In [1]:
```
# DSC530-T302
# Stephen Smitshoek
# Week09
# Exercise 12-2
```

In [2]:
```
import pandas
import numpy as np
import statsmodels.formula.api as smf

import thinkplot
import thinkstats2
import regression
import timeseries
```

In [3]:
```
def group_by_quality_and_day(transactions):
    groups = transactions.groupby('quality')
    dailies = {}
    for name, group in groups:
        dailies[name] = group_by_day(group)

    return dailies
```

In [4]:
```
def group_by_day(transactions, func=np.mean):
    grouped = transactions[['date', 'ppg']].groupby('date')
    daily = grouped.aggregate(func)

    daily['date'] = daily.index
    start = daily.date[0]
    one_year = np.timedelta64(1, 'Y')
    daily['years'] = (daily.date - start) / one_year

    return daily
```

In [5]:
```
def run_quadratic_model(daily):
    daily['years2'] = daily.years**2
    model = smf.ols('ppg ~ years + years2', data=daily)
    results = model.fit()
    return model, results
```

In [6]:
```
def run_linear_model(daily):
    model = smf.ols('ppg ~ years', data=daily)
    results = model.fit()
    return model, results
```

In [7]:
```
class SerialCorrelationTest(thinkstats2.HypothesisTest):
    def TestStatistic(self, data):
        series, lag = data
        corr = thinkstats2.SerialCorr(series, lag)

        return corr

    def RunModel(self):
        series, lag = self.data
        randomized_series = series[np.random.permutation(series.index)]

        return (randomized_series, lag)
```

In [8]:
```
transactions = pandas.read_csv('mj-clean.csv', parse_dates=[5])
dailies = group_by_quality_and_day(transactions)
daily = dailies['high']
```

In [9]:
```
serial_corr = SerialCorrelationTest((daily.ppg, 1))
corr = serial_corr.actual
p_val = serial_corr.PValue()

print(f'The serial correlation is {corr:.3f} with a P-Value of {p_val}')
```

```
The serial correlation is 0.485 with a P-Value of 0.0
```

In [12]:
```
_, results = run_linear_model(daily)
residuals = results.resid
serial_corr = SerialCorrelationTest((residuals, 1))
corr = serial_corr.actual
p_val = serial_corr.PValue()

print(f'The serial correlation is {corr:.3f} with a P-Value of {p_val}')
```

```
The serial correlation is 0.076 with a P-Value of 0.003
```

In [10]:
```
_, results = run_quadratic_model(daily)
residuals = results.resid
serial_corr = SerialCorrelationTest((residuals, 1))
corr = serial_corr.actual
p_val = serial_corr.PValue()

print(f'The serial correlation is {corr:.3f} with a P-Value of {p_val}')
```

```
The serial correlation is 0.056 with a P-Value of 0.022
```