Exponential Smoothing

Simon U., Michael Y., Ben H., Sachid Deshmukh

March 3, 2020

Introduction

- **What is exponential smoothing?**
  Forecasting future observations using weighted averages of past observations, with the weights decaying exponentially as observations recede further into the past
  - Presumably, more recent data points are more predictive than older points
  - This framework generates reliable forecasts quickly and for a wide range of time series
- **Exponential Smoothing Models**

1. Naive—all weight is given to the last observation
2. Average—each past observation is given equal weight
3. Exponential weighted average—Recent observations get higher weight, older observations less weight
4. Holt linear—Same as 3, but accounts for time series with trend
5. Holt-Winters—Same as 4, but also accounts for time series with seasonality
6. State Space—ETS models - also generate forecast/prediction

ES_1: Naive model

- The naive forecasting model can be thought of us exponential smoothing
- Where 100 percent of weight is given to the last observation:

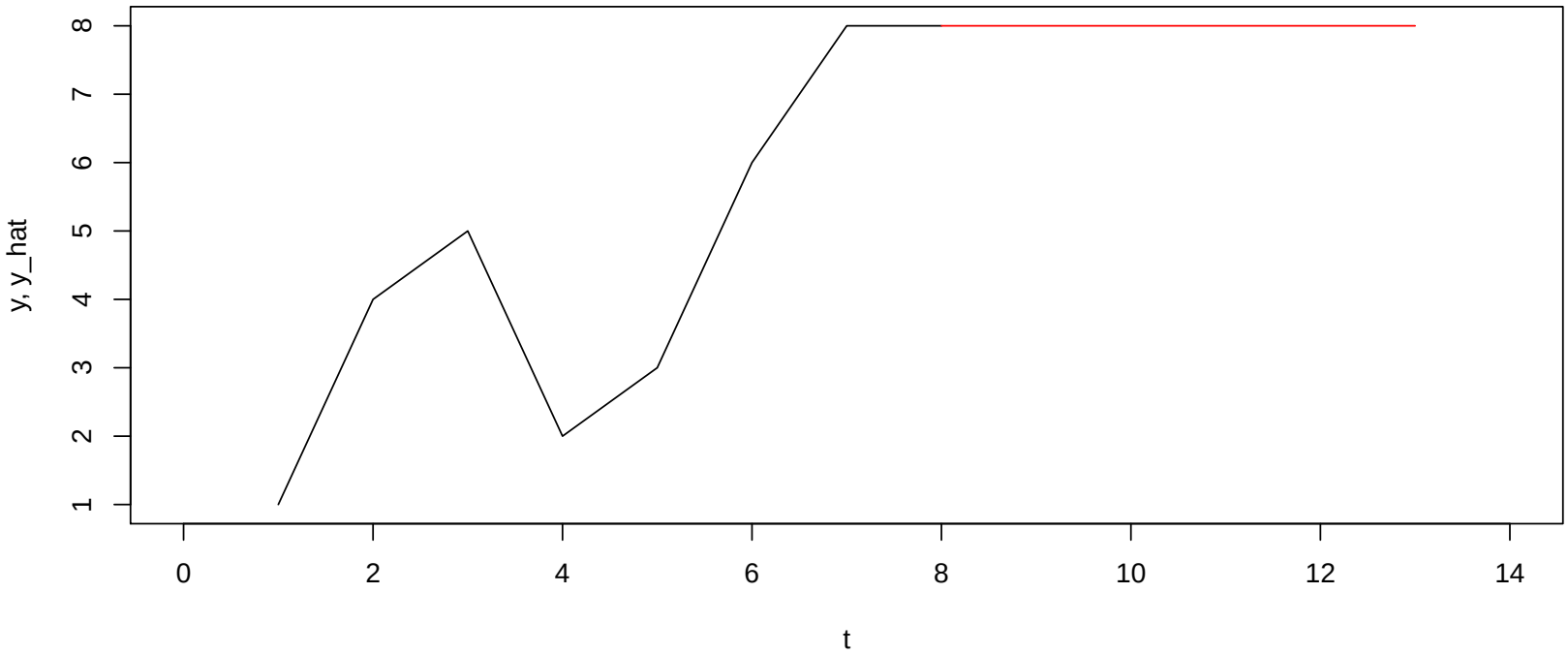
$$\hat{y}_{T+h|T} = y_T$$

```
forecast_naive <- function(y, h) {
    n <- length(y)
    y_hat <- rep(y[n], h)
    return( y_hat )
}
```

ES_1: Naive model: Example

```
y <- c(1, 4, 5, 2, 3, 6, 8)
y_hat <- forecast_naive(y, h=7)

plot_forecast(y, y_hat)
```



ES_2: Average model

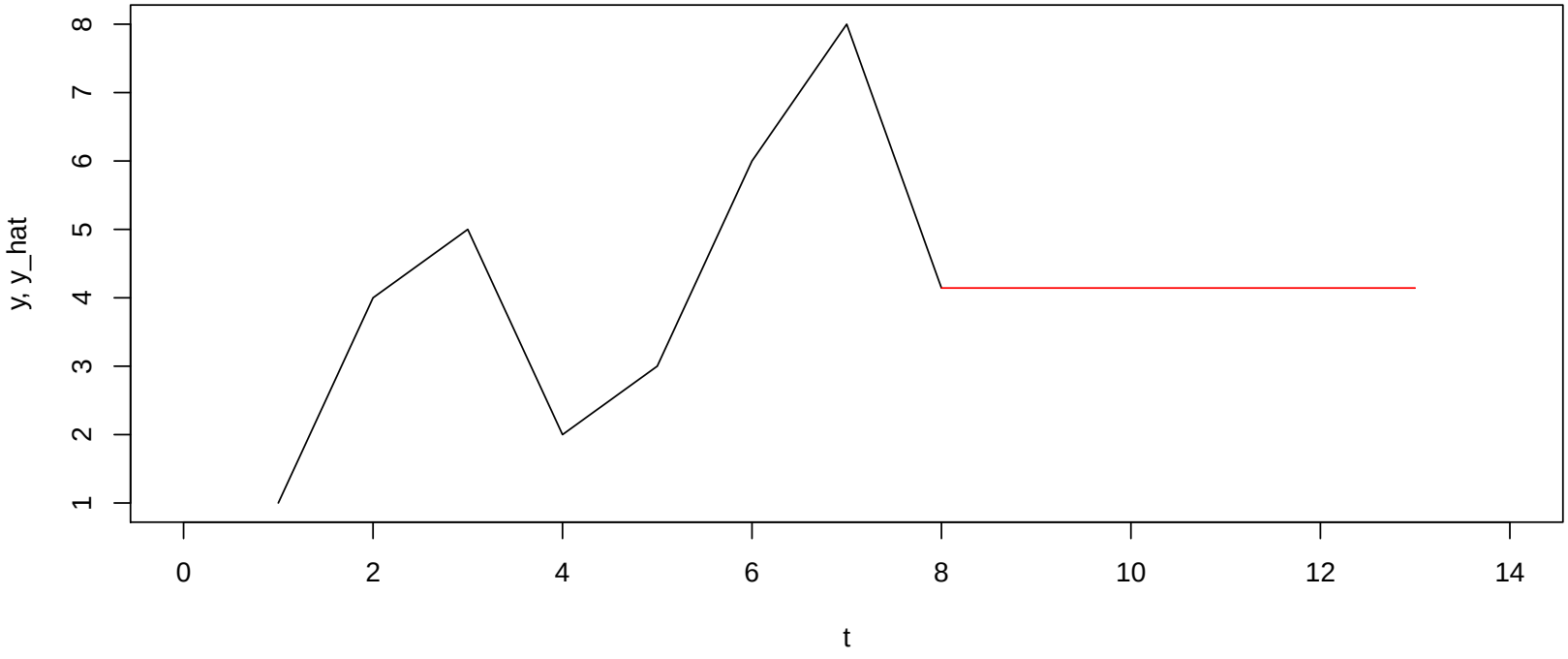
- All future values are forecast as the average of the observed data
- Equivalent to to exponential smoothing where each observation is given equal weight

$$\hat{y}_{T+h|T} = \frac{1}{T}\sum_{t=1}^{T} y_t,$$

```
forecast_avg <- function(y, h) {
    y_hat <- rep(mean(y), h)
    return( y_hat )
}
```

ES_2: Average model: Example

```
y_hat <- forecast_avg(y, h=7)
plot_forecast(y, y_hat)
```



ES_3: Simple Exponential Smoothing [SES]

- SES stands at the core and servers as the foundation for some of the most successful forecasting methods for modeling time series data. The produced forecasts are weighted averages of past observations, with the weights decaying exponentially as the observations get older. SES is actually suitable for forecasting data with no clear trend or seasonal pattern.

Exploring "Exponential" aspect

- The formula for estimating a value at $T+1$ is as follows:

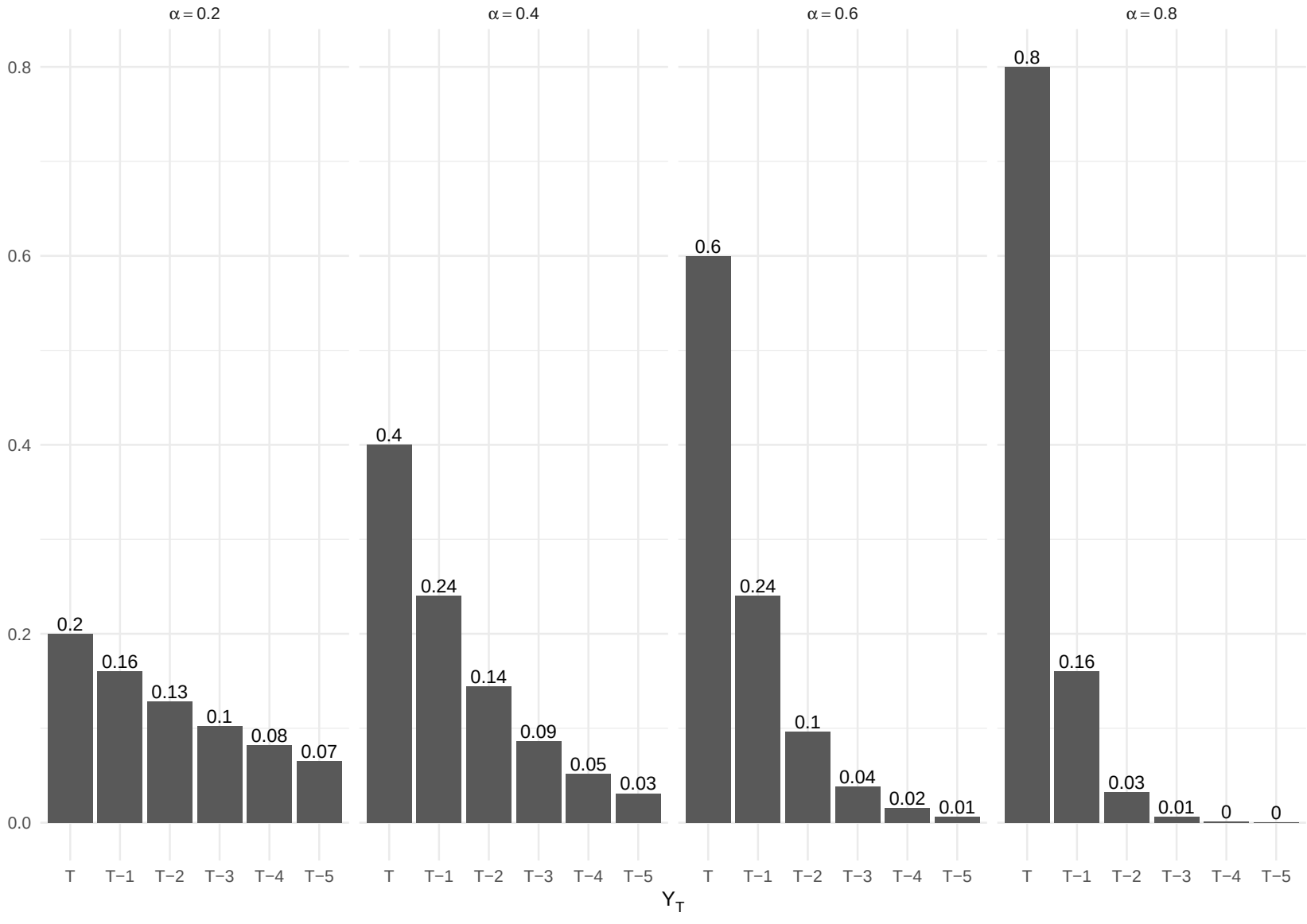
$$\hat{y}_{T+1|T} = \alpha y_T + \alpha(1-\alpha)y_{T-1} + \alpha(1-\alpha)^2 y_{T-2} + \cdots$$

Geometric Distribution

- If the probability of a success in one trial is p and the probability of a failure is $1-p$, then the probability of finding the first success in the n^{th} tiral is given by

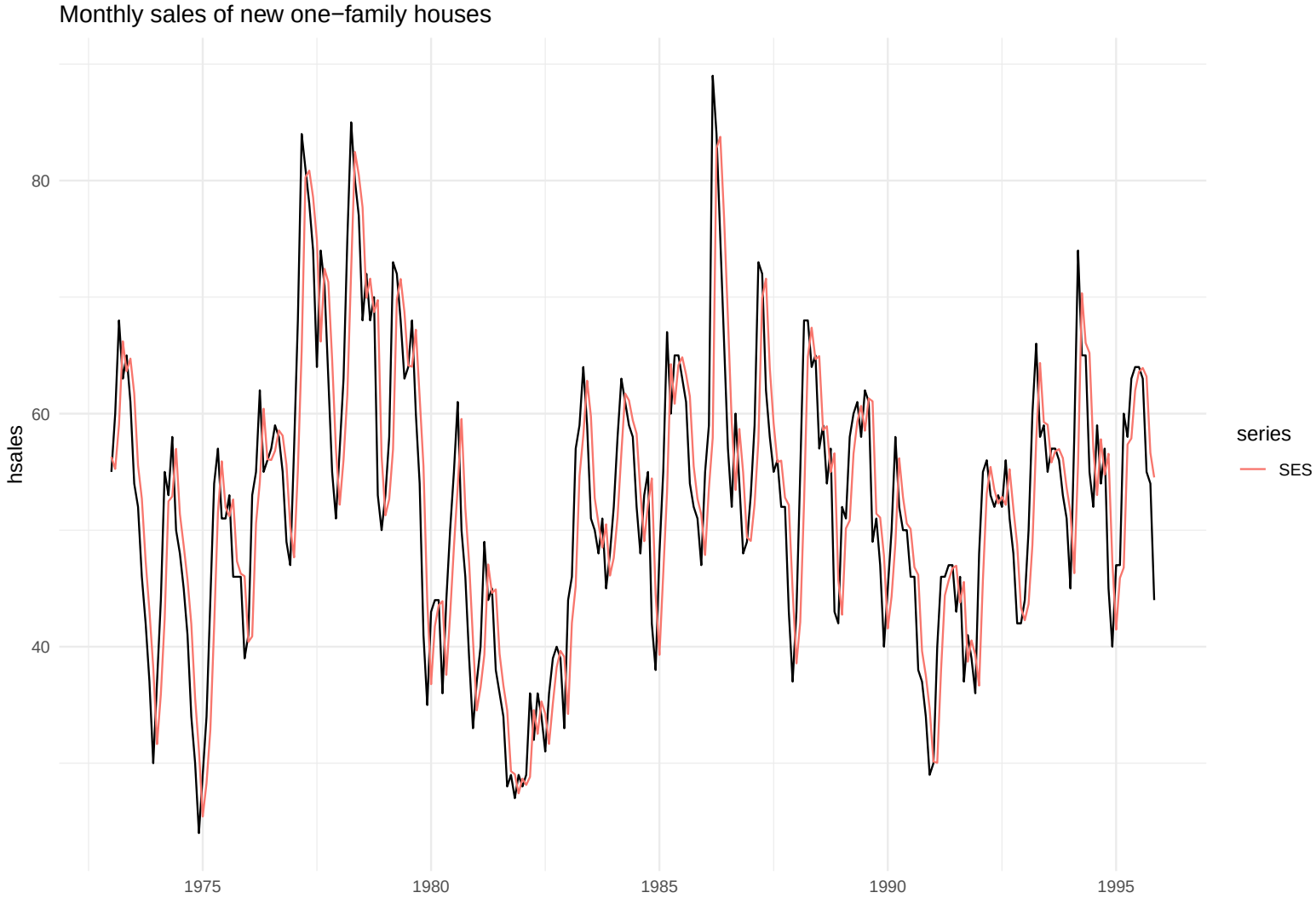
$$p(1-p)^{n-1}$$

Geometric Distribution graph for various values of α

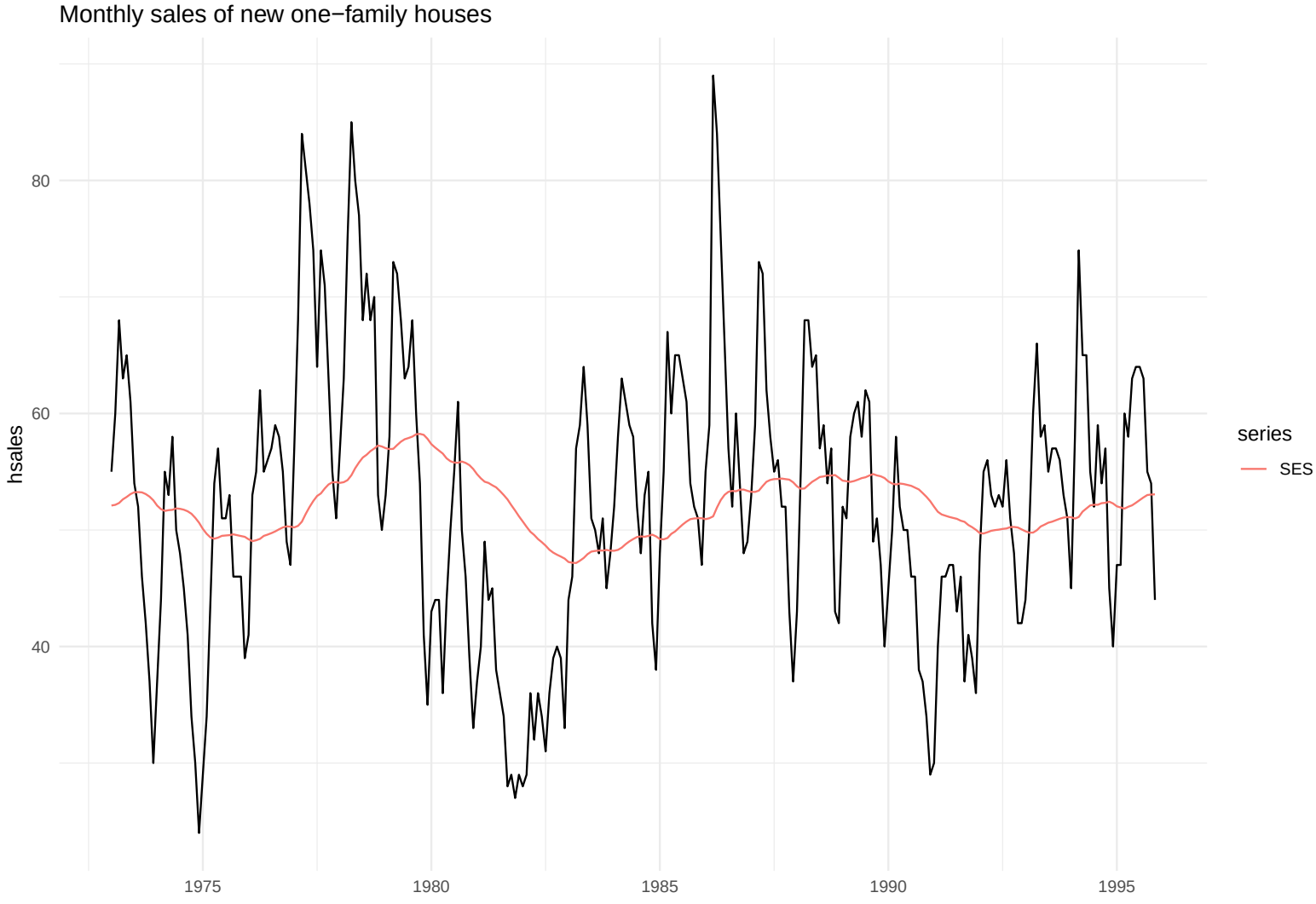


Exploring "Smoothing" aspect

- With $\alpha \approx 1$ [$\alpha = 0.8$], the model is very much ***reactive*** to the most recent observations



- With $\alpha \approx 0$ [$\alpha = 0.02$], the model is much less reactive (***smoother***) to react to change



SES - Mathematical Formulations

- **Weighted average form**

$$\hat{y}_{T+1|T} = \alpha y_T + (1-\alpha)\hat{y}_{T|T-1}$$

This is a *"recursive"* definition for the below

$$\hat{y}_{T+1|T} = \alpha y_T + \alpha(1-\alpha)y_{T-1} + \alpha(1-\alpha)^2 y_{T-2} + \cdots$$

which can also be written as:

$$\hat{y}_{T+1|T} = \sum_{j=0}^{T-1} \alpha(1-\alpha)^j y_{T-j} + (1-\alpha)^T \ell_0$$

- **Component Form**

$$\begin{aligned} \text{Forecast equation} \quad & \hat{y}_{t+h|t} = \ell_t \\ \text{Smoothing equation} \quad & \ell_t = \alpha y_t + (1-\alpha)\ell_{t-1} \end{aligned}$$

where ℓ_t is the ***level*** (*smoothed* value) of the series at time t

Interlude

- The previous method is effective for time series without trend or seasonality
- But what if your time series has trend?

ES_4: Holt Linear Trend Model

Appropriate for time series that can be described with

$$y_t = \beta_0 + \beta_1 t + \epsilon_t$$

where β_1 quantifies the trend

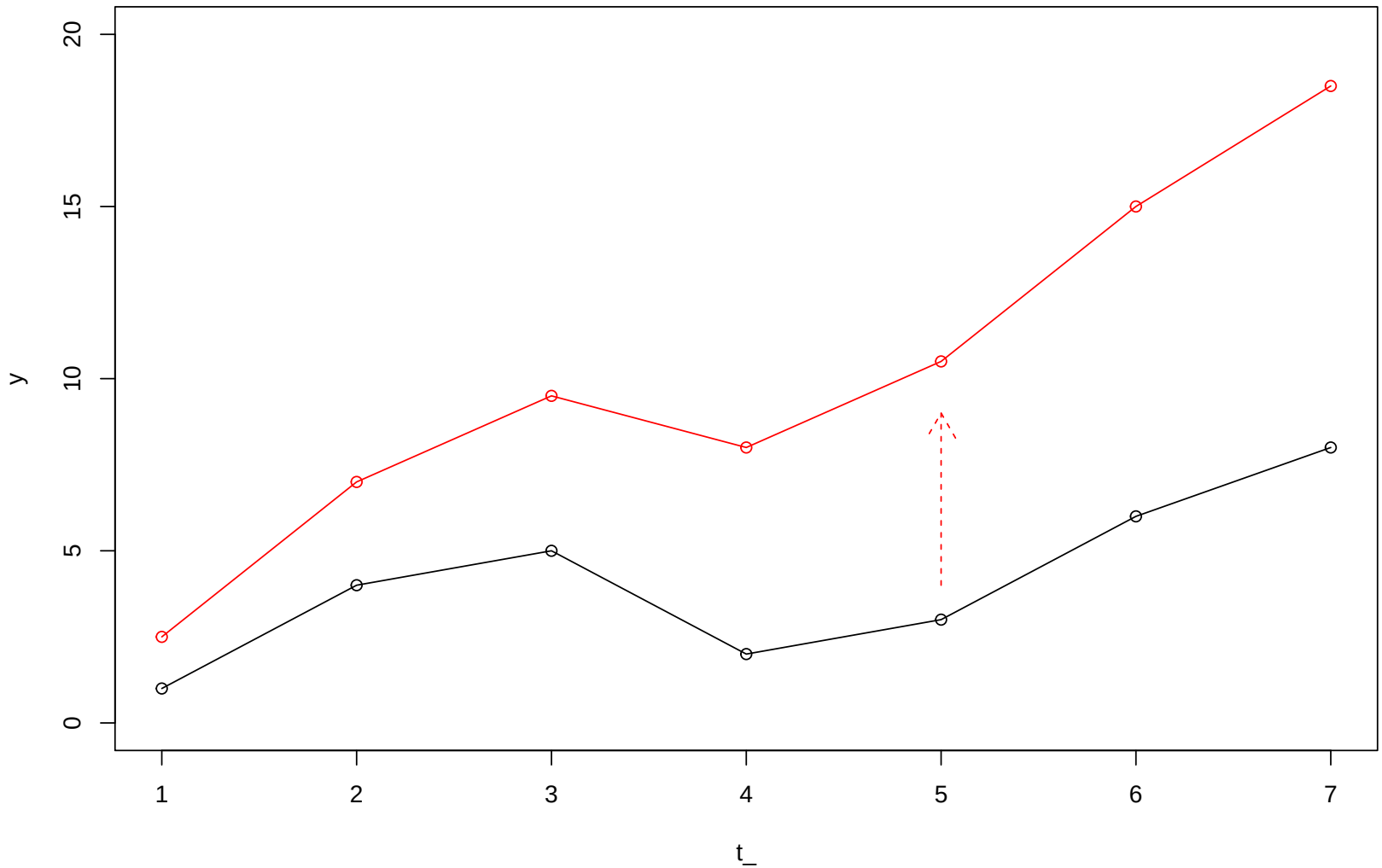
ES_4: Holt Linear Trend Model: Linear Time Series

- We can explicitly convert our previous time series to have trend using this formula:

```
beta_0 <- 0
beta_1 <- 1.5
t_ <- 1:7
( y_1 <- beta_0 + y + beta_1*t_ )
```

```
## [1]  2.5  7.0  9.5  8.0 10.5 15.0 18.5
```

ES_4: Holt Linear Trend Model: Linear Time Series



ES_4: Holt Linear Trend Model: Equations

$$\begin{aligned}
\text{Forecast equation} \quad & \hat{y}_{t+h|t} = L_t + hT_t \\
\text{Level equation} \quad & L_t = \alpha y_t + (1-\alpha)(L_{t-1} + T_{t-1}) \\
\text{Trend equation} \quad & T_t = \beta(L_t - L_{t-1}) + (1-\beta)L_{t-1},
\end{aligned}$$

where

- L_t denotes an estimate of the level of the series at time t,
- T_t denotes an estimate of the trend (slope) of the series at time t,
- α is the smoothing parameter for the level,
- β is the smoothing parameter for the trend, and
- $0 \leq \alpha, \beta \leq 1$

ES_4: Holt Linear Trend Model: Example

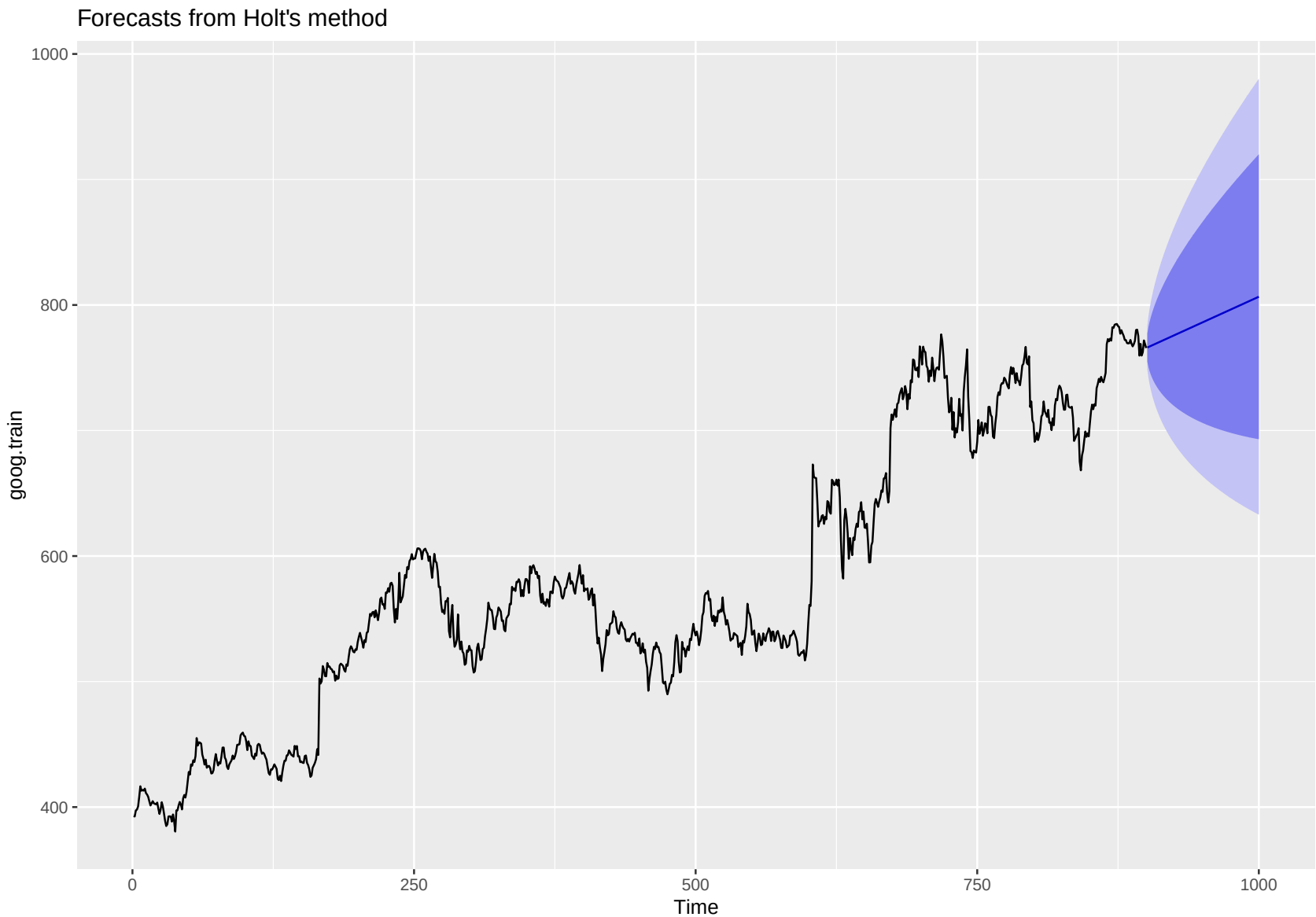
In the FPP2 package there is data set of the Google share price over 1000 days, from 2/25/2013 to 2/13/2017.

We can make a train/test split, and then train Holt's Linear Trend model on the first 900 days, and generate a prediction for the next 100 days:

```
# create training and validation of the Google stock data
goog.train <- window(goog, end = 900)
goog.test <- window(goog, start = 901)
# run holt's model on the training data
holt.goog <- holt(goog.train, h = 100)
```

ES_4: Holt Linear Trend Model: Graph

```
autoplot(holt.goog)
```



ES_4: Holt Linear Trend Model: Accuracy

We can check the accuracy of the model predictions vs. the test data:

```
accuracy(holt.goog,goog.test)[,c("RMSE","MAE","MAPE")]
```

```
##                      RMSE       MAE     MAPE
## Training set  8.795267  5.821057 1.000720
## Test set     16.328680 12.876836 1.646261
```

This indicates a Mean Average Percentage Error (MAPE) of about 1.6% on the test data.

ES_5: Holt-Winters method: incorporating seasonality

The **Holt-Winters** method accomodates both **trend** and **seasonality**.

There are two flavors: **Additive** and **Multiplicative**.

- Additive: use when the magnitude of the seasonal trend **stays the same** throughout the data set
- Multiplicative: use when the magnitude of the seasonality **changes**, based on the level

Each flavor involves four equations:

- One equation for the overall forecast
- Three smoothing equations, one each for Level, Trend, and Seasonality

ES_5: Holt-Winters method - Equations

Additive:

- Forecast: $\hat{y}_{t+h|t} = L_t + hT_t + S_{t-m+h_m^+}$
- Level: $L_t = \alpha(y_t - S_{t-m}) + (1-\alpha)(L_{t-1} + T_{t-1})$
- Trend: $T_t = \beta(L_t - L_{t-1}) + (1-\beta)T_{t-1}$
- Seasonality: $S_t = \gamma(y_t - L_t - T_t) + (1-\gamma)S_{t-m}$

Multiplicative:

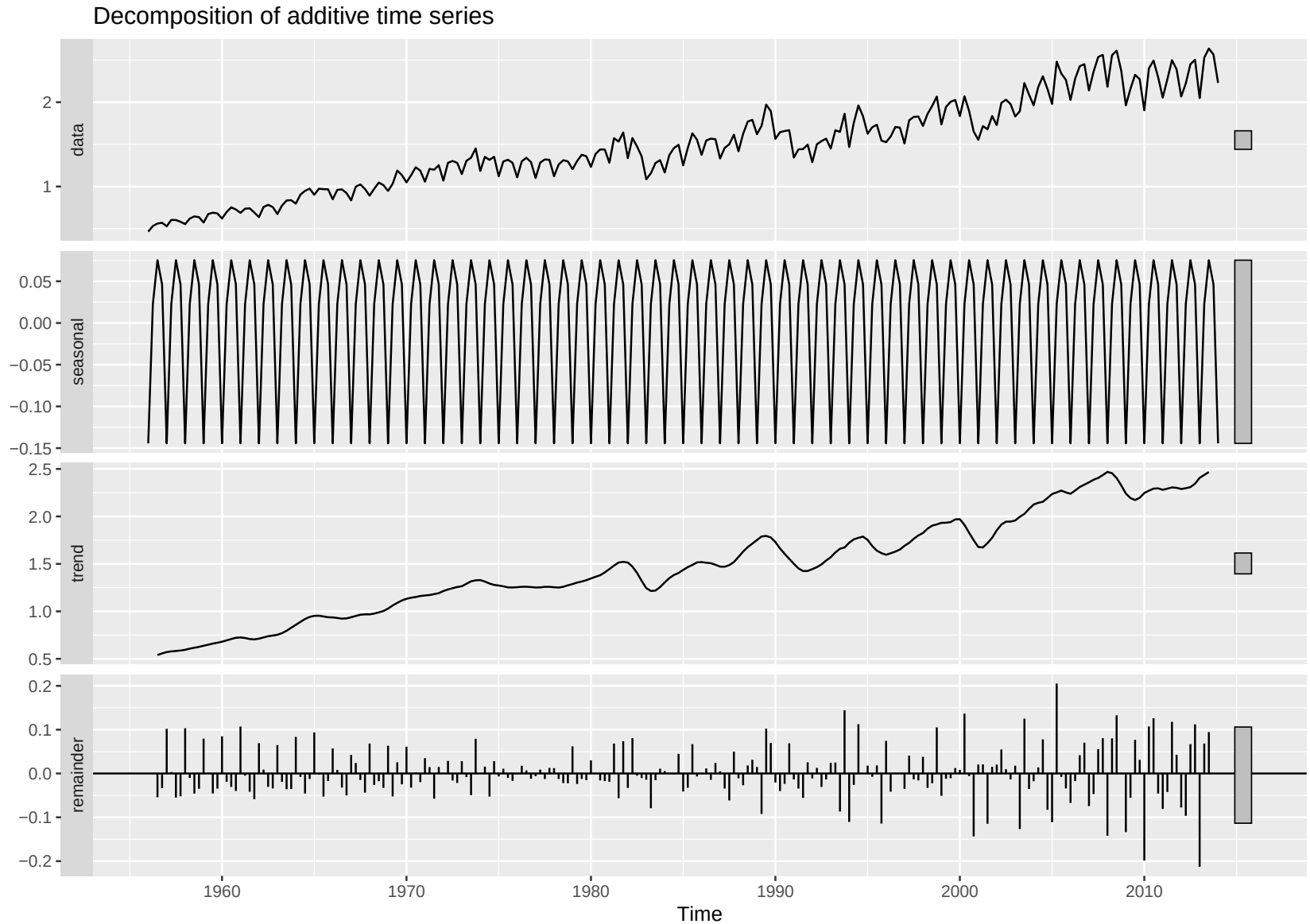
- Forecast: $\hat{y}_{t+h|t} = (L_t + hT_t)S_{t-m+h_m^+}$
- Level: $L_t = \frac{\alpha y_t}{S_{t-m}} + (1-\alpha)(L_{t-1} + T_{t-1})$
- Trend: $T_t = \beta(L_t - L_{t-1}) + (1-\beta)T_{t-1}$
- Seasonality: $S_t = \gamma(\frac{y_t}{L_t + T_t}) + (1-\gamma)S_{t-m}$

where

- m is the number of periods per year (e.g., 4 or 12),
- h is the number of periods ahead to forecast, and
- $h_m^+ = [(h-1) \mod m] + 1$
- $0 \leq \alpha, \beta, \gamma \leq 1$

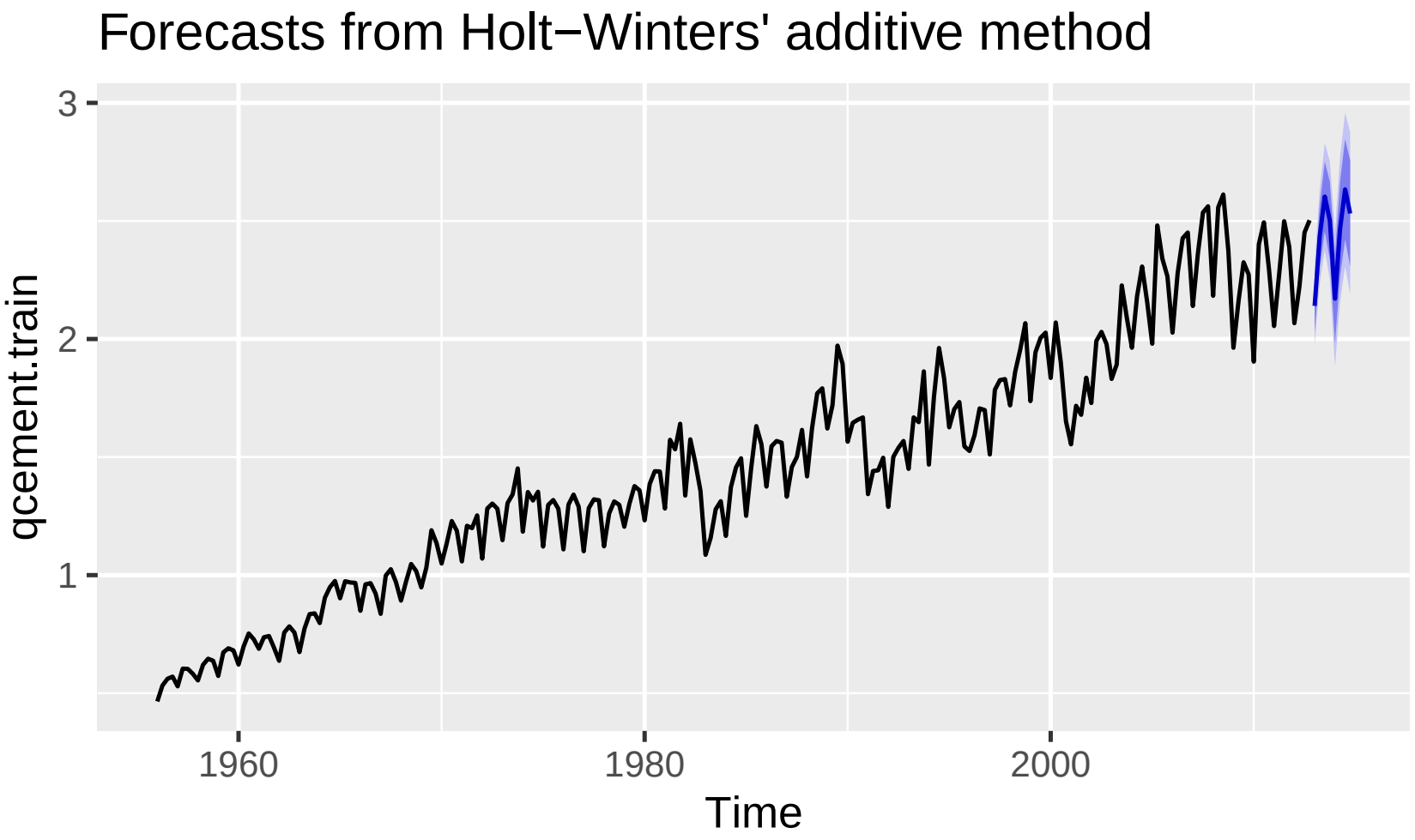
ES_5: Holt-Winters method - Example

The fpp2 package includes a dataset on Australian cement production, `qcement`, which exhibits both trend and seasonality:



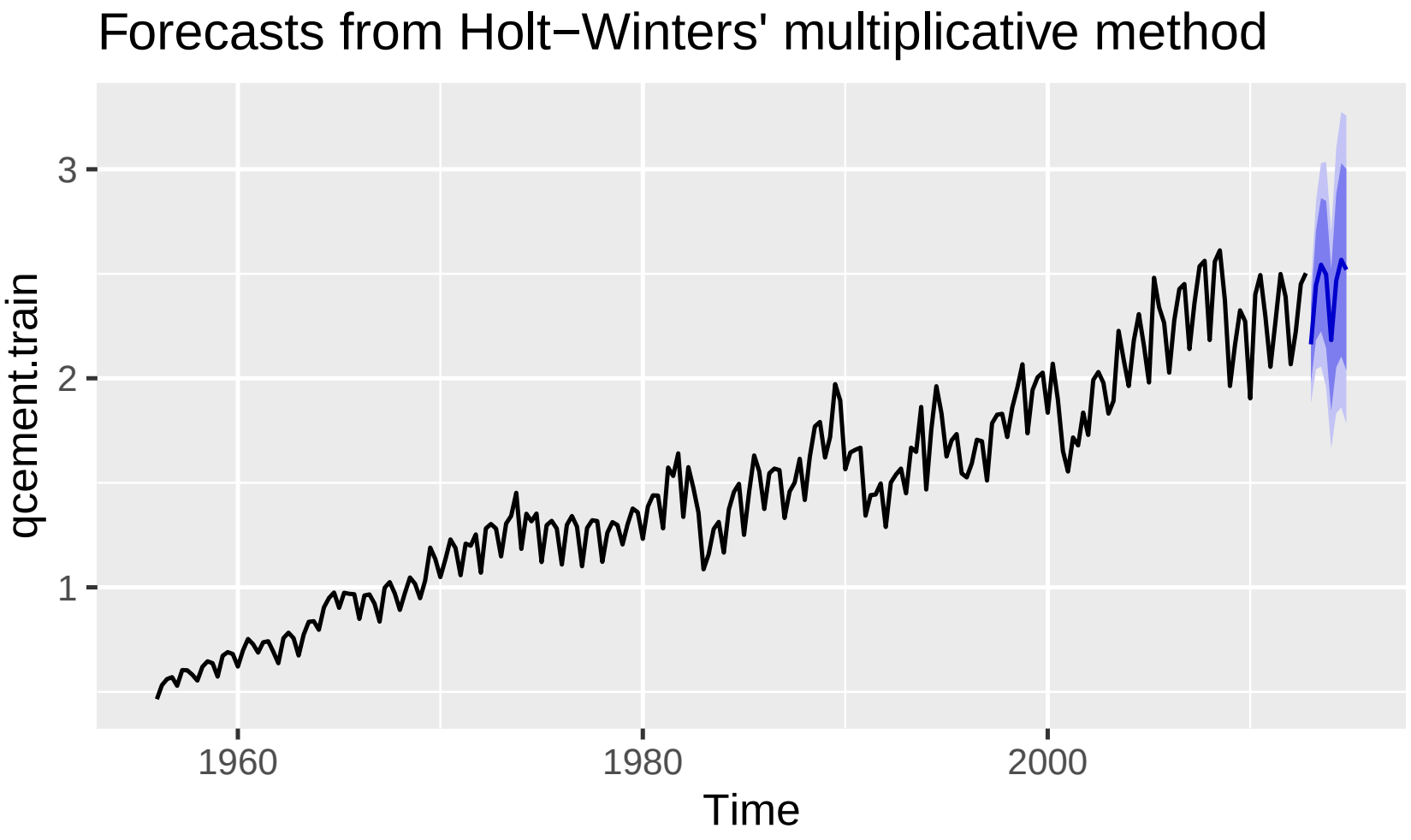
ES_5: Holt-Winters method - Additive Example

```
autoplot(forecast(hw(qcement.train)))
```



ES_5: Holt-Winters method - Multiplicative Example

```
autoplot(forecast(hw(qcement.train,
                     seasonal = "multiplicative")))
```



ES_6: ETS modeling (Innovations state space models) - 1

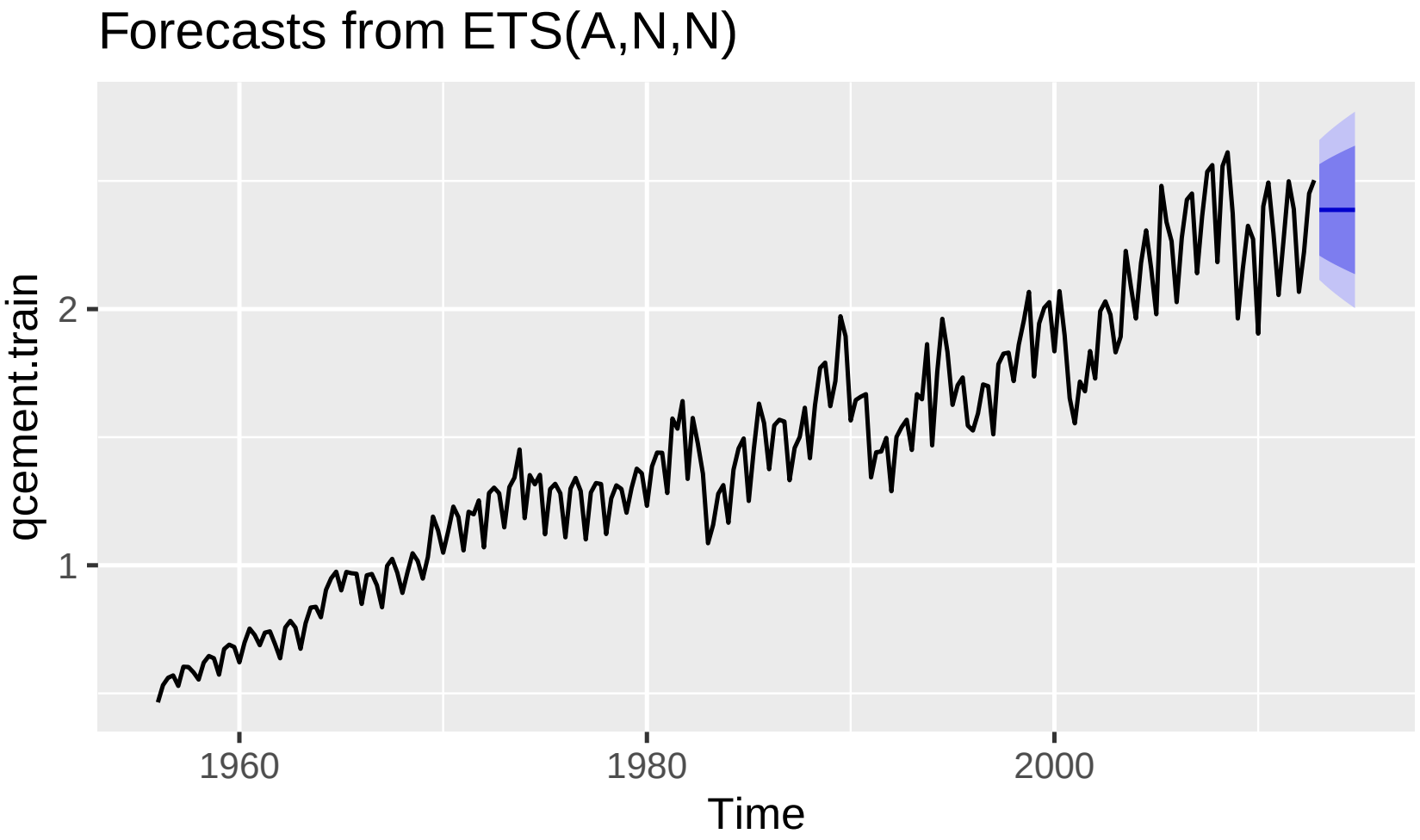
- Exponential time smoothing method discussed so far are good for producing point forecast
- For all practical purposes point forecast is not enough and we need to produce distribution forecast e.g. quantiles
- Statistical methods like ETS state space models are good for generating point forecast and distribution forecast
- Statistical Forecasting Models - State Space Model
  - Generated same point forecast
  - Generates prediction intervals
    - Parameter Estimates, Error Estimates, Error bounds
    - Parametric modelling (distribution assumptions)

ES_6: ETS modeling (Innovations state space models) - 2

- State Space model components
  - Measurement equation (observed)
  - State Equations (Unobserved- level, trend, seasonal)
- Different state space models combinations
  - Additive and multiplicative errors
  - ETS (Error,Trend, Seasonal)
- Examples
  - ETS(A,N,N) - Simple Exponential Smoothing with additive errors
  - ETS(M,N,N) - Simple Exponential Smoothing with multiplicative errors
  - ETS(A,A,N) - Holt's linear method with additive errors
  - ETS(M,A,N) - Holt's linear method with multiplicative errors

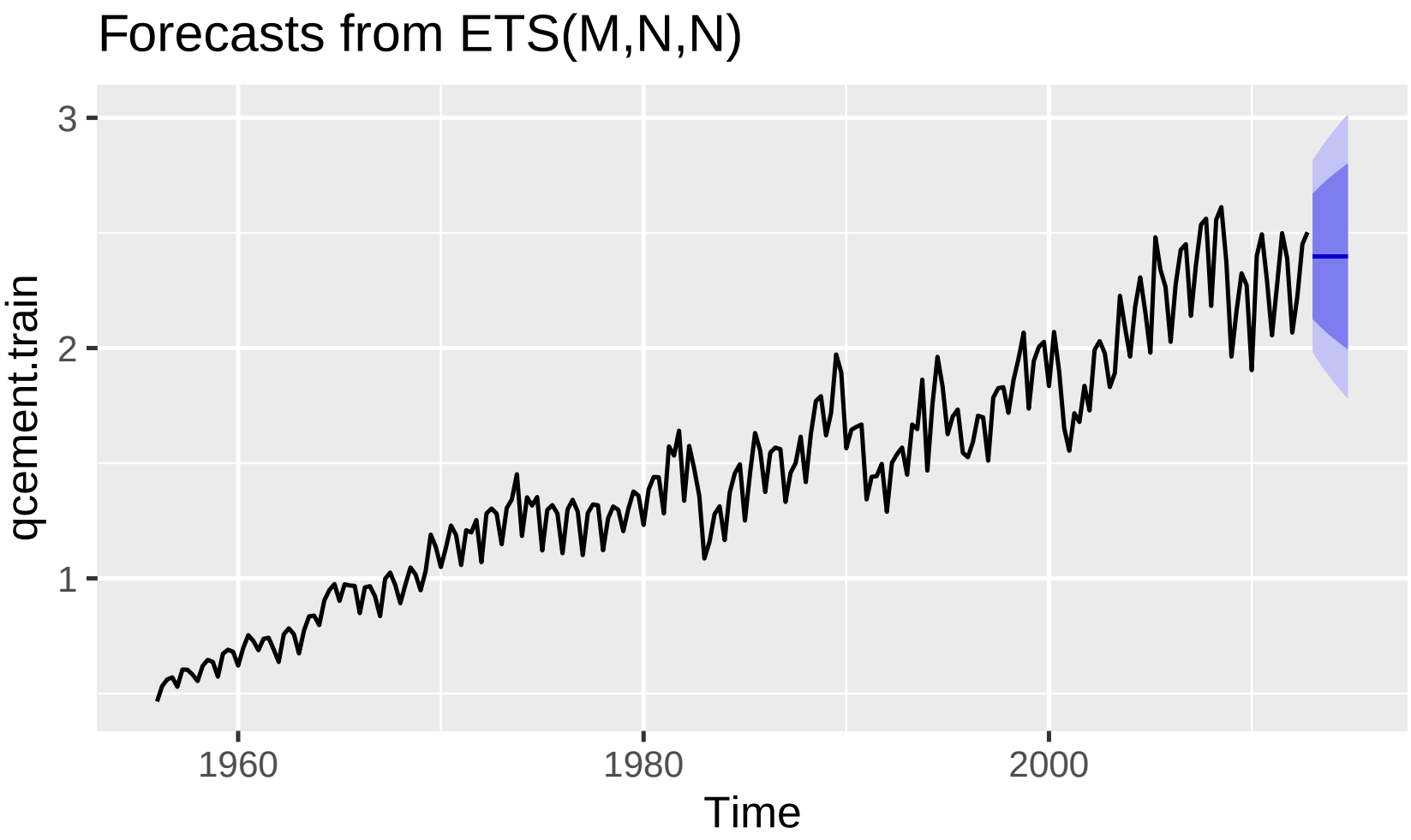
ES_6: ETS(A,N,N) - Simple Exponential Smoothing with additive errors

```
autoplot(forecast(ets(qcement.train, model = "ANN")))
```



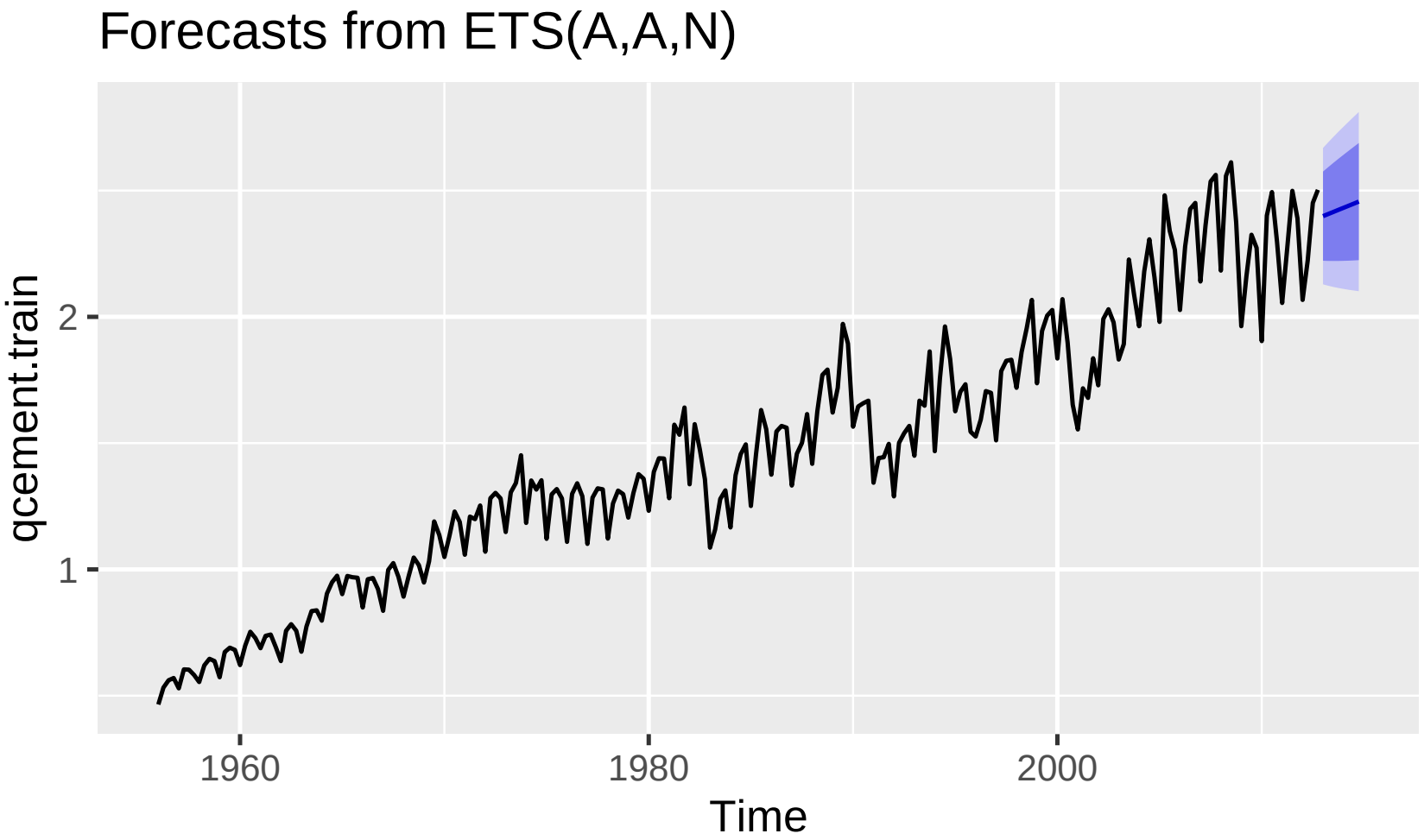
ES_6: ETS(M,N,N) - Simple Exponential Smoothing with multiplicative errors

```
autoplot(forecast(ets(qcement.train, model = "MNN")))
```



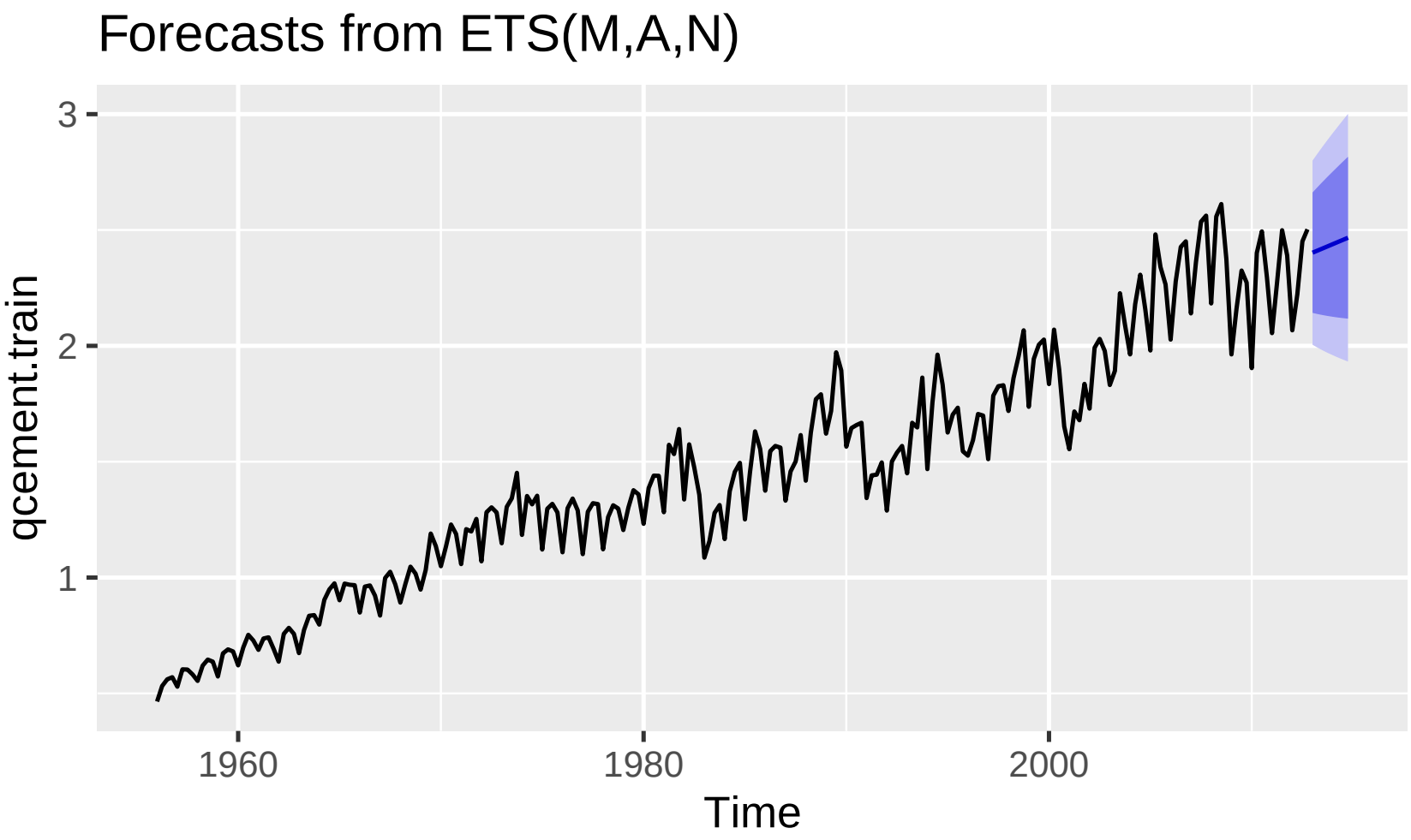
ES_6: ETS(A,A,N) - Holt's Linear Method with additive errors

```
autoplot(forecast(ets(qcement.train, model = "AAN")))
```



ES_6: ETS(M,A,N) - Holt's Linear Method with multiplicative errors

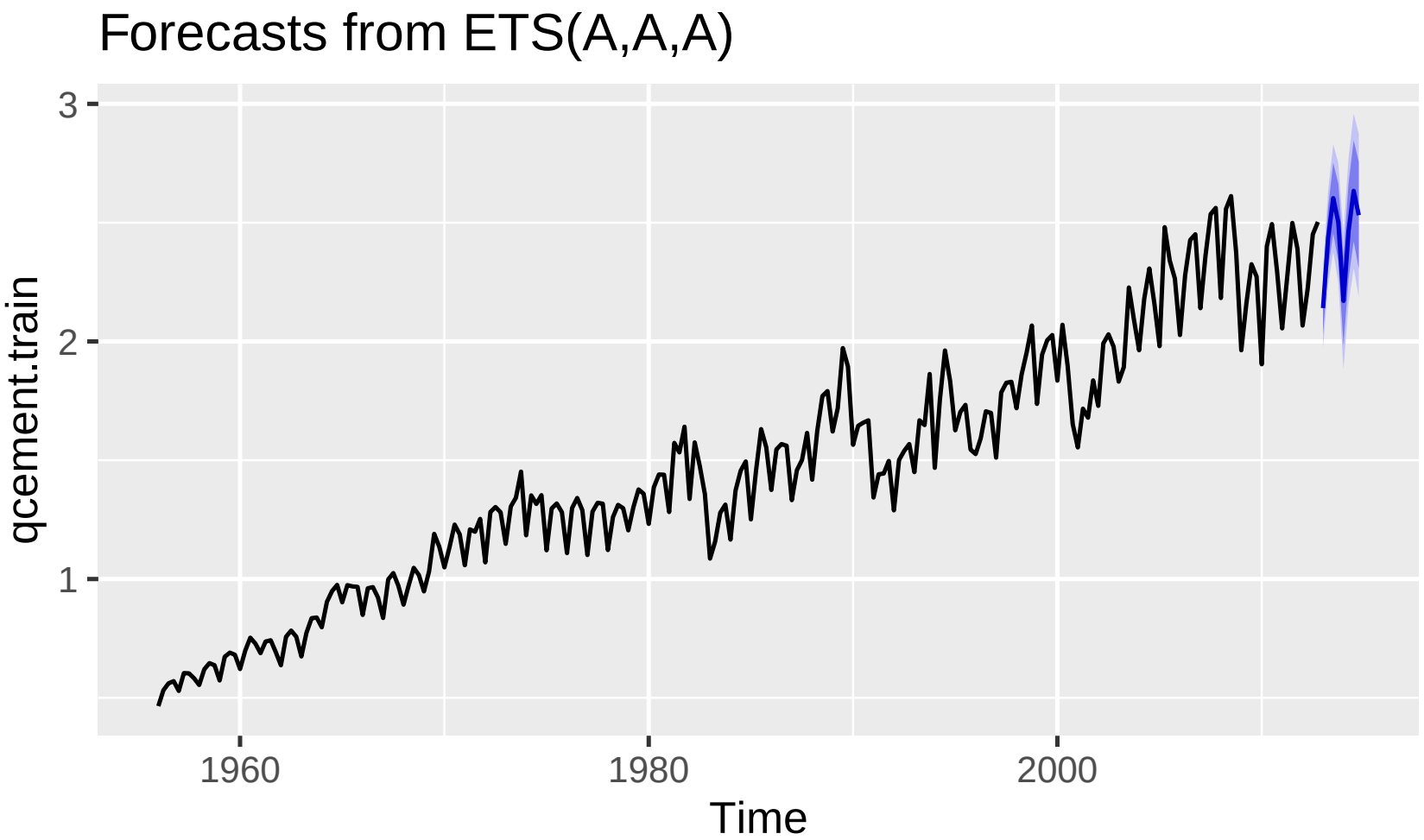
```
autoplot(forecast(ets(qcement.train, model = "MAN")))
```



ES_6: ETS(A,A,A) = Holt-Winters Additive

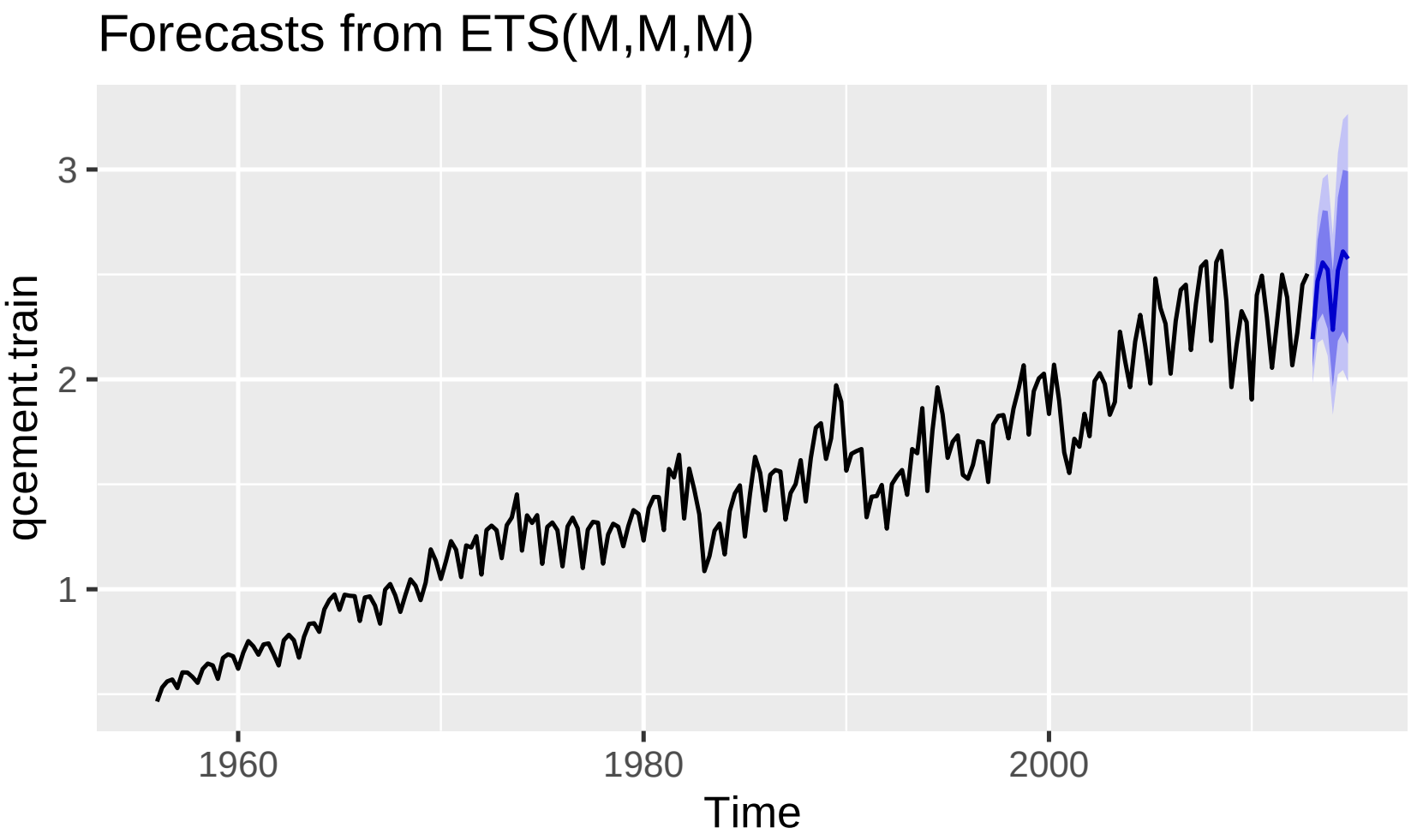
- ETS(A,A,A): Error=Additive, Trend=Additive, Seasonal=Additive

```
autoplot(forecast(ets(qcement.train, model = "AAA")))
```



ES_6: ETS(M,M,M) = Multiplicative Errors, Trend, Seasonality

```
autoplot(forecast(ets(qcement.train, model = "MMM")))
```



Resources & Further Readings

- Exponential Smoothing
  - https://otexts.com/fpp2/expsmooth.html
  - http://uc-r.github.io/ts_exp_smoothing
- Geometric Distribution
  - OpenIntro Statistics, Fourth Edition, David Diez, Mine Cetinkaya-Rundel, Christopher D Barr