In [12]:
```
# Imports
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns

from sklearn.metrics import mean_squared_error, mean_absolute_error

%config InlineBackend.figure_format = 'retina'
```

In [13]:
```
# Importing data
df_E = pd.read_csv('C:/Users/Shachee SB/PJME_hourly.csv', index_col='Datetime', parse_da
df_W = pd.read_csv('C:/Users/Shachee SB/PJMW_hourly.csv', index_col='Datetime', parse_da

display( df_E.head(),
         df_W.head())
```

	PJME_MW
Datetime	
2002-12-31 01:00:00	26498.0
2002-12-31 02:00:00	25147.0
2002-12-31 03:00:00	24574.0
2002-12-31 04:00:00	24393.0
2002-12-31 05:00:00	24860.0

	PJMW_MW
Datetime	
2002-12-31 01:00:00	5077.0
2002-12-31 02:00:00	4939.0
2002-12-31 03:00:00	4885.0
2002-12-31 04:00:00	4857.0
2002-12-31 05:00:00	4930.0

In [25]:
```
# Calculate mean and standard deviation
mean = df.mean()
std_dev = df.std()

print("Mean:")
print(mean)

print("\nStandard Deviation:")
print(std_dev)
```

```
Mean:
PJME_MW    32080.505139
PJMW_MW     5602.416524
dtype: float64

Standard Deviation:
PJME_MW    6463.874131
PJMW_MW     979.124070
dtype: float64
```

In [14]:
```
print('---'*20)
print('Energy Consumption Eastern: ')
```

```
print(df_E.info())

print('---'*20)

print('Energy Consumption Western: ')
print(df_W.info())
print('---'*20)

print(f'Shape of Eastern Data Frame: {df_E.shape}')
print(f'Shape of Western Data Frame: {df_W.shape}')
print('---'*20)
```

```
------------------------------------------------------------
Energy Consumption Eastern:
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 145366 entries, 2002-12-31 01:00:00 to 2018-01-02 00:00:00
Data columns (total 1 columns):
 #   Column   Non-Null Count   Dtype  
---  ------   --------------   -----  
 0   PJME_MW  145366 non-null  float64
dtypes: float64(1)
memory usage: 2.2 MB
None
------------------------------------------------------------
Energy Consumption Western:
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 143206 entries, 2002-12-31 01:00:00 to 2018-01-02 00:00:00
Data columns (total 1 columns):
 #   Column   Non-Null Count   Dtype  
---  ------   --------------   -----  
 0   PJMW_MW  143206 non-null  float64
dtypes: float64(1)
memory usage: 2.2 MB
None
------------------------------------------------------------
Shape of Eastern Data Frame: (145366, 1)
Shape of Western Data Frame: (143206, 1)
------------------------------------------------------------
```

In [15]:
```
df = pd.merge(df_E, df_W, how='outer', on='Datetime')
display( df )
```

Datetime	PJME_MW	PJMW_MW
2002-12-31 01:00:00	26498.0	5077.0
2002-12-31 02:00:00	25147.0	4939.0
2002-12-31 03:00:00	24574.0	4885.0
2002-12-31 04:00:00	24393.0	4857.0
2002-12-31 05:00:00	24860.0	4930.0
...	...	...
2018-01-01 20:00:00	44284.0	8401.0
2018-01-01 21:00:00	43751.0	8373.0
2018-01-01 22:00:00	42402.0	8238.0
2018-01-01 23:00:00	40164.0	7958.0
2018-01-02 00:00:00	38608.0	7691.0

145374 rows × 2 columns

In [16]:
```
print('Merged DataFrame Info:')
print(df.info())
print('---'*20)

print('Merged DataFrame NA values:')
print(df.isna().sum())
print('---'*20)

# Removing dumplicated created during merge
df = df[~ df.index.duplicated()]
```

```
Merged DataFrame Info:
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 145374 entries, 2002-12-31 01:00:00 to 2018-01-02 00:00:00
Data columns (total 2 columns):
 #   Column   Non-Null Count   Dtype  
---  ------   --------------   -----  
 0   PJME_MW  145374 non-null  float64
 1   PJMW_MW  143214 non-null  float64
dtypes: float64(2)
memory usage: 3.3 MB
None
------------------------------------------------------------
Merged DataFrame NA values:
PJME_MW       0
PJMW_MW    2160
dtype: int64
------------------------------------------------------------
```

In [17]:
```
def create_features(df):
    """
    Creates time series features from datetime index.
    Special thanks to Rob Mulla for this function!
    """

    from pandas.api.types import CategoricalDtype

    cat_type = CategoricalDtype(categories=['Monday',
                                            'Tuesday',
                                            'Wednesday',
                                            'Thursday',
                                            'Friday',
                                            'Saturday',
                                            'Sunday'],
                                ordered=True)

    df = df.copy()
    df['Datetime'] = df.index
    df['hour'] = df['Datetime'].dt.hour
    df['weekday'] = df['Datetime'].dt.day_name().astype(cat_type)
    df['month'] = df['Datetime'].dt.month
    df['quarter'] = df['Datetime'].dt.quarter
    df['year'] = df['Datetime'].dt.year
    df['dayofmonth'] = df['Datetime'].dt.day
    df['weekofyear'] = df.index.isocalendar().week
    df['date_offset'] = (df.Datetime.dt.month*100 + df.Datetime.dt.day - 320)%1300
    df['season'] = pd.cut(df['date_offset'], [0, 300, 602, 900, 1300],
                          labels=['Spring', 'Summer', 'Fall', 'Winter'])

    df = df.drop(columns='date_offset')
    df = df.set_index('Datetime')

    return df

df_ts = create_features(df)
```

In [18]:
```
display( df_ts )
```

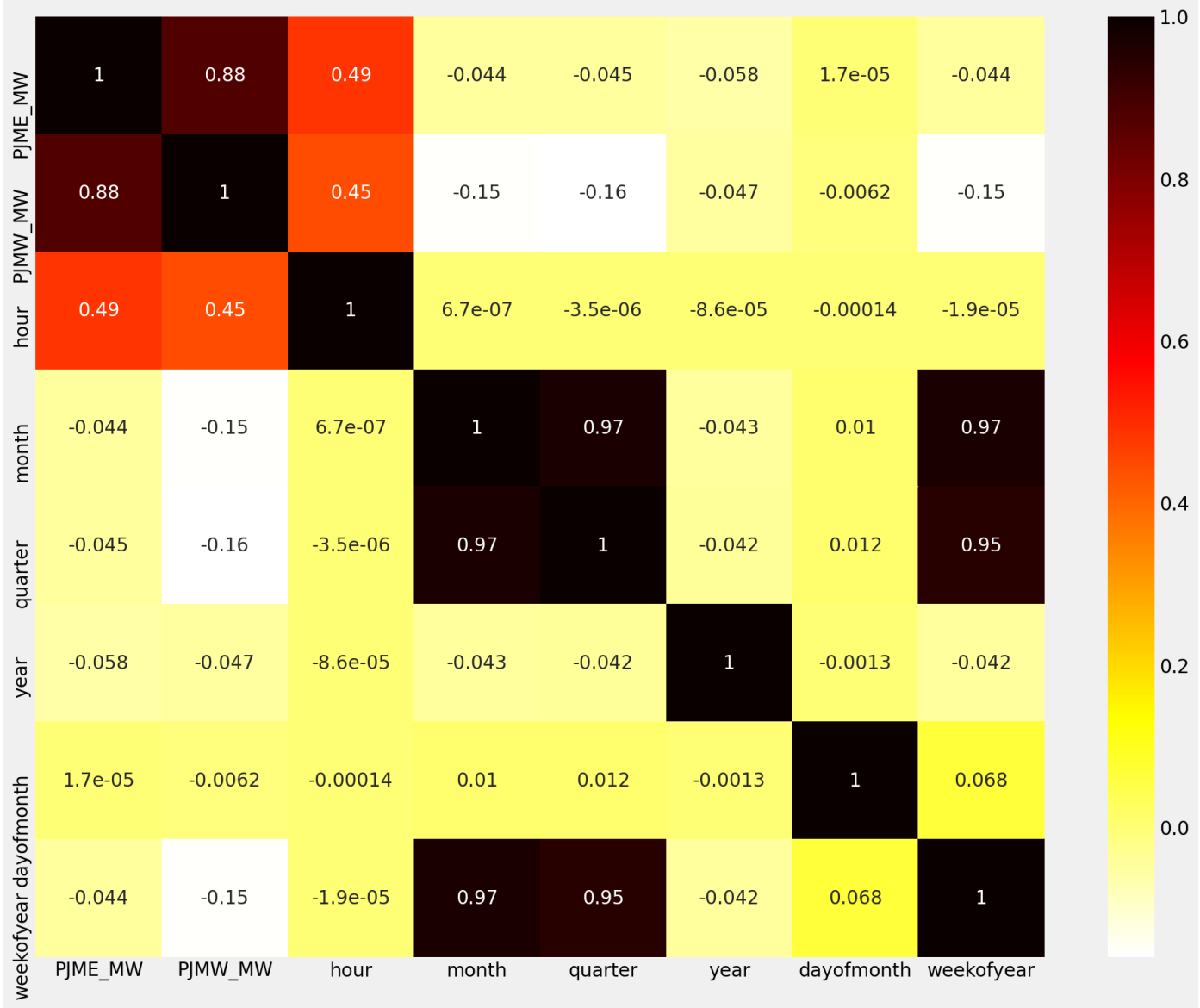
Datetime	PJME_MW	PJMW_MW	hour	weekday	month	quarter	year	dayofmonth	weekofyear	season
2002-12-31 01:00:00	26498.0	5077.0	1	Tuesday	12	4	2002	31	1	Winter
2002-12-31 02:00:00	25147.0	4939.0	2	Tuesday	12	4	2002	31	1	Winter
2002-12-31 03:00:00	24574.0	4885.0	3	Tuesday	12	4	2002	31	1	Winter
2002-12-31 04:00:00	24393.0	4857.0	4	Tuesday	12	4	2002	31	1	Winter
2002-12-31 05:00:00	24860.0	4930.0	5	Tuesday	12	4	2002	31	1	Winter
...	...	...	...	...	...	...	...	...	...	...
2018-01-01 20:00:00	44284.0	8401.0	20	Monday	1	1	2018	1	1	Winter
2018-01-01 21:00:00	43751.0	8373.0	21	Monday	1	1	2018	1	1	Winter
2018-01-01 22:00:00	42402.0	8238.0	22	Monday	1	1	2018	1	1	Winter
2018-01-01 23:00:00	40164.0	7958.0	23	Monday	1	1	2018	1	1	Winter
2018-01-02 00:00:00	38608.0	7691.0	0	Tuesday	1	1	2018	2	1	Winter

145362 rows × 10 columns

In [19]:
```
display( df_ts.describe().iloc[:, :2].T.style.background_gradient(cmap='tab20_r') )
```

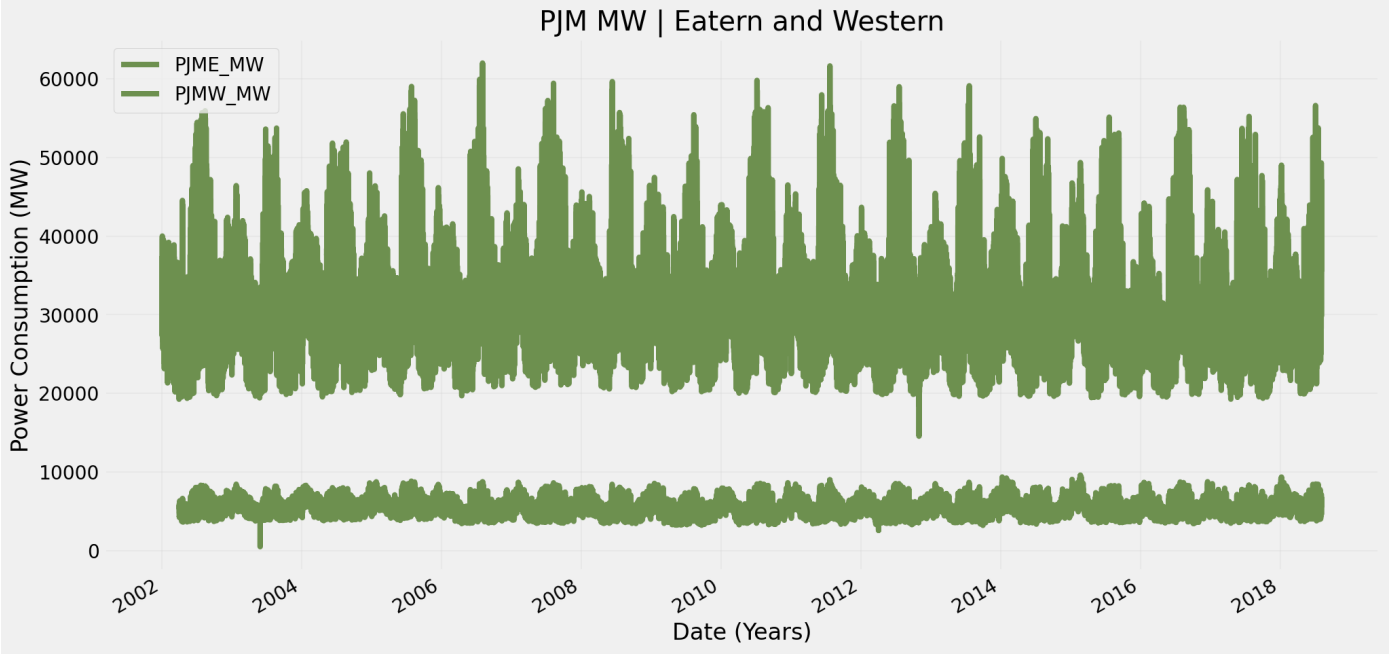
	count	mean	std	min	25%	50%	75%
PJME_MW	145362.000000	32080.505139	6463.874131	14544.000000	27573.000000	31421.000000	35650.000000
PJMW_MW	143202.000000	5602.416524	979.124070	487.000000	4907.000000	5530.000000	6252.000000

In [20]:
```
plt.figure(figsize=(15,12))
sns.heatmap(df_ts.corr(), annot=True, cmap='hot_r')
plt.show()
```



In [21]:

```
ax = df.plot(figsize=(15, 8),
             color=['C9', 'C3'],
             title='PJM MW | Eatern and Western',
             ylabel='Power Consumption (MW)',
             xlabel='Date (Years)',
             sharex=True,
             sharey=True)
ax.grid(alpha=0.2)
plt.show()
```



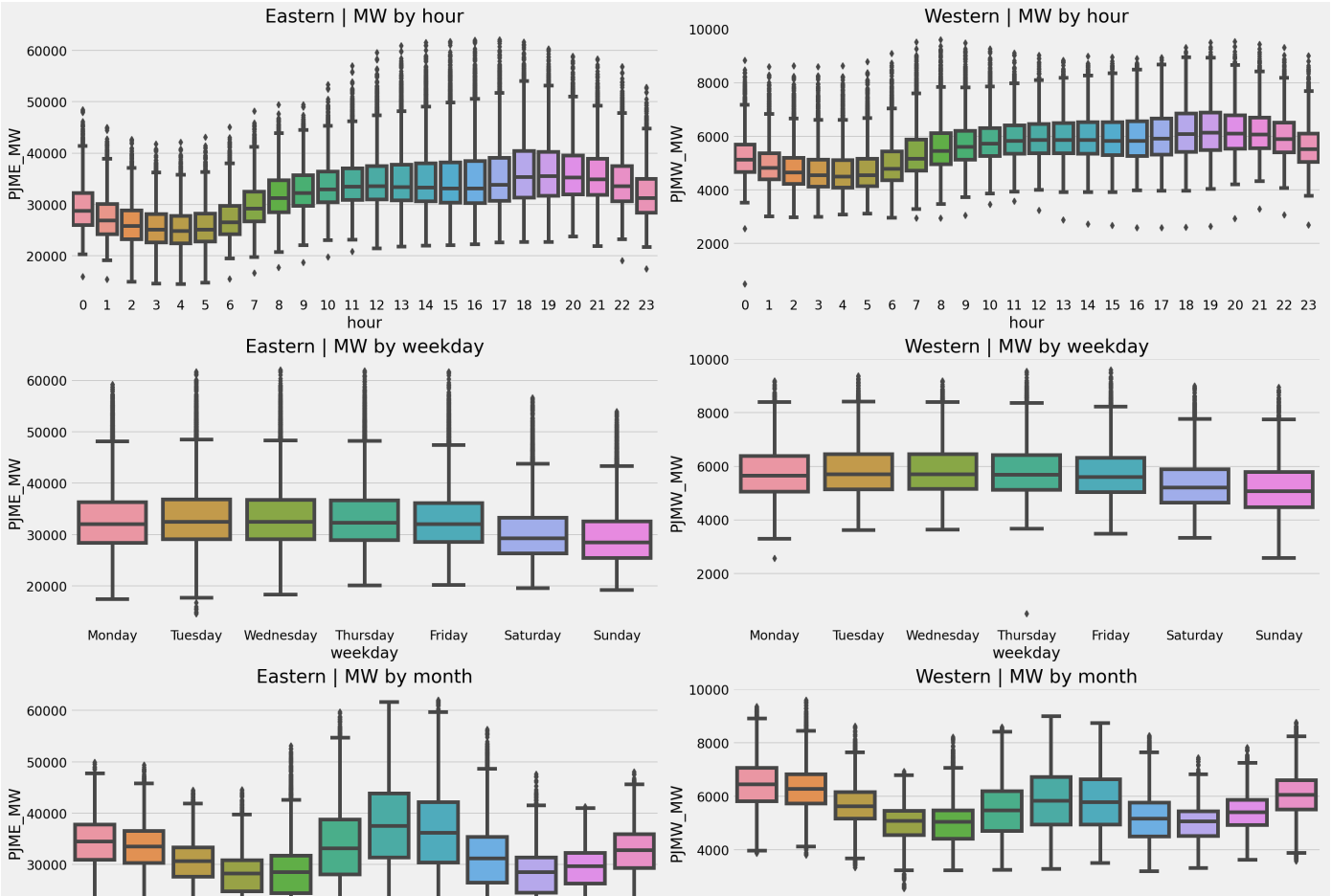
In [22]:
```
ts_curious = ['hour', 'weekday', 'month', 'quarter', 'year', 'dayofmonth', 'weekofyear',
ts_el = len(ts_curious)
```

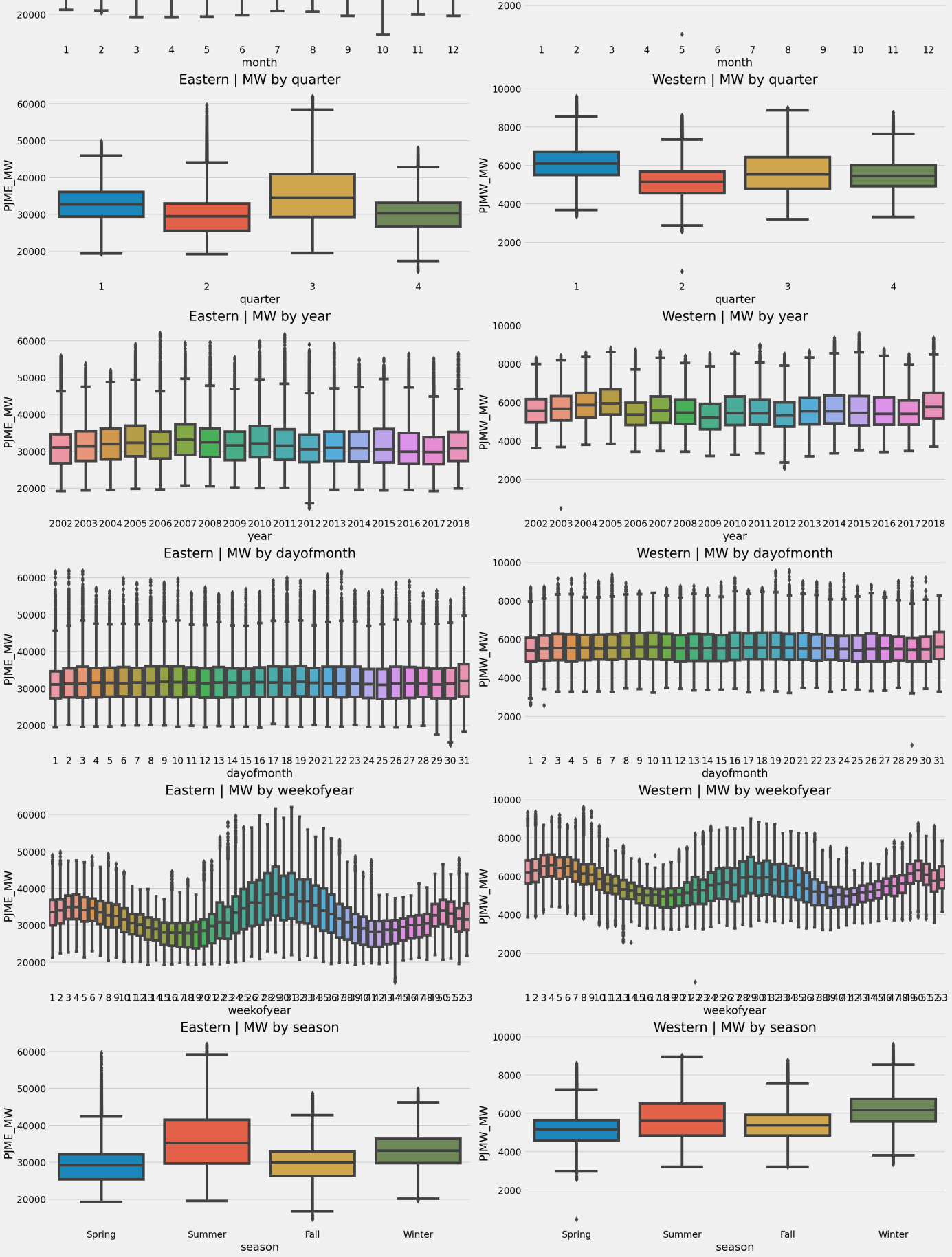
In [23]:
```
fig, ax = plt.subplots(nrows=ts_el, ncols=2, figsize=(20, 5*ts_el))

for i, ts in enumerate(ts_curious):
    sns.boxplot(data=df_ts, x=ts, y='PJME_MW', ax=ax[i, 0])
    ax[i, 0].set_title(f'Eastern | MW by {ts}')

    sns.boxplot(data=df_ts, x=ts, y='PJMW_MW', ax=ax[i, 1])
    ax[i, 1].set_title(f'Western | MW by {ts}')

plt.tight_layout(pad=0, w_pad=0.5, h_pad=0.5)
plt.show()
```





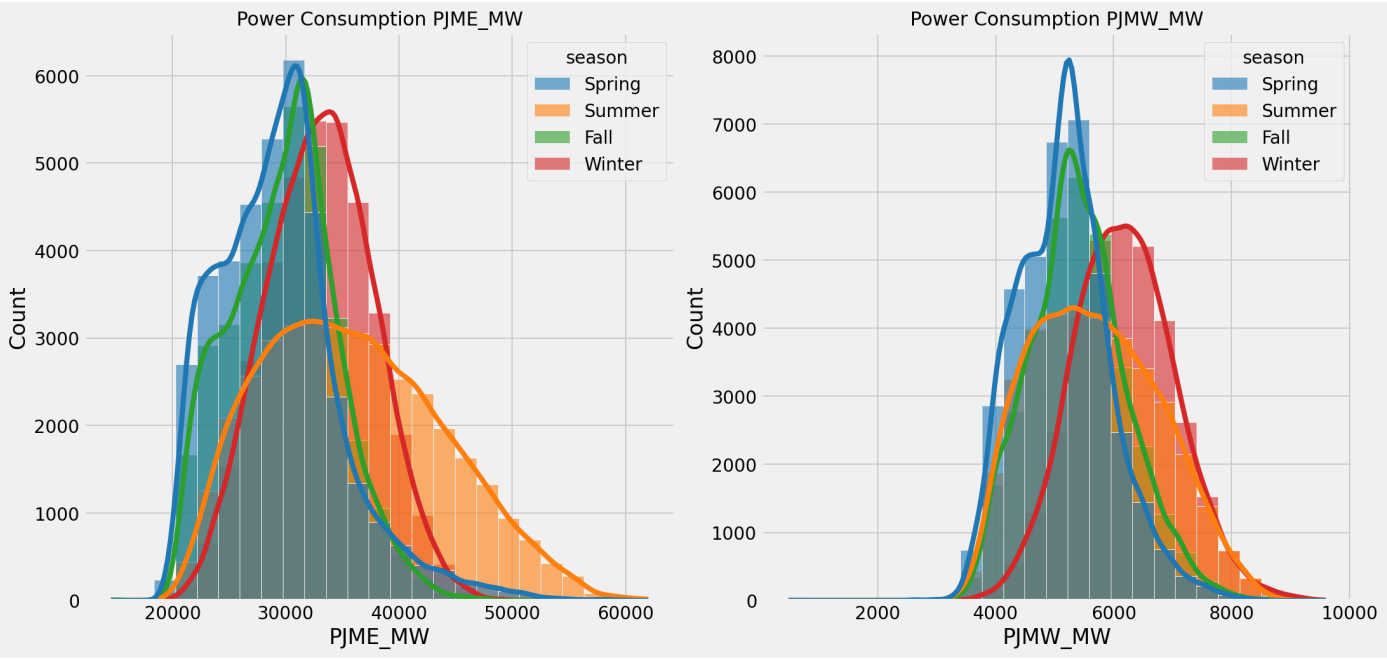
In [24]:

```
fig, ax = plt.subplots(ncols=2, figsize=(15,7))

for i, reg in enumerate(['PJME_MW','PJMW_MW']):
    sns.histplot(df_ts[['PJME_MW','PJMW_MW','year','season']],
                 x=reg,
                 kde=True,
                 bins=25,
```

```
                    hue='season',
                    palette='tab10',
                    alpha=0.6,
                    ax=ax[i])
    ax[i].set_title('Power Consumption ' + reg, fontsize=15)

plt.tight_layout(pad=0, w_pad=0.5, h_pad=0.5)
plt.show()
```

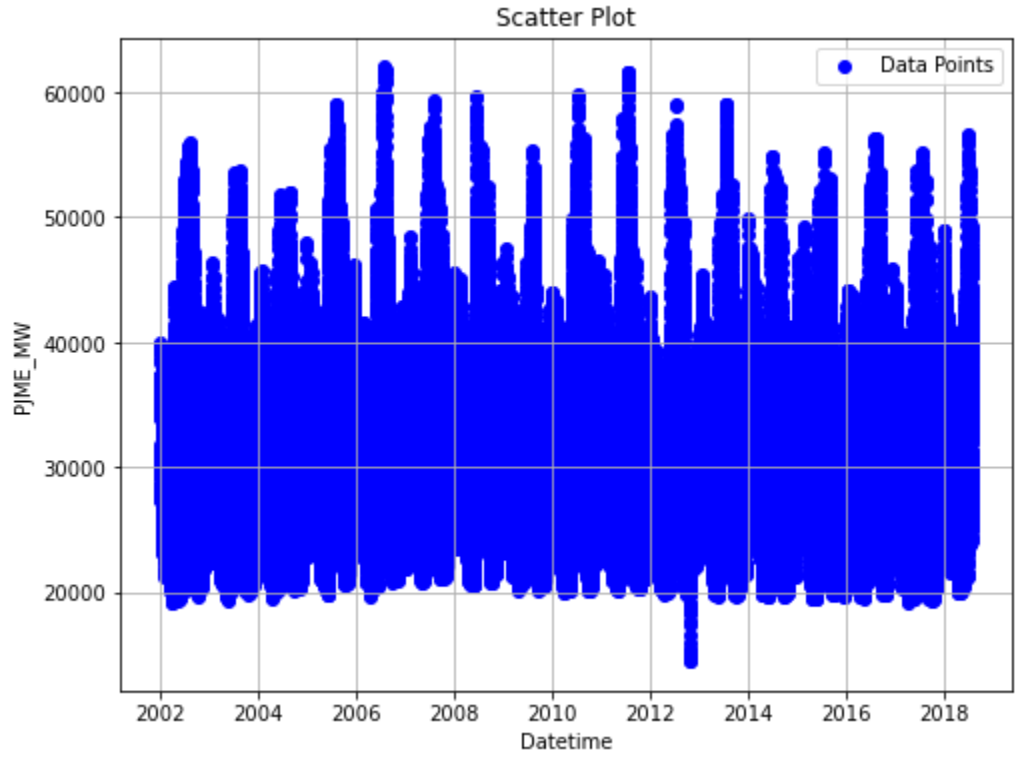


In [2]:

```
import pandas as pd
import matplotlib.pyplot as plt

# Read data from CSV file, parsing 'Datetime' column as datetime objects
data = pd.read_csv('C:/Users/Shachee SB/PJME_hourly.csv', parse_dates=['Datetime'])

# Plot
plt.figure(figsize=(8, 6))
plt.scatter(data['Datetime'], data['PJME_MW'], color='blue', label='Data Points')
plt.title('Scatter Plot')
plt.xlabel('Datetime')
plt.ylabel('PJME_MW')
plt.legend()
plt.grid(True)
plt.show()
```

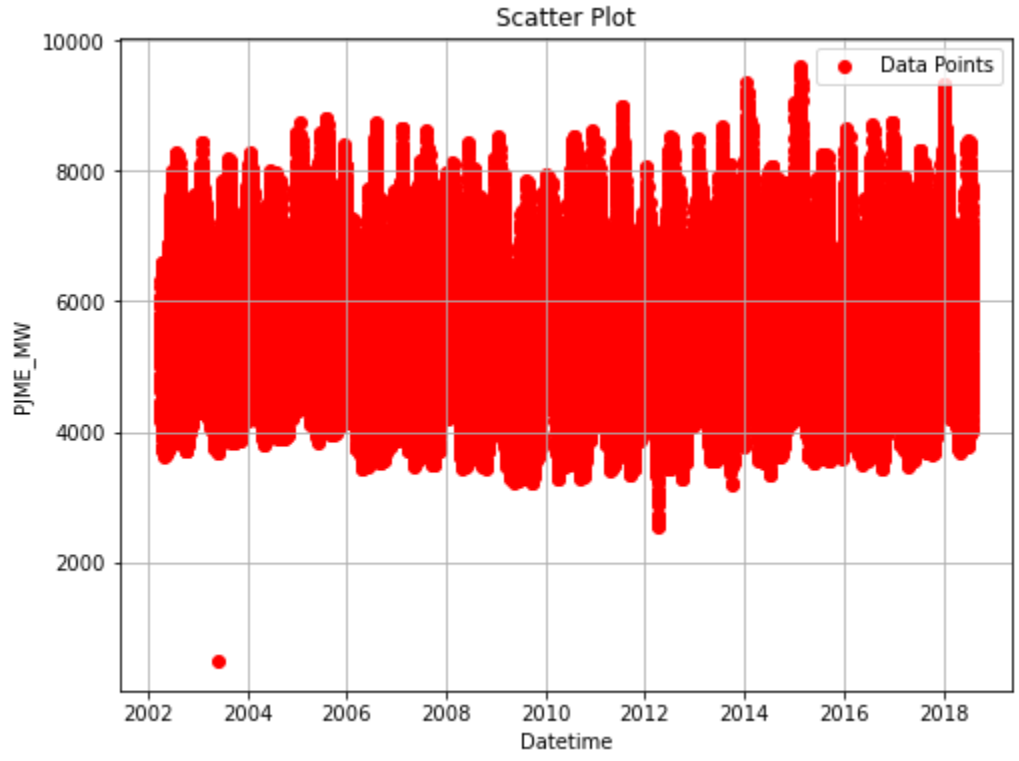


In [3]:

```
import pandas as pd
import matplotlib.pyplot as plt

# Read data from CSV file, parsing 'Datetime' column as datetime objects
data = pd.read_csv('C:/Users/Shachee SB/PJMW_hourly.csv', parse_dates=['Datetime'])

# Plot
plt.figure(figsize=(8, 6))
plt.scatter(data['Datetime'], data['PJMW_MW'], color='red', label='Data Points')
plt.title('Scatter Plot')
plt.xlabel('Datetime')
plt.ylabel('PJME_MW')
plt.legend()
plt.grid(True)
plt.show()
```



In [ ]: