Loading data into xts object

FORECASTING PRODUCT DEMAND IN R

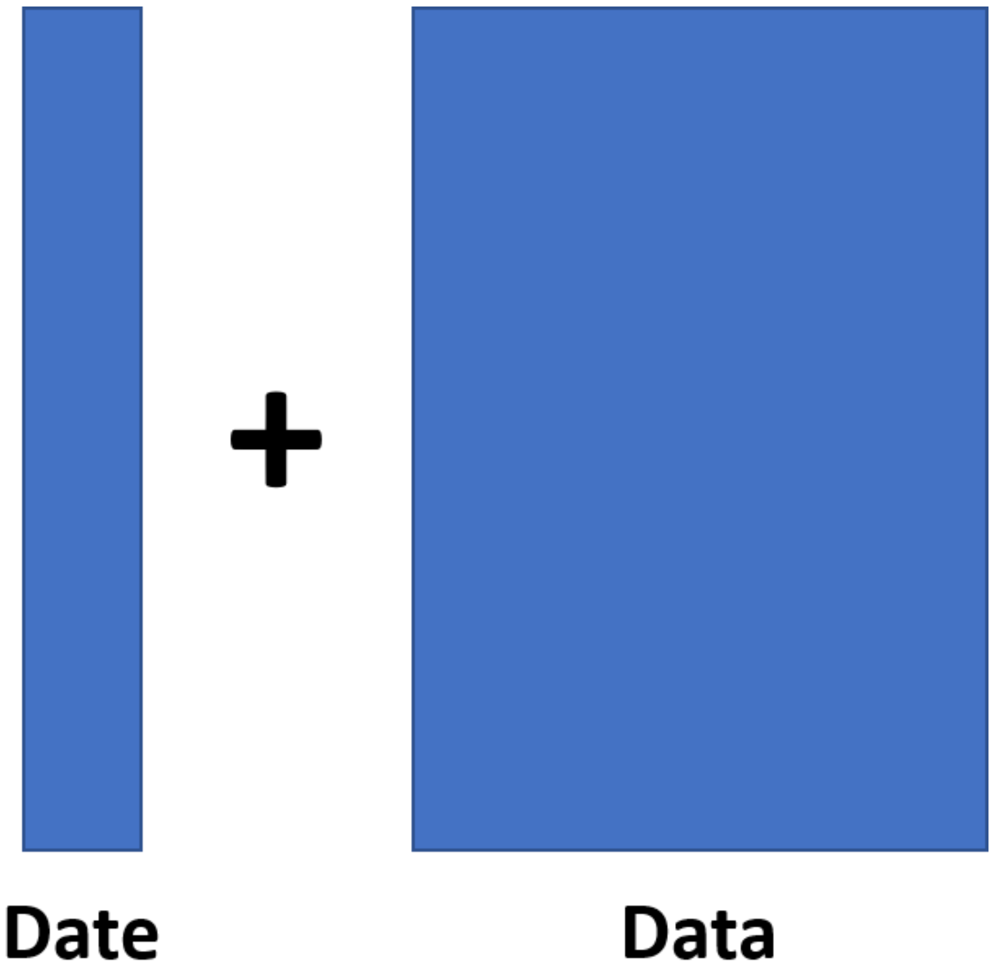
Aric LaBarr, Ph.D.
Senior Data Scientist, Elder Research

xts objects

- eXtensible Time Series object
- Builds upon zoo objects

Loading Data Into xts Object

- Attach a date index on to a data matrix
- Very easy to manipulate!



DataCamp courses about xts

- Manipulating Time Series Data in R with xts & zoo
- Manipulating Time Series Data in R: Case Studies

Loading Data Example

```
dates <- seq(as.Date("2014-01-19"), length = 176,
                                    by = "weeks")
bev_xts <- xts(bev, order.by = dates)
head(bev_xts[,"M.hi"], n = 3)
```

```
            M.hi
2014-01-19  458
2014-01-26  477
2014-02-02  539
```

Let's practice!

FORECASTING PRODUCT DEMAND IN R

ARIMA Time Series 101

FORECASTING PRODUCT DEMAND IN R

Aric LaBarr, Ph.D.
Senior Data Scientist, Elder Research

Other DataCamp time series content

- Time Series with R skill track

What is an ARIMA Model?

- **A**uto **R**egressive Models
- **I**ntegrated
- **M**oving **A**verage

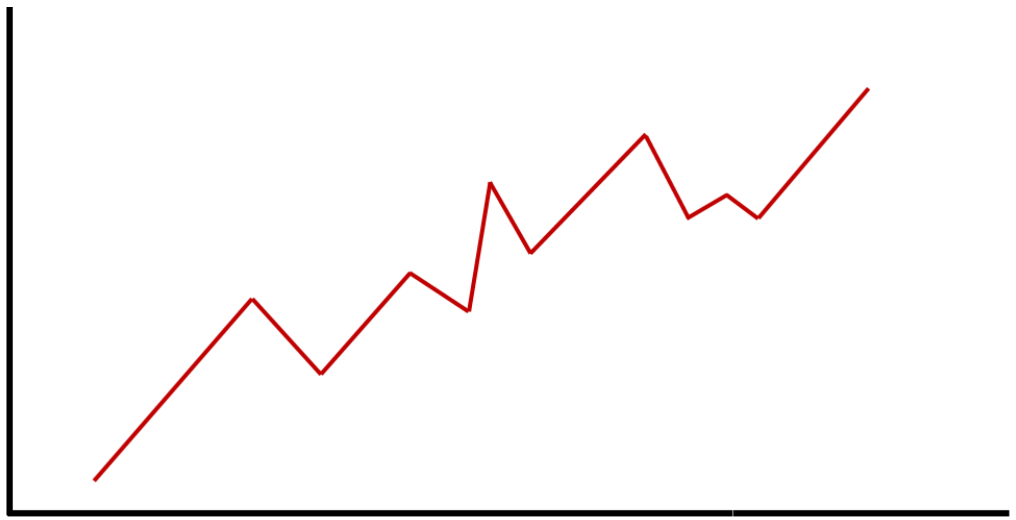
Integrated - Stationarity

- Does your data have a dependence across time?
- How long does this dependence last?
- **Stationarity**
  - Effect of an observation dissipates as time goes on
  - Best long term prediction is the mean of the series
  - Commonly achieved through differencing

Differencing

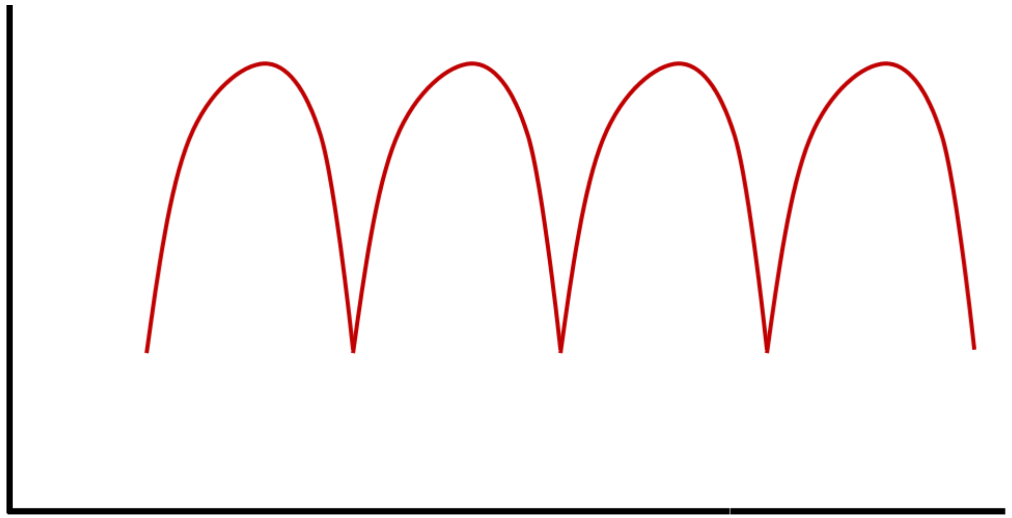
Typically used to remove trend...

$$Y_t - Y_{t-1}$$



... or seasonality

$$Y_t - Y_{t-12}$$



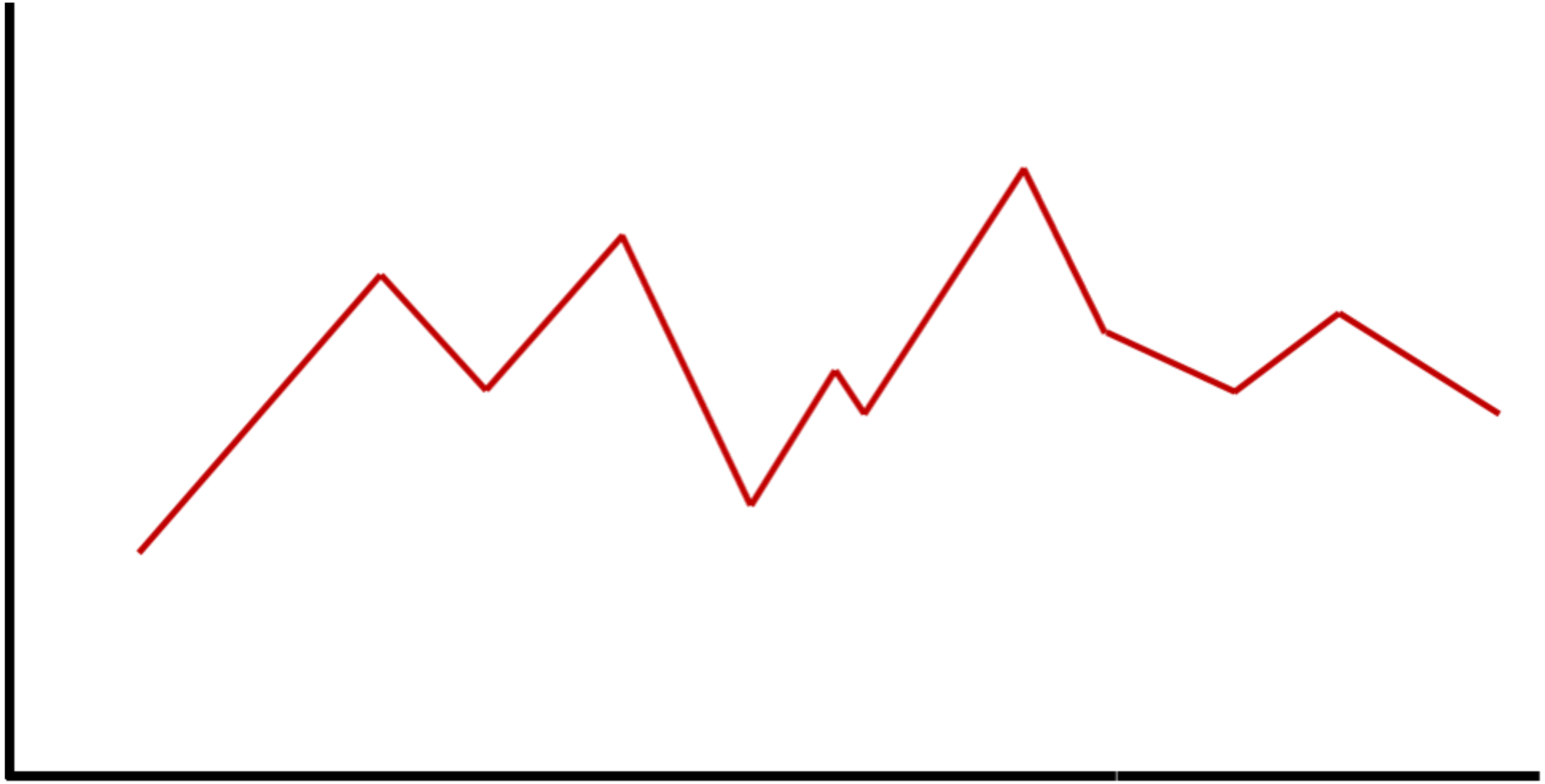
Autoregressive (AR) Piece

- Autoregressive Models
  - Depend only on previous values - called **lags**
  - $Y_t = \omega_0 + \alpha_1 Y_{t-1} + \alpha_2 Y_{t-2} + ... + \varepsilon_t$
  - Long-memory models - effect slowly dissipates

Moving Average (MA) Piece

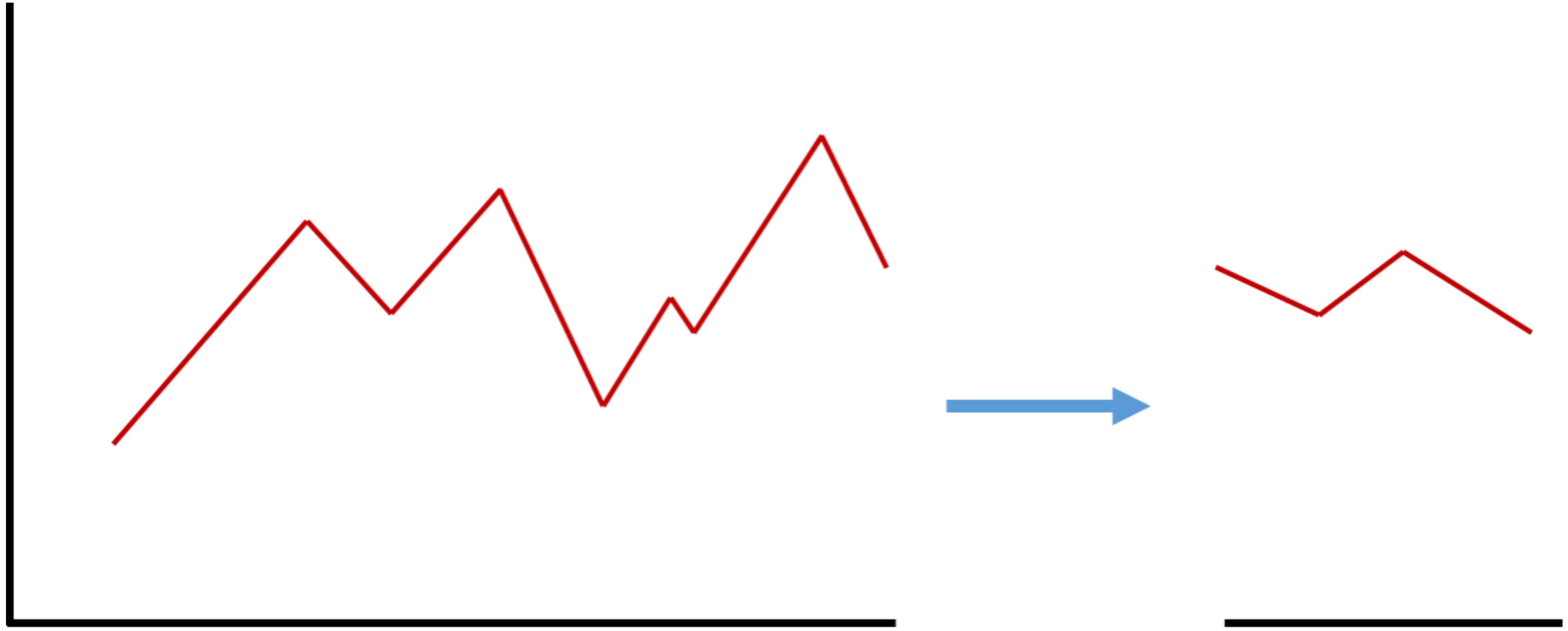
- Moving Average Models
  - Depend only on previous "shocks" or errors
  - $Y_t = \omega_0 + \varepsilon_t + \beta_1 \varepsilon_{t-1} + \beta_2 \varepsilon_{t-2} + \ldots$
  - Short-memory models - effects quickly disappear completely

Training vs. Validation



```
M_t <- bev_xts[,"M.hi"] + bev_xts[,"M.lo"]
```

Training vs. Validation



```
M_t_train <- M_t[index(M_t) < "2017-01-01"]
M_t_valid <- M_t[index(M_t) >= "2017-01-01"]
```

How to Build ARIMA Models?

```
auto.arima(M_t_train)
```

```
Series: M_t_train
ARIMA(4,0,1) with non-zero mean

Coefficients:
         ar1      ar2     ar3     ar4      ma1       mean
      1.3158  -0.5841  0.1546  0.0290  -0.6285  2037.5977
s.e.  0.3199   0.2562  0.1534  0.1165   0.3089    87.5028

sigma^2 estimated as 67471:  log likelihood=-1072.02
AIC=2158.05   AICc=2158.81   BIC=2179.31
```

Let's practice!

FORECASTING PRODUCT DEMAND IN R

Forecasting

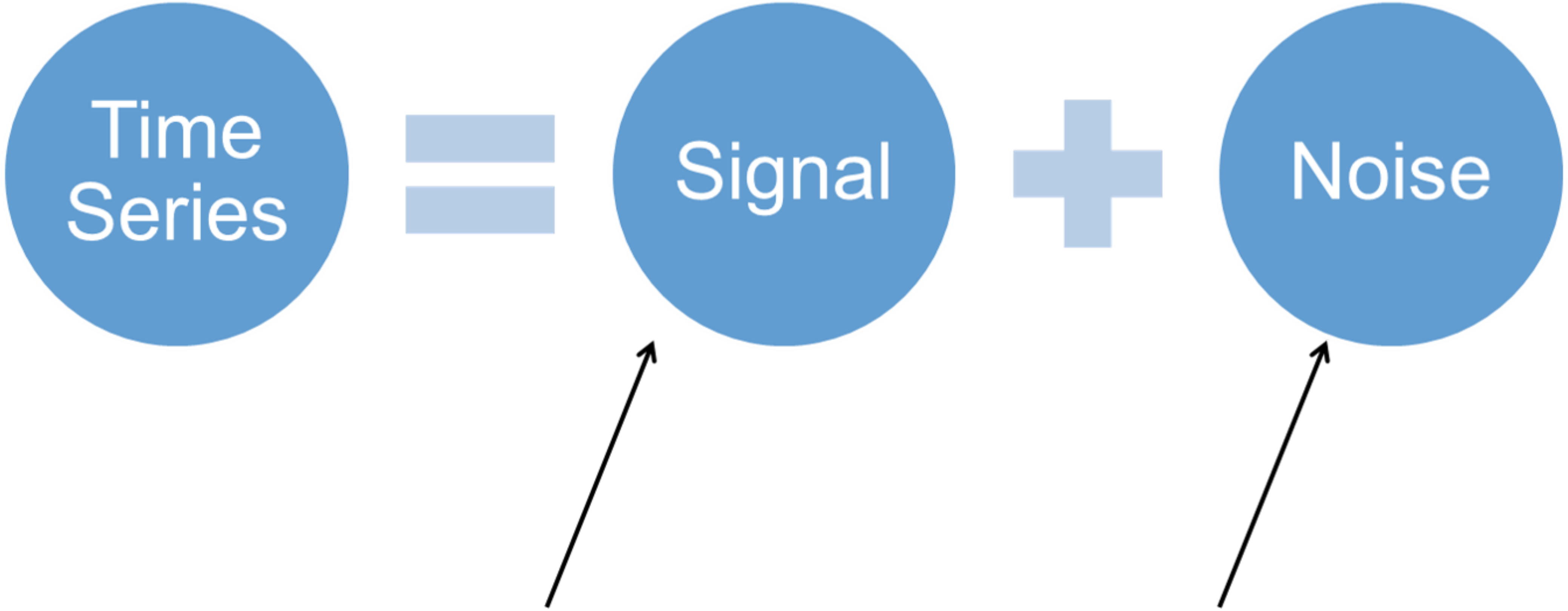
FORECASTING PRODUCT DEMAND IN R

Aric LaBarr, Ph.D.
Senior Data Scientist, Elder Research

Forecasting

- Goal of most time series models!
- Models use past values or "shocks" to predict the future
- Pattern recognition followed by pattern repetition

Forecasting

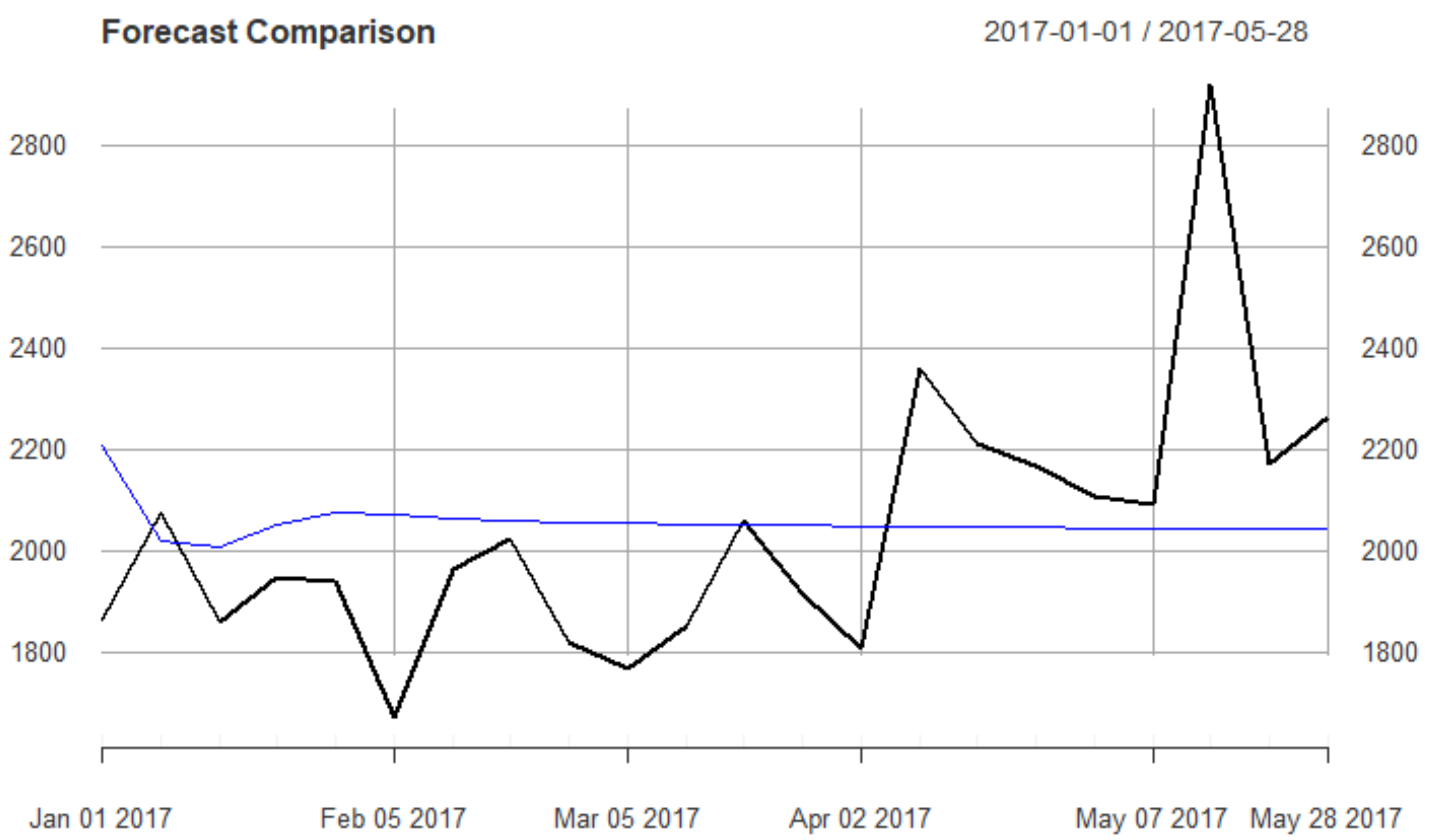


Forecasts extrapolate signal portion of model.

Confidence intervals account for uncertainty.

Forecasting Example

```
forecast_M_t <- forecast(M_t_model, h = 22)
for_dates <- seq(as.Date("2017-01-01"), length = 22,
                                         by = "weeks")
for_M_t_xts <- xts(forecast_M_t$mean, order.by = for_dates)
plot(M_t_valid, main = 'Forecast Comparison')
lines(for_M_t_xts, col = "blue")
```


Forecast Comparison
2017-01-01 / 2017-05-28
2800
2600
2400
2200
2000
1800
Jan 01 2017
Feb 05 2017
Mar 05 2017
Apr 02 2017
May 07 2017
May 28 2017

How to Evaluate Forecasts?

- 2 Common Measures of Accuracy:
  1. **M**ean **A**bsolute **E**rror (MAE)

$$\frac{1}{n}\sum_{i=1}^{n}|Y_t - \hat{Y}_t|$$

  2. **M**ean **A**bsolute **P**ercentage **E**rror (MAPE)

$$\frac{1}{n}\sum_{i=1}^{n}|\frac{Y_t - \hat{Y}_t}{Y_t}| \times 100$$

MAE and MAPE Example

```
for_M_t <- as.numeric(forecast_M_t$mean)
v_M_t <- as.numeric(M_t_valid)
MAE <- mean(abs(for_M_t - v_M_t))
MAPE <- 100*mean(abs((for_M_t - v_M_t)/v_M_t))
print(MAE)
```

```
[1] 198.7976
```

```
print(MAPE)
```

```
[1] 9.576247
```

Let's practice!

FORECASTING PRODUCT DEMAND IN R

Price elasticity

FORECASTING PRODUCT DEMAND IN R

Aric LaBarr, Ph.D.
Senior Data Scientist, Elder Research

Price vs. Demand

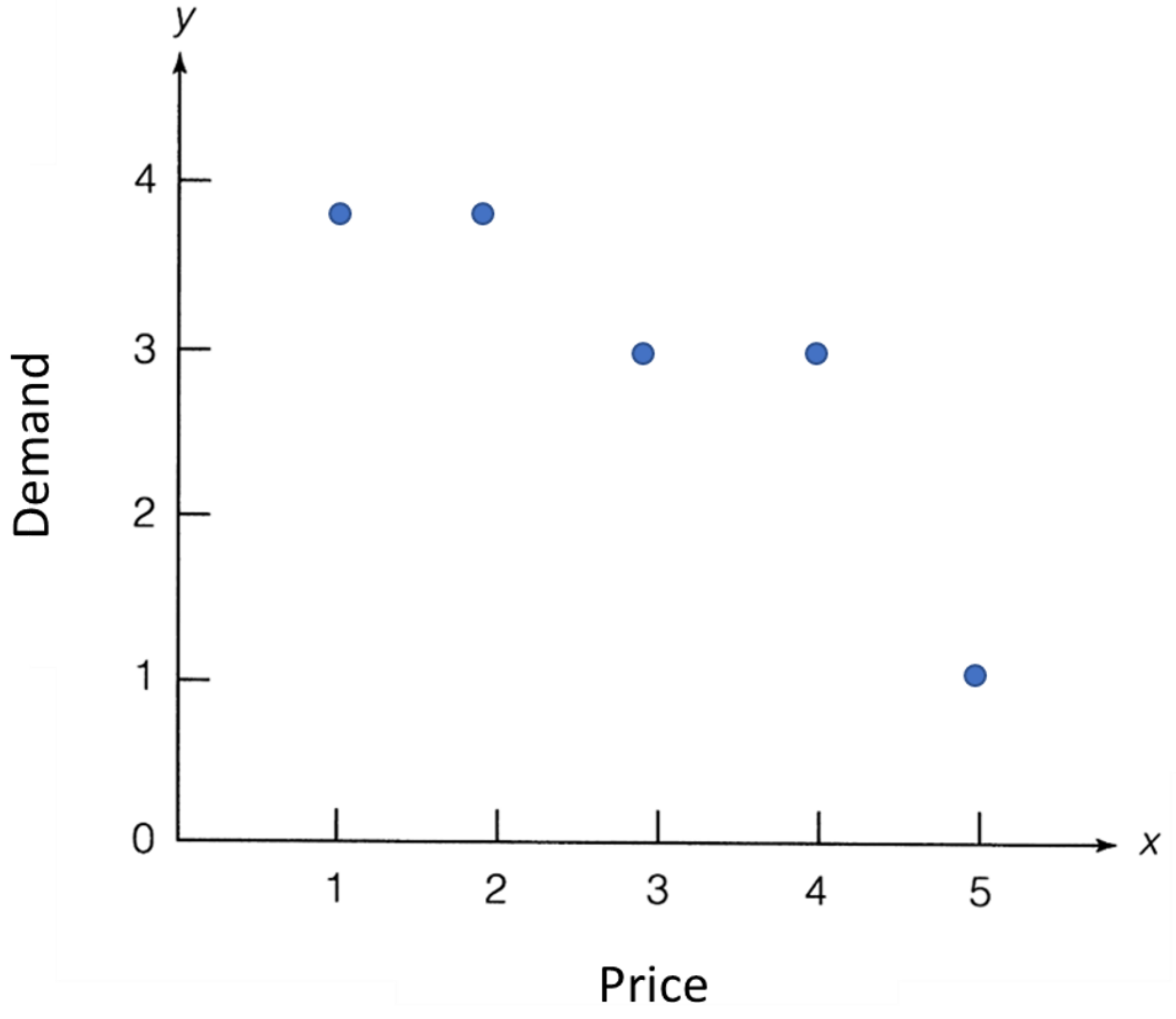
- **Price elasticity** is the economic measure of how much demand "reacts" to changes in price
- As price changes, it is expected that demand changes as well, but how much?

$$\text{Price Elasticity} = \frac{\%\text{Change in Demand}}{\%\text{Change in Price}}$$

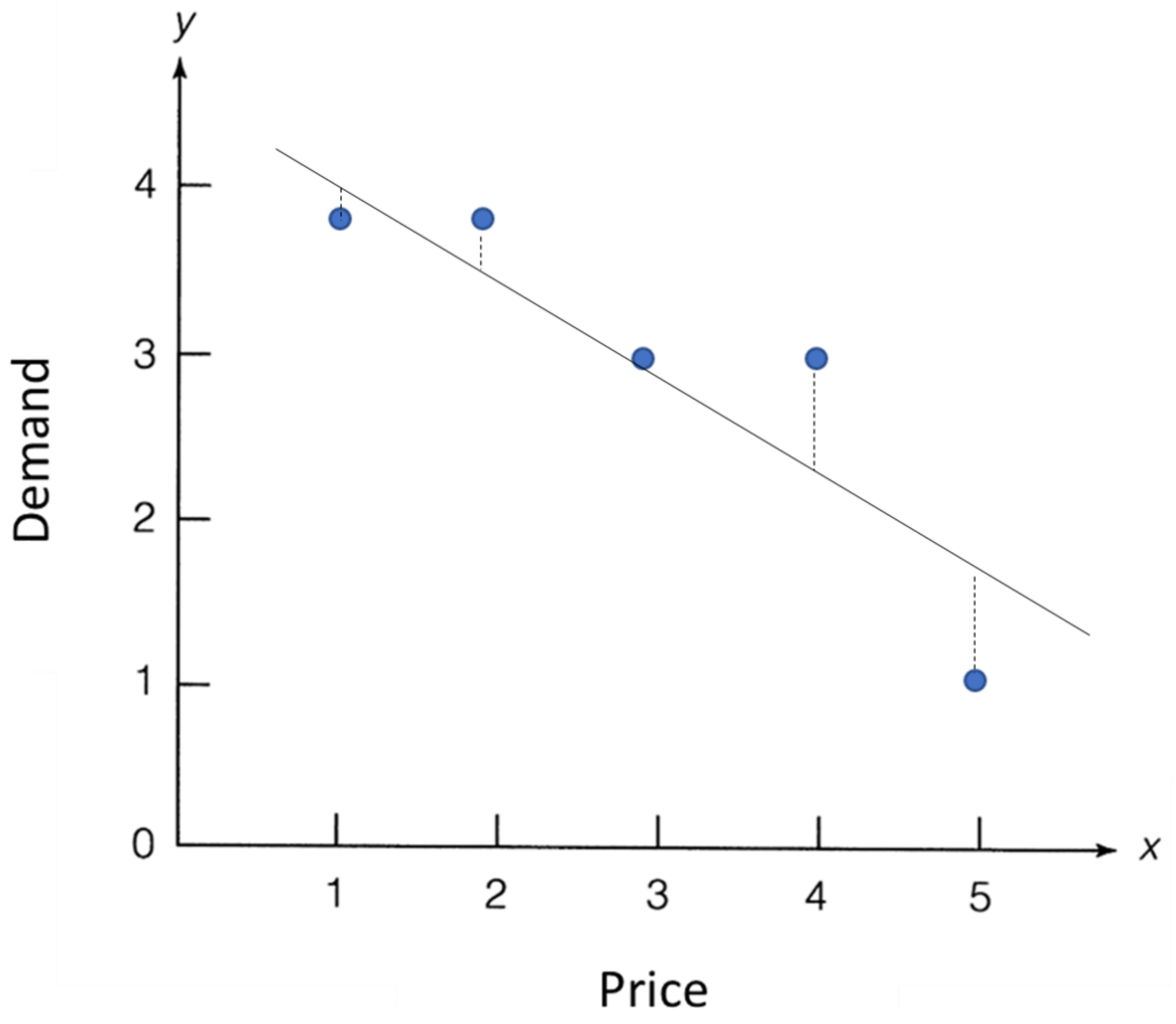
Elastic vs. Inelastic

- **Elastic** products have % changes in demand larger than the % change in price $(\text{Price Elasticity} > 1)$
- **Inelastic** products have % changes in demand smaller than the % change in price $(\text{Price Elasticity} < 1)$
- **Unit elastic** products have % changes in demand equal to the % change in price $(\text{Price Elasticity} = 1)$

Linear Regression



Linear Regression



Price Elasticity Example

```
M_hi <- as.vector(bev_xts_train[,"M.hi"])

M_hi_p <- as.vector(bev_xts_train[,"M.hi.p"])
M_hi_train <- data.frame(log(M_hi), log(M_hi_p))

colnames(M_hi_train) <- c("log_sales", "log_price")
model_M_hi <- lm(log_sales ~ log_price, data = M_hi_train)
```

```
Coefficients:
(Intercept)  log_price
     8.9907      -0.7138
```

Let's practice!

FORECASTING PRODUCT DEMAND IN R

Seasonal / holiday / promotional effects

FORECASTING PRODUCT DEMAND IN R

Aric LaBarr, Ph.D.
Senior Data Scientist, Elder Research

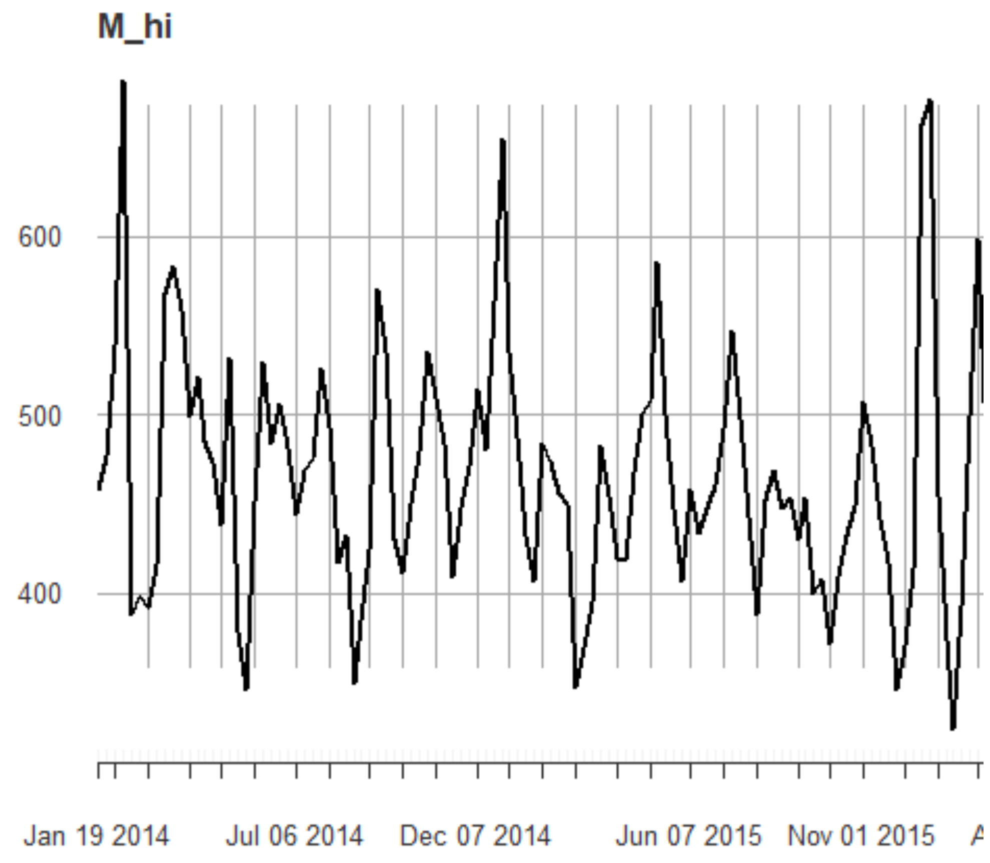
Influencers of Demand

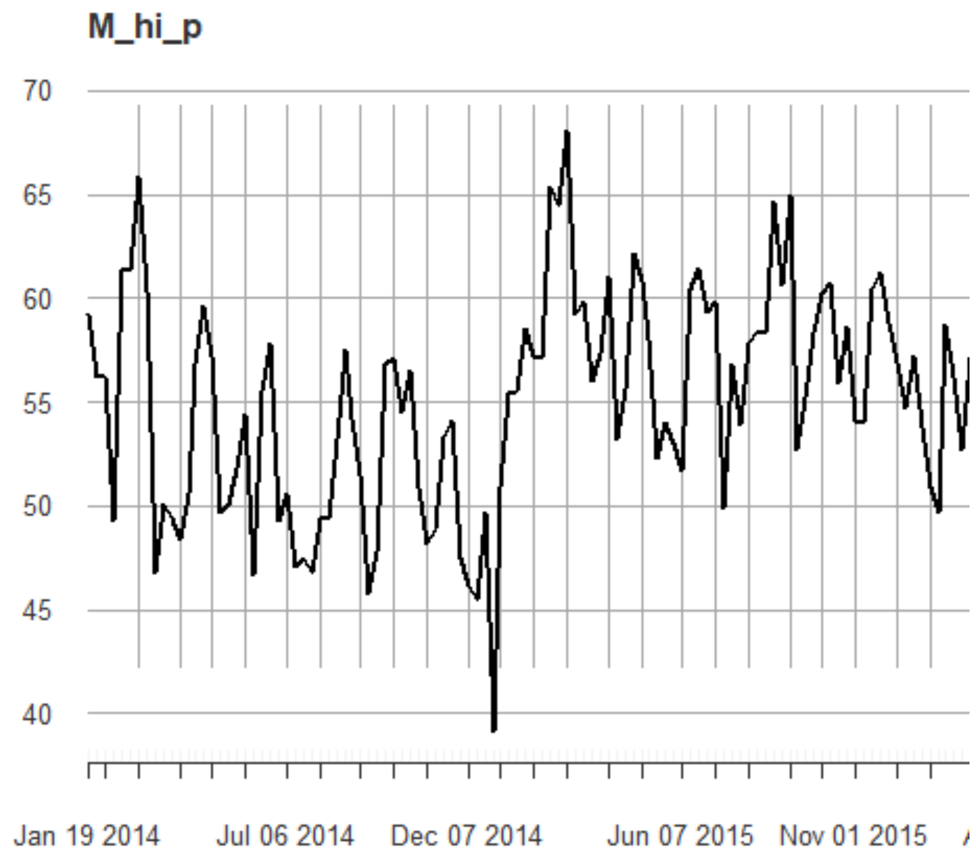
- Seasonal effects
  - Examples: Winter coats, bathing suits, school supplies, etc.
- Holiday effects
  - Examples: Retail sales, holiday decorations, candy, etc.
- Promotion effects
  - Examples: Digital marketing, shelf optimization, etc.

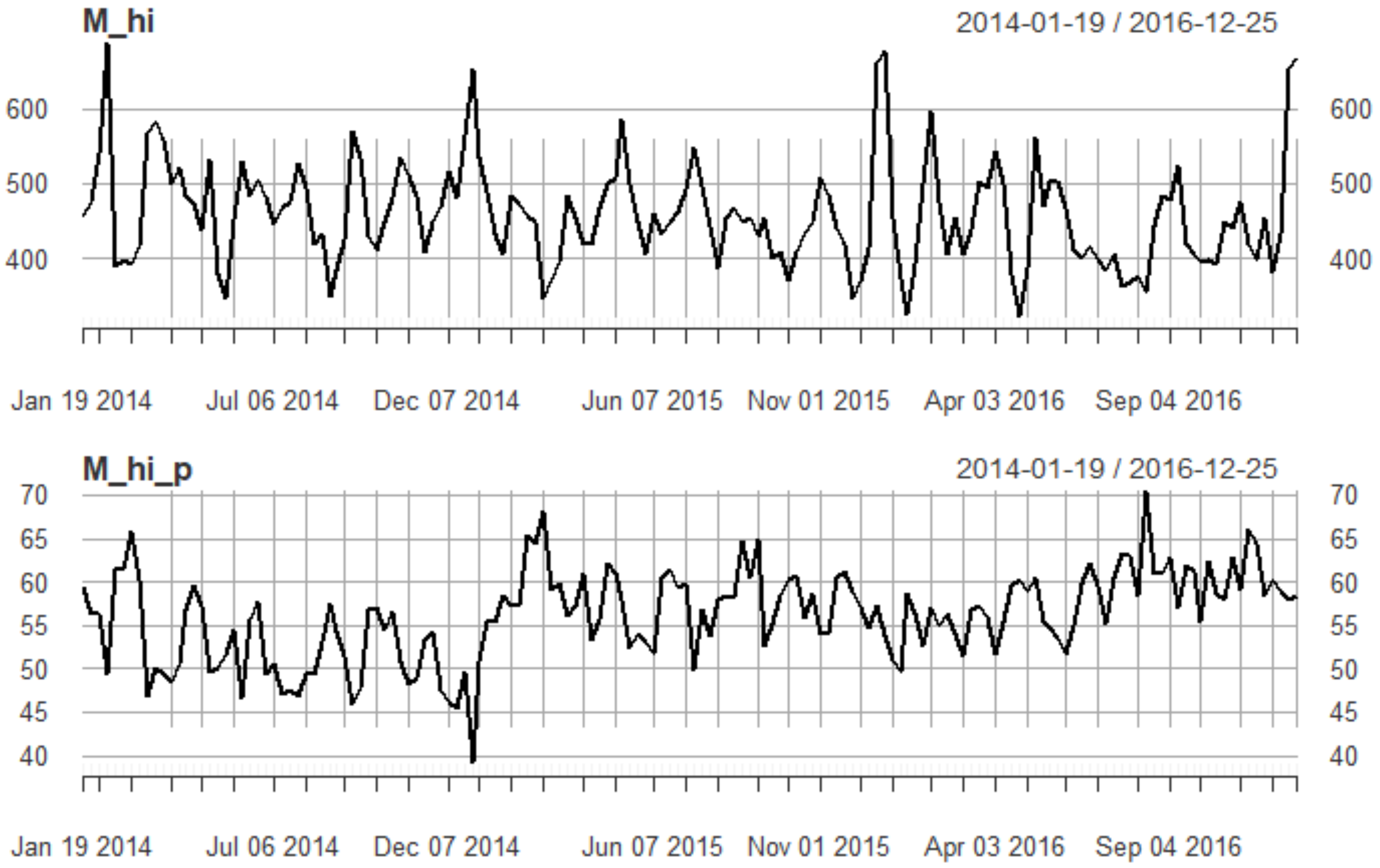
Seasonal / Holiday / Promotion?

```
plot(M_hi)
```

```
plot(M_hi_p)
```






M_hi
2014-01-19 / 2016-12-25
600
500
400
Jan 19 2014
Jul 06 2014
Dec 07 2014
Jun 07 2015
Nov 01 2015
Apr 03 2016
Sep 04 2016
M_hi_p
2014-01-19 / 2016-12-25
70
65
60
55
50
45
40
Jan 19 2014
Jul 06 2014
Dec 07 2014
Jun 07 2015
Nov 01 2015
Apr 03 2016
Sep 04 2016

Linear Regression! Again...

- Linear regression helps us evaluate the relationship between many factors and demand, not just price.
- Add seasonal, holiday, and promotion effects to previous regression!
- Any of these effects **statistically significant**?
  - Are the effects due to random chance or not?

Creating Effects Example

```
v.dates <- as.Date(c("2014-02-09", "2015-02-08", "2016-02-07"))
valentine <- as.xts(rep(1, 3), order.by = v.dates)
dates_train <- seq(as.Date("2014-01-19"), length = 154, by = "weeks")
valentine <- merge(valentine, dates_train, fill = 0)
head(valentine, n = 5)
```

```
           valentine
2014-01-19         0
2014-01-26         0
2014-02-02         0
2014-02-09         1
2014-02-16         0
```

Adding Effects Example

```
M_hi_train <- data.frame(M_hi_train, as.vector(valentine))

model_M_hi_full <- lm(log_sales ~ log_price + valentine,
                      data = M_hi_train)
summary(model_M_hi_full)
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  8.93102    0.44693  19.983  < 2e-16 ***
log_price   -0.70010    0.11103  -6.306    3e-09 ***
valentine    0.22942    0.07547   3.040  0.00279 **

Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1
```

Let's practice!

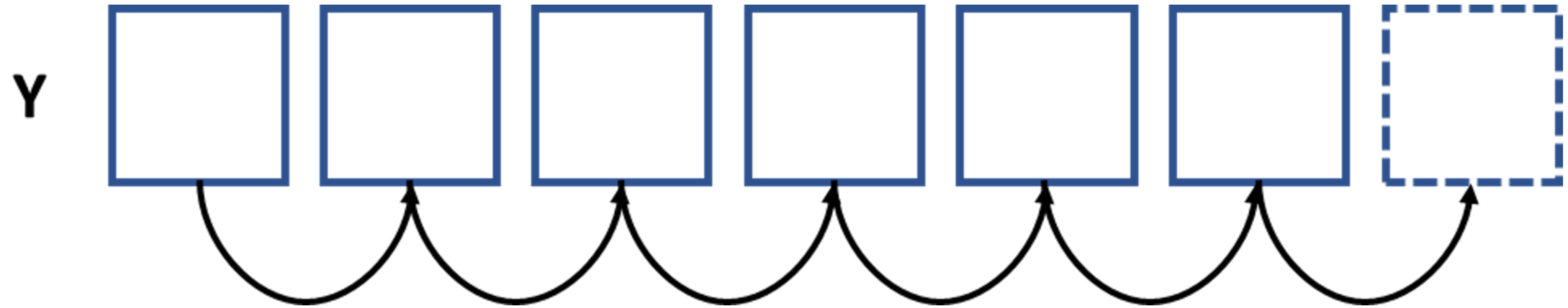
FORECASTING PRODUCT DEMAND IN R

Forecasting with regression

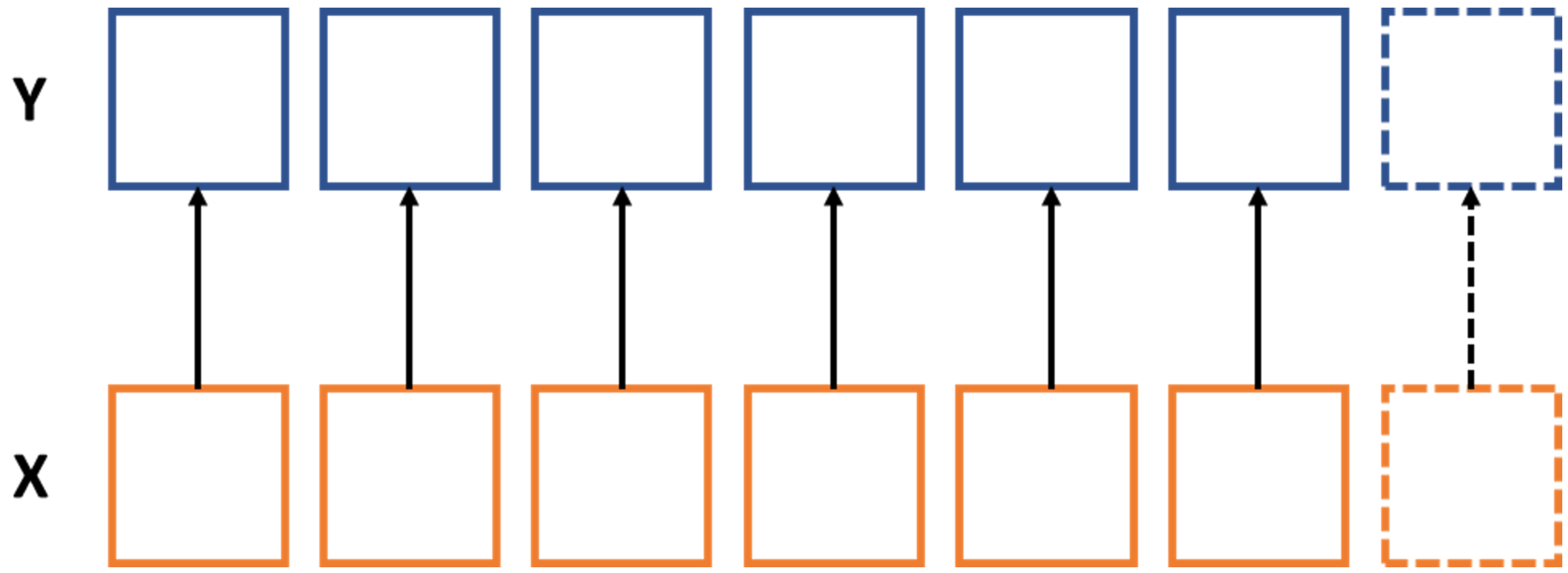
FORECASTING PRODUCT DEMAND IN R

Aric LaBarr, Ph.D.
Senior Data Scientist, Elder Research

Forecasting with Time Series



Forecasting with Regression



Future Input Variables

- How to "predict" future input variables?
  - Holidays and Promotions: NO WORRIES - we know these ahead of time
- Prices - Possible problem!
  - Prices set ahead of time (our assumption)
  - Forecast future prices with time series!

Future Input Variables Example

```
v.dates_v <- as.Date("2017-02-12")
valentine_v <- as.xts(1, order.by = v.dates_v)
dates_valid <- seq(as.Date("2017-01-01"), length = 22,
                                          by = "weeks")
valentine_v <- merge(valentine_v, dates_valid, fill = 0)
l_M_hi_p_valid <- log(bev_xts_valid[,"M.hi.p"])
model_M_valid <- data.frame(as.vector(l_M_hi_p_valid),
                            as.vector(valentine_v))
colnames(model_M_valid) <- c("log_price", "valentine")
```

Future Regression Example

```
pred_M_hi <- predict(model_M_hi_full, model_M_valid)
head(pred_M_hi)
```

```
       1        2        3        4        5        6
6.128652 6.129163 5.975786 6.030943 6.048169 6.099596
```

```
pred_M_hi <- exp(pred_M_hi)
head(pred_M_hi)
```

```
       1        2        3        4        5        6
458.8170 459.0519 393.7775 416.1070 423.3371 445.6778
```

Let's practice!

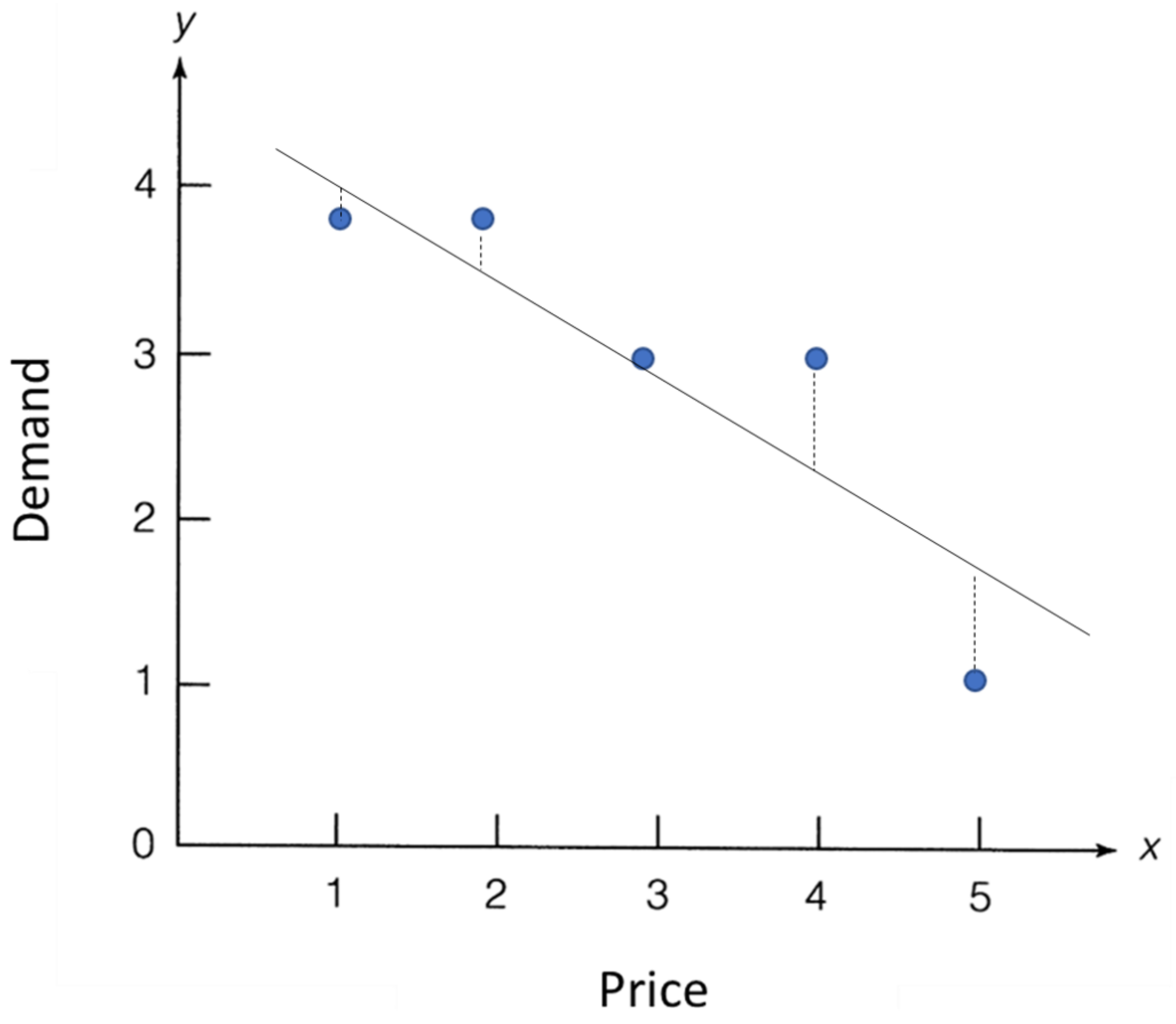
FORECASTING PRODUCT DEMAND IN R

Residuals from regression model

FORECASTING PRODUCT DEMAND IN R

Aric LaBarr, Ph.D.
Senior Data Scientist, Elder Research

Linear Regression



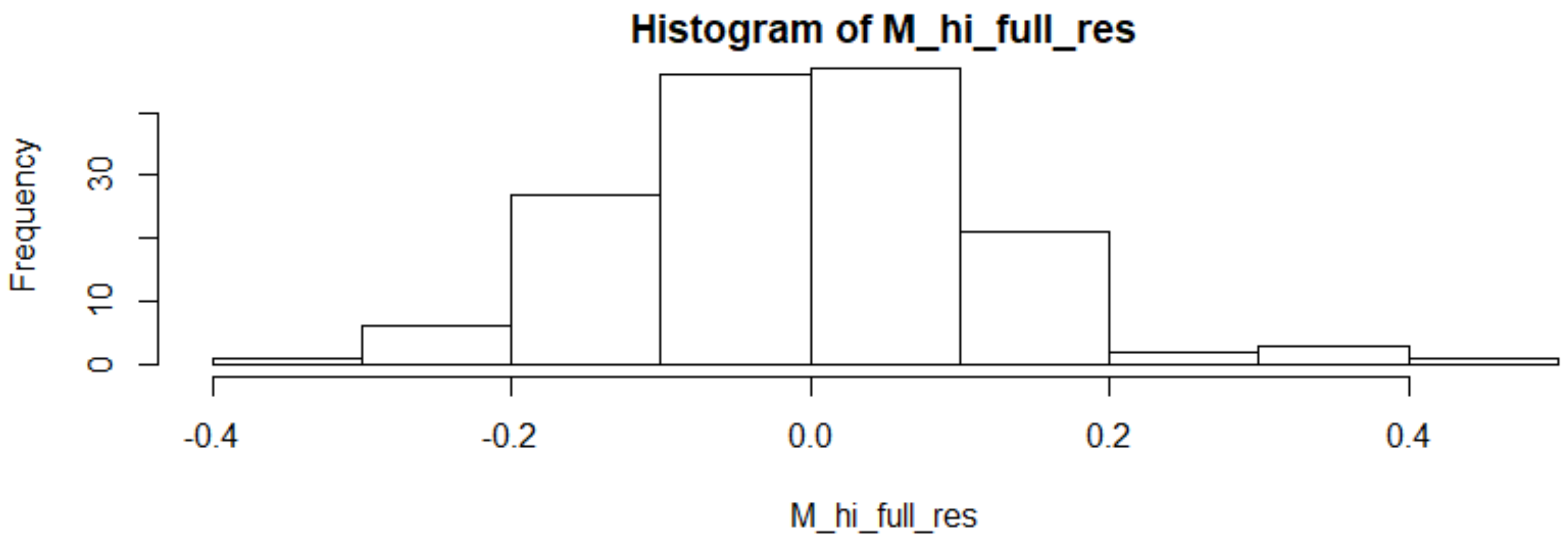
Regression Residuals

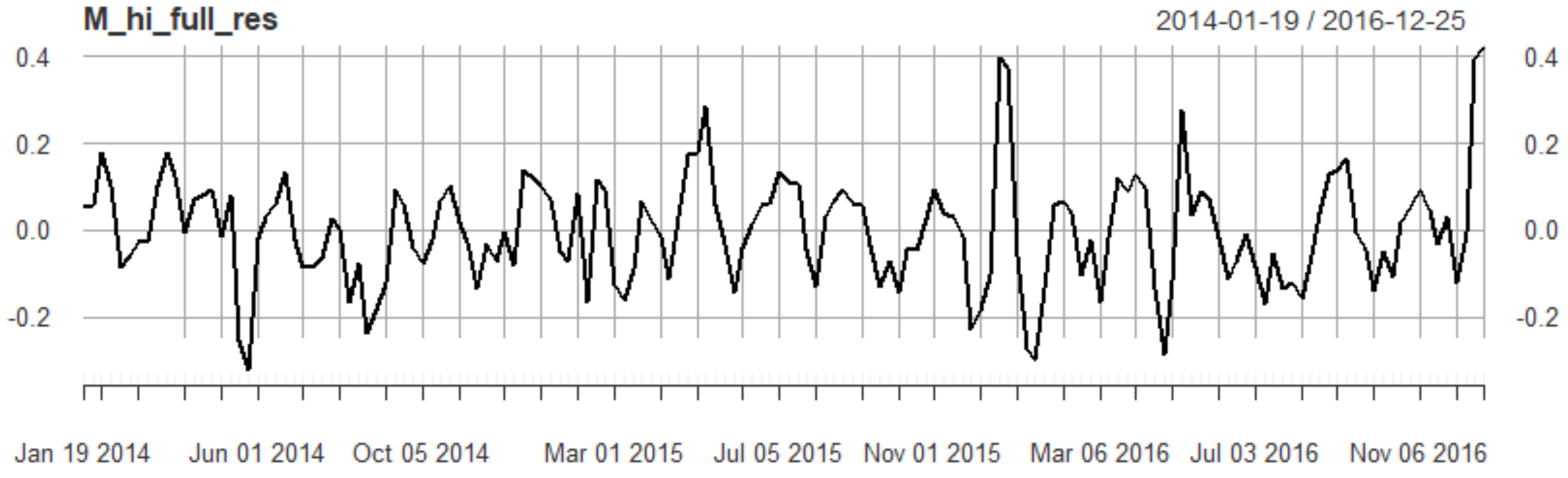
- Ways to reduce residuals further:
  1. Add more important variables to the model
  2. Use time series if your residuals are related over time

Examine Residuals

```
M_hi_full_res <- residuals(model_M_hi_full)
M_hi_full_res <- xts(M_hi_full_res, order.by = dates_train)
hist(M_hi_full_res)
plot(M_hi_full_res)
```

Residual Plots





Let's practice!

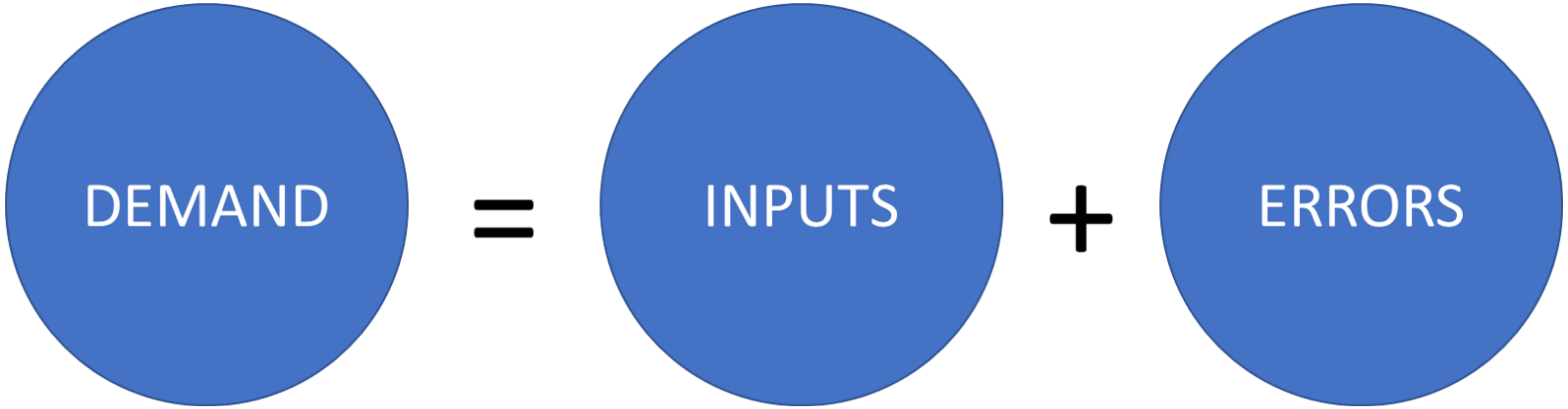
FORECASTING PRODUCT DEMAND IN R

Forecasting residuals

FORECASTING PRODUCT DEMAND IN R

Aric LaBarr, Ph.D.
Senior Data Scientist, Elder Research

Regression Pieces



ARIMA on Residuals

```
M_hi_arima <- auto.arima(M_hi_full_res)
summary(M_hi_arima)
```

```
Series: M_hi_full_res
ARIMA(2,0,1) with zero mean

Coefficients:
         ar1      ar2      ma1
      1.0077  -0.5535  -0.4082
s.e.  0.1291   0.0800   0.1412

sigma^2 estimated as 0.01078:  log likelihood=131.45
AIC=-254.9   AICc=-254.63   BIC=-242.75
```

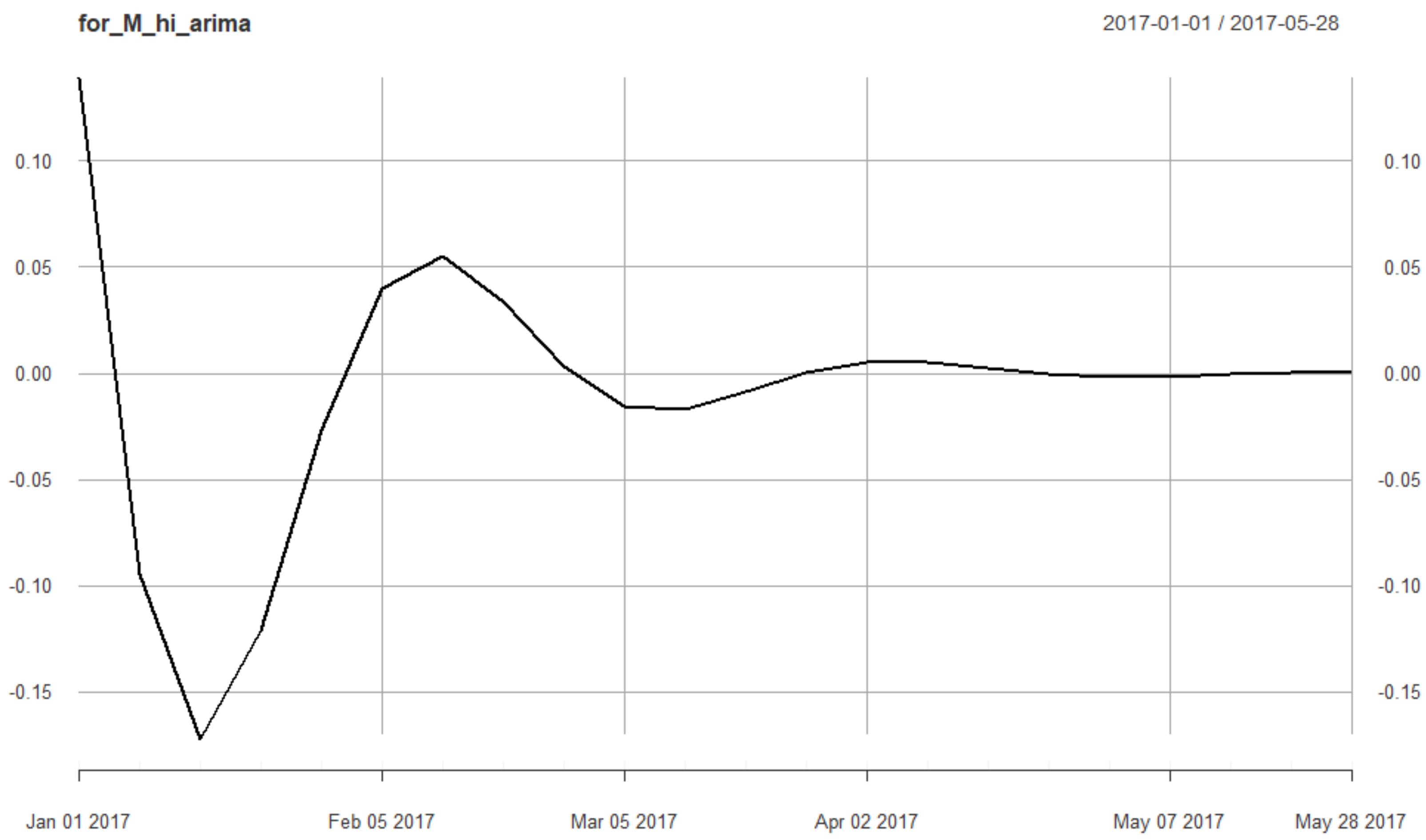
Forecasting Residuals

```r
for_M_hi_arima <- forecast(M_hi_arima, h = 22)
dates_valid <- seq(as.Date("2017-01-01"), length = 22,
                                          by = "weeks")

for_M_hi_arima <- xts(for_M_hi_arima$mean,
                      order.by = dates_valid)
head(for_M_hi_arima, n = 5)
```

```
                  [,1]
2017-01-01  0.13888498
2017-01-08 -0.09448731
2017-01-15 -0.17209098
2017-01-22 -0.12112306
2017-01-29 -0.02680729
```

Visualizing Forecasted Residuals

```
plot(for_M_hi_arima)
```



Let's practice!

FORECASTING PRODUCT DEMAND IN R

Transfer Functions & Ensembling

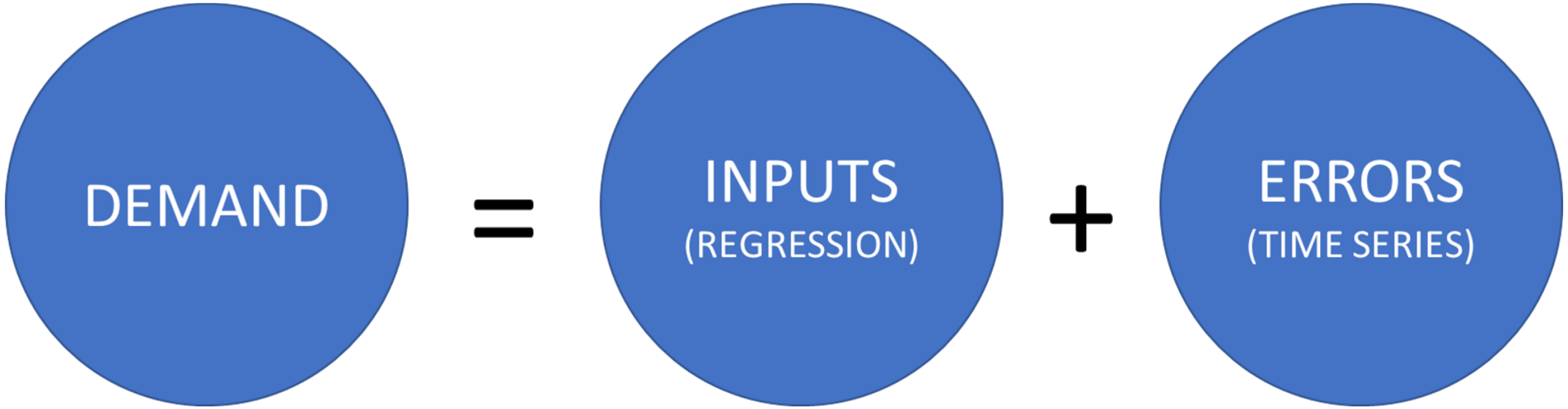
FORECASTING PRODUCT DEMAND IN R

Aric LaBarr, Ph.D.
Senior Data Scientist, Elder Research

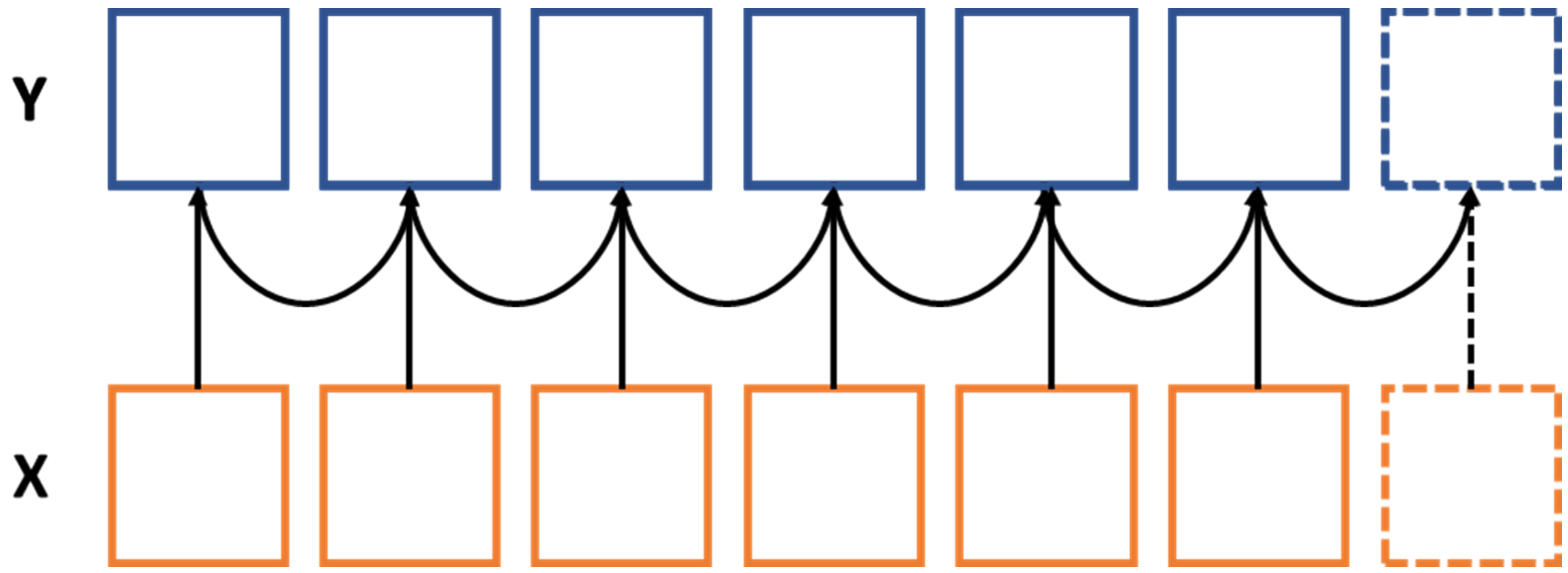
Combining Techniques

- Multiple ways to combine forecasting techniques:
  1. Transfer Functions - everything gets built into one model
  2. Ensembling - "Average" multiple types of model forecasts

Combining Techniques - Transfer Functions



Combining Forecasts



Mathematics in the Background

- Combining two different techniques into one mathematically:

$$\log(Y_t) = \beta_0 + \beta_1 \log(X_t) + \beta_2 X_2 + \ldots + \varepsilon_t$$

$$\varepsilon_t = \alpha_0 + \alpha_1 \varepsilon_{t-1} + \alpha_2 \varepsilon_{t-2} + \ldots + \epsilon$$

- Combining the forecasts into one mathematically:

$$\log(Y_t) = \log(\hat{Y}_t) + \hat{\varepsilon}_t$$

$$Y_t = \hat{Y}_t \times \exp(\hat{\varepsilon})$$

Transfer Function Example

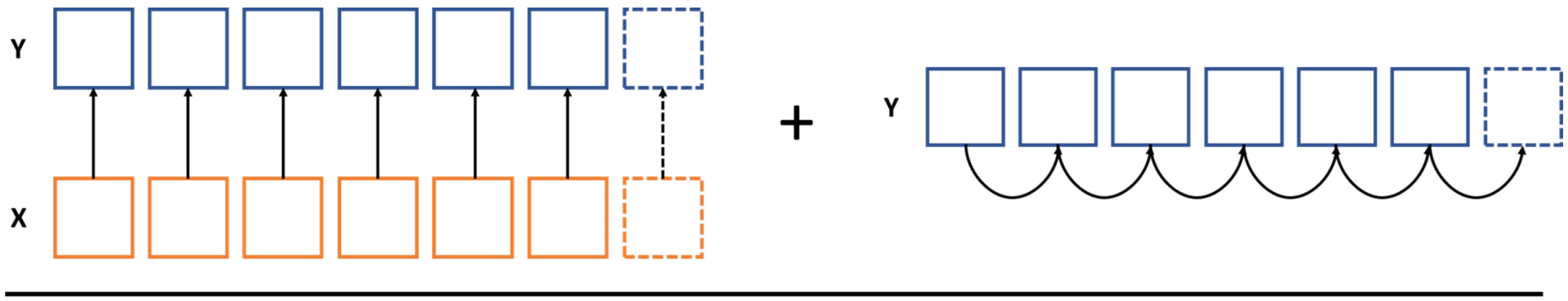
```
for_M_hi_arima <- exp(for_M_hi_arima)
for_M_hi_final <- pred_M_hi_xts * for_M_hi_arima
M_hi_v <- bev_xts_valid[,"M.hi"]

MAE <- mean(abs(for_M_hi_final - M_hi_v))
MAPE <- 100*mean(abs((for_M_hi_final - M_hi_v)/M_hi_v))
print(MAE)
```

```
[1] 61.46033
```

```
print(MAPE)
```

```
[1] 13.45189
```

Combining Forecasts - Ensembling



2

Time Series for Demand

```
M_hi_model_arima <- auto.arima(M_hi)
summary(M_hi_model_arima)
```

```
Series: M_hi
ARIMA(4,0,2) with non-zero mean

Coefficients:
          ar1     ar2      ar3      ar4     ma1     ma2      mean
      -0.1332  0.1546  -0.2638  -0.2063  0.7622  0.0492  458.7097
s.e.   0.4729  0.4150   0.2542   0.1399  0.4807  0.3204    5.7040

sigma^2 estimated as 3323:  log likelihood=-839.66
AIC=1695.33   AICc=1696.32   BIC=1719.62
```

```r
dates_valid <- seq(as.Date("2017-01-01"), length = 22,
                                          by = "weeks")
for_M_hi_xts <- xts(for_M_hi$mean, order.by = dates_valid)

MAE <- mean(abs(for_M_hi_xts - M_hi_v))
MAPE <- 100*mean(abs((for_M_hi_xts - M_hi_v)/M_hi_v))
```

```r
print(MAE)
```

```
[1] 71.43732
```

```r
print(MAPE)
```

```
[1] 16.29178
```

Ensembling Example

```
for_M_hi_en <- 0.5*(for_M_hi_xts + pred_M_hi_xts)
MAE <- mean(abs(for_M_hi_en - M_hi_v))
MAPE <- 100*mean(abs((for_M_hi_en - M_hi_v)/M_hi_v))
```

```
print(MAE)
```

```
[1] 64.12486
```

```
print(MAPE)
```

```
[1] 14.38913
```

Let's practice!