Time Series Analysis and Forecasting

Chapter 8: Modern Extensions

Seminar

Seminar Outline

Quiz: ARFIMA and Long Memory

Quiz: Machine Learning for Time Series

Quiz: LSTM Networks

True/False Questions

Practice Problems

Summary

Quiz 1: Hurst Exponent

Question

A time series has Hurst exponent $H = 0.8$. What does this indicate?

A) The series is a pure random walk

B) The series has long memory and is persistent (trend-following)

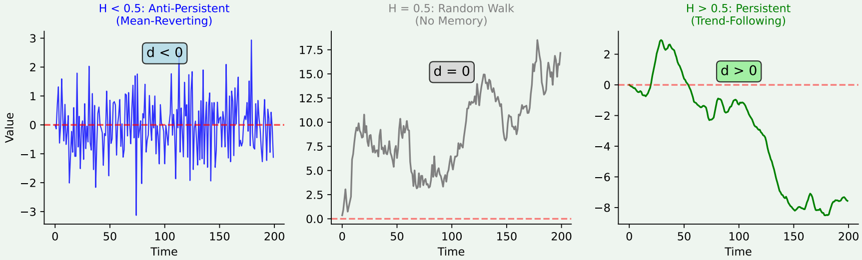
C) The series is anti-persistent (mean-reverting)

D) The series is stationary I(0)

Answer on next slide...

Quiz 1: Answer

Answer: B – Long memory and persistence



Hurst Exponent: $H = 0.5$ (random walk), $0.5 < H < 1$ (persistence), $0 < H < 0.5$ (mean reversion)

Quiz 2: Fractional Differencing Parameter

Question

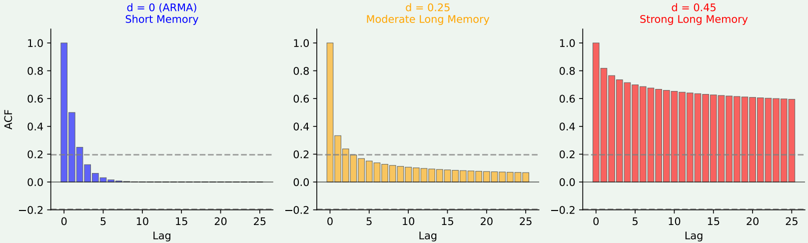
In the ARFIMA(p, d, q) model, the parameter d can take values:

A) Only integer values (0, 1, 2, ...)
B) Only $d = 0$ or $d = 1$
C) Any real value, including fractional
D) Only negative values

Answer on next slide...

Quiz 2: Answer

Answer: C – Any real value



Interpretation: $d = 0$ (ARMA), $0 < d < 0.5$ (long memory, stationary), $0.5 \leq d < 1$ (non-stationary), $d = 1$ (unit root).

Relation to Hurst: $d = H - 0.5$

Quiz 3: Long Memory in Financial Series

Question

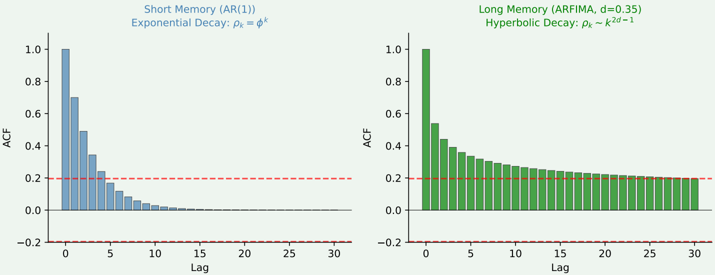
In which financial series is long memory most commonly documented?

A) Stock prices
B) Daily returns
C) Volatility (squared returns)
D) Trading volume

Answer on next slide...

Quiz 3: Answer

Answer: C – Volatility



Stylized facts: Returns are memoryless ($H \approx 0.5$), volatility has long memory ($H \approx 0.7 - 0.9$) due to clustering. Basis for FIGARCH/HAR-RV models.

Quiz 4: Feature Engineering

Question

To apply Random Forest to time series, we must create:

A) Dummy variables for each observation

B) Lag features and rolling statistics

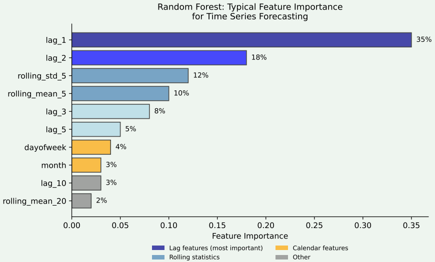
C) Fourier transforms of the series

D) Only the first difference of the series

Answer on next slide...

Quiz 4: Answer

Answer: B – Lag features and rolling statistics



Feature Engineering: Lag features $(y_{t-1}, \ldots, y_{t-k})$, rolling stats (mean, std), calendar features. Transforms forecasting into supervised regression.

Quiz 5: Time Series Cross-Validation

Question

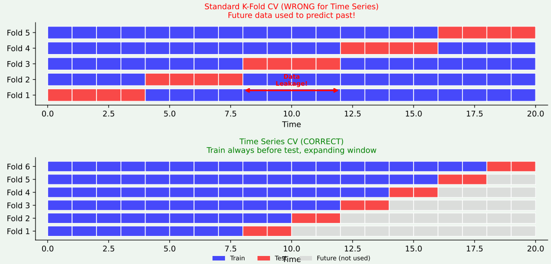
Why can't we use standard k-fold cross-validation for time series?

A) It's too slow for long series
B) It violates temporal order and causes data leakage
C) It only works for classification
D) It requires too much data

Answer on next slide...

Quiz 5: Answer

Answer: B – Violates temporal order



Problem: k-fold shuffles randomly, trains on future, tests on past $\Rightarrow$ data leakage. **Solution**: Time Series Split (Walk-Forward Validation).

Quiz 6: Feature Importance in Random Forest

Question

Feature importance in Random Forest for time series helps us:

A) Eliminate all low-importance variables
B) Identify which lags and features are most predictive
C) Determine Granger causality
D) Calculate confidence intervals

Answer on next slide...

Quiz 6: Answer

Answer: B – Identifies predictive features

Uses of feature importance:

- Understanding temporal structure
- Selecting optimal number of lags
- Identifying relevant factors

Caution:

- Importance does NOT imply causality
- Correlated variables may share importance
- Use for interpretation, not causal inference

Quiz 7: LSTM Advantage

Question

What is the main advantage of LSTM over simple RNNs?

A) It's faster to train

B) It solves the vanishing/exploding gradient problem

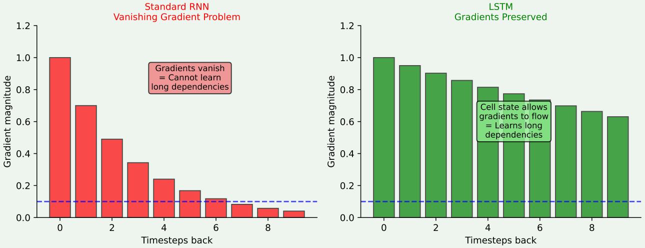
C) It requires less data

D) It's easier to interpret

Answer on next slide...

Quiz 7: Answer

Answer: B – Solves the gradient problem



RNN Problem: Gradients decay exponentially. **LSTM Solution**: Cell state (highway), forget/input/output gates control information flow.

Quiz 8: Data Preparation for LSTM

Question

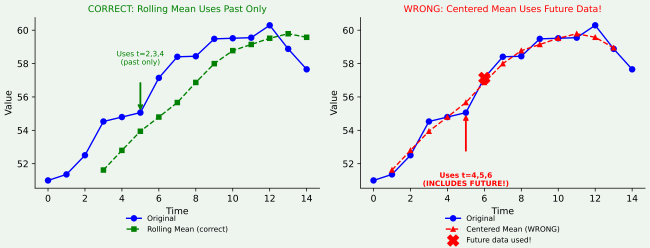
Before training an LSTM, data should be:

A) Log-transformed
B) Normalized/scaled to [0,1] or [-1,1]
C) Differenced twice
D) Converted to integers

Answer on next slide...

Quiz 8: Answer

Answer: B – Normalized/scaled



Why? Activation functions work in limited ranges; faster convergence; numerical stability. **Methods**: Min-Max → [0,1] or Standard (mean 0, std 1). Fit on train only!

Quiz 9: LSTM Hyperparameters

Question

Which is NOT a typical LSTM hyperparameter?

A) Number of units (neurons) per layer
B) Input sequence length
C) Learning rate
D) Differencing parameter d

Answer on next slide...

Quiz 9: Answer

Answer: D – The d parameter

d is specific to ARFIMA models, not LSTM!

LSTM Hyperparameters:

- **Architecture**: number of layers, units/layer
- **Sequence**: lookback length
- **Training**: learning rate, batch size, epochs
- **Regularization**: dropout, early stopping

Tuning: Grid search or Bayesian optimization with time series CV

Quiz 10: Model Selection

Question

When comparing ARFIMA, Random Forest, and LSTM for forecasting:

A) LSTM always wins because it's deep learning

B) ARFIMA is always best for financial data

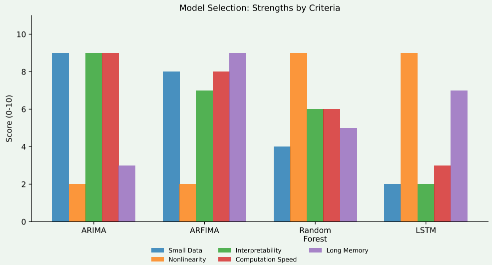
C) The best model depends on data characteristics and problem requirements

D) Random Forest can't be used for time series

Answer on next slide...

Quiz 10: Answer

Answer: C – Depends on data and requirements



Guidelines: ARFIMA (long memory, interpretability), RF (nonlinear, feature importance), LSTM (long sequences, complex patterns). Always validate with time series CV!

True/False Questions

Determine if each statement is True or False:

1. The Hurst exponent $H = 0.5$ indicates long memory.
2. ARFIMA reduces to ARIMA when d is an integer.
3. Standard k-fold CV is appropriate for time series data.
4. LSTM can capture long-range dependencies better than simple RNNs.
5. Feature importance in Random Forest proves causality.
6. Normalizing data is optional when training neural networks.

Answers on next slide...

True/False: Solutions

1. The Hurst exponent $H = 0.5$ indicates long memory. **FALSE**
   $H = 0.5$ means random walk (no memory). Long memory: $H > 0.5$.
2. ARFIMA reduces to ARIMA when d is an integer. **TRUE**
   With $d = 0$ or $d = 1$, ARFIMA becomes standard ARMA or ARIMA.
3. Standard k-fold CV is appropriate for time series data. **FALSE**
   Use time series split to maintain temporal order and avoid data leakage.
4. LSTM can capture long-range dependencies better than simple RNNs. **TRUE**
   Cell state and gates allow gradients to flow without vanishing.
5. Feature importance in Random Forest proves causality. **FALSE**
   Shows predictive power, not causal relationship.
6. Normalizing data is optional when training neural networks. **FALSE**
   Critical for convergence and stability with activation functions.

Problem 1: Hurst Exponent Estimation

Exercise

Given daily Bitcoin returns, estimate the Hurst exponent using the R/S method and interpret the result.

Solution Steps:

1. Calculate mean over subintervals of different lengths n
2. For each n: calculate Range(R) and Std(S)
3. The ratio R/S grows as n^H
4. Fit regression: $\log(R/S) = H \cdot \log(n) + c$

Python code: `nolds.hurst_rs(returns)`

Problem 1: Solution and Interpretation

Typical Bitcoin Results

- Returns: $H \approx 0.45 - 0.55$ (approximately random walk)
- Volatility ($|\text{returns}|$): $H \approx 0.75 - 0.85$ (long memory!)

Interpretation:

- Returns are hard to predict (EMH approximately valid)
- Volatility is predictable over long horizons
- Implications for risk management and VaR

Application: FIGARCH models may outperform standard GARCH

Problem 2: Random Forest for Forecasting

Exercise

Build a Random Forest model for 1-day ahead Bitcoin price forecasting. Evaluate using TimeSeriesSplit.

Pipeline:

1. **Feature engineering**:
   - Lags: $y_{t-1}, y_{t-2}, \ldots, y_{t-7}$
   - Rolling mean/std: 7, 14, 30 days
2. **Train/Test split**: TimeSeriesSplit(n_splits=5)
3. **Model**: RandomForestRegressor(n_estimators=100)
4. **Evaluation**: RMSE, MAE, Direction Accuracy

Problem 2: Code and Results

Python Code

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import TimeSeriesSplit

tscv = TimeSeriesSplit(n_splits=5)
rf = RandomForestRegressor(n_estimators=100)

for train_idx, test_idx in tscv.split(X):
    rf.fit(X[train_idx], y[train_idx])
    pred = rf.predict(X[test_idx])
```

Typical results:

- Direction accuracy: 52-55% (slightly above random)
- Feature importance: lag-1 and rolling_std dominate

Problem 3: LSTM for Time Series

Exercise

Implement a simple LSTM model for Bitcoin forecasting. Compare with Random Forest.

Simple LSTM Architecture:

1. Input: 30-day sequences
2. LSTM layer: 50 units
3. Dense output: 1 neuron (forecast)
4. Loss: MSE, Optimizer: Adam

Important steps:

- MinMaxScaler normalization
- Reshape to [samples, timesteps, features]
- Early stopping to prevent overfitting

Problem 3: LSTM Code

Keras/TensorFlow Code

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense

model = Sequential([
    LSTM(50, input_shape=(30, 1)),
    Dense(1)
])

model.compile(optimizer='adam', loss='mse')
model.fit(X_train, y_train, epochs=50,
          validation_split=0.1, verbose=0)
```

Typical RF vs LSTM comparison:

- RMSE similar (LSTM slightly better on smooth data)
- RF: faster, more interpretable
- LSTM: captures complex patterns better

Su

ARFIMA

- Series with long memory (volatility, hydrology)
- When $0 < d < 0.5$ is theoretically justified
- Statistical interpretability is important

Random Forest

- Nonlinear relationships between features
- Feature importance for understanding
- Structured data, not too long series

LSTM

- Long sequences with complex dependencies
- Sufficient data for deep learning
- Patterns difficult to capture with classical methods

Ke

ARFIMA and Long Memory

- Fractional differencing: $(1-L)^d y_t = \varepsilon_t$
- Hurst exponent: $d = H - 0.5$
- ACF for long memory: $\rho(k) \sim k^{2d-1}$ (slow decay)

Machine Learning

- Lag feature: $X_t = [y_{t-1}, y_{t-2}, \ldots, y_{t-k}]$
- RMSE: $\sqrt{\frac{1}{n}\sum(y_i - \hat{y}_i)^2}$
- Direction Accuracy: $\frac{1}{n}\sum 1[\text{sign}(\Delta y) = \text{sign}(\Delta \hat{y})]$

LSTM

- Forget gate: $f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$
- Cell update: $C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$

Thank You!

Questions?

`danpele@ase.ro`