PI World 2018 Lab

Fit for Purpose - Layers of Analytics using the PI System – AF, MATLAB, Machine Learning

OSIsoft, LLC
1600 Alvarado Street
San Leandro, CA 94577 USA
Tel: (01) 510-297-5800
Web: http://www.osisoft.com

Published: April 20, 2018

Fit for purpose – Layers of Analytics – Hands-on Lab – OSIsoft PI World 2018

Lead: Gopal GopalKrishnan, P.E., Solution Architect

Lead: Curt Hertler, Principal Technical Advisor

Instructor: Ronald Bechaalany, Senior Systems Engineer

Table of Contents

Fit for Purpose- Layers of Analytics using the PI System – AF, MATLAB, Machine Learning

Lab Description

This hands-on lab covers scenarios to illustrate the different levels of analytics that are fit-for-purpose when using the PI System - for example, what calculations and analysis do you do in AF, when do you use MATLAB and similar libraries for advanced calculations that hook into AF and when do you call on "data science and machine learning ". Use cases will include those focused on an equipment i.e. pump or motor or compressor etc. and those focused on a process. We will also cover examples of advanced analytics that are part of the data collection from the edge devices and contrast it with predictive analytics – both the model development and model deployment (scoring new incoming real-time data).

Level: PI User Track – Power User   Duration: 3 hours

Introduction

Layers of analytics can be viewed through many lenses. It can refer to the levels of complexity and the kinds of computations required to transform "raw data" to "actionable information/insight." The complexity level varies due to:

- ***amount of data required*** for computation
- ***nature of the computation/calculation***, and
- ***frequency of the computation***

In the above, the ***nature of the calculation*** dictates the amount of data required (a single measurement or a few measurements or 60 measurements for every minute of the last hour etc.) for one or more variables in the calculation. Plus, the physics of the process, dictates how often or the frequency of the computations as new measurements arrive in real-time.

The calculation performed on the "raw data" can be categorized into:

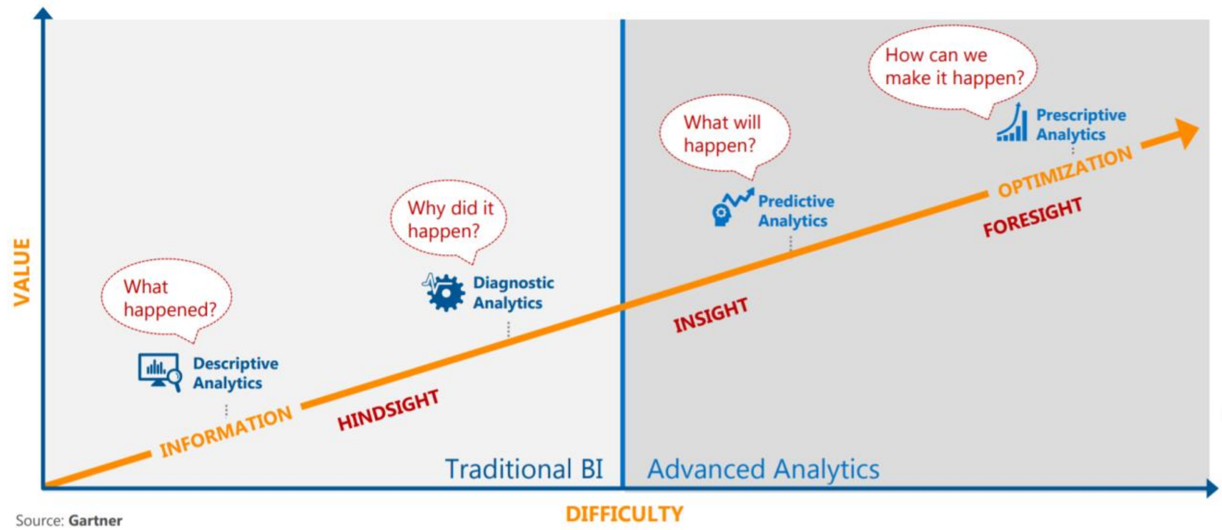
- ***Calculation that is once-through and done using a step-by-step procedure***
- ***Calculation involving iterative steps***
- ***Calculation involving complex math/statistics or data science/machine learning logic***

As such, the effort required - both human and computational - has a wide range. And, even though data science/machine learning tools are now widely available, they don't replace physics based calculations you can do with "raw data" to transform it to useful information.

Layers of analytics is also often categorized into:

- ***descriptive analytics*** - what happened
- ***diagnostic analytics*** - why did it happen
- ***predictive analytics*** - what can/will happen
- ***prescriptive analytics*** - what should I do, i.e. prescribing a course of action based on an understanding of historical data (what happened and why) and future events (what might happen)

The purpose of the analytics i.e. whether it is for *descriptive or diagnostic or predictive or prescriptive* will influence the "raw data" calculations and transforms. The following graph shows "value vs. difficulty" as you traverse the layers.



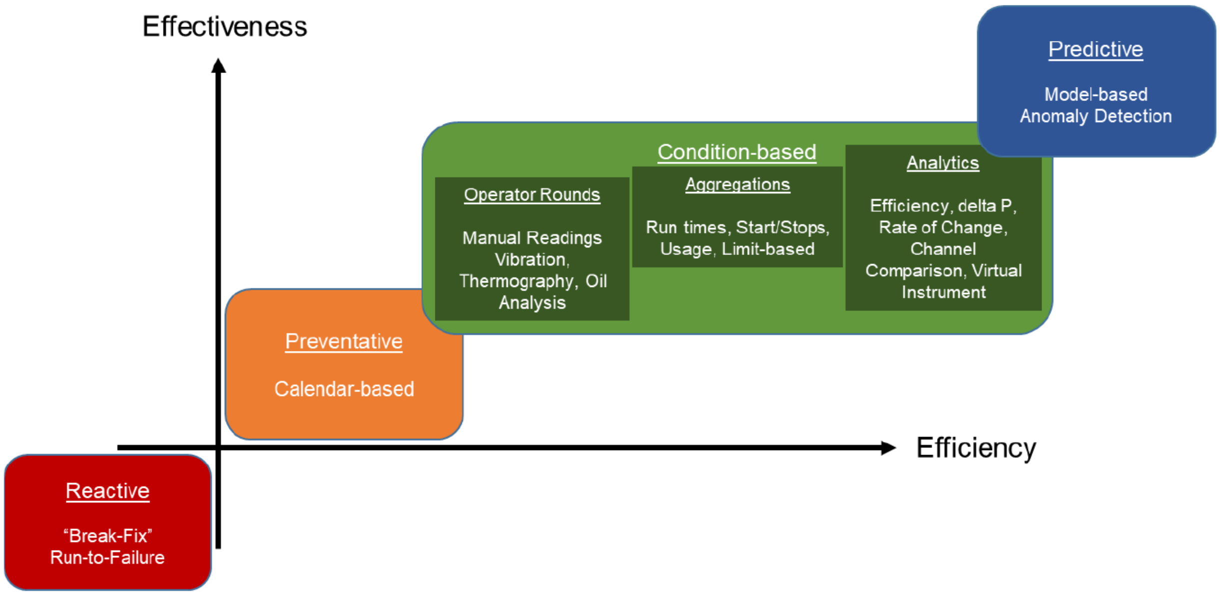
Layers of analytics can also be categorized by where the analytics is done, such as:

- ***Edge analytics***
- ***Server based analytics***
- ***Cloud-based analytics***

Analytics at the edge include those done immediately with the collected data, to lessen network load by reducing the amount of data forwarded to a server - for example, Fast-Fourier Transform (FFT) on vibration time wave-forms to extract frequency spectrums. Or, when an action is to be immediately taken based on the collected data without waiting for a round-trip to a remote analytics server - for example…

Layers of analytics can also be viewed through "scope of a business initiative" lens – for example, in asset maintenance and reliability, the layers are:

- ***UbM – Usage-based Maintenance - AF***
- ***CbM – Condition-based Maintenance - AF***
- ***PdM – Predictive Maintenance - AF plus third party libraries***



In the hands-on portion of this Lab:

- Exercise 1 uses an oil refinery process unit operation - Alkylation Feed Dryer - to walk-through the layers i.e. descriptive, diagnostic, predictive and prescriptive

- Exercise 2 uses a maintenance/reliability scenario (pump/motor assembly) to illustrate the layers i.e. UbM, CbM, and PdM

Items not included in the detailed hands-on portion will be covered as discussion topics during the Lab.

PI System software

The VM (virtual machine) used for this lab has the following PI System software installed:

Software	Version
PI Data Archive	2018 pre-release
PI Asset Framework (PI AF) server	2018 pre-release
PI Asset Framework (PI AF) client (PI System Explorer)	2018 pre-release
PI Analysis & PI Notifications Services	2018 pre-release
PI Vision	2017 R2
PI Web API	2018 pre-release
PI UFL Interface	3.4.22.28

For details on PI System software, please see http://www.osisoft.com/pi-system/pi-capabilities/product-list/

Exercise 1 – Alkylation Process Feed Dryer – Analytics

In this lab, we use of PI Event Frames to label operating conditions captured during the regeneration process of two process feed dryers. The dryers are cycled back and forth to continuously remove moisture from the feed before it combines with a strong acid which serves as a catalyst for the chemical reaction.

Even small amounts of moisture entering the reaction create a highly corrosive environment that will damage and compromise the equipment metallurgy. The modelling objective is to create a temperature profile representing proper regeneration of the dryer bed which is critical to this process.

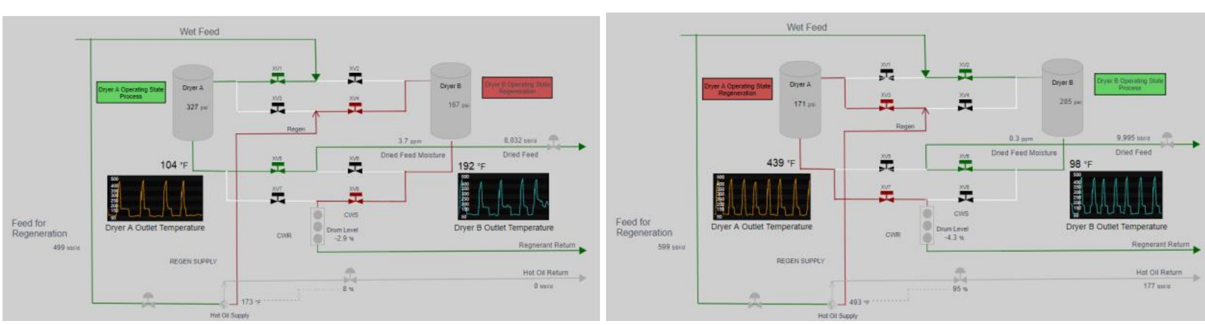
This profile will be developed in R/MATLAB and operationalized through AF Analytics and PI Vision. The data used for this Exercise comes from an actual processing facility and spans 2017 at six-minute intervals.

1a: Feed Drying Process

Use the desktop "Dryer Process Flow" shortcut to open a PI Vision display showing the feed dryer process flow

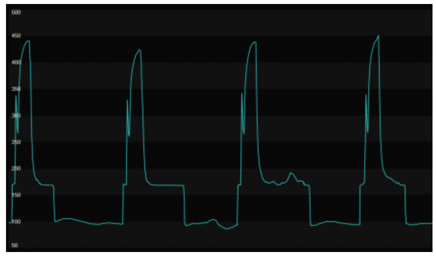
The dryers contain a bed of desiccant and molecular sieve material which removes moisture from the hydrocarbon stream passing through it at ambient conditions. The absorbed water is then removed during the regeneration process by passing the hydrocarbon stream through the bed at elevated temperature, 450-500 F.

In the display, green represents a "Process" state where the bed is drying (or available for drying) the process feed. Red represents a "Regeneration" state where the bed is being regenerated. You can see the alternating dryer modes by advancing the time range of the display.



Trends of the Outlet Temperatures for each bed are also shown; we focus on these temperature profiles in this Exercise.

The trend for Dryer B, below, shows the rise in temperature that occurs when the bed is being regenerated.   Therefore, the It is a good indicator for determining the current operating state of the dryer and is a critical variable when monitoring the regeneration cycle.

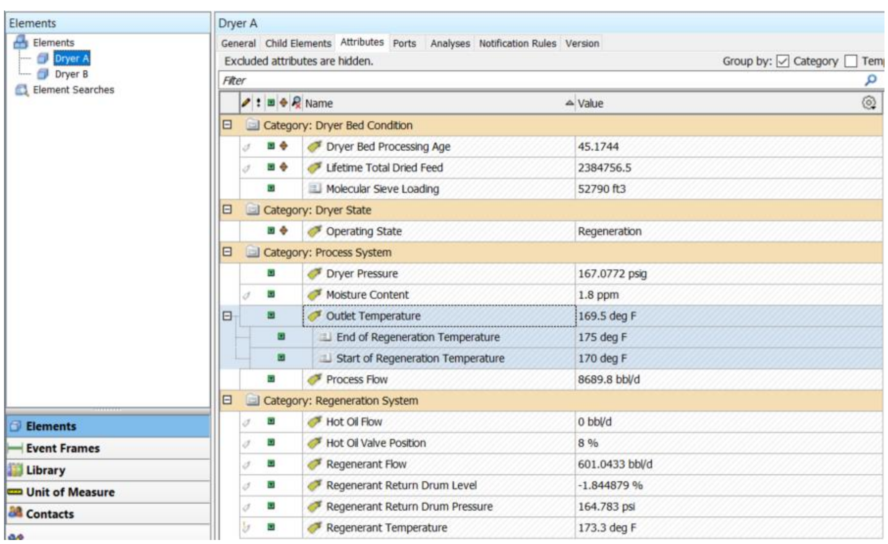


Dryer B Outlet Temperature, F

1b: Data Preparation with AF Expression Analytics

Open PI System Explorer from the taskbar and choose the "Dryers" database. Click on "Dryer A" to select this asset. You will use the "Attributes" tab to access the asset hierarchy and the "Analyses" tab to access the analytics.

The attributes associated with the "Dryer" template is shown below; note the "Process System" and 'Regeneration System" categories. The categories "Dryer Bed Conditions" and "Dryer State" contain attributes which are derived from descriptive analytic calculations we make using AF Analytics.

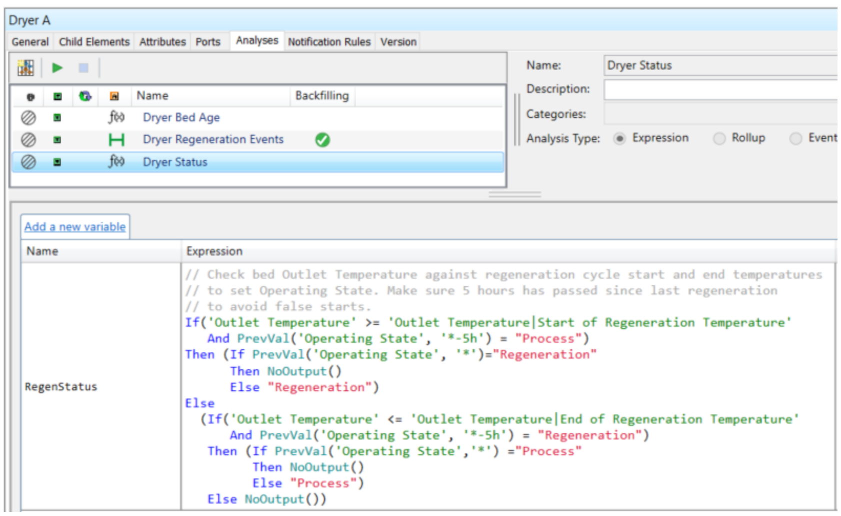


Sub-attributes under the "Outlet Temperature" attribute are temperature limits to define the start and end times of the regeneration cycle. These have been chosen after inspecting the PI Vision trends. Also note the "Molecular Sieve Loading" attribute of the Dryer asset model. This attribute has been looked up from an AF Table containing asset specifications.

Calculating “Operating State”

In PI System Explorer, select the “Analyses” tab at the top of the window. Select the “Dryer Status” analytic.

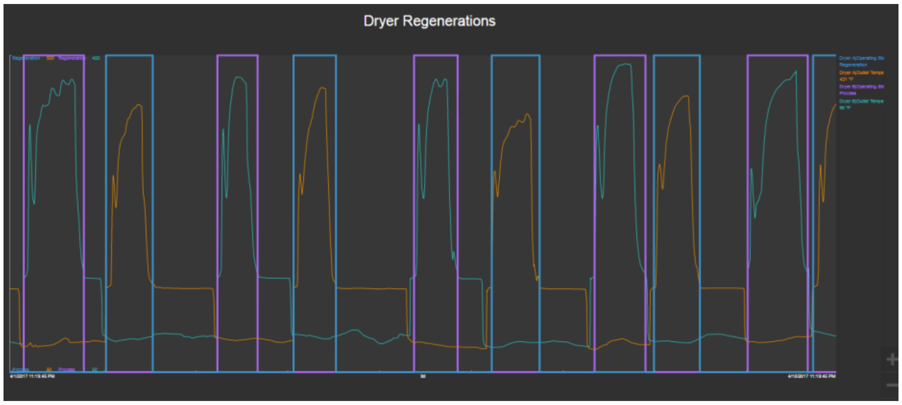
We will describe the operating state of each dryer using a digital state tag in PI. This will be a fundamental building block for the rest of our analysis. The analytic expression, “Dryer Status”, sets the digital state value of the “Operating State” attribute for each dryer. The digital state value is set to either “Process” or “Regeneration” based on the dryer “Outlet Temperature” and the past value of the “Operating State”.



The Operating State is set to “Regeneration” when the dryer Outlet Temperature rises above the “Start of Regeneration Temperature” (170 F) and set to “Process” when the dryer Outlet Temperature false below the “End of Regeneration Temperature” (175 F). We’ve added a second condition requiring at least a 5-hour elapsed time between this regeneration and the last, keeping small temperature fluctuations from falsely concluding Event Frames near the start of the regeneration.

Use the desktop “Dryer Status” shortcut to open the PI Vision display showing the feed dryer states.

Once this analytic was configured, the backfilling feature of AF Analytics was used to process the events in the PI Archive to populate the digital state tags for each dryer.

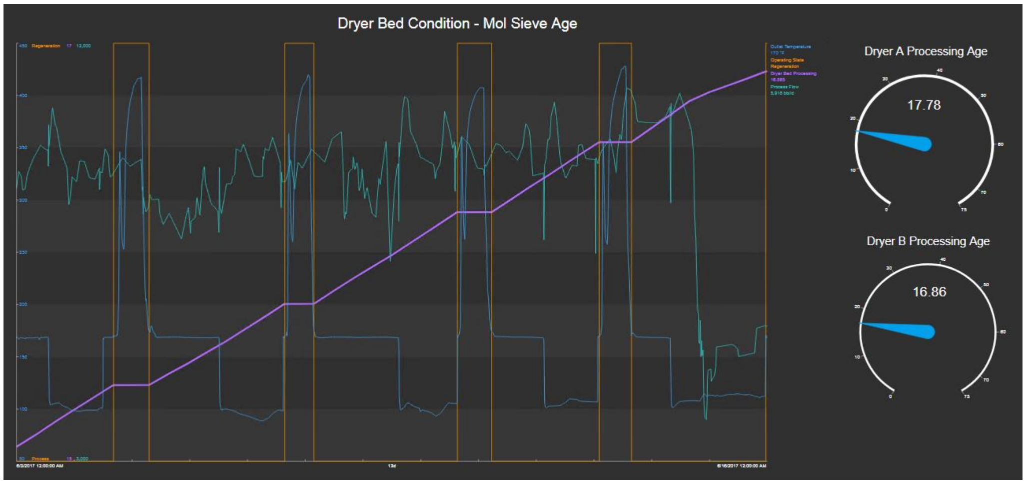


This display shows the outlet temperature and operating state for both dryers. Notice that the regenerations do not appear to follow a set schedule and that the cycle durations can vary – from about 6 hours to sometimes more than 12 hours.

Calculating "Dryer Bed Processing Age"

Open the PI Vision display showing the feed dryer process flow using the "Dryer Bed Condition" shortcut on the desktop, or from the Favorites list in PI Vision.

Repeated cycles between temperature extremes affect the bed's drying properties. To account for this, we will add a descriptive value to assess the processing age of each bed.



We have defined the processing age of a dryer bed to be:

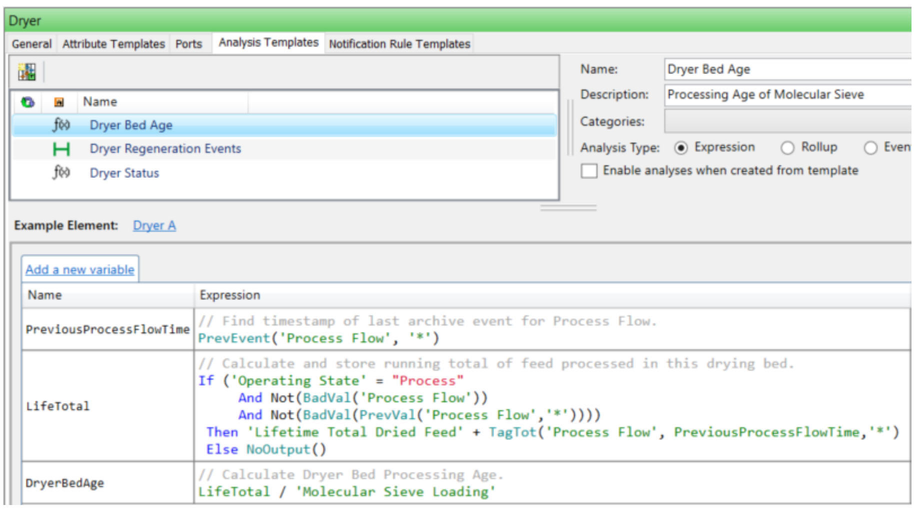
$$\frac{\text{Lifetime volume of feed dried by a bed (bbl.)}}{\text{Molecular sieve load in dryer (lb.)}}$$

The purple trace shows the Dryer Processing Age for Dryer A. The age is ever increasing except when the dryer is being regenerated. The teal trace in the trend shows the feed flowrate. We need to keep a running total of this value for each dryer to provide the Lifetime volume used in the Processing Age calculation. The gauge symbol shows that Dryer A has dried more barrels of feed than Dryer B.

Return to PI System Explorer, select "Dryer A" and use the "Analyses" tab to access the analytics. Select the "Dryer Bed Age" analytic.

The analytic expression, "Dryer Bed Age", calculates the processing age for each dryer bed using the following steps (names correspond to those in the AF Template configuration shown below).

- To obtain the interval start time for calculating the total feed volume, retrieve the timestamp for the last archived value of the Process Flow, (***PreviousProcesFlowTime***).
- If the dryer is in the "Process" state and the previous and current values of the Process Flow are good, add the incremental total to the lifetime total, (***LifeTotal***).
- Calculate the Dryer Processing Age based on the definition given above, (***DryerBedAge***).

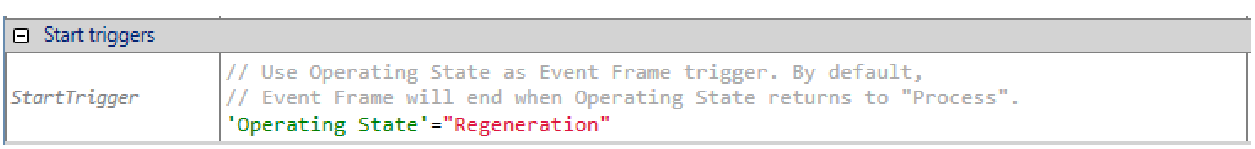


<u>Note</u>: The TotTag() function handles the unit conversion of Process Flow (bbl/d), a volumetric flow rate, and returns the correct volume, in bbls for any time increment. This is an inherent feature of AF Analytics which can be a difficult challenge when analyzing unevenly spaced data from instrumentation.

1c: Data Preparation with AF Event Frames

Within PI System Explorer, under the "Analyses" tab, select the "Dryer Regeneration Events" analytic.

Event Frames define and record interesting periods of process operation for analysis. In addition, Event Frame templates can make other process features available for analysis and insight. We can leverage the Operating State attribute for each dryer to define the regeneration Event Frames. A new Event Frame is started when a dryer's Operating State switches to "Regeneration". By default, the Event Frame will end when the Operating State returns to "Process".



We have used a NEW feature of the Event Frame Generation analytic (AF 2017 R2) to capture some other descriptive features of the regeneration process, calculated at the event end time. We have defined two variables; ***EFStartTime***- the current regeneration cycle start time, and ***PreviousEFEndTime*** – the previous regeneration cycle end time.

Variables	
EFStartTime	`// Find start time of this Event Frame by picking up time of` `// state change at beginning of regeneration.` `PrevEvent('Operating State', '*')`
PreviousEFEndTime	`// Find end time of previous regeneration cycle by picking up time of` `// state change prior to the beginnng of this regeneration.` `PrevEvent('Operating State',PrevEvent('Operating State', '*'))`

With these variables defined, we can add the following features related to each regeneration Event Frame.

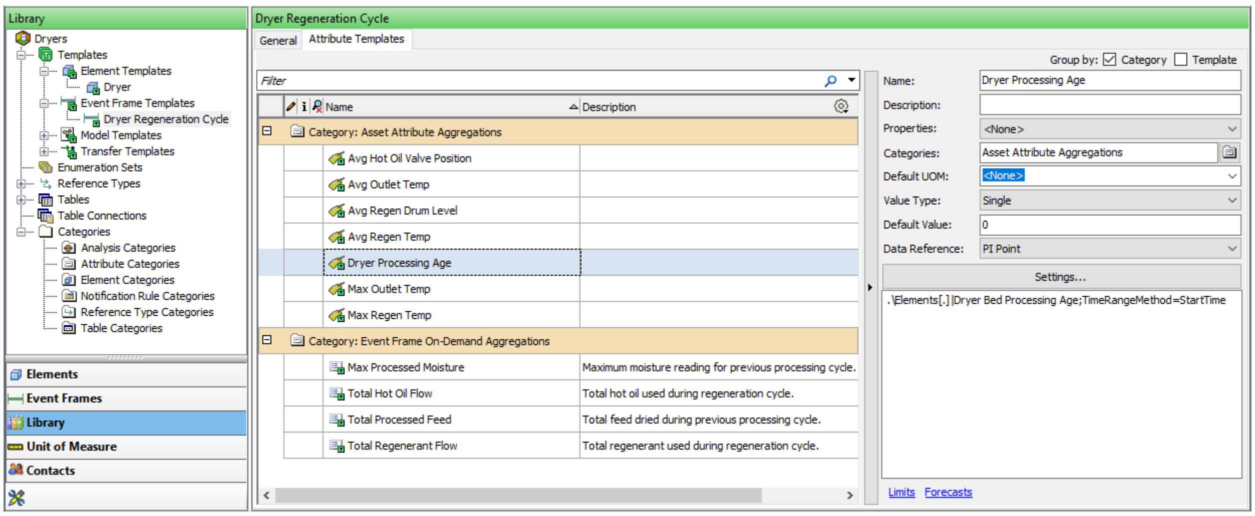
- ***TotalProcessedFeed*** – The barrels of feed passing through a dried in its previous Process cycle. Hypothetically, more barrels processed means a wetter bed at the start of the regeneration. This may affect the outlet temperature profile. (See picture below.)
- ***MaxProcessedMoisture*** – Maximum value of the moisture analyzer during the bed's previous Process cycle.
- ***TotalRegenerant*** – Total barrels of regenerant used during this Regeneration cycle.
- ***TotalHotOil*** – total barrels of hot oil used during tis Regeneration cycle.

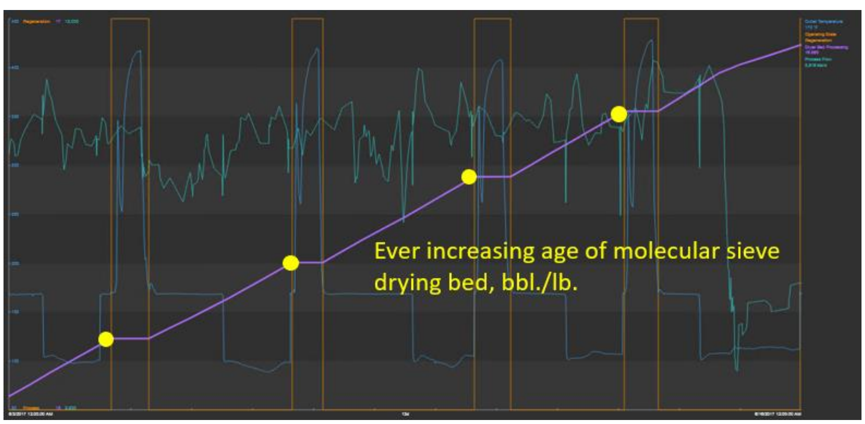
Outputs at close	
TotalProcessedFeed	`// Caculuate the total feed (barrels) dried during the previpous processing cycle.` `TagTot('Process Flow', PreviousEFEndTime, EFStartTime)`
MaxProcessedMoisture	`// Find maximum moisture reading for previous processing cycle.` `TagMax('Moisture Content', PreviousEFEndTime, EFStartTime)`
TotalRegenerant	`// Caculuate the regenerant (barrels) used during this regeneration cycle.` `TagTot('Regenerant Flow', EFStartTime, '*')`
TotalHotOil	`// Caculuate the hot oil (barrels) used during this regeneration cycle.` `TagTot('Hot Oil Flow', EFStartTime, '*')`

Total Processed Feed in Barrels

Within PI System Explorer, switch to the “Library” view (lower left-hand corner). Under Event Frame Templates, select the “Dryer Regeneration Cycle” template.

In addition to the calculations above, we can define features as attributes within the Event Frame template. These attributes provide basic aggregations of process values associated with the Dryer asset template. As shown in the “Dryer Regeneration Cycle” template below, we have included several quantities to give us insight for modelling the dryer regeneration outlet temperature profile. One in particular, “Dryer Processing Age”, will include this value, as of the start of the Regeneration cycle. Perhaps the age of the molecular sieve has an effect?



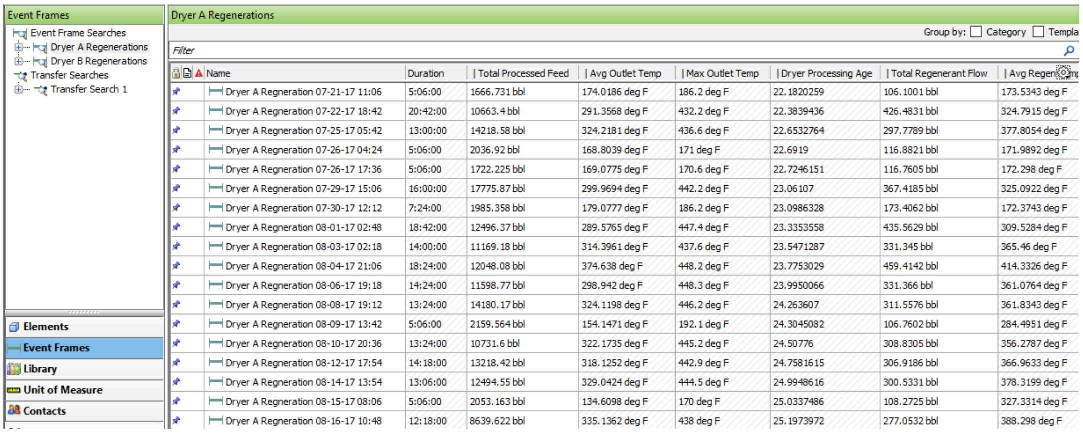


Dryer Processing Age at Start of Regeneration Cycle

Within PI System Explorer, switch to the “Event Frames” view (lower left-hand corner). Under Event Frame Searches, select the “Dryer A Regenerations” search to see the Event Frame records.

The Event Frame records shown below have been created by backfilling.

Next, we will publish this data along with interpolated values showing six-minute samples of the process data taken during the regeneration cycle. This dataset is then used in R/MATLAB.



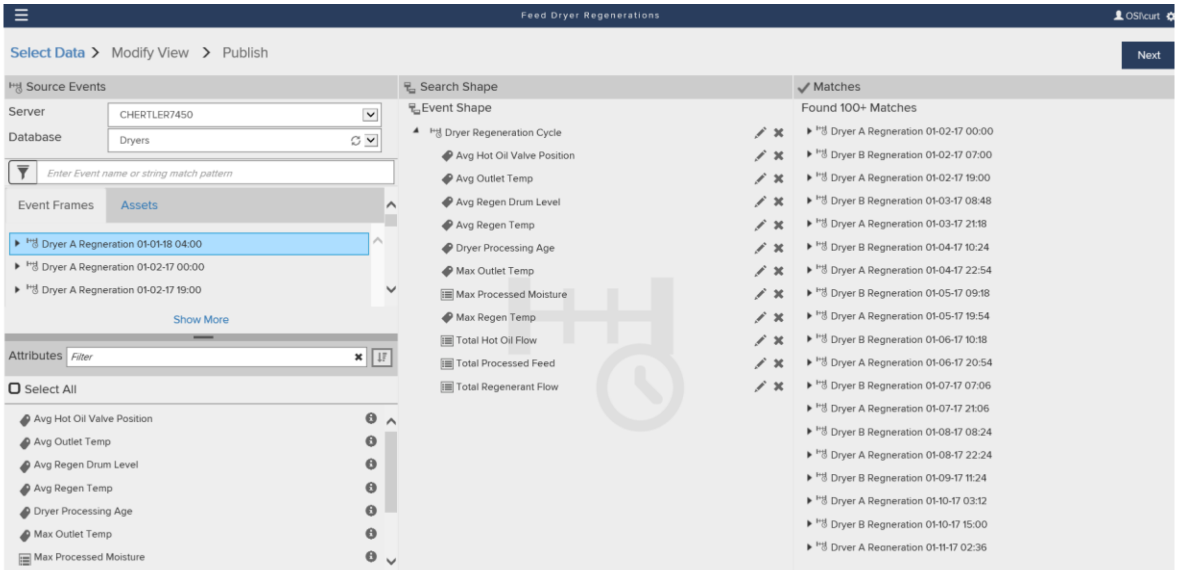
Name	Duration	Total Processed Feed	Avg Outlet Temp	Max Outlet Temp	Dryer Processing Age	Total Regenerant Flow	Avg Regen Temp
Dryer A Regneration 07-21-17 11:06	5:06:00	1666.731 bbl	174.0186 deg F	186.2 deg F	22.1820259	106.1001 bbl	173.5343 deg F
Dryer A Regneration 07-22-17 18:42	20:42:00	10663.4 bbl	291.3568 deg F	432.2 deg F	22.3839436	426.4831 bbl	324.7915 deg F
Dryer A Regneration 07-25-17 05:42	13:00:00	14218.58 bbl	324.2181 deg F	436.6 deg F	22.6532764	297.7789 bbl	377.8054 deg F
Dryer A Regneration 07-26-17 04:24	5:06:00	2036.92 bbl	168.8039 deg F	171 deg F	22.6919	116.8821 bbl	171.9892 deg F
Dryer A Regneration 07-26-17 17:36	5:06:00	1722.225 bbl	169.0775 deg F	170.6 deg F	22.7246151	116.7605 bbl	172.298 deg F
Dryer A Regneration 07-29-17 15:06	16:00:00	17775.87 bbl	299.9694 deg F	442.2 deg F	23.06107	367.4185 bbl	325.0922 deg F
Dryer A Regneration 07-30-17 12:12	7:24:00	1985.358 bbl	179.0777 deg F	186.2 deg F	23.0986328	173.4062 bbl	172.3743 deg F
Dryer A Regneration 08-01-17 02:48	18:42:00	12496.37 bbl	289.5765 deg F	447.4 deg F	23.3353558	435.5629 bbl	309.5284 deg F
Dryer A Regneration 08-03-17 02:18	14:00:00	11169.18 bbl	314.3961 deg F	437.6 deg F	23.5471287	331.345 bbl	365.46 deg F
Dryer A Regneration 08-04-17 21:06	18:24:00	12048.08 bbl	374.638 deg F	448.2 deg F	23.7753029	459.4142 bbl	414.3326 deg F
Dryer A Regneration 08-06-17 19:18	14:24:00	11598.77 bbl	298.942 deg F	448.3 deg F	23.9950066	331.366 bbl	361.0764 deg F
Dryer A Regneration 08-08-17 19:12	13:24:00	14180.17 bbl	324.1198 deg F	446.2 deg F	24.263607	311.5576 bbl	361.8343 deg F
Dryer A Regneration 08-09-17 13:42	5:06:00	2159.564 bbl	154.1471 deg F	192.1 deg F	24.3045082	106.7602 bbl	284.4951 deg F
Dryer A Regneration 08-10-17 20:36	13:24:00	10731.6 bbl	322.1735 deg F	445.2 deg F	24.50776	308.8305 bbl	356.2787 deg F
Dryer A Regneration 08-12-17 17:54	14:18:00	13218.42 bbl	318.1252 deg F	442.9 deg F	24.7581615	306.9186 bbl	366.9633 deg F
Dryer A Regneration 08-14-17 13:54	13:06:00	12494.55 bbl	329.0424 deg F	444.5 deg F	24.9948616	300.5331 bbl	378.3199 deg F
Dryer A Regneration 08-15-17 08:06	5:06:00	2053.163 bbl	134.6098 deg F	170 deg F	25.0337486	108.2725 bbl	327.3314 deg F
Dryer A Regneration 08-16-17 10:48	12:18:00	8639.622 bbl	335.1362 deg F	438 deg F	25.1973972	277.0532 bbl	388.298 deg F

1d: Shaping and Publishing Data with the PI Integrator for Business Analytics

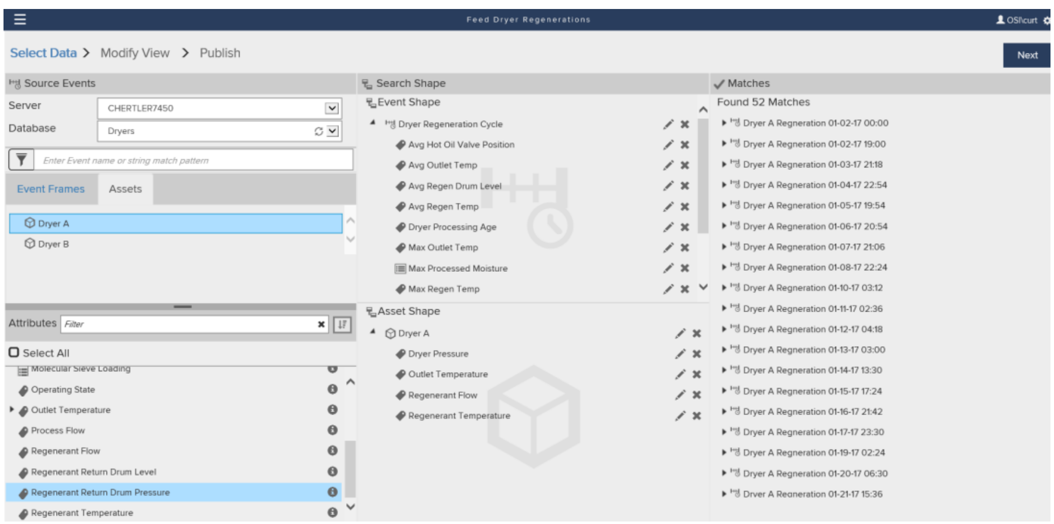
Open the PI Integrator for Business Analytics from the desktop shortcut.

To review the Event View publication configuration, open the desktop shortcut “PI Integrator for Business Analytics”. Select the “Feed Dryer Regenerations” publication and click on the ***Modify Publication*** button at the top pf the browser window. Modify the publication name to be something different and when prompted and click Ok to proceed to the *Select Data* page.

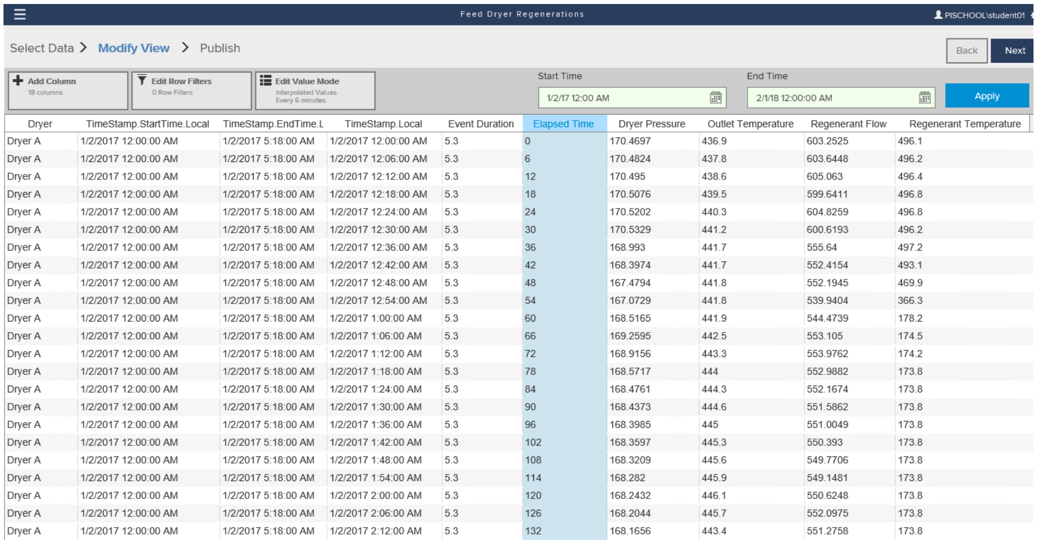
Event View shaping can involve two steps, which is the case in this example. First, we will select from the attributes defined in the Event Frame template and included in their records. These are attributes defined in the Event Frame template and are shown under the “Event Frames” tab in the left-hand pane. They usually report aggregations, (total, average, maximum, minimum) of important values, taken over the duration of the Event Frame. In this example, we will add all attributes to make them available for our analysis in R/MATLAB.



Since we need to know the details of what happened during each regeneration Event Frame, we need to add attributes associated with the Dryer asset template. These are shown under the "Assets" tab in the left-hand pane. The completed Event View shape is shown below. Click ***Next*** in the upper right-hand corner to move to the *Modify View* page.

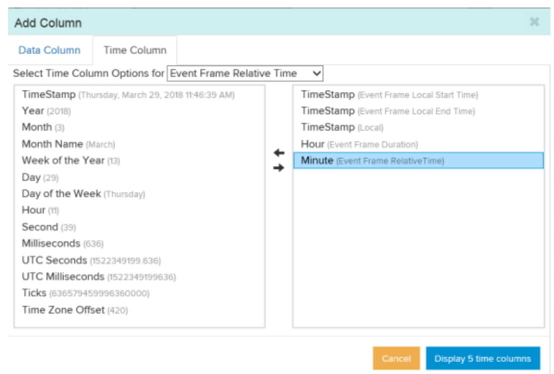


The *Modify View* page provides the publication preview shown below. On this page, we have configured the start and end times to correspond with the available data and have set the "Value Mode" to include sampled values at 6-minute increments to give us the data needed to create the profile model.



Dryer	TimeStamp.StartTime.Local	TimeStamp.EndTime.L	TimeStamp.Local	Event Duration	Elapsed Time	Dryer Pressure	Outlet Temperature	Regenerant Flow	Regenerant Temperature
Dryer A	1/2/2017 12:00:00 AM	1/2/2017 5:18:00 AM	1/2/2017 12:00:00 AM	5.3	0	170.4697	436.9	603.2525	496.1
Dryer A	1/2/2017 12:00:00 AM	1/2/2017 5:18:00 AM	1/2/2017 12:06:00 AM	5.3	6	170.4824	437.8	603.6448	496.2
Dryer A	1/2/2017 12:00:00 AM	1/2/2017 5:18:00 AM	1/2/2017 12:12:00 AM	5.3	12	170.495	438.6	605.063	496.4
Dryer A	1/2/2017 12:00:00 AM	1/2/2017 5:18:00 AM	1/2/2017 12:18:00 AM	5.3	18	170.5076	439.5	599.6411	496.8
Dryer A	1/2/2017 12:00:00 AM	1/2/2017 5:18:00 AM	1/2/2017 12:24:00 AM	5.3	24	170.5202	440.3	604.8259	496.8
Dryer A	1/2/2017 12:00:00 AM	1/2/2017 5:18:00 AM	1/2/2017 12:30:00 AM	5.3	30	170.5329	441.2	600.6193	496.2
Dryer A	1/2/2017 12:00:00 AM	1/2/2017 5:18:00 AM	1/2/2017 12:36:00 AM	5.3	36	168.993	441.7	555.64	497.2
Dryer A	1/2/2017 12:00:00 AM	1/2/2017 5:18:00 AM	1/2/2017 12:42:00 AM	5.3	42	168.3974	441.7	552.4154	493.1
Dryer A	1/2/2017 12:00:00 AM	1/2/2017 5:18:00 AM	1/2/2017 12:48:00 AM	5.3	48	167.4794	441.8	552.1945	469.9
Dryer A	1/2/2017 12:00:00 AM	1/2/2017 5:18:00 AM	1/2/2017 12:54:00 AM	5.3	54	167.0729	441.8	539.9404	366.3
Dryer A	1/2/2017 12:00:00 AM	1/2/2017 5:18:00 AM	1/2/2017 1:00:00 AM	5.3	60	168.5165	441.9	544.4739	178.2
Dryer A	1/2/2017 12:00:00 AM	1/2/2017 5:18:00 AM	1/2/2017 1:06:00 AM	5.3	66	169.2595	442.5	553.105	174.5
Dryer A	1/2/2017 12:00:00 AM	1/2/2017 5:18:00 AM	1/2/2017 1:12:00 AM	5.3	72	168.9156	443.3	553.9762	174.2
Dryer A	1/2/2017 12:00:00 AM	1/2/2017 5:18:00 AM	1/2/2017 1:18:00 AM	5.3	78	168.5717	444	552.9882	173.8
Dryer A	1/2/2017 12:00:00 AM	1/2/2017 5:18:00 AM	1/2/2017 1:24:00 AM	5.3	84	168.4761	444.3	552.1674	173.8
Dryer A	1/2/2017 12:00:00 AM	1/2/2017 5:18:00 AM	1/2/2017 1:30:00 AM	5.3	90	168.4373	444.6	551.5862	173.8
Dryer A	1/2/2017 12:00:00 AM	1/2/2017 5:18:00 AM	1/2/2017 1:36:00 AM	5.3	96	168.3985	445	551.0049	173.8
Dryer A	1/2/2017 12:00:00 AM	1/2/2017 5:18:00 AM	1/2/2017 1:42:00 AM	5.3	102	168.3597	445.3	550.393	173.8
Dryer A	1/2/2017 12:00:00 AM	1/2/2017 5:18:00 AM	1/2/2017 1:48:00 AM	5.3	108	168.3209	445.6	549.7706	173.8
Dryer A	1/2/2017 12:00:00 AM	1/2/2017 5:18:00 AM	1/2/2017 1:54:00 AM	5.3	114	168.282	445.9	549.1481	173.8
Dryer A	1/2/2017 12:00:00 AM	1/2/2017 5:18:00 AM	1/2/2017 2:00:00 AM	5.3	120	168.2432	446.1	550.6248	173.8
Dryer A	1/2/2017 12:00:00 AM	1/2/2017 5:18:00 AM	1/2/2017 2:06:00 AM	5.3	126	168.2044	445.7	552.0975	173.8
Dryer A	1/2/2017 12:00:00 AM	1/2/2017 5:18:00 AM	1/2/2017 2:12:00 AM	5.3	132	168.1656	443.4	551.2758	173.8

For sampled Event Views it is often important to know the elapsed time into the Event Frame for each record. The "Add Columns" dialog (shown right) can be used to add a column showing this value. From the "Time Column" tab change the "Time Column Options" to "Event Frame Relative Time". Adding Minutes to the columns list will add the elapsed time, in minutes to the publication. We renamed this column "Elapsed Time".



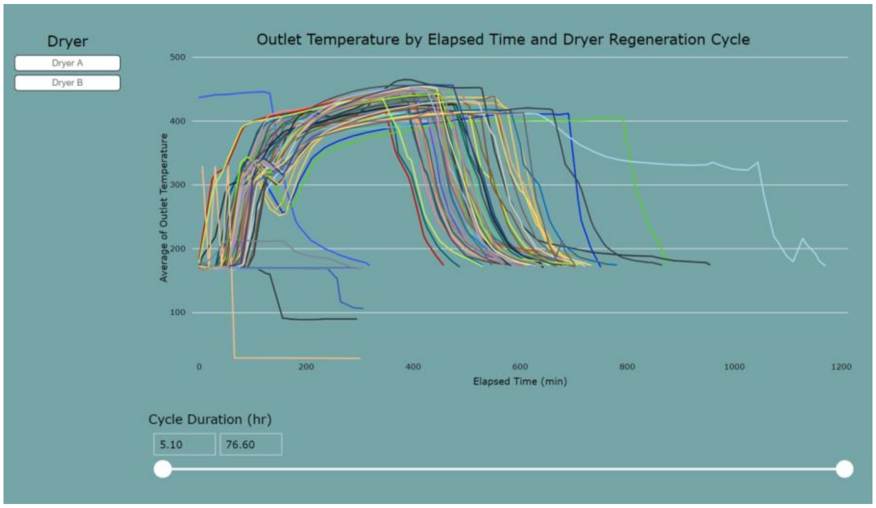
Clicking ***Next*** in the upper-right hand corner of the page will open the *Publish* page. From here, we could select "Feed Dryer Data", corresponding to a VM subdirectory, as the publication target. Clicking ***Publish*** and ***Acknowledge*** would start the publication, but we will not be doing this as part of this lab.

1e: Diagnostic Analysis with Power BI

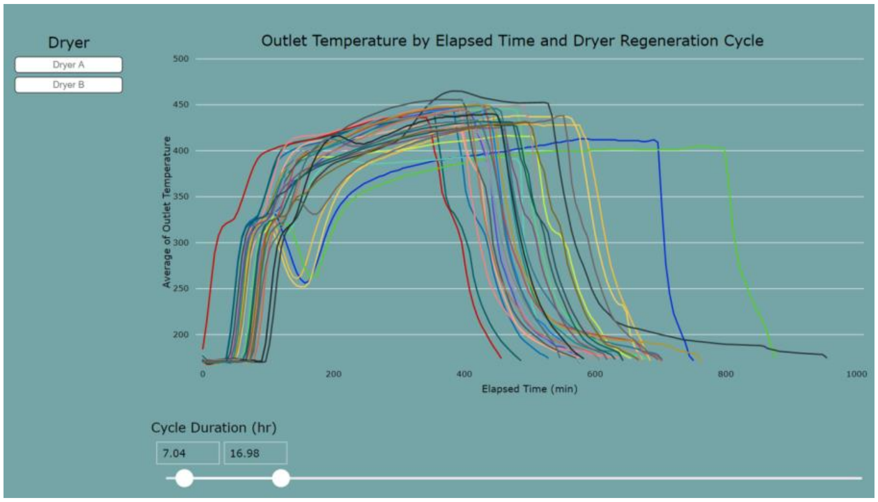
Open the Power BI workbook titled "Dryer Regenerations Analysis" from the desktop shortcut.

Once the sampled Event View has been published, the resulting dataset can be used for diagnostic analytics. Before performing predictive analysis in MATLAB, let's look at the dataset in Power BI. Open the Power BI workbook, "Dryer Regeneration Analysis" from the desktop shortcut. We have imported the dryer dataset from the published text file. You can view this data by clicking the "Data" icon on the left-hand toolbar of the Power BI window.

The first page of the Power BI workbook shows outlet temperature profiles for all Event Frames. A "Dryer" slicer has been added to filter the plot to show Event Frames generated from each dryer independently. The "Cycle Duration" slider allows filtering of Event Frame by duration. (Note: The "Elapsed Time" axis is in minutes and the slider is in hours.)

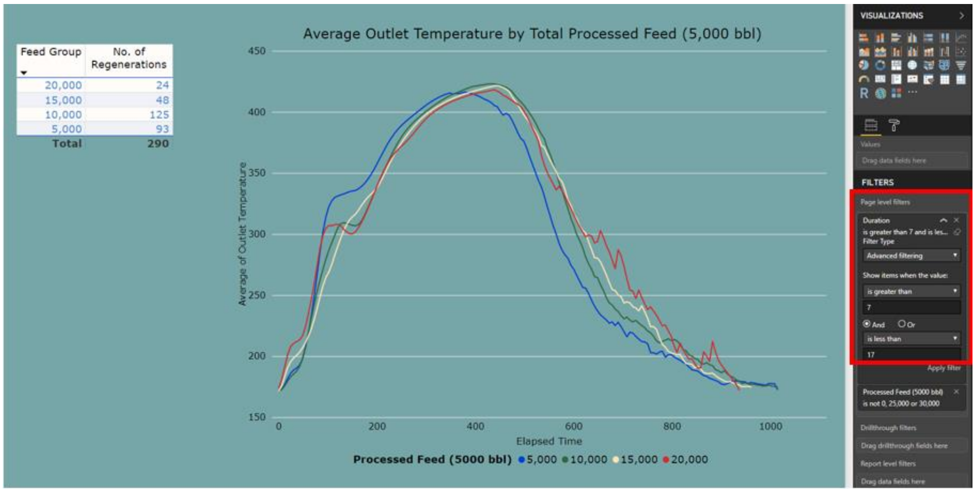


From the chart above, we can see that there are Event Frames that seem to be different than the rest. Perhaps our Event Framing rule needs some refinement. We can use Power BI to adjust the cycle duration to eliminate these Event Frames from our analysis.



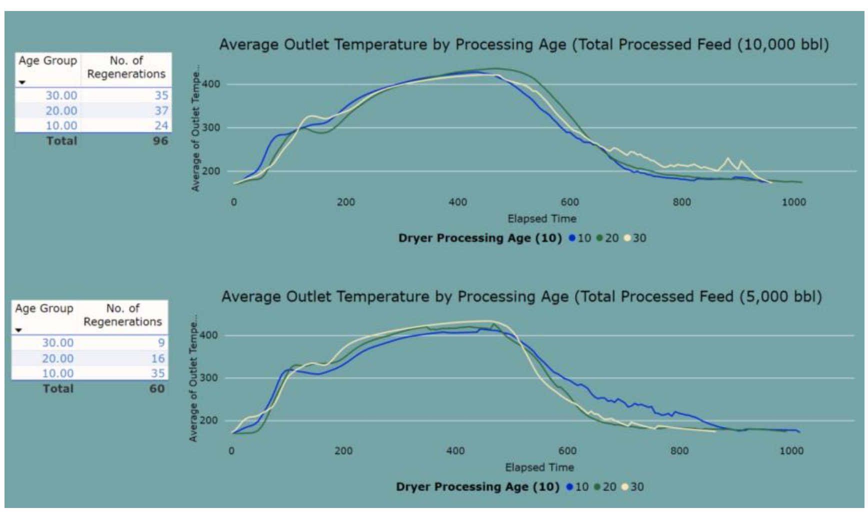
After some filtering, we will continue our Power BI analysis using Event Frames with cycle durations between 7 and 17 hours.

Move to the second page of the workbook by selecting “Processed Feed Analysis” tab below the workspace. Based on our analysis above, we have filtered this page to only show Event Frames with cycle durations between 7 and 17 hours (red box, below).



Here we are taking a looking at the averaged outlet temperature profiles of groups of Event Frames. These groups are based on the Total Processed Feed feature recorded in each Event Frame record. It is easy to create groups within multidimensional analysis tools. In this case, the groups have been created in 5,000 bbl. Increments, e.g. “Feed Group” 10,000 contains regeneration Event Frames where 5,001 to 10,000 bbls. of feed was dried in the prior Process cycle. We have filtered our Feed Groups 0, 25,000 and 30,000 because there were not enough Event Frames in these groups to consider in our analysis. What conclusions can you make from this plot?

Move to the third page of the workbook by selecting “Bed Processing Age Analysis” tab below the workspace. Again, we have filtered this page to only show Event Frames with cycle durations between 7 and 17 hours.

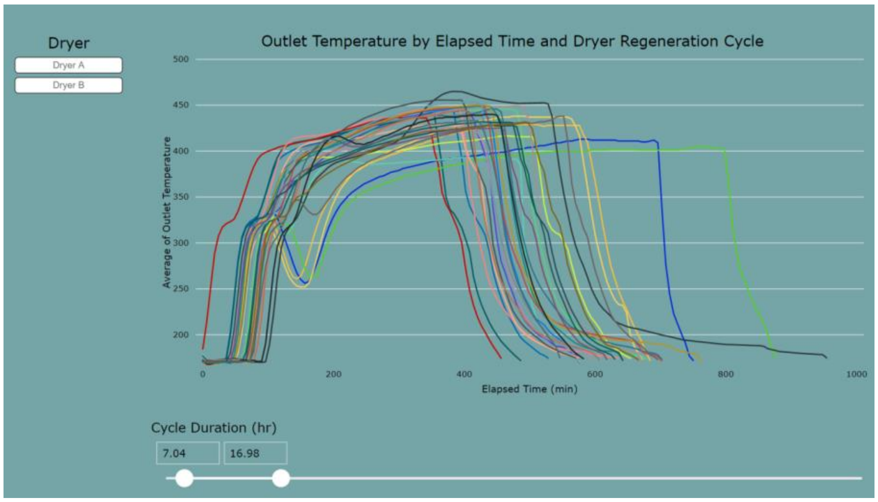


Filtering the data differently, this final page shows the effect of Dryer Processing Age on the outlet temperature profiles. We’ve grouped the Processing Age into 10 bbl/lb ranges and filtered out groups having a small number of Event Frames. What conclusions can we make here?

R and MATLAB

In this portion of the Lab, we will apply R/MATLAB to utilize historical feed dryer data to derive an expected profile for the Outlet Temperature during dryer regeneration.

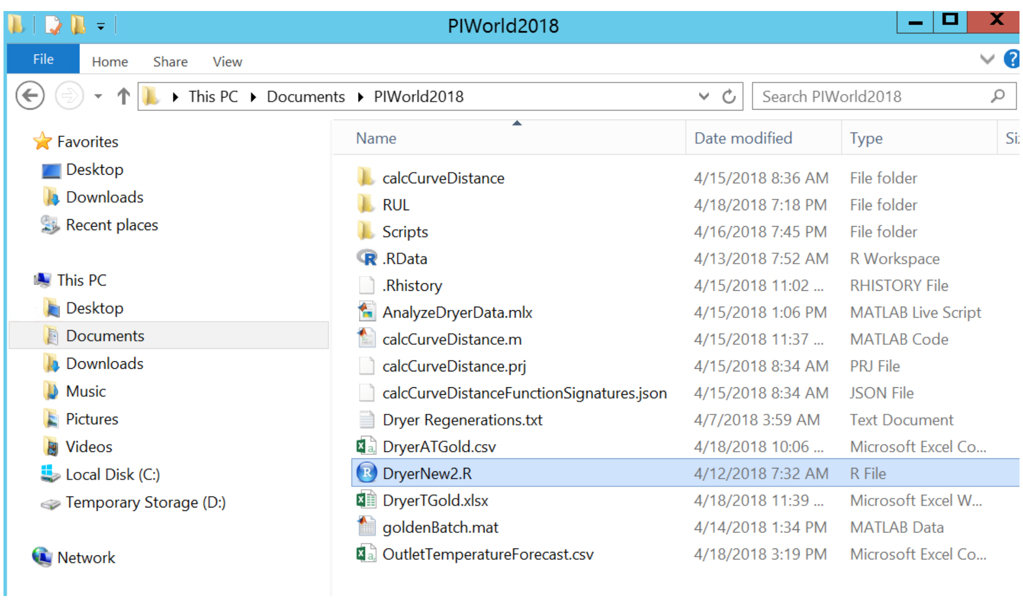
In an earlier section, we saw the following:



Using a series of data transformation steps and by applying process engineering principles, we will extract a "golden profile" for the Outlet Temperature which signifies "good operation."

1f: Extracting a Golden Run

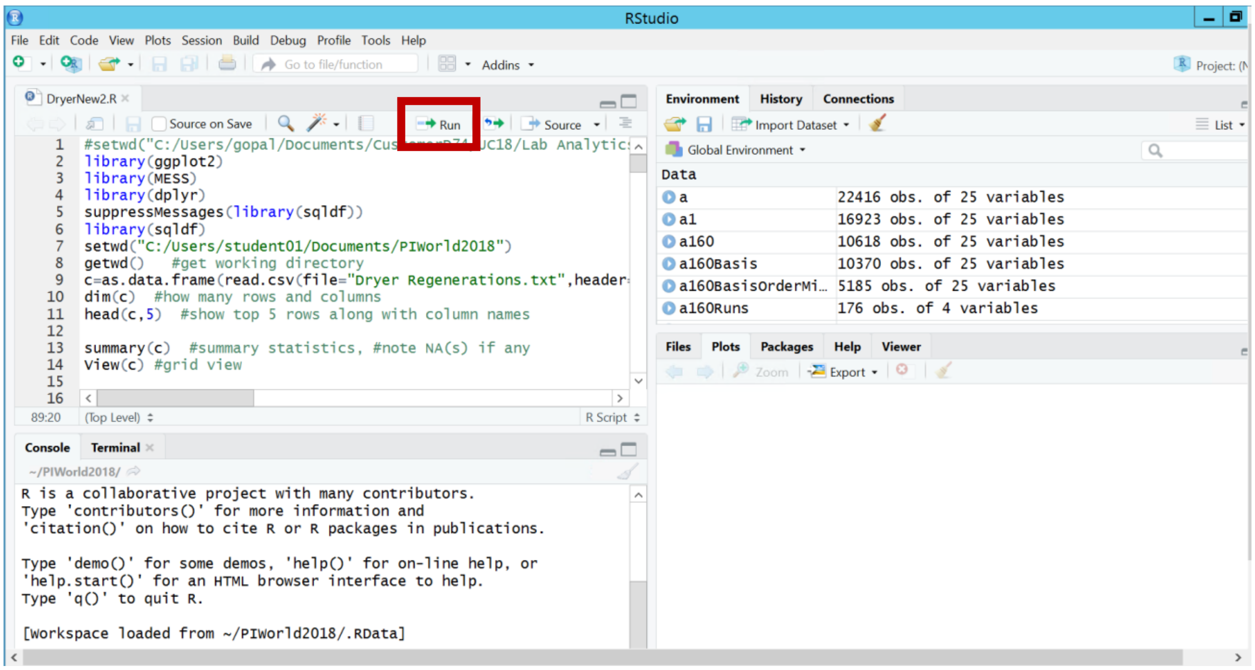
From the PIWorld2018 folder, select R file shown below and double-click to open it in R Studio (please be patient, R Studio takes several seconds to open).



The focus of the R code walk-through is to illustrate the data analysis methodology and not on the syntax of the language.

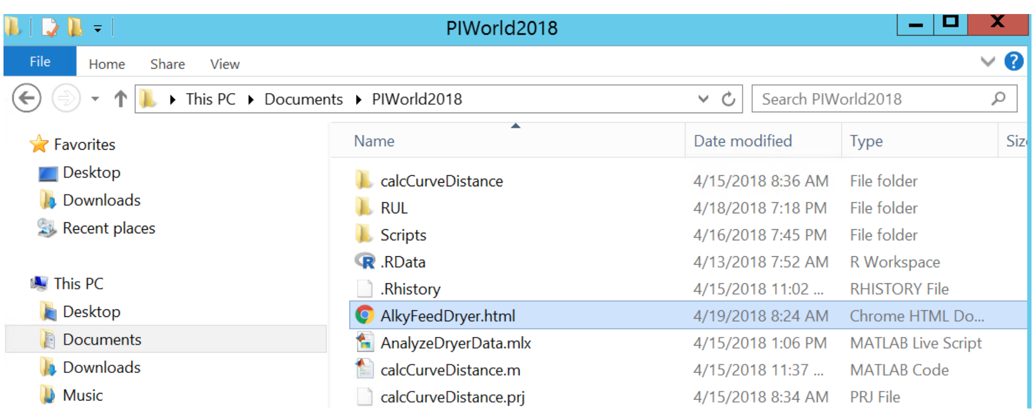
Equivalent MATLAB code is in *goldenBatch.mat*. Time permitting, we will also do a walk-through in MATLAB.

The screen below shows the R Studio user interface.



The following sections show the output when you step through the script line by line using **Run**.

The pages below have been extracted from the script output document ***.html** (see the lab folder for the latest revision).



Link to R document (HTML format)

Alky Feed Dryer

OSIsoft PI World 2018 - Power User Track - Fit for Purpose: Layers of Analytics using the PI System - AF, MATLAB, Machine Learning

set working directory, load libraries and read data

```
#setwd("C:/Users/student01/Documents/PIWorld2018/AlkyFeedDryer")   #set working directory
setwd("C:/Users/gopal/Documents/CustomerD74/UC18/Lab Analytics")
getwd()   #get working directory
```

```
## [1] "C:/Users/gopal/Documents/CustomerD74/UC18/Lab Analytics"
```

```
library(ggplot2)

suppressMessages(library(sqldf))
library(sqldf)

suppressMessages(library(MESS))
library(MESS)

suppressMessages(library(dplyr))
library(dplyr)
```

read the PI Integrator file output

```
c=as.data.frame(read.csv(file="Dryer Regenerations.txt",header=TRUE,sep="\t"))

dim(c)  #how many rows and columns
```

```
## [1] 49058    22
```

```
head(c,5)  #show top 5 rows along with column names
```

```
##   ï..Id   Dryer TimeStamp.StartTime.Local TimeStamp.EndTime.Local
## 1     1 Dryer A      1/2/2017 12:00:00 AM     1/2/2017 5:18:00 AM
## 2     2 Dryer A      1/2/2017 12:00:00 AM     1/2/2017 5:18:00 AM
## 3     3 Dryer A      1/2/2017 12:00:00 AM     1/2/2017 5:18:00 AM
## 4     4 Dryer A      1/2/2017 12:00:00 AM     1/2/2017 5:18:00 AM
## 5     5 Dryer A      1/2/2017 12:00:00 AM     1/2/2017 5:18:00 AM
##        TimeStamp.Local Duration ElapsedTime Dryer.Pressure
## 1 1/2/2017 12:00:00 AM      5.3           0       170.4697
## 2 1/2/2017 12:06:00 AM      5.3           6       170.4824
## 3 1/2/2017 12:12:00 AM      5.3          12       170.4950
## 4 1/2/2017 12:18:00 AM      5.3          18       170.5076
## 5 1/2/2017 12:24:00 AM      5.3          24       170.5202
##   Outlet.Temperature Regenerant.Flow Regenerant.Temperature
## 1              436.9        603.2525                  496.1
## 2              437.8        603.6448                  496.2
## 3              438.6        605.0630                  496.4
## 4              439.5        599.6411                  496.8
## 5              440.3        604.8259                  496.8
```

```
##            Dryer.Regeneration.Cycle Avg.Hot.Oil.Valve.Position
## 1 Dryer A Regneration 01-02-17 00:00                   15.05472
## 2 Dryer A Regneration 01-02-17 00:00                   15.05472
## 3 Dryer A Regneration 01-02-17 00:00                   15.05472
## 4 Dryer A Regneration 01-02-17 00:00                   15.05472
## 5 Dryer A Regneration 01-02-17 00:00                   15.05472
##   Avg.Outlet.Temp Avg.Regen.Drum.Level Avg.Regen.Temp Max.Outlet.Temp
## 1        319.9179            -2.085758       229.7132           446.1
## 2        319.9179            -2.085758       229.7132           446.1
## 3        319.9179            -2.085758       229.7132           446.1
## 4        319.9179            -2.085758       229.7132           446.1
## 5        319.9179            -2.085758       229.7132           446.1
##   Max.Processed.Moisture Max.Regen.Temp Total.Processed.Feed PIIntTSTicks
## 1                   -1.3          497.2             4474.716 6.361894e+17
## 2                   -1.3          497.2             4474.716 6.361894e+17
## 3                   -1.3          497.2             4474.716 6.361894e+17
## 4                   -1.3          497.2             4474.716 6.361894e+17
## 5                   -1.3          497.2             4474.716 6.361894e+17
##   PIIntShapeID
## 1            0
## 2            0
## 3            0
## 4            0
## 5            0
```

```
summary(c)  #summary statistics, #note NA(s) if any
```

```
##      ï..Id           Dryer               TimeStamp.StartTime.Local
##  Min.   :    1   Dryer A:22416   2/27/2017 1:48:00 AM :  767    
##  1st Qu.:12265   Dryer B:26642   3/2/2017 11:30:00 AM :  578    
##  Median :24530                   7/27/2017 10:24:00 AM:  438    
##  Mean   :24530                   10/6/2017 6:30:00 AM :  343    
##  3rd Qu.:36794                   6/15/2017 11:24:00 PM:  335    
##  Max.   :49058                   9/11/2017 2:12:00 PM :  289    
##                                  (Other)              :46308    
##         TimeStamp.EndTime.Local              TimeStamp.Local 
##  3/2/2017 6:24:00 AM :  767      11/2/2017 10:36:00 AM:    2  
##  3/4/2017 9:12:00 PM :  578      11/2/2017 10:42:00 AM:    2  
##  7/29/2017 6:06:00 AM:  438      11/2/2017 10:48:00 AM:    2  
##  10/7/2017 4:42:00 PM:  343      11/2/2017 10:54:00 AM:    2  
##  6/17/2017 8:48:00 AM:  335      11/2/2017 11:00:00 AM:    2  
##  9/12/2017 7:00:00 PM:  289      11/2/2017 11:06:00 AM:    2  
##  (Other)             :46308      (Other)              :49046  
##     Duration      ElapsedTime     Dryer.Pressure     Outlet.Temperature
##  Min.   : 5.10   Min.   :   0.0   Min.   :  0.5452   Min.   : 85.4     
##  1st Qu.:11.20   1st Qu.: 186.0   1st Qu.:167.9203   1st Qu.:198.6     
##  Median :12.70   Median : 384.0   Median :168.9629   Median :337.8     
##  Mean   :15.12   Mean   : 453.5   Mean   :170.8071   Mean   :312.2     
##  3rd Qu.:15.00   3rd Qu.: 594.0   3rd Qu.:170.3075   3rd Qu.:407.1     
##  Max.   :76.60   Max.   :4596.0   Max.   :348.8279   Max.   :465.0     
##                                                                        
##  Regenerant.Flow  Regenerant.Temperature
##  Min.   :   0.0   Min.   :107.0         
##  1st Qu.: 501.5   1st Qu.:173.5         
##  Median : 552.1   Median :457.9         
##  Mean   : 566.9   Mean   :364.6         
##  3rd Qu.: 601.2   3rd Qu.:480.9         
##  Max.   :1388.0   Max.   :537.7         
## 
```

```
##                        Dryer.Regeneration.Cycle Avg.Hot.Oil.Valve.Position
##  Dryer B Regneration 02-27-17 01:48:  767       Min.   : 7.199
##  Dryer B Regneration 03-02-17 11:30:  578       1st Qu.:39.793
##  Dryer B Regneration 07-27-17 10:24:  438       Median :61.894
##  Dryer B Regneration 10-06-17 06:30:  343       Mean   :56.654
##  Dryer B Regneration 06-15-17 23:24:  335       3rd Qu.:75.246
##  Dryer B Regneration 09-11-17 14:12:  289       Max.   :95.208
##  (Other)                           :46308
##  Avg.Outlet.Temp Avg.Regen.Drum.Level Avg.Regen.Temp  Max.Outlet.Temp
##  Min.   :104.0   Min.   :-5.000       Min.   :171.8   Min.   :170.0
##  1st Qu.:311.6   1st Qu.: 1.612       1st Qu.:354.9   1st Qu.:414.3
##  Median :324.8   Median : 5.184       Median :379.3   Median :429.0
##  Mean   :313.3   Mean   : 6.220       Mean   :365.3   Mean   :411.4
##  3rd Qu.:336.4   3rd Qu.: 9.576       3rd Qu.:392.3   3rd Qu.:442.1
##  Max.   :381.9   Max.   :32.497       Max.   :455.1   Max.   :465.0
##
##  Max.Processed.Moisture Max.Regen.Temp  Total.Processed.Feed
##  Min.   :-1.30          Min.   :171.9   Min.   :  672.9
##  1st Qu.: 5.80          1st Qu.:471.9   1st Qu.: 6407.4
##  Median :15.00          Median :485.5   Median :11062.3
##  Mean   :19.43          Mean   :473.8   Mean   :11211.7
##  3rd Qu.:25.80          3rd Qu.:496.2   3rd Qu.:14553.9
##  Max.   :63.00          Max.   :537.7   Max.   :32177.2
##
##   PIIntTSTicks         PIIntShapeID
##  Min.   :6.362e+17   Min.   :0
##  1st Qu.:6.363e+17   1st Qu.:0
##  Median :6.363e+17   Median :0
##  Mean   :6.363e+17   Mean   :0
##  3rd Qu.:6.364e+17   3rd Qu.:0
##  Max.   :6.365e+17   Max.   :0
##
```

```
#c=na.omit(c)  #remove rows containing NULL values, if any
```

grid view

```
View(c)
```

data manipulation

```
a=subset(c,Dryer=='Dryer A')
b=subset(c,Dryer=='Dryer B')
dim(a)
```

```
## [1] 22416    22
```

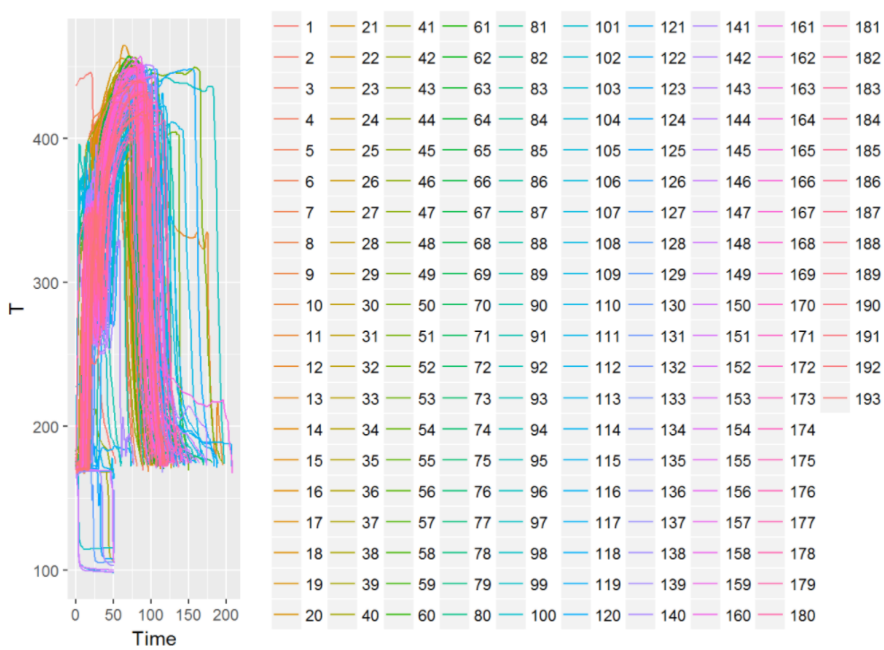
```
dim(b)
```

```
## [1] 26642    22
```

```r
View(a)  #Note the Dryer.Regeneration.Cycle column values - it is unique to each Regen Cycle and represents the Event Frame,
 the timestamps in the rows are in sequence and in increasing order for each Cycle

#it is easier to work with Cycle number instead of names, so convert to numeric IDs 1, 2, 3 etc.
a$Cycle=0
k=1
a$Cycle[1]=1
for (j in 2:nrow(a)) {
  if (a$Dryer.Regeneration.Cycle[j]==a$Dryer.Regeneration.Cycle[j-1]) {
    a$Cycle[j]=k
  }
  else  {
    k=k+1
    a$Cycle[j]=k
  }
}

max(a$Cycle)  #193 Cycles
```

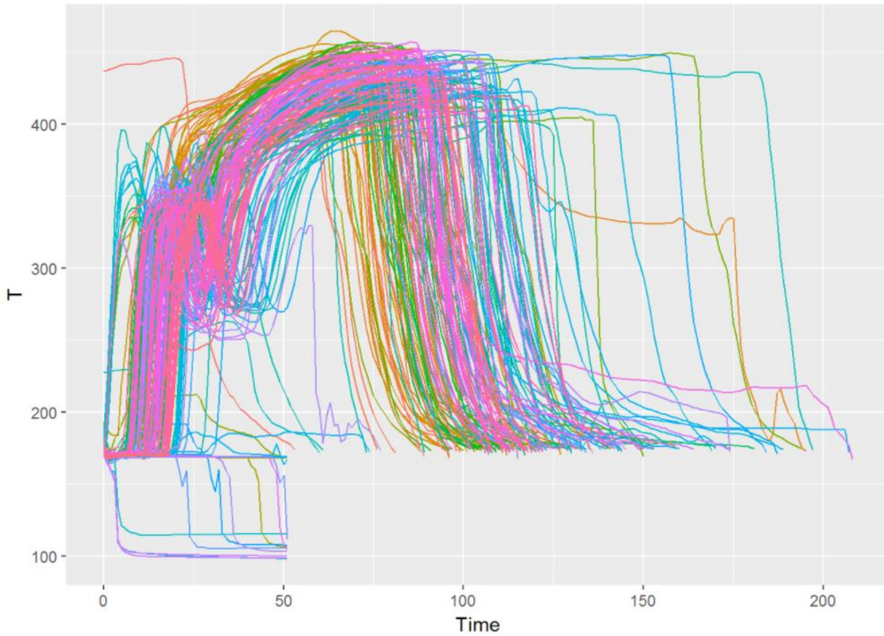
```
## [1] 193
```

```r
a$T=a$Outlet.Temperature  #shorten the name

a$Time=a$ElapsedTime/6  #easier work in 6 minute (0.1 hour) increments, Regen Cycle runs for several hours

#plot of Outlet Temperature for each Cycle (Event Frame) - overlaid
ggplot(data=a)+aes(x=Time,y=T)+geom_line(aes(color=as.factor(Cycle)))
```



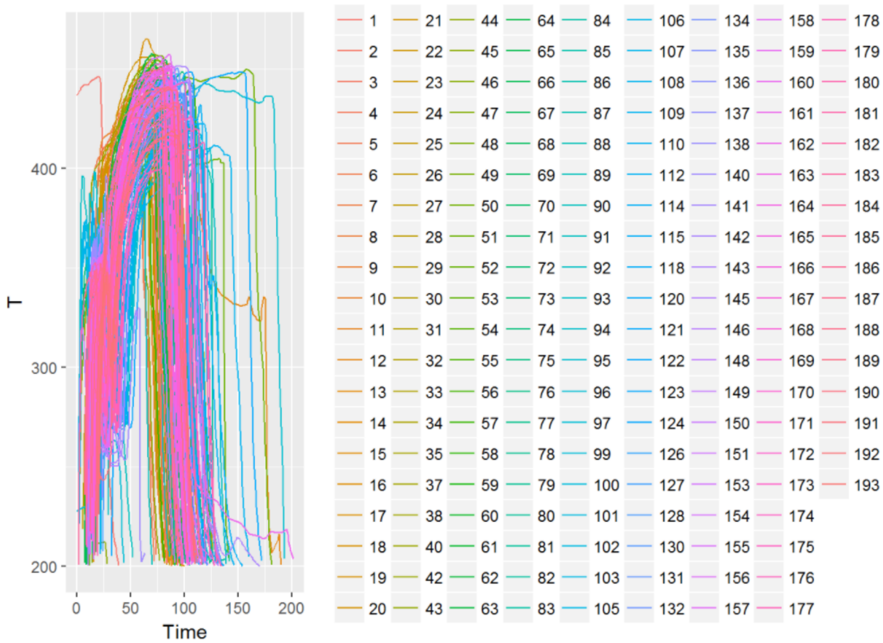

```r
ggplot(data=a)+aes(x=Time,y=T)+geom_line(aes(color=as.factor(Cycle))) + theme(legend.position='none')
```

```
ggplot(data=a)+aes(x=Time,y=T)+geom_line(aes(color=as.factor(Cycle))) + theme(legend.position='none')
```

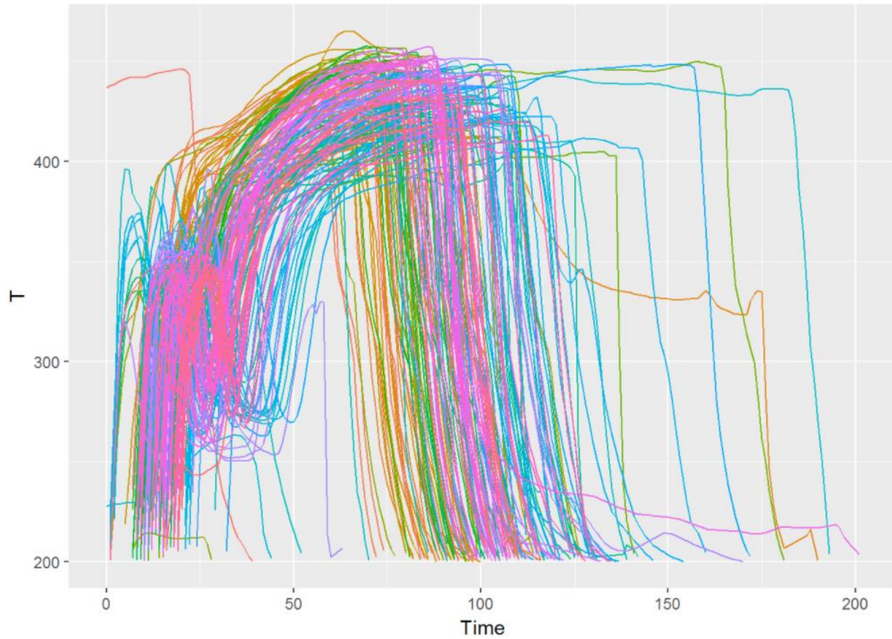


more plots and data prep

```
#filter to keep records with T>200; filtered dataset is renamed as a1
a1=subset(a, a$T>200)

ggplot(data=a1)+aes(x=Time,y=T)+geom_line(aes(color=as.factor(Cycle)))
```



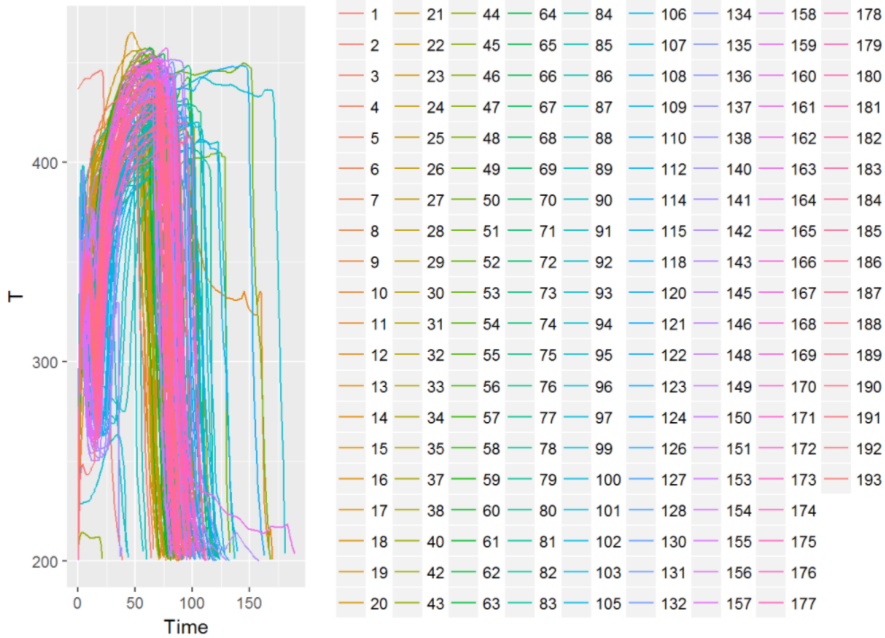
```r
ggplot(data=a1)+aes(x=Time,y=T)+geom_line(aes(color=as.factor(Cycle)))+ theme(legend.position='none')
```



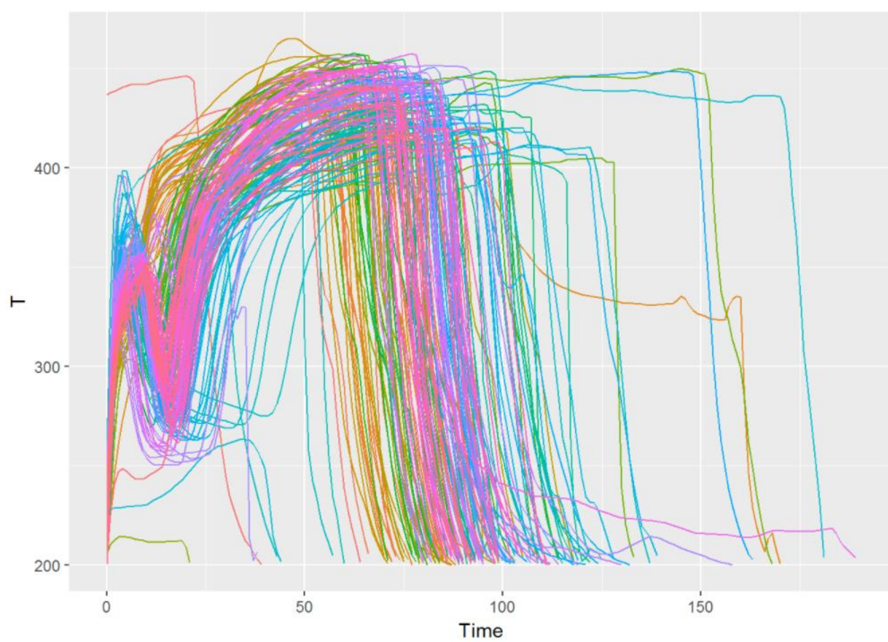

```r
#line up start time to 0
a1$Time=0
k=0
a1$Time[1]=0
for (j in 2:nrow(a1)) {
  if (a1$Cycle[j]==a1$Cycle[j-1]) {
    k=k+1
  }
  else  {
    k=0
  }
  a1$Time[j]=k
}

a1$StartTime=a1$TimeStamp.StartTime.Local  #shorten column name

#get aggregate values for each Cycle
a1Runs=sqldf("select max(Time) as TimeMax, max(T) as TMax, Time as TimeTMax, StartTime, Cycle from a1 group by Cycle", drv =
 'SQLite')
nrow(a1Runs)
```

```
## [1] 176
```

```r
ggplot(data=a1)+aes(x=Time,y=T)+geom_line(aes(color=as.factor(Cycle)))
```

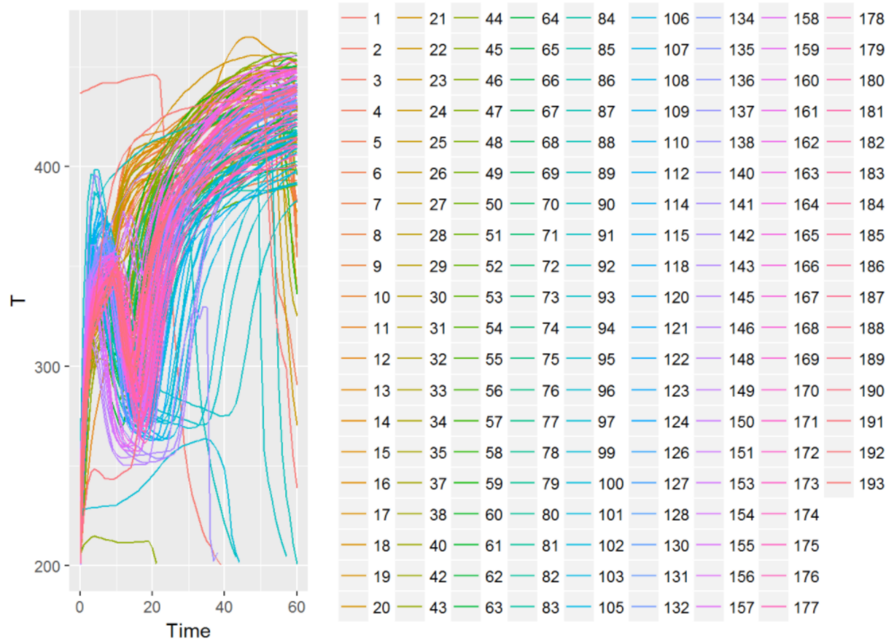



```
ggplot(data=a1)+aes(x=Time,y=T)+geom_line(aes(color=as.factor(Cycle)))+ theme(legend.position='none')
```

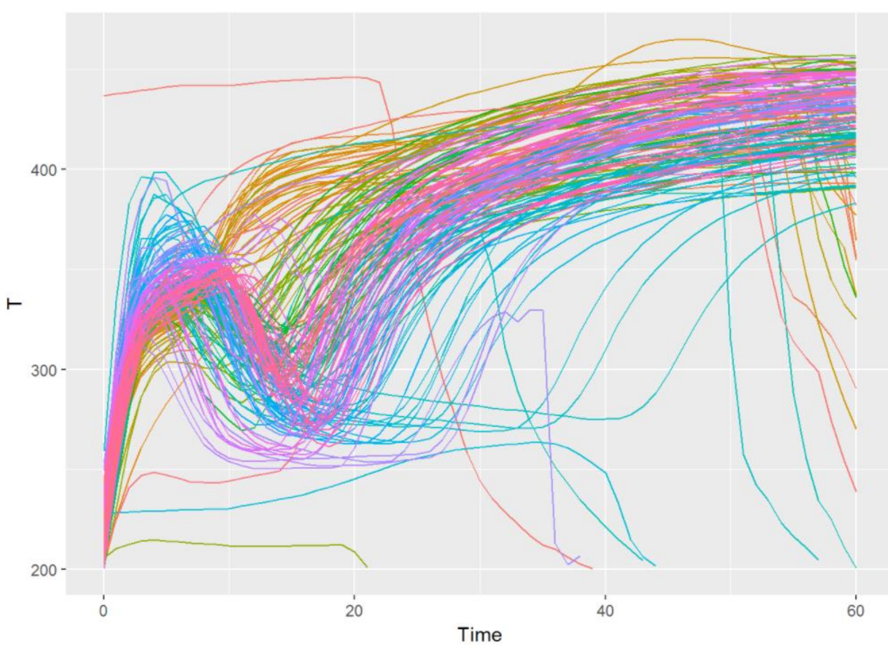


more plots and data prep2

```
#keep records for only the first 6 hours of each Cycle
a160=subset(a1,a1$Time<=60)
ggplot(data=a160)+aes(x=Time,y=T)+geom_line(aes(color=as.factor(Cycle)))
```




```
ggplot(data=a160)+aes(x=Time,y=T)+geom_line(aes(color=as.factor(Cycle)))+ theme(legend.position='none')
```



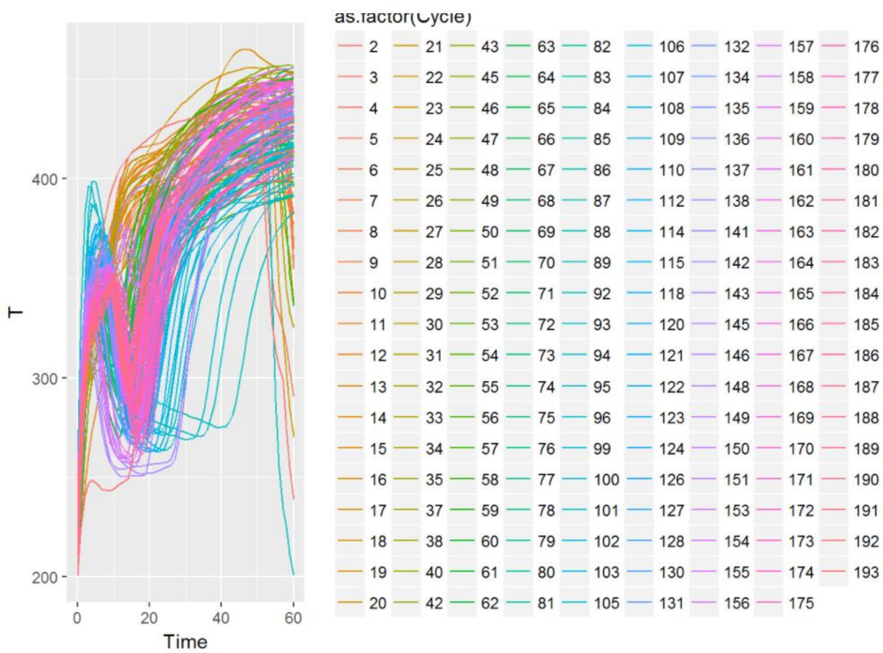
```
#get aggregate values for each Cycle
a160Runs=sqldf("select max(Time) as TimeMax, max(T) as TMax, Time as TimeTMax, StartTime, Cycle from a160 group by Cycle", d
rv = 'SQLite')
View(a160Runs)
nrow(a160Runs)
```

```
## [1] 176
```

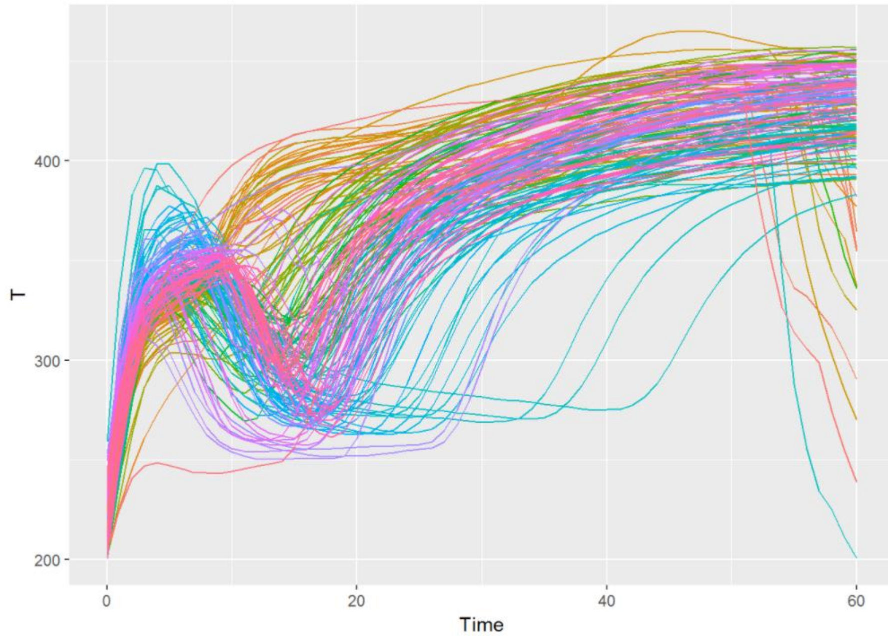
```
#select Cycles that ran for at least 6hours i.e. has full 6 hours of data
a160RunsBasis=subset(a160Runs,a160Runs$TimeMax==60)
View(a160RunsBasis)
nrow(a160RunsBasis)
```

```
## [1] 170
```

```
a160Basis=subset(a160,a160$Cycle %in% a160RunsBasis$Cycle)

ggplot(data=a160Basis)+aes(x=Time,y=T)+geom_line(aes(color=as.factor(Cycle)))
```

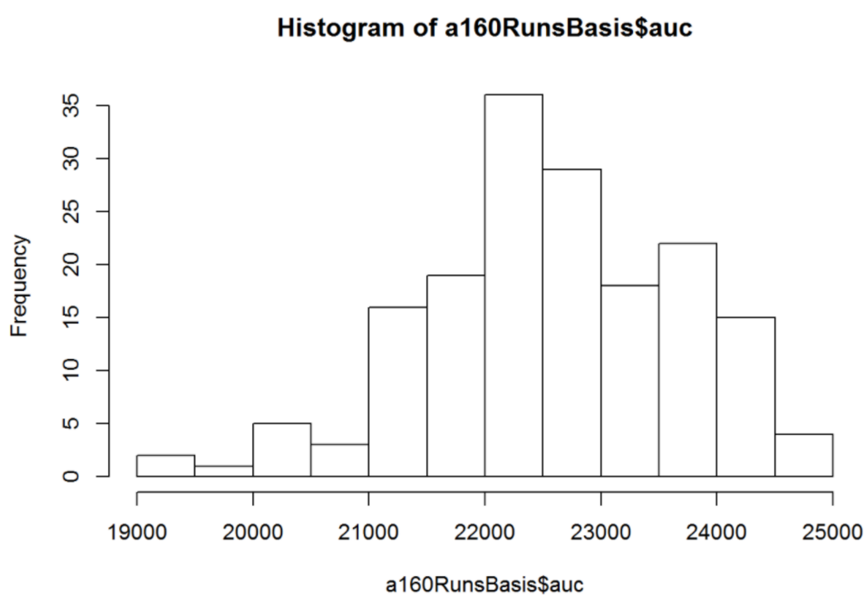


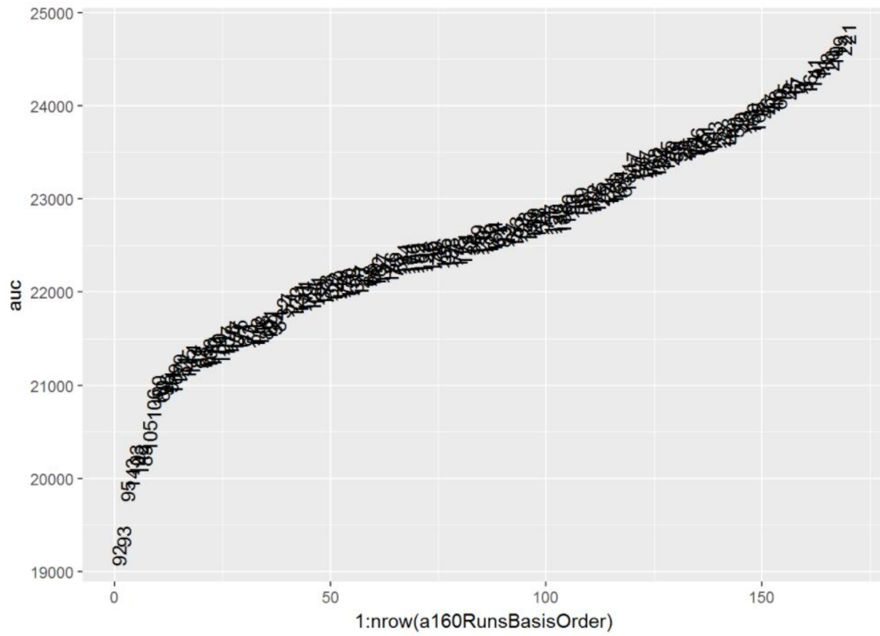
```
ggplot(data=a160Basis)+aes(x=Time,y=T)+geom_line(aes(color=as.factor(Cycle)))+ theme(legend.position='none')
```



more data prep to select golden run

```
#auc (aread under the curve)
a160RunsBasis$auc=0
for (j in 1:nrow(a160RunsBasis)) {
  k=a160RunsBasis$Cycle[j]
  cfRun=data.matrix(subset(a160Basis,Cycle==k & Time<=60)$T)
  a160RunsBasis$auc[j]=auc(c(1:61),cfRun,type='spline')  #note there are 61 rows per Cycle (time=0 to time=60)
}

hist(a160RunsBasis$auc)  #histogram
```






```
nrow(a160RunsBasisOrder)
```

```
## [1] 170
```

```
#pick middle 50%
indexStart=as.integer(nrow(a160RunsBasisOrder)/4)
indexEnd=as.integer(nrow(a160RunsBasisOrder)/4)*3
indexStart
```

```
## [1] 42
```

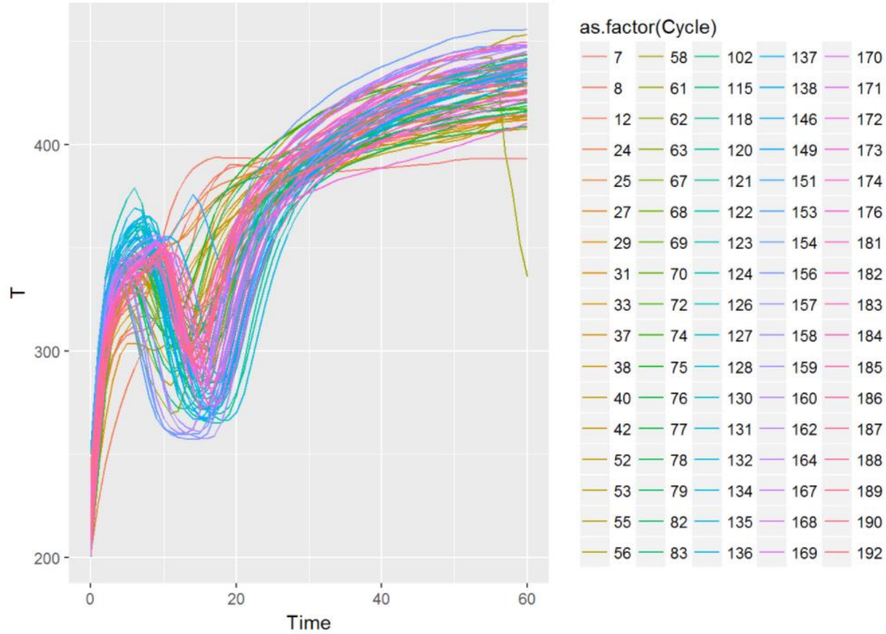
```
indexEnd
```

```
## [1] 126
```

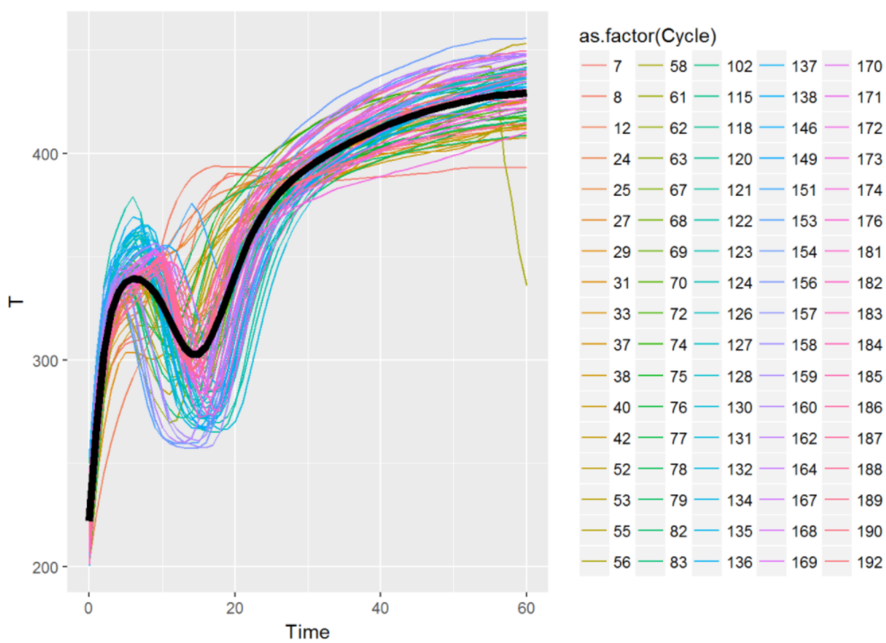
```
a160RunsBasisOrderMidQ=a160RunsBasisOrder[indexStart:indexEnd,]
nrow(a160RunsBasisOrderMidQ)
```

```
## [1] 85
```

```
a160BasisOrderMidQ=subset(a160Basis,Cycle %in% a160RunsBasisOrderMidQ$Cycle)

ggplot(data=a160BasisOrderMidQ)+aes(x=Time,y=T)+geom_line(aes(color=as.factor(Cycle)))
```



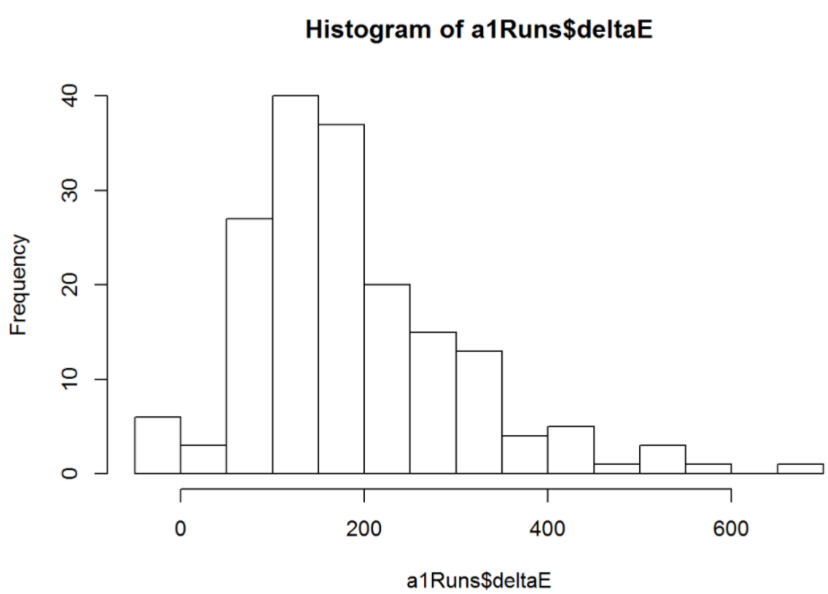

```
aTGold=a160BasisOrderMidQ %>% group_by(Time) %>% summarise(T=mean(T))

ggplot(data=a160BasisOrderMidQ)+aes(x=Time,y=T)+geom_line(aes(color=as.factor(Cycle))) +geom_line(data=aTGold, size=2)
```



testing and validation

```
endTime=61
endTimeText='6hrs'
deltaE30=300
deltaE50=350
deltaE60=400
deltaELimit=deltaE60

shapeOK=paste('Shape OK ',endTimeText)
shapeNotOK=paste('Shape Not OK ',endTimeText)
shapeBadData=paste('Shape bad data ',endTimeText)

#calc auc and distance metric
a1Runs$deltaE=0
a1Runs$deltaauc=0
aaucGold=auc(c(1:endTime),aTGold$T[1:endTime],type='spline')

for (j in 1:nrow(a1Runs)) {
  k=a1Runs$Cycle[j]
  cfRun=data.matrix(subset(a1,Cycle==k & Time<=60)$T)
  if (nrow(cfRun) < endTime)  {
    #a1Runs$delta[j]=-1
    a1Runs$deltaE[j]=-1
    a1Runs$deltaauc[j]=-1
  }
  else {
    #a1Runs$delta[j]=Frechet(data.matrix(aTGold$T),data.matrix(subset(a1,Cycle==j & Time<=60)$T))
    a1Runs$deltaE[j]=sqrt(sum((aTGold$T[1:endTime]-cfRun[1:endTime])^2))
    a1Runs$deltaauc[j]=aaucGold-auc(c(1:endTime),cfRun[1:endTime],type='spline')
  }
}
```

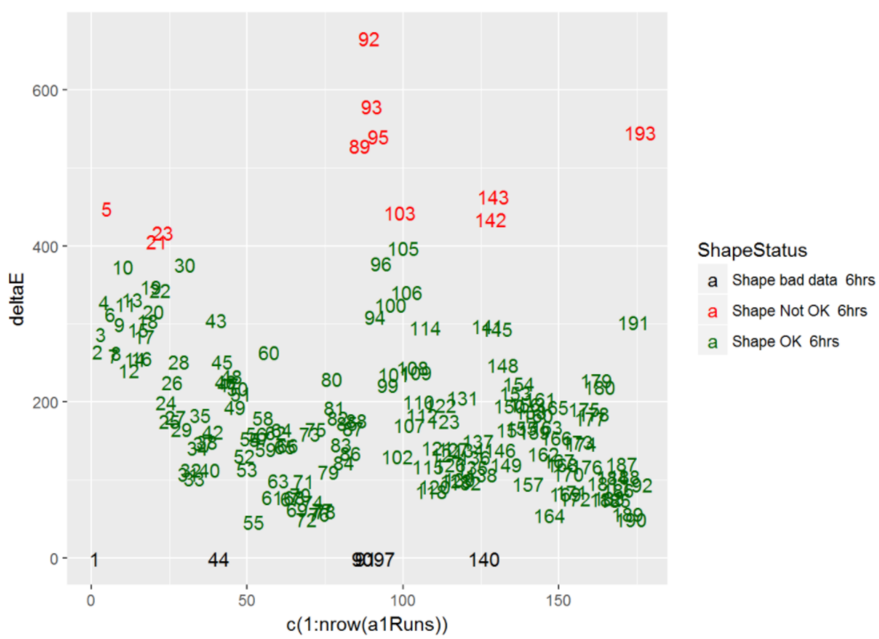
```
hist(a1Runs$deltaE,10)
```



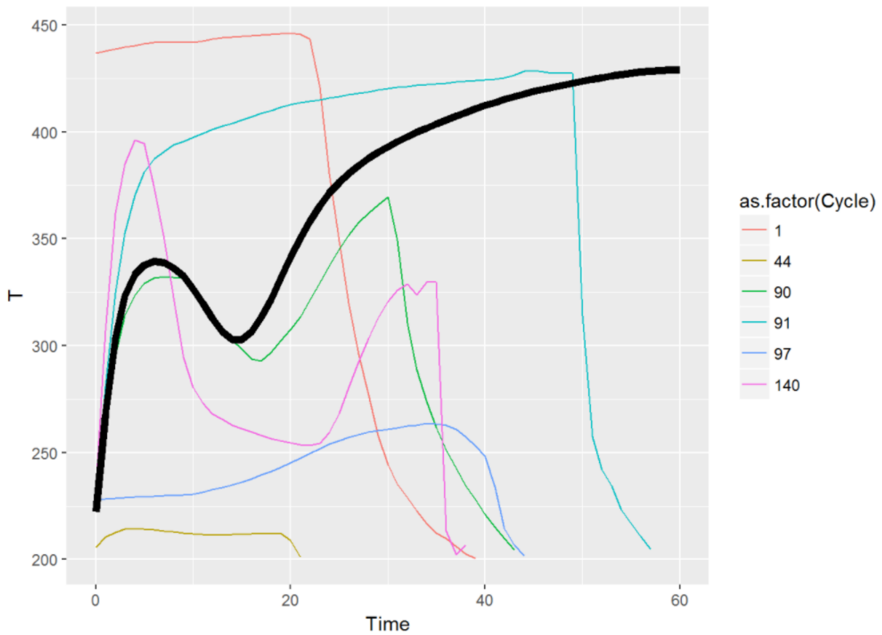
```
max(a1Runs$deltaE)
```

```
## [1] 665.8177
```

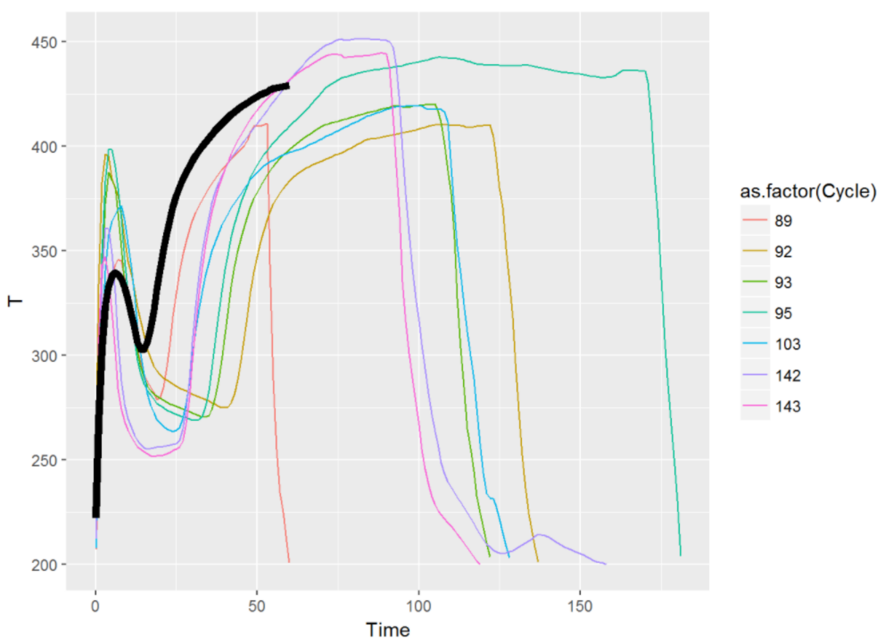
```
View(a1Runs)


a1Runs$ShapeNotOK=0
a1Runs$ShapeBadData=0
a1Runs$ShapeStatus=shapeOK

#criteria for alert
index = a1Runs$deltaE>deltaELimit

index2= a1Runs$deltaE==-1
a1Runs$ShapeStatus[index]=shapeNotOK
a1Runs$ShapeStatus[index2]=shapeBadData

a1Runs$ShapeNotOK[index]=1
a1Runs$ShapeBadData[index2]=1

a1Runs$ShapeNotOK=a1Runs$ShapeNotOK*a1Runs$Cycle*sign(a1Runs$deltaauc)  #ignore dissimilar shape if auc is higher
a1Runs$ShapeBadData=a1Runs$ShapeBadData*a1Runs$Cycle

mycolors = c("black","red","darkgreen")

ggplot(a1Runs)+aes(c(1:nrow(a1Runs)),deltaE)+geom_text(aes(label=Cycle, color=ShapeStatus)) + scale_colour_manual(values = m
ycolors)
```



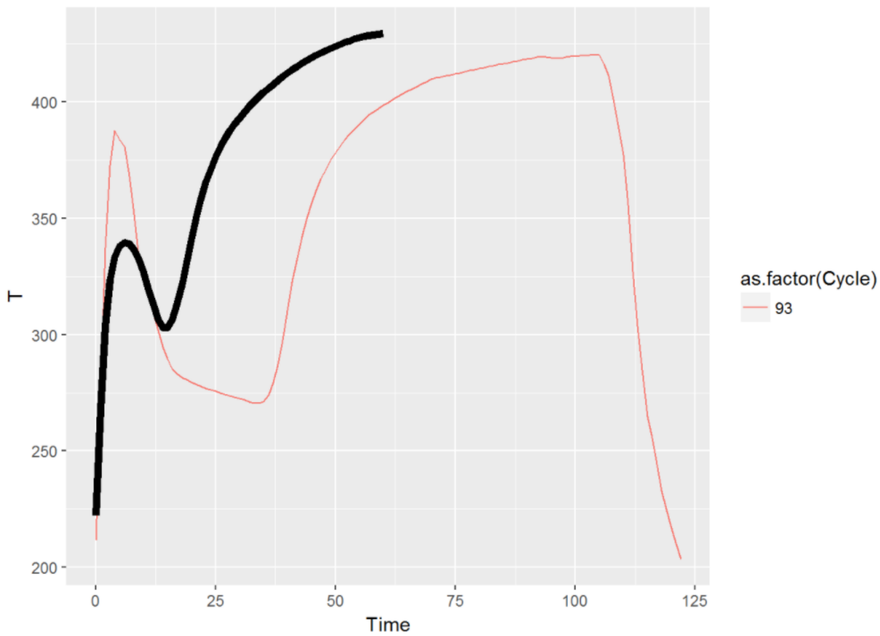

```
ggplot(data=subset(a1,Cycle%in%a1Runs$ShapeBadData))+aes(x=Time,y=T)+geom_line(aes(color=as.factor(Cycle)))+geom_line(data=a
TGold, size=2)
```




```
ggplot(data=subset(a1,Cycle%in%a1Runs$ShapeNotOK))+aes(x=Time,y=T)+geom_line(aes(color=as.factor(Cycle)))+geom_line(data=aTG
old, size=2)
```




```
ggplot(data=subset(a1,Cycle==93))+aes(x=Time,y=T)+geom_line(aes(color=as.factor(Cycle)))+geom_line(data=aTGold, size=2)
```



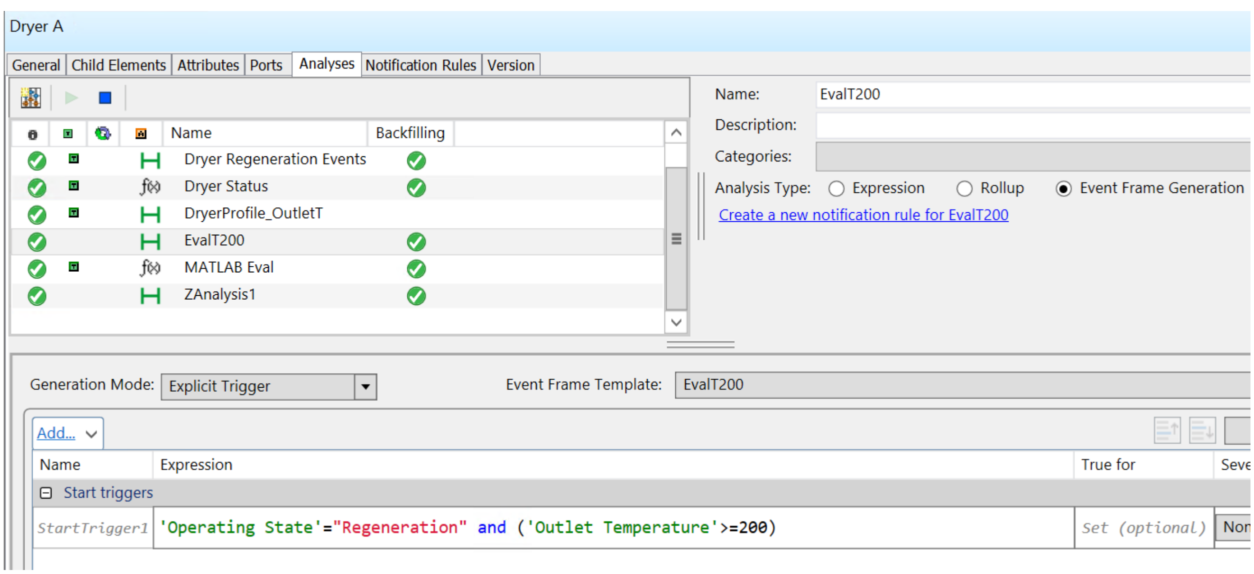

```
#end of script
```

1g: Operationalize the Golden Run

Expected temperature via PI future data tag

To operationalize the golden Dryer Outlet Temperature profile, it is written to a PI future tag (orange trace) for each Regeneration Cycle so that an operator can used it as a guide.

The EvalT200 generates an EF for outlet temperature>200 condition; this replicates the 200 deg F cut-off used during R/MATLAB model development.

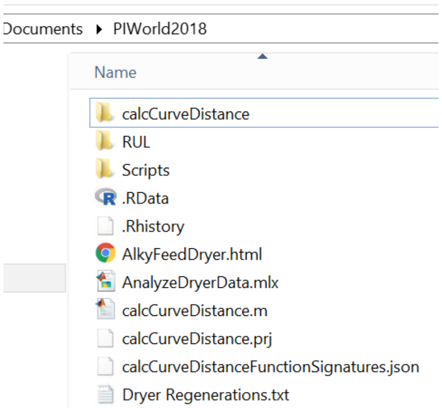


A background Windows Script (not shown here) writes the expected temperature to a PI future data Tag via PI UFL.

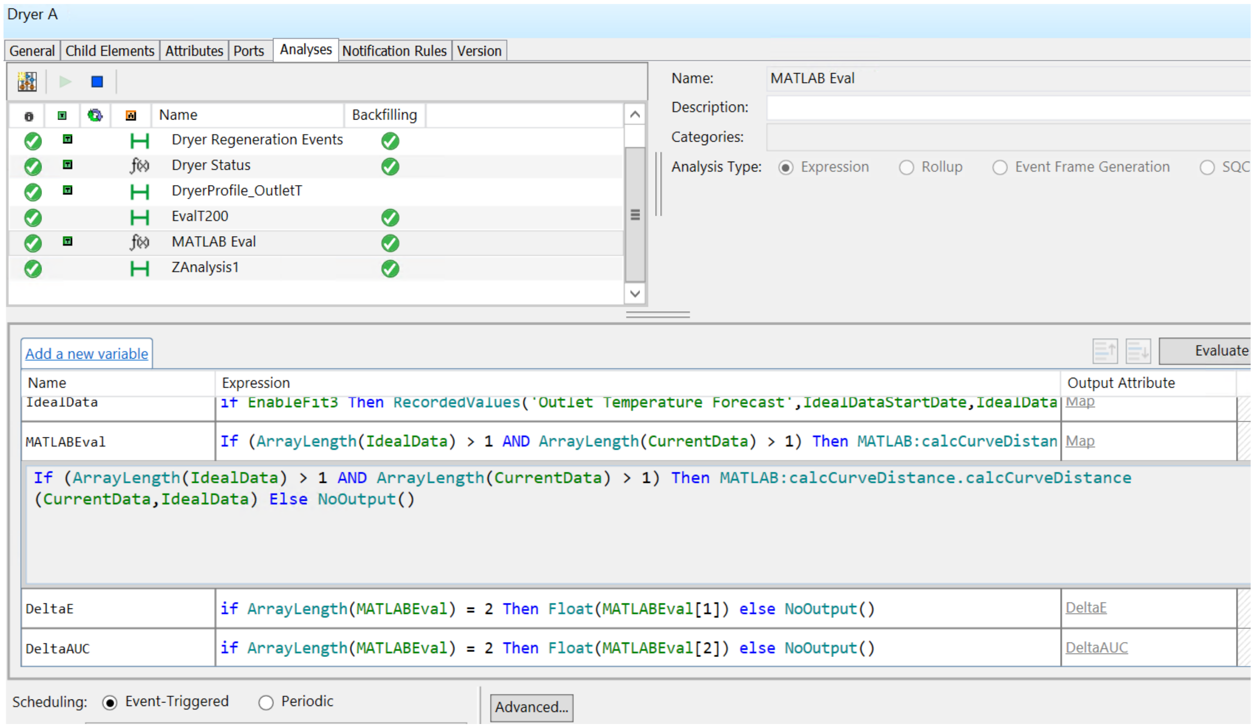
Real-time scoring of temperature profile during regeneration

For real-time scoring, the temperature profile during Regeneration is compared to the golden profile using the shape measures/distance metrics developed in R/MATLAB.

The logic is encoded in a MATLAB function that is callable from AF Asset Analytics. See files with *calcCurve* prefix in the lab folder.



MATLAB code compilation and its deployment into MATAB Production Server (MPS) is outside the scope of this Lab. For those details, see “*Streaming calculation with the PI System and MATLAB*” session.



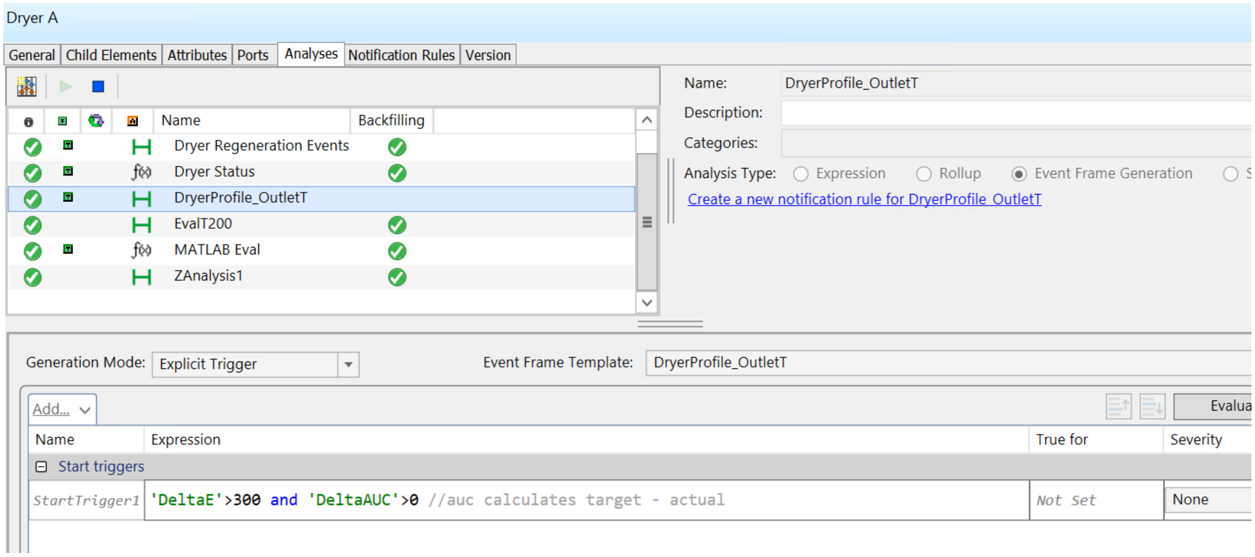
In the MATLABEval calculation (shown above), *CurrentData* is an array with current Regeneration Cycle Dryer Outlet Temperatures; *IdealData* is an array of Outlet Temperatures representing an ideal i.e. a golden run.

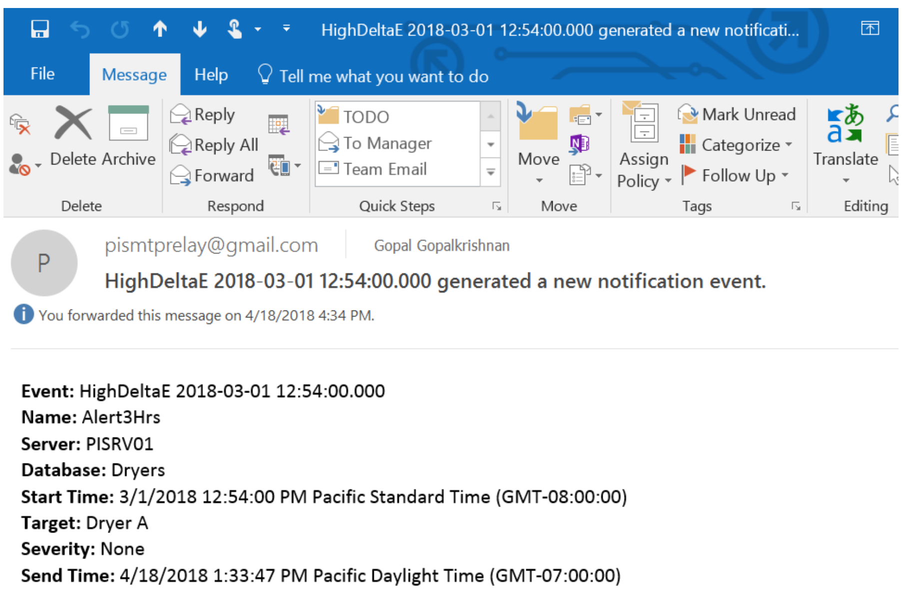
The function returns:

- DeltaE – shape metric comparing current profile and ideal profile
- DeltaAUC – delta of the AUC between current and ideal

MATLAB function signatures are specified in *calcCurveDistanceFunctionSignatures.json*

And, periodically, say, 3 hours into the process, PI Notification generates an email alert with an embedded link to a PI Vision display) if the temperature profile is dissimilar and outside specified limits.

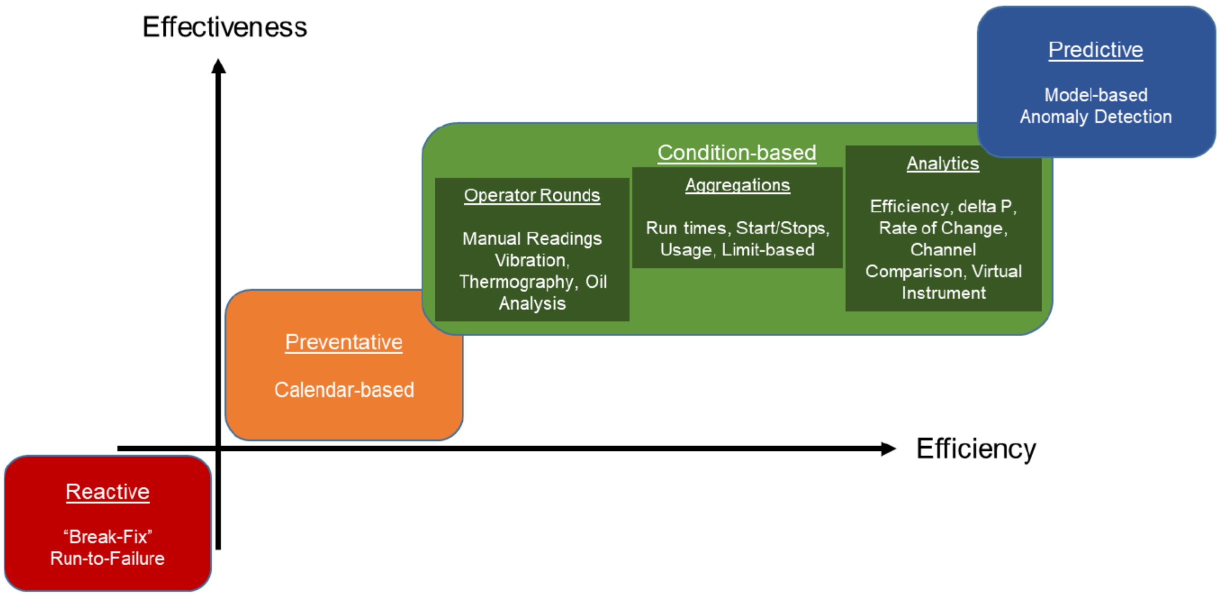

Dryer A
General | Child Elements | Attributes | Ports | Analyses | Notification Rules | Version
Name
Backfilling
Dryer Regeneration Events
Dryer Status
DryerProfile_OutletT
EvalT200
MATLAB Eval
ZAnalysis1
Name: DryerProfile_OutletT
Description:
Categories:
Analysis Type: Expression Rollup Event Frame Generation
Create a new notification rule for DryerProfile_OutletT
Generation Mode: Explicit Trigger
Event Frame Template: DryerProfile_OutletT
Add...
Name | Expression | True for | Severity
Start triggers
StartTrigger1 'DeltaE'>300 and 'DeltaAUC'>0 //auc calculates target - actual
Not Set
None


HighDeltaE 2018-03-01 12:54:00.000 generated a new notificati...
File Message Help Tell me what you want to do
Delete Archive
Reply
Reply All
Forward
TODO
To Manager
Team Email
Move
Assign Policy
Mark Unread
Categorize
Follow Up
Translate
Delete Respond Quick Steps Move Tags Editing
pismtprelay@gmail.com
Gopal Gopalkrishnan
HighDeltaE 2018-03-01 12:54:00.000 generated a new notification event.
You forwarded this message on 4/18/2018 4:34 PM.
Event: HighDeltaE 2018-03-01 12:54:00.000
Name: Alert3Hrs
Server: PISRV01
Database: Dryers
Start Time: 3/1/2018 12:54:00 PM Pacific Standard Time (GMT-08:00:00)
Target: Dryer A
Severity: None
Send Time: 4/18/2018 1:33:47 PM Pacific Daylight Time (GMT-07:00:00)

Exercise 2 Pump/Motor – Analytics

In an equipment/asset maintenance and reliability context, the layers of analytics can be viewed as:

- ***UbM – Usage-based Maintenance***
- ***CbM – Condition-based Maintenance***
- ***PdM – Predictive Maintenance - AF plus third party libraries***



PI System natively supports the required analytics for UbM and CbM.   Step-by-step tutorials are already available from hands-on lab sessions offered previously:

http://cdn.osisoft.com/learningcontent/pdfs/TechCon2016_ConditionBasedMaintenancewithPIAF.pdf

https://pisquare.osisoft.com/servlet/JiveServlet/download/96950-21243/TechCon%202017%20CBM%20Lab%20Workbook_Final.pdf

As such, in this lab, we illustrate PdM using vibration data for a pump to predict its remaining useful life (RUL).

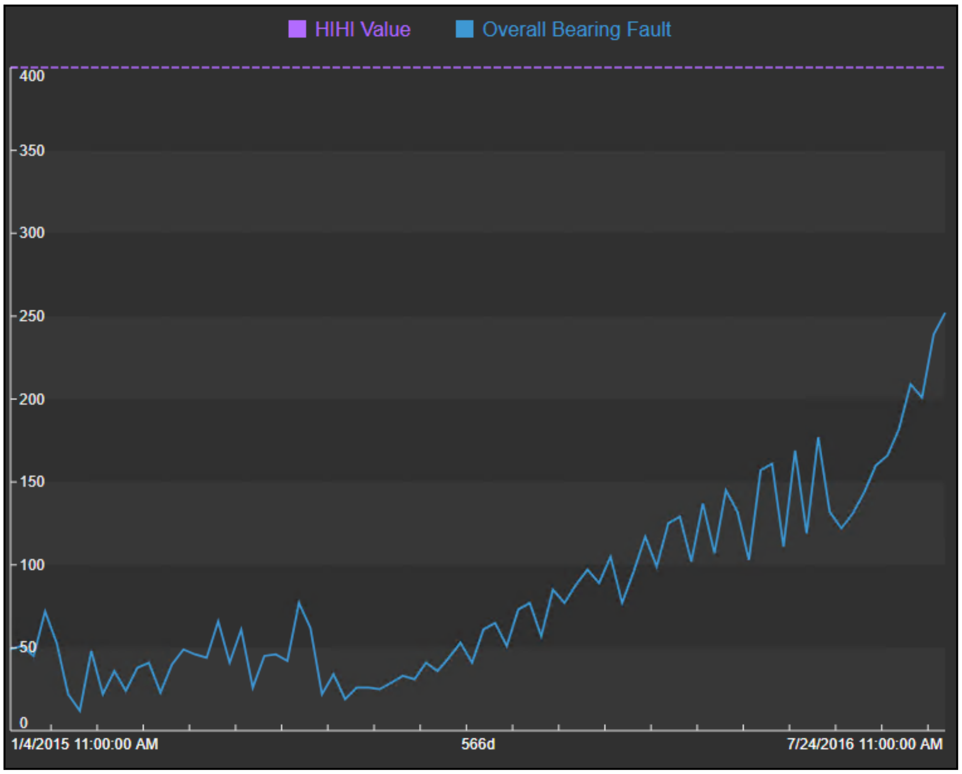
We will use 2 methods for this calculation:

- the first method uses built-in AF Asset Analytics function (linear regression via exponential curve-fit)
- the second method calls a MATLAB function (non-linear curve fit) from Asset Analytics

MATLAB code compilation and its deployment into MATAL Production Server (MPS) is outside the scope of this Lab. For those details, see "*Streaming calculation with the PI System and MATLAB*" session.

The vibration data represents the nominal percent (of normal) vibration over a period of 82 weeks; a new value is collected every week. The vibration values increase over time as you can see from the below graph.

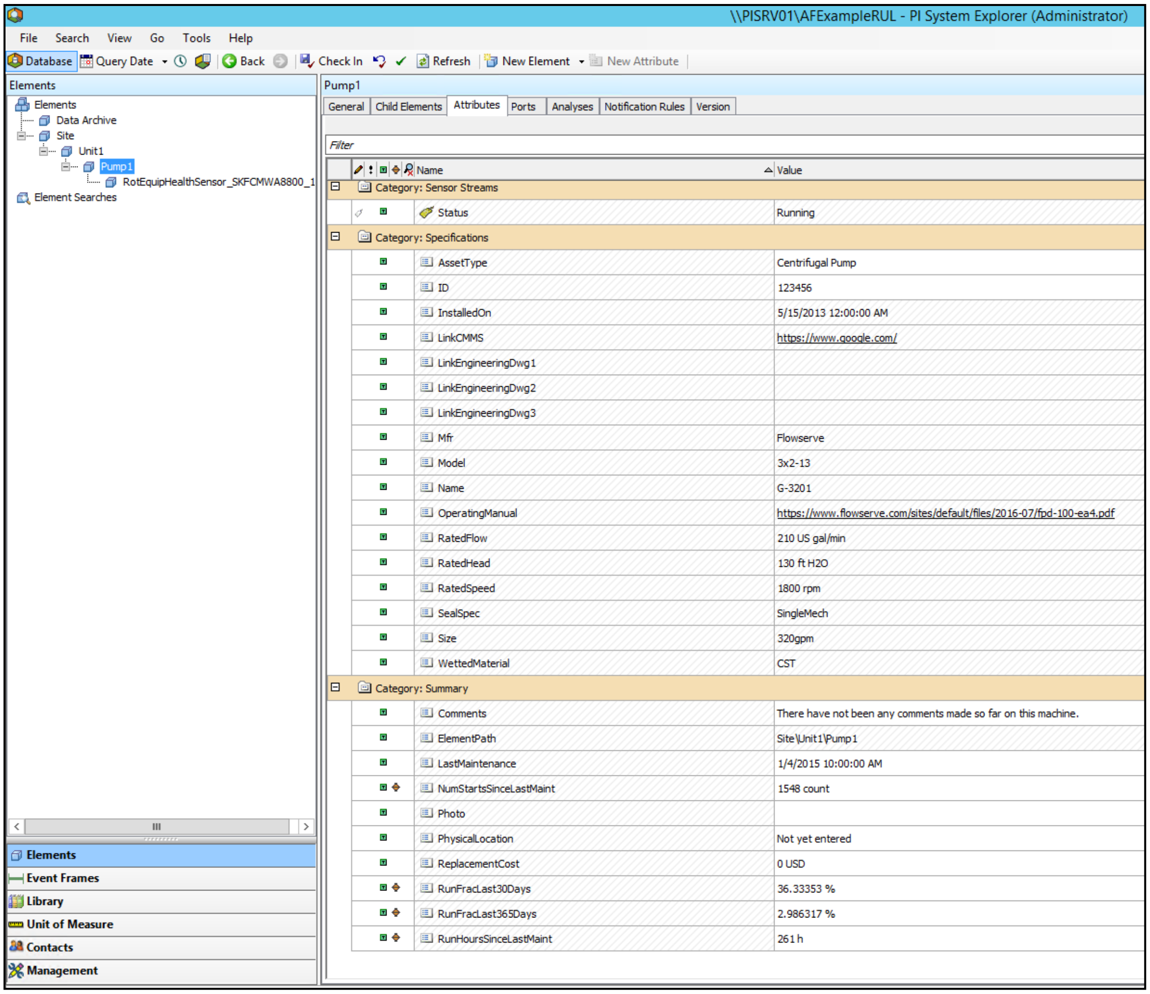
In this exercise we will calculate the estimated RUL till we reach the HIHI value of 400.



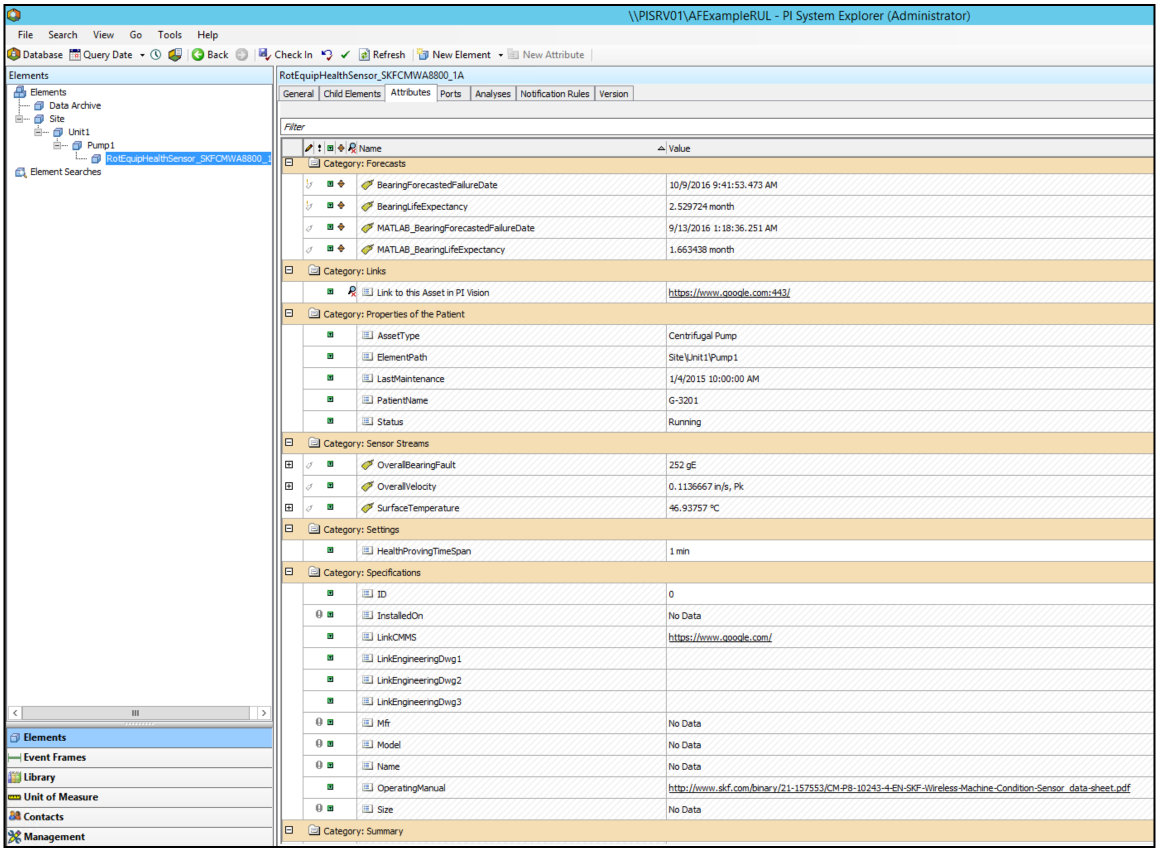
2a: Examine the Data

Let's first take a look at the AF structure:

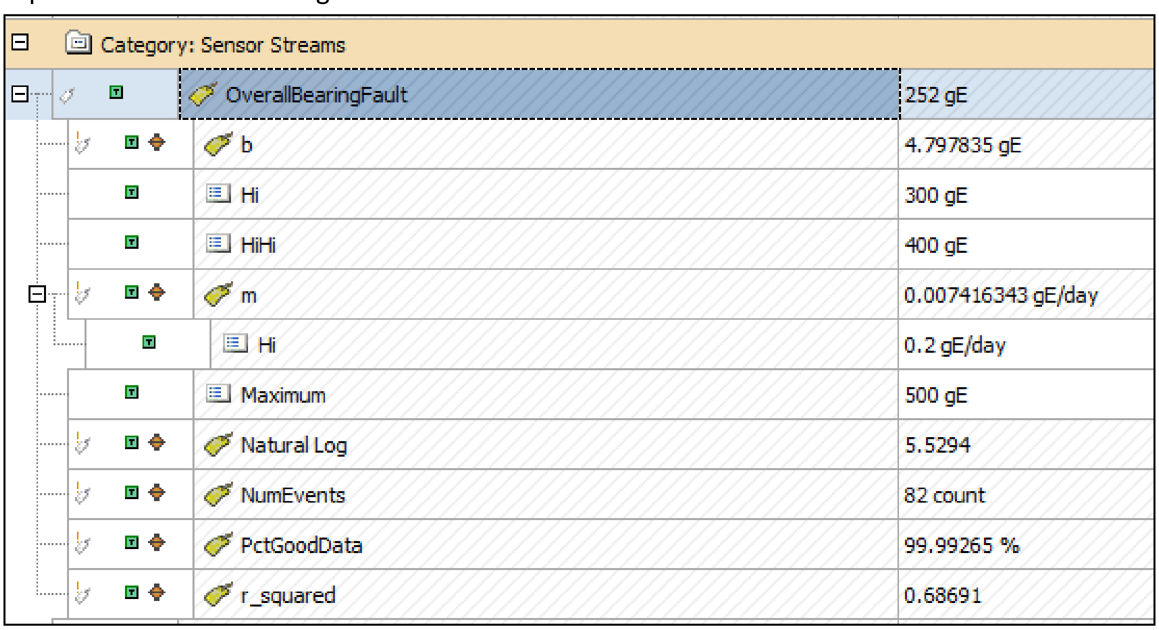
1. Open PI System Explorer
2. On the top left corner, click on Database, and then double click "AFExampleRUL"
3. Expand Site > Unit 1 > Pump 1
4. Click on "Pump 1" and then click the Attributes Tab to examine it



5. Click on the Rotating Equipment child element of “Pump 1”, and check its attributes



6. Expand the “OverallBearingFault” attribute and check its child attributes



7. Below are some important attributes we’ll be dealing with as part of the RUL calculation:
   a. **OverallBearingFault**: the vibration nominal value. We have 82 values for this attribute, values update every week, and will range from 49 to 252 over a period of 82 weeks

b. **OverallBearingFault|HIHI**: this is the HIHI value for the vibration; we need to calculate the RUL till we reach this HIHI value
c. **OverallBearingFault|Natural Log**: this will store the natural log (log base e) values of the vibration data. The linear fit will be done on the natural log values, as opposed to the actual vibration values
d. **OverallBearingFault|b**: this will store the intercept of the line being generated from the moving linear regression calculation we will implement in AF Analytics on the natural log values
e. **OverallBearingFault|m**: this will store the slope of the line being generated from the moving linear regression calculation we will implement in AF Analytics on the natural log values
f. **OverallBearingFault|r_squared**: this will store the $\mathbf{r^2}$ of the line being generated from the moving linear regression calculation we will implement in AF Analytics on the natural log values. The closer $\mathbf{r^2}$ is to 1, the better the fit
g. **BearingLifeExpectancy**: the calculated RUL in months till we reach the HIHI value. This will be calculated using a moving built-in linear regression function in PI Asset Analytics
h. **BearingForecastedFailureDate**: the predicted failure date based on the BearingLifeExpectancy calculation
i. **MATLAB_BearingLifeExpectancy**: the calculated RUL in months till we reach the HIHI value. This will be calculated using a MATLAB model already in place which does a non-linear fit for the data and "learns" as we gradually send more values to it
j. **MATLAB_BearingForecastedFailureDate**: the predicted failure date based on the MATLAB_BearingLifeExpectancy calculation

8. Please note that a MATLAB function was already implemented for this class and deployed in the MATLAB Production Server (MPS) instance installed on your machine. Below are some details of this MATLAB function:
    a. The function returns the RUL in days after passing to it the timestamps of the vibration values, the vibration values, and the threshold (the HIHI value of 400 in our example)
    b. Function definition: *predictNewRUL.predictNewRUL(**array time, array values, number threshold**)*

2b: Examine the moving linear regression calculations in AF Analytics – Method 1 for calculating the RUL

In this section, we will examine AF Analytics where we calculate the RUL based on a moving linear regression algorithm. This is implemented because when observing the data, you can clearly see that the curve has different slopes at different times.

1. Open PI System Explorer
2. On the top left corner, click on Database, and then double click "AFExampleRUL"
3. Expand Site > Unit 1 > Pump 1, and click on "RotEquipHealthSensor_SKFCMWA8800_1A"
4. Click the "Analyses" tab on the right
5. Check the "Bearing Life Calculations" analysis

6. The first 12 formulas of this analysis are related to this first method of calculating the RUL based on a moving linear regression fit.

   The last 4 formulas of this analysis calculate the RUL based on a non-linear MATLAB model, this will be discussed in the next section.

7. The first 12 formulas of the “Bearing Life Calculations” analysis which relate to this section are shown below:

<table>
<tr><th>Name</th><th>Expression</th><th>Output Attribute</th></tr>
<tr><td>pctgoodDATA</td><td>PctGood('OverallBearingFault','LastMaintenance', '*')</td><td>OverallBearingFault|PctGoodData</td></tr>
<tr><td>NumOfEvents</td><td>EventCount('OverallBearingFault','LastMaintenance', '*')</td><td>OverallBearingFault|NumEvents</td></tr>
<tr><td>EnableFit</td><td>/*'Status' = "Running" and */NumOfEvents > 24</td><td>Map</td></tr>
<tr><td>NatLog</td><td>if 'OverallBearingFault'>0 then Log('OverallBearingFault') else NoOutput()</td><td>OverallBearingFault|Natural Log</td></tr>
<tr><td>Time12back</td><td>'*-12w'</td><td>Map</td></tr>
<tr><td>LinearRegr</td><td>if EnableFit then LinRegr('OverallBearingFault|Natural Log', Time12back, '*',50) Else(NoOutput())</td><td>Map</td></tr>
<tr><td>Fit</td><td>if BadVal(LinearRegr) Then NoOutput() Else LinearRegr</td><td>Map</td></tr>
<tr><td>m</td><td>if EnableFit AND Not(BadVal(LinearRegr)) then Convert(Fit[1]*24*3600, "gE/day") else NoOutput()</td><td>OverallBearingFault|m</td></tr>
<tr><td>b</td><td>if EnableFit AND Not(BadVal(LinearRegr)) then Convert(Fit[2], "gE") else NoOutput()</td><td>OverallBearingFault|b</td></tr>
<tr><td>rsquared</td><td>if EnableFit AND Not(BadVal(LinearRegr)) then Convert(Fit[3],"ratio") else NoOutput()</td><td>OverallBearingFault|r squared</td></tr>
<tr><td>LifeExpectancy</td><td>if EnableFit AND Not(BadVal(LinearRegr)) and m>0 and rsquared>.400 then Convert((log('OverallBearingFault</td><td>BearingLifeExpectancy</td></tr>
<tr><td>ForecastedFailDate</td><td>if EnableFit AND Not(BadVal(LinearRegr)) and m>0 and rsquared>.400 then TimeStamp('OverallBearingFault')</td><td>BearingForecastedFailureDate</td></tr>
</table>

8. Below is a brief description of each formula:
   a. **pctgoodDATA**: checks the percentage of good data for the vibration values, from the Last Maintenance Date till the time of the calculation
   b. **NumOfEvents**: calculates the number of vibration values from the Last Maintenance Date till the time of the calculation
   c. **EnableFit**: a Boolean to determine if we can attempt to fit a linear line. We will enable the fit if we have more than 24 values
   d. **NatLog**: calculates the natural log value of the vibration data
   e. **Time12back**: gets the time 12 weeks earlier than the calculation time, which is 12 values in the past (because we have one value per week). This is used for doing a linear regression on every 12 values, since the slope of the curve is changing
   f. **LinearRegr**: performs a linear regression on the past 12 natural log values, and returns and array of 3 values; the slope of the line, the intercept, and the $\mathbf{r^2}$
   g. **Fit**: stores the array value from the previous formula assuming no errors were generated
   h. **m**: extracts the slope of the line generated from the array result of the **LinearRegr** formula
   i. **b**: extracts the intercept of the line generated from the array result of the **LinearRegr** formula
   j. **rsquared**: extracts the $\mathbf{r^2}$ value of the line generated from the array result of the **LinearRegr** formula
   k. **LifeExpectancy**: calculates the RUL till we reach the HIHI value of 400, based on the slope, the intercept, and the time of the calculation. The $\mathbf{r^2}$ is checked to make sure its

value is not too low, which might result in an unreliable RUL prediction. The prediction is converted to hours and mapped to the **BearingLifeExpectancy** attribute, which is set to convert hours to months in the attribute template

l. **ForecastedFailDate**: uses the calculated **LifeExpectancy** and the calculation time to determine the estimated fail date

2c: Examine the MATLAB calculations in AF Analytics – Method 2 for calculating the RUL

In this section, we will examine the calculations in AF Analytics which calls the *predictNewRUL* MATLAB function described in section 2a, in order to calculate the RUL based on a non-linear model which constantly "learns" as it gradually receives new data.

1. Open PI System Explorer
2. On the top left corner, click on Database, and then double click "AFExampleRUL"
3. Expand Site > Unit 1 > Pump 1, and click on "RotEquipHealthSensor_SKFCMWA8800_1A"
4. Click the "Analyses" tab on the right
5. Check the "Bearing Life Calculations" analysis
6. The last 4 formulas of this analysis are related to this second method of calculating the RUL using the integration with MATLAB. The first 12 formulas of this analysis were discussed in the previous section.
7. The last 4 formulas of the "Bearing Life Calculations" analysis which relate to this section are shown below:

RawValues	RecordedValues('OverallBearingFault', '*-12w', '*',"Inside")	Map
RawTimestamps	If BadVal(RawValues) Then NoOutput() Else TimeStamp(RawValues)	Map
LifeExpectancyMatlab	If (BadVal(RawTimestamps) OR BadVal(RawValues)) Then NoOutput() Else IF (ArrayLength(RawValues) = 0) Then	MATLAB BearingLifeExpectancy
ForecastedFailDateMatlab	If (BadVal(RawTimestamps) OR BadVal(RawValues)) Then NoOutput() Else (IF (ArrayLength(RawValues) = 0) the	MATLAB BearingForecastedFailureDate

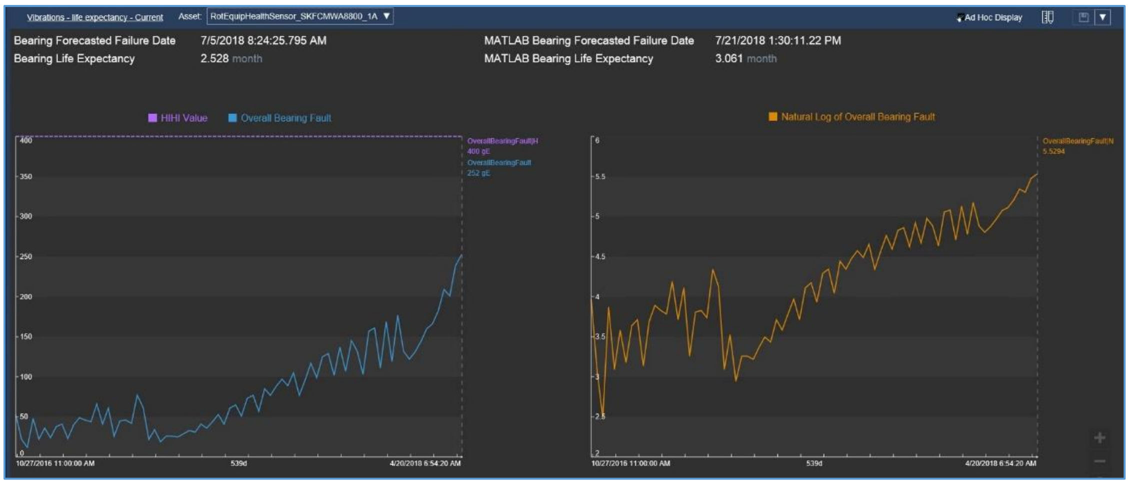
8. Below is a brief description of each formula:
    a. **RawValues**: retrieves an array of values of vibration data for the past 12 weeks (12 values in total)
    b. **RawTimestamps**: retrieves an array of timestamps of the values of vibration data for the past 12 weeks (12 timestamps in total)
    c. **LifeExpectancyMatlab**: calculates the RUL till we reach the HIHI value of 400 by calling the *predictNewRUL* MATLAB function and passing to it the **RawTimestamps array, RawValues array, and the Vibration HIHI Threshold of 400**. The prediction is converted to hours and mapped to the **MATLAB_BearingLifeExpectancy** attribute, which is set to convert hours to months in the attribute template
    d. **ForecastedFailDateMatlab**: uses the calculated **LifeExpectancyMatlab** and the calculation time to determine the estimated fail date

2d: Simulating the calculations

In this section, we will simulate receiving the vibration data over a period of 82 weeks and examine the results of the RUL calculations for both methods.

1. Open Internet Explorer and click on the "PI Vision" Bookmark

2. Use the tree menu on the left to navigate to the “Vibration RUL” displays
3. Click on the “Vibrations - life expectancy - Current” display
4. Open the following folder on your Desktop: Replay Data
5. Right click the “PI_ReplayVibrationValues.ps1” file and choose Edit. The file should open in PowerShell
6. Click the “Debug” tab from the top menu, and choose “Run/Continue”, this will start filling up the 82 vibration values and will call the “Bearing Life Calculations” Analysis at the end, to calculate the RUL using both methods
7. Go back to the PI Vision display you previously opened in step 3, and observe as the data is coming in, and the RUL calculations are displayed. ***Are both methods generating similar RUL values?***



Other resources

At TechCon 2016, we reviewed an end-to-end use case for developing a machine learning (multivariate PCA - principal component analysis) model to predict equipment failure.

And, in TechCon 2017, we covered an anomaly detection use case in HVAC air handler operations – using both PCA and one-class SVM algorithms.

Both labs are available at http://learning.osisoft.com VLE (virtual learning environment)

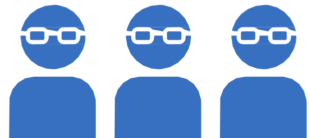

Have an idea how to improve our products?
OSIsoft wants to hear from you!
https://feedback.osisoft.com/

Save the Date!

OSIsoft PI World Users Conference in Barcelona. September 24-27, 2018.

Register your interest now to receive updates and notification early bird registration opening.

http://pages.osisoft.com/UC-CORP-Q3-18-EMEAUsersConference_RegisterYourInterest2018.html



CONTINUE YOUR PI SYSTEM LEARNING

After the conference, the PI SYSTEM LEARNING does not have to stop. All registered attendees for PI World SFO 2018 will have access to all PI World Hands-on Lab cloud environments for 21 days using the discount cod below. You will receive detailed instructions via email after the conference.

Discount Code: 2018UCSF-LAB-100

Offer expires June 30, 2018

learning.osisoft.com