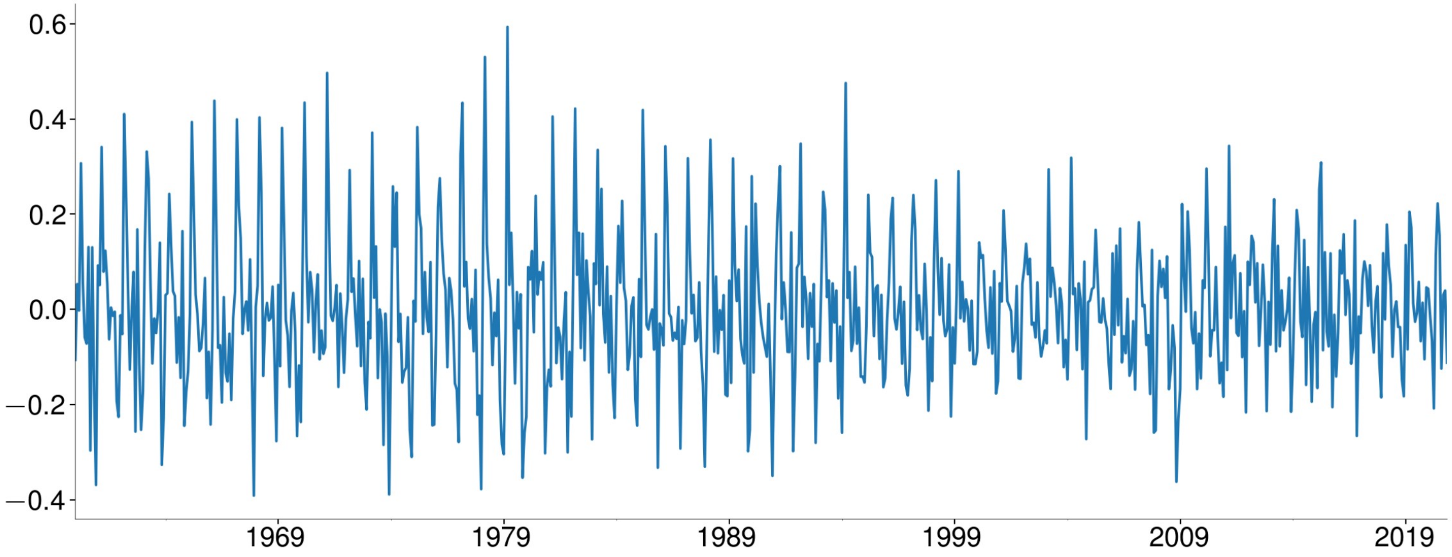
Univariate Time Series Analysis

Kevin Sheppard

Seasonality

- A seasonal time series has a deterministic pattern that repeats on an annual basis

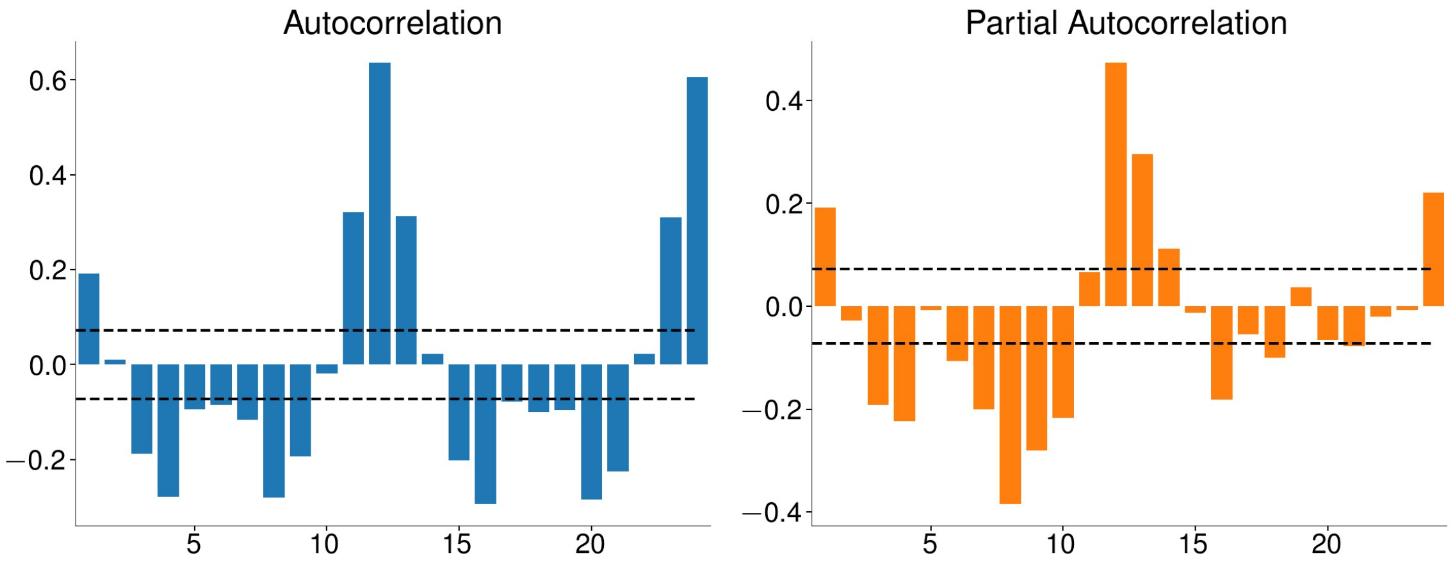
In [5]:
```
plot(housing, y=-2)
```



Seasonal Autocorrelation Pattern

- Seasonal data has dynamics at the annual frequency

In [6]:
```
acf_pacf_plot(housing, 24, size=-2)
```



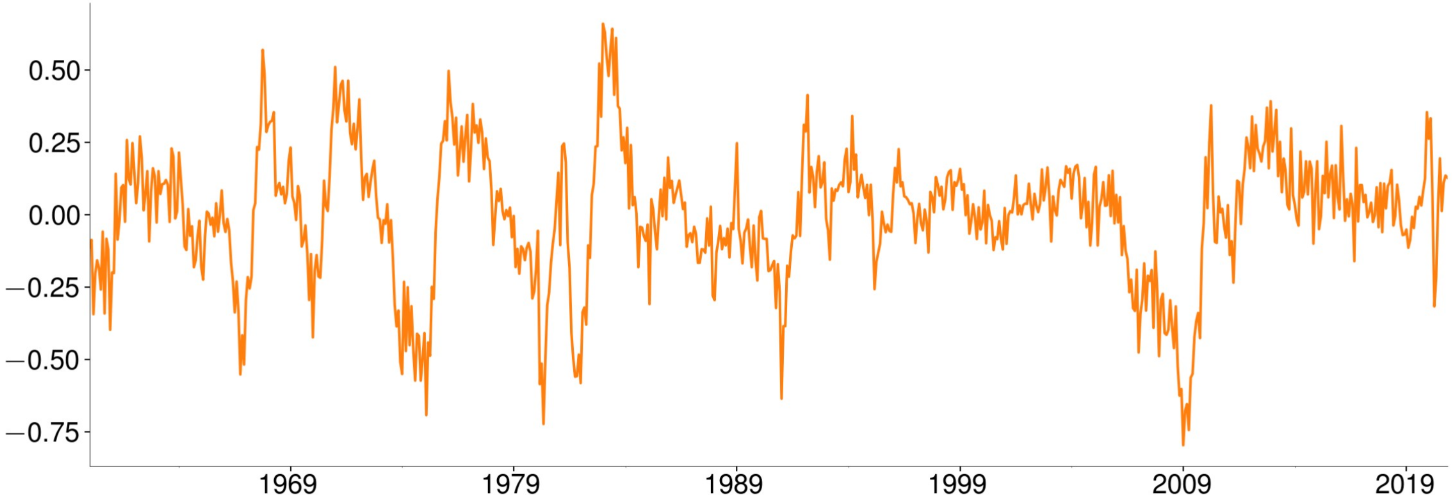
Seasonal Differencing

- Seasonal differencing differences using the seasonal period s

$$\Delta_s Y_t = Y_t - Y_{t-s}$$

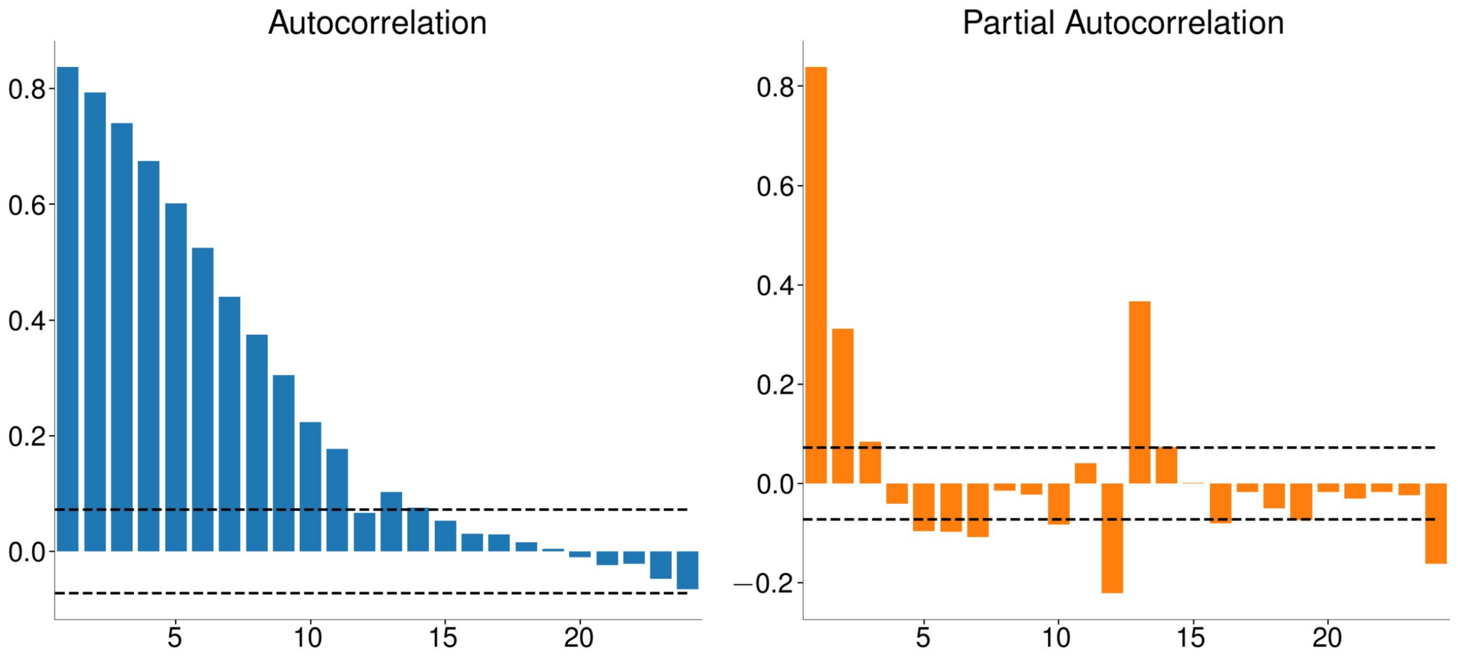
- Can reduce or eliminate seasonalities

In [7]:
```
plot(housing_yoy, y=12)
```



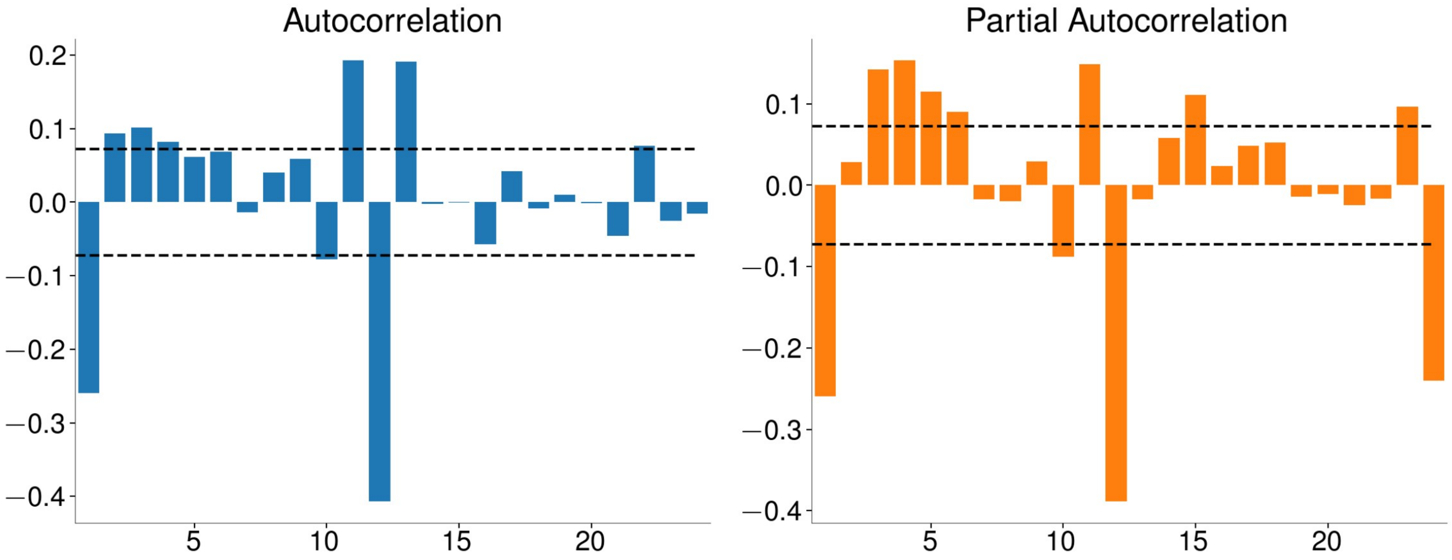
Seasonal Difference Autocorrelation Pattern

In [8]:
```
acf_pacf_plot(housing_yoy, 24)
```



Modeling Seasonal Differences

In [9]:
```
res = SARIMAX(housing_yoy, order=(1, 0, 0)).fit()
resids = res.resid.iloc[1:]
acf_pacf_plot(resids, 24, size=-2)
```



Seasonal ARMA Models

- SARMA models data at both the observational and seasonal frequency
- In lag polynomial representation

$$(1-\phi_1 L-\phi_2 L^2)Y_t=\phi_0+(1+\theta_s L^s)\epsilon_t$$

- In standard ARMA representation

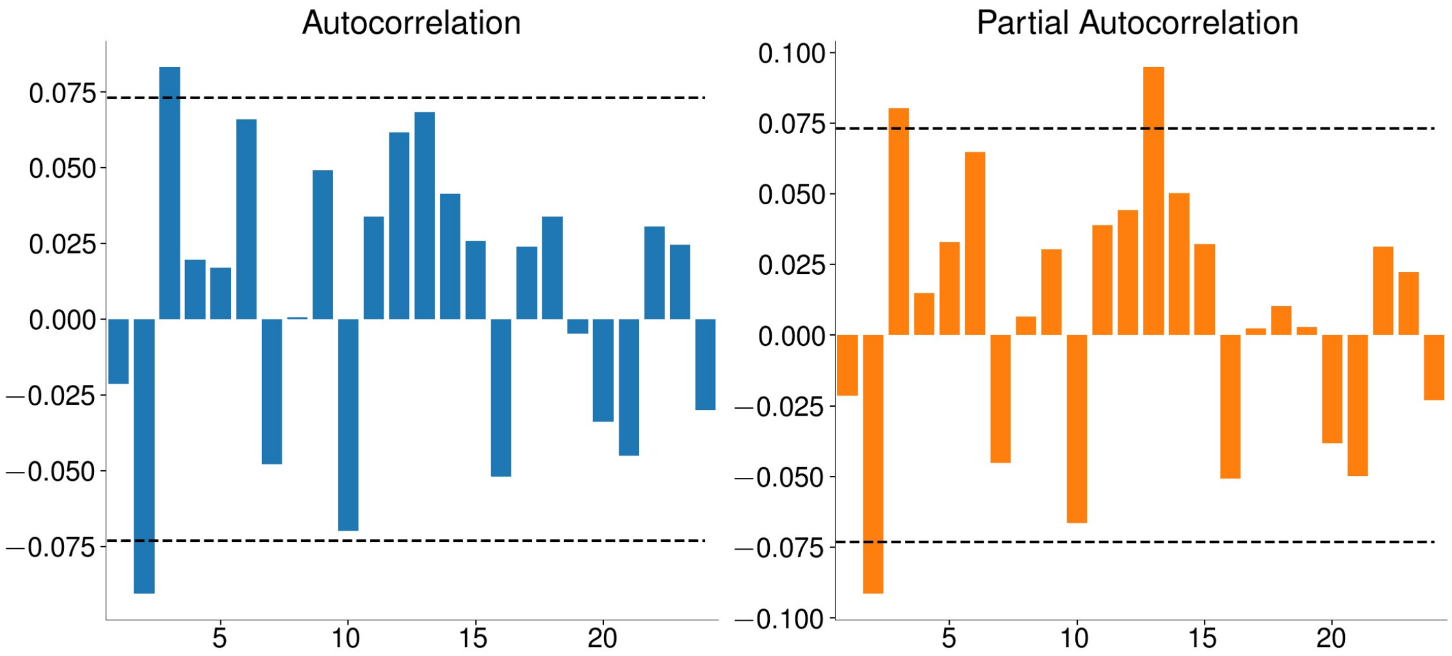
$$Y_t=\phi_0+\phi_1 Y_{t-1}+\phi_2 Y_{t-2}\theta_s\epsilon_{t-s}+\epsilon_t$$
$$SARMA\$(P,0,Q)\times(P_s,0,Q_s,s)$$

In [10]:
```
res = SARIMAX(housing_yoy, order=(2, 0, 0), seasonal_order=(0, 0, 1, 12)).fit()
summary(res)
```

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	0.6809	0.034	20.284	0.000	0.615	0.747
ar.L2	0.2824	0.034	8.233	0.000	0.215	0.350
ma.S.L12	-0.8795	0.017	-50.520	0.000	-0.914	-0.845
sigma2	0.0083	0.000	21.791	0.000	0.008	0.009

SARMA Residual Diagnostics

In [11]:
```
acf_pacf_plot(res.resid.iloc[13:], 24)
```



Extended Dynamics $(1,0,1) \times (1,0,1)_{12}$

- In lag polynomial representation

$$(1-\phi_1 L)(1-\phi_s L^s)Y_t = \phi_0 + (1+\theta_1 L)(1+\theta_s L^s)\epsilon_t$$

- In standard ARMA representation

$$Y_t = \phi_0 + \phi_1 Y_{t-1} + \phi_s Y_{t-s} - \phi_1\phi_s Y_{t-s-1} + \theta_1\epsilon_{t-1} + \theta_s\epsilon_{t-s} + \theta_1\theta_s\epsilon_{t-s-1} + \epsilon_t$$

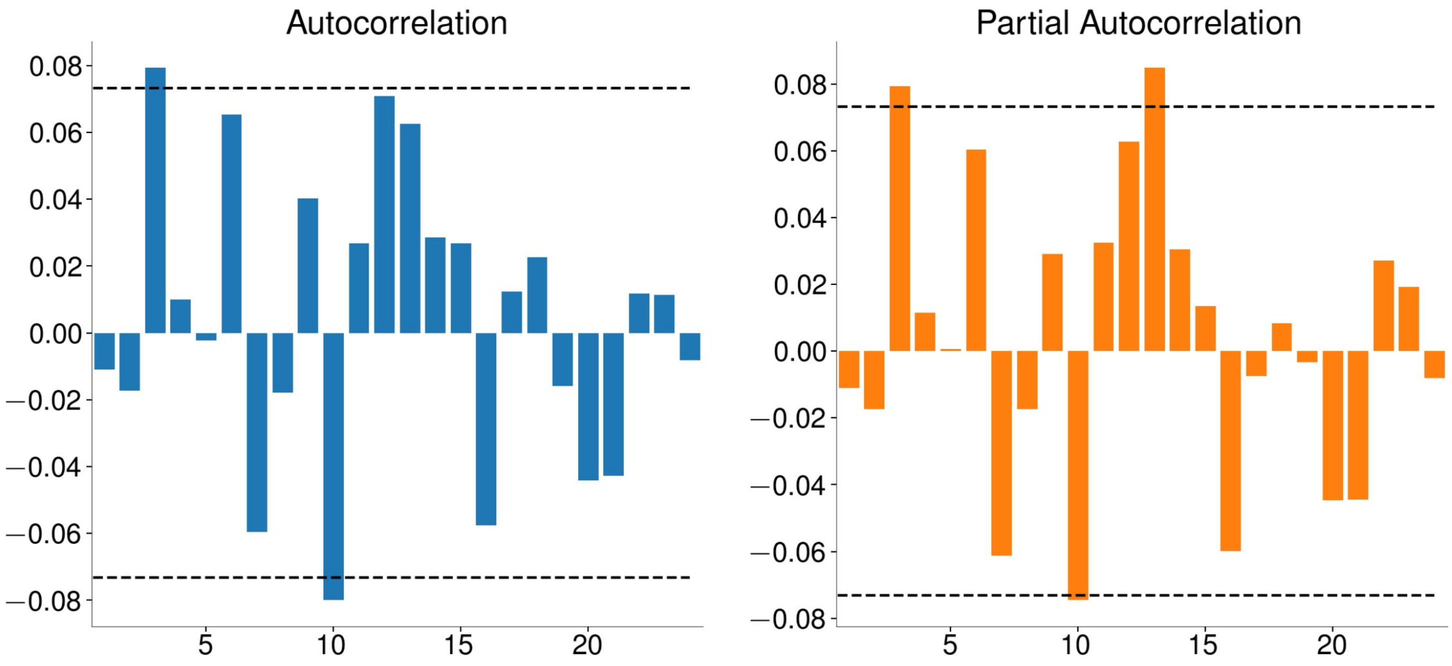
In [12]:

```
res = SARIMAX(housing_raw, order=(1, 0, 1), seasonal_order=(1, 0, 1, 12)).fit()
summary(res)
```

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	0.9994	0.001	1441.218	0.000	0.998	1.001
ma.L1	-0.3196	0.031	-10.367	0.000	-0.380	-0.259
ar.S.L12	0.9987	0.001	1469.955	0.000	0.997	1.000
ma.S.L12	-0.8957	0.016	-54.402	0.000	-0.928	-0.863
sigma2	0.0083	0.000	22.277	0.000	0.008	0.009

Extended Model Residual Diagnostics

In [13]:
```
acf_pacf_plot(res.resid.iloc[13:], 24)
```



Incorporating differencing

- SARIMA - Seasonal Autoregressive Integrated Moving Average

$$\mathrm{SARIMA}(P, D, Q) \times (P_s, D_s, Q_s, s)$$

$$\Phi(L)\Phi_s(L)(1-L)^D(1-L^s)^{D_s}Y_t = \phi_0 + \Theta(L)\Theta_s(L)\epsilon_t$$

- **AR**

$$\Phi(L) = \$1 - \phi_1 L - \phi_2 L^2 - \ldots - \phi_P L^P$$

- **Seasonal AR**:

$$\Phi_s(L) = 1 - \phi_s L^s - \phi_{2s} L^{2s} - \ldots - \phi_{P_s s} L^{P_s s}$$

- **MA**:

$$\Theta(L) = 1 + \theta_1 L + \theta_2 L^2 + \ldots + \phi_Q L^Q$$

- **Seasonal MA**:

$$\Theta_s(L) = 1 + \theta_s L^s + \theta_{2s} L^{2s} + \ldots + \theta_{Q_s s} L^{Q_s s}$$

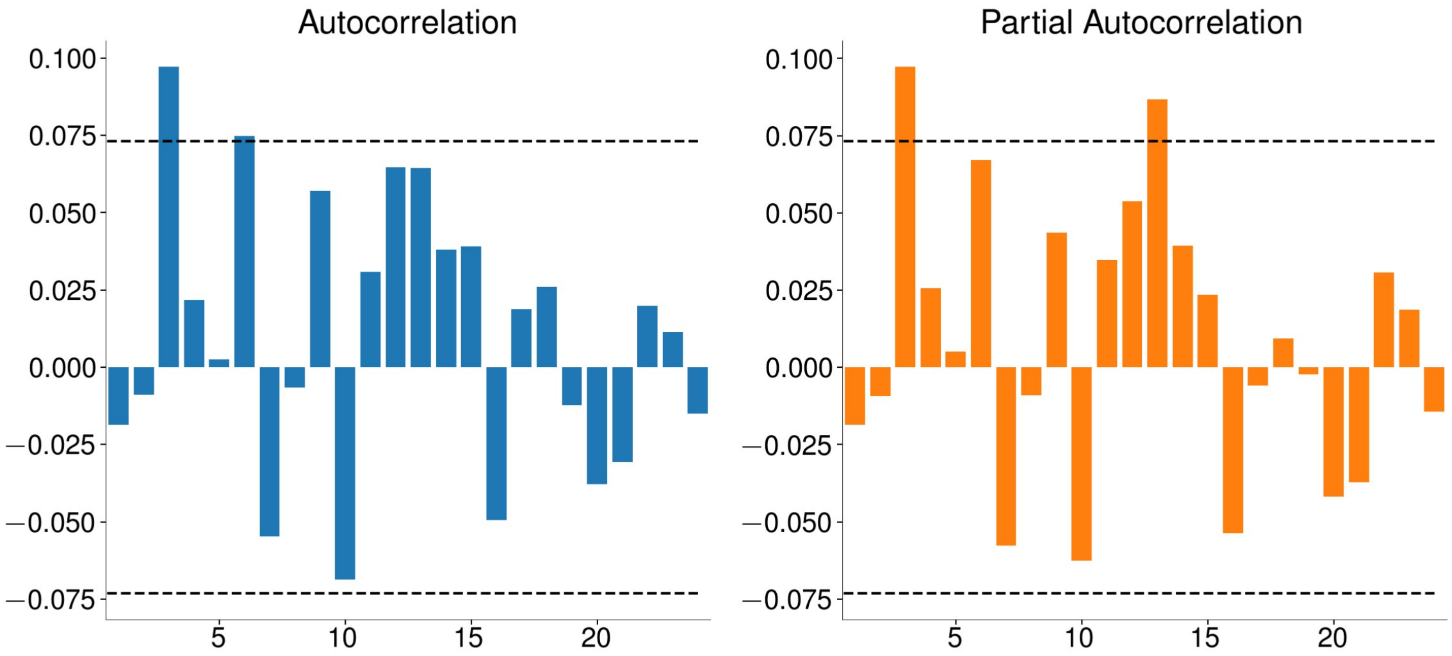
Estimating SARIMA Models

In [14]:
```
res = SARIMAX(housing_raw, order=(1, 0, 1), seasonal_order=(0, 1, 1, 12)).fit()
summary(res)
```

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	0.9777	0.008	126.932	0.000	0.963	0.993
ma.L1	-0.3128	0.033	-9.352	0.000	-0.378	-0.247
ma.S.L12	-0.8775	0.018	-48.061	0.000	-0.913	-0.842
sigma2	0.0083	0.000	21.771	0.000	0.008	0.009

SARIMA$(1, 0, 1) \times (0, 1, 1)_{12}$ Residual Analysis

In [15]:
```
acf_pacf_plot(res.resid.iloc[13:], 24)
```



Enforcing Unit Roots

- Set both differences to 1
  - $D = 1$
  - $D_s = 1$
- Model is

$$(1 - L)(1 - L^s)Y_t = \theta_s \epsilon_{t-s} + \epsilon_t$$
$$\Delta\Delta_s Y_t = \theta_s \epsilon_{t-s} + \epsilon_t$$

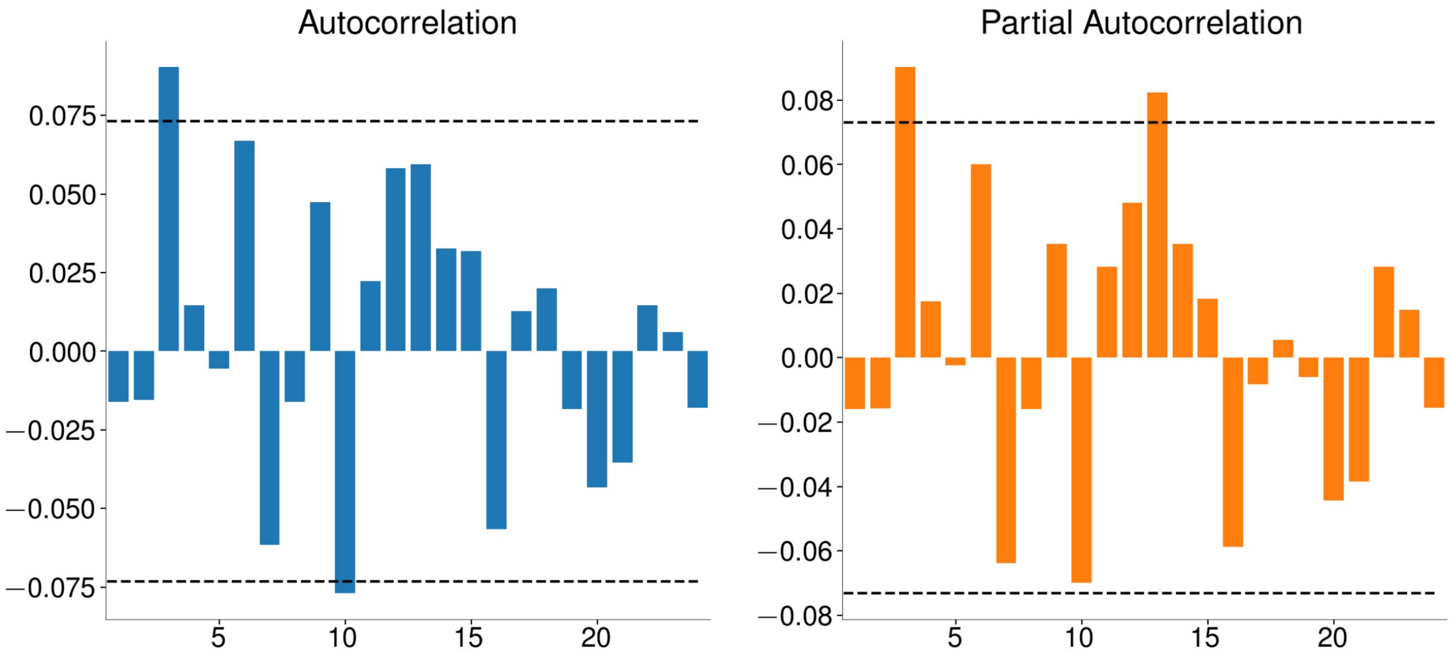
In [16]:
```
res = SARIMAX(housing_raw, order=(0, 1, 1), seasonal_order=(0, 1, 1, 12)).fit()
summary(res)
```

	coef	std err	z	P>|z|	[0.025	0.975]
ma.L1	-0.3246	0.031	-10.362	0.000	-0.386	-0.263
ma.S.L12	-0.8774	0.019	-47.035	0.000	-0.914	-0.841
sigma2	0.0084	0.000	22.025	0.000	0.008	0.009

Double Difference Residual Diagnostics

In [17]:

```
acf_pacf_plot(res.resid.iloc[13:], 24)
```



Seasonal Dummies

- Alternative approach to SARIMA
- Model shifts as a deterministic processes
- One dummy for each of $s - 1$ periods
  - Avoid dummy variable trap

In [18]:

```
month = housing_raw.index.strftime("%b")
dummies = pd.get_dummies(month)
dummies = dummies[
    ["Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"]
]
dummies.index = housing_raw.index
dummies.head()
```

Out[18]:

	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
1960-01-01	0	0	0	0	0	0	0	0	0	0	0
1960-02-01	1	0	0	0	0	0	0	0	0	0	0
1960-03-01	0	1	0	0	0	0	0	0	0	0	0
1960-04-01	0	0	1	0	0	0	0	0	0	0	0
1960-05-01	0	0	0	1	0	0	0	0	0	0	0

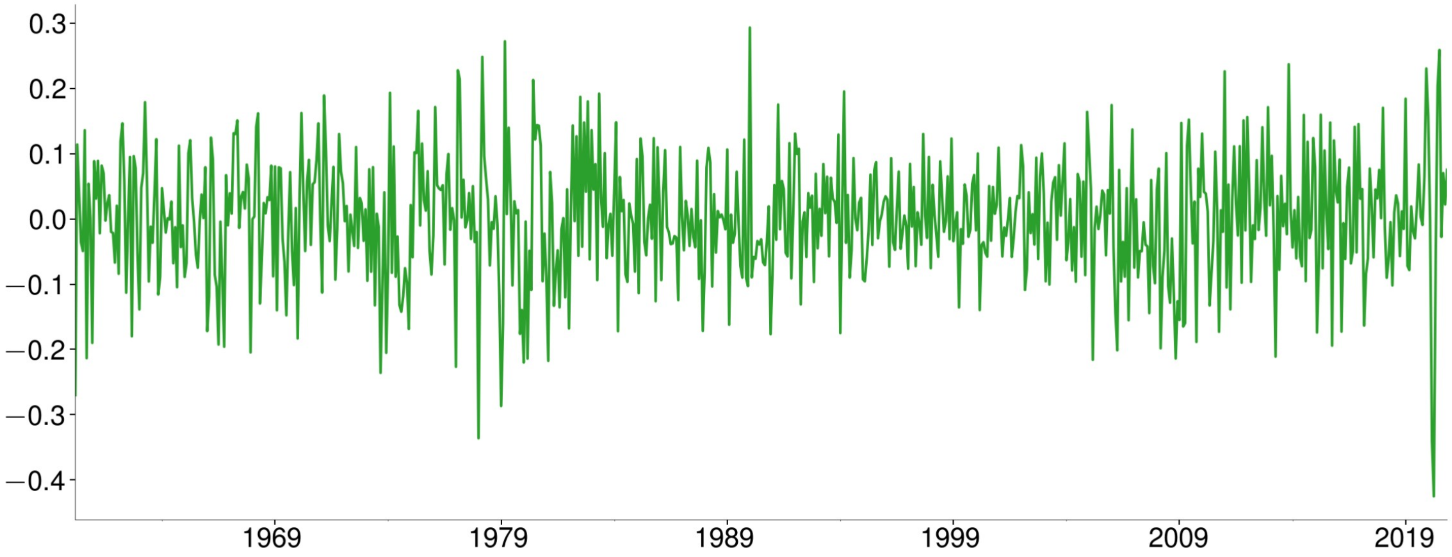
Incorporating Dummies

In [19]:
```
res = SARIMAX(housing_raw, order=(2, 1, 0), exog=dummies, trend="c").fit()
summary(res)
```

	coef	std err	z	P>|z|	[0.025	0.975]
intercept	0.0002	0.004	0.048	0.962	-0.007	0.008
Feb	0.0358	0.012	2.965	0.003	0.012	0.059
Mar	0.3075	0.012	24.777	0.000	0.283	0.332
Apr	0.4289	0.015	29.516	0.000	0.400	0.457
May	0.4669	0.018	26.260	0.000	0.432	0.502
Jun	0.4697	0.019	25.308	0.000	0.433	0.506
Jul	0.4328	0.019	23.265	0.000	0.396	0.469
Aug	0.4117	0.019	22.227	0.000	0.375	0.448
Sep	0.3657	0.017	21.802	0.000	0.333	0.399
Oct	0.3921	0.015	26.252	0.000	0.363	0.421
Nov	0.2169	0.013	16.942	0.000	0.192	0.242
Dec	0.0502	0.010	5.240	0.000	0.031	0.069
ar.L1	-0.2675	0.033	-8.115	0.000	-0.332	-0.203
ar.L2	-0.1107	0.034	-3.276	0.001	-0.177	-0.044
sigma2	0.0087	0.000	21.344	0.000	0.008	0.010

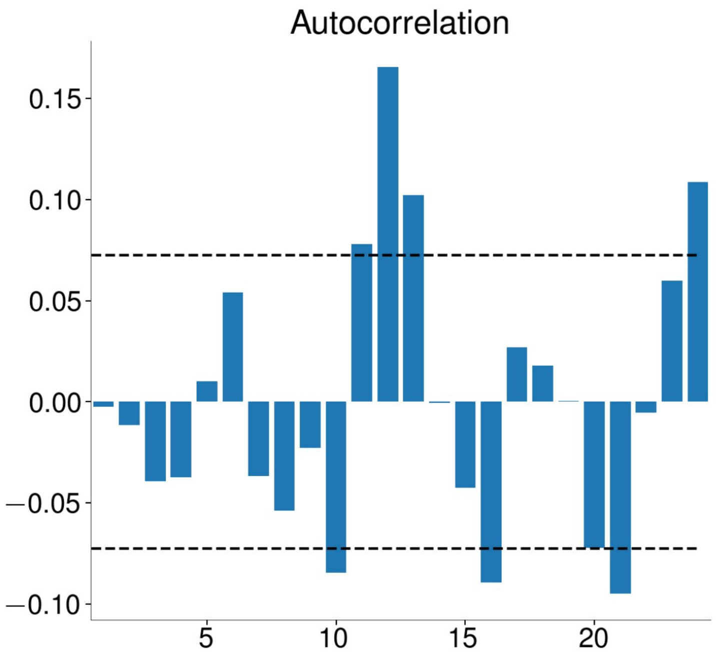
Seasonal Dummy Model Residuals

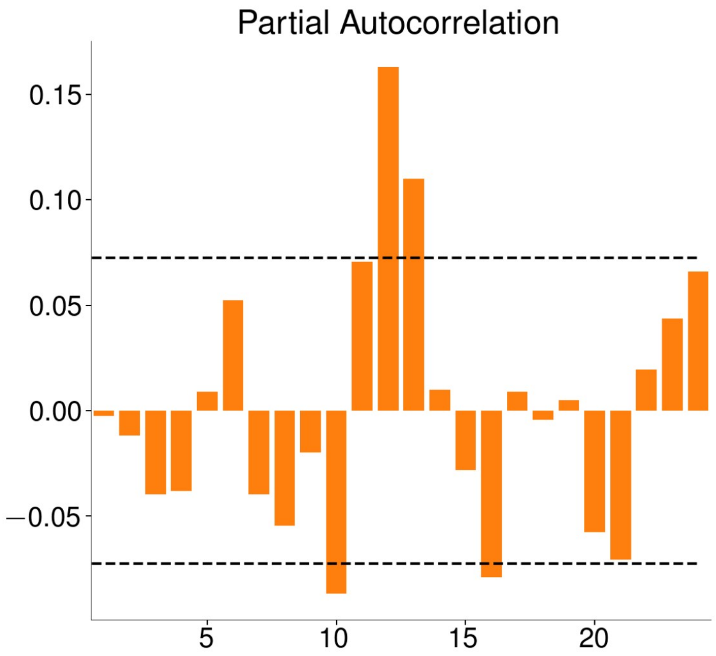
In [20]:
```
resid = res.resid.iloc[2:]
plot(resid, y=-2)
```



Seasonal Dummy Model Residuals Diagnostics

In [21]:
```
acf_pacf_plot(resid, 24)
```





Unit Root Testing

- Test Y_t for a unit root
- If null is not rejected, ensure that all required deterministic terms are included
- If null is still *not* rejected, difference and test ΔY_t
- Positive Y_t series should usually be transformed with $\ln$
- Select lag length of ΔY_{t-j} terms using IC, usually AIC
- $H_0 : \gamma = 0, H_1 : \gamma < 0$

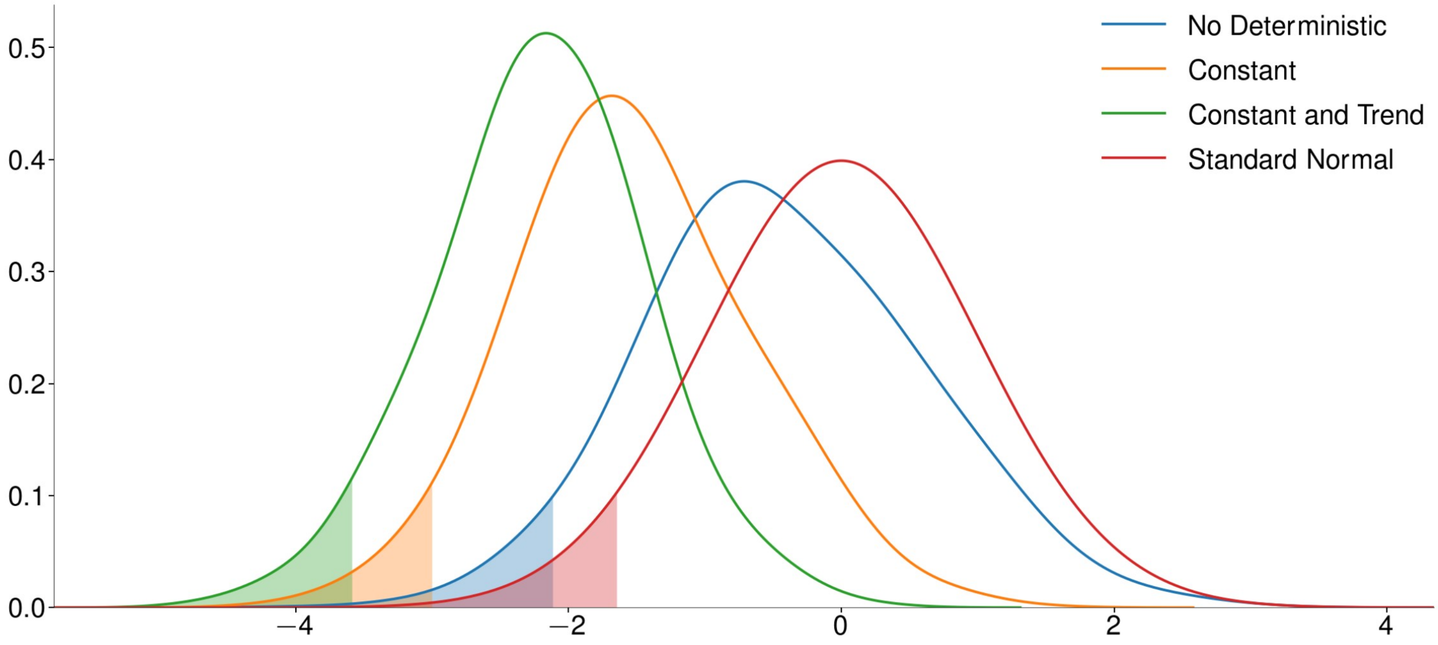
$$Y_t = \delta_0 + \delta_1 t + \gamma Y_{t-1} + \sum_{i=1}^{P} \Delta Y_{t-i} + \epsilon_t$$

- Deterministic terms are needed to remove deterministic components from Y_{t-1}

The Dickey-Fuller distributions

In [23]:

```
adf_cv_plot()
```



Testing Default Premium

In [24]:
```
from arch.unitroot import ADF

adf = ADF(default)
adf.summary()
```

Out[24]:

Augmented Dickey-Fuller Results

Test Statistic	-3.866
P-value	0.002
Lags	10

Trend: Constant
Critical Values: -3.44 (1%), -2.87 (5%), -2.57 (10%)
Null Hypothesis: The process contains a unit root.
Alternative Hypothesis: The process is weakly stationary.

Testing Curvature

In [25]:

```
adf = ADF(curve)
adf.summary()
```

Out[25]:

Augmented Dickey-Fuller Results

Test Statistic	-4.412
P-value	0.000
Lags	19

Trend: Constant
Critical Values: -3.44 (1%), -2.87 (5%), -2.57 (10%)
Null Hypothesis: The process contains a unit root.
Alternative Hypothesis: The process is weakly stationary.

Testing Industrial Production

In [26]:
```
adf = ADF(np.log(orig.INDPRO), trend="c")
adf.summary()
```

Out[26]:

Augmented Dickey-Fuller Results

Test Statistic	-2.186
P-value	0.211
Lags	4

Trend: Constant
Critical Values: -3.44 (1%), -2.87 (5%), -2.57 (10%)
Null Hypothesis: The process contains a unit root.
Alternative Hypothesis: The process is weakly stationary.

ADF Regression Results

In [27]:

```
summary(adf.regression)
```

	coef	std err	t	P>|t|	[0.025	0.975]
Level.L1	-0.0017	0.001	-2.186	0.029	-0.003	-0.000
Diff.L1	0.3637	0.037	9.860	0.000	0.291	0.436
Diff.L2	-0.1110	0.039	-2.832	0.005	-0.188	-0.034
Diff.L3	0.0447	0.039	1.138	0.255	-0.032	0.122
Diff.L4	0.0217	0.037	0.589	0.556	-0.051	0.094
const	0.0083	0.003	2.579	0.010	0.002	0.015

Increasing the deterministic order

In [28]:
```
adf = ADF(np.log(orig.INDPRO), trend="ct")
adf
```

Out[28]:

Augmented Dickey-Fuller Results

Test Statistic	-1.831
P-value	0.690
Lags	6

Trend: Constant and Linear Time Trend
Critical Values: -3.97 (1%), -3.42 (5%), -3.13 (10%)
Null Hypothesis: The process contains a unit root.
Alternative Hypothesis: The process is weakly stationary.

Testing the difference

- The Industrial Productivity Index is $I(1)$ since 1 difference makes it stationary

In [29]:
```
delta_indpro = np.log(orig.INDPRO).diff().dropna()
adf = ADF(delta_indpro, trend="c")
adf
```

Out[29]:

Augmented Dickey-Fuller Results

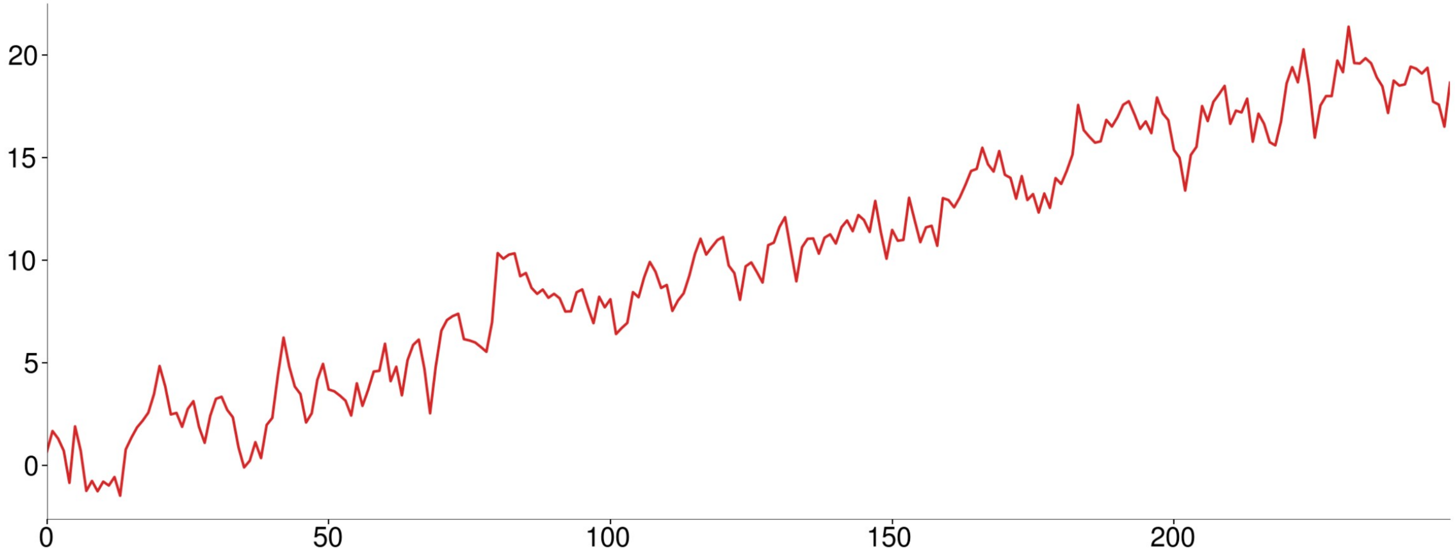
Test Statistic	-11.945
P-value	0.000
Lags	3

Trend: Constant
Critical Values: -3.44 (1%), -2.87 (5%), -2.57 (10%)
Null Hypothesis: The process contains a unit root.
Alternative Hypothesis: The process is weakly stationary.

The importance of time trends

- Incorrect trend specification results in no power to reject false nulls
- Always find a unit root in trending time series

In [31]:
```
plot(trending, 13)
```



Under-specifying Deterministic Terms

- Do not use model with no deterministic terms on real data unless Y_t is known to be mean 0

In [32]:
```
ADF(trending, trend="n")
```

Out[32]:

Augmented Dickey-Fuller Results

Test Statistic	1.934
P-value	0.988
Lags	9

Trend: No Trend
Critical Values: -2.58 (1%), -1.94 (5%), -1.62 (10%)
Null Hypothesis: The process contains a unit root.
Alternative Hypothesis: The process is weakly stationary.

Under-specifying Deterministic Terms

In [33]:
```
ADF(trending, trend="c")
```

Out[33]:

Augmented Dickey-Fuller Results

Test Statistic	-1.146
P-value	0.696
Lags	9

Trend: Constant
Critical Values: -3.46 (1%), -2.87 (5%), -2.57 (10%)
Null Hypothesis: The process contains a unit root.
Alternative Hypothesis: The process is weakly stationary.

Correctly specifying Deterministic Terms

In [34]:

```
ADF(trending, trend="ct")
```

Out[34]:

Augmented Dickey-Fuller Results

Test Statistic	-6.790
P-value	0.000
Lags	0

Trend: Constant and Linear Time Trend
Critical Values: -4.00 (1%), -3.43 (5%), -3.14 (10%)
Null Hypothesis: The process contains a unit root.
Alternative Hypothesis: The process is weakly stationary.

Over-specifying Deterministic Terms

In [35]:

```
ADF(y, trend="ctt")
```

Out[35]:

Augmented Dickey-Fuller Results

Test Statistic	-6.885
P-value	0.000
Lags	0

Trend: Constant, Linear and Quadratic Time Trends
Critical Values: -4.42 (1%), -3.86 (5%), -3.57 (10%)
Null Hypothesis: The process contains a unit root.
Alternative Hypothesis: The process is weakly stationary.

Incorporating Time Trends into ARMA Models

- Can jointly estimate trends and stationary components of ARMA models

$$Y_t = \phi_0 + \delta_1 t + \phi_1 Y_{t-1} + \epsilon_t$$

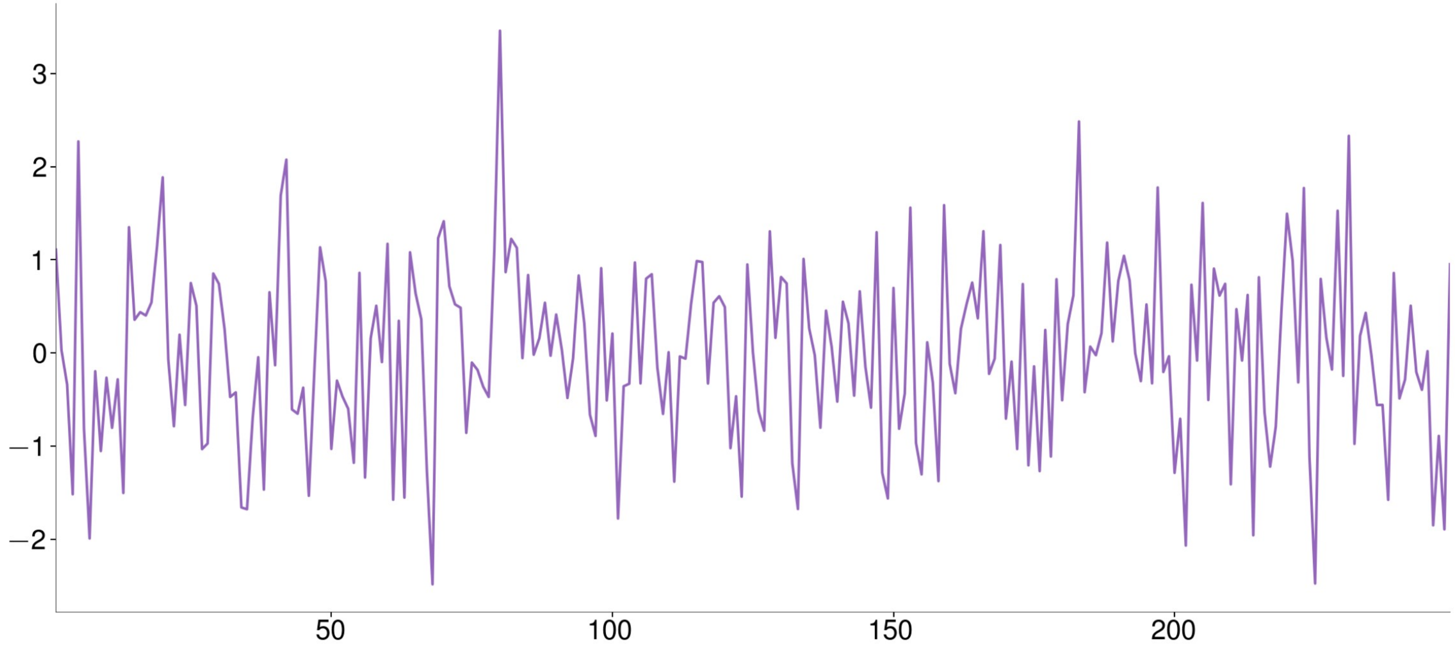
In [36]:
```
res = SARIMAX(trending, order=(1, 0, 0), trend="ct").fit()
summary(res)
```

	coef	std err	z	P>|z|	[0.025	0.975]
intercept	0.0796	0.116	0.687	0.492	-0.148	0.307
drift	0.0255	0.004	6.149	0.000	0.017	0.034
ar.L1	0.6822	0.050	13.534	0.000	0.583	0.781
sigma2	0.9213	0.080	11.451	0.000	0.764	1.079

Time Trend Model Residuals

In [37]:

```
plot(res.resid.iloc[1:])
```



Next Week

- Univariate Volatility Modeling