# Modelo continuo de estabilidad de LADERAS en cárcavas

**INFORME nº 02**

**Inscrito en el Registro Propiedad Intelectual**

**Proyecto CARCAVA. Influencia Climática y Agronómica en la formación y evolución de la Red de CArcaVas en la campiña Andaluza.**

*Proyectos de investigación de excelencia, en régimen de concurrencia competitiva, destinadas a entidades calificadas como agentes del Sistema Andaluz del Conocimiento, en el ámbito del Plan Andaluz de Investigación, Desarrollo e Innovación (PAIDI 2020). BOJA 239 - 15 de diciembre de 2021. Consejería de Universidad, Investigación e Innovación de la Junta de Andalucía.*

Contenido





ETSIAM Escuela Técnica Superior de Ingeniería Agronómica y de Montes
UNIVERSIDAD Ð CÓRDOBA
UNIVERSIDAD Ð CÓRDOBA
Junta de Andalucía
Consejería de Universidad, Investigación e Innovación

# Modelo continuo de estabilidad de LADERAS en cárcavas

## 1. Objeto

Definir las especificaciones técnicas para el desarrollo e implementación de un modelo continuo de estabilidad de taludes en cárcavas, aplicado a fincas de agrícolas, integrando datos meteorológicos, series de humedad del suelo medidas con sensores, parámetros edáficos e hidrogeotécnicos (medidos y estimados), cálculo dinámico del Factor de Seguridad (FS) y análisis probabilístico mediante Monte Carlo, incluyendo salidas tipo curvas Intensidad–Duración (I–D).

## 2. Alcance

El sistema se diseña para:

- Operar con series temporales continuas (mínimo resolución horaria).
- Ejecutarse por finca y por cárcava piloto, permitiendo comparación entre fincas.
- Producir resultados en formatos reproducibles (CSV/PNG) para análisis estadístico y documentación.

Quedan fuera del alcance inicial (posibles mejoras futuras):

- Modelización 3D geotécnica avanzada (p. ej., elementos finitos).
- Resolución numérica completa de la ecuación de Richards.
- Implementación completa del modelo físico transitorio de Iverson (se contempla como ampliación).
- Escalado espacial GIS a nivel de cuenca/región (el diseño lo admite, pero no es necesario en la fase básica).

## 3. Arquitectura funcional y flujo de cálculo

El flujo de cálculo es: M1. Ingesta y preprocesado → M2. Balance hídrico → M3. Estabilidad (FS) → M4. Monte Carlo → M5. Curvas I–D y exportación.

### 3.1 M1. Ingesta y preprocesado

- Lectura de ficheros meteorológicos y humedad del suelo.
- Normalización de unidades y control básico de calidad (valores negativos, rangos plausibles, NA).
- Alineación temporal (resample) a una malla común (por defecto horaria).

### 3.2 M2. Balance hídrico por capas (modelo conceptual)

- Suelo discretizado en 3 capas (superficial, media, profunda), asociadas a las profundidades de sensores.
- Infiltración efectiva aproximada por I(t)=min(P(t), Ksat).

- Redistribución vertical por percolación cuando una capa supera saturación (exceso sobre θs).
- Extracción por evapotranspiración/evaporación potencial simplificada (ET0 constante o estimada).

### 3.3 M3. Estabilidad: talud infinito (Factor de Seguridad)

- Cálculo de FS(t) con el modelo de talud infinito.
- Aproximación humedad–presión de poros mediante una altura saturada equivalente hw(t) derivada de la humedad profunda.
- Criterio de fallo: FS(t) < 1.

### 3.4 M4. Incertidumbre: Monte Carlo

- Asignación de distribuciones (pdf) a parámetros no medidos: Ksat, c', ϕ', γ y H.
- Simulación repetida para obtener P(fallo)=P(min(FS)<1) en un periodo o para escenarios I–D.

### 3.5 M5. Curvas I–D, eventos y exportación

- Extracción automática de eventos de lluvia a partir de series horarias (separación por periodo seco mínimo).
- Etiquetado de eventos por FS mínimo durante el evento (fallo_modelo).
- Cálculo de grid I–D (intensidad, duración) y probabilidad de fallo mediante Monte Carlo.
- Exportación automática a CSV y generación de figuras PNG.

## 4. Requisitos de datos de entrada

Estructura recomendada por finca: data/<Finca_ID>/meteo.csv, data/<Finca_ID>/humedad.csv y un fichero de parámetros/configuración.

### 4.1 Meteorología (forzamiento)

| Variable | Unidad | Resolución mínima | Descripción/uso |
|---|---|---|---|
| fecha | datetime | horaria | Marca temporal (ISO 8601). |
| P_mm_h | mm/h | horaria | Intensidad media horaria (o precipitación horaria equivalente). |
| ta_c | °C | horaria | Temperatura del aire (para ET0; recomendable). |
| rh_pct | % | horaria | Humedad relativa (para ET0; recomendable). |

Formato CSV recomendado (cabecera obligatoria):



ETSIAM Escuela Técnica Superior de Ingeniería Agronómica y de Montes UNIVERSIDAD D CÓRDOBA
UNIVERSIDAD D CÓRDOBA
Junta de Andalucía Consejería de Universidad, Investigación e Innovación

fecha,P_mm_h,ta_c,rh_pct
2025-01-01 00:00,0.0,8.2,81

2025-01-01 01:00,2.3,8.0,83

### 4.2 Humedad del suelo (sensores)

| Variable | Unidad | Resolución mínima | Descripción/uso |
|---|---|---|---|
| fecha | datetime | ≤ horaria | Marca temporal (se resamplea a horaria si es minutal). |
| superficial | $m^3/m^3$ | ≤ horaria | Humedad volumétrica capa superficial. |
| media | $m^3/m^3$ | ≤ horaria | Humedad volumétrica capa intermedia. |
| profunda | $m^3/m^3$ | ≤ horaria | Humedad volumétrica capa profunda (clave para hw). |

Formato CSV recomendado:

fecha,superficial,media,profunda
2025-01-01 00:00,0.21,0.24,0.27

### 4.3 Suelo e hidrogeotecnia

En la fase básica se usa un suelo base (EXPS08-P1) y distribuciones para parámetros no medidos. Formato recomendado: JSON.

| Parámetro | Unidad | Tipo | Origen | Uso en el modelo |
|---|---|---|---|---|
| Textura (arena/limo/arcilla) | % | Fijo | Muestra EXPS08-P1 | Parametrización edáfica y PTF si se aplica. |
| ρb (densidad aparente) | $g/cm^3$ | Fijo | EXPS08-P1 | Cálculo de porosidad θs y peso unitario aproximado. |
| θFC, θWP | $m^3/m^3$ | Fijo | EXPS08-P1 | Almacenamiento y estados iniciales plausibles. |
| Ksat | mm/h | Aleatorio (LogNormal) | Estimación sin ensayo | Infiltración máxima I(t)=min(P,Ksat). |
| c', ϕ' | kPa / ° | Aleatorio (Normal trunc.) | Estimación sin ensayo | Resistencia al corte (FS). |
| H, γ | m / $kN/m^3$ | Aleatorio | Estimación sin ensayo | Geometría efectiva de rotura y tensiones. |

### 4.4 Geometría/topografía del talud

Requisitos mínimos por finca/cárcava (tramo instrumentado):

- Pendiente del talud θ (grados).
- Profundidad efectiva del plano de rotura H (si no se mide, se modela como distribución triangular).
- Opcional: perfil 2D del talud (x,z) para documentación y mejoras futuras.

## 5. Especificación del modelo (formulación)

### 5.1 Infiltración y balance por capas

- Infiltración efectiva (primer orden): I(t)=min(P(t), Ksat).
- Actualización conceptual por capa i: θ_i(t+Δt)=θ_i(t) + (In_i(t) − ET(t) − Perc_i(t)) / Z_i.

La percolación se activa cuando θ_i excede la saturación θs (exceso sobre θs) y se transfiere a la capa inferior.

### 5.2 Relación humedad–presión de poros (aproximación)

Altura saturada equivalente en el plano de rotura: hw(t)=H·(θ_profunda(t)/θs).

### 5.3 Factor de Seguridad (talud infinito)

$FS(t)= [c' + (\gamma_s \cdot H \cdot \cos^2\theta - \gamma_w \cdot hw(t)) \cdot \tan\phi'] / [\gamma_s \cdot H \cdot \sin\theta \cdot \cos\theta]$. Criterio de fallo: FS(t)<1.

### 5.4 Monte Carlo y probabilidad de fallo

Se muestrean parámetros y se calcula min(FS) del periodo. Se estima P_fallo = P(min(FS)<1).

## 6. Salidas, tablas y gráficas

### 6.1 Salidas por finca (exportación automática)

Carpeta outputs/<Finca_ID>/

| Figura | Formato | Contenido |
|---|---|---|
| humedad_sim | CSV | Humedad simulada (3 capas) a resolución horaria. |
| FS | CSV | Serie temporal de Factor de Seguridad. |
| eventos_lluvia_FS | CSV | Eventos de lluvia y FS mínimo durante cada evento. |
| curvas_ID_prob | CSV | Grid I–D con probabilidad de fallo (Monte Carlo). |
| serie_tiempo | PNG | Barras lluvia + línea FS (y figura de humedad). |
| hist_FSmin / curvas_ID_prob | PNG | Histograma de min(FS) y nube I–D coloreada por P(fallo). |

### 6.2 Gráficas recomendadas (mínimo)

- Serie temporal conjunta: precipitación (barras) + FS(t) (línea) + línea umbral FS=1.
- Humedad simulada (superficial/media/profunda) y, si existe, observada (trazo).
- Histograma de mínimos de FS por simulación (Monte Carlo).
- Curvas I–D probabilísticas: nube (D,I) con color = P(fallo).
- Comparación entre fincas: resumen por duración (P media por duración) y/o superficies de probabilidad.

## 7. Validación y análisis estadístico

### 7.1 Validación hidrológica (si hay observaciones de humedad)

- RMSE por capa.
- NSE (Nash–Sutcliffe) por capa.
- Sesgo medio (mean bias) por capa.

### 7.2 Validación de inestabilidad (si existe inventario/observaciones)

- Etiquetado de eventos lluvia con deslizamiento (0/1).
- Curva ROC y AUC para umbrales de probabilidad o FS (a nivel evento).
- Métricas: precisión, recall, F1 y matriz de confusión.

### 7.3 Sensibilidad e incertidumbre

- Análisis de sensibilidad (tornado o Morris/Sobol) sobre Ksat, H, c', ϕ' y γ.
- Bandas de confianza (percentiles 5–50–95) para FS(t) y para umbrales I–D.

## 8. Implementación software

Lenguaje: Python 3.10+

Dependencias mínimas: numpy, pandas, matplotlib

Opcionales: scipy (calibración), scikit-learn (ROC/AUC), pyarrow (Parquet)

Estructura recomendada de carpetas:

- data/<Finca_ID>/{meteo.csv, humedad.csv}
- outputs/<Finca_ID>/{*.csv, *.png}
- modelo_carcavas.py (script principal)



ETSIAM Escuela Técnica Superior de Ingeniería Agronómica y de Montes UNIVERSIDAD D CÓRDOBA
UNIVERSIDAD D CÓRDOBA
Junta de Andalucía Consejería de Universidad, Investigación e Innovación

**9. Supuestos y limitaciones**

- El balance hídrico por capas es conceptual y no resuelve la ecuación de Richards.
- La relación humedad–presión de poros se aproxima mediante hw(t)=H·θ/θs; puede refinarse con una curva θ–ψ (van Genuchten).
- Los parámetros resistentes (c', ϕ') y Ksat se consideran inciertos y se tratan probabilísticamente (Monte Carlo).
- Los resultados deben interpretarse como probabilidades y rangos mientras no existan ensayos geotécnicos.

**10. Referencias bibliográficas (base científica utilizada)**

1. Frattini, P., Crosta, G. B., & Sosio, R. (2009). Approaches for defining thresholds and return periods for rainfall-triggered shallow landslides. Hydrological Processes, 23(10), 1444–1460. https://doi.org/10.1002/hyp.7269
2. Gioia, A., Iacobellis, V., Manfreda, S., & Fiorentino, M. (2008). Runoff thresholds in derived flood frequency distributions. Hydrology and Earth System Sciences, 12, 1295–1307. https://doi.org/10.5194/hess-12-1295-2008
3. Gioia, A., Iacobellis, V., Manfreda, S., & Fiorentino, M. (2012). Influence of infiltration and soil storage capacity on the skewness of the annual maximum flood peaks in a theoretically derived distribution. Hydrology and Earth System Sciences, 16(3), 937–951. https://doi.org/10.5194/hess-16-937-2012
4. Laio, F., Porporato, A., Ridolfi, L., & Rodríguez-Iturbe, I. (2001). Plants in water-controlled ecosystems: active role in hydrologic processes and response to water stress. Advances in Water Resources, 24(7), 707–723. https://doi.org/10.1016/S0309-1708(01)00004-5

**11. Aplicación del modelo de estabilidad en cárcava instrumentada**

**11.1 Objeto de la prueba**

Se ha realizado una primera aplicación del modelo de estabilidad de ladera tipo talud infinito, utilizando:

- Series reales de precipitación horaria (estación en finca).
- Series reales de humedad volumétrica del suelo (4 sensores a 4 profundidades, ubicados a 15 cm del borde de la cárcava).
- Parámetros edáficos derivados de la muestra EXPS08-P1.
- Geometría estimada a partir de perfiles transversales del vídeo suministrado.

El análisis se ha realizado diferenciando ambos márgenes de la cárcava:

- **Margen derecho:** sensores 1 y 2.

- **Margen izquierdo:** sensores 3 y 4.

Se han evaluado dos escenarios:

- **Escenario A (conservador):** pendiente elevada, cohesión baja, saturación total del espesor.
- **Escenario B (más realista):** pendiente efectiva reducida, incremento de cohesión aparente (raíces) y saturación parcial del espesor.

El objetivo no ha sido predecir fallos reales, sino evaluar la coherencia físico-mecánica del sistema y la sensibilidad del modelo a los parámetros geométricos y geotécnicos.

### 11.2 Resultados principales

#### 11.2.1 Margen derecho (sensores 1–2)

- En el **Escenario A**, la probabilidad de fallo en el periodo analizado es muy elevada ($P_f \approx 0.98$).
- En el **Escenario B**, la probabilidad de fallo desciende significativamente ($P_f \approx 0.21$).

El Factor de Seguridad mínimo determinista desciende por debajo de 1 en periodos de alta humedad profunda ($\theta$80 cm elevada), especialmente tras episodios de lluvia acumulada.

#### 11.2.2 Margen izquierdo (sensores 3–4)

- En el **Escenario A**, se obtiene $P_f \approx 0.76$.
- En el **Escenario B**, la probabilidad de fallo es prácticamente nula ($\approx 0$).

El comportamiento es sensiblemente más estable que el margen derecho, coherente con la menor pendiente observada en los perfiles.

### 11.3 Interpretación técnica

#### 11.3.1 Control geométrico de la estabilidad

La diferencia entre márgenes confirma que la **pendiente del talud es el factor dominante** en la estabilidad:

- El margen derecho, con mayor inclinación, presenta sensibilidad elevada a pequeños incrementos de presión de poros.
- El margen izquierdo, con pendiente más tendida, mantiene $FS > 1$ en condiciones realistas.

Esto es coherente con la formulación del modelo de talud infinito, donde el término desestabilizador depende de $\sin\theta \cdot \cos\theta$, creciendo rápidamente con la pendiente.

#### 11.3.2 Papel de la humedad profunda (80 cm)

La evolución temporal del FS muestra que:

- El descenso del FS no responde únicamente a picos de lluvia instantáneos.
- El parámetro crítico es el **estado de humedad antecedente en profundidad**.

La humedad a 80 cm actúa como variable integradora de:

- Lluvia acumulada,
- Capacidad de almacenamiento,
- Tiempo de drenaje.

Esto valida la importancia de los sensores profundos como indicadores operativos de estabilidad.

### 11.3.3 Sensibilidad a parámetros geotécnicos

El paso del Escenario A al B demuestra que el modelo es altamente sensible a:

- Cohesión efectiva (c'),
- Espesor movilizable (H),
- Grado real de saturación del perfil,
- Pendiente efectiva del plano de rotura.

El Escenario A representa una **cota superior conservadora de riesgo** (hipótesis de saturación total y baja cohesión).
El Escenario B introduce mecanismos físicos plausibles en campo (raíces, drenaje parcial), obteniendo un comportamiento más realista.

## 11.4 Coherencia con la morfología observada

El análisis es coherente con la observación geométrica de los perfiles:

- La cárcava presenta comportamiento asimétrico.
- El margen más vertical concentra el potencial de inestabilidad.
- El margen más tendido actúa como zona estructuralmente más estable.

Este patrón es consistente con procesos típicos de cárcavas en suelos arcillosos con incisión progresiva lateral.

## 11.5 Limitaciones de la prueba

1. La pendiente se ha considerado constante por margen.
2. No se ha utilizado modelización transitoria de infiltración (ecuación de Richards).



ETSIAM Escuela Técnica Superior de Ingeniería Agronómica y de Montes UNIVERSIDAD D CÓRDOBA
UNIVERSIDAD D CÓRDOBA
Junta de Andalucía Consejería de Universidad, Investigación e Innovación

3. La presión de poros se ha aproximado mediante relación lineal con θ.
4. No se dispone de inventario independiente de deslizamientos para validación ROC.

Por tanto, los resultados deben interpretarse como **análisis de sensibilidad y consistencia física**, no como predicción determinista de fallo.

### 11.6 Conclusiones técnicas

1. El modelo reproduce adecuadamente el comportamiento diferencial entre márgenes.
2. El margen derecho es el sector crítico desde el punto de vista mecánico.
3. La humedad profunda (80 cm) es un excelente predictor del descenso del FS.
4. El riesgo estimado depende fuertemente de la geometría efectiva y la cohesión.
5. La introducción de saturación parcial y cohesión aparente reduce sustancialmente el sobreajuste conservador.
6. El sistema de instrumentación instalado es adecuado para construir umbrales operativos de estabilidad.

### 11.7 Recomendaciones para la siguiente fase

1. Extraer pendiente real por sección cada 5 m a partir de los perfiles originales (CSV).
2. Calcular probabilidad de fallo por evento de lluvia (no solo por periodo completo).
3. Derivar umbrales tipo:
    - θ80cm crítica
    - Lluvia acumulada 24–72h crítica
4. Introducir modelización de infiltración simplificada acoplada (siguiente nivel del modelo).

## 12. Implicaciones para gestión y monitorización en campo

### 12.1 Enfoque general

Los resultados obtenidos permiten trasladar el modelo desde un marco analítico a un **sistema operativo de apoyo a la gestión del riesgo en cárcavas agrícolas**.

El análisis demuestra que:

- La estabilidad es **asimétrica entre márgenes**.
- La variable más informativa no es la lluvia instantánea, sino la **humedad profunda del suelo (80 cm)**.



ETSIAM Escuela Técnica Superior de Ingeniería Agronómica y de Montes UNIVERSIDAD Ð CÓRDOBA
UNIVERSIDAD Ð CÓRDOBA
Junta de Andalucía Consejería de Universidad, Investigación e Innovación

- El comportamiento es altamente sensible a pendiente efectiva y cohesión aparente.

Esto permite diseñar un sistema de monitorización **selectivo, eficiente y orientado a decisiones**.

### 12.2 Identificación de zonas prioritarias

#### 12.2.1 Priorización por margen

El modelo identifica el **margen derecho** como sector crítico.

**Implicación práctica:**

- La vigilancia, inspección visual y actuaciones preventivas deben concentrarse en este margen.
- El margen izquierdo puede considerarse zona de menor prioridad, salvo eventos extremos.

Esto optimiza recursos de mantenimiento.

#### 12.3.1 Variables críticas a monitorizar

**Humedad profunda (80 cm)**

La humedad a 80 cm ha mostrado:

- Relación directa con descenso del FS.
- Capacidad de integrar estado antecedente de lluvia.
- Mayor valor predictivo que la precipitación puntual.

**Recomendación operativa:**

- Establecer umbral de alerta basado en θ80cm.
- Mantener calibración y verificación periódica de sensores profundos.
- Priorizar mantenimiento de estos sensores frente a superficiales.

#### 12.3.2 Lluvia acumulada (24–72 h)

Aunque la lluvia horaria no correlaciona instantáneamente con el fallo, sí actúa como disparador cuando el perfil ya está húmedo.

**Sistema recomendado de doble umbral:**

Alerta si se cumple:



ETSIAM Escuela Técnica Superior de Ingeniería Agronómica y de Montes UNIVERSIDAD D CÓRDOBA
UNIVERSIDAD D CÓRDOBA
Junta de Andalucía Consejería de Universidad, Investigación e Innovación

- θ80cm > umbral crítico
  Y
- Lluvia acumulada 48–72 h > umbral secundario

Este esquema es más robusto que usar lluvia sola.

**12.4 Sistema de niveles de alerta**

Se propone una clasificación operativa sencilla:

| Nivel | Condición hidromecánica | Acción recomendada |
|---|---|---|
| Verde | FS estimado > 1.2 | Sin actuación |
| Amarillo | 1.0 < FS ≤ 1.2 | Inspección visual |
| Naranja | FS cercano a 1 (p05 < 1) | Revisión técnica |
| Rojo | FS < 1 persistente | Actuación preventiva |

El modelo puede actualizar FS automáticamente cada hora.

**12.5 Actuaciones preventivas derivadas del modelo**

**12.5.1 Gestión de vegetación**

El escenario B muestra que el incremento de cohesión aparente reduce significativamente el riesgo.

**Implicación:**

- Mantener cobertura vegetal en margen crítico.
- Evitar desbroces intensivos en periodos húmedos.
- Considerar refuerzo con especies de raíz profunda.

**12.5.2 Control de infiltración en coronación**

Dado que la saturación profunda controla el FS:

- Evitar concentración de escorrentía en el borde.
- Revisar drenajes.
- Minimizar compactaciones que favorezcan escorrentía hacia el talud.

**12.5.3 Intervención estructural (solo si necesario)**

Si el modelo confirma episodios recurrentes con FS<1:

- Reducción de pendiente efectiva.

- Bermas intermedias.
- Drenaje subsuperficial.

El modelo permite evaluar virtualmente el efecto de estas medidas antes de ejecutarlas.

**12.6 Integración en sistema digital de explotación**

El modelo puede integrarse en:

- Datalogger local con procesamiento básico.
- Servidor remoto con actualización automática.
- Plataforma digital agrícola (AgTech).

Salida recomendada:

- Dashboard con:
    - θ80cm actual
    - Lluvia acumulada 72h
    - FS estimado
    - Nivel de alerta

Esto transforma sensores pasivos en sistema activo de gestión.

**12.7 Ventajas estratégicas del sistema**

1. Bajo coste adicional (ya existe instrumentación).
2. Escalable a otras cárcavas.
3. Transferible a otras fincas con parámetros calibrados.
4. Permite anticipación en lugar de reacción.
5. Alineado con digitalización agrícola y gestión sostenible del suelo.

**12.8 Recomendaciones para fase siguiente**

Para consolidar el sistema como herramienta operativa:

1. Extraer pendiente efectiva real por sección.
2. Validar con inventario de microdeslizamientos (si existen).
3. Calcular umbrales estadísticos I–D por margen.
4. Incorporar modelo continuo de infiltración (siguiente nivel).

**12.9 Conclusión operativa**



ETSIAM Escuela Técnica Superior de Ingeniería Agronómica y de Montes UNIVERSIDAD D CÓRDOBA
UNIVERSIDAD D CÓRDOBA
Junta de Andalucía Consejería de Universidad, Investigación e Innovación

La instrumentación instalada es adecuada para desarrollar un **sistema de alerta temprana de inestabilidad en cárcavas agrícolas**.

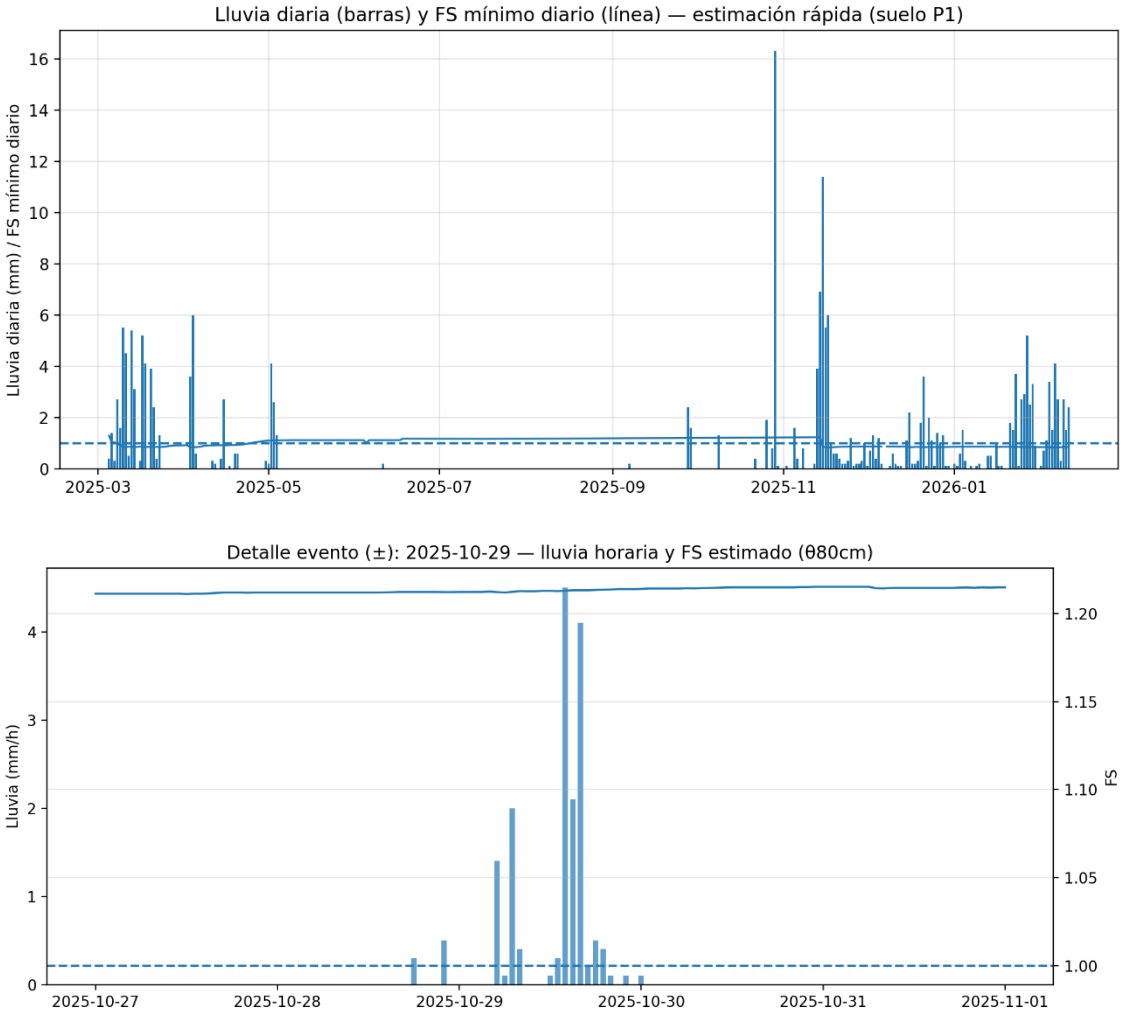
El margen derecho constituye el sector prioritario de intervención.

La humedad profunda del suelo debe considerarse la variable central de monitorización.

El modelo permite pasar de una aproximación reactiva (reparar tras fallo) a una estrategia preventiva basada en estado hidromecánico.

## 13. Figuras

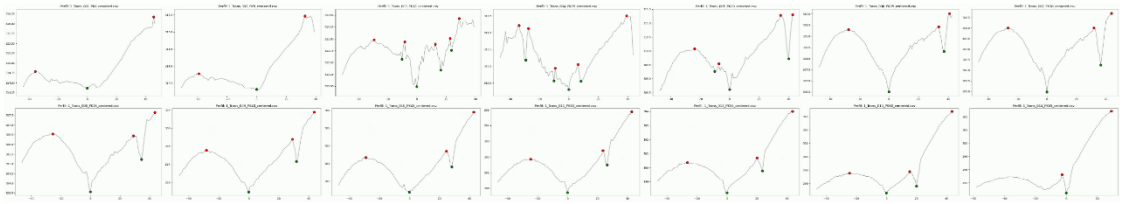
### 13.1 Datos

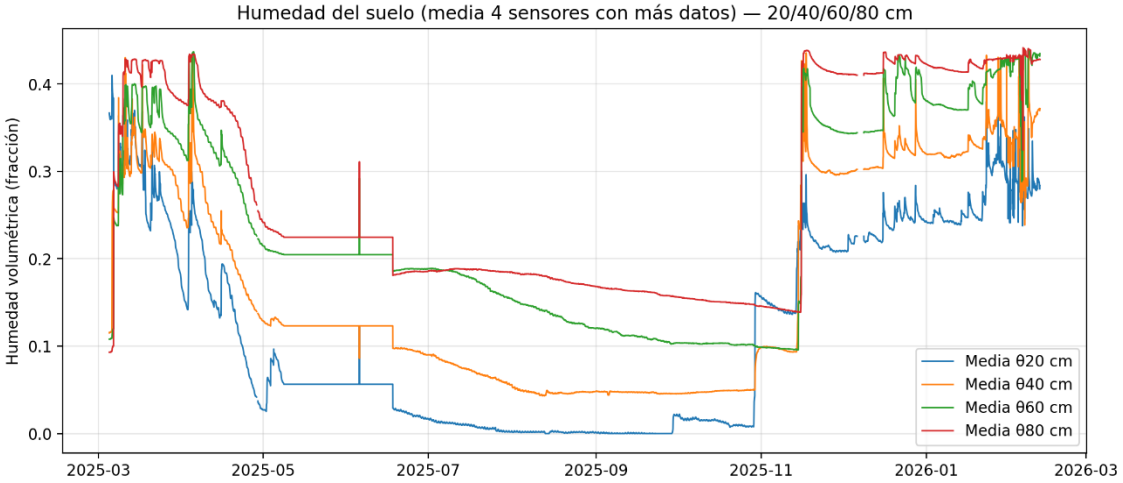


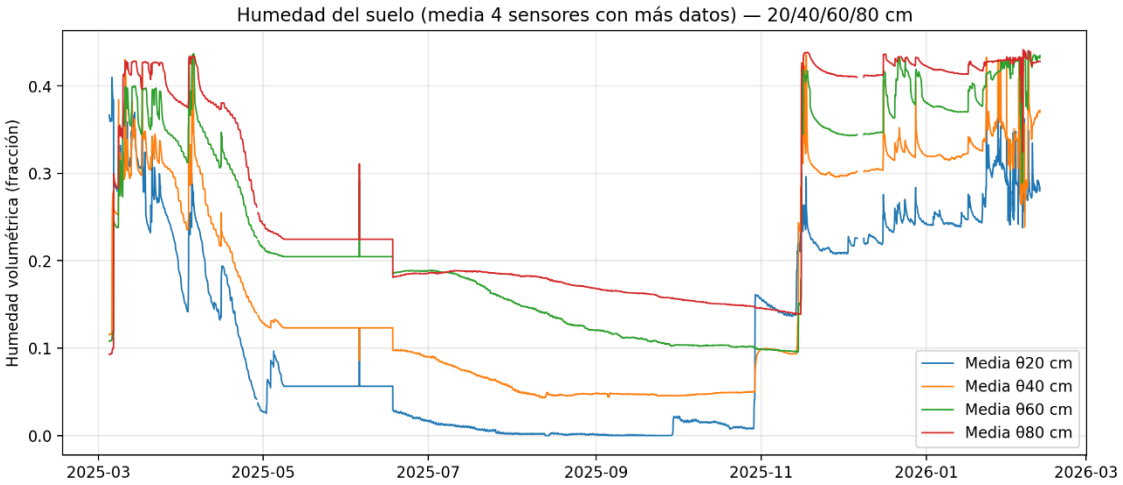



ETSIAM Escuela Técnica Superior de Ingeniería Agronómica y de Montes UNIVERSIDAD DE CÓRDOBA
UNIVERSIDAD DE CÓRDOBA
Junta de Andalucía Consejería de Universidad, Investigación e Innovación

Montaje de frames del vídeo (perfil 1) — 14 secciones

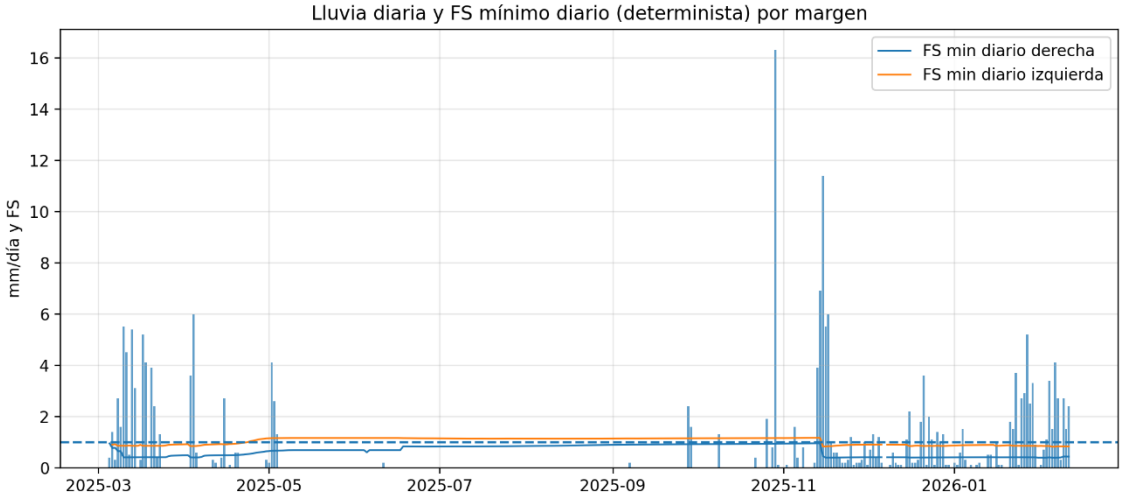


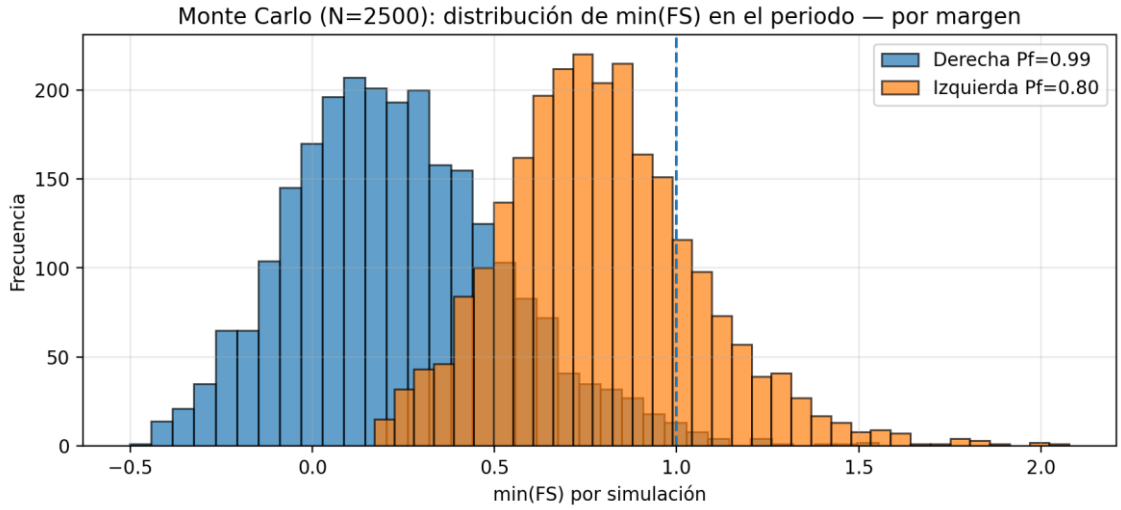


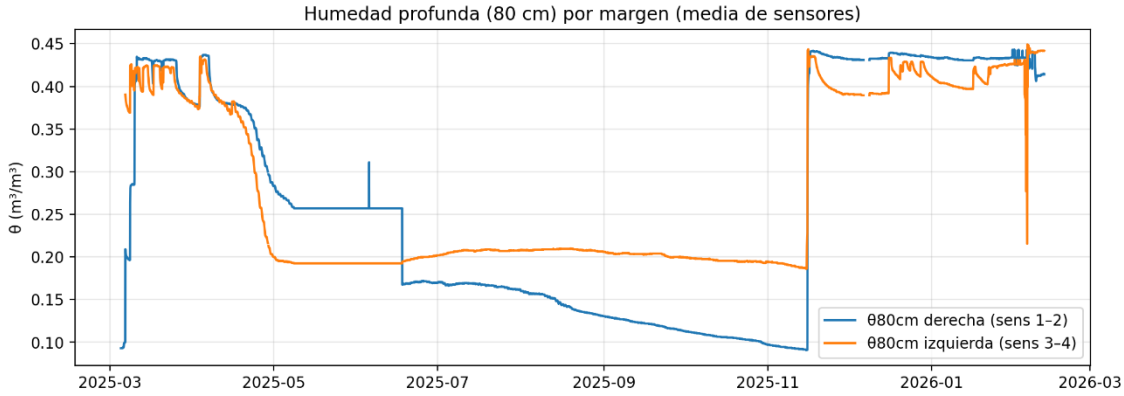



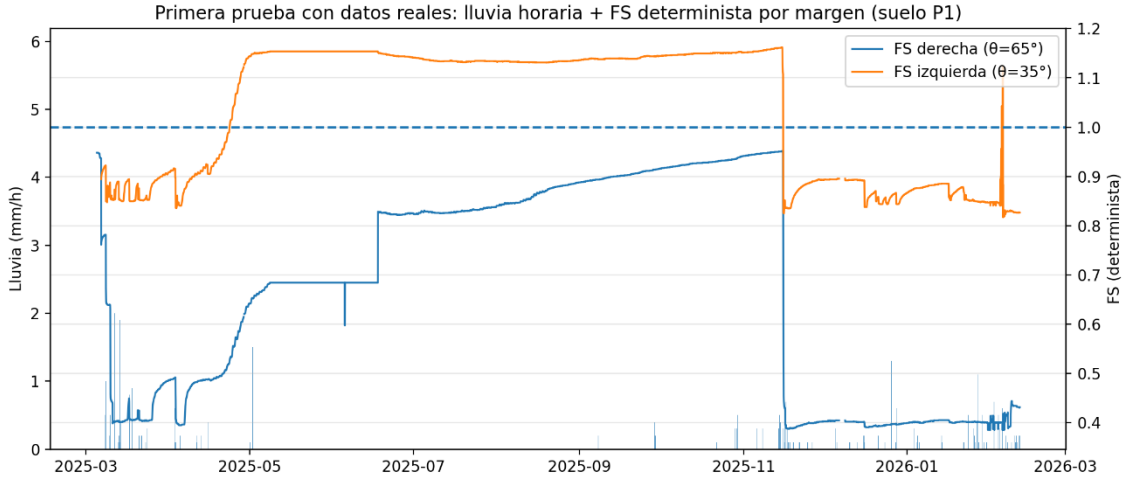
## 13.2 Resultados

**Partida**

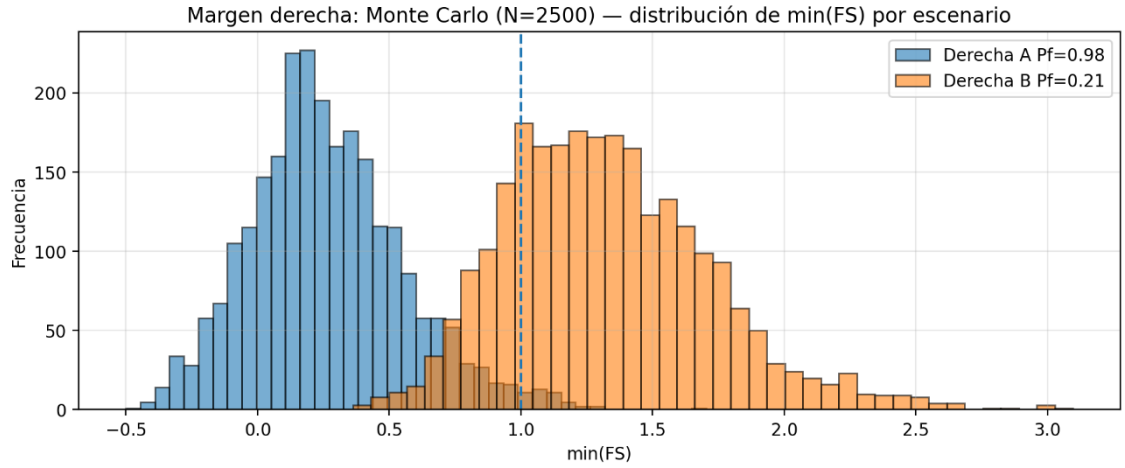


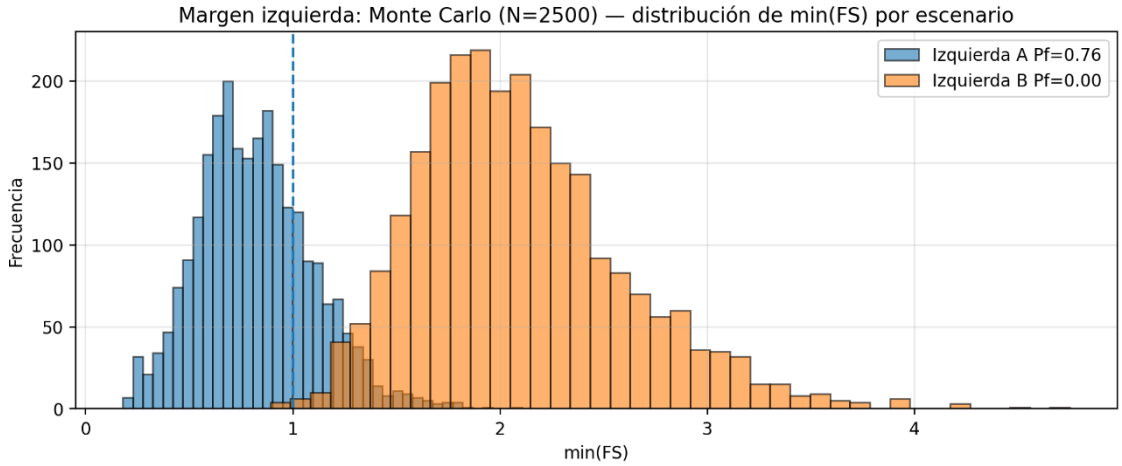



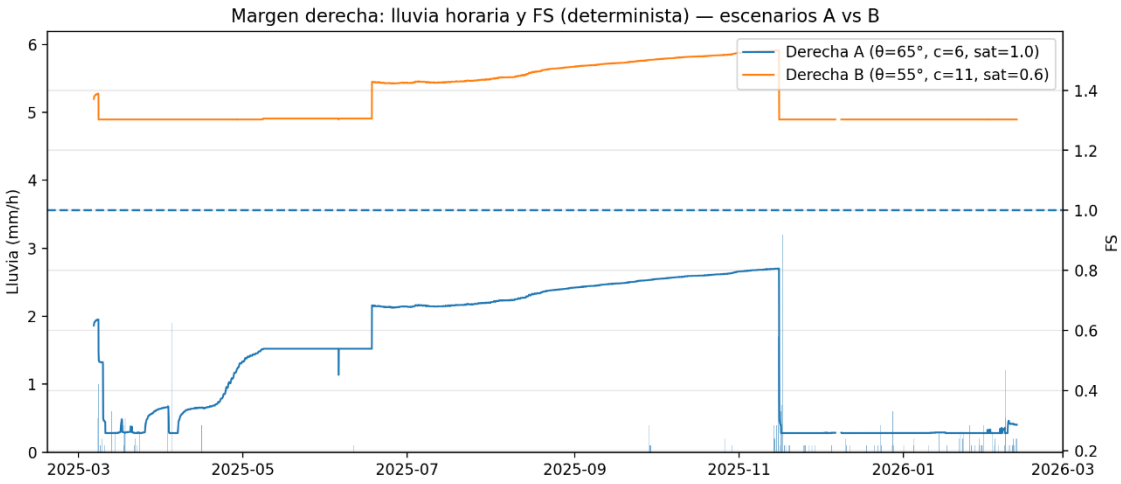

Humedad profunda (80 cm) por margen (media de sensores)
θ (m³/m³)
θ80cm derecha (sens 1–2)
θ80cm izquierda (sens 3–4)
2025-03
2025-05
2025-07
2025-09
2025-11
2026-01
2026-03

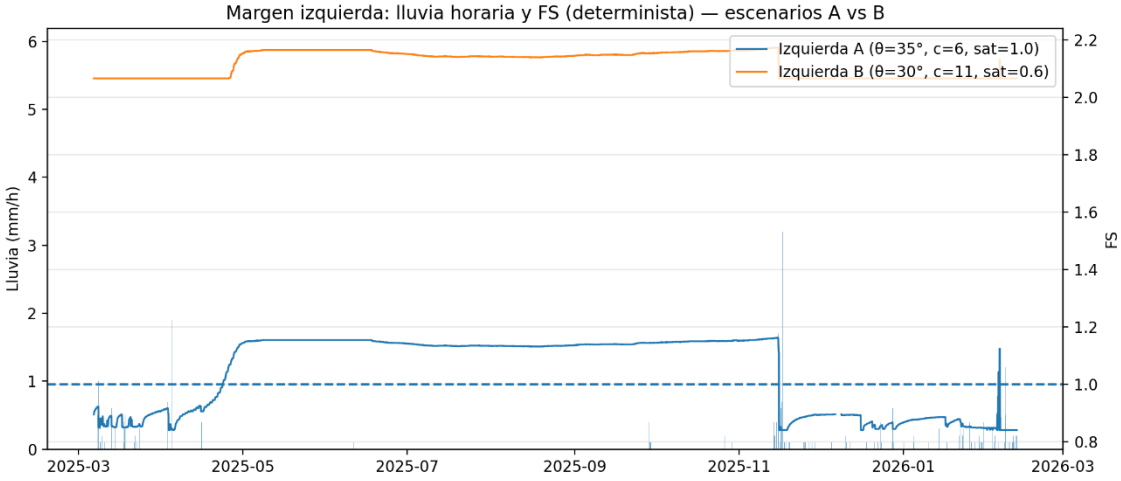

Primera prueba con datos reales: lluvia horaria + FS determinista por margen (suelo P1)
Lluvia (mm/h)
FS (determinista)
FS derecha (θ=65°)
FS izquierda (θ=35°)
2025-03
2025-05
2025-07
2025-09
2025-11
2026-01
2026-03

## Escenarios


Margen derecha: Monte Carlo (N=2500) — distribución de min(FS) por escenario
Frecuencia
min(FS)
Derecha A Pf=0.98
Derecha B Pf=0.21






ETSIAM Escuela Técnica Superior de Ingeniería Agronómica y de Montes UNIVERSIDAD D CÓRDOBA
UNIVERSIDAD D CÓRDOBA
Junta de Andalucía Consejería de Universidad, Investigación e Innovación

## 14. Código

```
# ============================================================
# MODELO CONTINUO ESTABILIDAD DE LADERAS EN CÁRCAVAS
# ============================================================
# Autor: Adolfo Peña (UCO) + ChatGPT
# Objetivo:
#   - Modelo continuo lluvia → humedad → FS
#   - Monte Carlo (incertidumbre c', phi', Ksat, gamma, H)
#   - Curvas I-D probabilísticas y comparación entre fincas
#
# Entradas recomendadas por finca:
#   data/<FincaX>/meteo.csv    columnas: fecha, P_mm_h, ta_c (opt),
rh_pct (opt)
#   data/<FincaX>/humedad.csv  columnas: fecha, superficial, media,
profunda  (m3/m3)
#   (opcional) data/<FincaX>/eventos.csv columnas: fecha_inicio,
fecha_fin o duracion_h, etiqueta_fallo(0/1)
#
# Salidas:
#   outputs/<FincaX>/*.csv, *.png
#   outputs/comparacion_fincas/*.png
# ============================================================

from __future__ import annotations
import os
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# ------------------------------------------------------------
# 0) CONFIG / UTILIDADES
# ------------------------------------------------------------

def ensure_dir(path: str) -> None:
    os.makedirs(path, exist_ok=True)

def to_datetime_index(df: pd.DataFrame, col="fecha") -> pd.DataFrame:
    if col in df.columns:
        df[col] = pd.to_datetime(df[col])
        df = df.set_index(col)
    df = df.sort_index()
    return df

def resample_to_hourly(series_df: pd.DataFrame, how="mean") ->
pd.DataFrame:
    """
    Convierte series subhorarias a horarias.
    - lluvia: suele ser sum (mm por hora) o mean (mm/h). Si tu meteo es
P_mm_h como intensidad,
      y viene cada 10 min, lo mejor es media.
    - humedad: mean.
    """
    if how == "sum":
        return series_df.resample("1H").sum()
```

```
    return series_df.resample("1H").mean()

def clip_df(df: pd.DataFrame, lo=0.0, hi=1.0) -> pd.DataFrame:
    return df.clip(lower=lo, upper=hi)

# ------------------------------------------------------------------
# 1) SUELO BASE P1 (EXPS08-P1) + PDFs PARA MONTE CARLO
# ------------------------------------------------------------------

SOIL_P1_FIXED = {
    # Directos / derivados
    "theta_fc": 0.3323,
    "theta_wp": 0.2346,
    "theta_s": 0.4315,       # porosidad (fine earth)
    "theta_s_eff": 0.2848,  # porosidad ajustada por CF (si la quieres
usar)
    "rho_b_g_cm3": 1.5065,
    "coarse_frag_frac": 0.34,
    "texture_pct": {"sand": 14.10, "silt": 36.47, "clay": 49.43},
}

# PDFs (estimadas) para completar estabilidad e infiltración
SOIL_P1_PDFS = {
    "ksat": {"dist": "lognormal", "median_mm_h": 1.5, "gstd": 3.0},
    "c":    {"dist": "normal_trunc", "mu_kpa": 6.0, "sigma_kpa": 2.0,
"min_kpa": 1.0, "max_kpa": 12.0},
    "phi":  {"dist": "normal_trunc", "mu_deg": 26.0, "sigma_deg": 3.0,
"min_deg": 18.0, "max_deg": 35.0},
    "gamma":{"dist": "normal_trunc", "mu": 18.0, "sigma": 0.8, "min":
16.0, "max": 20.0},  # kN/m3
    "H":    {"dist": "triangular", "min_m": 0.6, "mode_m": 1.0, "max_m":
1.6},
}

def trunc_normal(mu, sigma, lo, hi, size=1, rng=None):
    rng = rng or np.random.default_rng()
    x = rng.normal(mu, sigma, size=size)
    return np.clip(x, lo, hi)

def lognormal_from_median_gstd(median, gstd, size=1, rng=None):
    rng = rng or np.random.default_rng()
    mu = np.log(median)
    sigma = np.log(gstd)
    return rng.lognormal(mean=mu, sigma=sigma, size=size)

def triangular(min_v, mode_v, max_v, size=1, rng=None):
    rng = rng or np.random.default_rng()
    return rng.triangular(min_v, mode_v, max_v, size=size)

def sample_soil_params_P1(N: int, rng=None) -> dict[str, np.ndarray]:
    rng = rng or np.random.default_rng()
    Ksat =
lognormal_from_median_gstd(SOIL_P1_PDFS["ksat"]["median_mm_h"],
SOIL_P1_PDFS["ksat"]["gstd"], size=N, rng=rng)
    c    = trunc_normal(SOIL_P1_PDFS["c"]["mu_kpa"],
SOIL_P1_PDFS["c"]["sigma_kpa"],
```

```
                          SOIL_P1_PDFS["c"]["min_kpa"],
SOIL_P1_PDFS["c"]["max_kpa"], size=N, rng=rng)
    phi   = trunc_normal(SOIL_P1_PDFS["phi"]["mu_deg"],
SOIL_P1_PDFS["phi"]["sigma_deg"],
                          SOIL_P1_PDFS["phi"]["min_deg"],
SOIL_P1_PDFS["phi"]["max_deg"], size=N, rng=rng)
    gamma= trunc_normal(SOIL_P1_PDFS["gamma"]["mu"],
SOIL_P1_PDFS["gamma"]["sigma"],
                          SOIL_P1_PDFS["gamma"]["min"],
SOIL_P1_PDFS["gamma"]["max"], size=N, rng=rng)
    H     = triangular(SOIL_P1_PDFS["H"]["min_m"],
SOIL_P1_PDFS["H"]["mode_m"], SOIL_P1_PDFS["H"]["max_m"], size=N, rng=rng)
    return {"Ksat": Ksat, "c": c, "phi": phi, "gamma_s": gamma, "H": H}

# ------------------------------------------------------------------
# 2) MODELO HIDROLÓGICO SIMPLE POR CAPAS (3 capas)
# ------------------------------------------------------------------

def balance_hidrico_3capas(lluvia_mm_h: pd.Series,
                           params: dict,
                           theta_ini: list[float],
                           Z_mm: list[float],
                           theta_s: float,
                           ET0_mm_h: float = 0.05) -> pd.DataFrame:
    """
    Balance hídrico simple:
    - infiltración limitada por Ksat (mm/h)
    - percolación de excedente cuando capa satura
    - ET0 constante (puedes sustituir por ET0(t))
    """
    n = len(lluvia_mm_h)
    theta = np.zeros((n, 3), dtype=float)
    theta[0, :] = np.array(theta_ini, dtype=float)

    Ksat = float(params["Ksat"])
    Z = np.array(Z_mm, dtype=float)
    # theta_s por capa (mismo para las 3, simplificado)
    theta_s_arr = np.array([theta_s, theta_s, theta_s], dtype=float)

    for t in range(1, n):
        P = float(lluvia_mm_h.iloc[t])  # mm/h

        infil = min(P, Ksat)           # mm/h
        # excedente superficial (no lo usamos aquí; podría ir a
escorrentía)
        # exced = max(0.0, P - Ksat)

        # Capa 1
        theta[t, 0] = theta[t-1, 0] + (infil - ET0_mm_h) / Z[0]
        percola = 0.0
        if theta[t, 0] > theta_s_arr[0]:
            percola = (theta[t, 0] - theta_s_arr[0]) * Z[0]
            theta[t, 0] = theta_s_arr[0]
        theta[t, 0] = max(0.0, theta[t, 0])

        # Capa 2
```

```
        theta[t, 1] = theta[t-1, 1] + (percola - ET0_mm_h) / Z[1]
        percola2 = 0.0
        if theta[t, 1] > theta_s_arr[1]:
            percola2 = (theta[t, 1] - theta_s_arr[1]) * Z[1]
            theta[t, 1] = theta_s_arr[1]
        theta[t, 1] = max(0.0, theta[t, 1])

        # Capa 3
        theta[t, 2] = theta[t-1, 2] + (percola2 - ET0_mm_h) / Z[2]
        theta[t, 2] = np.clip(theta[t, 2], 0.0, theta_s_arr[2])

    return pd.DataFrame(theta, index=lluvia_mm_h.index,
columns=["superficial", "media", "profunda"])

# ------------------------------------------------------------------
# 3) ESTABILIDAD: TALUD INFINITO (FS)
# ------------------------------------------------------------------

def factor_seguridad_infinite_slope(c_kpa: float,
                                    phi_deg: float,
                                    gamma_s_kN_m3: float,
                                    H_m: float,
                                    pendiente_deg: float,
                                    hw_m: float,
                                    gamma_w_kN_m3: float = 9.81) ->
float:
    """
    FS = [c' + (σn - u) tanφ] / τ
    Aproximación con:
      σn ≈ γ_s * H * cos^2(θ)
      u  ≈ γ_w * hw
      τ  ≈ γ_s * H * sinθ * cosθ
    """
    th = np.radians(pendiente_deg)
    sigma_n = gamma_s_kN_m3 * H_m * (np.cos(th)**2)
    u = gamma_w_kN_m3 * hw_m
    num = c_kpa + (sigma_n - u) * np.tan(np.radians(phi_deg))
    den = gamma_s_kN_m3 * H_m * np.sin(th) * np.cos(th)
    # evita divisiones raras
    if den <= 1e-9:
        return np.nan
    return num / den

def fs_series_from_theta(humedad: pd.DataFrame,
                         params: dict,
                         theta_s: float,
                         pendiente_deg: float) -> pd.Series:
    """
    Vincula humedad profunda con altura saturada:
      hw(t) = H * (theta_profunda / theta_s)
    """
    H = float(params["H"])
    c = float(params["c"])
    phi = float(params["phi"])
    gamma_s = float(params["gamma_s"])
```

```
    hw = H * (humedad["profunda"].values / max(theta_s, 1e-9))
    fs = [factor_seguridad_infinite_slope(c, phi, gamma_s, H,
pendiente_deg, float(h)) for h in hw]
    return pd.Series(fs, index=humedad.index, name="FS")

# ------------------------------------------------------------------
# 4) MONTE CARLO: probabilidad de fallo en un periodo (FS<1)
# ------------------------------------------------------------------

def monte_carlo_period_failure(lluvia_mm_h: pd.Series,
                               theta_ini: list[float],
                               Z_mm: list[float],
                               pendiente_deg: float,
                               N: int = 1000,
                               ET0_mm_h: float = 0.05,
                               rng=None) -> dict:
    """
    Simula N realizaciones de parámetros de suelo y devuelve:
      - P_fail_periodo: probabilidad de que min(FS)<1 en el periodo
      - FSmin: array con mínimos por simulación
    """
    rng = rng or np.random.default_rng()
    samples = sample_soil_params_P1(N, rng=rng)
    FS_min = np.zeros(N, dtype=float)

    for i in range(N):
        pars = {
            "Ksat": float(samples["Ksat"][i]),
            "c": float(samples["c"][i]),
            "phi": float(samples["phi"][i]),
            "gamma_s": float(samples["gamma_s"][i]),
            "H": float(samples["H"][i]),
        }
        hum_sim = balance_hidrico_3capas(lluvia_mm_h, pars, theta_ini,
Z_mm,

theta_s=SOIL_P1_FIXED["theta_s"],
                                         ET0_mm_h=ET0_mm_h)
        FS = fs_series_from_theta(hum_sim, pars,
theta_s=SOIL_P1_FIXED["theta_s"], pendiente_deg=pendiente_deg)
        FS_min[i] = np.nanmin(FS.values)

    P_fail = float(np.mean(FS_min < 1.0))
    return {"P_fail_periodo": P_fail, "FSmin": FS_min}

# ------------------------------------------------------------------
# 5) EVENTOS DE LLUVIA: extracción y métricas (I, D)
# ------------------------------------------------------------------

def extract_rain_events(lluvia_mm_h: pd.Series,
                        min_gap_h: int = 6,
                        min_total_mm: float = 1.0) -> pd.DataFrame:
    """
    Identifica eventos de lluvia a partir de serie horaria (mm/h).
    - min_gap_h: horas consecutivas sin lluvia para separar eventos
    - min_total_mm: descarta eventos muy pequeños
```

```
    Devuelve dataframe con: start, end, duracion_h, total_mm, I_mean,
I_max
    """
    rain = lluvia_mm_h.fillna(0.0).copy()
    is_rain = rain > 0.0

    events = []
    i = 0
    idx = rain.index

    while i < len(rain):
        if not is_rain.iloc[i]:
            i += 1
            continue

        start = idx[i]
        j = i
        gap = 0

        while j < len(rain):
            if is_rain.iloc[j]:
                gap = 0
            else:
                gap += 1
                if gap >= min_gap_h:
                    break
            j += 1

        end = idx[j-1]
        ev = rain.loc[start:end]
        total = float(ev.sum())  # si es mm/h a resolución 1h, suma = mm
        dur = int(len(ev))
        if total >= min_total_mm:
            events.append({
                "start": start,
                "end": end,
                "duracion_h": dur,
                "total_mm": total,
                "I_mean_mm_h": float(ev.mean()),
                "I_max_mm_h": float(ev.max()),
            })

        i = j

    return pd.DataFrame(events)

def label_events_by_FS(events: pd.DataFrame,
                       FS: pd.Series,
                       fs_threshold: float = 1.0) -> pd.DataFrame:
    """
    Etiqueta cada evento como 'fallo_modelo' si min FS durante evento <
umbral.
    """
    out = events.copy()
    labels = []
    fsmins = []
```

```
    for _, row in out.iterrows():
        sub = FS.loc[row["start"]:row["end"]]
        m = float(np.nanmin(sub.values)) if len(sub) else np.nan
        fsmins.append(m)
        labels.append(int(m < fs_threshold))
    out["FS_min"] = fsmins
    out["fallo_modelo"] = labels
    return out

# ---------------------------------------------------------------
# 6) CURVAS I-D PROBABILÍSTICAS
# ---------------------------------------------------------------

def generate_ID_prob_grid(pendiente_deg: float,
                          theta_ini: list[float],
                          Z_mm: list[float],
                          intensidades_mm_h: np.ndarray,
                          duraciones_h: np.ndarray,
                          N_mc: int = 300,
                          ET0_mm_h: float = 0.05,
                          rng=None) -> pd.DataFrame:
    """
    Para cada (I,D), crea un evento de lluvia constante y calcula P(FS<1)
(Monte Carlo).
    """
    rng = rng or np.random.default_rng()
    rows = []
    for I in intensidades_mm_h:
        for D in duraciones_h:
            idx = pd.date_range("2025-01-01", periods=int(D), freq="1H")
            lluvia = pd.Series(np.full(int(D), float(I)), index=idx,
name="P_mm_h")
            res = monte_carlo_period_failure(lluvia, theta_ini, Z_mm,
pendiente_deg,
                                             N=N_mc, ET0_mm_h=ET0_mm_h,
rng=rng)
            rows.append({"I_mm_h": float(I), "D_h": int(D), "P_fail":
res["P_fail_periodo"]})
    return pd.DataFrame(rows)

def plot_ID_prob(df: pd.DataFrame, out_png: str | None = None, title="I-D
Probabilidad de fallo"):
    plt.figure(figsize=(8, 6))
    sc = plt.scatter(df["D_h"], df["I_mm_h"], c=df["P_fail"], s=90,
edgecolor="k")
    plt.colorbar(sc, label="P(fallo) = P(min(FS)<1)")
    plt.xlabel("Duración (h)")
    plt.ylabel("Intensidad (mm/h)")
    plt.title(title)
    plt.grid(True)
    if out_png:
        plt.savefig(out_png, dpi=300, bbox_inches="tight")
    plt.show()

# ---------------------------------------------------------------
# 7) GRAFICADO SERIE TIEMPO
```

```
# ------------------------------------------------------------------

def plot_timeseries(lluvia_mm_h: pd.Series,
                    humedad_obs: pd.DataFrame | None,
                    humedad_sim: pd.DataFrame,
                    FS: pd.Series,
                    out_png: str | None = None,
                    title="Serie temporal lluvia-humedad-FS"):
    fig, ax1 = plt.subplots(figsize=(12, 6))
    ax1.bar(lluvia_mm_h.index, lluvia_mm_h.values, width=0.03, alpha=0.6)
    ax1.set_ylabel("Precipitación (mm/h)")
    ax1.set_xlabel("Tiempo")

    ax2 = ax1.twinx()
    ax2.plot(FS.index, FS.values, linewidth=1.5)
    ax2.axhline(1.0, linestyle="--")
    ax2.set_ylabel("FS")

    plt.title(title)
    plt.grid(True, axis="y", alpha=0.3)
    if out_png:
        plt.savefig(out_png, dpi=300, bbox_inches="tight")
    plt.show()

    # Humedad
    plt.figure(figsize=(12, 5))
    humedad_sim.plot(ax=plt.gca())
    if humedad_obs is not None:
        for col in humedad_obs.columns:
            if col in humedad_sim.columns:
                plt.plot(humedad_obs.index, humedad_obs[col],
linestyle="--", alpha=0.7, label=f"obs_{col}")
        plt.legend()
    plt.title("Humedad simulada (línea) y observada (trazo)")
    plt.ylabel("θ (m³/m³)")
    plt.grid(True, alpha=0.3)
    if out_png:
        base, ext = os.path.splitext(out_png)
        plt.savefig(f"{base}_humedad{ext}", dpi=300, bbox_inches="tight")
    plt.show()

# ------------------------------------------------------------------
# 8) PIPELINE POR FINCA
# ------------------------------------------------------------------

def run_finca_pipeline(
    finca_id: str,
    meteo_csv: str,
    humedad_csv: str | None,
    pendiente_deg: float,
    theta_ini: list[float] = None,
    Z_mm: list[float] = None,
    ET0_mm_h: float = 0.05,
    out_dir_root: str = "outputs",
    do_ID: bool = True,
    intensidades_mm_h: np.ndarray | None = None,
```

```
    duraciones_h: np.ndarray | None = None,
    N_mc_ID: int = 300,
    N_mc_period: int = 1000,
    resample_humedad: bool = True,
    resample_lluvia: bool = True,
) -> dict:
    """
    Ejecuta:
      - lectura
      - (opcional) resample a horario
      - simulación humedad con parámetros "mediana" (Ksat = mediana)
      - FS determinista con parámetros "mediana"
      - eventos + etiquetado por FS
      - Monte Carlo periodo
      - curvas I-D probabilísticas (opcional)
    """
    ensure_dir(out_dir_root)
    out_dir = os.path.join(out_dir_root, finca_id)
    ensure_dir(out_dir)

    # Defaults
    theta_ini = theta_ini or [0.22, 0.24, 0.26]
    Z_mm = Z_mm or [300, 300, 400]

    # Leer meteo
    met = pd.read_csv(meteo_csv)
    met = to_datetime_index(met, "fecha")
    if "P_mm_h" not in met.columns:
        raise ValueError(f"{meteo_csv} debe contener columna P_mm_h")
    met = met[["P_mm_h"]].copy()

    if resample_lluvia:
        met = resample_to_hourly(met, how="mean")  # intensidad media por
hora

    lluvia = met["P_mm_h"].fillna(0.0)

    # Leer humedad (opcional)
    humedad_obs = None
    if humedad_csv and os.path.exists(humedad_csv):
        hum = pd.read_csv(humedad_csv)
        hum = to_datetime_index(hum, "fecha")
        # esperamos columnas superficial/media/profunda
        cols = [c for c in ["superficial", "media", "profunda"] if c in
hum.columns]
        humedad_obs = hum[cols].copy()
        if resample_humedad:
            humedad_obs = resample_to_hourly(humedad_obs, how="mean")
        humedad_obs = clip_df(humedad_obs, 0.0, 1.0)

    # Parámetros "mediana" para simulación determinista
    params_det = {
        "Ksat": float(SOIL_P1_PDFS["ksat"]["median_mm_h"]),
        "c": float(SOIL_P1_PDFS["c"]["mu_kpa"]),
        "phi": float(SOIL_P1_PDFS["phi"]["mu_deg"]),
        "gamma_s": float(SOIL_P1_PDFS["gamma"]["mu"]),
```

```
        "H": float(SOIL_P1_PDFS["H"]["mode_m"]),
    }

    # Simulación humedad determinista
    humedad_sim = balance_hidrico_3capas(lluvia, params_det, theta_ini,
Z_mm,

theta_s=SOIL_P1_FIXED["theta_s"],
                                         ET0_mm_h=ET0_mm_h)

    # FS determinista
    FS = fs_series_from_theta(humedad_sim, params_det,
theta_s=SOIL_P1_FIXED["theta_s"], pendiente_deg=pendiente_deg)

    # Export determinista
    humedad_sim.to_csv(os.path.join(out_dir, "humedad_sim.csv"))
    FS.to_csv(os.path.join(out_dir, "FS.csv"), header=True)

    # Eventos y etiqueta por FS
    eventos = extract_rain_events(lluvia, min_gap_h=6, min_total_mm=1.0)
    if len(eventos):
        eventos_fs = label_events_by_FS(eventos, FS, fs_threshold=1.0)
        eventos_fs.to_csv(os.path.join(out_dir, "eventos_lluvia_FS.csv"),
index=False)
    else:
        eventos_fs = eventos

    # Plot series
    plot_timeseries(lluvia, humedad_obs, humedad_sim, FS,
                    out_png=os.path.join(out_dir, "serie_tiempo.png"),
                    title=f"{finca_id} | lluvia-humedad-FS")

    # Monte Carlo periodo con lluvia real (probabilidad de que min(FS)<1
en el periodo)
    mc_period = monte_carlo_period_failure(lluvia, theta_ini, Z_mm,
pendiente_deg,
                                           N=N_mc_period,
ET0_mm_h=ET0_mm_h)
    pd.Series({
        "P_fail_periodo": mc_period["P_fail_periodo"],
        "FSmin_mean": float(np.mean(mc_period["FSmin"])),
        "FSmin_p05": float(np.quantile(mc_period["FSmin"], 0.05)),
        "FSmin_p50": float(np.quantile(mc_period["FSmin"], 0.50)),
        "FSmin_p95": float(np.quantile(mc_period["FSmin"], 0.95)),
    }).to_csv(os.path.join(out_dir, "montecarlo_periodo_resumen.csv"),
header=False)

    plt.figure(figsize=(7, 4))
    plt.hist(mc_period["FSmin"], bins=30, edgecolor="k", alpha=0.8)
    plt.axvline(1.0, linestyle="--")
    plt.title(f"{finca_id} | Distribución min(FS) Monte Carlo (periodo)")
    plt.xlabel("min(FS) por simulación")
    plt.ylabel("Frecuencia")
    plt.grid(True, alpha=0.3)
    plt.savefig(os.path.join(out_dir, "hist_FSmin.png"), dpi=300,
bbox_inches="tight")
```

```
    plt.show()

    # Curvas I-D probabilísticas (grid)
    df_ID = None
    if do_ID:
        intensidades_mm_h = intensidades_mm_h if intensidades_mm_h is not
None else np.linspace(5, 60, 10)
        duraciones_h = duraciones_h if duraciones_h is not None else
np.array([1, 2, 3, 6, 12, 24, 48])

        df_ID = generate_ID_prob_grid(pendiente_deg, theta_ini, Z_mm,

intensidades_mm_h=intensidades_mm_h,
                                      duraciones_h=duraciones_h,
                                      N_mc=N_mc_ID,
                                      ET0_mm_h=ET0_mm_h)
        df_ID.to_csv(os.path.join(out_dir, "curvas_ID_prob.csv"),
index=False)
        plot_ID_prob(df_ID, out_png=os.path.join(out_dir,
"curvas_ID_prob.png"),
                     title=f"{finca_id} | I-D Probabilidad de fallo")

    return {
        "finca_id": finca_id,
        "out_dir": out_dir,
        "params_det": params_det,
        "P_fail_periodo": mc_period["P_fail_periodo"],
        "eventos": eventos_fs,
        "ID": df_ID
    }

# ------------------------------------------------------------------
# 9) COMPARACIÓN ENTRE FINCAS + EMPÍRICO
# ------------------------------------------------------------------

def compare_fincas_ID(results: dict[str, dict],
                      empirico: dict[str, pd.DataFrame] | None = None,
                      out_dir: str = "outputs/comparacion_fincas"):
    ensure_dir(out_dir)
    plt.figure(figsize=(9, 7))

    # líneas: prob media por D (agregada sobre intensidades) para cada
finca
    for finca_id, res in results.items():
        df = res.get("ID")
        if df is None or df.empty:
            continue
        # prob media por duración (promedio sobre intensidades)
        g = df.groupby("D_h")["P_fail"].mean().reset_index()
        plt.plot(g["D_h"], g["P_fail"], marker="o", label=f"{finca_id} (P
media)")

    # empírico: puntos (D, I) si los tienes
    if empirico:
        for finca_id, df_emp in empirico.items():
            if df_emp is None or df_emp.empty:
```

```
                continue
            # Espera columnas: duracion_h, intensidad_mm_h (o D_h,
I_mm_h)
            if "duracion_h" in df_emp.columns and "intensidad_mm_h" in
df_emp.columns:
                D = df_emp["duracion_h"].values
                I = df_emp["intensidad_mm_h"].values
            elif "D_h" in df_emp.columns and "I_mm_h" in df_emp.columns:
                D = df_emp["D_h"].values
                I = df_emp["I_mm_h"].values
            else:
                continue
            # aquí los ploteo en un segundo eje (porque no comparten
unidades con P_fail)
            # Para mantenerlo simple, los dejo como anotación en la misma
figura:
            for d, i in zip(D, I):
                plt.annotate(f"{finca_id}:I={i:.0f}", (d, 0.02),
fontsize=8, rotation=0)

    plt.xlabel("Duración (h)")
    plt.ylabel("Probabilidad media de fallo (promedio sobre I)")
    plt.title("Comparación entre fincas | Curvas I-D (resumen por
duración)")
    plt.grid(True, alpha=0.3)
    plt.legend()
    plt.savefig(os.path.join(out_dir, "comparacion_ID_prob_media.png"),
dpi=300, bbox_inches="tight")
    plt.show()

# ------------------------------------------------------------------
# 10) EJEMPLO DE EJECUCIÓN
# ------------------------------------------------------------------

if __name__ == "__main__":
    # ========================
    # EDITA ESTO A TU CASO REAL
    # ========================
    # Pendientes por finca (ejemplo)
    fincas = {
        "Finca_A": {"pendiente_deg": 35, "meteo_csv":
"data/Finca_A/meteo.csv", "humedad_csv": "data/Finca_A/humedad.csv"},
        "Finca_B": {"pendiente_deg": 33, "meteo_csv":
"data/Finca_B/meteo.csv", "humedad_csv": "data/Finca_B/humedad.csv"},
        "Finca_C": {"pendiente_deg": 37, "meteo_csv":
"data/Finca_C/meteo.csv", "humedad_csv": "data/Finca_C/humedad.csv"},
    }

    # Profundidades/capas (mm) y theta_ini (puedes ajustar por finca)
    Z_mm = [300, 300, 400]
    theta_ini = [0.22, 0.24, 0.26]

    # Grid I-D
    intensidades = np.linspace(5, 60, 10)
    duraciones = np.array([1, 2, 3, 6, 12, 24, 48])
```

```
    results = {}
    for finca_id, info in fincas.items():
        if not os.path.exists(info["meteo_csv"]):
            print(f"[WARN] No existe {info['meteo_csv']} (saltando
{finca_id})")
            continue

        res = run_finca_pipeline(
            finca_id=finca_id,
            meteo_csv=info["meteo_csv"],
            humedad_csv=info["humedad_csv"] if
os.path.exists(info["humedad_csv"]) else None,
            pendiente_deg=float(info["pendiente_deg"]),
            theta_ini=theta_ini,
            Z_mm=Z_mm,
            ET0_mm_h=0.05,           # ajustar después (o calcular de
meteo)
            out_dir_root="outputs",
            do_ID=True,
            intensidades_mm_h=intensidades,
            duraciones_h=duraciones,
            N_mc_ID=300,
            N_mc_period=1000,
            resample_humedad=True,
            resample_lluvia=True,
        )
        results[finca_id] = res

    # Comparación fincas
    if results:
        compare_fincas_ID(results, empirico=None,
out_dir="outputs/comparacion_fincas")
        print("Listo. Revisa outputs/ para CSV y PNG.")
    else:
        print("No se ejecutó ninguna finca: revisa rutas en 'fincas'.")
```