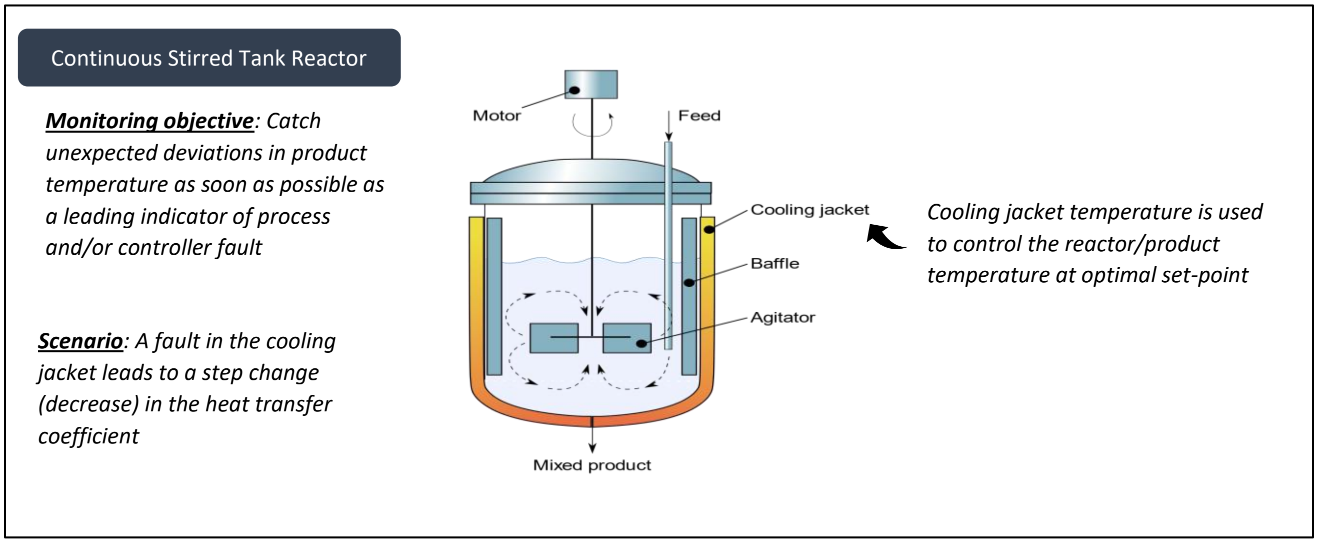
Monitoring Controlled Variables in a CSTR using Univariate Control Charts for Time Series Model Residuals

To illustrate an application of time series model-based univariate SPM, we will consider the continuous stirred-tank reactor. Here, the product temperature will be monitored to ensure it is not deviating away from its optimal setpoint due to any process fault. The datafile *CSTR_controlledTemperature.csv* contains 4 hours of temperature measurements[1] sampled at 0.1 minutes (totaling 2401 samples). A process fault occurs at 3.5 hours (around sample # 2100) causing a 20% decrease in the heat transfer coefficient. Let's see if this process fault causes any deviations in the product temperature and if our monitoring methodology can detect these deviations.


Continuous Stirred Tank Reactor

***Monitoring objective**: Catch unexpected deviations in product temperature as soon as possible as a leading indicator of process and/or controller fault*

***Scenario**: A fault in the cooling jacket leads to a step change (decrease) in the heat transfer coefficient*

Cooling jacket temperature is used to control the reactor/product temperature at optimal set-point

Figure: Process fault detection using TSA: case-study set-up[2]

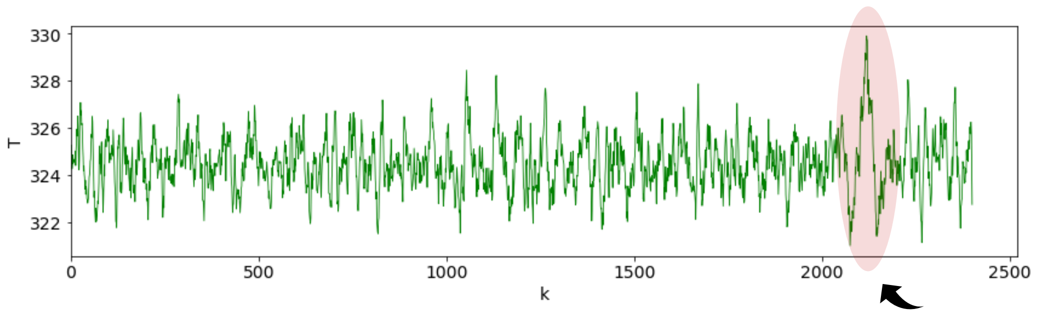
Let's start by importing the required packages and exploring the provided I/O dataset. The first 3 hours of data will be used to build a model and the last 1 hour will be used as test data.

[1] The CSTR model provided at https://github.com/APMonitor/pdc/blob/master/CSTR_Control.ipynb is used to generate the data.

[2] CSTR diagram created by Daniele Pugliesi under Creative Commons Attribution-Share Alike 3.0, https://commons.wikimedia.org/wiki/File:Agitated_vessel.svg

```
# import packages
import matplotlib.pyplot as plt, numpy as np, control
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.tsa.stattools import arma_order_select_ic
from statsmodels.tsa.arima.model import ARIMA

# read data and split into training and test data
T = np.loadtxt('CSTR_controlledTemperature.csv', delimiter=',')
T_train = T[:1800]; T_test = T[1800:]
```

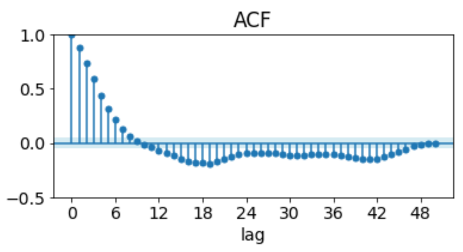


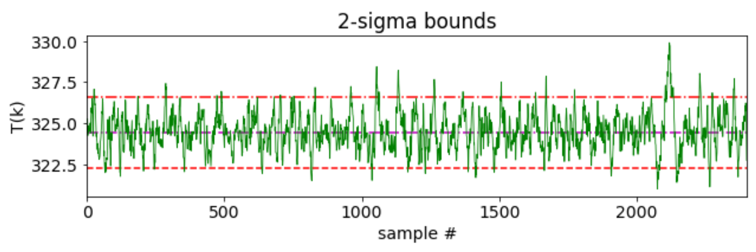
Short-lived deviation in absolute value at start of the process-fault

The plot above shows that the temperature seems to show only a short-lived[3] deviation in its absolute value due to the process fault. Therefore, on the basis of this time-plot alone, plant operators could very well ignore this as an unimportant fluctuation and the process fault could go unattended.

The trivial univariate monitoring approach of using $mean \pm 2 * standard\ deviation$ bounds would also be inadequate because this approach assumes that the samples are independent of each-other, which as shown by the below ACF plot of *T_train* data, is certainly not true for the product outlet temperature (and dynamic processes in general). The control chart shown below shows the failure of this approach as the controlled variable does not show any sustained unusual violations of the control bounds (bounds are computed using the training data only).

[3] The controller adjusts the jacket temperature to compensate for the reduction in the heat transfer coefficient.





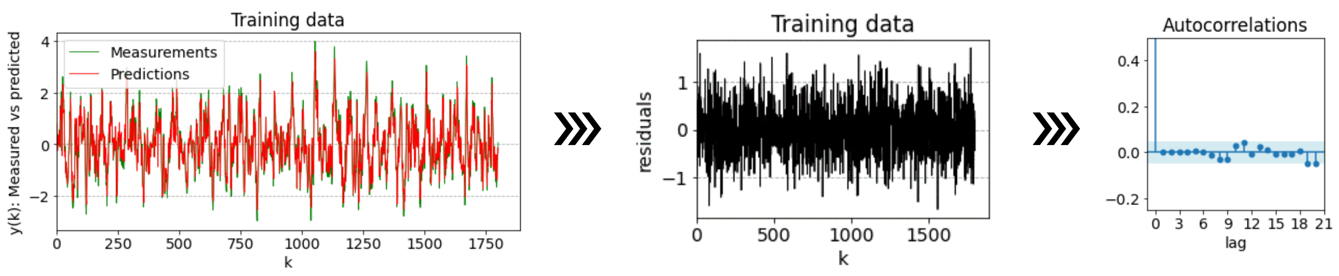
Let's now see if using time-series models can do a better job of fault detection. We will attempt to fit an ARMA model.

```
# Determine the optimal AR and MA orders
T_train_centered = T_train - np.mean(T_train); T_test_centered = T_test - np.mean(T_train)
res = arma_order_select_ic(T_train_centered, max_ar=5, max_ma=5, ic=["aic"])
p, r = res.aic_min_order
print('(p, r) = ', res.aic_min_order)

>>> (p, r) =  (2, 4)

# Fit an ARMA(p,r) model and get residuals
model = ARIMA(T_train_centered, order=(p, 0, r))
results = model.fit()
T_train_centered_pred = results.fittedvalues # same as results.predict()
residuals_train = T_train_centered - T_train_centered_pred # same as results.resid
```

The residuals pass the model quality check.



It is time now to use the fitted ARMA model for fault detection. The rationale is straight-forward: if any process-fault leads to changes in process behavior then the model fitted using 'normal' operating condition data won't be very accurate with faulty data and therefore, the residuals will show higher values. Consequently, the residuals are monitored for fault detection as summarized in Figure below.

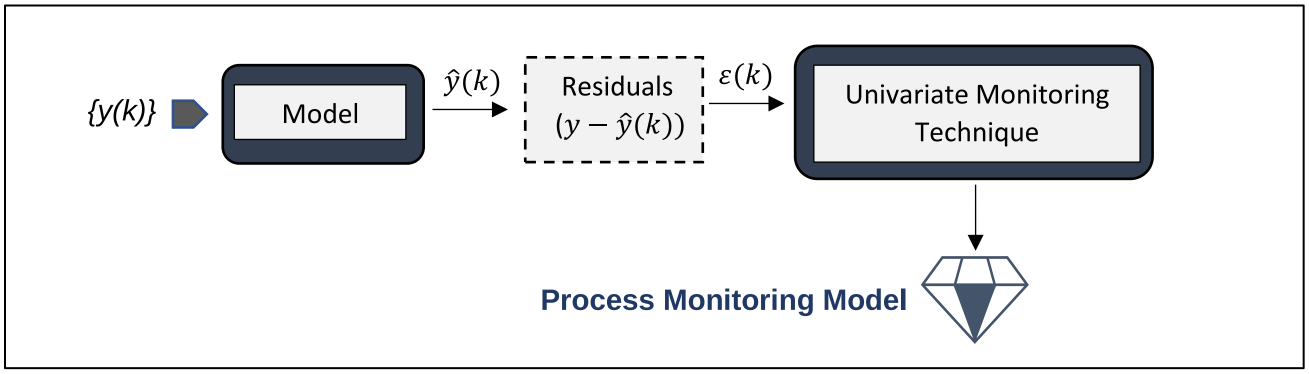
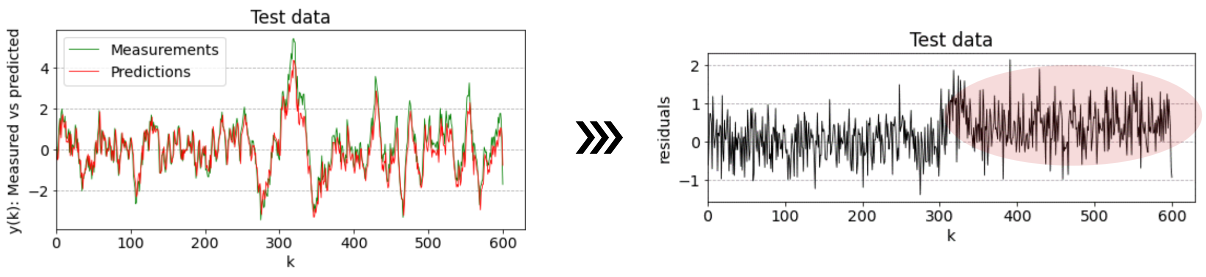
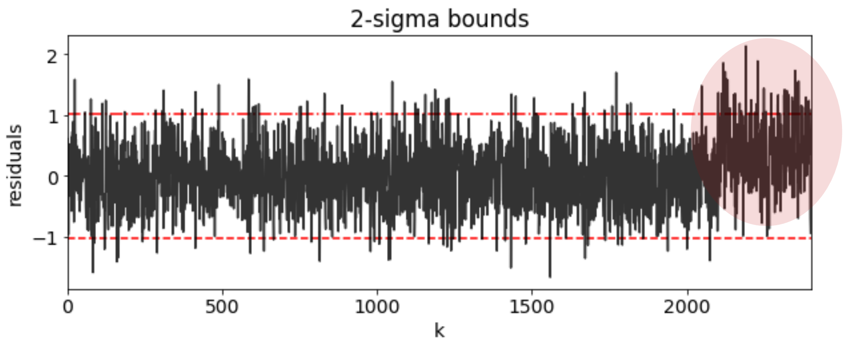


Figure: Process monitoring methodology via monitoring of time-series analysis-based residuals.

```
# 1-step ahead predictions for test data
results_test = results.apply(T_test_centered)
T_test_centered_pred = results_test.fittedvalues # same as results_test.predict()
residuals_test = T_test_centered - T_test_centered_pred
```



The above plots show that the residuals exhibit an increase in the mean value after the onset of process fault (after sample # 300 for test dataset). The control chart for the whole dataset makes the sustained unusual values of the residuals more evident. The sustained deviations in the residuals would be more prominent in a CUSUM control chart


2-sigma bounds
residuals
k