WORLD BANK GROUP

Manual de explicación del código

Versión 4

Assessment of the Small-scale hydropower potential in Colombia

13 febrero 2026

MINSAIT EasyHydro GRUPO ELEMENTAL

indracompany.com

Contenido

1. Introducción

En este documento se presenta el código fuente completo del script Hydro.py con explicaciones detalladas de cada sección. El objetivo es facilitar la comprensión del funcionamiento interno de la herramienta, permitiendo a desarrolladores y técnicos entender, modificar y mantener el código.

El documento está organizado en formato de dos columnas:

- **Columna izquierda (fondo oscuro):** Código Python exacto tal como aparece en el archivo, con números de línea indicados en el encabezado de cada sección.
- **Columna derecha (fondo claro):** Explicación detallada de qué hace cada parte del código, incluyendo el propósito de cada variable, función y estructura, así como ejemplos prácticos de cálculo.

Nota: Este documento cubre la parte de configuración y funciones de cálculo (líneas 1-289). El resto del código (generación del mapa HTML, JavaScript interactivo, estilos CSS) representa ~3,200 líneas adicionales que siguen la misma lógica modular.

2. Explicación del Código principal

ENCABEZADO - DOCUMENTACIÓN DEL MÓDULO	EXPLICACIÓN

"""
 	
Mapa de Colombia - Filtros selectivos
 	
- Filtros: Caudal (0.15-0.50), Pendiente (>0.05)
 	
- VSS (Viviendas Sin Servicio): - empieza en 0
 	
- Distancia al municipio: - empieza en máximo
 	
- FILTRO ESPACIAL: Solo Parques Arqueológicos y Parques Nacionales (PNN)
 	
- Las demás capas solo se muestran visualmente (NO excluyen puntos)
 	
- CALCULA: Distancia a la capital MÁS CERCANA (puede ser de otro departamento)
 	
"""
	DOCSTRING DEL MÓDULO PYTHON
	
	El texto entre """ es la documentación oficial del módulo.
	
	CONTENIDO:
	• Nombre: "Mapa de Colombia - Filtros selectivos"
	
	• Filtros disponibles:
	- Caudal: 0.15 a 0.50 m³/s
	- Pendiente: mayor a 0.05 (5%)
	
	• Filtro espacial:
	SOLO excluye puntos en:
	- Parques Arqueológicos
	- Parques Nacionales (PNN)
	Las demás capas son solo visuales.
	
	• Cálculo especial:
	La distancia se calcula a la capital MÁS CERCANA, que puede ser de otro departamento.
	
	Este texto aparece al ejecutar:
	>>> import Hydro
	>>> help(Hydro)

IMPORTACIÓN DE LIBRERÍAS	EXPLICACIÓN

`import os`
`import geopandas as gpd`
`import folium`
`from folium import plugins`
`import requests`
`import pandas as pd`
`import json`
`import math`
`import warnings`
`import urllib3`
`import zipfile`
	LIBRERÍAS EXTERNAS DE PYTHON
	
	import geopandas as gpd
	→ Extiende pandas para datos geoespaciales
	→ Lee shapefiles (.shp, .dbf, .shx, .prj)
	→ Operaciones espaciales: within, intersects, buffer
	→ Reproyección entre sistemas de coordenadas
	→ Alias "gpd" para escribir menos
	
	import folium
	→ Crea mapas web interactivos
	→ Genera HTML con Leaflet.js embebido
	→ Añade marcadores, polígonos, popups
	→ Soporta múltiples capas base
	
	from folium import plugins
	→ Extensiones adicionales de Folium
	→ Search: búsqueda de lugares
	→ MeasureControl: medir distancias
	→ MiniMap: minimapa de referencia
	
	import requests
	→ Cliente HTTP para Python
	→ Métodos GET, POST, etc.
	→ Descarga datos de APIs REST
	→ Manejo de headers y timeouts
	
	import pandas as pd
	→ Manipulación de datos tabulares
	→ DataFrames (tablas 2D)
	→ Operaciones con columnas
	→ Alias "pd"
	
	import json
	→ Serialización JSON
	→ Convierte dict Python → string JSON
	→ Necesario para pasar datos a JavaScript
	
	import math
	→ Funciones matemáticas
	→ sin, cos, sqrt, atan2, radians
	→ Usadas en fórmula de Haversine
	
	import warnings
	import urllib3
	→ Control de mensajes de advertencia
	→ Se usan para silenciar warnings

SUPRESIÓN DE ADVERTENCIAS	EXPLICACIÓN

`# Suprimir warnings`
`warnings.filterwarnings('ignore')`
`urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)`
	CONFIGURACIÓN DE WARNINGS
	
	# Suprimir warnings
	warnings.filterwarnings('ignore')
	
	→ Desactiva TODOS los warnings de Python
	→ Evita mensajes como:
	"DeprecationWarning: ..."
	"FutureWarning: ..."
	→ Hace la salida de consola más limpia
	
	urllib3.disable_warnings(
	urllib3.exceptions.InsecureRequestWarning
	)
	
	→ Desactiva warnings específicos de SSL
	→ El servidor de capas usa HTTPS pero su certificado no es verificable
	→ Sin esta línea, cada descarga mostraría:
	"InsecureRequestWarning: Unverified HTTPS request is being made..."

RUTA DEL ARCHIVO SHAPEFILE	EXPLICACIÓN

`# ======================================================================`
`# CONFIGURACIÓN`
`# ======================================================================`
 	
`RUTA_SHAPEFILE_PUNTOS = r"vss_2024.zip"`
	Ruta donde va a buscar el shapefile

NOMBRES DE COLUMNAS DEL SHAPEFILE	EXPLICACIÓN

# Nombres de columnas 	
COLUMNA_CAUDAL = "Caudal_med"
COLUMNA_PENDIENTE = "Pendiente"
COLUMNA_MUNICIPIO = "Municipio"
COLUMNA_VSS = "VSS"
COLUMNA_DISTANCIA = "Distancia1"
	MAPEO DE COLUMNAS - DATOS BÁSICOS
	
	# Nombres de columnas
	COLUMNA_CAUDAL = "Caudal_med"
	→ Columna con el caudal medio
	→ Unidad: metros cúbicos por segundo (m³/s)
	→ Usado para filtrar y calcular potencia
	
	COLUMNA_PENDIENTE = "Pendiente"
	→ Columna con la pendiente del terreno
	→ Valor decimal (0.05 = 5%)
	→ Usado para filtrar puntos
	
	COLUMNA_MUNICIPIO = "Municipio"
	→ Nombre del municipio donde está el punto
	→ Se muestra en el ventana emergente del mapa
	→ Texto libre
	
	COLUMNA_VSS = "VSS"
	→ Viviendas Sin Servicio eléctrico
	→ Número entero de viviendas
	→ Indica el beneficio social del proyecto
	
	COLUMNA_DISTANCIA = "Distancia1"
	→ Distancia al municipio más cercano
	→ Unidad: metros
	→ Afecta viabilidad del proyecto

COLUMNAS ADICIONALES DEL SHAPEFILE	EXPLICACIÓN

# Nombres de columnas 	
COLUMNA_CAUDAL = "Caudal_med"
COLUMNA_PENDIENTE = "Pendiente"
COLUMNA_MUNICIPIO = "Municipio"
COLUMNA_VSS = "VSS"
COLUMNA_DISTANCIA = "Distancia1"
COLUMNA_CAIDA = "Caida_hidr"
COLUMNA_POTENCIA_K = "Potencia_k"
COLUMNA_REGION = "Region"
COLUMNA_ZONA_CLIMA = "Zona_clima"
	MAPEO DE COLUMNAS - DATOS TÉCNICOS
	
	# Nombres de columnas
	(repetición de algunas + nuevas)
	
	COLUMNA_CAIDA = "Caida_hidr"
	→ Caída hidráulica en metros
	→ Diferencia de altura aprovechable
	→ H en la fórmula P = ρ × g × Q × H
	→ Determina qué turbinas son viables
	
	COLUMNA_POTENCIA_K = "Potencia_k"
	→ Potencia teórica en kW
	→ Calculada previamente en el shapefile
	→ Se ajusta con: P = Pk × 0.9 × η
	
	COLUMNA_REGION = "Region"
	→ Región de Colombia
	→ Valores posibles:
	"Región Pacífico"
	"Región Eje Cafetero – Antioquia"
	"Región Centro Sur"
	"Región Centro Oriente"
	"Región Caribe"
	"Región Llano"
	→ Afecta multiplicador de costes
	
	COLUMNA_ZONA_CLIMA = "Zona_clima"
	→ Zona climática del punto
	→ Puede afectar diseño y materiales
	→ Información complementaria

CONTROL DE DESCARGA DE CAPAS	EXPLICACIÓN

`# Control de capas`
 	
`DESCARGAR_CAPAS = True  # Cambiar a False si el servidor sig.cicolombiaenaccion.org tiene problemas`
	INTERRUPTOR DE CONEXIÓN A INTERNET
	
	# Control de capas
	DESCARGAR_CAPAS = True
	
	Esta variable booleana controla si el script descarga las capas geográficas desde el servidor de internet.
	
	═══════════
	VALOR: True (por defecto)
	═══════════
	• Descarga las 10 capas del servidor
	• Requiere conexión a internet activa
	• Primera ejecución tarda ~2-5 minutos
	• Las capas se descargan cada vez (no se guardan en caché)
	
	═══════════
	VALOR: False
	═══════════
	• NO descarga ninguna capa
	• El mapa funciona sin conexión
	• Los puntos se muestran normalmente
	• NO aparecen las áreas protegidas
	• NO se aplica filtro espacial
	
	¿CUÁNDO PONER False?
	• Sin conexión a internet
	• El servidor está caído o lento
	• Solo quieres probar los puntos
	• Estás depurando el código

URL DEL SERVIDOR DE CAPAS	EXPLICACIÓN
BASE_URL = "https://sig.cicolombiaenaccion.org/server/rest/services/Hosted/biodiversidad_ci_vista/FeatureServer"	DIRECCIÓN DEL SERVIDOR ARCGIS BASE_URL = "https://sig.cicolombiaenaccion.org/ server/rest/services/Hosted/ biodiversidad_ci_vista/FeatureServer" DESGLOSE DE LA URL: https://sig.cicolombiaenaccion.org → Dominio del servidor de CI Colombia → Conservación Internacional /server/rest/services → Ruta estándar de ArcGIS Server REST /Hosted/biodiversidad_ci_vista → Nombre del servicio de mapas → Contiene capas de biodiversidad /FeatureServer → Tipo de servicio: Feature Service → Permite consultar y descargar datos → Formato GeoJSON disponible CAPAS DISPONIBLES (0-10): Cada capa tiene un número ID que se usa para construir la URL de consulta: {BASE_URL}/1/query → Tierras Negras {BASE_URL}/2/query → Resguardos {BASE_URL}/3/query → Parques Arqueológicos ... etc. El servidor es público y gratuito pero puede estar lento o caído ocasionalmente.

CONFIGURACIÓN DE CAPAS GEOGRÁFICAS	EXPLICACIÓN

CAPAS_CONFIG = {
 	
'tierras_negras': {'id': 1, 'nombre': 'Tierras Comunidades Negras', 'color': '#8B4513', 'fill_opacity': 0.3},
 	
'resguardo_indigena': {'id': 2, 'nombre': 'Resguardos Indígenas', 'color': '#FF8C00', 'fill_opacity': 0.3},
 	
'parque_arqueologico': {'id': 3, 'nombre': 'Parques Arqueológicos', 'color': '#DAA520', 'fill_opacity': 0.4},
 	
'complejos_paramo': {'id': 4, 'nombre': 'Complejos de Páramo', 'color': '#87CEEB', 'fill_opacity': 0.4},
 	
'area_proteccion_local': {'id': 5, 'nombre': 'Áreas Protección Local', 'color': '#90EE90', 'fill_opacity': 0.3},
 	
'area_proteccion_regional': {'id': 6, 'nombre': 'Áreas Protección Regional', 'color': '#3CB371', 'fill_opacity': 0.3},
 	
'limite_rnsc': {'id': 7, 'nombre': 'Reservas Naturales (RNSC)', 'color': '#228B22', 'fill_opacity': 0.3},
 	
'limite_runap': {'id': 8, 'nombre': 'Áreas Protegidas (RUNAP)', 'color': '#006400', 'fill_opacity': 0.3},
 	
'limite_pnn': {'id': 9, 'nombre': 'Parques Nacionales (PNN)', 'color': '#004d00', 'fill_opacity': 0.4},
 	
'reservas_forestales': {'id': 10, 'nombre': 'Reservas Forestales', 'color': '#2F4F4F', 'fill_opacity': 0.3},
 	
}
	DICCIONARIO DE CAPAS A DESCARGAR
	
	CAPAS_CONFIG = { ... }
	
	Define las 10 capas geográficas que se descargan del servidor. Cada entrada:
	
	'clave_interna': {
	'id': número, ← ID en el servidor
	'nombre': 'texto', ← Nombre para mostrar
	'color': '#RRGGBB', ← Color hexadecimal
	'fill_opacity': 0.X ← Transparencia
	}
	
	═══════════
	CAPA 1: TIERRAS COMUNIDADES NEGRAS
	═══════════
	'tierras_negras': {id:1, color:'#8B4513'}
	Color: Marrón (SaddleBrown)
	→ Territorios colectivos de comunidades afrodescendientes (Ley 70 de 1993)
	→ Requiere consulta previa para proyectos
	→ NO es restrictiva (solo visual)
	
	═══════════
	CAPA 2: RESGUARDOS INDÍGENAS
	═══════════
	'resguardo_indigena': {id:2, color:'#FF8C00'}
	Color: Naranja oscuro (DarkOrange)
	→ Territorios de pueblos indígenas
	→ Propiedad colectiva inalienable
	→ Requiere consulta previa
	→ NO es restrictiva (solo visual)
	
	═══════════
	CAPA 3: PARQUES ARQUEOLÓGICOS ⚠️
	═══════════
	'parque_arqueologico': {id:3, color:'#DAA520'}
	Color: Dorado (GoldenRod)
	→ Sitios de patrimonio arqueológico
	→ San Agustín, Tierradentro, etc.
	⚠️ ES RESTRICTIVA: Elimina puntos dentro
	
	═══════════
	CAPA 4: COMPLEJOS DE PÁRAMO
	═══════════
	'complejos_paramo': {id:4, color:'#87CEEB'}
	Color: Azul cielo (SkyBlue)
	→ Ecosistemas de alta montaña
	→ Fuentes de agua, muy protegidos
	→ NO es restrictiva (solo visual)
	pero proyectos requieren permisos
	
	═══════════
	CAPA 5: ÁREAS PROTECCIÓN LOCAL
	═══════════
	'area_proteccion_local': {id:5, color:'#90EE90'}

Color: Verde claro (LightGreen)
→ Áreas protegidas por municipios
→ NO es restrictiva (solo visual)

═══════════════════════════════════════
CAPA 6: ÁREAS PROTECCIÓN REGIONAL
═══════════════════════════════════════
'area_proteccion_regional': {id:6, color:'#3CB371'}
Color: Verde medio (MediumSeaGreen)
→ Áreas de CARs (Corporaciones)
→ NO es restrictiva (solo visual)

═══════════════════════════════════════
CAPA 7: RESERVAS NATURALES (RNSC)
═══════════════════════════════════════
'limite_rnsc': {id:7, color:'#228B22'}
Color: Verde bosque (ForestGreen)
→ Reservas de la Sociedad Civil
→ Privadas, voluntarias
→ NO es restrictiva (solo visual)

═══════════════════════════════════════
CAPA 8: ÁREAS PROTEGIDAS (RUNAP)
═══════════════════════════════════════
'limite_runap': {id:8, color:'#006400'}
Color: Verde oscuro (DarkGreen)
→ Registro Único de Áreas Protegidas
→ Incluye varias categorías
→ NO es restrictiva (solo visual)

═══════════════════════════════════════
CAPA 9: PARQUES NACIONALES (PNN) ⚠️
═══════════════════════════════════════
'limite_pnn': {id:9, color:'#004d00'}
Color: Verde muy oscuro
→ Parques Nacionales Naturales
→ Máxima categoría de protección
⚠️ ES RESTRICTIVA: Elimina puntos dentro

═══════════════════════════════════════
CAPA 10: RESERVAS FORESTALES
═══════════════════════════════════════
'reservas_forestales': {id:10, color:'#2F4F4F'}
Color: Gris verdoso (DarkSlateGray)
→ Ley 2ª de 1959
→ Zonas de reserva forestal
→ NO es restrictiva (solo visual)

RESUMEN DE RESTRICCIONES:
⚠️ Solo 2 capas ELIMINAN puntos:
- Parques Arqueológicos (id:3)
- Parques Nacionales PNN (id:9)
Las otras 8 solo se muestran en el mapa.

VALORES DE DIMENSIONAMIENTO HIDRÁULICO	EXPLICACIÓN

`# Valores iniciales`
`CAUDAL_MIN_INICIAL = 0.15`
`CAUDAL_MAX_INICIAL = 0.50`
`PENDIENTE_MIN_INICIAL = 0.05`
 	
`TAMAÑO_PUNTOS = 2`
	CONFIGURACIÓN DE SLIDERS POR DEFECTO
	
	# Valores iniciales
	CAUDAL_MIN_INICIAL = 0.15
	→ Caudal mínimo en m³/s
	→ Puntos con caudal < 0.15 están ocultos
	→ Rango típico para microcentrales
	
	CAUDAL_MAX_INICIAL = 0.50
	→ Caudal máximo en m³/s
	→ Puntos con caudal > 0.50 están ocultos
	→ Limita a proyectos pequeños
	
	PENDIENTE_MIN_INICIAL = 0.05
	→ Pendiente mínima (en decimal)
	→ 0.05 = 5% de pendiente
	→ Necesaria para generar caída hidráulica
	
	TAMAÑO_PUNTOS = 2
	→ Radio de los círculos en píxeles
	→ Valor pequeño para mapas con muchos puntos
	→ Aumentar si hay pocos puntos
	
	ESTOS VALORES SE PUEDEN CAMBIAR:
	• Aquí: valores por defecto al abrir
	• En el mapa: usando los sliders del panel lateral izquierdo
	
	El usuario puede ajustar los filtros en tiempo real sin recargar el mapa.

ZONA DE APLICACIÓN POR TIPO DE TURBINA	EXPLICACIÓN

def determinar_tipo_turbina(caudal_m3s, caida_m, potencia_k):
"""
Determina el tipo de turbina aplicable según caudal y caída hidráulica
usando polígonos basados en el diagrama de selección de turbinas
y calcula la potencia y CAPEX para cada tipo
 	
Args:
caudal_m3s: Caudal en m³/s
caida_m: Caída hidráulica en metros
potencia_k: Potencia_k del punto (kW)
 	
Returns:
Lista de tipos de turbina aplicables con sus potencias y CAPEX
"""
from shapely.geometry import Point, Polygon
 	
# Conversión de unidades 	
caudal_cfs = caudal_m3s * 35.3147 # m³/s a pies³/s
caida_ft = caida_m * 3.28084 # metros a pies
 	
# Crear punto con las coordenadas del sitio 	
punto = Point(caudal_cfs, caida_ft)
 	
turbinas_aplicables = []
 	
# Definir polígonos de cada tipo de turbina basados en el diagrama 	
# Coordenadas: (caudal_cfs, caida_ft) 	
 	
# PAT - bombas como turbinas 	
pat = Polygon([
(1, 30), (1.2, 550), (7, 550), (15, 400), (13, 30)
])
 	
# Pelton - alta caída, bajo caudal 	
	CALCULO DE APLICACIÓN DE LAS TURBINAS POSIBLES EN CADA PUNTO

```
    pelton = Polygon([
        (1, 200), (1, 3000), (40, 3000), (70, 2000), (70, 1600), (30, 200), (10, 100)
    ])

    # Cross Flow - rango medio
    cross_flow = Polygon([
        (10, 10), (10, 800), (20, 800), (300, 40), (300, 10)
    ])

    # Francis - rango amplio central
    francis = Polygon([
        (15, 200), (35, 1150), (120, 1150), (800, 170), (500, 30), (80, 30)
    ])

    # Kaplan - bajo head, alto caudal
    kaplan = Polygon([
        (1, 10), (1, 80), (80, 200), (700, 200), (1800, 80), (1800, 25), (1300, 10)
    ])

    # Turgo - similar a Pelton pero menor caída
    turgo = Polygon([
        (1, 180), (1, 900), (35, 900), (350, 180)
    ])

    # Deriaz - grandes instalaciones
    deriaz = Polygon([
        (50, 100), (50, 260), (140, 260), (5000, 460), (17500, 100), (10500, 100)
    ])

    # Bulbo - muy bajo head, muy alto caudal
    bulbo = Polygon([
        (106, 100), (14000, 100), (25000, 33), (7000, 16), (106, 16)
    ])
```

POTENCIA APROVECHABLE	EXPLICACIÓN

`if pelton.contains(punto):`
 	
`potencia = potencia_k * 0.9 * EFICIENCIAS_TURBINAS['Pelton']`
 	
`capex = potencia * COSTOS_CAPEX['Pelton']`
 	
`turbinas_aplicables.append({'tipo': 'Pelton', 'potencia': potencia, 'capex': capex})`
 	
`if turgo.contains(punto):`
 	
`potencia = potencia_k * 0.9 * EFICIENCIAS_TURBINAS['Turgo']`
 	
`capex = potencia * COSTOS_CAPEX['Turgo']`
 	
`turbinas_aplicables.append({'tipo': 'Turgo', 'potencia': potencia, 'capex': capex})`
 	
`if francis.contains(punto):`
 	
`potencia = potencia_k * 0.9 * EFICIENCIAS_TURBINAS['Francis']`
 	
`capex = potencia * COSTOS_CAPEX['Francis']`
 	
`turbinas_aplicables.append({'tipo': 'Francis', 'potencia': potencia, 'capex': capex})`
 	
`if cross_flow.contains(punto):`
 	
`potencia = potencia_k * 0.9 * EFICIENCIAS_TURBINAS['Cross Flow']`
 	
`capex = potencia * COSTOS_CAPEX['Cross Flow']`
 	
`turbinas_aplicables.append({'tipo': 'Cross Flow', 'potencia': potencia, 'capex': capex})`
 	
`if kaplan.contains(punto):`
 	
`potencia = potencia_k * 0.9 * EFICIENCIAS_TURBINAS['Kaplan']`
 	
`capex = potencia * COSTOS_CAPEX['Kaplan']`
 	
`turbinas_aplicables.append({'tipo': 'Kaplan', 'potencia': potencia, 'capex': capex})`
 	
`if pat.contains(punto):`
 	
`potencia = potencia_k * 0.9 * EFICIENCIAS_TURBINAS['PAT']`
 	
`capex = potencia * COSTOS_CAPEX['PAT']`
 	
`turbinas_aplicables.append({'tipo': 'PAT', 'potencia': potencia, 'capex': capex})`
 	
`if deriaz.contains(punto):`
 	
`potencia = potencia_k * 0.9 * EFICIENCIAS_TURBINAS['Deriaz']`
	CÁLCULO DE LA POTENCIA APROVECHABLE SEGÚN EL TIPO DE TURBINA Y SU EFICIENCIA

```
        capex = potencia * COSTOS_CAPEX['Deriaz']

        turbinas_aplicables.append({'tipo': 'Deriaz',
'potencia': potencia, 'capex': capex})



    if bulbo.contains(punto):

        potencia = potencia_k * 0.9 *
EFICIENCIAS_TURBINAS['Bulbo']

        capex = potencia * COSTOS_CAPEX['Bulbo']

        turbinas_aplicables.append({'tipo': 'Bulbo',
'potencia': potencia, 'capex': capex})
```

EFICIENCIAS POR TIPO DE URBINAS	EXPLICACIÓN

`# Eficiencias de cada tipo de turbina`
`EFICIENCIAS_TURBINAS = {`
`'Francis': 0.92,`
`'PAT': 0.86,`
`'Pelton': 0.90,`
`'Kaplan': 0.89,`
`'Low Head': 0.89,`
`'Turgo': 0.87,`
`'Cross Flow': 0.70,`
`'Deriaz': 0.90,`
`'Bulbo': 0.90`
`}`
	RENDIMIENTO DE CADA TIPO DE TURBINA

DIMENSIONAMIENTO SEGÑUN DEMANDA (VSS)	EXPLICACIÓN

`if zona_clima in ['TEMPLADO', 'FRÍO', 'CÁLIDO SECO']:`
`potencia_pico = 1.54`
`elif zona_clima == 'CÁLIDO HÚMEDO':`
`potencia_pico = 2.06`
`else:`
`potencia_pico = 1.54 # Por defecto`
 	
`# Calcular potencia a abastecer para cubrir las VSS`
`potencia_abastecer_vss = vss * potencia_pico`
 	
`turbinas_data, caudal_cfs, caida_ft = determinar_tipo_turbina(caudal, caida, potencia_k)`
	CÁLCULO DE LA ZONA CLIMATICA Y LA POTENCIA REQUERIDA

COSTES TURBINA Y OTROS EQUIPOS	EXPLICACIÓN

`# 1. coste_turbina = (`
`potencia_kw *`
`COSTOS_CAPEX[tipo_turbina]`
`)`
 	
`# 2. coste_equipos = coste_turbina * 2.1`
	Fórmula: Potencia × Costo unitario
	Ejemplo con PAT de 50 kW:
	50 kW × $150/kW = $7,500
	
	EQUIPOS ELECTROMECÁNICOS
	Fórmula: Costo turbina × 2.1
	Los equipos representan 2.1 veces el costo de la turbina.
	
	Ejemplo con PAT de 50 kW:
	$7,500 × 2.1 = $15,750

COSTOS CAPEX DE TURBINAS	EXPLICACIÓN

COSTOS_CAPEX = {
'Francis': 950,
'PAT': 150,
'Pelton': 400,
'Kaplan': 600,
'Deriaz': 600,
'Turgo': 400,
'Cross Flow': 250,
'Deriaz': 550,
'Bulbo': 400
}
	COSTE BASE POR KILOVATIO INSTALADO

COSTES INSTALACIÓN Y PUESTA EN MARCHA	EXPLICACIÓN
# 3. complejidad = COMPLEJIDAD_INSTALACION[tipo_turbina] coste_instalacion = (coste_equipos + coste_turbina) * complejidad costes['coste_instalacion'] = coste_instalacion	COSTE DE INSTALACIÓN Y PUESTA EN MARCHA

COMPLEJIDAD DE INSTALACIÓN	EXPLICACIÓN

COMPLEJIDAD_INSTALACION = {
'PAT': 0.10,
'Cross Flow': 0.15,
'Francis': 0.20,
'Pelton': 0.20,
'Kaplan': 0.20,
'Turgo': 0.20,
'Deriaz': 0.20,
'Bulbo': 0.20,
'Deriaz': 0.20
}
	PORCENTAJE DEL FACTOR MULTIPLICADOR DE COMPLEJDAD PARA CALCULAR EL COSTE DE INSTALACIÓN Y PUESTA EN MARCHA

COSTES OBRA CIVIL	EXPLICACIÓN
`# 4. coste_obra_civil = 30350`	OBRA CIVIL Valor fijo: $30,350 USD

COSTES LÍNEA DE CONEXIÓN ELÉCTRICA	EXPLICACIÓN
`# 5. coste_linea = 10124`	LÍNEA DE CONEXIÓN ELÉCTRICA Valor fijo: $10,124 USD.

COSTES AMBIENTALES	EXPLICACIÓN

`# 6.  Costes ambientales (2% de turbina + equipos + transporte + línea + obra civil)`
 	
`    coste_ambiental = (coste_turbina + coste_equipos + coste_transporte + coste_linea + coste_obra_civil) * 0.02`
 	
`    costes['coste_ambiental'] = coste_ambiental`
	COSTES AMBIENTALES CON 2%

COSTES TRANSPORTE	EXPLICACIÓN

`# 7.     C_transporte = (C_fijo + α×P + β×D_real) × M_región × M_turbina`
 	
`    dist_capital_km = dist_capital_m / 1000`
 	
`    D_real = dist_capital_km * 1.45  # Factor de sinuosidad`
 	
`    # Parámetros de transporte`
 	
`    C_fijo = 8000  # Costo base de movilización (USD)`
 	
`    alfa = 60     # Costo por potencia (USD/kW)`
 	
`    beta = 6      # Costo por distancia (USD/km)`
 	
`    M_turb = MULTIPLICADOR_TRANSPORTE[tipo_turbina]`
 	
`    M_region = MULTIPLICADOR_REGION.get(region, 1.2)`
 	
`    coste_transporte = (C_fijo + alfa * potencia_kw + beta * D_real) * M_region * M_turb`
 	
`    costes['coste_transporte'] = coste_transporte`
	Fórmula completa:
	
	(C_fijo + α×P + β×D_real) × M_turb × M_region
	
	Componentes:
	
	- C_fijo = $8,000 (costo base logístico)
	
	- α = $60/kW (componente por potencia)
	
	- β = $6/km (componente por distancia)
	
	- Factor sinuosidad = 1.45 (carreteras colombianas)
	
	Distancia real = Distancia en línea recta × 1.45
	
	Ejemplo PAT 50 kW en Antioquia, 100 km:
	
	- Distancia real = 100 × 1.45 = 145 km
	
	- Base = $8,000 + (60×50) + (6×145)
	
	- Base = $8,000 + $3,000 + $870 = $11,870
	
	- M_turb PAT = 2.0
	
	- M_region Antioquia = 1.4
	
	- Total = $11,870 × 2.0 × 1.4 = $33.236

MULTIPLICADOR DE TRANSPORTE	EXPLICACIÓN

MULTIPLICADOR_TRANSPORTE = {
'PAT': 2.0,
'Pelton': 2.0,
'Cross Flow': 2.5,
'Bulbo': 2.5,
'Francis': 3.0,
'Kaplan': 3.0,
'Turgo': 3.0,
'Deriaz': 3.0,
'Deriaz': 3.0
}
	FACTOR DE DIFICULTAD LOGÍSTICA EL CUAL ES NECESARIO PARA EL CALCULO DEL COSTE DE TRANSPORTE

MULTIPLICADOR POR REGIÓN	EXPLICACIÓN

# Tabla 4: Multiplicador por región 	
MULTIPLICADOR_REGION = {
'Región Pacífico': 1.6,
'Región Eje Cafetero – Antioquia': 1.4,
'Región Centro Sur': 1.3,
'Región Centro Oriente': 1.2,
'Región Caribe': 1.0,
'Región Llano': 1.0,
'': 1.2 # Sin dato
}
	FACTOR SEGÚN UBICACIÓN EN COLOMBIA. NECESARIO PARA EL CALCULO DEL COSTE DE TRANSPORTE

DICCIONARIO DE CAPITALES - PARTE 2	EXPLICACIÓN
'HUILA': {'capital': 'Neiva', 'lat': 2.9273, 'lon': -75.2819}, 'LA GUAJIRA': {'capital': 'Riohacha', 'lat': 11.5444, 'lon': -72.9072}, 'MAGDALENA': {'capital': 'Santa Marta', 'lat': 11.2408, 'lon': -74.1990}, 'META': {'capital': 'Villavicencio', 'lat': 4.1420, 'lon': -73.6266}, 'NARIÑO': {'capital': 'Pasto', 'lat': 1.2136, 'lon': -77.2811}, 'NORTE DE SANTANDER': {'capital': 'Cúcuta', 'lat': 7.8939, 'lon': -72.5078}, 'PUTUMAYO': {'capital': 'Mocoa', 'lat': 1.1514, 'lon': -76.6428}, 'QUINDIO': {'capital': 'Armenia', 'lat': 4.5339, 'lon': -75.6811}, 'RISARALDA': {'capital': 'Pereira', 'lat': 4.8133, 'lon': -75.6961}, 'ARCHIPIELAGO SAN ANDRES Y PROVIDENCIA': {'capital': 'San Andrés', 'lat': 12.5847, 'lon': -81.7006}, 'SANTANDER': {'capital': 'Bucaramanga', 'lat': 7.1195, 'lon': -73.1227}, 'SUCRE': {'capital': 'Sincelejo', 'lat': 9.3047, 'lon': -75.3978}, 'TOLIMA': {'capital': 'Ibagué', 'lat': 4.4389, 'lon': -75.2322}, 'VALLE DEL CAUCA': {'capital': 'Cali', 'lat': 3.4516, 'lon': -76.5320}, 'VAUPES': {'capital': 'Mitú', 'lat': 1.2533, 'lon': -70.1733}, 'VICHADA': {'capital': 'Puerto Carreño', 'lat': 6.1850, 'lon': -67.4860} }	COORDENADAS DE CAPITALES (H-V) CONTINUACIÓN DEL DICCIONARIO: HUILA → Neiva (2.93, -75.28) Valle del Magdalena, petróleo LA GUAJIRA → Riohacha (11.54, -72.91) Península, cultura Wayúu MAGDALENA → Santa Marta (11.24, -74.20) Sierra Nevada, turismo de playa META → Villavicencio (4.14, -73.63) Puerta al Llano NARIÑO → Pasto (1.21, -77.28) Frontera con Ecuador NORTE DE SANTANDER → Cúcuta (7.89, -72.51) Frontera con Venezuela ... hasta ... VICHADA → Puerto Carreño (6.19, -67.49) Confluencia ríos Meta y Orinoco USO EN EL CÓDIGO: Para cada punto del shapefile: 1. Se calcula distancia a TODAS las capitales 2. Se guarda la MÁS CERCANA 3. Esta distancia afecta coste de transporte Un punto en límite de Cundinamarca puede tener Tunja (Boyacá) más cerca que Bogotá.

COSTES OTROS	EXPLICACIÓN
# 8. suma_parcial = (coste_turbina + coste_equipos + coste_obra_civil + coste_instalacion + coste_linea + coste_ambiental + coste_transporte) otros_costes = suma_parcial * 0.05 costes['otros_costes'] = otros_costes	COSTE OTROS CON 5%

COSTES CAPEX TOTAL	EXPLICACIÓN

`# CAPEX Total`
`capex_total = suma_parcial + otros_costes`
`costes['capex_total'] = capex_total`
	COSTE DEL CAPEX TOTAL

COSTES OPEX	EXPLICACIÓN

`# OPEX`
`opex = capex_total * 0.03`
`costes['opex'] = opex`
	COSTE DEL OPEX CON 3 %

FUNCIÓN COSTES	EXPLICACIÓN

`def calcular_costes_detallados(`
`potencia_kw,`
`tipo_turbina,`
`departamento,`
`distancia_km`
`):`
	Definición de la función principal de cálculo económico.
	Parámetros de entrada:
	potencia_kw: Potencia nominal de la instalación en kilovatios (kW).
	tipo_turbina: Tipo de turbina seleccionada ("Francis", "PAT", "Pelton", "Kaplan", "Cross Flow", "Bulbo").
	departamento: Departamento de Colombia donde se ubica el proyecto.
	distancia_km: Distancia en kilómetros desde la capital departamental más cercana.
	La función retorna un diccionario con el desglose de costes por partida.

3. Resto del código

El código restante del archivo Hydro.py comprende la generación del mapa interactivo HTML con toda su funcionalidad. A continuación, se describe cada sección.

Nota: los números de línea indicados son aproximados y pueden variar ligeramente según modificaciones menores del código.

- ~Líneas 543-1050: PANEL HTML Código HTML que genera el panel lateral izquierdo de la interfaz. Incluye los controles de filtrado (sliders para caudal, pendiente, distancia), checkboxes para activar/desactivar capas geográficas (parques nacionales, resguardos indígenas, etc.), selectores desplegables para tipo de turbina y departamento, y botones de acción (actualizar, exportar, generar informe).
- ~Líneas 1050-1270: PANEL DE PARAMETRIZACIÓN Sección del HTML con campos de entrada (inputs numéricos) que permiten al usuario modificar en tiempo real los parámetros técnicos: eficiencias de cada tipo de turbina, costos CAPEX por kW, y multiplicadores de complejidad, transporte y región. Permite personalizar los cálculos sin modificar el código fuente.
- ~Líneas 1270-1400: SECCIÓN DE FÓRMULAS Inputs editables para los coeficientes de las fórmulas de costes: valor fijo de obra civil, costo base de línea eléctrica, coeficientes α y β de transporte, porcentajes de ambiental y otros gastos, y factor de OPEX. El usuario puede ajustar estos valores y recalcular instantáneamente.
- ~Líneas 1400-1500: LEYENDA Elemento HTML que muestra la explicación visual del mapa: significado de los colores de los marcadores (según potencia o priorización), iconos utilizados (viviendas sin servicio, distancia), y escala de valores. Ayuda al usuario a interpretar correctamente la información mostrada.
- ~Líneas 1500-1600: ESTILOS CSS Hojas de estilo que definen la apariencia visual de toda la interfaz: colores corporativos, fuentes, tamaños, márgenes del panel lateral, estilos de botones, aspecto de los sliders, formato de las tablas en popups, y diseño responsive para diferentes tamaños de pantalla.
- ~Líneas 1600-1800: VARIABLES JAVASCRIPT Declaración e inicialización de variables globales: array con todos los puntos hidroeléctricos (coordenadas, atributos), objetos con parámetros editables (eficiencias, costos), estado actual de los filtros, referencias a elementos del DOM, y configuración inicial del mapa (centro, zoom, capa base).
- ~Líneas 1800-2100: FUNCIÓN actualizar() Función principal que se ejecuta cada vez que el usuario modifica un filtro. Lee los valores actuales de todos los controles, filtra el array de puntos según los criterios activos, elimina los marcadores anteriores del mapa, y dibuja solo los puntos que cumplen las condiciones. Actualiza también el contador de puntos visibles.
- ~Líneas 2100-2250: FUNCIONES DE POPUP Funciones que generan el contenido HTML de los popups informativos de cada punto. Construyen tablas con datos del punto (coordenadas, caudal, pendiente, VSS), calculan los costes detallados llamando a la función de cálculo, y formatean la información con estilos para mostrarla al hacer clic en un marcador.
- ~Líneas 2250-2400: FUNCIÓN priorizar() Activa el modo de priorización que ordena los puntos por ratio CAPEX/VSS (costo por vivienda beneficiada). Cambia el color de los marcadores según su posición en el ranking, resalta los más eficientes, y permite identificar rápidamente los proyectos con mejor relación costo-beneficio social.
- ~Líneas 2400-2500: FUNCIÓN mostrarAnalisisMultiescenario() Genera una gráfica interactiva que muestra la curva de VSS acumuladas vs CAPEX acumulado. Permite

visualizar cuántas viviendas se pueden electrificar con diferentes niveles de inversión, identificando el punto óptimo de inversión y los rendimientos decrecientes.

- ~Líneas 2500-2700: FUNCIÓN descargarDatos() Exporta los puntos actualmente visibles (filtrados) a un archivo CSV. Recorre los marcadores del mapa, extrae sus atributos y costes calculados, formatea los datos en columnas separadas por punto y coma, y genera la descarga del archivo en el navegador del usuario.
- ~Líneas 2700-2900: FUNCIÓN descargarInforme() Genera un informe en formato Word (.docx) con el resumen del análisis. Utiliza la librería docx.js embebida para crear un documento con tablas de puntos seleccionados, gráficos de distribución, resumen de parámetros utilizados, y conclusiones. Permite documentar el análisis realizado.
- ~Líneas 2900-3100: FUNCIÓN calcularCostesDetallados() JS Réplica en JavaScript de la función Python del mismo nombre. Recibe potencia, tipo de turbina, departamento y distancia, y calcula las 8 partidas de costes más OPEX. Usa los parámetros editables del panel, permitiendo que los cambios del usuario se reflejen inmediatamente en los cálculos.
- ~Líneas 3100-3300: FUNCIÓN mostrarPuntoEnMapa() Crea y añade un marcador circular al mapa para cada punto hidroeléctrico. Define el color según potencia o priorización, configura el popup con la información detallada, asigna eventos de clic, y gestiona la pertenencia a grupos de capas para poder filtrar por categorías.
- ~Líneas 3300-3420: FUNCIONES AUXILIARES Conjunto de funciones de utilidad: resetearParametros() restaura valores por defecto, formatearNumero() aplica separadores de miles y decimales, validarInput() comprueba rangos válidos, calcularDistanciaHaversine() obtiene distancia entre coordenadas, y otras funciones de soporte.
- ~Líneas 3420-3492: PROGRAMA PRINCIPAL Código de ejecución principal en Python: carga los datos geográficos (shapefiles de puntos y capas), aplica los filtros espaciales iniciales, calcula distancias a capitales, genera el objeto mapa de Folium, inyecta todo el HTML/CSS/JavaScript, y guarda el archivo HTML final.

Tech for
impact

indracompany.com

MINSAIT