`AutoPlot` Modules

Global variables

```
# Global variables
line_color = '#3f51b5'      # line (trace1) color for any plot
marker_color = '#43a047'    # marker color for any plot
marker_border_color = '#ffffff'     # marker border color for any plot
cl_color = '#ffa000'    # control limit line color for any plot
sl_color = '#e53935'    # spec limit line color for any plot
cp_plot_title = 'CP Plot for RESP1B'  # title for CP plot
cp_plot_xlabel = 'Date'   # xaxis name for CP plot
cp_plot_ylabel = 'delta CP (no.s)'     # yaxis name for CP plot
cp_plot_html_file = 'RESP1B_CP-Plot.html'   # HTML filename for CP plot
cp_plot_trace_count = 2    # no. of traces in CP plot
er_barc_plot_title = 'BARC ER Plot for RESP1B'  # title for ER plot
er_barc_plot_xlabel = 'Date'        # xaxis name for ER plot
er_barc_plot_ylabel = 'BARC ER (A/min)'   # yaxis name for ER plot
er_barc_plot_html_file = 'RESP1B_BARC_ER-Plot.html'   # HTML filename for ER plot
er_barc_plot_trace_count = 5    # no. of traces in ER plot
unif_barc_plot_title = 'BARC Uniformity Plot for RESP1B'  # title for Unif plot
unif_barc_plot_xlabel = 'Date'      # xaxis name for Unif plot
unif_barc_plot_ylabel = 'BARC Unif (%)'    # yaxis name for Unif plot
unif_barc_plot_html_file = 'RESP1B_BARC_Unif-Plot.html'   # HTML filename for Unif plot
unif_barc_plot_trace_count = 2    # no. of traces in Unif plot
sht_cp_columns = ["Date (MM/DD/YYYY)", "delta CP", "USL", "Remarks"]    # Columns for CP Dataframe
sht_er_barc_columns = ["Date (MM/DD/YYYY)", "Layer", "Etch Rate (A/Min)", "% Uni", "Remarks", "LSL", "USL", "LCL", "UCL", "% Uni

# Excel file directory
excel_file_directory = "\\\\vmfg\\VFD FILE SERVER\\SECTIONS\\DRY ETCH\\QC Log Book\\Final QC Log Book\\CNT_01_LOG_BOOK\\CNT01_QC_
```

CP Plot

```
"""
"Description": This function plots CP Chart with traces v/s Date.
"draw_plotly_resp1b_cp_plot": Draw Plotly's Plot for RESP1B CP
"x": Date (x-axis) for CP Chart
"y1": Delta-CP (y-axis) for CP Chart
"y2": USL (y-axis) for CP Chart
# "y3": UCL (y-axis) for CP Chart
"""
def draw_plotly_resp1b_cp_plot(x, y1, y2, remarks):
    trace1 = go.Scatter(
            x = x,
            y = y1,
            name = 'delta-CP',
            mode = 'lines+markers',
            line = dict(
                    color = line_color,
                    width = 2),
            marker = dict(
                    color = marker_color,
                    size = 8,
                    line = dict(
                        color = marker_border_color,
                        width = 0.5),
                    ),
            text = remarks
    )

    trace2 = go.Scatter(
            x = x,
            y = y2,
            name = 'USL',
            mode = 'lines',
```

```
            line = dict(
                    color = sl_color,
                    width = 3)
    )

    trace3 = go.Scatter(
            x = x,
            y = y3,
            name = 'UCL',
            mode = 'lines',
            line = dict(
                    color = cl_color,
                    width = 3)
    )

    data = [trace1, trace2, trace3]
    layout = dict(
            title = cp_plot_title,
            xaxis = dict(title= cp_plot_xlabel),
            yaxis = dict(title= cp_plot_ylabel)
        )
    fig = dict(data= data, layout= layout)
    py.offline.plot(fig, filename= cp_plot_html_file)
```

ER Plot

```
"""
"Description": This function plots ER Chart with traces v/s Date.
"draw_plotly_resp1b_er_barc_plot": Draw Plotly's Plot for RESP1B BARC ER
"x": Date (x-axis) for ER Chart
"y1": ER (y-axis) for ER Chart
"y2": USL (y-axis) for ER Chart
"y3": LSL (y-axis) for ER Chart
"y4": UCL (y-axis) for ER Chart
"y5": LCL (y-axis) for ER Chart
"""
def draw_plotly_resp1b_er_barc_plot(x, y1, y2, y3, y4, y5, remarks):
    trace1 = go.Scatter(
            x = x,
            y = y1,
            name = 'ER',
            mode = 'lines+markers',
            line = dict(
                    color = line_color,
                    width = 2),
            marker = dict(
                    color = marker_color,
                    size = 8,
                    line = dict(
                        color = marker_border_color,
                        width = 0.5),
                    ),
            text = remarks
    )

    trace2 = go.Scatter(
            x = x,
            y = y2,
            name = 'USL',
            mode = 'lines',
            line = dict(
                    color = sl_color,
                    width = 3)
    )

    trace3 = go.Scatter(
            x = x,
            y = y3,
            name = 'LSL',
            mode = 'lines',
            line = dict(
```

```
                color = sl_color,
                width = 3)
    )

    trace4 = go.Scatter(
            x = x,
            y = y4,
            name = 'UCL',
            mode = 'lines',
            line = dict(
                    color = cl_color,
                    width = 3)
    )

    trace5 = go.Scatter(
            x = x,
            y = y5,
            name = 'LCL',
            mode = 'lines',
            line = dict(
                    color = cl_color,
                    width = 3)
    )

    data = [trace1, trace2, trace3, trace4, trace5]
    layout = dict(
            title = er_barc_plot_title,
            xaxis = dict(title= er_barc_plot_xlabel),
            yaxis = dict(title= er_barc_plot_ylabel)
        )
    fig = dict(data= data, layout= layout)
    py.offline.plot(fig, filename= er_barc_plot_html_file)
```

Unif Plot

```
"""
"Description": This function plots Unif Chart with traces v/s Date.
"draw_plotly_resp1b_unif_barc_plot": Draw Plotly's Plot for RESP1B BARC Unif
"x": Date (x-axis) for Unif Chart
"y1": Unif (y-axis) for Unif Chart
"y2": USL (y-axis) for Unif Chart
"y3": UCL (y-axis) for Unif Chart
"""
def draw_plotly_resp1b_unif_barc_plot(x, y1, y2, y3, remarks):
    trace1 = go.Scatter(
            x = x,
            y = y1,
            name = 'Unif',
            mode = 'lines+markers',
            line = dict(
                    color = line_color,
                    width = 2),
            marker = dict(
                    color = marker_color,
                    size = 8,
                    line = dict(
                        color = marker_border_color,
                        width = 0.5),
                    ),
            text = remarks
    )

    trace2 = go.Scatter(
            x = x,
            y = y2,
            name = 'USL',
            mode = 'lines',
            line = dict(
                    color = sl_color,
                    width = 3)
    )
```

```
trace3 = go.Scatter(
        x = x,
        y = y3,
        name = 'UCL',
        mode = 'lines',
        line = dict(
                color = cl_color,
                width = 3)
)

data = [trace1, trace2, trace3]
layout = dict(
        title = unif_barc_plot_title,
        xaxis = dict(title= unif_barc_plot_xlabel),
        yaxis = dict(title= unif_barc_plot_ylabel)
    )
fig = dict(data= data, layout= layout)
py.offline.plot(fig, filename= unif_barc_plot_html_file)
```