1. Modelo técnico y algoritmo implementado

En esta sección formalizamos con detalle matemático lo que se ha implementado en el código `py12_timeweighted.py` para ajustar tendencias penalizadas tipo Guerrero (2007) en una familia de series financieras y seleccionar el índice de suavización a partir de múltiples criterios de error.:contentReference[oaicite:0]index=0

1.1. Datos, universo de activos y transformación logarítmica

Sea $k \in \{1, \dots, K\}$ el índice de activo (ticker) y $t \in \{1, \dots, N_k\}$ el índice de tiempo (días de negociación).

- Para cada activo k observamos precios de cierre

$$P_{k,t} > 0, \qquad t = 1, \dots, N_k.$$

- En el estudio actual se consideran $K = 137$ activos mezclando: acciones grandes (AAPL, MSFT, ...), índices (`^GSPC`, `^MXX`, ...), criptomonedas (BTC–USD, ETH–USD, ...) y otros índices globales.

Sobre cada serie de precios $\{P_{k,t}\}$ se trabaja con *logprecios*

$$Z_{k,t} := \log P_{k,t}, \qquad t = 1, \dots, N_k.$$

Razones técnicas para trabajar con $\{Z_{k,t}\}$ en lugar de $\{P_{k,t}\}$:

1. **Adición de logretornos.** Los logretornos son

$$R_{k,t} := \log P_{k,t} - \log P_{k,t-1} = Z_{k,t} - Z_{k,t-1},$$

   de modo que productos de factores porcentuales en precios se convierten en sumas, lo cual es algebraicamente más estable.

2. **Escala y varianza.** Cambios porcentuales similares producen amplitudes comparables en $\{Z_{k,t}\}$ aunque los niveles de $P_{k,t}$ sean muy distintos; esto reduce heterocedasticidad inducida por el nivel.

3. **Positividad.** Si más adelante se requiere una tendencia sobre precios, se puede reconstruir como $\widehat{P}_{k,t} = \exp(\widehat{t}_{k,t}) > 0$.

En el código, la carga de datos se realiza mediante `load_sp500_series` (para cualquier ticker compatible con Yahoo Finance), que devuelve (t_k, Z_k, meta) con $t_k = 0, \dots, N_k - 1$ y, por defecto, `use_log=True`.:contentReference[oaicite:1]index=1

1.2. Operador de diferencia y matriz de penalización

Fijamos un orden de diferencia $d \in \{0, 1, 2, 3, 4\}$. Para una serie $\boldsymbol{\tau} = (\tau_0, \ldots, \tau_{N-1})^\top \in \mathbb{R}^N$ definimos la d-ésima *diferencia hacia adelante* en el índice t por

$$(\Delta^d \tau)_t := \sum_{k=0}^{d} (-1)^{d-k} \binom{d}{k} \tau_{t+k}, \qquad t = 0, \ldots, N-d-1.$$

Definición 1 (Matriz de diferencias)**.** Sea $N \in \mathbb{N}$ y $d \in \mathbb{N}_0$. Definimos $K \in \mathbb{R}^{(N-d)\times N}$ como la matriz que implementa Δ^d por la izquierda:

$$(K\boldsymbol{\tau})_r = (\Delta^d \tau)_r, \qquad r = 0, \ldots, N-d-1.$$

En componentes,

$$K_{r,j} := \begin{cases} (-1)^{d-(j-r)} \binom{d}{j-r}, & j \in \{r, \ldots, r+d\}, \\ 0, & \text{en otro caso.} \end{cases}$$

En el código, esto se construye mediante `difference_matrix(N,d)`:

- Para $d = 0$ se devuelve I_N.
- Para $d \geq 1$ se genera el vector de coeficientes binomiales

$$c_k = (-1)^{d-k} \binom{d}{k}, \quad k = 0, \ldots, d,$$

y se colocan como *ventanas deslizantes* a lo largo de la diagonal de K.:contentReference[oaicite:2]index=2

A partir de K definimos la matriz

$$B := K^\top K \in \mathbb{R}^{N\times N},$$

que es simétrica semidefinida positiva (PSD), ya que para todo $\boldsymbol{x} \in \mathbb{R}^N$ se cumple

$$\boldsymbol{x}^\top B \boldsymbol{x} = \|K\boldsymbol{x}\|_2^2 \;\geq\; 0.$$

1.3. Descomposición espectral y matriz de suavización

Definición 2 (Descomposición espectral de B)**.** Sea $B = K^\top K$. Calculamos su descomposición espectral

$$B = Q\Lambda Q^\top,$$

donde $Q \in \mathbb{R}^{N\times N}$ es ortogonal ($Q^\top Q = QQ^\top = I_N$) y $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_N)$ contiene los autovalores reales no negativos de B.:contentReference[oaicite:3]index=3

Para un parámetro de suavización $\lambda > 0$ definimos la matriz *de penalización* de Guerrero:

$$A(\lambda) := I_N + \lambda B \in \mathbb{R}^{N\times N}.$$

Usando la descomposición espectral, obtenemos

$$A(\lambda) = I_N + \lambda Q\Lambda Q^\top = Q(I_N + \lambda\Lambda)Q^\top,$$

y, en particular,

$$A(\lambda)^{-1} = Q\ \operatorname{diag}\left(\frac{1}{1+\lambda\lambda_i}\right)_{i=1}^{N} Q^\top.$$

Lema 1 (Traza de $A(\lambda)^{-1}$)**.** Se tiene

$$\operatorname{tr}\left(A(\lambda)^{-1}\right) = \sum_{i=1}^{N} \frac{1}{1+\lambda\lambda_i}.$$

Demostración. Por invariancia de la traza bajo conjugación ortogonal,

$$\operatorname{tr}\left(A(\lambda)^{-1}\right) = \operatorname{tr}\left(Q\operatorname{diag}\left(\frac{1}{1+\lambda\lambda_i}\right)Q^\top\right) = \operatorname{tr}\left(\operatorname{diag}\left(\frac{1}{1+\lambda\lambda_i}\right)\right) = \sum_{i=1}^{N}\frac{1}{1+\lambda\lambda_i}.$$

□

1.4. Índice de suavidad y mapeo $s \leftrightarrow \lambda$

Guerrero propone un *índice de suavidad crudo*

$$S_{\text{raw}}(\lambda) := 1 - \frac{1}{N}\operatorname{tr}\left(A(\lambda)^{-1}\right) = 1 - \frac{1}{N}\sum_{i=1}^{N}\frac{1}{1+\lambda\lambda_i}.$$

Este índice es creciente en λ y toma valores entre 0 y $S_{\text{máx}}$ con

$$S_{\text{máx}} = 1 - \frac{d}{N},$$

valor asociado al ajuste cuando la tendencia se aproxima a un polinomio de grado d (máxima suavidad posible dada la penalización en diferencias de orden d).

Definición 3 (Índice de suavidad normalizado)**.** Definimos el índice adimensional

$$s_{\text{unit}}(\lambda) := \frac{S_{\text{raw}}(\lambda)}{S_{\text{máx}}} \in [0, 1),$$

y lo interpretamos como la fracción de "suavidad máxima" alcanzada por el ajuste. En la implementación se trabaja con s_{unit} (dominado por `s_unit`) y se recupera λ numéricamente a partir de la ecuación

$$S_{\text{raw}}(\lambda) = s_{\text{unit}}\, S_{\text{máx}}.$$

En el código `lambda_from_s` se implementa este mapeo como una búsqueda unidimensional en $\lambda \geq 0$:

1. Se fija un objetivo
$$\text{target} = s_{\text{unit}} \cdot S_{\text{máx}}.$$
2. Se define
$$S_{\text{raw}}(\lambda) = 1 - \frac{1}{N} \sum_{i=1}^{N} \frac{1}{1 + \lambda\lambda_i}$$
usando los autovalores $\{\lambda_i\}$ precalculados de B.
3. Se construye un intervalo inicial $[0, \lambda_{\text{hi}}]$ ampliando λ_{hi} (multiplicando por 10) hasta que $S_{\text{raw}}(\lambda_{\text{hi}}) \geq \text{target}$.
4. Sobre ese intervalo se aplica un esquema de bisección que actualiza
$$\lambda_{\text{mid}} = \frac{1}{2}(\lambda_{\text{lo}} + \lambda_{\text{hi}})$$
y compara $S_{\text{raw}}(\lambda_{\text{mid}})$ con `target` hasta que
$$\left|S_{\text{raw}}(\lambda_{\text{mid}}) - \text{target}\right| < \texttt{tol}.$$

El resultado es un mapeo numéricamente estable $s_{\text{unit}} \mapsto \lambda$, que se reutiliza cada vez que se evalúa una función de pérdida en función de s.

1.5. Ajuste de tendencia penalizada y parámetro de deriva m

Para una serie observada $\boldsymbol{Z} \in \mathbb{R}^N$ (por ejemplo, el segmento de entrenamiento de $Z_{k,t}$) Guerrero propone ajustar una tendencia $\boldsymbol{t} \in \mathbb{R}^N$ que resuelva

$$\widehat{\boldsymbol{t}}(\lambda) := \arg\min_{\boldsymbol{t}} \left\{ \|\boldsymbol{Z} - \boldsymbol{t}\|_2^2 + \lambda \sum_{r=0}^{N-d-1} \big((K\boldsymbol{t})_r - m\big)^2 \right\}, \tag{1}$$

donde m es un escalar que representa la media del proceso $\Delta^d t_t$ (deriva en diferencias de orden d). En la implementación, m se estima de forma conjunta con $\boldsymbol{t}$ mediante un esquema de punto fijo.

En notación matricial, sea

$$\boldsymbol{u} := K\boldsymbol{t} \in \mathbb{R}^{N-d}, \qquad \mathbf{1} := (1, \ldots, 1)^\top \in \mathbb{R}^{N-d}.$$

La penalización se puede escribir como

$$\sum_{r=0}^{N-d-1} \big(u_r - m\big)^2 = \|\boldsymbol{u} - m\mathbf{1}\|_2^2.$$

El gradiente respecto de $\boldsymbol{t}$, para m fijo, da la ecuación normal

$$\Big(I_N + \lambda K^\top K\Big)\boldsymbol{t} = \boldsymbol{Z} + \lambda m K^\top \mathbf{1}.$$

Definimos, como antes, $B = K^\top K$ y $K\mathbf{1}_{\_(N-d)} = K^\top \mathbf{1}$. Entonces

$$A(\lambda)\boldsymbol{t} = \boldsymbol{Z} + \lambda m K^\top \mathbf{1}.$$

Definición 4 (Solución para m fijo)**.** Para un m dado, definimos

$$\boldsymbol{t}(m, \lambda) := A(\lambda)^{-1}\big(\boldsymbol{Z} + \lambda m K^\top \mathbf{1}\big).$$

El modelo impone además que m sea la media de $\Delta^d t_t$, es decir $m = \frac{1}{N-d}\mathbf{1}^\top K\boldsymbol{t}$. Esto induce una ecuación fija en m:

$$m = \frac{1}{N-d}\mathbf{1}^\top K\boldsymbol{t}(m, \lambda).$$

En el código se resuelve iterativamente:

1. Inicialización:

   $$m^{(0)} := \frac{1}{N-d}\mathbf{1}^\top K\boldsymbol{Z} = \text{media de las diferencias crudas de orden } d \text{ de } \boldsymbol{Z}.$$

2. Dado $m^{(k)}$, se calcula

   $$\boldsymbol{t}^{(k)} := A(\lambda)^{-1}\big(\boldsymbol{Z} + \lambda m^{(k)} K^\top \mathbf{1}\big),$$

   usando la representación espectral $A(\lambda)^{-1} = Q\,\mathrm{diag}(\alpha_i)Q^\top$ con $\alpha_i = 1/(1+\lambda\lambda_i)$.

3. Se actualiza

   $$m^{(k+1)} := \frac{1}{N-d}\mathbf{1}^\top K\boldsymbol{t}^{(k)}.$$

4. Se repiten los pasos hasta convergencia, $\left|m^{(k+1)} - m^{(k)}\right| < \texttt{m\_tol}$, o hasta un máximo de iteraciones.

El resultado de este procedimiento es

$$(\widehat{\boldsymbol{t}}, \widehat{m}, \widehat{\sigma}^2, \mathrm{diag}(A(\lambda)^{-1}), s_{\text{unit,real}})$$

para cada valor de s_{unit} pasado a la función `fit_for_s`, donde:

- $\widehat{\boldsymbol{t}} = \widehat{\boldsymbol{t}}_k^{\text{tr}}(d,s)$ es la tendencia ajustada en el segmento de entrenamiento.
- $\widehat{m}$ es la deriva estimada de $\Delta^d t_t$.
- $\widehat{\sigma}^2$ es una estimación tipo GLS de la varianza residual, con grados de libertad ajustados.
- $\mathrm{diag}(A(\lambda)^{-1})$ se computa como

  $$\mathrm{diag}(A(\lambda)^{-1}) = (Q \odot Q)\,\boldsymbol{\alpha},$$

  donde $\odot$ es el producto elemento a elemento, $\boldsymbol{\alpha}$ es el vector $(\alpha_1, \ldots, \alpha_N)^\top$ y $((Q \odot Q)\boldsymbol{\alpha})_j = \sum_i \alpha_i Q_{ji}^2$.
- $s_{\text{unit,real}}$ es el índice de suavidad $S_{\text{raw}}(\lambda)/S_{\text{máx}}$ recalculado a partir de la traza de $A(\lambda)^{-1}$.

1.6. Extensión polinómica global en diferencias

La penalización controla la estructura de $\Delta^d t_t$ en toda la serie. En la práctica, una vez ajustada $\widehat{\boldsymbol{t}}^{\mathrm{tr}}$ sobre el segmento de entrenamiento de longitud N^{tr}, se reconstruye una trayectoria polinómica global

$$\boldsymbol{p}^{(d,s)} = (p_0^{(d,s)}, \ldots, p_{N-1}^{(d,s)})^\top,$$

que:

- coincide con la tendencia ajustada en los últimos d puntos de entrenamiento;
- satisface la ecuación en diferencias

  $$\Delta^d p_t^{(d,s)} = \widehat{m}, \qquad \forall t,$$

  con $\widehat{m}$ la deriva obtenida arriba.

En el código esto se implementa en `build_polynomial_from_train_tail`.:contentReference[oaicite:

Anclaje y recursiones. Sea $j_0 = N^{\mathrm{tr}} - 1$ el último índice de entrenamiento, y $d_{\mathrm{eff}} = \mín(d, N^{\mathrm{tr}})$.

1. Se anclan los últimos d_{eff} puntos de entrenamiento en el polinomio:

   $$p_{j_0 - d_{\mathrm{eff}} + 1 : j_0} := \widehat{t}^{\,\mathrm{tr}}_{N^{\mathrm{tr}} - d_{\mathrm{eff}} + 1 : N^{\mathrm{tr}}}.$$

2. **Recursión hacia atrás**: usando la relación

   $$\sum_{k=0}^{d_{\mathrm{eff}}} (-1)^{d_{\mathrm{eff}} - k} \binom{d_{\mathrm{eff}}}{k} p_{j+k} = \widehat{m},$$

   y resolviendo para p_j en función de $p_{j+1}, \ldots, p_{j+d_{\mathrm{eff}}}$, se obtiene

   $$p_j = (-1)^{d_{\mathrm{eff}}} \left(\widehat{m} - \sum_{k=1}^{d_{\mathrm{eff}}} (-1)^{d_{\mathrm{eff}} - k} \binom{d_{\mathrm{eff}}}{k} p_{j+k} \right),$$

   que se aplica para $j = j_0 - d_{\mathrm{eff}}, \ldots, 0$.

3. **Recursión hacia adelante**: análogamente, se utiliza una expresión basada en $p_{i - d_{\mathrm{eff}}}, \ldots, p_{i-1}$ para construir p_i para $i = j_0 + 1, \ldots, N - 1$.

El resultado es que $\boldsymbol{p}^{(d,s)}$ es un polinomio en t de grado d cuyos coeficientes están determinados por la cola de $\widehat{\boldsymbol{t}}^{\mathrm{tr}}$ y la deriva $\widehat{m}$; esta trayectoria se interpreta como la "tendencia extrapolada" que usamos en validación y prueba.

1.7. Esquema de partición: entrenamiento, validación, prueba

Para cada ticker k se aplica un corte en tres partes contiguas sobre la serie $\boldsymbol{Z}_k$:

$$\boldsymbol{Z}_k = (\boldsymbol{Z}_k^{\mathrm{tr}}, \boldsymbol{Z}_k^{\mathrm{va}}, \boldsymbol{Z}_k^{\mathrm{te}}),$$

donde

$$\begin{aligned}
\boldsymbol{Z}_k^{\mathrm{tr}} &= (Z_{k,1}, \dots, Z_{k,N_k^{\mathrm{tr}}}),\\
\boldsymbol{Z}_k^{\mathrm{va}} &= (Z_{k,N_k^{\mathrm{tr}}+1}, \dots, Z_{k,N_k^{\mathrm{tr}}+N_k^{\mathrm{va}}}),\\
\boldsymbol{Z}_k^{\mathrm{te}} &= (Z_{k,N_k^{\mathrm{tr}}+N_k^{\mathrm{va}}+1}, \dots, Z_{k,N_k}).
\end{aligned}$$

En la función `train_val_test_split` se construye esta partición con fracciones aproximadas 0,6 (entrenamiento) y 0,2 (validación), imponiendo mínimos `min_train` y `min_val`, y asignando el resto a prueba.:contentReference[oaicite:5]index=5

1.8. Funciones objetivo en s y pesos temporales

Para un orden d fijo, y dados

$$(\boldsymbol{Z}^{\mathrm{tr}}, \boldsymbol{Z}^{\mathrm{va}}, \boldsymbol{Z}^{\mathrm{te}})$$

de longitudes $N^{\mathrm{tr}}, N^{\mathrm{va}}, N^{\mathrm{te}}$ (respectivamente), definimos, para cada s:

1. Ajuste de Guerrero en entrenamiento:

   $$(\widehat{\boldsymbol{t}}^{\mathrm{tr}}(s), \widehat{m}(s), \lambda(s), \dots) := \texttt{fit\_for\_s}(\boldsymbol{Z}^{\mathrm{tr}}, s).$$

2. Construcción del polinomio global $\boldsymbol{p}^{(d,s)}$ sobre todo el horizonte de longitud $N = N^{\mathrm{tr}} + N^{\mathrm{va}} + N^{\mathrm{te}}$. En particular

   $$\boldsymbol{p}^{\mathrm{tr}}(s) = \boldsymbol{p}^{(d,s)}_{1:N^{\mathrm{tr}}}, \quad \boldsymbol{p}^{\mathrm{va}}(s) = \boldsymbol{p}^{(d,s)}_{N^{\mathrm{tr}}+1:N^{\mathrm{tr}}+N^{\mathrm{va}}}, \quad \boldsymbol{p}^{\mathrm{te}}(s) = \boldsymbol{p}^{(d,s)}_{N^{\mathrm{tr}}+N^{\mathrm{va}}+1:N}.$$

3. Errores por segmento:

   $$\boldsymbol{e}^{\mathrm{tr}}(s) := \boldsymbol{p}^{\mathrm{tr}}(s) - \boldsymbol{Z}^{\mathrm{tr}}, \quad \boldsymbol{e}^{\mathrm{va}}(s) := \boldsymbol{p}^{\mathrm{va}}(s) - \boldsymbol{Z}^{\mathrm{va}}, \quad \boldsymbol{e}^{\mathrm{te}}(s) := \boldsymbol{p}^{\mathrm{te}}(s) - \boldsymbol{Z}^{\mathrm{te}}.$$

   Para entrenamiento+validación,

   $$\boldsymbol{e}^{\mathrm{tv}}(s) := \big(\boldsymbol{e}^{\mathrm{tr}}(s), \boldsymbol{e}^{\mathrm{va}}(s)\big) \in \mathbb{R}^{N^{\mathrm{tr}}+N^{\mathrm{va}}}.$$

Objetivos no ponderados. Definimos las MSE (objetivos) no ponderados:

$$J_{\mathrm{tr}}(s) := \frac{1}{N^{\mathrm{tr}}} \|\boldsymbol{e}^{\mathrm{tr}}(s)\|_2^2,$$
$$J_{\mathrm{va}}(s) := \frac{1}{N^{\mathrm{va}}} \|\boldsymbol{e}^{\mathrm{va}}(s)\|_2^2,$$
$$J_{\mathrm{both}}(s) := \frac{N^{\mathrm{tr}} J_{\mathrm{tr}}(s) + N^{\mathrm{va}} J_{\mathrm{va}}(s)}{N^{\mathrm{tr}} + N^{\mathrm{va}}}.$$

Pesos temporales linealmente crecientes. Para enfatizar errores hacia el final de cada segmento, definimos pesos

$$\tilde{w}_j^{(A)} := j, \quad j = 1, \ldots, N^A, \quad A \in \{\mathrm{va}, \mathrm{tv}\}$$

y normalizamos

$$w_j^{(A)} := \frac{\tilde{w}_j^{(A)}}{\sum_{u=1}^{N^A} \tilde{w}_u^{(A)}} = \frac{2j}{N^A(N^A+1)}.$$

El último índice tiene N^A veces más peso que el primero.

Las MSE ponderadas se definen como:

$$J_{\mathrm{va},w}(s) := \sum_{j=1}^{N^{\mathrm{va}}} w_j^{(\mathrm{va})} \big(e_j^{\mathrm{va}}(s)\big)^2,$$
$$J_{\mathrm{both},w}(s) := \sum_{j=1}^{N^{\mathrm{tv}}} w_j^{(\mathrm{tv})} \big(e_j^{\mathrm{tv}}(s)\big)^2.$$

En el código estas cinco funciones se computan en `analyze_all_objectives_for_d`: J_{tr}, J_{va}, J_{both}, $J_{\mathrm{va},w}$ y $J_{\mathrm{both},w}$.:contentReference[oaicite:6]index=6

1.9. RMSE no ponderadas y ponderadas por segmento

Además de las MSE se reportan errores medios cuadrados raíz (RMSE) por segmento, tanto simples como ponderados:

RMSE no ponderadas.

$$\mathrm{RMSE}^{\mathrm{tr}}(s) := \sqrt{\frac{1}{N^{\mathrm{tr}}}\|\boldsymbol{e}^{\mathrm{tr}}(s)\|_2^2},$$
$$\mathrm{RMSE}^{\mathrm{va}}(s) := \sqrt{\frac{1}{N^{\mathrm{va}}}\|\boldsymbol{e}^{\mathrm{va}}(s)\|_2^2},$$
$$\mathrm{RMSE}^{\mathrm{te}}(s) := \sqrt{\frac{1}{N^{\mathrm{te}}}\|\boldsymbol{e}^{\mathrm{te}}(s)\|_2^2},$$
$$\mathrm{RMSE}^{\mathrm{tv}}(s) := \sqrt{\frac{1}{N^{\mathrm{tr}}+N^{\mathrm{va}}}\|\boldsymbol{e}^{\mathrm{tv}}(s)\|_2^2}.$$

RMSE ponderadas. Usando los pesos $w^{(\mathrm{tr})}$, $w^{(\mathrm{va})}$, $w^{(\mathrm{te})}$ y $w^{(\mathrm{tv})}$ (definidos de forma análoga con crecimiento lineal dentro de cada segmento), definimos

$$\mathrm{RMSE}_w^{\mathrm{tr}}(s) := \sqrt{\sum_{j=1}^{N^{\mathrm{tr}}} w_j^{(\mathrm{tr})}\big(e_j^{\mathrm{tr}}(s)\big)^2},$$
$$\mathrm{RMSE}_w^{\mathrm{va}}(s) := \sqrt{\sum_{j=1}^{N^{\mathrm{va}}} w_j^{(\mathrm{va})}\big(e_j^{\mathrm{va}}(s)\big)^2},$$
$$\mathrm{RMSE}_w^{\mathrm{te}}(s) := \sqrt{\sum_{j=1}^{N^{\mathrm{te}}} w_j^{(\mathrm{te})}\big(e_j^{\mathrm{te}}(s)\big)^2},$$
$$\mathrm{RMSE}_w^{\mathrm{tv}}(s) := \sqrt{\sum_{j=1}^{N^{\mathrm{tr}}+N^{\mathrm{va}}} w_j^{(\mathrm{tv})}\big(e_j^{\mathrm{tv}}(s)\big)^2}.$$

Estas RMSE se evalúan para cada mínimo local de cada función objetivo en `rmse_all_segments_for_s` y se imprimen en el *verbose* del código, así como se utilizan en las figuras de `plot_d_train_val_test_strict`.:contentReference[oaicite:7]index=7

1.10. Búsqueda sobre rejilla y detección de mínimos locales

Para un d fijo se define una rejilla uniforme de suavidad:

$$\mathcal{S} = \{s_1, \ldots, s_M\} \subset [s_{\mathrm{mín}}, s_{\mathrm{máx}}],$$

típicamente con $M \approx 250500$. Para cada s_i:

1. Se calcula $\lambda(s_i)$ vía `lambda_from_s`.

2. Se ajusta $\widehat{\boldsymbol{t}}^{\mathrm{tr}}(s_i)$ y se construye $\boldsymbol{p}^{(d,s_i)}$.

3. Se evalúan las cinco funciones objetivo $J_{\mathrm{tr}}(s_i)$, $J_{\mathrm{va}}(s_i)$, $J_{\mathrm{both}}(s_i)$, $J_{\mathrm{va},w}(s_i)$, $J_{\mathrm{both},w}(s_i)$.

Para cada objetivo $J(\cdot)$ con valores en la rejilla $\{J(s_i)\}_{i=1}^{M}$ se detectan mínimos locales discretos:

Definición 5 (Mínimo local discreto en la rejilla)**.** Un índice $i \in \{2, \ldots, M-1\}$ es candidato a mínimo local si

$$J(s_i) \leq J(s_{i-1}) \quad \text{y} \quad J(s_i) \leq J(s_{i+1}),$$

y $J(s_i)$ es finito. También se consideran los extremos $i = 1$ y $i = M$ si satisfacen las desigualdades análogas.

Cada candidato $(s_i, J(s_i))$ se puede refinar mediante *goldensection search* sobre el intervalo local que lo rodea:

1. Dado un candidato i, se elige un intervalo $[a, b] \subset [s_{\mathrm{mín}}, s_{\mathrm{máx}}]$ alrededor de s_i.

2. Se aplica el algoritmo de sección áurea: si definimos $\varphi = (1 + \sqrt{5})/2$, y c, b como

$$c = b - \frac{1}{\varphi}(b - a), \qquad d = a + \frac{1}{\varphi}(b - a),$$

   se evalúan $J(c)$ y $J(d)$ y se desecha el subintervalo donde J es mayor, repitiendo hasta alcanzar un número fijo de iteraciones.

3. El punto medio del intervalo final se adopta como $s^\star$, y $J(s^\star)$ como valor refinado.

Esto se implementa de forma genérica en `find_all_local_minima_s` y `golden_local`, y luego se reutiliza en `analyze_all_objectives_for_d` para cada uno de los cinco objetivos.:contentReference[oaicite:8]index=8

1.11. Resumen simbólico

Para cada ticker k, orden de diferencia d y parámetro de suavidad s:

Matrices: $K \in \mathbb{R}^{(N-d)\times N}, \quad B = K^\top K, \quad A(\lambda) = I_N + \lambda B,$

Espectral: $B = Q\Lambda Q^\top, \quad A(\lambda)^{-1} = Q\,\mathrm{diag}\left(\frac{1}{1+\lambda\lambda_i}\right)Q^\top,$

Suavidad: $S_{\mathrm{raw}}(\lambda) = 1 - \frac{1}{N}\sum_{i=1}^{N}\frac{1}{1+\lambda\lambda_i}, \quad s_{\mathrm{unit}}(\lambda) = \frac{S_{\mathrm{raw}}(\lambda)}{1-d/N},$

Trend Guerrero: $\widehat{\boldsymbol{t}}^{\mathrm{tr}}(s),\ \widehat{m}(s)$ a partir de punto fijo en $m = \frac{1}{N-d}\mathbf{1}^\top K\boldsymbol{t},$

Polinomio global: $\boldsymbol{p}^{(d,s)}$ tal que $\Delta^d p_t^{(d,s)} = \widehat{m}(s)$ y $p_t^{(d,s)} \approx \widehat{t}_t^{\mathrm{tr}}(s)$ en la cola de entrenamiento,

Errores: $\boldsymbol{e}^A(s) = \boldsymbol{p}^A(s) - \boldsymbol{Z}^A, \quad A \in \{\mathrm{tr}, \mathrm{va}, \mathrm{tv}, \mathrm{te}\},$

MSE/RMSE: $J_A(s) = \frac{1}{N^A}\|\boldsymbol{e}^A(s)\|_2^2, \quad J_{A,w}(s) = \sum_j w_j^{(A)}(e_j^A(s))^2,$

$$\mathrm{RMSE}^A(s) = \sqrt{J_A(s)}, \quad \mathrm{RMSE}_w^A(s) = \sqrt{\sum_j w_j^{(A)}(e_j^A(s))^2}.$$

La tarea de selección de suavizado consiste en estudiar, para cada (k, d), las curvas $s \mapsto J_\bullet(s)$ en una rejilla, identificar todos sus mínimos locales (posiblemente refinados por sección áurea) y caracterizar el desempeño asociado en términos de RMSE (simple y ponderada) en entrenamiento, validación y prueba.

2. Estructura polinómica de la tendencia y posibles extensiones

En esta sección explicamos con más detalle por qué la tendencia extrapolada que construimos es un polinomio de grado d, por qué en los experimentos se restringió a órdenes $d \in \{1, 2, 3, 4\}$, y cómo se podría generalizar el modelo para que la tendencia pronosticada sea una serie suavizada más flexible y no necesariamente un polinomio de grado $fijo$.

2.1. Por qué la tendencia extrapolada es un polinomio de grado d

Recordemos el operador de diferencia hacia adelante de orden d sobre una sucesión $\{p_t\}_{t\in\mathbb{Z}}$:

$$(\Delta^d p)_t := \sum_{k=0}^{d} (-1)^{d-k} \binom{d}{k} p_{t+k}, \qquad t \in \mathbb{Z}.$$

En nuestro esquema de extensión, después de ajustar la tendencia de Guerrero en el segmento de entrenamiento, imponemos fuera de la muestra la condición

$$\Delta^d p_t = \widehat{m}, \qquad \text{para todo } t \text{ en el segmento extendido (val+test)}, \tag{2}$$

donde $\widehat{m}$ es la estimación de la media de $\Delta^d t_t$ que sale del ajuste penalizado.

Tomando una diferencia adicional, se obtiene

$$\Delta^{d+1} p_t = \Delta\big(\Delta^d p_t\big) = \Delta(\widehat{m}) = 0,$$

es decir, la sucesión $\{p_t\}$ satisface

$$\Delta^{d+1} p_t = 0 \qquad \forall t. \tag{3}$$

Lo anterior es la versión discreta de la $(d+1)$-ésima derivada es cero, que en el contexto continuo caracteriza a polinomios de grado a lo sumo d. En el contexto discreto se tiene el análogo siguiente.

Lema 2. Sea $\{p_t\}_{t\in\mathbb{Z}}$ una sucesión real tal que $\Delta^{d+1} p_t = 0$ para todo t. Entonces existe un polinomio $q : \mathbb{Z} \to \mathbb{R}$ de grado a lo sumo d tal que

$$p_t = q(t), \qquad \forall t.$$

Además, si $\Delta^d p_t = \widehat{m} \neq 0$ para todo t, el polinomio q tiene grado exactamente d.

Esbozo de demostración. La prueba se puede hacer por inducción en d o usando la base combinatoria $\left\{\binom{t}{0}, \ldots, \binom{t}{d}\right\}$.

Base $d = 0$**.** La condición $\Delta p_t = 0$ equivale a $p_{t+1} - p_t = 0$, de donde p_t es constante, es decir, un polinomio de grado 0.

Paso inductivo. Supongamos que la afirmación es cierta para orden d. Sea ahora $d+1$ y supongamos $\Delta^{d+2} p_t = 0$. Definamos $q_t := \Delta p_t = p_{t+1} - p_t$. Entonces

$$\Delta^{d+1} q_t = \Delta^{d+1}(\Delta p_t) = \Delta^{d+2} p_t = 0.$$

Por hipótesis inductiva, existe un polinomio $r(t)$ de grado a lo sumo d tal que $q_t = r(t)$ para todo t. Esto es,

$$p_{t+1} - p_t = r(t),$$

y por tanto

$$p_{t+1} = p_0 + \sum_{u=0}^{t} r(u).$$

Pero la suma finita de un polinomio de grado d en u es, de nuevo, un polinomio en t de grado $d+1$ (esto es la versión discreta de que la integral de un polinomio de grado d es un polinomio de grado $d+1$). Así, p_t es polinómico de grado a lo sumo $d+1$.

Aplicando el argumento con d reemplazado por $d+1$ y la condición $\Delta^{d+1} p_t = 0$, se concluye que p_t es polinomio de grado a lo sumo d. La afirmación sobre grado exactamente d cuando $\Delta^d p_t$ es una constante no nula se deduce de que la diferencia de orden d de un polinomio anula todos los términos de grado menor a d y produce una constante proporcional al coeficiente de t^d. □

Aplicando el Lema 2 a nuestra extensión, la condición (3) implica que la trayectoria extrapolada $\{p_t^{(d,s)}\}$ que construimos es exactamente un polinomio de grado d (salvo casos degenerados donde la estimación $\widehat{m}$ resulte cero y otras constantes también se anulen).

En otras palabras:

$$p_t^{(d,s)} = \beta_0 + \beta_1 t + \cdots + \beta_d t^d, \qquad t = 1, \ldots, N,$$

para ciertos coeficientes $\beta_0, \ldots, \beta_d$ que están completamente determinados por:

- los últimos d valores de la tendencia ajustada en entrenamiento, y

- el valor estimado de la deriva $\widehat{m}$ en diferencias de orden d.

Este es el motivo estructural por el que, en el esquema actual, la tendencia que usamos para pronosticar en validación y prueba es *siempre* un polinomio de grado d: imponemos explícitamente que la $(d+1)$-ésima diferencia sea cero y la d-ésima diferencia sea constante.

2.2. Por qué se consideraron solo $d = 1, 2, 3, 4$

En principio, el modelo permite considerar órdenes de diferencia arbitrarios $d = 0, 1, 2, \ldots$; en la práctica hemos restringido el análisis a $d \in \{1, 2, 3, 4\}$ por razones tanto computacionales como estadísticas.

Interpretación de los órdenes bajos

Los órdenes $d = 1, 2, 3, 4$ tienen una interpretación directa:

- $d = 1$: se penaliza Δt_t, por lo que la extensión impone $\Delta t_t \approx \widehat{m}$ constante. La tendencia extrapolada es aproximadamente *lineal* en el tiempo (nivel con deriva).
- $d = 2$: se penaliza $\Delta^2 t_t$, de modo que la tendencia extrapolada tiene *curvatura* constante (polinomio cuadrático). Modela aceleraciones suaves en el nivel.
- $d = 3$ y $d = 4$: permiten capturar patrones de cambio de curvatura (tercer y cuarto grado), útiles si la tendencia de la serie exhibe segmentos con distinta concavidad/convexidad o inflexiones suaves.

Para datos financieros diarios (logprecios de acciones, índices y criptoactivos) estos cuatro órdenes ya cubren la mayoría de las formas suaves plausibles en horizontes no demasiado largos. Órdenes mucho mayores tienden a producir extrapolaciones muy oscilatorias en los bordes o polinomios demasiado flexibles que no tienen interpretación económica clara.

Coste computacional y estabilidad numérica

Desde el punto de vista computacional:

- La matriz de diferencias K_d se vuelve más ancha y con coeficientes binomiales crecientes al aumentar d, lo que hace que $B_d = K_d^\top K_d$ tenga autovalores más dispersos y el número de condición de $A(\lambda) = I + \lambda B_d$ crezca.

- El cálculo de la descomposición espectral de B_d (un paso clave porque toda la mecánica se basa en $B_d = Q\Lambda Q^\top$) es de complejidad cúbica en N en general. Aunque se reutiliza por orden d, el problema se multiplica por el número de combinaciones (ticker, d, valor de s) que evaluamos.
- Para cada (ticker, d, s) se ejecuta, además, un esquema iterativo de punto fijo para estimar $\widehat{m}$, que se vuelve más inestable cuando d es grande porque la penalización actúa sobre diferencias de orden alto (magnificando ruido numérico y errores en la cola).

En el experimento actual se trabaja con:

- $K = 137$ series,
- $d \in \{1, 2, 3, 4\}$ (4 órdenes),
- una rejilla relativamente fina de s (por ejemplo cientos de puntos por orden),
- y cinco funciones objetivo distintas (MSE en train, val, train+val, val ponderada, train+val ponderada).

Esto ya implica del orden de decenas de miles de evaluaciones de `fit_for_s` por ejecución completa del estudio. Extender a $d = 5, 6, \ldots$ multiplica casi linealmente (por el número de órdenes), pero en la práctica el tiempo crece más que linealmente debido a la mala condición numérica y a que se requieren más iteraciones en el punto fijo para que m converja.

Comportamiento de residuos para órdenes altos

Empíricamente, para órdenes $d > 4$ se observa:

- La tendencia extrapolada se aproxima a polinomios de grado alto que, aunque ajustan muy bien el segmento de entrenamiento, exhiben extrapolaciones poco razonables en validación y prueba (crecimientos explosivos o curvaturas excesivas).
- Los residuos en validación y prueba muestran patrones sistemáticos (no ruido blanco) que indican sobreajuste del segmento de entrenamiento.
- La magnitud de los residuos en las colas (sobre todo en prueba) aumenta, de modo que no hay ganancia clara frente a $d \in \{1, 2, 3, 4\}$.

Por estas razones se decidió, en esta primera etapa, restringirse a $d \in \{1, 2, 3, 4\}$ y dejar la exploración de órdenes mayores como una extensión futura, posiblemente con regularización explícita sobre el grado efectivo de la tendencia (por ejemplo, usando criterios tipo AIC/BIC entre órdenes, o combinaciones convexas de penalizaciones de diferentes órdenes).

2.3. Cómo obtener tendencias pronosticadas no polinómicas

El carácter polinómico de la tendencia extrapolada proviene exclusivamente de la condición fuerte (2) (o equivalentemente (3)). Si deseamos que la tendencia fuera de muestra no sea necesariamente un polinomio de grado d, sino una serie suavizada más general, hay varias extensiones posibles dentro del mismo marco penalizado.

Suavizamiento sobre un horizonte extendido

Partimos del funcional penalizado, pero ahora sobre un horizonte extendido de longitud $N + H$, donde los últimos H puntos corresponden a horizontes de pronóstico:

$$J(\boldsymbol{t}) := \sum_{t=1}^{N} (Z_t - t_t)^2 \; + \; \lambda \sum_{r=1}^{N+H-d} \left(\Delta^d t_r\right)^2, \tag{4}$$

donde $\boldsymbol{t} = (t_1, \ldots, t_{N+H})^\top$ y para $t > N$ no hay observaciones (no aparece Z_t en el primer sumando).

Observaciones:

- En (4) *no* imponemos que $\Delta^d t_r$ sea constante ni que $\Delta^{d+1} t_r = 0$; tan solo penalizamos su magnitud cuadrática.
- El segundo término define un operador de suavizamiento global sobre toda la malla de tiempo $1, \ldots, N + H$.

En forma matricial, sea K_{ext} la matriz de diferencias de orden d sobre el horizonte extendido $(N + H)$, y W una matriz diagonal de pesos de observación:

$$W := \operatorname{diag}(w_1, \ldots, w_{N+H}), \qquad w_t := \begin{cases} 1, & t \le N, \\ 0, & t > N. \end{cases}$$

Entonces

$$J(\boldsymbol{t}) = (\boldsymbol{Z}_{\text{ext}} - \boldsymbol{t})^\top W (\boldsymbol{Z}_{\text{ext}} - \boldsymbol{t}) + \lambda \|K_{\text{ext}} \boldsymbol{t}\|_2^2,$$

donde $\boldsymbol{Z}_{\text{ext}} = (Z_1, \ldots, Z_N, 0, \ldots, 0)^\top \in \mathbb{R}^{N+H}$. El minimizador viene dado por

$$\widehat{\boldsymbol{t}} = \left(W + \lambda K_{\text{ext}}^\top K_{\text{ext}}\right)^{-1} W \boldsymbol{Z}_{\text{ext}}.$$

Las componentes $\widehat{t}_{N+1}, \ldots, \widehat{t}_{N+H}$ son entonces *pronósticos suavizados* que:

- siguen siendo suaves en el sentido de penalizar diferencias de orden d, pero
- ya no satisfacen $\Delta^{d+1} t_t = 0$, por lo que *no* son polinomios de grado d; en general, tienen la forma de una extensión tipo spline discreta con curvatura controlada.

Desde el punto de vista de implementación, esto requiere:

- construir K_{ext} para el horizonte extendido,
- resolver un sistema lineal de tamaño $(N + H) \times (N + H)$ (o usar la descomposición espectral de $K_{\text{ext}}^\top K_{\text{ext}}$),
- trabajar con pesos w_t que apaguen la contribución de los puntos futuros (sin observación) en el primer sumando.

Esta formulación entrega una tendencia pronosticada suavizada sin estructura polinómica rígida, aunque sigue controlada por el orden d de las diferencias en el término de penalización.

Relajar la constancia de $\Delta^d t_t$

Otra variante es mantener el horizonte original $1, \ldots, N$ pero cambiar la forma de la penalización. En lugar de penalizar $(\Delta^d t_t - m)^2$ con un m constante, podemos considerar un m_t que varía lentamente:

$$J(\boldsymbol{t}, \boldsymbol{m}) := \sum_{t=1}^{N} (Z_t - t_t)^2 + \lambda_1 \sum_{r=1}^{N-d} (\Delta^d t_r - m_r)^2 + \lambda_2 \sum_{r=2}^{N-d} (m_r - m_{r-1})^2.$$

Aquí:

- $\boldsymbol{m} = (m_1, \ldots, m_{N-d})^\top$ es una deriva local de $\Delta^d t_t$,
- λ_1 controla qué tanto $\Delta^d t_r$ sigue a m_r,

- λ_2 controla la suavidad de $\boldsymbol{m}$ en el tiempo.

El modelo anterior induce que $\Delta^d t_t$ sea suave pero no necesariamente constante, por lo que, de nuevo, la solución $\widehat{\boldsymbol{t}}$ no es un polinomio de grado d, sino una curva más flexible que puede cambiar gradualmente su aceleración de orden d.

De forma análoga, el pronóstico se obtiene extrapolando el par $(\boldsymbol{t}, \boldsymbol{m})$ bajo una dinámica suave (por ejemplo, asumiendo que los últimos pasos de $\boldsymbol{m}$ continúan). Esto ya es un modelo de mayor dimensión que el caso escalar m de Guerrero, pero permanece dentro del marco de mínimos cuadrados penalizados.

Conexión con suavizadores clásicos

Si eliminamos por completo el parámetro m y trabajamos con la penalización

$$J(\boldsymbol{t}) = \sum_{t=1}^{N} (Z_t - t_t)^2 + \lambda \sum_{r=1}^{N-d} (\Delta^d t_r)^2,$$

obtenemos exactamente el suavizador tipo spline discreto que subyace a muchos *P-splines* y modelos de tendencia local. La parte de ajuste en muestra (la trayectoria $\widehat{t}_1, \ldots, \widehat{t}_N$) es idéntica a la que ya usamos en el interior de la muestra; la diferencia está en cómo definimos el comportamiento fuera de la muestra:

- Si imponemos $\Delta^d t_t$ constante fuera de la muestra, volvemos al caso polinómico.
- Si definimos un horizonte extendido con penalización pura (como en (4)), la tendencia de pronóstico es una extensión suavizada, no polinómica.

En resumen:

La tendencia extrapolada es polinómica de grado d porque imponemos $\Delta^{d+1} t_t = 0$. Si relajamos esa c