BNP PARIBAS
CORPORATE & INSTITUTIONAL BANKING

Transformer-based Architectures for Time Series Foundation Models

Literature Synthesis

MARZOUG AYOUB

Mai 2025

Context & Motivation

Foundation Models

- Self-supervised learning on massive, diverse data.
- Strong zero-shot / few-shot generalization.
- Success in NLP & Vision $\rightarrow$ Can we transpose this to time series ?

Why Time Series ?

- Financial, industrial, and environmental applications.
- Unstructured, irregular, noisy signals.
- Temporal dynamics vary by domain.

Research Challenge

Internship Objective

- Identify an optimal **Transformer-based architecture**.
- Build a **foundation model for financial time series**.
- Capture key *stylized facts* : mean-reversion, volatility clustering, etc.

Core Challenges

- **Tokenization** of irregular, multiscale signals.
- Encoding statistical models : OU, Heston, GARCH.
- Balancing generalization with domain specificity.

Goal : Ground Transformer design in financial model priors.

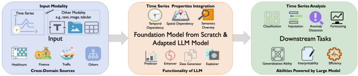
What Are Time Series Foundation Models ?

Foundation Models are :

- Pretrained on large, diverse time series corpora.
- Used across domains/tasks with minimal adaptation.

Goal

Transfer knowledge across time series domains.



Difference from traditional models :

→ Train once, adapt everywhere.

→ Zero-shot & few-shot forecasting.

Desirable Properties of TSFM

Generalization
Across datasets, tasks

Scalability
Large model &
data regimes

Temporal Fidelity
Preserve fre-
quency/dynamics

Light Adaptation
Few-shot or
prompt-tuned

Time series FMs must balance **generalization**, **fidelity**, *and* **lightweight adaptation**.

Overview of Time Series Foundation Models

Model	Arch. Type	Tokenization	Probabilistic ?	Pretraining
Lag-Llama	Decoder-only	Lag features	✓	27 real datasets
TimesFM	Decoder-only	Patching	× (point)	Synthetic + real
Chronos	Decoder-only	Quantization	✓	Quantized tokens
ForecastPFN	Encoder-only	Synthetic signals	× (point)	100% synthetic

A spectrum of design choices : **token type**, **architecture**, **forecast type**, *and* **training corpus***.*

Tokenization Strategies : A Comparative Lens

Chronos

- Quantizes real values $\rightarrow$ fixed vocabulary.
- Compatible with NLP-style LLMs.

Lag-Llama

- Lagged features + time covariates.
- Rich temporal encoding, causal-friendly.

TimesFM

- Patch-based tokenization.
- Transformer-efficient, local temporal context.

ForecastPFN

- No true tokens — only synthetic inputs.
- Learned inductive bias over synthetic families.

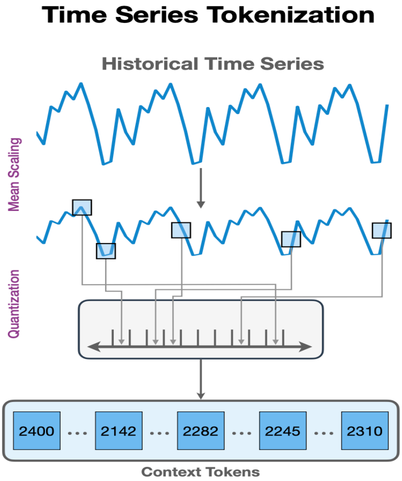
Tokenization defines **inductive bias**, **transferability**, *and* **model-task compatibility**.

In NLP :

- Words → Tokens → Meaning.
- Fixed vocabulary, grammar.
- Transformers excel via token attention.

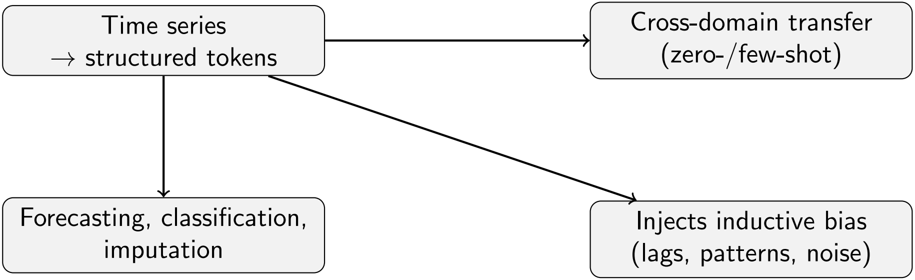
In Time Series :

- No fixed vocabulary.
- Continuous, irregular, noisy data.
- Tokenization = Encoding inductive bias.



Good tokenization is the foundation of transferability and generalization in TS models

What Tokenization Enables



Tokenization is not just preprocessing — it defines how the model learns and transfers.

Lag-Llama : Tokenization Strategy

Core Idea

Lag-Llama tokenizes time series by constructing **lag-based feature vectors** for each time step, rather than patches or quantized bins.

Definition : Lag Tokenization

Given a sorted list of lag indices $L = l_1, l_2, \ldots, l_n$, the token k_t for time t is :

$$k_t[j] = x_{t-l_j}, \quad \text{for } j = 1, \ldots, n$$

Each $k_t \in \mathbb{R}^{|L|}$ is augmented with :

- **Date-time covariates :** second, minute, hour, day, etc.
- **Summary stats :** mean (μ) and scale (σ) from robust scaling (IQR).

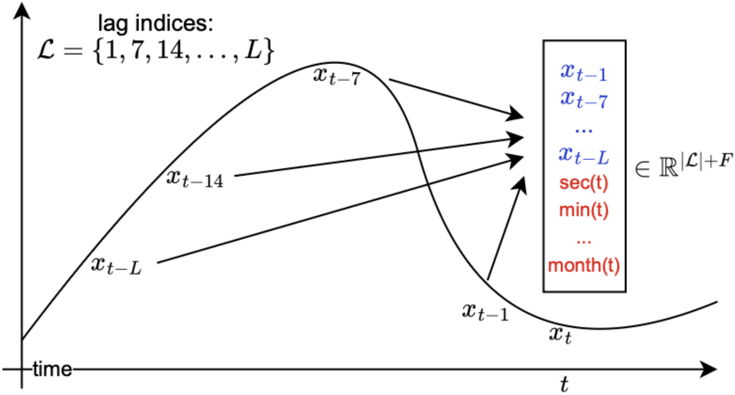


Illustration of Lag-Llama token construction : lags + datetime + summary features.

Tokenization Purpose & Design Motivation

- **Frequency-Invariance :** Lag indices span multiple time granularities (hourly, daily, monthly).
- **Causal Structure :** Only past values used for token $k_t \Rightarrow$ respects autoregressive setup.
- **Semantic Encoding :** Time features help align data across domains (*e.g.* finance, weather).

Practical Considerations

- Needs L historical points to construct each token k_t.
- Suitable for decoder-only autoregressive forecasting.

Robust Normalization Strategy

Challenge

Diverse magnitude ranges across datasets.

Solution : Robust IQR-based scaling

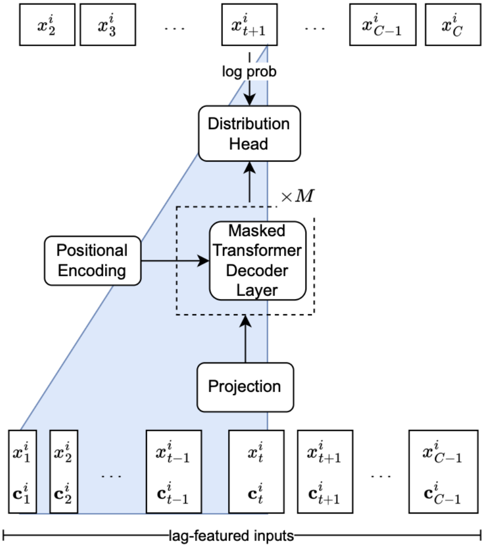
$$x'_t = \frac{x_t - \mathrm{Med}(x1 : C)}{\mathrm{IQR}(x_{1:C})}$$

$$\mathrm{IQR}(x_{1:C}) = \mathrm{Med}(x_{\lceil C/2 \rceil : C}) - \mathrm{Med}(x_{1:\lfloor C/2 \rfloor})$$

- Ensures scale-invariance.
- Median-based $\Rightarrow$ robust to outliers.

Lag-Llama includes mean and scale as features in token embedding.

Lag-Llama Full Architecture



Lag-Llama Forecasting Pipeline

Inference :

- Predicts $\phi =$ (mean, scale, df) of **Student's t-distribution** per timestep.
- Uses autoregressive decoding with greedy sampling to simulate future paths.

TimesFM : Tokenization Strategy (Patching)

Core Idea

TimesFM transforms time series into **non-overlapping fixed-length patches**. Each patch acts as a token in the Transformer input.

Token Construction

Given a time series $y\_1, \ldots, y\_L$ and patch size p :

$$\tilde{y}_j = y_{p(j-1)+1:pj} \in \mathbb{R}^p, \quad \text{for } j = 1, \ldots, \lfloor L/p \rfloor$$

Each patch is :

- Processed by a Residual MLP Block $R(\cdot)$
- Embedded to model dimension d : $t_j = R(\tilde{y}_j) + PE_j$.

Motivation Behind Patching

- **Efficiency :** Reduces sequence length for Transformer $\Rightarrow$ faster training.
- **Generalization :** Compatible with variable-length contexts and prediction horizons.
- **Modularity :** Patches serve as atomic temporal units, facilitating domain-agnostic modeling.

Training-Time Features

- Random masking of full and partial patches.
- Ensures robustness to different context lengths.

Prediction Tokens : Long Output Patches

Problem

Auto-regressive forecasting is inefficient for long horizons.

Solution : Predict longer output patches.

Let h be the output patch length :

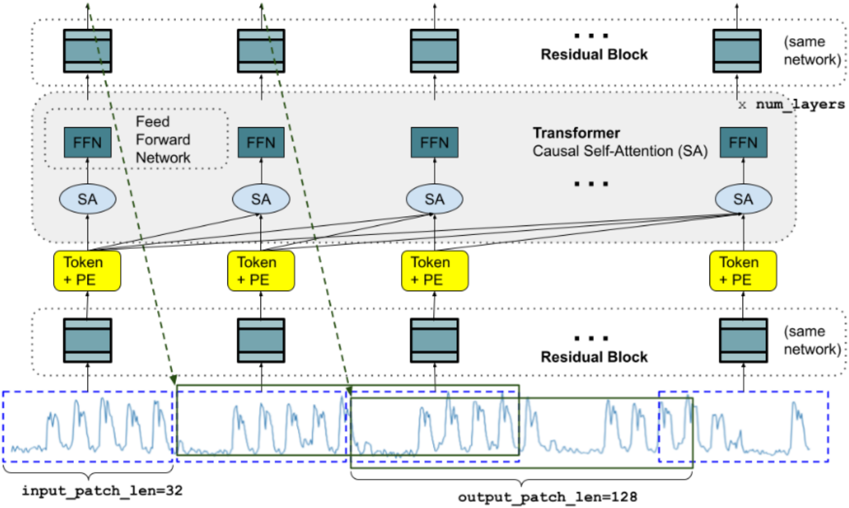
$$\hat{y}_{pj+1:pj+h} = \mathrm{OutputResidualBlock}(o_j)$$

- Each output token predicts h future time steps.
- Requires fewer decoding steps $\Rightarrow$ speed $+$ better accuracy.

Trade-off

Cannot use very large h for short input sequences.

Architecture Summary



Decoder-only architecture with patch tokenization, causal attention, and MLP heads

Chronos : Tokenization Strategy

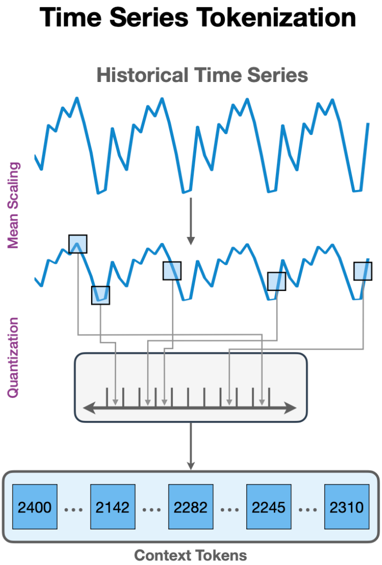
Core Idea

Chronos converts real-valued time series into sequences of **discrete tokens** via **mean scaling** and **uniform quantization**. This enables use of off-the-shelf language models.

Token Construction

- Given $x_1, \ldots, x_{C+H}$: context (1 to C), forecast (next H).
- **Mean Scaling :** $\tilde{x}i = \frac{x_i}{\frac{1}{C}\sum j = 1^C |x_j|}$
- **Quantization :** Uniform binning over $[-15, +15]$ into B bins (e.g., $B = 4096$)
- **Token :** $z_i = q(\tilde{x}_i) \in 1, ..., B$

Chronos Tokenization



Why Discrete Tokenization ?

Motivation :

- Language models require finite vocabulary.
- Continuous values $\rightarrow$ categorical tokens.
- Enables use of **cross-entropy loss** over token predictions.

Quantization Function :

$$q(x) = \begin{cases} 1 & \text{if } x < b_1 \; 2 \\ \text{if } b_1 \leq x < b_2 \; \vdots \; B & \text{if } x \geq b_{B-1} \end{cases}, \quad d(j) = c_j$$

No Positional Features :

Unlike other models, Chronos **ignores time/frequency features** $\Rightarrow$ sequence-only input.

Forecasting as Language Modeling

Model :

- Uses pretrained T5 (encoder-decoder) or GPT2 (decoder-only).
- Trained on tokenized sequences : $z_1, \ldots, z_{C+H}$.

Objective :

$$\ell(\theta) = -\sum_{h=1}^{H+1} \sum_{i=1}^{|V_{ts}|} \mathbf{1}(z_{C+h+1} = i) \log p_\theta(z_{C+h+1} = i | z_{1:C+h})$$

- Categorical prediction over $|V_{ts}|$ bins.
- Sequence decoding with **sampling** $\rightarrow$ multiple futures.

Result

Probabilistic forecasting with multimodal support.

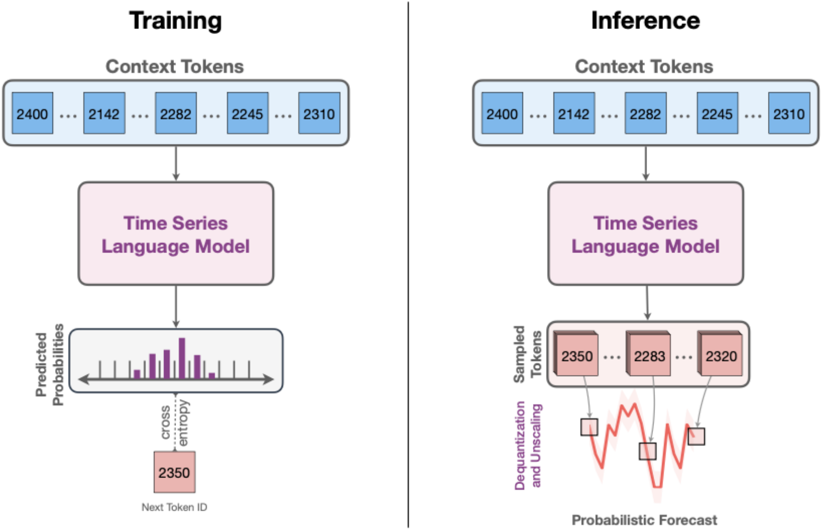
Architectural Simplicity & Flexibility

- No need to change LLM architecture (just adjust vocab size).
- Train using standard cross-entropy + language modeling libraries.
- Forecasts are obtained via token sampling $\rightarrow$ dequantized + unscaled.

Advantages :

- Scalable across datasets and domains.
- Multimodal + probabilistic predictions.
- General-purpose **tokenization interface for time series**.

Chrons Training & Inference



ForecastPFN : Tokenization by Attribute Encoding

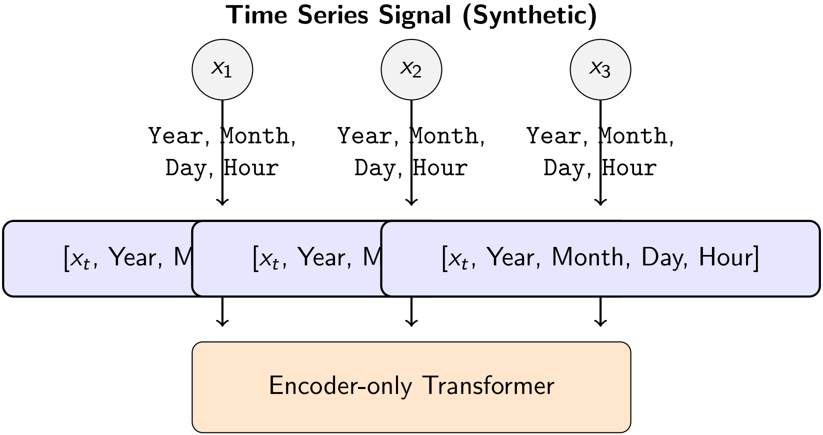
Core Idea

ForecastPFN does not rely on real-world time series tokenization. Instead, it uses **synthetic signals** and encodes them through time-indexed attribute vectors.

Token Construction

- Each token = value x_t + temporal metadata.
- Metadata includes : `year`, `month`, `day`, `weekday`, `hour`, `minute`, `second`.

ForecastPFN Tokenization



Synthetic Training Paradigm

Why synthetic ?

- Real-world datasets are domain-limited, sparse, and noisy.
- Goal : encode *families of dynamics* (seasonal, autoregressive, trend).

Signal Families :

- Seasonal signals : $x_t = A \sin(\omega t + \phi)$
- Trend : $x_t = At + B$
- Random walk : $x_t = x_{t-1} + \epsilon_t$, $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$

Tokens encode both value and context $\rightarrow$ general-purpose forecaster.

Architecture and Forecasting Pipeline

Architecture Highlights :

- Encoder-only Transformer.
- Uses dense token embeddings.
- Fully trained on synthetic corpora.

Forecasting :

- Given past tokens (value + timestamp), predict future values.
- No quantization, no patches : token = scalar + metadata.

Tokenization : Comparative Summary

Model	**Tokens**	**Time**	**Input**	**Vocab**	**Arch**
Lag-Llama	Lag + Time	Yes	$\mathbb{R}^n$	Real	Dec
TimesFM	Patches	Yes	$\mathbb{R}^p$	Real	Dec
Chronos	Quantized	No	$\mathbb{Z}_B$	Disc.	Dec
ForecastPFN	Scalar + Meta	Yes	(x_t, t)	Real	Enc

Token design shapes compatibility, temporal alignment, and inductive bias.

Tokenization Trade-offs

Real vs Discrete Tokens

- **Lag-Llama, TimesFM, ForecastPFN** keep raw values → higher fidelity.
- **Chronos** trades precision for compatibility with pretrained LLMs.

Time-awareness

- **Chronos :** no temporal context.
- **Lag-Llama :** explicit lags.
- **TimesFM :** implicit order via patching.
- **ForecastPFN :** full timestamp embedding.

Tokenization is the root of model bias and generalization behavior.

Architectural Styles Across Models

Decoder-only Models

- Used by : **Lag-Llama**, **TimesFM**, **Chronos**.
- Predict tokens auto-regressively : each prediction depends on all prior inputs.
- Supports flexible context windows and causal attention.

Encoder-only Model

- Used by : **ForecastPFN**.
- Whole context is encoded at once → no autoregression.
- Fully parallel inference → fast, suitable for short contexts.

Decoder-only = autoregressive prediction loop ; Encoder-only = full input encoding at once.

Distribution Heads : Point vs Probabilistic Forecasting

Point Forecasts

- **TimesFM, ForecastPFN**.
- Output : predicted value $\hat{y}_t$ directly.
- Loss : MSE, MAE.

Probabilistic Forecasts

- **Lag-Llama** : Student-t output $\rightarrow (\mu, \sigma, \nu)$.
- **Chronos** : discrete token sampling $+$ dequantization.
- Output : distribution over values $\rightarrow$ supports uncertainty.
- Loss : Negative log-likelihood, CRPS, token cross-entropy.

Probabilistic heads model uncertainty — crucial for financial volatility modeling.

Pretraining Corpora and Scaling

Real vs Synthetic Pretraining

- **Lag-Llama** : 27 real-world datasets (weather, traffic, finance, etc.).
- **TimesFM** : Google Trends, Wikipedia traffic, synthetic.
- **Chronos** : Public + synthetic (quantized tokens).
- **ForecastPFN** : Fully synthetic (AR, seasonality, trends).

Scaling Factors

- Number of time series (millions vs thousands).
- Length of context windows.
- Number of model parameters (Chronos : 124M, 770M, 1.5B).

Diverse, large-scale corpora enhance robustness across domains.

Normalization and Temporal Handling

Normalization Strategies

- **Lag-Llama** : Robust IQR-based scaling.
- **Chronos** : Mean scaling $x'_t = \frac{x_t}{\frac{1}{C}\sum_{i=1}^{C}|x_i|}$.
- **TimesFM, ForecastPFN** : Implicit normalization inside residual/embedding layers.

Handling Temporal Structure

- **Lag-Llama, ForecastPFN** : Use of explicit time features.
- **TimesFM** : Implicitly ordered patches.
- **Chronos** : No explicit time index (token-only).

Normalization + temporal structure = key for cross-domain transfer.

Zero-shot and Few-shot Performance

Zero-shot Capable Models

- **Lag-Llama** : Strong zero-shot forecasting across diverse domains.
- **Chronos** : Learns discrete priors ; surprisingly robust zero-shot performance.
- **TimesFM** : Competitive zero-shot for long-term trends.

Few-shot Adaptation

- **Lag-Llama, Chronos** : Improved with minimal fine-tuning (e.g., 20% target data).
- **TimesFM** : Resilient to variable horizon prediction.
- **ForecastPFN** : Poor zero-shot on real data (trained only on synthetic).

Pretraining design and tokenization shape few-shot success.

Finetuning and Transfer Efficiency

Finetuning Efficiency

- **Lag-Llama** : Benefits most from task-specific finetuning.
- **Chronos** : Quick adaptation due to discrete token structure.
- **TimesFM** : Finetuning useful for extreme long-horizon tasks.
- **ForecastPFN** : No real-world finetuning path (synthetic-only training).

Transfer Challenges

- **Chronos** : Quantization limits resolution.
- **ForecastPFN** : Domain shift from synthetic $\rightarrow$ real.

Finetuning ability matters most in financial settings with scarce labels.

Towards Wavelet-Based Tokenization for Time Series

Why Wavelets ? [Zhu & Soricut, 2024]

- In vision : wavelets replace patch tokenizers → lower token count, better throughput, robustness.
- Key benefits :
  - Efficient compression of redundant local structure.
  - Resistance to noise/adversarial perturbations.
  - Sparse token embeddings with semantic meaning.

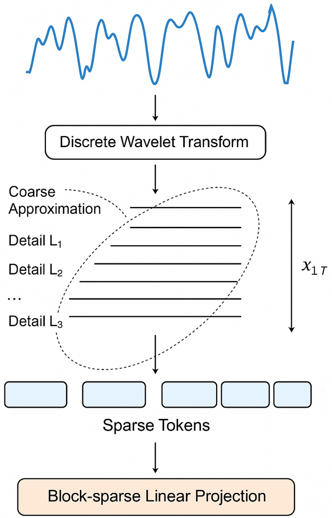
Relevance to Time Series

- TS also contains multiscale, localized patterns (*e.g.* volatility spikes, seasonal trends).
- Wavelet transforms already used in TS denoising and frequency decomposition.
- Could provide a frequency-aware, compressed, and sparse tokenization method.

Wavelet Tokenization Pipeline for TS

- Apply discrete wavelet transform (DWT) to input sequence $x_{1:T}$.
- Keep coarse approximation + a selection of detail coefficients.
- Group coefficients over time to form sparse tokens.
- Project to model dimension via block-sparse linear layers (as in paper).

Next steps : Evaluate sparsity, generalization, compatibility with decoder-only autoregression



Limitations & Open Research Directions

Current Limitations

- **Tokenization bottlenecks :** No standard method across domains ; quantization (Chronos) loses precision, patches (TimesFM) lose fine temporal alignment.
- **Pretraining limitations :** Synthetic-only training (ForecastPFN) underperforms on real-world signals.
- **Lack of interpretability :** Attention weights and latent representations are opaque ; difficult to relate to financial model parameters.

Open Research Directions

- **Wavelet-based tokenization :** Leverage sparsity and multi-resolution encoding for better temporal structure capture.
- **Hybrid tokenization** : Combine frequency (wavelets) and time-based features (lags, patches) for richer representations.
- **Financial model grounding :** Align internal representations with OU, GARCH, Heston dynamics.
- **Multi-resolution attention :** Scale attention with temporal resolution → low-frequency → high-resolution refinement.

From token design to inductive bias : foundation TS models still lack domain alignment.

Concluding Synthesis

What We've Seen

- Transformer-based TS models differ primarily in **tokenization strategies**.
- **Lag-Llama** : lag-based, interpretable ; **TimesFM** : patch-based, scalable ; **Chronos** : quantized, LLM-compatible ; **ForecastPFN** : metadata-driven, synthetic.
- Architecture choices (decoder vs encoder) influence generalization and adaptation.
- Probabilistic forecasting is central for financial applications.

Next Steps

- Investigate wavelet-based tokenization for multiscale temporal structure.
- Begin design of a transformer architecture that can **recover classical financial models**.
- Move toward building a candidate *foundation model architecture* for financial TS.

From comparative study to architectural design — the foundation begins with tokens.

Appendix & References I

K. Rasul, A. Ashok, A. R. Williams, et al.
Lag-Llama : Towards Foundation Models for Probabilistic Time Series Forecasting.
arXiv :2310.08278, 2024.

A. Das, W. Kong, R. Sen, Y. Zhou.
TimesFM : A Decoder-Only Foundation Model for Time-Series Forecasting.
arXiv :2310.10688, 2024.

A. F. Ansari, L. Stella, C. Turkmen, et al.
Chronos : Learning the Language of Time Series.
Transactions on Machine Learning Research, 2024. arXiv :2403.07815.

J. Dooley, A. W. Senior, A. G. de G. Matthews.
ForecastPFN : Towards Foundation Models for Time-Series Forecasting.
arXiv :2306.09333, 2023.

Z. Zhu and R. Soricut.
Wavelet-Based Image Tokenizer for Vision Transformers.
arXiv :2405.18616, 2024.

All models and analysis in this presentation are derived from the above foundational sources.