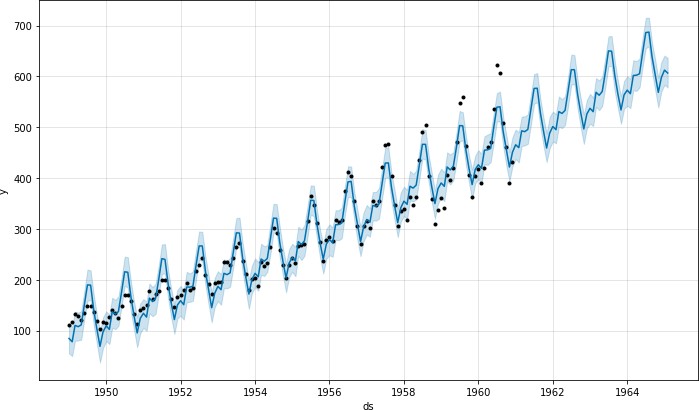
Multiplicative Seasonality

# By default Prophet fits additive seasonalities, 默认情况下，Prophet拟合可加性的季节性。meaning the effect of the seasonality is added to the trend to get the forecast. 这意味着季节性的影响被添加在趋势中进行预测。This time series of the number of air passengers is an example of when additive seasonality does not work:接下来的示例给出了一个默认可加性季节性不能使用的例子。

1. # Python
2. df = pd.read\_csv('https://raw.githubusercontent.com/facebook/prophet/main/examples/example\_air\_passengers.csv') 3 m = Prophet()
3. m.fit(df)
4. future = m.make\_future\_dataframe(50, freq='MS') 6 forecast = m.predict(future)

7 fig = m.plot(forecast)



# This time series has a clear yearly cycle, but the seasonality in the forecast is too large at the start of the time series and too

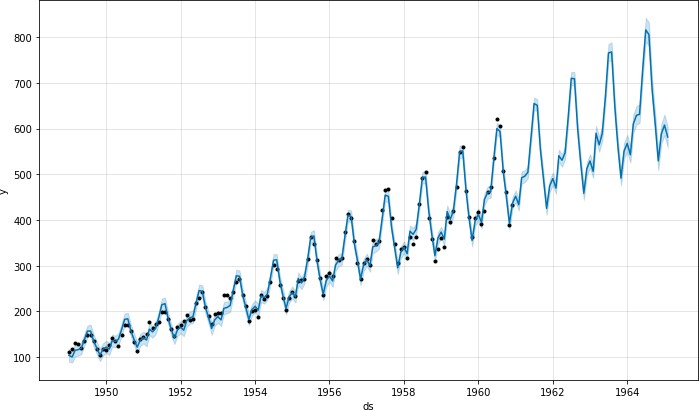
small at the end. In this time series, the seasonality is not a constant additive factor as assumed by Prophet, rather it grows with the trend. This is multiplicative seasonality. 从图中可知，这个时间序列周期明显，但预测值的拟合程度在刚开始的几个周期内过大。最后又变得很小。***在这个时间序列中，季节性并不是Prophet所假设的一个常数可加性因子，而是随着趋势而增长的。这就是乘性季节性***

Prophet can model multiplicative seasonality by setting seasonality\_mode='multiplicative' in the input arguments:

通过在输入参数中设置seasonality\_mode='multiplicative'， Prophet可以对乘性季节性进行建模

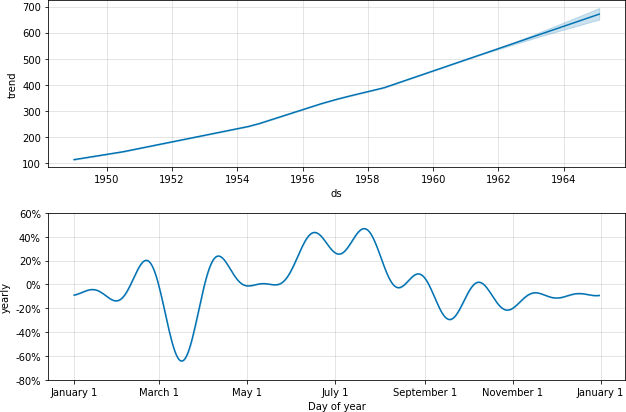
1. # Python
2. m = Prophet(seasonality\_mode='multiplicative') 3 m.fit(df)

4 forecast = m.predict(future) 5 fig = m.plot(forecast)



# The components figure will now show the seasonality as a percent of the trend: 下面的成分图将显示季节性占趋势的百分比:

1. # Python
2. fig = m.plot\_components(forecast)



With ***seasonality\_mode='multiplicative'***, holiday effects will also be modeled as multiplicative. 在建立Prophet对象时，使用seasonality\_mode='multiplicative'，假日效应也会被建模为multiplicative。

Any added seasonalities or extra regressors will by default use whatever ***seasonality\_mode*** is set to, 任何增加的季节性因素或额外的回归因素将默认使用***seasonality\_mode***设置的值but can be overridden by specifying ***mode='additive' or mode='multiplicative'*** as an argument when adding the seasonality or regressor.但是当增加季节性和回归时，这可以被通过指定***mode='additive' or mode='multiplicative'*** 实现覆盖

**就是说当我建立 *Prophet()* 对象时，可以对季节性模式进行优先设置值 *seasonality\_mode = ‘additive’ / ‘multiplicatve’* ，这种情况下，接下来的所有的季节性因素和额外的回归因素都会被改写为相同模式。不过，在后续自定义添加季节性*[ ( Prophet().add\_seasonality () ]*和额外回归*[ ( Prophet().add\_regressor () ]*时也可以使用*mode*参数进行自定义。即*mode = ‘additive’/’’multiplicative***

# For example, this block sets the built-in seasonalities to multiplicative, but includes an additive quarterly seasonality and an additive regressor:

1. # Python
2. m = Prophet(seasonality\_mode='multiplicative')
3. m.add\_seasonality('quarterly', period=91.25, fourier\_order=8, mode='additive') 4 m.add\_regressor('regressor', mode='additive')

# Additive and multiplicative extra regressors will show up in separate panels on the components plot. Note, however, that it is pretty unlikely to have a mix of additive and multiplicative seasonalities, so this will generally only be used if there is a reason to expect that to be the case.

[Edit on GitHub](https://github.com/facebook/prophet/blob/main/docs/_docs/multiplicative_seasonality.md)