



Analiza și Prognoza seriilor de timp

Capitolul 9: Prophet și TBATS



Daniel Traian PELE

Academia de Studii Economice din București

IDA Institute Digital Assets

Blockchain Research Center

AI4EFin Artificial Intelligence for Energy Finance

Academia Română, Institutul de Prognoză Economică

MSCA Digital Finance

Cuprins

Fundamente

- ▣ Sezonaliități Multiple
- ▣ Modelul TBATS
- ▣ Facebook Prophet

Aplicații

- ▣ Comparatie și Ghid de Selecție
- ▣ Studiu de Caz
- ▣ Rezumat și Quiz

Problema: tipare sezoniere complexe

Exemple din Lumea Reală

- ▣ **Cerere de electricitate pe oră:** Tipare zilnice + săptămânale + anuale
- ▣ **Trafic web:** Zilnic + săptămânal + efecte de sărbători
- ▣ **Vânzări retail:** Săptămânal + lunar + anual + sărbători
- ▣ **Volum call center:** Pe oră + zilnic + săptămânal

Limitarea SARIMA

$SARIMA(p, d, q)(P, D, Q)_s$ standard gestionează doar **o singură** perioadă sezonieră s .

Pentru date orare cu tipare zilnice și săptămânale, avem nevoie de $s_1 = 24$ și $s_2 = 168$.

Soluții pentru sezonalități multiple

Abordări Tradiționale

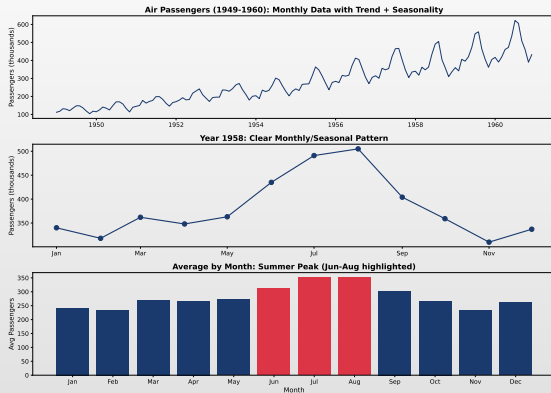
- **Termeni Fourier:** Adăugare regresori sin/cos
- **Variabile dummy:** Mulți parametri
- **Modele nested:** Specificare complexă

Abordări Moderne

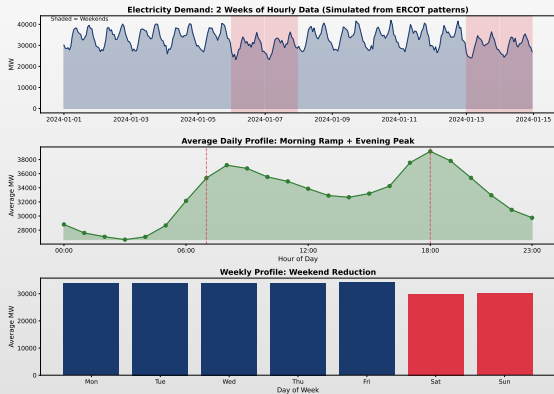
- **TBATS:** Automat, gestionează multe perioade
- **Prophet:** Flexibil, interpretabil
- **Metode neurale:** Deep learning

Metodă	Nr. Max Sezonalități	Interpretabil
SARIMA	1	Da
Fourier + ARIMA	Multiple	Moderat
TBATS	Multiple	Moderat
Prophet	Multiple	Da

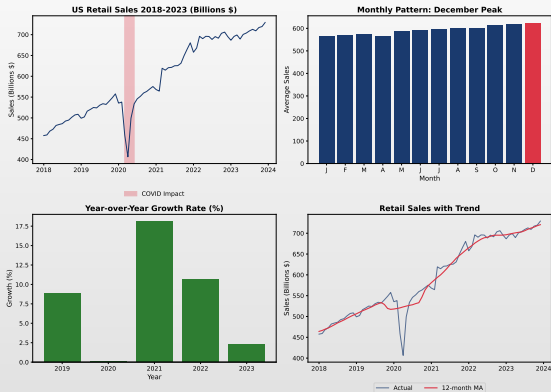
Exemplu: date orare cu sezonalități multiple



Exemplu real: cerere de electricitate



Exemplu real: vânzări retail cu sărbători



TBATS: ce înseamnă?

Componentele TBATS

- T Sezonalitate **Trigonometrică** folosind termeni Fourier
- B Transformare **Box-Cox** pentru stabilizarea varianței
- A Erori **ARMA** pentru autocorelația reziduală
- T Componentă de **Trend** (posibil amortizat)
- S Componente **Sezoniere** (multiple permise)

Inovația Cheie

TBATS folosește **reprezentare trigonometrică** pentru sezonalitate:

$$s_t^{(i)} = \sum_{j=1}^{k_i} \left[s_j^{(i)} \cos\left(\frac{2\pi jt}{m_i}\right) + s_j^{*(i)} \sin\left(\frac{2\pi jt}{m_i}\right) \right]$$

unde m_i este perioadă sezonieră i și k_i este numărul de armonici.

Structura modelului TBATS

Specificația Completă a modelului

$$y_t^{(\omega)} = \ell_{t-1} + \phi b_{t-1} + \sum_{i=1}^M s_t^{(i)} + d_t \quad (1)$$

$$\ell_t = \ell_{t-1} + \phi b_{t-1} + \alpha d_t \quad (2)$$

$$b_t = \phi b_{t-1} + \beta d_t \quad (3)$$

$$d_t = \sum_{i=1}^p \varphi_i d_{t-i} + \sum_{j=1}^q \theta_j \varepsilon_{t-j} + \varepsilon_t \quad (4)$$

Unde:

- ▣ $y_t^{(\omega)}$ este seria transformată Box-Cox (dacă $\omega \neq 1$)
- ▣ ℓ_t este nivelul local, b_t este trendul cu amortizare ϕ
- ▣ $s_t^{(i)}$ sunt M componente sezoniere cu perioade m_1, \dots, m_M
- ▣ d_t este procesul de eroare ARMA(p, q)

TBATS: sezonaliitate trigonometrică

De ce Termeni Fourier/Trigonometrici?

1. **Simplu:** Mai puțini parametri decât variabilele dummy
2. **Neted:** Captează natural tiparele sezoniere netede
3. **Flexibil:** Numărul de armonici k controlează complexitatea
4. **Perioade non-întregi:** Poate gestiona $s = 365.25$ pentru date zilnice

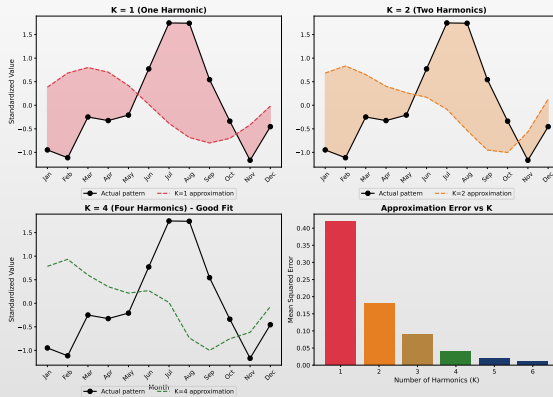
k mic (puține armonici)

- Tipar neted
- Mai puțini parametri
- Poate rata vârfuri abrupte

k mare (multe armonici)

- Poate capta orice tipar
- Mai mulți parametri
- Risc de supraajustare

Aproximarea Fourier a sezonaliității



TBATS în practică

implementare Python

Pachetul `tbats` oferă selecție automată a modelului:

- ▣ Selectează automat parametrul Box-Cox ω
- ▣ Alege numărul de armonici k_i pentru fiecare perioadă sezonieră
- ▣ Selectează ordinele ARMA (p, q)
- ▣ Testează trend amortizat vs neamortizat

Exemplu de Cod

```
from tbats import TBATS
estimator = TBATS(seasonal_periods=[7, 365.25])
model = estimator.fit(y)
forecast = model.forecast(steps=30)
```

Notă: BATS este versiunea mai simplă fără termeni trigonometrici (folosește stări sezoniere tradiționale).

TBATS: avantaje și limitări

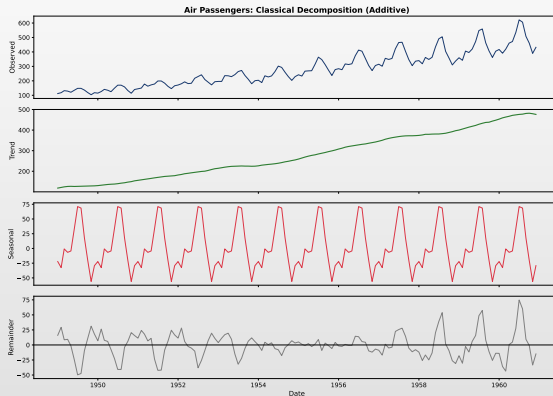
Avantaje

- ▣ Gestionează **multiple** perioade sezoniere
- ▣ Selecție **automată** a modelului
- ▣ Gestionează perioade **non-întregi** (365.25)
- ▣ **Box-Cox** pentru heteroscedasticitate
- ▣ Bun pentru date de **frecvență înaltă**

Limitări

- ▣ **Intensiv computațional**
- ▣ Fără **regresori externi**
- ▣ Mai puțin **interpretabil** decât Prophet
- ▣ Poate fi **lent** pentru serii foarte lungi
- ▣ Necesită **suficiente date** per sezon

Exemplu descompunere TBATS



Prophet: prezentare generală

Ce este Prophet?

Prophet este o procedură de prognoză dezvoltată de Facebook (Meta) în 2017.

Proiectat pentru **serii de timp de business** cu:

- ▣ Efecte sezoniere puternice (zilnice, săptămânale, anuale)
- ▣ Efecte de sărbători
- ▣ Schimbări de trend (changepoints)
- ▣ Date lipsă și outlieri

Filosofia Cheie

“Analyst-in-the-loop” forecasting

Prophet este proiectat pentru a fi ajustat de analiști cu cunoștințe de domeniu, dar care nu sunt neapărat experți în serii de timp.

Structura modelului Prophet

Abordare prin Descompunere

Prophet folosește o **descompunere aditivă**:

$$y(t) = g(t) + s(t) + h(t) + \varepsilon_t$$

$g(t)$: Trend

- ▣ Liniar sau logistic
- ▣ Changepoints automate
- ▣ Saturație de creștere

$s(t)$: Sezonalitate

- ▣ Serii Fourier
- ▣ Perioade multiple
- ▣ Sezonalitate custom

$h(t)$: Sărbători

- ▣ Sărbători pe țară
- ▣ Evenimente custom
- ▣ Efecte de fereastră

Prophet: componenta de trend

trend liniar cu Changepoints

$$g(t) = (k + \mathbf{a}(t)^T \boldsymbol{\delta}) \cdot t + (m + \mathbf{a}(t)^T \boldsymbol{\gamma})$$

unde:

- ▣ k este rata de creștere de bază
- ▣ $\boldsymbol{\delta}$ este un vector de ajustări de rată la changepoints
- ▣ $\mathbf{a}(t)$ indică ce changepoints sunt active la momentul t
- ▣ m este offset-ul, $\boldsymbol{\gamma}$ asigură continuitatea

Creștere Logistică (pentru trenduri cu saturație)

$$g(t) = \frac{C(t)}{1 + \exp(-(k + \mathbf{a}(t)^T \boldsymbol{\delta})(t - (m + \mathbf{a}(t)^T \boldsymbol{\gamma})))}$$

unde $C(t)$ este capacitatea maximă (posibil variabilă în timp).

Prophet: componenta de sezonaliitate

Reprezentare prin Serii Fourier

Pentru o perioadă sezonieră P , Prophet folosește:

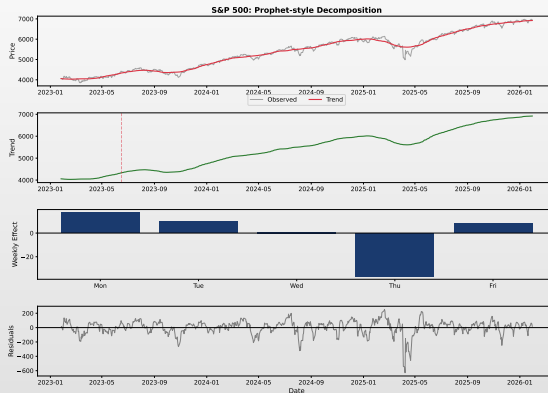
$$s(t) = \sum_{n=1}^N \left[a_n \cos\left(\frac{2\pi nt}{P}\right) + b_n \sin\left(\frac{2\pi nt}{P}\right) \right]$$

Setări Implicite

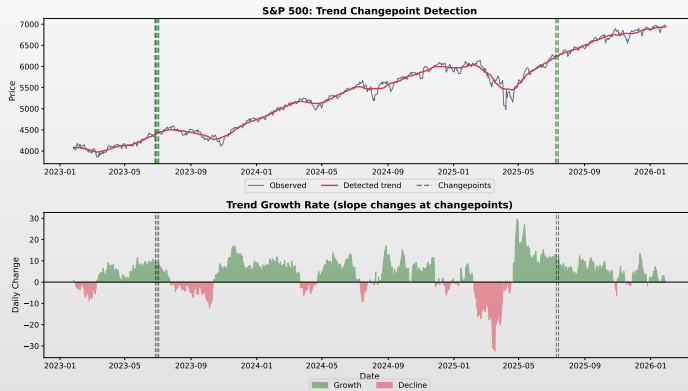
Sezonaliitate	Perioadă	Ordin Fourier
Anuală	365.25 zile	10
Săptămânală	7 zile	3
Zilnică	1 zi	4

Ordin Fourier N mai mare = mai multă flexibilitate (poate ajusta tipare mai complexe) dar risc mai mare de supraajustare.

Descompunerea componentelor Prophet



Detectarea changepoints în trend



Prophet: efecte de sărbători

Modelul de Sărbători

$$h(t) = Z(t) \cdot \kappa$$

unde $Z(t)$ este o matrice indicator pentru sărbători și κ sunt efectele sărbătorilor.

Caracteristici

- ▣ **Sărbători integrate:** 60+ țări suportate
- ▣ **Sărbători custom:** Adăugați propriile evenimente (Black Friday, evenimente companie)
- ▣ **Efecte de fereastră:** Sărbătorile pot afecta zilele înainte/după
- ▣ **Prior scale:** Controlează regularizarea efectelor de sărbătoare

Exemplu de Cod

```
holidays = pd.DataFrame({'holiday': 'black_friday', ...})  
model = Prophet(holidays=holidays)
```

Prophet în practică

Utilizare de Bază

```
from prophet import Prophet
import pandas as pd

# Datele trebuie să aibă coloane 'ds' (dată) și 'y' (valoare)
df = pd.DataFrame({'ds': dates, 'y': values})

model = Prophet()
model.fit(df)

future = model.make_future_dataframe(periods=365)
forecast = model.predict(future)
```

Adăugare Sezonality Custom

```
model = Prophet(weekly_seasonality=False)
model.add_seasonality(name='monthly', period=30.5, fourier_order=5)
model.add_seasonality(name='quarterly', period=91.25, fourier_order=3)
```

Prophet: cuantificarea incertitudinii

Trei Surse de Incertitudine

1. **Incertitudine de trend:** Changepoints viitoare sunt incerte
2. **Incertitudine de sezonaliitate:** Incertitudine în estimarea parametrilor
3. **Zgomot de observație:** Aleatorietate inerentă

Intervale de Predicție

Prophet oferă:

- Prognoză punctuală: `yhat`
- Limita inferioară: `yhat_lower`
- Limita superioară: `yhat_upper`

Implicit este interval de 80%.

Schimbați cu `interval_width=0.95`

Notă

Incetitudinea crește cu orizontul de prognoză, în special pentru incertitudinea de trend.

Prophet: parametri de ajustare

Parametri cheie

Parametru	Efect
<code>changepoint_prior_scale</code>	Flexibilitate trend (implicit: 0.05)
<code>seasonality_prior_scale</code>	Flexibilitate sezonality (implicit: 10)
<code>holidays_prior_scale</code>	Mărime efect sărbători (implicit: 10)
<code>seasonality_mode</code>	'additive' sau 'multiplicative'
<code>changepoint_range</code>	Porțiune din istoric pentru changepoints

Sfaturi Practice

- ▣ **Supraajustare pe trend?** Micșorați `changepoint_prior_scale`
- ▣ **Subajustare pe sezonality?** Măriți `seasonality_prior_scale`
- ▣ **Amplitudinea sezonieră variază?** Folosiți `seasonality_mode='multiplicative'`

Prophet: avantaje și limitări

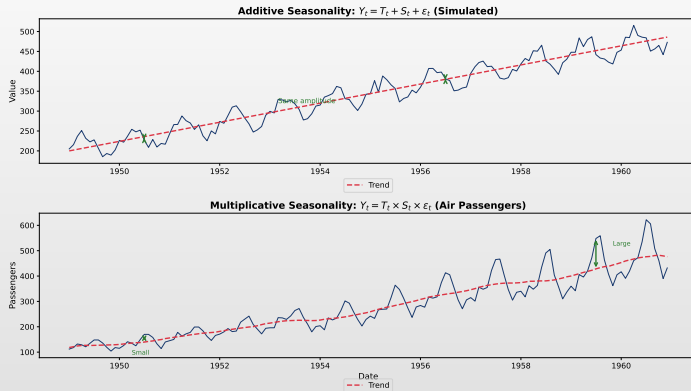
Avantaje

- ▣ **Ușor de folosit:** Ajustare minimă necesară
- ▣ **Interpretabil:** Descompunere clară
- ▣ **Gestionează date lipsă** bine
- ▣ **Efecte sărbători** integrate
- ▣ **Sezonalități multiple**
- ▣ **Regresori externi** suportați
- ▣ **Ajustare rapidă**

Limitări

- ▣ **Nu bazat pe ARIMA:** Fără modelare autocorelație
- ▣ **Focus pe date zilnice:** Mai puțin potrivit pentru frecvență foarte înaltă
- ▣ **Ipoteze de trend:** Liniar/logistic poate să nu se potrivească
- ▣ **CV integrat:** `cross_validation()` disponibil, dar necesită configurare atentă
- ▣ **Risc supraajustare** cu multe sezonalități

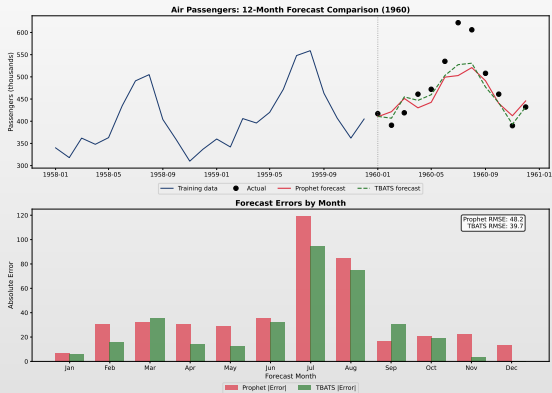
Sezonalitate aditivă vs multiplicativă



TBATS vs Prophet: comparație directă

Caracteristică	TBATS	Prophet
Sezonalități multiple	Da (automat)	Da (manual sau auto)
Efecte sărbători	Nu	Da (integrat)
Regresori externi	Nu	Da
Changepoints trend	Nu (neted)	Da (automat)
Date lipsă	Necesită interpolare	Gestionează nativ
Interpretabilitate	Moderată	Înaltă
Viteză calcul	Lent	Rapid
Date frecvență înaltă	Bun	Moderat
Perioade non-întregi	Da (ex: 365.25)	Da
Intervale incertitudine	Da	Da

Comparație Prophet vs TBATS: prognoze



Când să folosim fiecare model

Folosiți TBATS când:

- Date de frecvență înaltă (orare, sub-zilnice)
- Multiple perioade sezoniere complexe
- Nu sunt necesari regresori externi
- Se preferă selecție automată a modelului
- Se dorește framework tradițional state-space

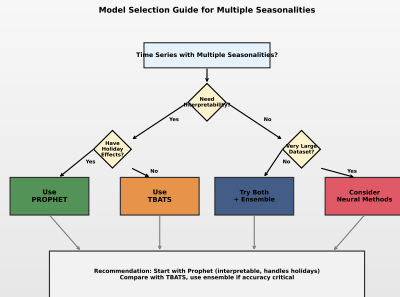
Folosiți Prophet când:

- Prognoză de business (zilnic/săptămânal)
- Efectele sărbătorilor sunt importante
- Trendul are rupturi structurale
- Sunt prezente date lipsă
- Interpretabilitatea este cheie
- Sunt disponibili regresori externi

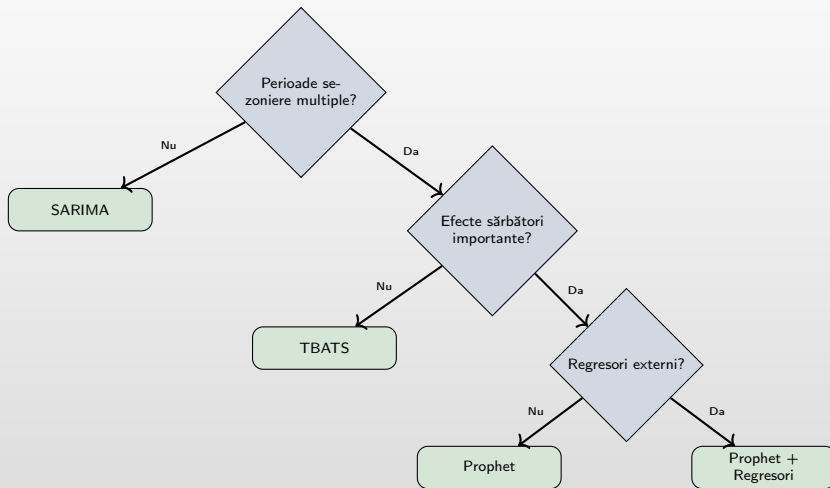
Ghid General

Prophet pentru aplicații de business cu date zilnice
TBATS pentru aplicații tehnice cu date de frecvență înaltă

Ghid selecție model



Diagramă de decizie



Studiu de caz: prognoza cererii de energie

Problema

Prognozați cererea de electricitate pe oră cu:

- ▣ **Tipar zilnic:** Vârf la prânz și seara
- ▣ **Tipar săptămânal:** Mai scăzut în weekend
- ▣ **Tipar anual:** Mai mare vara (AC) și iarna (încălzire)
- ▣ **Efecte sărbători:** Cerere mai mică în sărbători

Abordare

1. Încercați TBATS cu perioade [24, 168, 8766]
2. Încercați Prophet cu sezonality zilnică, săptămânală, anuală + sărbători
3. Comparați folosind cross-validation

Studiu de caz: interpretarea rezultatelor

Metrici de Evaluare

- ▣ **MAPE:** Mean Absolute Percentage Error
- ▣ **RMSE:** Root Mean Square Error
- ▣ **Acoperire:** % din valori reale în intervalul de predicție

Rezultate Tipice

Model	MAPE	RMSE	Acoperire
SARIMA (doar zilnic)	8.5%	450 MW	75%
TBATS	4.2%	220 MW	82%
Prophet	4.8%	250 MW	85%
Prophet + sărbători	3.9%	200 MW	88%

Modelele cu sezonalități multiple depășesc semnificativ SARIMA cu o singură sezonalitate.

Concluzii cheie

Sezonalități Multiple

- ▣ Datele din lumea reală au adesea tipare sezoniere multiple
- ▣ SARIMA standard gestionează doar o perioadă sezonieră
- ▣ TBATS și Prophet sunt proiectate pentru această provocare

Selecția modelului

- ▣ **TBATS**: Automat, gestionează frecvență înaltă, fără regresori externi
- ▣ **Prophet**: Interpretabil, efecte sărbători, regresori externi
- ▣ Ambele folosesc termeni Fourier pentru reprezentare eficientă a sezonaliității

De Reținut

Validați întotdeauna cu cross-validation adecvat pentru serii de timp!

Întrebări?

Întrebări?

Pași Următori:

- ▣ Exersați cu notebook-ul Jupyter
- ▣ Încercați Prophet pe propriile date
- ▣ Explorați NeuralProphet pentru extensia deep learning

Quiz rapid

1. Ce sunt sezonalitățile multiple și de ce SARIMA standard nu le poate gestiona?
2. Cum aproximează termenii Fourier o componentă sezonieră?
3. Ce reprezintă acronimul TBATS?
4. Ce sunt *changepoints* în modelul Prophet?
5. Când alegeți Prophet în loc de TBATS (și invers)?

Răspunsuri quiz

1. **Sezonalități multiple:** O serie prezintă mai multe cicluri periodice suprapuse (ex: zilnic + săptămânal + anual). SARIMA acceptă doar o singură perioadă sezonieră s .
2. **Termeni Fourier:** Aproximează sezonalitatea prin combinații de $\sin(2\pi kt/s)$ și $\cos(2\pi kt/s)$. Cu K armonici se captează forme complexe fără a estima un coeficient pentru fiecare perioadă.
3. **TBATS:** Trigonometric seasonality, **B**ox-Cox transformation, **A**RMA errors, **T**rend, **S**easonal components.
4. **Changepoints:** Puncte în timp unde rata de creștere a trendului se modifică brusc. Prophet le detectează automat și permite ajustarea flexibilității prin parametrul `changepoint_prior_scale`.
5. **Prophet vs TBATS:** Prophet—când aveți regresori externi, efecte de sărbători, date lipsă sau rupturi structurale. TBATS—când frecvența este foarte înaltă sau când doriți o abordare complet automată fără intervenție manuală.

Bibliografie I

Prophet

- ▣ Taylor, S.J., & Letham, B. (2018). Forecasting at Scale, *The American Statistician*, 72(1), 37–45.
- ▣ Harvey, A.C. (1989). *Forecasting, Structural Time Series Models and the Kalman Filter*, Cambridge University Press.

TBATS și netezire exponențială

- ▣ De Livera, A.M., Hyndman, R.J., & Snyder, R.D. (2011). Forecasting Time Series with Complex Seasonal Patterns Using Exponential Smoothing, *JASA*, 106(496), 1513–1527.
- ▣ Hyndman, R.J., Koehler, A.B., Ord, J.K., & Snyder, R.D. (2008). *Forecasting with Exponential Smoothing: The State Space Approach*, Springer.
- ▣ Taylor, J.W. (2003). Short-term Electricity Demand Forecasting Using Double Seasonal Exponential Smoothing, *Journal of the Operational Research Society*, 54(8), 799–805.

Bibliografie II

Comparații și competiții de prognoză

- ▣ Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2020). The M4 Competition, *International Journal of Forecasting*, 36(1), 54–74.
- ▣ Hyndman, R.J., & Athanasopoulos, G. (2021). *Forecasting: Principles and Practice*, 3rd ed., OTexts.
- ▣ Petropoulos, F., et al. (2022). Forecasting: Theory and Practice, *International Journal of Forecasting*, 38(3), 845–1054.

Resurse online și cod

- ▣ **Quantlet**: <https://quantlet.com> — Depozit de cod pentru statistică
- ▣ **Quantinar**: <https://quantinar.com> — Platformă de învățare metode cantitative
- ▣ **GitHub TSA**: <https://github.com/QuantLet/TSA> — Cod Python pentru acest curs