



Time Series Analysis and Forecasting

# Chapter 9: Prophet & TBATS

Seminar



## Seminar Outline

Review Quiz

True/False Questions

Key Definitions

Practice Problems

Worked Examples

Discussion Topics

Exercises for Self-Study

## Quiz 1: Multiple Seasonality Challenge

### Question

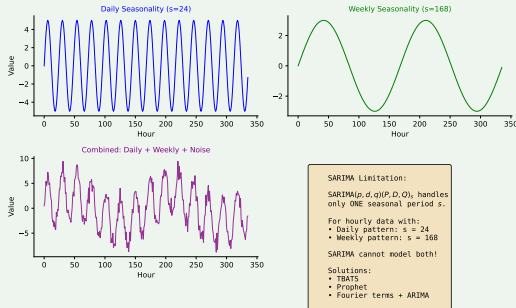
Why can't standard SARIMA handle hourly electricity demand data?

- A) SARIMA can only handle monthly data
- B) SARIMA allows only one seasonal period ( $m$  parameter)
- C) SARIMA doesn't support trend components
- D) SARIMA requires normally distributed data

*Answer on next slide...*

## Quiz 1: Answer

Answer: B – SARIMA allows only one seasonal period



**Key:** Hourly data has daily (24h), weekly (168h), and annual (8760h) patterns. SARIMA's single  $m$  parameter cannot capture all these simultaneously.

## Quiz 2: TBATS Acronym

### Question

What does TBATS stand for?

- A) Trend, Baseline, ARMA, Transform, Seasonal
- B) Trigonometric, Box-Cox, ARMA, Trend, Seasonal
- C) Time-Based Automatic Time Series
- D) Temporal Bayesian Adaptive Trend System

*Answer on next slide...*

## Quiz 2: Answer

Answer: B – Trigonometric, Box-Cox, ARMA, Trend, Seasonal

## TBATS: What Does It Stand For?

<b>T</b>	<b>Trigonometric</b>	Fourier terms for seasonality $\sum [a_n \cos(\frac{2\pi n t}{m}) + b_n \sin(\frac{2\pi n t}{m})]$
<b>B</b>	<b>Box-Cox</b>	Variance stabilization $y^{(adj)} = (y^{(adj)} - 1)/\omega$
<b>A</b>	<b>ARMA</b>	Error autocorrelation $\phi(L)d_t = \theta(L)\varepsilon_t$
<b>T</b>	<b>Trend</b>	Level + slope (possibly damped) $\ell_t = \ell_{t-1} + \phi d_{t-1}$
<b>S</b>	<b>Seasonal</b>	Multiple seasonal periods $m_1, m_2, \dots, m_T$

TBATS: Trigonometric (Fourier for seasonality), Box-Cox (variance stabilization), ARMA (error autocorrelation), Trend (damped local), Seasonal (multiple periods).

## Quiz 3: Fourier Terms

### Question

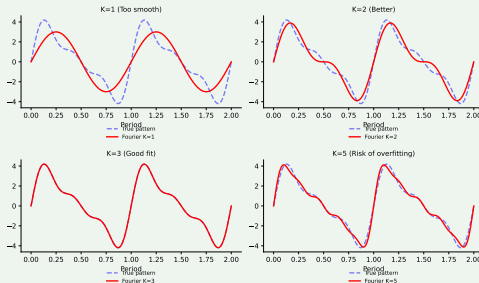
In TBATS, increasing the number of Fourier harmonics ( $K$ ) for a seasonal pattern:

- A) Always improves forecast accuracy
- B) Allows more flexible (complex) seasonal shapes
- C) Reduces the model complexity
- D) Eliminates the need for Box-Cox transformation

*Answer on next slide...*

## Quiz 3: Answer

Answer: B – Allows more flexible seasonal shapes



**Trade-off:** More harmonics = more flexibility but also more parameters.

$$s_t^{(i)} = \sum_{j=1}^{K_i} \left[ a_j^{(i)} \cos \left( \frac{2\pi j t}{m_i} \right) + b_j^{(i)} \sin \left( \frac{2\pi j t}{m_i} \right) \right]$$



## Quiz 4: Prophet Decomposition

### Question

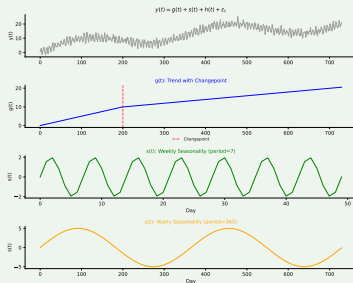
Prophet decomposes a time series into which components?

- A) AR, MA, and seasonal components
- B) Trend, seasonality, holidays, and error
- C) Mean, variance, and autocorrelation
- D) Level, slope, and curvature

*Answer on next slide...*

## Quiz 4: Answer

Answer: B – Trend, seasonality, holidays, and error



**Prophet:**  $y(t) = g(t) + s(t) + h(t) + \varepsilon_t$  where  $g(t)$  = trend,  $s(t)$  = seasonality (Fourier),  $h(t)$  = holidays,  $\varepsilon_t$  = error.

## Quiz 5: Prophet vs TBATS

### Question

When would you choose Prophet over TBATS?

- A) When you need automatic model selection
- B) When you have known holidays and changepoints to incorporate
- C) When you need the most parsimonious model
- D) When your data has no trend

*Answer on next slide...*

## Quiz 5: Answer

Answer: B – Known holidays and changepoints

**TBATS vs Prophet: Head-to-Head Comparison**

Feature	TBATS	Prophet
Multiple seasonalities	Yes (automatic)	Yes (manual/auto)
Holiday effects	No	Yes (built-in)
External regressors	No	Yes
Trend changepoints	No (smooth)	Yes (automatic)
Missing data	Needs interpolation	Handles natively
Interpretability	Moderate	High
Computation speed	Slow	Fast
High-frequency data	Good	Moderate
Non-integer periods	Yes (e.g., 365.25)	Yes
Best for	Technical/high-freq	Business/daily

**Prophet advantages:** Easy holiday integration, analyst-in-the-loop, handles missing data, interpretable components.

**TBATS advantages:** Automatic model selection, handles complex seasonality without domain expertise.

## Quiz 6: Seasonality Mode

### Question

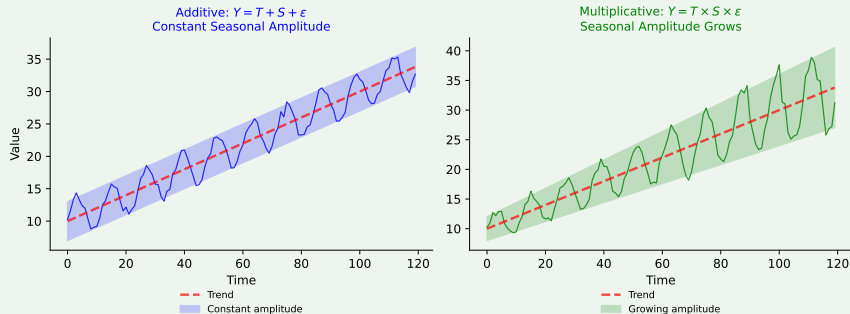
For retail sales data where December sales are 3x the monthly average, which seasonality mode is more appropriate in Prophet?

- A) Additive seasonality
- B) Multiplicative seasonality
- C) Both work equally well
- D) Neither—use ARIMA instead

*Answer on next slide...*

## Quiz 6: Answer

Answer: B – Multiplicative seasonality



**Key:** When seasonal amplitude scales with the level, use multiplicative.

**Additive:**  $y = g(t) + s(t)$  (constant seasonal effect)

**Multiplicative:**  $y = g(t) \cdot (1 + s(t))$  (proportional seasonal effect)

## Quiz 7: Prophet Changepoints

### Question

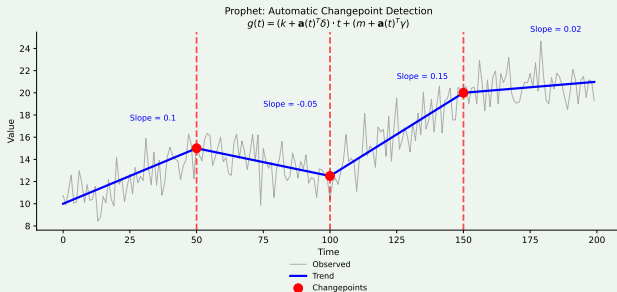
In Prophet, changepoints allow the model to:

- A) Change the seasonal period automatically
- B) Adjust the trend slope at specific points in time
- C) Switch between additive and multiplicative modes
- D) Detect and remove outliers

*Answer on next slide...*

## Quiz 7: Answer

Answer: B – Adjust trend slope at specific points



**Changepoints:** Allow piecewise linear trend with different slopes.

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

Prophet automatically detects changepoints or you can specify them manually.



## Quiz 8: Model Selection

### Question

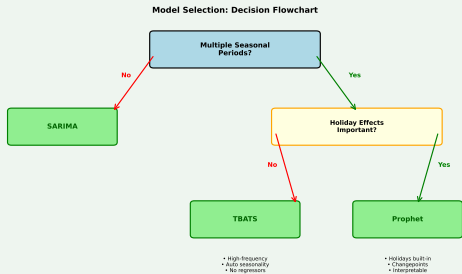
You have daily call center data with weekly seasonality only. Which model is most appropriate?

- A) TBATS (designed for multiple seasonality)
- B) Prophet (handles any seasonality well)
- C) Standard SARIMA (simpler and sufficient)
- D) LSTM neural network (most flexible)

*Answer on next slide...*

## Quiz 8: Answer

Answer: C – Standard SARIMA is sufficient



**Principle of parsimony:** Use the simplest model that fits the data.

With only weekly seasonality ( $m = 7$ ), SARIMA works fine.

Use TBATS/Prophet when you *need* multiple seasonalities or special features.

## Quiz 9: Prophet Uncertainty

### Question

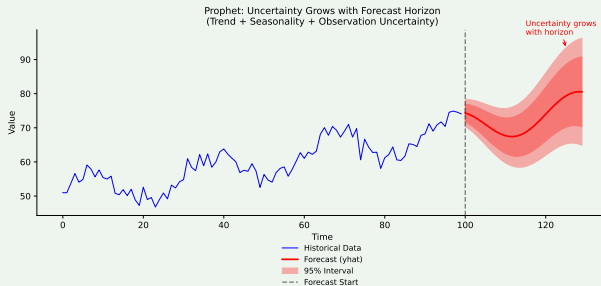
Prophet generates prediction intervals by:

- A) Assuming normally distributed residuals
- B) Sampling from the posterior distribution of parameters
- C) Using bootstrap resampling of historical errors
- D) Applying a fixed multiplier to point forecasts

*Answer on next slide...*

## Quiz 9: Answer

Answer: B – Sampling from posterior distribution



Prophet uses Bayesian estimation: MAP for point forecasts, MCMC/simulation for intervals. Uncertainty from both trend (change-points) and observation noise.

## Quiz 10: Practical Application

### Question

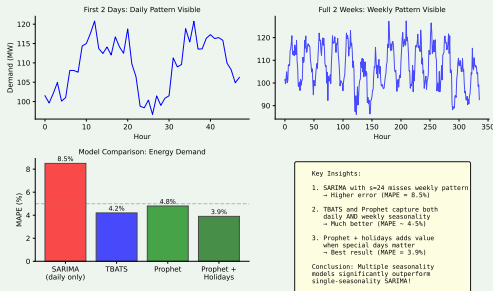
For forecasting hourly energy demand with daily, weekly, and annual patterns plus holiday effects, which approach is best?

- A) SARIMA with  $m = 24$
- B) TBATS with three seasonal periods
- C) Prophet with custom holidays
- D) Either TBATS or Prophet, depending on whether holidays are important

*Answer on next slide...*

## Quiz 10: Answer

Answer: D – TBATS or Prophet depending on needs



Both handle multiple seasonality: Holiday effects crucial  $\Rightarrow$  Prophet; automatic selection  $\Rightarrow$  TBATS. Often try both and compare via cross-validation.

## True/False Questions

Determine if each statement is True or False:

1. Prophet was developed by Facebook (Meta) for business forecasting.
2. TBATS can only handle two seasonal periods at most.
3. In Prophet, the default trend is logistic growth.
4. Fourier terms approximate seasonality using sine and cosine functions.
5. Prophet requires equally spaced time series data.
6. The Box-Cox transformation in TBATS stabilizes variance.

*Answers on next slide...*

## True/False: Solutions

1. Prophet was developed by Facebook (Meta) for business forecasting.

TRUE

Released in 2017, designed for “analyst in the loop” forecasting at scale.

2. TBATS can only handle two seasonal periods at most.

FALSE

TBATS can handle any number of seasonal periods (e.g., daily, weekly, annual).

3. In Prophet, the default trend is logistic growth.

FALSE

Default is piecewise linear. Logistic growth must be explicitly specified.

4. Fourier terms approximate seasonality using sine and cosine functions.

TRUE

$$s(t) = \sum_{k=1}^K [a_k \cos(2\pi kt/m) + b_k \sin(2\pi kt/m)]$$

5. Prophet requires equally spaced time series data.

FALSE

Prophet handles missing data and irregular timestamps gracefully.

6. The Box-Cox transformation in TBATS stabilizes variance.

TRUE

$$y^{(\lambda)} = (y^\lambda - 1)/\lambda \text{ for } \lambda \neq 0; \log(y) \text{ for } \lambda = 0.$$



## Fourier Series Representation

### Definition: Fourier Harmonics

A **Fourier series** represents a periodic function  $s(t)$  with period  $m$  as:

$$s(t) = a_0 + \sum_{k=1}^K \left[ a_k \cos\left(\frac{2\pi kt}{m}\right) + b_k \sin\left(\frac{2\pi kt}{m}\right) \right]$$

where:

- ▣  $k$ : harmonic number (frequency multiplier)
- ▣  $a_k, b_k$ : Fourier coefficients (estimated from data)
- ▣  $K$ : number of harmonics (controls flexibility)

### Why It Works

**Fourier's Theorem:** Any periodic function can be expressed as a (possibly infinite) sum of sine and cosine waves. With finite  $K$ , we get an approximation.

## Problem 1: Fourier Terms Calculation

### Exercise

For daily data with weekly seasonality ( $m = 7$ ), you want to use Fourier terms with  $K = 3$  harmonics. How many parameters does this add to the model?

*Answer on next slide...*

## Problem 1: Solution

**Solution: 6 parameters****Each harmonic** requires 2 parameters (sine and cosine coefficients):

$$s(t) = \sum_{k=1}^K \left[ a_k \cos\left(\frac{2\pi kt}{m}\right) + b_k \sin\left(\frac{2\pi kt}{m}\right) \right]$$

With  $K = 3$  harmonics:

- ▣  $k = 1$ :  $a_1, b_1$  (fundamental frequency 1/7 cycles per day)
- ▣  $k = 2$ :  $a_2, b_2$  (first overtone, 2/7 cycles per day)
- ▣  $k = 3$ :  $a_3, b_3$  (second overtone, 3/7 cycles per day)

**Total:**  $2 \times K = 2 \times 3 = 6$  parameters**Nyquist Limit**

Maximum useful  $K = \lfloor m/2 \rfloor$ . With  $m$  discrete observations per period, we can identify at most  $m/2$  distinct frequencies. For  $m = 7$ :  $K_{\max} = 3$ .

## Problem 2: Choosing Seasonality Mode

### Exercise

You're forecasting monthly hotel bookings. The data shows:

- ▣ July 2020: 1000 bookings (peak season)
- ▣ January 2020: 400 bookings (off-season)
- ▣ July 2023: 2000 bookings (peak season)
- ▣ January 2023: 800 bookings (off-season)

Should you use additive or multiplicative seasonality? Why?

*Answer on next slide...*

## Problem 2: Solution

### Solution: Multiplicative seasonality

**Analysis:** Check if seasonal amplitude is proportional to level.

Year	July	January	Ratio (Jul/Jan)
2020	1000	400	2.5
2023	2000	800	2.5

**Key observation:** The *ratio* stays constant (2.5), not the difference!

- ▣ Additive would mean: July always +600 above January
- ▣ But 2020:  $1000 - 400 = 600$ ; 2023:  $2000 - 800 = 1200$

**Conclusion:** Use multiplicative: `seasonality_mode='multiplicative'`

## Problem 3: TBATS Model Interpretation

### Exercise

A TBATS model fitted to hourly electricity data reports:

- Box-Cox  $\lambda = 0.5$
- Seasonal periods:  $m_1 = 24$ ,  $m_2 = 168$
- Fourier terms:  $K_1 = 5$ ,  $K_2 = 3$

What does each component tell you about the data?

*Answer on next slide...*

## Problem 3: Solution

### Solution

**Box-Cox**  $\lambda = 0.5$ :

- Square root transformation applied
- Data had increasing variance with level
- Transformation:  $y^{(0.5)} = \sqrt{y}$

**Seasonal periods:**

- $m_1 = 24$ : Daily pattern (24 hours)
- $m_2 = 168$ : Weekly pattern ( $7 \times 24 = 168$  hours)

**Fourier terms:**

- $K_1 = 5$  for daily: Complex intraday pattern (5 harmonics capture peaks, valleys)
- $K_2 = 3$  for weekly: Simpler weekly pattern (weekday vs weekend)

**Total seasonal parameters:**  $2(K_1 + K_2) = 2(5 + 3) = 16$

## Problem 4: Prophet Holiday Effects

### Exercise

You're forecasting daily restaurant revenue. You want to add these holiday effects to Prophet:

- ☐ Valentine's Day (Feb 14) – major boost
- ☐ Easter (variable date) – restaurant closed
- ☐ Christmas (Dec 25) – restaurant closed

Write the Python code to create the holidays dataframe for 2024-2025.

*Answer on next slide...*



## Problem 4: Solution

### Solution

```
import pandas as pd
from prophet import Prophet

# Create holidays dataframe with window effects
holidays = pd.DataFrame({
    'holiday': ['valentines', 'valentines', 'easter', 'easter',
               'christmas', 'christmas'],
    'ds': pd.to_datetime(['2024-02-14', '2025-02-14',
                          '2024-03-31', '2025-04-20',
                          '2024-12-25', '2025-12-25']),
    'lower_window': [-1, -1, 0, 0, -1, -1], # days before
    'upper_window': [0, 0, 0, 0, 0, 0] # days after
})

model = Prophet(holidays=holidays)

model.fit(df) # df must have 'ds' and 'y' columns
```

### Key Points

- `lower_window=-1`: Effect starts 1 day before
- Easter: closed = zero revenue (handled by model as large negative effect)
- Valentine's: expect positive effect coefficient

## Example: Retail Sales Forecasting with Prophet

 TSA\_ch9\_prophet\_tbats

### Scenario

Monthly retail sales (2018-2023): December peaks, COVID-19 break in 2020, growing trend.

### Prophet Configuration

```
model = Prophet(seasonality_mode='multiplicative',  
                 changepoint_prior_scale=0.5, yearly_seasonality=True)  
model.add_country_holidays(country_name='US')
```

### Key Decision

Multiplicative seasonality: December effect proportional to baseline level.

## Example: Energy Demand with TBATS

 TSA\_ch9\_prophet\_tbats

### Scenario

Hourly electricity: intraday (24h), weekly (168h), annual (8760h) patterns.

### TBATS in R

```
library(forecast)
energy_msts <- msts(energy_data, seasonal.periods = c(24, 168, 8760))
fit <- tbats(energy_msts); fc <- forecast(fit, h = 168)
```

### Note

TBATS automatically selects  $K$  for each seasonal period via AIC.

## Example: Cross-Validation Comparison

TSA ch9 prophet tbats

### Objective

Compare Prophet, TBATS, and SARIMA on 2 years of daily sales data.

### Prophet Cross-Validation

```
from prophet.diagnostics import cross_validation, performance_metrics
df_cv = cross_validation(model, initial='365 days',
    period='90 days', horizon='30 days')
metrics = performance_metrics(df_cv)
print(f"MAPE: {metrics['mape'].mean():.2%}")
```

### Typical Results

Model	MAPE	Computation Time
SARIMA (weekly only)	8.5%	Fast
TBATS (weekly + yearly)	6.2%	Moderate
Prophet (weekly + yearly + holidays)	5.8%	Fast

## Discussion: When to Use Which Model?

### Key Question

You have a new forecasting task. How do you choose between SARIMA, TBATS, and Prophet?

### Decision Framework

- 1. How many seasonal periods?**
  - ▶ One  $\Rightarrow$  SARIMA may suffice
  - ▶ Multiple  $\Rightarrow$  TBATS or Prophet
- 2. Do you have domain knowledge to encode?**
  - ▶ Holidays, events, changepoints  $\Rightarrow$  Prophet
  - ▶ Let the data speak  $\Rightarrow$  TBATS
- 3. Interpretability requirements?**
  - ▶ Need to explain components  $\Rightarrow$  Prophet
  - ▶ Just need forecasts  $\Rightarrow$  Either

## Discussion: Overfitting with Fourier Terms

### Key Question

Can you have too many Fourier terms? What are the symptoms?

### Answer: Yes!

Symptoms of overfitting:

- In-sample fit is excellent, but out-of-sample is poor
- Seasonality looks “jagged” or unrealistic
- Forecasts oscillate wildly

### Guidelines

- Maximum  $K \leq m/2$  (Nyquist limit)
- Start with  $K = 3-5$  for most applications
- Use cross-validation to select  $K$
- Prophet default:  $K = 10$  for yearly,  $K = 3$  for weekly

## Discussion: Handling Structural Breaks

### Scenario

Your historical data includes COVID-19 period (2020-2021). How do you handle this when forecasting 2024?

### Options

1. **Exclude COVID period:** Train only on pre-COVID and post-COVID data
2. **Use changepoints:** Let Prophet detect/specify breaks
3. **Add regressors:** Include COVID indicator variable
4. **Adjustment:** Manually adjust 2020-2021 values to “normal”

### Prophet Approach

```
model = Prophet(changepoints=['2020-03-15', '2021-06-01'])  
df['covid'] = (df['ds'] >= '2020-03-15') & (df['ds'] < '2021-06-01')  
model.add_regressor('covid')
```

## Take-Home Exercises

1. **Theoretical:** Prove that  $K = m/2$  Fourier terms can represent any periodic function with period  $m$  (for even  $m$ ).
2. **Computation:** For the seasonal pattern below (daily data, weekly cycle), determine the minimum number of Fourier harmonics needed:  

Mon: 100, Tue: 110, Wed: 115, Thu: 110, Fri: 120, Sat: 80, Sun: 65
3. **Applied:** Download hourly electricity demand data from a public source:
  - ▶ Fit both TBATS (in R) and Prophet (in Python)
  - ▶ Compare forecast accuracy using RMSE and MAPE
  - ▶ Visualize the component decompositions
4. **Critical Thinking:** Why might Prophet perform poorly on high-frequency financial data (e.g., minute-by-minute stock prices)?



## Exercise Solutions Hints

### Hints

1. By Fourier theorem, any periodic function can be represented as sum of sines and cosines. With period  $m$ , frequencies are  $k/m$  for  $k = 1, \dots, m/2$ .
2. The pattern has:
  - ▶ One peak (Friday) and one trough (Sunday)
  - ▶ Fairly smooth transitions
  - ▶  $K = 2$  or  $K = 3$  likely sufficient (try and compare)
3. For electricity data:
  - ▶ Include daily (24h) and weekly (168h) patterns
  - ▶ Add holidays for your region in Prophet
  - ▶ Expect MAPE around 3-5% for hourly forecasts
4. Financial data issues:
  - ▶ No clear seasonality (market efficiency)
  - ▶ High noise-to-signal ratio
  - ▶ Prophet designed for “business” data with trends and seasons

## Key Takeaways from This Seminar

### Multiple Seasonality Models

1. **TBATS**: Automatic, Fourier-based, handles any number of seasonal periods
2. **Prophet**: Analyst-friendly, explicit holiday/event handling, interpretable
3. Use **SARIMA** when only one seasonal period exists

### Key Decisions

- ▣ **Seasonality mode**: Additive (constant amplitude) vs Multiplicative (proportional)
- ▣ **Fourier terms**: More = flexible but risk overfitting; use CV to select
- ▣ **Changepoints**: Allow trend to adapt to structural breaks

### Remember

**Prophet**: Great when you have domain knowledge to encode

**TBATS**: Great for automatic modeling of complex seasonality