# PROJECT 6: TIME SERIES

In this project, we analyse the Superstore data used in the course. We'll apply three decomposition models to the sales data, assess their effectiveness, and draw conclusions about the data.

### Data description

The original data has 9994 rows and 21 columns

```
print(df.shape)
df.head()
```

```
(9994, 21)
```

| | Row ID | Order ID | Order Date | Ship Date | Ship Mode | Customer ID | Customer Name | Segment | Country | City | ... | Postal Code | Region | Product ID | Category | Sub-Category | Pro |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | CA-2013-152156 | 2013-11-09 | 2013-11-12 | Second Class | CG-12520 | Claire Gute | Consumer | United States | Henderson | ... | 42420 | South | FUR-BO-10001798 | Furniture | Bookcases | Som Colle Book |
| 1 | 2 | CA-2013-152156 | 2013-11-09 | 2013-11-12 | Second Class | CG-12520 | Claire Gute | Consumer | United States | Henderson | ... | 42420 | South | FUR-CH-10000454 | Furniture | Chairs | Hon D F Uphols Sta Cha |
| 2 | 3 | CA-2013-138688 | 2013-06-13 | 2013-06-17 | Second Class | DV-13045 | Darrin Van Huff | Corporate | United States | Los Angeles | ... | 90036 | West | OFF-LA-10000240 | Office Supplies | Labels | Adh Ad Labe Typew |
| 3 | 4 | US-2012-108966 | 2012-10-11 | 2012-10-18 | Standard Class | SO-20335 | Sean O'Donnell | Consumer | United States | Fort Lauderdale | ... | 33311 | South | FUR-TA-10000577 | Furniture | Tables | Br CR Series Rectar |
| 4 | 5 | US-2012-108966 | 2012-10-11 | 2012-10-18 | Standard Class | SO-20335 | Sean O'Donnell | Consumer | United States | Fort Lauderdale | ... | 33311 | South | OFF-ST-10000760 | Office Supplies | Storage | Eldor 'N Rol Sy |

For this project, we are interested only in sales and order date, so we group the data by order dates and take sum of sales. We also fill in missing dates.

```
df2 = df.groupby('Order Date')['Sales'].sum().reset_index()
df2 = df2.set_index('Order Date')
df2.head()
```

3]:

| Order Date | Sales |
|---|---|
| 2011-01-04 | 16.448 |
| 2011-01-05 | 288.060 |
| 2011-01-06 | 19.536 |
| 2011-01-07 | 4407.100 |
| 2011-01-08 | 87.158 |

```
new_index = pd.date_range(df.index.min(), df.index.max())
df = df.reindex(new_index, fill_value = 0)
```

:

| | Sales |
|---|---|
| 2011-01-04 | 16.4480 |
| 2011-01-05 | 288.0600 |
| 2011-01-06 | 19.5360 |
| 2011-01-07 | 4407.1000 |
| 2011-01-08 | 87.1580 |
| ... | ... |
| 2014-12-27 | 814.5940 |
| 2014-12-28 | 177.6360 |
| 2014-12-29 | 1657.3508 |
| 2014-12-30 | 2915.5340 |
| 2014-12-31 | 713.7900 |

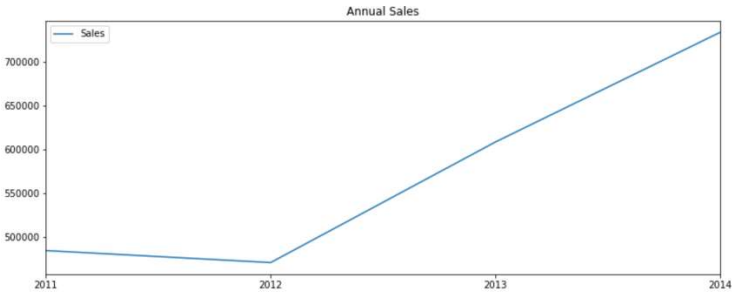
1458 rows × 1 columns

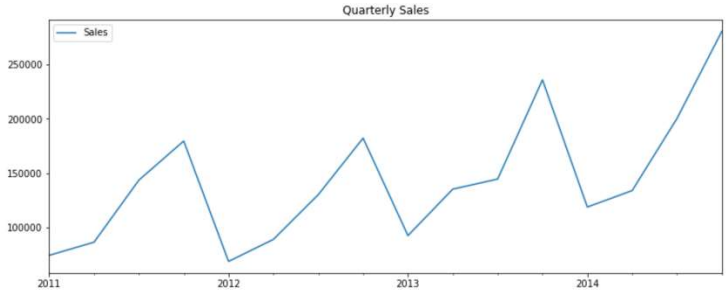
## Exploration

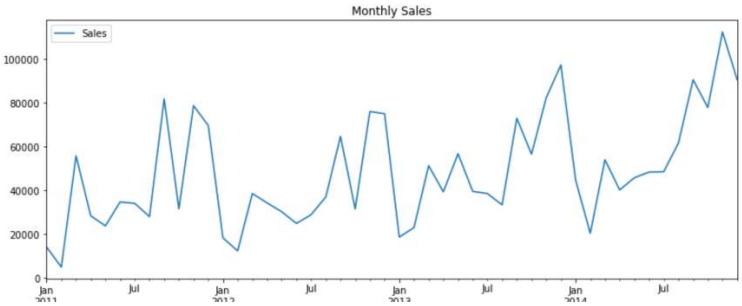
We want to further group the data into either weekly, monthly, quarterly, annual sales. We explore these options and visualise each case:

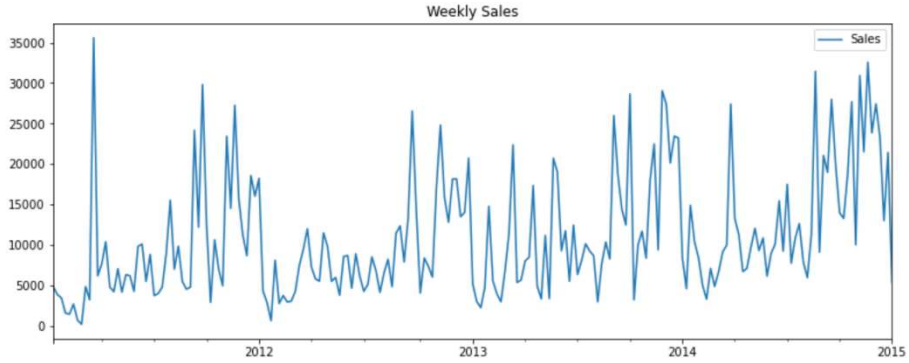
```
sales_weekly = df.resample('W').sum()
print('Weekly Sales')
print(sales_weekly.head(), '\n')

sales_monthly = df.resample('M').sum()
print('Monthly Sales')
print(sales_monthly.head(), '\n')

sales_quarterly = df.resample('Q').sum()
print('Quarterly Sales')
print(sales_quarterly.head(), '\n')

sales_annual = df.resample('Y').sum()
print('Annual Sales')
print(sales_annual.head())
```

```
Weekly Sales
              Sales
2011-01-09  4818.302
2011-01-16  3871.019
2011-01-23  3442.540
2011-01-30  1573.868
2011-02-06  1443.208

Monthly Sales
               Sales
2011-01-31  13946.229
2011-02-28   4810.558
2011-03-31  55691.009
2011-04-30  28295.345
2011-05-31  23648.287

Quarterly Sales
                Sales
2011-03-31   74447.7960
2011-06-30   86538.7596
2011-09-30  143633.2123
2011-12-31  179627.7302
2012-03-31   68851.7386

Annual Sales
                Sales
2011-12-31  484247.4981
2012-12-31  470532.5090
2013-12-31  608473.8300
2014-12-31  733947.0232
```



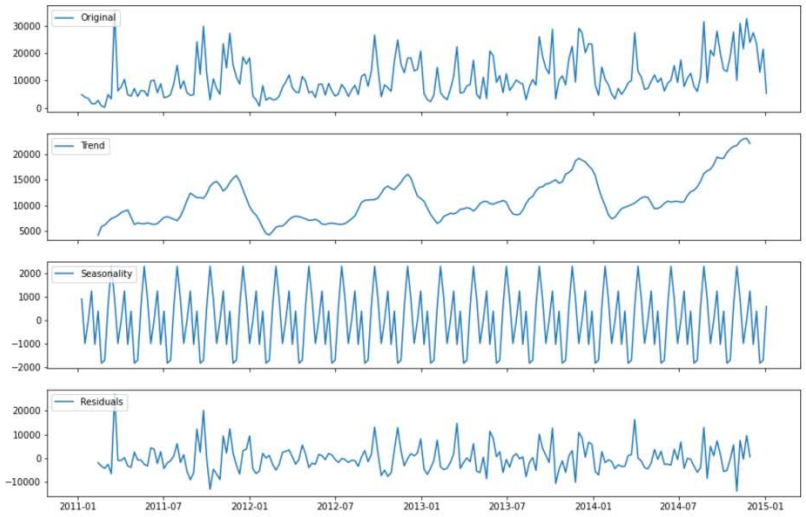
The annual sales data is too sparse to do much analysis. Quarterly and monthly sales data exhibit some trends that are observable by visual inspection. Weekly sales data is richer and looks more challenging, so this will be the target of our analysis.

**Decomposition models**

We tried three decomposition models. First we did an additive decomposition with period 10.

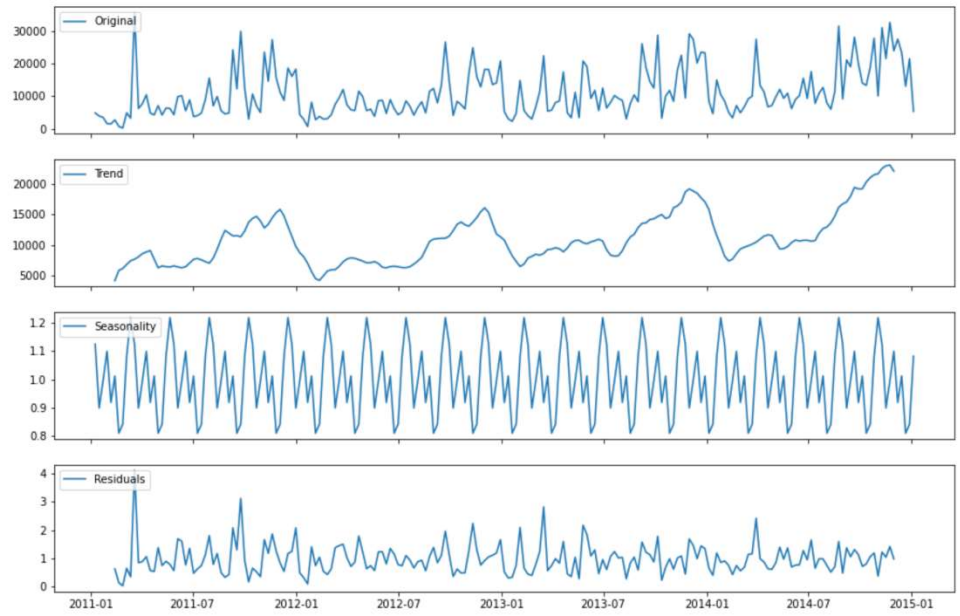
```
# additive
from statsmodels.tsa.seasonal import seasonal_decompose

ss_decomposition = seasonal_decompose(x=sales_weekly, model='additive', period=10)
estimated_trend = ss_decomposition.trend
estimated_seasonal = ss_decomposition.seasonal
estimated_residual = ss_decomposition.resid
```

```
fig, axes = plt.subplots(4, 1, sharex=True, sharey=False)
fig.set_figheight(10)
fig.set_figwidth(15)

axes[0].plot(sales_weekly, label='Original')
axes[0].legend(loc='upper left');

axes[1].plot(estimated_trend, label='Trend')
axes[1].legend(loc='upper left');

axes[2].plot(estimated_seasonal, label='Seasonality')
axes[2].legend(loc='upper left');

axes[3].plot(estimated_residual, label='Residuals')
axes[3].legend(loc='upper left');
```



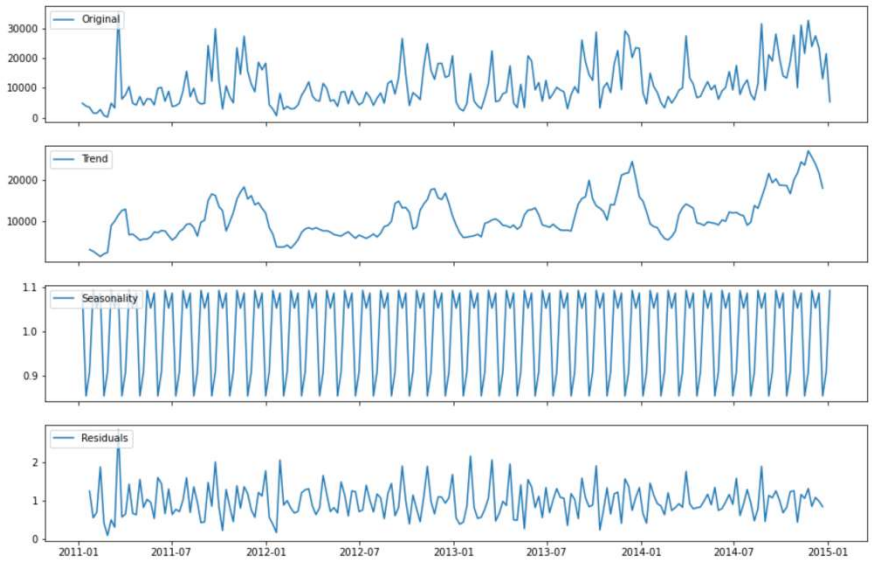

Next we tried a multiplicative model with the same period

```
ss_decomposition = seasonal_decompose(x=sales_weekly, model='multiplicative', period=10)
estimated_trend = ss_decomposition.trend
estimated_seasonal = ss_decomposition.seasonal
estimated_residual = ss_decomposition.resid

fig, axes = plt.subplots(4, 1, sharex=True, sharey=False)
fig.set_figheight(10)
fig.set_figwidth(15)

axes[0].plot(sales_weekly, label='Original')
axes[0].legend(loc='upper left');

axes[1].plot(estimated_trend, label='Trend')
axes[1].legend(loc='upper left');

axes[2].plot(estimated_seasonal, label='Seasonality')
axes[2].legend(loc='upper left');

axes[3].plot(estimated_residual, label='Residuals')
axes[3].legend(loc='upper left');
```




Finally we tried a multiplicative model with period 5

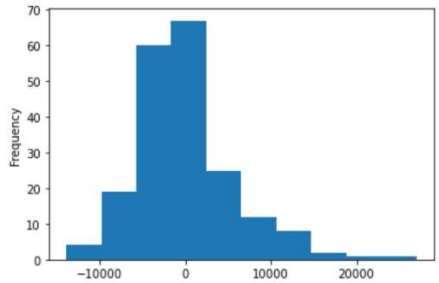
```
ss_decomposition = seasonal_decompose(x=sales_weekly, model='multiplicative', period=5)
estimated_trend = ss_decomposition.trend
estimated_seasonal = ss_decomposition.seasonal
estimated_residual = ss_decomposition.resid

fig, axes = plt.subplots(4, 1, sharex=True, sharey=False)
fig.set_figheight(10)
fig.set_figwidth(15)

axes[0].plot(sales_weekly, label='Original')
axes[0].legend(loc='upper left');

axes[1].plot(estimated_trend, label='Trend')
axes[1].legend(loc='upper left');

axes[2].plot(estimated_seasonal, label='Seasonality')
axes[2].legend(loc='upper left');

axes[3].plot(estimated_residual, label='Residuals')
axes[3].legend(loc='upper left');
```



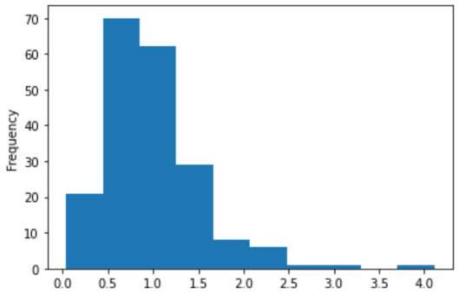
**Key findings**

To assess the models, we plot the residuals to look for normalcy. The respective results are as follows

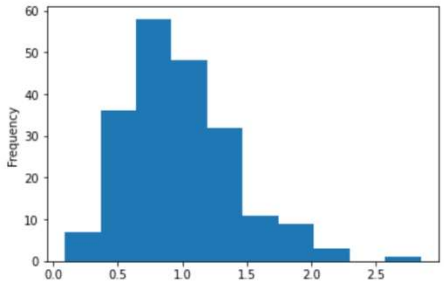
Model 1:




Model 2:




Model 3:




Visual inspection suggests that model 3 comes closest to normal, but it is hard to compare models 1 and 2. To do a more precise assessment, we used Scipy's normaltest with the following results:

Model 1 has a p-value of 6.4e-11
Model 2 has a p-value of 2.5e-20
Model 3 has a p-value of 1.9e-6

Hence we conclude that model 3 performed best, followed by model 1, then model 2.

**Conclusion**

Since the additive model outperformed the multiplicative model with the same parameters, we conclude that the trend and seasonality factors affecting weekly sales are likely to be independent.

Since the model with period 5 outperformed the model with period 10, we conclude that the seasonality factor is more likely to have period 5 than 10.