**Inputs Required**

1. **Tower (Antenna) Coordinates** – (Lat, Lon)
2. **Antenna Height** – in meters above ground
3. **MR Data Grid** – Each MR point with:
   * Latitude, Longitude
   * RSRP (Reference Signal Received Power)
   * Distance from tower (can be computed)
   * Elevation (from DEM or external elevation API)
4. **Cost-Hata Model** – To simulate signal path loss vs. distance
5. **Antenna Azimuth and Beamwidth (optional)** – for fine-tuning directional analysis

**📐 Goal**

Estimate the **vertical tilt** (usually in degrees) of the antenna that best explains the **serving MR distribution** and **elevation differences** between the antenna and UE location.

**🧮 Step-by-Step Algorithm**

1. **Compute Elevation Angle for Each MR Point**

θmeasured=arctan⁡(hantenna−hMR pointdground)\theta\_{\text{measured}} = \arctan\left(\frac{h\_{\text{antenna}} - h\_{\text{MR point}}}{d\_{\text{ground}}}\right)θmeasured​=arctan(dground​hantenna​−hMR point​​)

where:

* + hantennah\_{\text{antenna}}hantenna​: Tower elevation + antenna height
  + hMR pointh\_{\text{MR point}}hMR point​: Ground elevation at UE point (from DEM or API)
  + dgroundd\_{\text{ground}}dground​: Horizontal distance from tower to MR point (use Haversine formula)

1. **Filter MR Points** (optional):
   * Only consider MR points with RSRP > threshold (e.g., -110 dBm)
   * Exclude outliers or points not in main lobe of antenna (based on azimuth ± half beamwidth)
2. **Get Average (or Peak Density) Elevation Angle**
   * Aggregate by either **average elevation angle** weighted by RSRP
   * Or take the **mode/peak angle** where most strong MRs are clustered
3. **Apply Correction Based on Cost-Hata Model**
   * Simulate path loss for various tilt values using Cost-Hata
   * Compare simulated RSRP with actual MR RSRP for validation
4. **Estimate Effective Tilt**

θestimated≈mean or median of θmeasured\theta\_{\text{estimated}} \approx \text{mean or median of } \theta\_{\text{measured}}θestimated​≈mean or median of θmeasured​

**📉 Optional Refinement**

* Use **grid-based interpolation** to smooth the elevation profile
* Fit a **Gaussian distribution** over the tilt angles to get the peak
* Use **optimization (e.g., least squares)** to minimize the difference between simulated and observed RSRP for various tilt values

**🧠 What You Can Output**

* **Estimated Tilt (degrees)** for each sector
* **Tilt heatmap** overlayed on map
* **Comparison of observed vs simulated RSRP** with the inferred tilt

**✅ Final Notes**

* **Mechanical Tilt** cannot be directly estimated unless physical records are available, but **Electrical Tilt** is dominant in most networks today.
* The **accuracy improves** with more MR samples and elevation data precision.

[Skip to content](https://chatgpt.com/c/683585c4-ac50-8007-8c39-5bfaf522a4c5#main)

**Chat history**

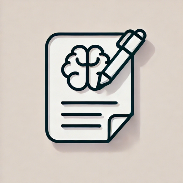
[New chat](https://chatgpt.com/)

Search chats

Ctrl K

[Library](https://chatgpt.com/library)

[Presentaion maker](https://chatgpt.com/g/g-6767e8ffa1448191a54dc18bece681eb-presentaion-maker)

[[](https://chatgpt.com/g/g-676c1d825b3081919b8c5ed04507a11f-ai-multimodal-research-assistant)](https://chatgpt.com/g/g-676c1d825b3081919b8c5ed04507a11f-ai-multimodal-research-assistant)

[AI Multimodal Research Assistant](https://chatgpt.com/g/g-676c1d825b3081919b8c5ed04507a11f-ai-multimodal-research-assistant)

[Thesis Writer GPT](https://chatgpt.com/g/g-676c1f76bbdc8191a1c8c22951b645bf-thesis-writer-gpt)

[[](https://chatgpt.com/g/g-B3hgivKK9-write-for-me)](https://chatgpt.com/g/g-B3hgivKK9-write-for-me)

[Write For Me](https://chatgpt.com/g/g-B3hgivKK9-write-for-me)

[SZAKTRADER BOT](https://chatgpt.com/g/g-67c0c0dcb8c88191b1a915a51a8ad879-szaktrader-bot)

[[](https://chatgpt.com/g/g-alKfVrz9K-canva)](https://chatgpt.com/g/g-alKfVrz9K-canva)

[Canva](https://chatgpt.com/g/g-alKfVrz9K-canva)

[GPTs](https://chatgpt.com/gpts)

[Sora](https://sora.chatgpt.com/?utm_source=chatgpt" \t "_blank)

New project

[Asaad\_Thesis](https://chatgpt.com/g/g-p-675d516da15c8191a9fa2241574e911b-asaad-thesis/project)

[latest](https://chatgpt.com/g/g-p-67fcedf8a35881919946df962de4f9fa-latest/project)

Today

[Antenna Tilt Calculation MR Data](https://chatgpt.com/c/683585c4-ac50-8007-8c39-5bfaf522a4c5)

[Assignment Criteria Evaluation](https://chatgpt.com/c/68358052-f464-8007-b901-ff1e2c31b425)

[Entrepreneurship Assignment Criteria](https://chatgpt.com/c/68354eee-6220-8007-bdc8-187d053afe36)

[Telecom Sector Fusion Demo](https://chatgpt.com/c/683564cb-40cc-8007-8b30-bc5e51f694d7)

[Job Opportunity Inquiry UAE](https://chatgpt.com/c/683568d8-d238-8007-b79d-9742a3eb7476)

[Percentage Improvement Calculation](https://chatgpt.com/c/6834a532-6a14-8007-b159-aaf5e2910e3e)

Yesterday

[RTP Stream VLC Issue](https://chatgpt.com/c/6834ace5-fb20-8007-8dd5-adf47b950305)

[Negative Number Comparison](https://chatgpt.com/c/68349d4b-f580-8007-8648-c958e0c08bb5)

[EC2 User Request Capacity](https://chatgpt.com/c/68345981-fc04-8007-87db-eb7a56d3d232)

[GPT-4 API Pricing](https://chatgpt.com/c/68345121-083c-8007-912c-f12f943ab621)

[Final POC Sector Fusion](https://chatgpt.com/c/683443a3-bbf4-8007-ac76-02b6552a55c4)

[Agile Loop Product Overview](https://chatgpt.com/c/68342bd7-e798-8007-97ac-1a6e88f7ca4b)

[Questions for Beyond Search](https://chatgpt.com/c/68341f2e-f534-8007-b3d4-50fcadfa8802)

[HUMA Airdrop Analysis](https://chatgpt.com/c/6833997d-dcdc-8007-94ce-3050e0b91870)

Previous 7 Days

[CSV Row Comparison](https://chatgpt.com/c/6832ffe3-2a08-8007-84f0-6e87fa5edf17)

[Huma Finance Overview](https://chatgpt.com/c/6832dce0-9418-8007-a490-6a577793f5d3)

[Command syntax issue](https://chatgpt.com/c/6831f68e-1b74-8007-b169-4155d2c99d1b)

[Ikralic aur Lime ka Mix](https://chatgpt.com/c/68317416-d23c-8007-9ed5-b9ee38fcd896)

[OOP Basics in Python](https://chatgpt.com/c/68300b4e-5bc4-8007-ac8e-055a206428da)

[BERTScore Text Preprocessing](https://chatgpt.com/c/6830b263-3794-8007-a67d-ea4ecfae6617)

[CUDA OOM Fix](https://chatgpt.com/c/682f6ed0-02d0-8007-8d52-2edf864959f8)

[CSV strict comparison](https://chatgpt.com/c/682f5898-3688-8007-83e1-b3323f177247)

[B1 vs C1 Schwierigkeitsgrad](https://chatgpt.com/c/682f29b3-2184-8007-8583-020776e872d7)

[5505 Project Plan Rubric](https://chatgpt.com/c/682ed49e-caec-8007-9937-3dd1456d6324)

[RadCliQ Performance Explanation](https://chatgpt.com/c/682e48fe-b94c-8007-ae96-85377ef80d60)

[Harvard to IEEE Citation](https://chatgpt.com/c/68250234-8810-8007-9aa5-c53b57aa66f2)

[Self Cleaning Function Explained](https://chatgpt.com/c/682e3fbd-3658-8007-bf5b-3a54087060c3)

[BLEU2 for Medical Language](https://chatgpt.com/c/682dc486-b9b0-8007-85b1-dd33c0009db0)

View plans

Unlimited access, team features, and more

ChatGPT 4o

Share



PLUS

**You said:**

if I have Antenna Height , MR data for each 4G Cell per carreir can you calcuate the Tilt of the antenna from MR Data i am using cost hatta propagation model with also analyzing elevation by MR Grid points and then caluculate the exact tilt of the antenna also Tower coordinates can be provide from where antenna is serving.

**ChatGPT said:**

Yes, you can estimate the **electrical tilt (or total tilt)** of a 4G antenna using **MR (Measurement Report) data**, **antenna height**, **tower (site) coordinates**, and **Cost-Hata propagation model**, especially if you're analyzing **elevation differences using MR grid points**. Here’s a breakdown of how you can calculate the antenna tilt:

**🔧 Inputs Required**

1. **Tower (Antenna) Coordinates** – (Lat, Lon)
2. **Antenna Height** – in meters above ground
3. **MR Data Grid** – Each MR point with:
   * Latitude, Longitude
   * RSRP (Reference Signal Received Power)
   * Distance from tower (can be computed)
   * Elevation (from DEM or external elevation API)
4. **Cost-Hata Model** – To simulate signal path loss vs. distance
5. **Antenna Azimuth and Beamwidth (optional)** – for fine-tuning directional analysis

**📐 Goal**

Estimate the **vertical tilt** (usually in degrees) of the antenna that best explains the **serving MR distribution** and **elevation differences** between the antenna and UE location.

**🧮 Step-by-Step Algorithm**

1. **Compute Elevation Angle for Each MR Point**

θmeasured=arctan⁡(hantenna−hMR pointdground)*θ*measured​=arctan(*d*ground​*h*antenna​−*h*MR point​​)

where:

* + hantenna*h*antenna​: Tower elevation + antenna height
  + hMR point*h*MR point​: Ground elevation at UE point (from DEM or API)
  + dground*d*ground​: Horizontal distance from tower to MR point (use Haversine formula)

1. **Filter MR Points** (optional):
   * Only consider MR points with RSRP > threshold (e.g., -110 dBm)
   * Exclude outliers or points not in main lobe of antenna (based on azimuth ± half beamwidth)
2. **Get Average (or Peak Density) Elevation Angle**
   * Aggregate by either **average elevation angle** weighted by RSRP
   * Or take the **mode/peak angle** where most strong MRs are clustered
3. **Apply Correction Based on Cost-Hata Model**
   * Simulate path loss for various tilt values using Cost-Hata
   * Compare simulated RSRP with actual MR RSRP for validation
4. **Estimate Effective Tilt**

θestimated≈mean or median of θmeasured*θ*estimated​≈mean or median of *θ*measured​

**📉 Optional Refinement**

* Use **grid-based interpolation** to smooth the elevation profile
* Fit a **Gaussian distribution** over the tilt angles to get the peak
* Use **optimization (e.g., least squares)** to minimize the difference between simulated and observed RSRP for various tilt values

**🧠 What You Can Output**

* **Estimated Tilt (degrees)** for each sector
* **Tilt heatmap** overlayed on map
* **Comparison of observed vs simulated RSRP** with the inferred tilt

**✅ Final Notes**

* **Mechanical Tilt** cannot be directly estimated unless physical records are available, but **Electrical Tilt** is dominant in most networks today.
* The **accuracy improves** with more MR samples and elevation data precision.

Would you like me to help you write the **Python code** to implement this logic using Pandas, NumPy, and Geo libraries (like geopy, elevation, or rasterio)?

**You said:**

yes

**ChatGPT said:**

Updated saved memory

Great! Below is a step-by-step Python pipeline to calculate antenna tilt using MR data, tower coordinates, antenna height, and the Cost-Hata model:

**✅ 1. Install Required Libraries**

bash

CopyEdit

pip install pandas numpy geopy elevation srtm matplotlib

**✅ 2. Load Your MR Dataset**

python

CopyEdit

import pandas as pd *# Sample structure: each row is a Measurement Report point* *# Columns: MR\_Lat, MR\_Lon, RSRP, MR\_Altitude (optional), CellID* mr\_data = pd.read\_csv("mr\_data.csv") *# Tower site information* tower\_lat = 33.6844 tower\_lon = 73.0479 antenna\_height = 35 *# meters above ground* tower\_ground\_elevation = 500 *# meters (optional, else fetched)*

**✅ 3. Get Elevation of MR Points (If Missing)**

python

CopyEdit

import srtm elevation\_data = srtm.get\_data() mr\_data['MR\_Elevation'] = mr\_data.apply( lambda row: elevation\_data.get\_elevation(row['MR\_Lat'], row['MR\_Lon']), axis=1 )

**✅ 4. Compute Distance & Elevation Angle**

python

CopyEdit

from geopy.distance import geodesic import numpy as np def compute\_distance\_and\_angle(row): mr\_coord = (row['MR\_Lat'], row['MR\_Lon']) tower\_coord = (tower\_lat, tower\_lon) d = geodesic(tower\_coord, mr\_coord).meters mr\_elev = row['MR\_Elevation'] if pd.notnull(row['MR\_Elevation']) else 0 tower\_elev\_total = tower\_ground\_elevation + antenna\_height angle\_rad = np.arctan((tower\_elev\_total - mr\_elev) / d) angle\_deg = np.degrees(angle\_rad) return pd.Series({'Distance': d, 'ElevationAngle': angle\_deg}) mr\_data[['Distance', 'ElevationAngle']] = mr\_data.apply(compute\_distance\_and\_angle, axis=1)

**✅ 5. Estimate Tilt by Averaging Angles for Strong Signals**

python

CopyEdit

*# Filter strong MRs (optional)* strong\_mrs = mr\_data[mr\_data['RSRP'] > -105] *# Weighted average based on RSRP (stronger = more weight)* weighted\_tilt = np.average(strong\_mrs['ElevationAngle'], weights=strong\_mrs['RSRP'].apply(lambda x: 120 + x)) print(f"Estimated Antenna Tilt: {weighted\_tilt:.2f} degrees")

**✅ 6. Visualize the MR Points and Angles (Optional)**

python

CopyEdit

import matplotlib.pyplot as plt plt.scatter(mr\_data['Distance'], mr\_data['ElevationAngle'], c=mr\_data['RSRP'], cmap='viridis', s=10) plt.colorbar(label='RSRP (dBm)') plt.xlabel("Distance from Tower (m)") plt.ylabel("Elevation Angle (deg)") plt.title("Elevation Angle vs Distance") plt.grid(True) plt.show()

**✅ 7. (Optional) Validate with Cost-Hata Model**

We can simulate RSRP using Cost-Hata for a range of tilt angles and compare it with actual MR RSRP to find the best match. Let me know if you'd like to add this next.

Would you like me to:

1. Help you **integrate Cost-Hata simulation** to validate tilt?
2. Package this into a **single script or notebook**?
3. Help you adapt this to **multiple cells per site** in a loop?

Let me know!

**You said:**

yes validate

**ChatGPT said:**

Perfect — let’s **validate the estimated tilt** using the **Cost-Hata propagation model** by simulating the received signal power (RSRP) and comparing it with your MR data. We'll find the **best-fit tilt angle** by minimizing the difference between **simulated RSRP and actual RSRP** across your MR points.

**📦 Cost-Hata Model Recap (Urban/Suburban)**

The **Cost-Hata path loss formula** (in dB) is:

PL=69.55+26.16⋅log⁡10(f)−13.82⋅log⁡10(hb)−a(hm)+(44.9−6.55⋅log⁡10(hb))⋅log⁡10(d)*PL*=69.55+26.16⋅log10​(*f*)−13.82⋅log10​(*hb*​)−*a*(*hm*​)+(44.9−6.55⋅log10​(*hb*​))⋅log10​(*d*)

Where:

* f*f*: frequency in MHz (e.g., 1800 or 2100)
* hb*hb*​: base station antenna height (meters)
* hm*hm*​: mobile station height (usually 1.5m)
* d*d*: distance between tower and UE in km
* a(hm)=(1.1⋅log⁡10(f)−0.7)⋅hm−(1.56⋅log⁡10(f)−0.8)*a*(*hm*​)=(1.1⋅log10​(*f*)−0.7)⋅*hm*​−(1.56⋅log10​(*f*)−0.8)

Then estimate **RSRP** as:

RSRP=Ptx−PL*RSRP*=*Ptx*​−*PL*

Where Ptx*Ptx*​ is transmit power in dBm.

**✅ Add to the Script: Cost-Hata Simulation and Tilt Fitting**

python

CopyEdit

*# Constants* frequency\_mhz = 1800 *# adjust to your carrier* tx\_power\_dbm = 43 *# typical eNodeB power per antenna* mobile\_height = 1.5 *# UE height in meters* def mobile\_correction\_a(f, h\_m): return (1.1 \* np.log10(f) - 0.7) \* h\_m - (1.56 \* np.log10(f) - 0.8) def cost\_hata\_path\_loss(f\_mhz, d\_km, h\_b, h\_m): a\_hm = mobile\_correction\_a(f\_mhz, h\_m) pl = 69.55 + 26.16 \* np.log10(f\_mhz) - 13.82 \* np.log10(h\_b) - a\_hm + \ (44.9 - 6.55 \* np.log10(h\_b)) \* np.log10(d\_km) return pl *# Compute simulated RSRP for a range of tilt angles and find best match* angle\_range = np.arange(0, 15, 0.1) *# 0 to 15 degrees in 0.1 steps* best\_angle = None min\_error = float('inf') for tilt\_deg in angle\_range: simulated\_rsrps = [] for \_, row in strong\_mrs.iterrows(): d\_km = row['Distance'] / 1000.0 elevation\_angle = row['ElevationAngle'] *# Simple gain factor: stronger signal when beam aligns with elevation angle* gain\_factor = np.cos(np.radians(tilt\_deg - elevation\_angle)) \*\* 2 *# narrower beam ~ cos²* *# Apply path loss model* pl = cost\_hata\_path\_loss(frequency\_mhz, d\_km, antenna\_height, mobile\_height) rsrp\_sim = tx\_power\_dbm - pl + 10 \* np.log10(gain\_factor + 1e-6) *# avoid log(0)* simulated\_rsrps.append(rsrp\_sim) *# Calculate mean squared error with real RSRP* error = np.mean((np.array(simulated\_rsrps) - strong\_mrs['RSRP'].values) \*\* 2) if error < min\_error: min\_error = error best\_angle = tilt\_deg print(f"✅ Best-fit Antenna Tilt from Cost-Hata Simulation: {best\_angle:.2f}°")

**📊 Optional: Visualize the Fit**

python

CopyEdit

import matplotlib.pyplot as plt errors = [] for tilt\_deg in angle\_range: simulated\_rsrps = [] for \_, row in strong\_mrs.iterrows(): d\_km = row['Distance'] / 1000.0 elevation\_angle = row['ElevationAngle'] gain\_factor = np.cos(np.radians(tilt\_deg - elevation\_angle)) \*\* 2 pl = cost\_hata\_path\_loss(frequency\_mhz, d\_km, antenna\_height, mobile\_height) rsrp\_sim = tx\_power\_dbm - pl + 10 \* np.log10(gain\_factor + 1e-6) simulated\_rsrps.append(rsrp\_sim) error = np.mean((np.array(simulated\_rsrps) - strong\_mrs['RSRP'].values) \*\* 2) errors.append(error) plt.plot(angle\_range, errors) plt.xlabel("Tilt Angle (°)") plt.ylabel("Mean Squared Error") plt.title("Tilt Angle vs RSRP Error") plt.grid(True) plt.show()

**📍 Final Output**

You'll get:

* A **numerically validated tilt angle**
* A **visual error curve** showing optimal tilt
* Ability to apply this across multiple 4G cells

Would you like me to help:

* Wrap this into a **function or class** for many cells?
* Add a **geospatial map visualization** (e.g., using Folium or Geopandas)?
* Create a **Jupyter Notebook version** for easy analysis?

Let me know what’s next!

**You said:**

yes wrap it for many cells

**ChatGPT said:**

Great! Here's a **modularized Python function** that wraps the full antenna tilt estimation pipeline — including **elevation angle calculation**, **Cost-Hata simulation**, and **tilt optimization** — for **multiple 4G cells** using MR data.

**✅ Input Format Expectations**

Your MR dataset should contain at least the following columns:

* CellID
* MR\_Lat
* MR\_Lon
* RSRP (dBm)

You should also provide a dictionary with:

* Tower coordinates
* Antenna height
* Ground elevation (optional) per CellID

**🧱 Full Python Function**

python

CopyEdit

import pandas as pd import numpy as np from geopy.distance import geodesic import srtm *# Constants* frequency\_mhz = 1800 tx\_power\_dbm = 43 mobile\_height = 1.5 angle\_range = np.arange(0, 15, 0.1) *# Elevation fetcher* elevation\_data = srtm.get\_data() *# Path loss and antenna gain model* def mobile\_correction\_a(f, h\_m): return (1.1 \* np.log10(f) - 0.7) \* h\_m - (1.56 \* np.log10(f) - 0.8) def cost\_hata\_path\_loss(f\_mhz, d\_km, h\_b, h\_m): a\_hm = mobile\_correction\_a(f\_mhz, h\_m) pl = 69.55 + 26.16 \* np.log10(f\_mhz) - 13.82 \* np.log10(h\_b) - a\_hm + \ (44.9 - 6.55 \* np.log10(h\_b)) \* np.log10(d\_km) return pl def compute\_tilt\_for\_cell(cell\_df, tower\_lat, tower\_lon, antenna\_height, tower\_ground\_elev): *# Compute elevation of MR points if not provided* if 'MR\_Elevation' not in cell\_df.columns: cell\_df['MR\_Elevation'] = cell\_df.apply( lambda row: elevation\_data.get\_elevation(row['MR\_Lat'], row['MR\_Lon']), axis=1 ) *# Compute distance & angle* def compute\_distance\_angle(row): mr\_coord = (row['MR\_Lat'], row['MR\_Lon']) tower\_coord = (tower\_lat, tower\_lon) d = geodesic(tower\_coord, mr\_coord).meters tower\_total\_elev = tower\_ground\_elev + antenna\_height elev\_angle = np.degrees(np.arctan((tower\_total\_elev - row['MR\_Elevation']) / d)) return pd.Series({'Distance': d, 'ElevationAngle': elev\_angle}) cell\_df[['Distance', 'ElevationAngle']] = cell\_df.apply(compute\_distance\_angle, axis=1) strong\_mrs = cell\_df[cell\_df['RSRP'] > -105] best\_angle = None min\_error = float('inf') for tilt in angle\_range: simulated\_rsrps = [] for \_, row in strong\_mrs.iterrows(): d\_km = row['Distance'] / 1000 gain\_factor = np.cos(np.radians(tilt - row['ElevationAngle'])) \*\* 2 pl = cost\_hata\_path\_loss(frequency\_mhz, d\_km, antenna\_height, mobile\_height) rsrp\_sim = tx\_power\_dbm - pl + 10 \* np.log10(gain\_factor + 1e-6) simulated\_rsrps.append(rsrp\_sim) error = np.mean((np.array(simulated\_rsrps) - strong\_mrs['RSRP'].values) \*\* 2) if error < min\_error: min\_error = error best\_angle = tilt return best\_angle

**🧪 Usage Example for Multiple Cells**

python

CopyEdit

*# Load your full MR dataset* all\_mr\_data = pd.read\_csv("mr\_data.csv") *# Tower info per CellID* tower\_info = { "Cell\_001": {"lat": 33.68, "lon": 73.05, "height": 35, "elev": 500}, "Cell\_002": {"lat": 33.70, "lon": 73.02, "height": 30, "elev": 490}, *# Add more cells...* } tilt\_results = {} for cell\_id, info in tower\_info.items(): cell\_df = all\_mr\_data[all\_mr\_data["CellID"] == cell\_id].copy() if len(cell\_df) < 10: print(f"Skipping {cell\_id}, not enough MR points") continue tilt\_deg = compute\_tilt\_for\_cell( cell\_df, tower\_lat=info["lat"], tower\_lon=info["lon"], antenna\_height=info["height"], tower\_ground\_elev=info["elev"] ) tilt\_results[cell\_id] = round(tilt\_deg, 2) print(f"✅ {cell\_id}: Estimated Tilt = {tilt\_deg:.2f}°")

**📄 Output**

You’ll get a dictionary tilt\_results like:

python

CopyEdit

{ 'Cell\_001': 4.7, 'Cell\_002': 6.3, ... }

**🛠 Next Suggestions**

Would you like me to:

* Export results to Excel or CSV?
* Add parallel processing for faster performance?
* Provide code to fetch elevations via API (in case SRTM is too slow)?

Let me know what you'd like next!

**You said:**

how to get API for elevations

**ChatGPT said:**

To get **elevation data via API**, the most reliable and easy-to-use services are:

**✅ 1. Open-Elevation API (Free & Open Source)**

**🔗 API Endpoint**

bash

CopyEdit

https://api.open-elevation.com/api/v1/lookup?locations=LAT,LON

**🔁 Example (Python)**

python

CopyEdit

import requests def get\_elevation\_open\_elevation(lat, lon): url = f"https://api.open-elevation.com/api/v1/lookup?locations={lat},{lon}" response = requests.get(url) if response.status\_code == 200: return response.json()['results'][0]['elevation'] else: print(f"Failed to get elevation for ({lat}, {lon})") return None *# Example* elevation = get\_elevation\_open\_elevation(33.6844, 73.0479) print("Elevation (m):", elevation)

⚠️ **Rate Limits**: Free but may be slow or blocked if overused. For production use, consider self-hosting or switching to a paid API.

**✅ 2. Google Maps Elevation API (Paid after Free Quota)**

**🔗 Documentation**

https://developers.google.com/maps/documentation/elevation/start

**🔁 Setup**

1. Go to Google Cloud Console
2. Create a new project
3. Enable “Elevation API”
4. Get your **API key**

**🔁 Python Example**

python

CopyEdit

import requests def get\_elevation\_google(lat, lon, api\_key): url = f"https://maps.googleapis.com/maps/api/elevation/json?locations={lat},{lon}&key={api\_key}" response = requests.get(url) if response.status\_code == 200: return response.json()['results'][0]['elevation'] else: print(f"Error: {response.text}") return None *# Example usage* api\_key = "YOUR\_GOOGLE\_API\_KEY" elevation = get\_elevation\_google(33.6844, 73.0479, api\_key) print("Elevation (m):", elevation)

💰 **Pricing**: First 1,000 requests/day are free. After that, it's $5/1,000 calls.

**✅ 3. OpenTopoData (Free & Supports Multiple Elevation Models)**

**🔗 API Endpoint**

bash

CopyEdit

https://api.opentopodata.org/v1/srtm90m?locations=LAT,LON

**🔁 Example**

python

CopyEdit

def get\_elevation\_opentopo(lat, lon): url = f"https://api.opentopodata.org/v1/srtm90m?locations={lat},{lon}" response = requests.get(url) if response.status\_code == 200: return response.json()['results'][0]['elevation'] else: print(f"Failed for ({lat}, {lon})") return None *# Example* elevation = get\_elevation\_opentopo(33.6844, 73.0479) print("Elevation (m):", elevation)

🟢 **Free and reliable**. SRTM90, ASTER, and other datasets supported.

**🧩 Recommendation**

| **Use Case** | **Recommended API** |
| --- | --- |
| Quick Free Test | Open-Elevation API |
| High Accuracy | Google Elevation API |
| Batch / Open | OpenTopoData (SRTM) |

Would you like help integrating one of these APIs into your tilt calculation pipeline?

Top of Form

Tools

Bottom of Form

ChatGPT can make mistakes. Check important info.