Assignment 7: CME Monthly Seat Prices

Joshua Goldberg

May, 30 2019

Data

Seat prices for the Chicago Mercantile Exchange. There are three classes of seats CME, IMM, and IOM which confer the rights to trade different sets of commodities traded. CME seat owners can trade everything, IMM seat owners are allowed to trade everything except agricultural products, and IOM seat owners are allowed to trade only in index products and all options.

The seat price files are marked as ***S. The files contain the price for which CME seats sold and the date of the sale. As expected, the seat price time series is irregularly spaced in time.

Your task is to adopt an algorithm to create a time series that contains a seat price for each month starting from January 2001 to December 2013. You need to do this for the 3 classes of seats. Note that all 3 seat classes have sale prices for January 2001 so you should not have any start up issues. Please clearly explain why you adopted your algorithm and what other alternatives you may have considered and why you rejected the alternatives.

```r
read_data <- function(.path) {
  read_csv(
    .path,
    col_types = cols(
      DateOfSale = col_character(),
      Year = col_character(),
      Month = col_character(),
      price = col_double(),
      division = col_character()
    )
  )
}

files <- c("cmeS.csv", "immS.csv", "iomS.csv")
all_divisions <- map_dfr(files, ~ read_data(.x))
```

```r
make_ts <- function(.data) {
  .data %>%
  mutate(year_month = yearmonth(mdy(DateOfSale))) %>%
  group_by(year_month, division) %>%
  summarise(monthly_avg_price = mean(price)) %>%
  ungroup() %>%
  as_tsibble(index = year_month, key = division)
}

all_divisions_ts <- make_ts(all_divisions)
```

All divisions have missing values at the `year_month` level:

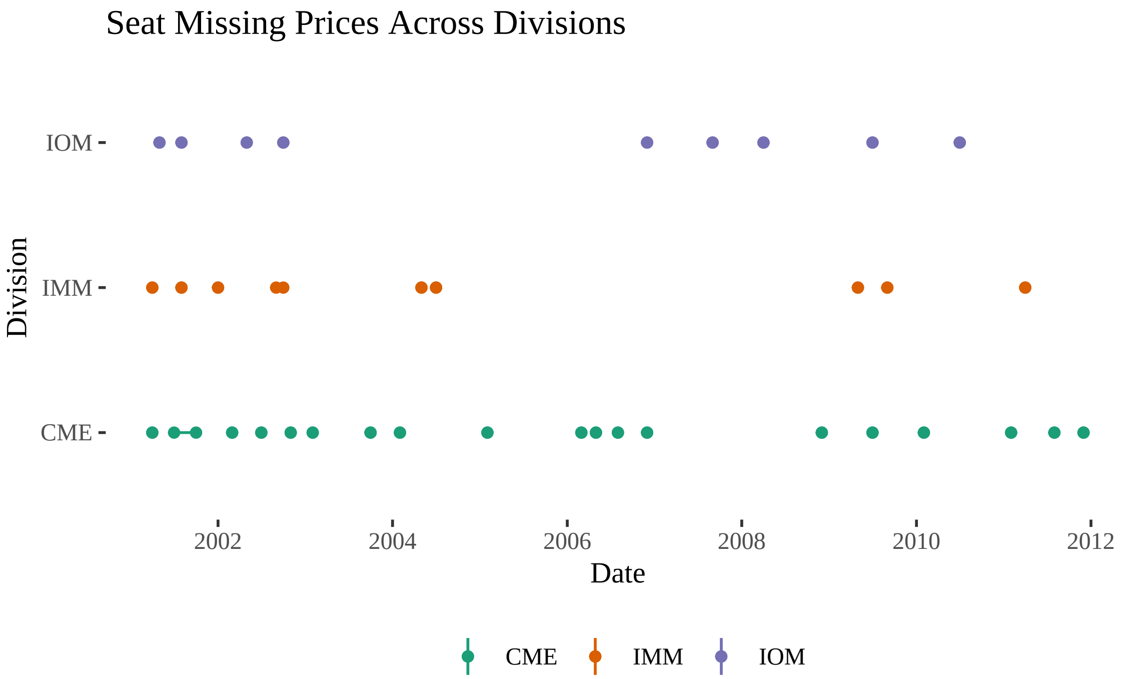
```r
all_divisions_ts %>% has_gaps(.full = TRUE)
```

```
## # A tibble: 3 x 2
##   division .gaps
##   <chr>    <lgl>
## 1 CME      TRUE
## 2 IMM      TRUE
## 3 IOM      TRUE
```

CME has the most consistent missing values across the time series.

```
all_divisions_gaps <- all_divisions_ts %>%
  count_gaps(.full = TRUE)

all_divisions_gaps %>%
  ggplot(aes(division, color = division)) +
  geom_linerange(aes(ymin = .from, ymax = .to)) +
  geom_point(aes(y = .from)) +
  geom_point(aes(y = .to)) +
  coord_flip() +
  labs(title = "Seat Missing Prices Across Divisions",
       x = "Division",
       y = "Date") +
  scale_color_brewer(name = NULL, palette = 2, type = "qual") +
  theme(legend.position = "bottom")
```



```
missing_prices <- all_divisions_ts %>%
  fill_gaps() %>%
  mutate(missing = ifelse(is.na(monthly_avg_price), "missing", "complete")) %>%
  pull(missing)
```

```
divisions_list <- all_divisions_ts %>% split(.$division)

impute_ts <- function(.ts, division, method = "spline") {
  if (method == "spline") {
  .ts %>%
    fill_gaps() %>%
    as.ts() %>%
    na.interpolation(option = method) %>%
    as_tsibble() %>%
    mutate(division = division) %>%
    as_tibble()
  } else if (method == "locf") {
  .ts %>%
    fill_gaps() %>%
    as.ts() %>%
    na.locf(option = method) %>%
    as_tsibble() %>%
    mutate(division = division) %>%
    as_tibble()
  } else if (method == "linear") {
    .ts %>%
      fill_gaps() %>%
      as.ts() %>%
      na.interpolation(option = method) %>%
      as_tsibble() %>%
      mutate(division = division) %>%
      as_tibble()
  }
}

impute_df <- function(.list_data, method) {
  imap_dfr(.list_data, ~ impute_ts(.x, .y, method)) %>%
    mutate(year_month = yearmonth(index)) %>%
    as_tsibble(key = division, index = year_month) %>%
    select(division, year_month, price = value)
}

methods <- c("linear", "spline", "locf")

full_imputes <- map(methods, ~ impute_df(divisions_list, .x)) %>%
  set_names(methods)
```

Check for missing values again.

```
map(full_imputes, ~ .x %>% has_gaps())
```

```
## $linear
## # A tibble: 3 x 2
##   division .gaps
##   <chr>    <lgl>
## 1 CME      FALSE
## 2 IMM      FALSE
## 3 IOM      FALSE
##
## $spline
## # A tibble: 3 x 2
##   division .gaps
##   <chr>    <lgl>
## 1 CME      FALSE
## 2 IMM      FALSE
## 3 IOM      FALSE
```

```
##
## $locf
## # A tibble: 3 x 2
##   division .gaps
##   <chr>    <lgl>
## 1 CME      FALSE
## 2 IMM      FALSE
## 3 IOM      FALSE
```

Adding missing identification back for visualizing the imputations.

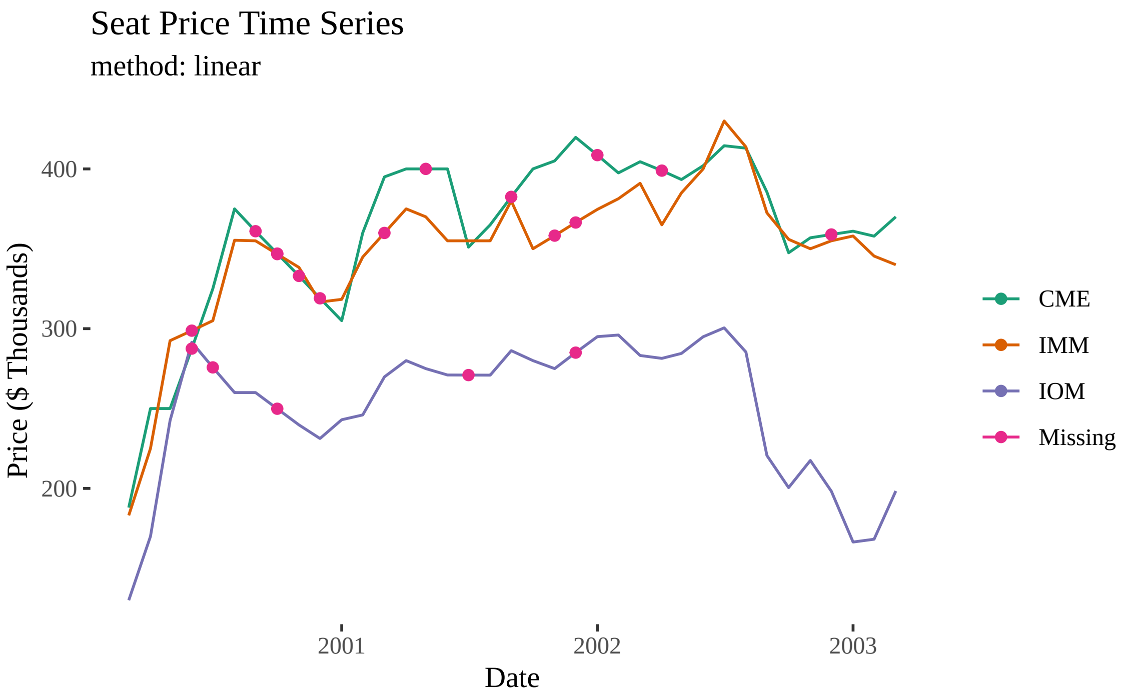
```
full_imputes <- map(full_imputes, ~ .x %>% mutate(missing = missing_prices))
```

We now have no missing values after using spline method for imputation.

```
plot_ts <- function(.data, method, filter_dates = NULL) {
  if (!is.null(filter_dates)) {
  .data %>%
    filter_index(filter_dates) %>%
    ggplot(aes(year_month, price / 1000, color = division)) +
    geom_line() +
    geom_point(data = .data %>% filter(missing == "missing") %>%     filter_index(filter_dates), aes(co
    scale_color_brewer(name = NULL, palette = 2, type = "qual",
                       labels = c("CME", "IMM", "IOM", "Missing")) +
    labs(title = "Seat Price Time Series",
         subtitle = glue::glue("method: {method}"),
         x = "Date",
         y = "Price ($ Thousands)") +
    scale_x_date(date_breaks = "12 month", date_labels = "%C%y") +
    scale_y_continuous(labels = scales::comma)
  } else {
  .data %>%
    ggplot(aes(year_month, price / 1000, color = division)) +
    geom_line() +
    geom_point(data = .data %>% filter(missing == "missing"), aes(color = missing)) +
    scale_color_brewer(name = NULL, palette = 2, type = "qual",
                       labels = c("CME", "IMM", "IOM", "Missing")) +
    labs(title = "Seat Price Time Series",
         subtitle = glue::glue("method: {method}"),
         x = "Date",
         y = "Price ($ Thousands)") +
    scale_x_date(date_breaks = "12 month", date_labels = "%C%y") +
    scale_y_continuous(labels = scales::comma)
  }
}
```

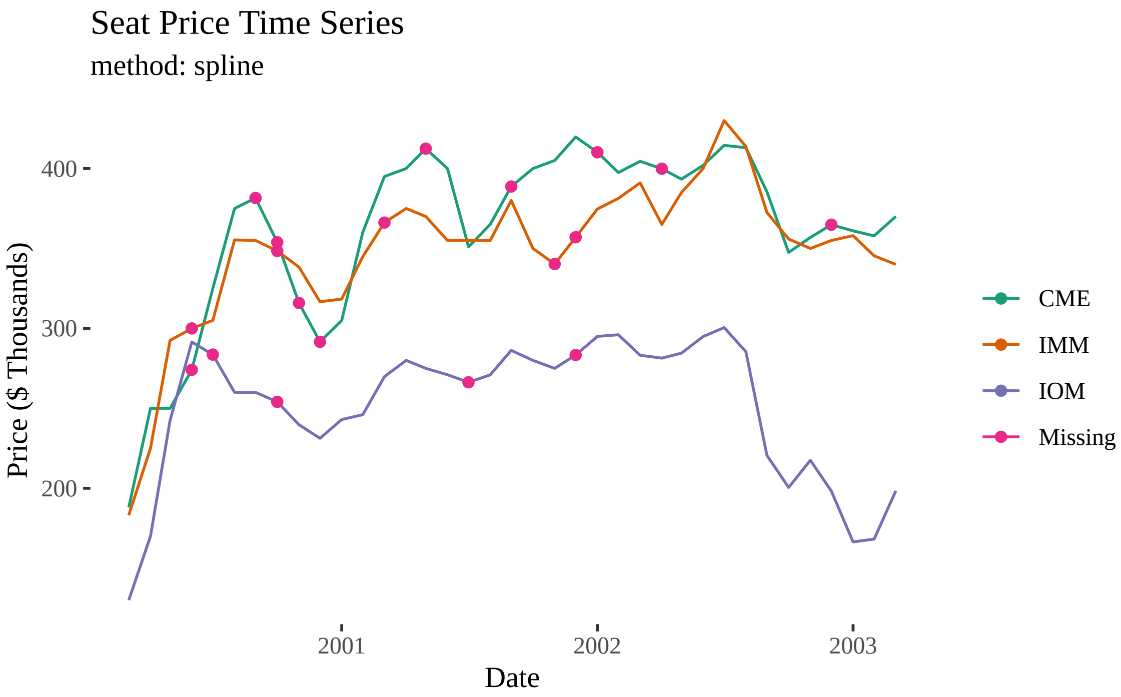
Linear imputation:

```
f_dates <- "2001 Jan" ~ "2004 Jan"
plot_ts(full_imputes$linear, "linear", f_dates)
```



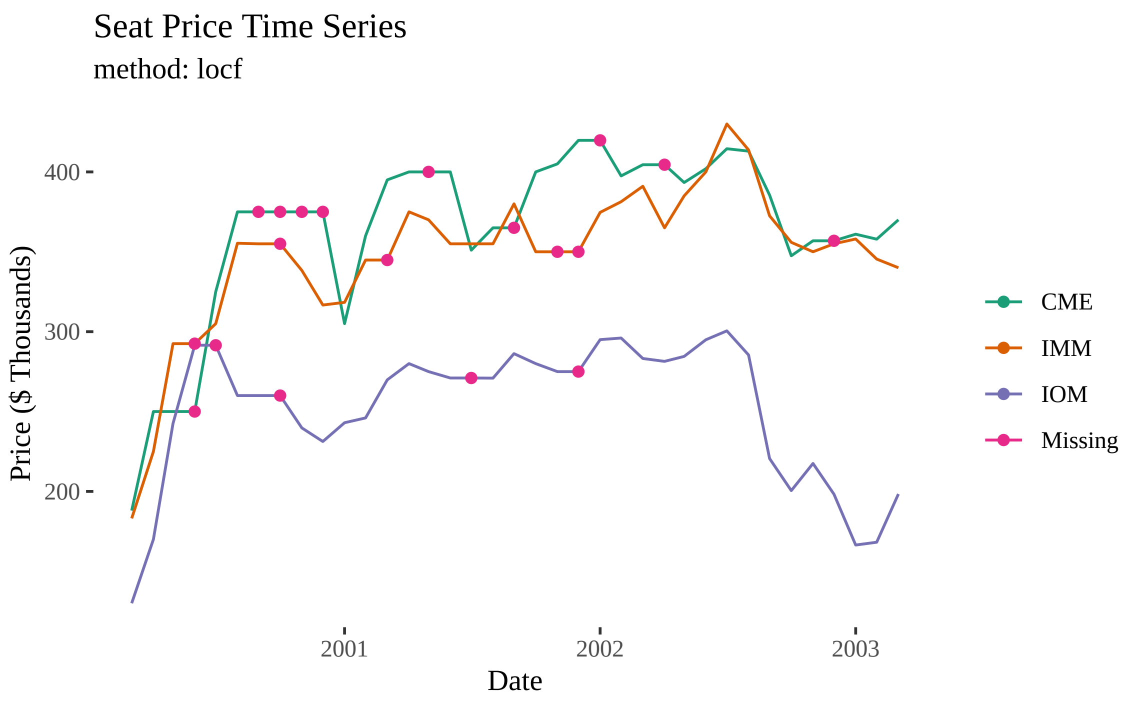
Spline imputation:

```
plot_ts(full_imputes$spline, "spline", f_dates)
```



LOCF imputation:

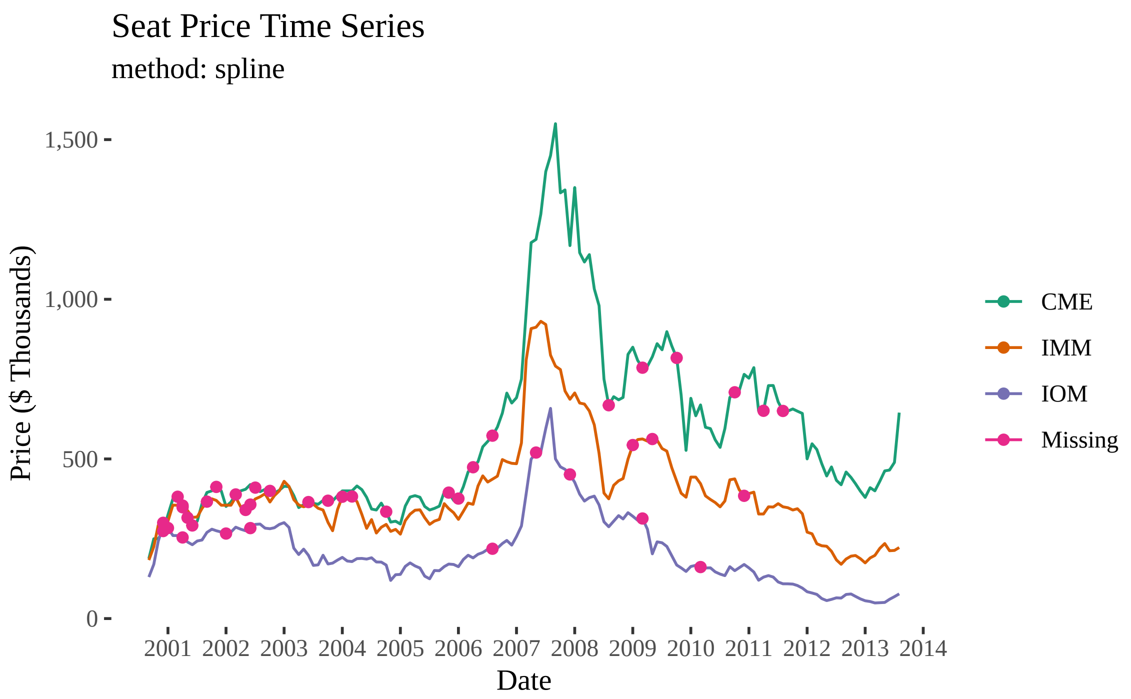
```
plot_ts(full_imputes$locf, "locf", f_dates)
```



Spline provides a higher order of flexibility, resulting in a more reasonable time series imputation. The main benefit of spline is the non linearity as it fits higher order polynomials to estimate the missing values.

Full time series with spline imputation:

```
plot_ts(full_imputes$spline, "spline")
```


Seat Price Time Series
method: spline
Price ($ Thousands)
Date
CME
IMM
IOM
Missing