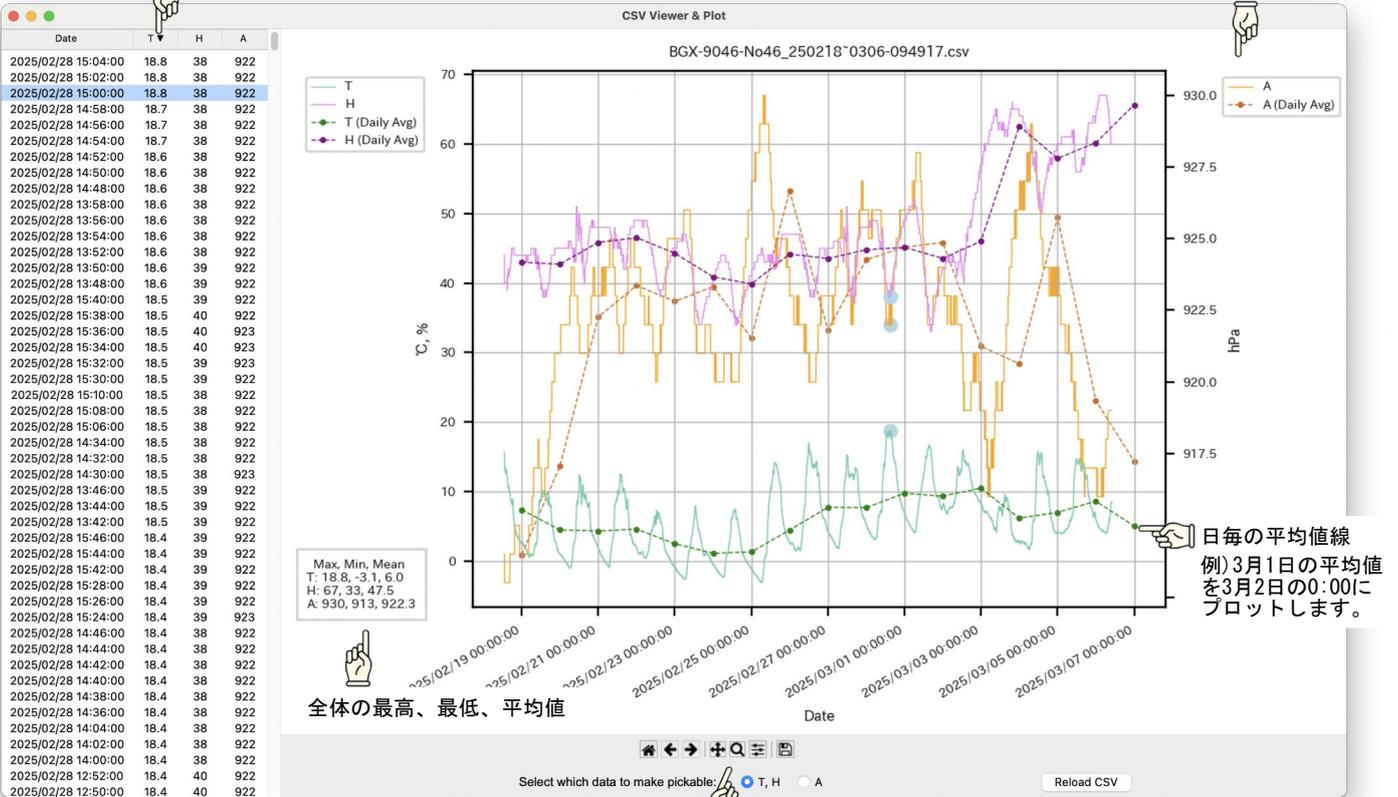
重力加速度X, Y, Z バージョン



・前ページの“温度、湿度、気圧 対応版”を基に、更にChatGPTとチャット&修正を重ねて機能を追加しました。

ヘッダークリックでソート順変更

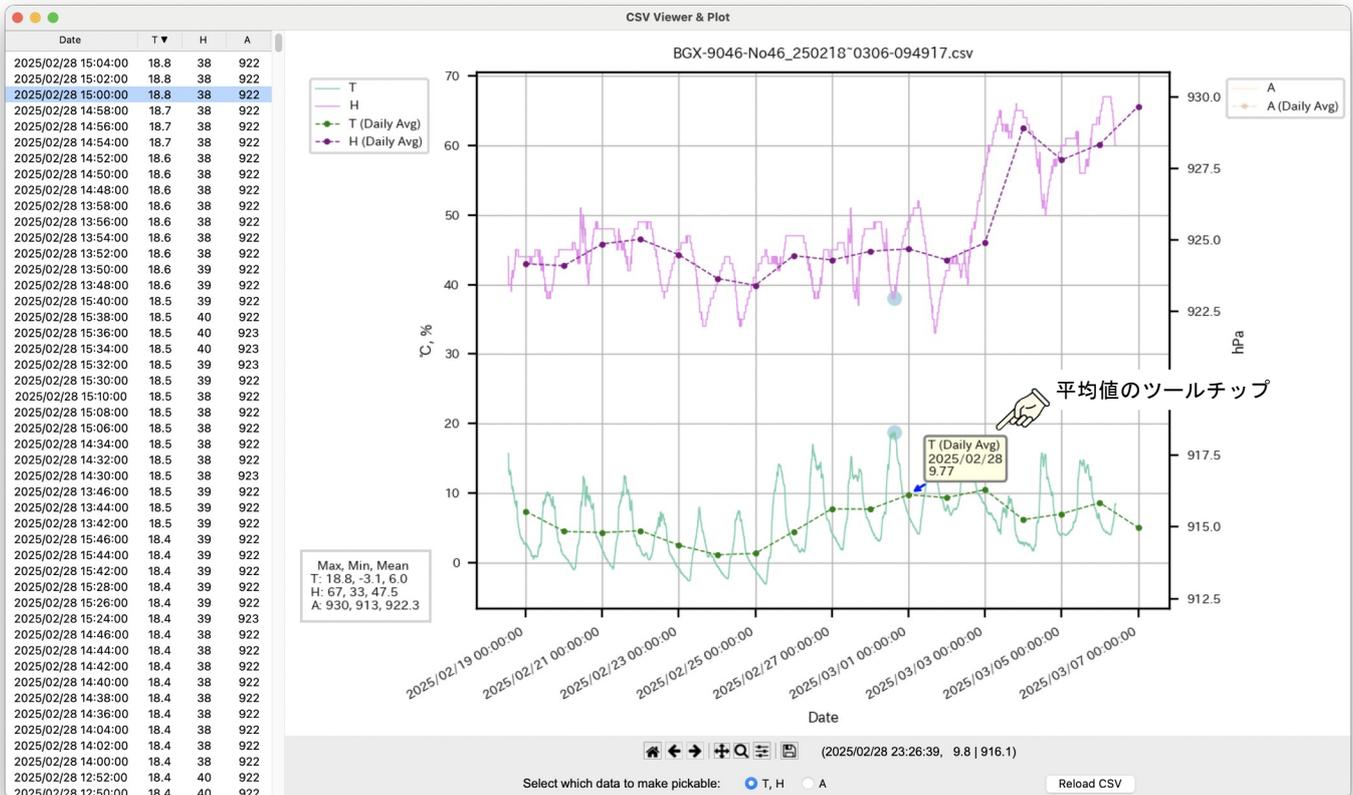
凡例クリックで線を表示/非表示



全体の最高、最低、平均値

チャートのtoolbar

・気圧を非表示



平均値のツールチップ

■ GPSウェイポイントを地図上に表示

・ ChatGPTへの指示

pythonでウェイポイントファイル .kml を読み込み地図に表示する

・ 出力されたコード

```
import folium
import xml.etree.ElementTree as ET

# KMLファイルの読み込み
def parse_kml(kml_file):
    tree = ET.parse(kml_file)
    root = tree.getroot()
    namespaces = {'kml': 'http://www.opengis.net/kml/2.2'}

    # KMLファイル内のすべてのポイント (<Placemark>タグ) の座標を抽出
    coordinates = []
    for placemark in root.findall('./kml:Placemark', namespaces):
        point = placemark.find('./kml:Point/kml:coordinates', namespaces)
        if point is not None:
            # 座標を抽出してリストに追加
            coords = point.text.strip().split(',')
            lon, lat = float(coords[0]), float(coords[1])
            coordinates.append((lat, lon))
    return coordinates

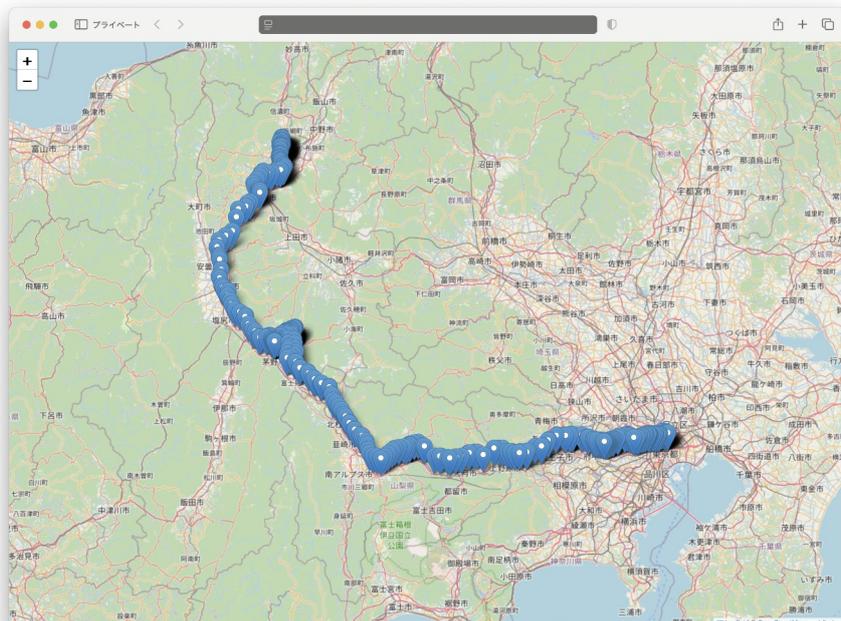
# 地図にプロット
def plot_on_map(coordinates):
    # 地図の中心を最初の座標に設定
    if coordinates:
        m = folium.Map(location=coordinates[0], zoom_start=12)

        # 各ウェイポイントを地図にプロット
        for lat, lon in coordinates:
            folium.Marker([lat, lon]).add_to(m)

        # 地図をHTMLとして保存
        m.save('map.html')
        print('地図をmap.htmlとして保存しました。')

# KMLファイルを指定して処理
kml_file = '/sample_wp.kml' # ここにKMLファイルのパスを指定
coordinates = parse_kml(kml_file)
plot_on_map(coordinates)
```

・ 保存されたhtmlをSafariで実行



OpenStreetMap (オープンソースの無料で使える地図データ)

■ GPSトラックを地図上に表示

・ ChatGPTへの指示

pythonでトラックデータファイル .kml を読み込み地図に表示する

・ 出力されたコード

```
import folium
import xml.etree.ElementTree as ET

# KMLファイルを読み込んでトラックデータを解析
def parse_kml(kml_file):
    tree = ET.parse(kml_file)
    root = tree.getroot()
    namespaces = {'kml': 'http://www.opengis.net/kml/2.2'}

    # KMLファイル内の<LineString>タグから座標を抽出
    coordinates = []
    for placemark in root.findall('..//kml:Placemark', namespaces):
        line_string = placemark.find('..//kml:LineString/kml:coordinates', namespaces)
        if line_string is not None:
            # 座標を抽出してリストに追加
            coords = line_string.text.strip().split()
            for coord in coords:
                lon, lat, _ = map(float, coord.split(','))
                coordinates.append((lat, lon))
    return coordinates

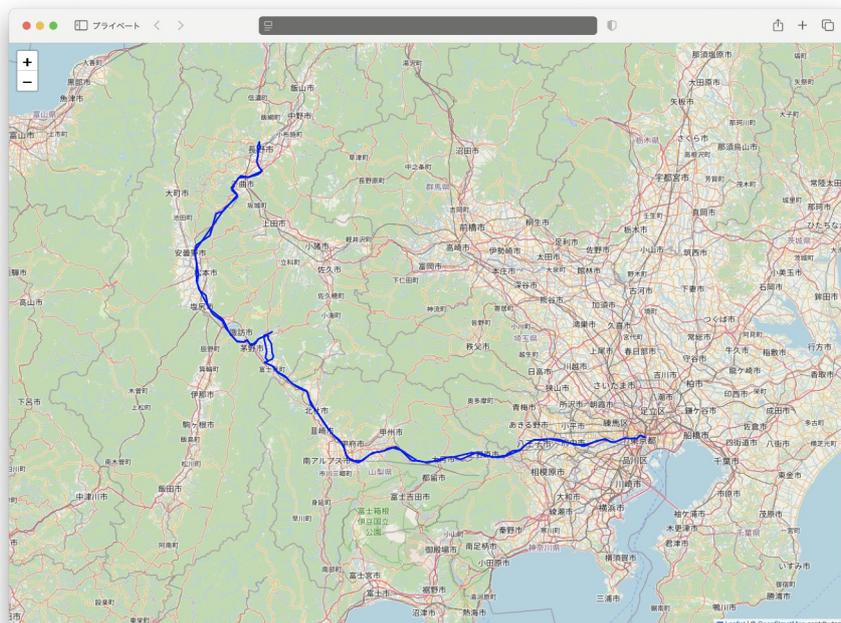
# 地図にトラックをプロット
def plot_on_map(coordinates):
    # 地図の中心を最初の座標に設定
    if coordinates:
        m = folium.Map(location=coordinates[0], zoom_start=12)

        # トラックの線を描画
        folium.PolyLine(coordinates, color="blue", weight=2.5, opacity=1).add_to(m)

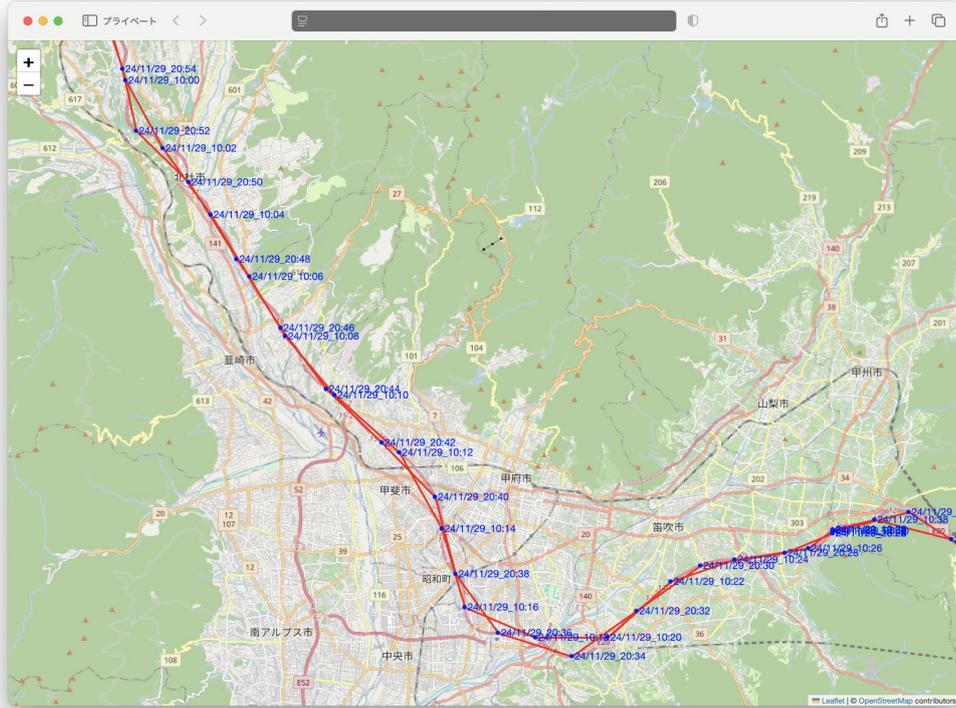
        # 地図をHTMLとして保存
        m.save('track_map.html')
        print('トラックを地図に表示しました。track_map.htmlとして保存されました。')

# KMLファイルを指定して処理
kml_file = '/sample_tr.kml' # ここにKMLファイルのパスを指定
coordinates = parse_kml(kml_file)
plot_on_map(coordinates)
```

・ 保存されたhtmlをSafariで実行



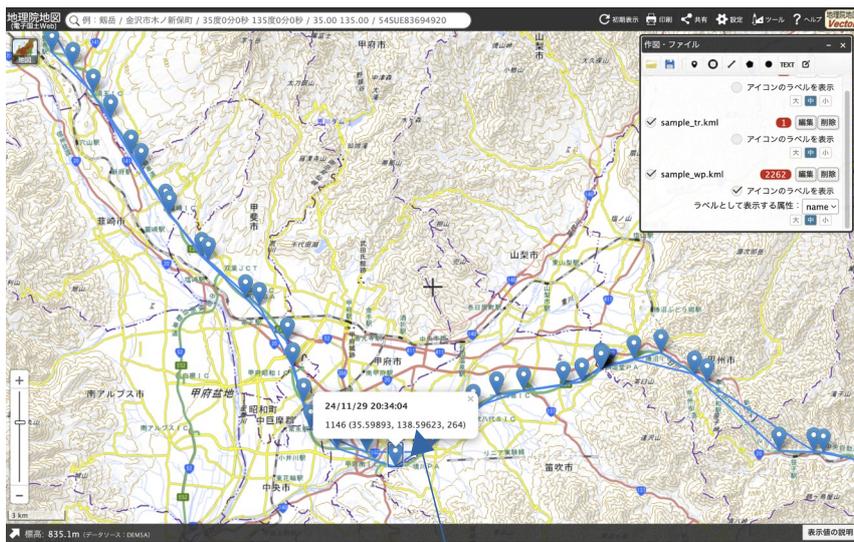
・上記を踏まえて、ウェイポイントとトラック(赤線)を重ねて表示。
 ウェイポイントはアイコンの代わりにタイムスタンプとしました。



(部分拡大)

・ちなみに、国土地理の地図でタイムスタンプを表示しようとしたけれど...

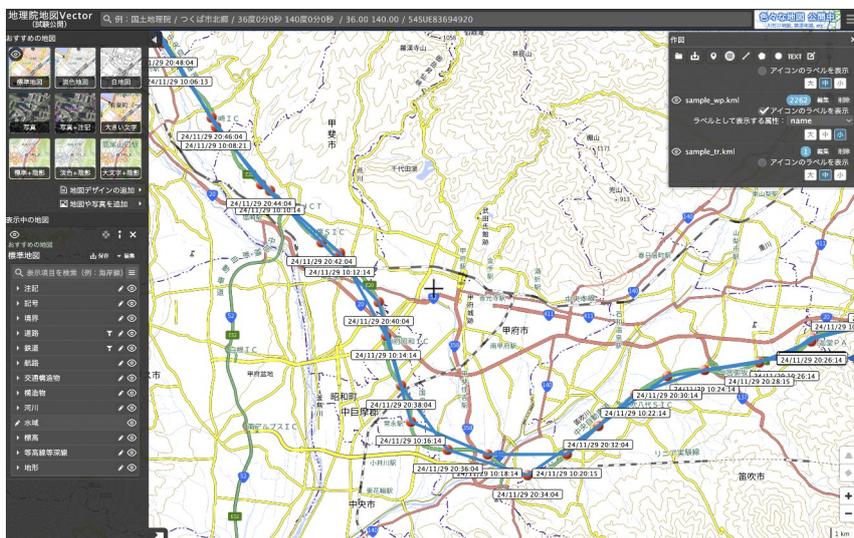
国土地理Vectorへジャンプ



アイコンをクリックすればpopupする

国土地理GSI Mapsでウェイポイントの「アイコンのラベルを表示」をチェックしたが、切り替わらない。

「Vector」でウェイポイントを読み込むと<name>タグを表示した。



✓アイコンのラベルを表示
属性:name

タイムスタンプ

■ jupyter notebook python (macにインストール)

jupyter notebookは、インタラクティブなデータ分析やコード実行、文書作成を行うためのツールです。Pythonなどのプログラミング言語を使用し、コードセルとテキストセルを組み合わせ、データの解析結果やグラフをリアルタイムで表示しながら作業できます。主にデータサイエンスや機械学習の分野で利用されます。(ChatGPTより)

・ Jupyter Notebookで作成したGPSマップ

CSVとウェイポイントkml、トラックkmlを読み込み、時間リスト、地図とチャートを作成します。

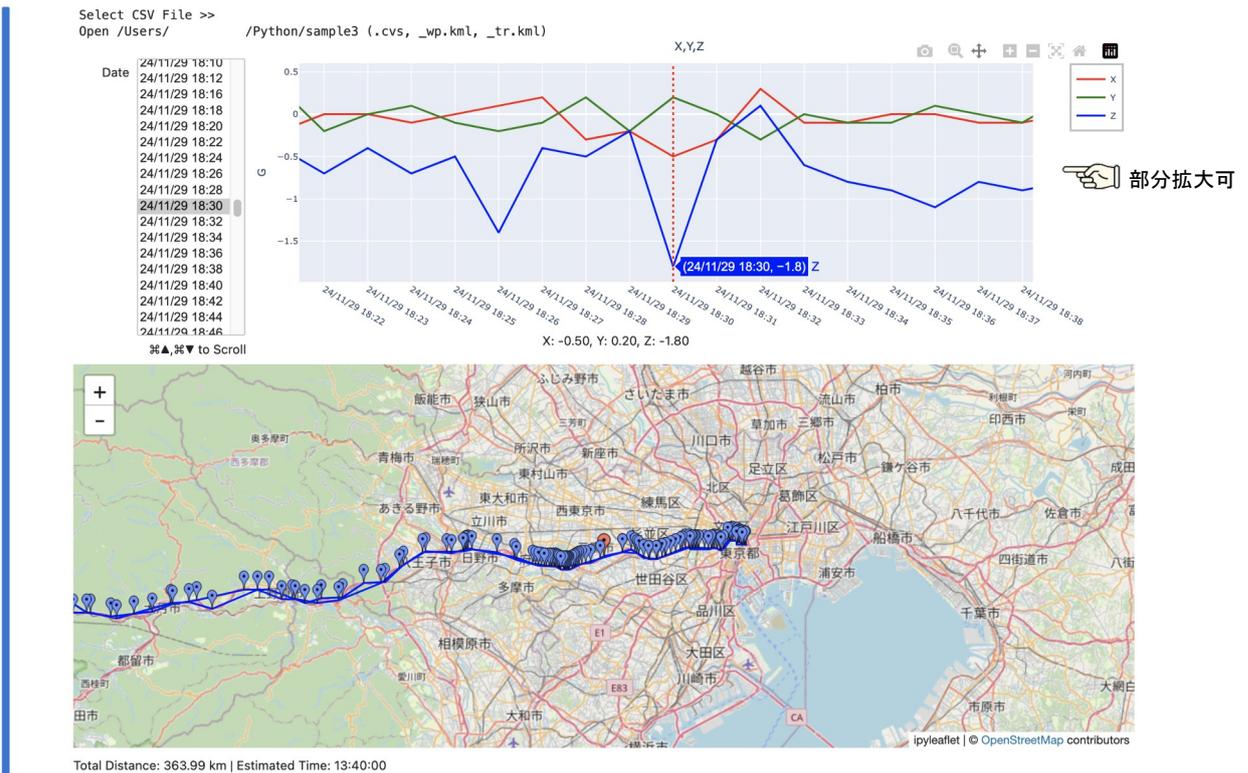
ファイルはダイアログで選択。ローカルファイルを扱えます。

リストまたはアイコンのクリックでチャートの縦線を移動します。

・ 温度、湿度、気圧 バージョン



・ X, Y, Zバージョン



■ Google Colaboratory

Google Colaboratory (Google Colab) は、クラウドベースのJupyter Notebook環境で、Pythonコードをブラウザ上で実行できるツールです。Googleアカウントで無料で使用でき、GPUやTPUも利用可能です。データサイエンスや機械学習の実験に便利です。(ChatGPTより)

- ・ チャート描いてみました

ローカルファイルを直接読めないで、ファイルオープンダイアログでローカルファイルを選択しドライブにアップロード→読み込み→ラジオボタンでカラム選択→チャートボタンで描画します。

※iPadはオープンダイアログからアップロードできないようなので別途ドライブにアップしてファイルパスを直接指定。

- ・ Colabでフーリエ変換

Colabへの指示

csvファイルを読み込んでフーリエ変換をしてグラフを描く。カラムは"X"、データは1秒で1024個とする。グラフで表示するのは160個。最大周波数を表示する。

出力されたコード

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from google.colab import files
```

```
# ファイルのアップロード
uploaded = files.upload()
```

```
# アップロードされたCSVファイルの読み込み
for fn in uploaded.keys():
    df = pd.read_csv(fn)
```

```
# 'X'列のデータを取得
try:
    x_data = df['X']
except KeyError:
    print("Error: Column 'X' not found in the CSV file.")
    exit()
```

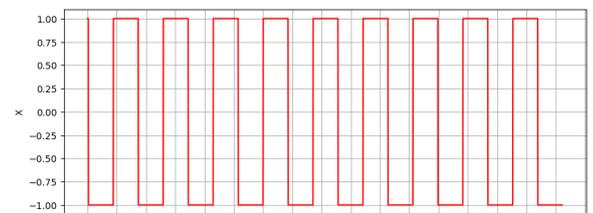
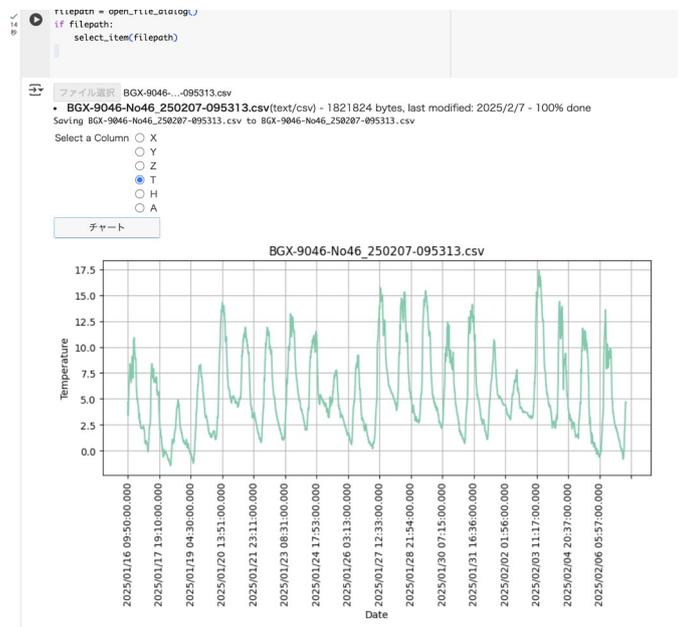
```
# 高速フーリエ変換
fft_result = np.fft.fft(x_data)
```

```
# 周波数軸の作成 (サンプリングレートは1024Hzと仮定)
sampling_rate = 1024
frequencies = np.fft.fftfreq(len(x_data), 1/sampling_rate)
```

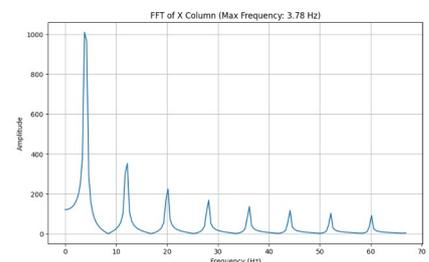
```
# 振幅スペクトルを計算
amplitude_spectrum = np.abs(fft_result)
```

```
# 最大周波数を見つける
max_frequency_index = np.argmax(amplitude_spectrum[:len(frequencies)//2]) # 正の周波数成分のみを考慮
max_frequency = frequencies[max_frequency_index]
```

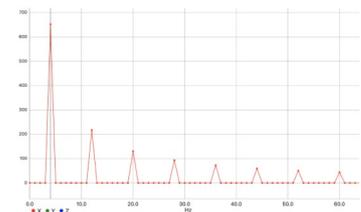
```
# グラフ描画
plt.figure(figsize=(10, 6))
plt.plot(frequencies[:160], amplitude_spectrum[:160]) # 最初の160個の周波数成分のみ表示
plt.xlabel('Frequency (Hz)')
plt.ylabel('Amplitude')
plt.title(f'FFT of X Column (Max Frequency: {max_frequency:.2f} Hz)') # 最大周波数をタイトルに表示
plt.grid(True)
plt.show()
```



元の波形

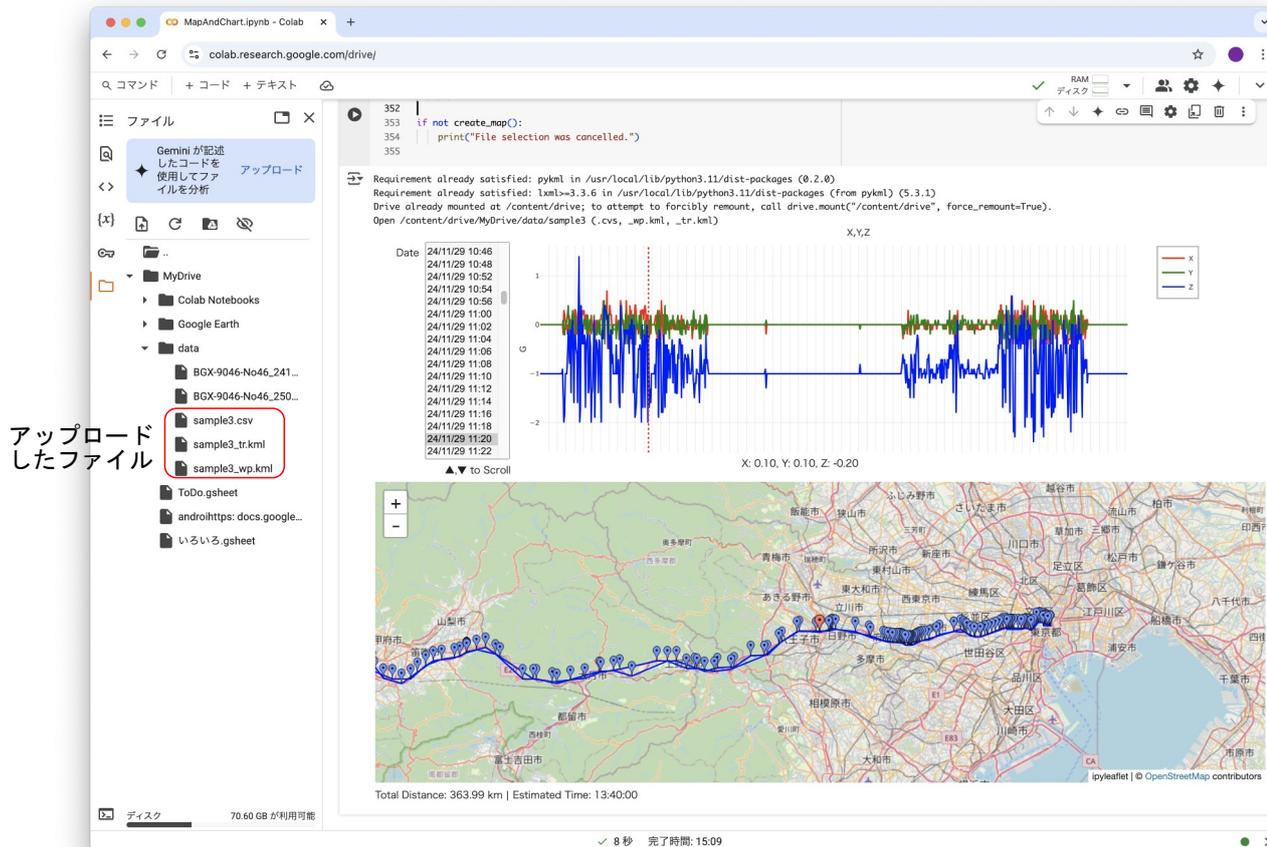


Colabの出力

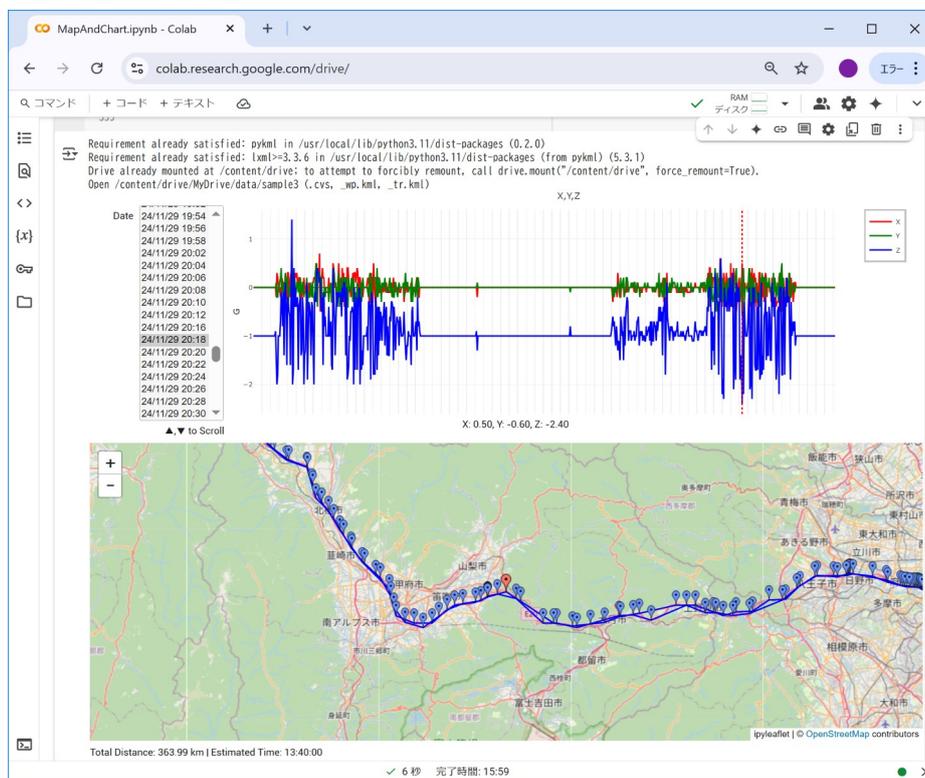


ちなみにJvqBleでは

・前述の jupyter notebook で作成した時間vs地図スクリプトを Google Colab で実行してみました。MyDrive にアップロードしたファイルを読み込みます。多少の修正をしましたが jupyter notebook と同様に動作します。



Colabはブラウザ上で動作するので Windows からでもOkです。



株式会社アクセル

391-0005 長野県茅野市仲町16-32 トウビル5F

PHONE 0266-72-8465

FAX 0266-72-8436

ACCEL
ACCEL Corporation

<http://jvq.jp/>

2025/03/26