---

title: "C-CDL-Ocillation-Builder-v01a"

author: 'Author: To\_Be\_Added'

date: "`r format(Sys.time(), '%d&period; %B %Y')`"

output:

 html\_document: default

 pdf\_document: default

 word\_document: default

subtitle: "Industrial Plant Assessment - Part B"

---

<style type="text/css">

body{ /\* Normal \*/

 font-size: 12px;

 }

td { /\* Table \*/

 font-size: 12px;

}

h1.title {

 font-size: 38px;

 color: DarkRed;

}

h1 { /\* Header 1 \*/

 font-size: 28px;

 color: DarkBlue;

}

h2 { /\* Header 2 \*/

 font-size: 22px;

 color: DarkBlue;

}

h3 { /\* Header 3 \*/

 font-size: 18px;

 font-family: "Times New Roman", Times, serif;

 color: DarkBlue;

}

code.r{ /\* Code block \*/

 font-size: 12px;

}

pre { /\* Code block - determines code spacing between lines \*/

 font-size: 12px;

}

</style>

# Introduction

The \*\*Industrial Plant Assessment\*\* (IPA) has been developed for the continuous processes such as refineries or chemical plants.

The following folders are used:

 N | Folder Name | Description

---|:--------------------------------|:----------------------

 1 | IPA-Industrial-Plant-Assessment | Top folder

It contains

 N | Folder Name | Description

---|:--------------------------------|:----------------------

 1 | IPA-Tools | IPA Tools such as XLSM and RMD

 2 | Project... | Project generated by IPA tools

The \*\*IPA-Tools\*\* folder contains

 N | Folder Name | Description

---|:--------------------------|:----------------------

 1 | A-Work-Process | Step by step documentation and presentations

 2 | B-Programs | R And Excel programs

 3 | C-References | Academic References

 3 | D-Sample-Data | Sample Data to check the program

The data and results of the analysis are stored in \*\*Project Folder\*\*:

 N | Folder Name | Description

---|:-----------------------------------|:----------------------

 1 | Proj-CDL-T123-Case-A-PC-2023-10-16 | Sample project folder

The project folder is divided into the following sub-folders:

 N | Folder Name | Description

---|:---------------------------|:----------------------

 1 | A-Raw-Data | Data collected from historian

 2 | B-Consolidate-Data | Raw data consolidated into one file

 3 | C-PCA-Analysis | Results of PCA Analysis

 4 | D-Oscillation-Analysis | Results of Oscillation Analysis

 5 | E-Controller-Data | Controllers data

 6 | F-Controller-Performance | Controllers Performance Analysis

 7 | R-Reports | Report of Analysis Analysis

## Steps of the work process

The following steps are execute:

### Select Tags

Use the documentation and select the Analog-Inputs (AI) and Analog-Outputs (AO)

 N | Step | Description

---|:----------------------------------------|:----------------------

 1 | Select Tags AI and AO | Get list of tags

 2 | Check Historian | Check if Tags are stored very minutes

### Collect Raw Data and consolidate Data

 N | Step | Description

---|:----------------------------------------|:----------------------

 1 | Collect minute data (3 to 7 days) | Normally collect 100 tags store in A-Raw-Data folder

 2 | cleanup raw data | Review and cleanup data

 3 | Consolidate raw data | Consolidate all Raw data in one CVS file

### Run PCA-Analysis

 N | Step | Description

---|:----------------------------------------|:----------------------

 1 | Start RStudio | Platform to run RMD Programs

 2 | Open B-CDL-PCA-Builder-v01c.Rmd | Open it from b-Program folder

 3 | Click the Knit buttons | generate a HTML report

### Run Ocillation-Analysis

 N | Step | Description

---|:----------------------------------------------|:----------------------

 1 | Start RStudio | Platform to run RMD Programs

 2 | Open C-CDL-Ocillation-Builder-2023-07-03a.Rmd Open it from b-Program folder

 3 | Click the Knit buttons | generate a HTML report

## Imported Setup Data

The external data needed is imported from file \*\*CDM\_Setup.CSV\*\* in the program folder \*\*B-Programs\*\*.

```{r 0.0 setup echo , echo = FALSE}

 # +-------------------------------------------------------------------------------------------

 # | rm(list = setdiff(ls(), lsf.str())) will remove all variables

 # +-------------------------------------------------------------------------------------------

 rm(list = setdiff(ls(), lsf.str()))

 # +-------------------------------------------------------------------------------------------

 # | Set echo\_flag to TRUE if you want to see the code when running KNIT

 # +-------------------------------------------------------------------------------------------

 echo\_flag = TRUE

 warning\_flag = FALSE

```

```{r 0.1 definition , warning= warning\_flag, echo = echo\_flag}

# Chunk output can be customized with knitr options, arguments set in the {} of a chunk header.

# we use five arguments:

#

# include = FALSE

# prevents code and results from appearing in the finished file.

# R Markdown still runs the code in the chunk, and the results can be used by other chunks.

#

# echo = FALSE

# prevents code, but not the results from appearing in the finished file.

# This is a useful way to embed figures.

# message = FALSE

# prevents messages that are generated by code from appearing in the finished file.

# warning = FALSE

# prevents warnings that are generated by code from appearing in the finished.

# fig.cap = "..."

# adds a caption to graphical results.

```

```{r 1.0 - setup, warning= warning\_flag, echo = echo\_flag}

 # +-------------------------------------------------------------------------------------------

 # | the knitr::opts\_chunk$set(echo = FALSE) in a chunk at the beginning of your document

 # | is the same of having {r echo = FALSE} for all chunks.

 # +-------------------------------------------------------------------------------------------

 knitr::opts\_chunk$set(echo = FALSE)

 # +-------------------------------------------------------------------------------------------

 # | date\_format <-"%Y-%m-%d %H:%M"

 # | where

 # | %Y Year

 # | %m Month

 # | %d Day

 # | %H Hour

 # | %M Minute

 # +-------------------------------------------------------------------------------------------

 Sys.setenv(TZ="UTC")

 date\_format <-"%Y-%m-%d %H:%M"

 # +-------------------------------------------------------------------------------------------

 # |start\_time <- Sys.time()

 # |start\_time is set to current time like "2023-06-29 17:12:44 EDT"

 # +-------------------------------------------------------------------------------------------

 start\_time <- Sys.time()

 # +-------------------------------------------------------------------------------------------

 # |Assign cdl\_engine\_path to the program folder name

 # +-------------------------------------------------------------------------------------------

 cdl\_engine\_path <- dirname(rstudioapi::getSourceEditorContext()$path)

 # +-------------------------------------------------------------------------------------------

 # | List all files in the cdl\_engine\_path

 # +-------------------------------------------------------------------------------------------

 dir(cdl\_engine\_path)

 # +-------------------------------------------------------------------------------------------

 # | Read CDL\_Setup.CSV file located in the cdl\_engine\_path folder

 # | read.csv function that reads a CSV file

 # | paste0 function that add string together

 # | / path split character for R (Windows is \ )

 # | header TRUE means file first row is a header

 # +-------------------------------------------------------------------------------------------

 sysinf <- Sys.info()

 os <- sysinf['sysname']

 if (os =="Darwin") {

 RCmdData <- read.csv (paste0(cdl\_engine\_path,"/","Setup\_MAC.csv"),header=TRUE)

 } else {

 RCmdData <- read.csv (paste0(cdl\_engine\_path,"/","Setup\_PC.csv"),header=TRUE)

 }

 # +-------------------------------------------------------------------------------------------

 # | Assign CSV file data to R variables

 # | as.vector(RCmdData[row,column])

 # |

 # +-------------------------------------------------------------------------------------------

 proj\_folder <- as.vector(RCmdData[1,3])

 company\_name <- as.vector(RCmdData[2,3])

 case\_name <- as.vector(RCmdData[3,3])

 first\_col\_AI <- as.vector(RCmdData[4,3])

 last\_col\_AI <- as.vector(RCmdData[5,3])

 first\_col\_AO <- as.vector(RCmdData[6,3])

 last\_col\_AO <- as.vector(RCmdData[7,3])

 first\_row <- as.vector(RCmdData[8,3])

 last\_row <- as.vector(RCmdData[9,3])

 n\_acv <- as.vector(RCmdData[10,3])

 n\_cluster <- as.vector(RCmdData[11,3])

 install\_flag <- as.vector(RCmdData[12,3])

 echo\_flag <- as.vector(RCmdData[13,3])

 info\_flag <- as.vector(RCmdData[14,3])

 data\_file <- as.vector(RCmdData[15,3])

 glos\_file <- as.vector(RCmdData[16,3])

 info\_file <- as.vector(RCmdData[17,3])

 perl <- as.vector(RCmdData[18,3])

 # +-------------------------------------------------------------------------------------------

 # | The folder name in Windows is separated by character "\" in R it should be changed to "/"

 # |

 # +-------------------------------------------------------------------------------------------

 proj\_folder <- gsub('\\\\', '/', proj\_folder)

```

The following setup data is imported

 N | What | value

---|:--------------------|:-----------------

 1 | Project Folder | `r proj\_folder`

 2 | Company Name | `r company\_name`

 3 | Case Name | `r case\_name`

 4 | First Column of AI | `r first\_col\_AI`

 5 | Last Column of AI | `r last\_col\_AI`

 6 | First Column of AO | `r first\_col\_AO`

 7 | Last Column of AO | `r last\_col\_AO`

 8 | N Auto-Correlation | `r n\_acv`

 9 | N of Cluster | `r n\_cluster`

10 | Lib Install Flag | `r install\_flag`

11 | Echo Flag | `r echo\_flag`

12 | Info Flag | `r info\_flag`

13 | Data File | `r data\_file`

14 | Glos File | `r glos\_file`

15 | Info File | `r info\_file`

16 | Perl Folder | `r perl`

## Install Liberaries

The following data has been set by the setup file

 N | what | Value

 ---|:---------------------------|:----------

 1 | install\_flag | `r install\_flag`

```{r 1.2 - Install Liberaries, warning= FALSE, echo = FALSE}

 # +---------------------------------------------------------------------------

 # |

 # | Installing required packages

 # | install.packages("package name")

 # |

 # +----------------------------------------------------------------------------

 # The install\_flag must be set to TRUE in the CDL\_Setup.CSV

 if (install\_flag == TRUE ) {

 install.packages("corrr") # Correlations in R

 install.packages("dplyr") # A Grammar of Data Manipulation

 install.packages("dygraphs") # Interface to 'Dygraphs' Interactive Time Series Charting Library

 install.packages("gdata") # Various R Programming Tools for Data Manipulation

 install.packages("ggplot2") # Create Elegant Data Visualisations Using the Grammar of Graphics

 install.packages("gtools") # Various R Programming Tools

 install.packages("knitr") # A General-Purpose Package for Dynamic Report Generation in R

 install.packages("lares") # Analytics & Machine Learning Sidekick

 install.packages("lubridate") # Make Dealing with Dates a Little Easier

 install.packages("markdown") # Render Markdown with 'commonmark'

 install.packages("mclust") # Gaussian Mixture Modelling for Model-Based Clustering, Classification

 install.packages("onion") # Octonions and Quaternions

 install.packages("PerformanceAnalytics") # Econometric Tools for Performance and Risk Analysis

 install.packages("plot3D") # Plotting Multi-Dimensional Data

 install.packages("pracma") # Practical Numerical Math Functions

 install.packages("quantmod") # Quantitative Financial Modelling Framework

 install.packages("rgl") # 3D Visualization Using OpenGL

 install.packages("rmarkdown") # Dynamic Documents for R

 install.packages("stringr") # Simple, Consistent Wrappers for Common String Operations

 install.packages("tibble") # Simple Data Frames

 install.packages("tidyr") # Tidy Messy Data

 install.packages("xts") # eXtensible Time Series

 install.packages("zoo") # S3 Infrastructure for Regular and Irregular Time Series (Z's O

 #

 # https://www.rdocumentation.org/packages/anomalyDetection/versions/1.0

 #

 install.packages("AnomalyDetection") # Anomaly Detection

 }

```

```{r 1.3 - Load Liberaries, warning= FALSE, echo = FALSE}

 # +---------------------------------------------------------------------------

 # |

 # | Load all needed libraries

 # |

 # +---------------------------------------------------------------------------

 library(corrr)

 library(dplyr)

 library(dygraphs)

 library(gdata)

 library(ggplot2)

 library(gtools)

 library(knitr)

 library(lares)

 library(lubridate)

 library(markdown)

 library(mclust)

 library(onion)

 library(PerformanceAnalytics)

 library(plot3D)

 library(pracma)

 library(quantmod)

 library(rgl)

 library(rmarkdown)

 library(stringr)

 library(tibble)

 library(tidyr)

 library(xts)

 library(zoo)

 # +---------------------------------------------------------------------------

 # |

 # | List all loaded Libraries if info\_flag is set to TRUE in the CDL\_Setup.csv

 # |

 # +---------------------------------------------------------------------------

 if (info\_flag) {

 my\_packages <- library()$results[,c(1,3)]

 nrow(my\_packages)

 View(my\_packages)

 }

```

The following Libraries are installed:

N | Library | Description

:--|:---------------------|:------------------------------------------------------------------------------

1 | dygraphs | Interface to 'Dygraphs' Interactive Time Series Charting Library

2 | gdata | Various R Programming Tools for Data Manipulation

3 | ggplot2 | Create Elegant Data Visualizations Using the Grammar of Graphics

4 | gtools | Various R Programming Tools

5 | knitr | A General-Purpose Package for Dynamic Report Generation in R

6 | lubridate | Make Dealing with Dates a Little Easier

7 | markdown | Render Markdown with the C Library 'Sundown'

8 | mclust | Gaussian Mixture Modeling for Model-Based Clustering

9 | onion | Octonions and Quaternions

10 | PerformanceAnalytics | Econometric Tools for Performance and Risk Analysis

11 | plot3D | Plotting Multi-Dimensional Data

12 | pracma | Practical Numerical Math Functions

13 | quantmod | Quantitative Financial Modelling Framework

14 | rgl | 3D Visualization Using OpenGL

15 | rmarkdown | Dynamic Documents for R

16 | xts | EXtensible Time Series

17 | zoo | S3 Infrastructure for Regular and Irregular Time Series

# Read Consolidated Data

Read the data from the data file from the \*\*B-Consolidate-Data\*\* folder

The data is store as

Time Stamp | AI-Tag1 | AI-Tag2 | ... | AI-TagN | AO-Tag1 | AO-Tag2 | ... | AO-TagM |

:---------------|:--------|:--------|:----|:--------|:--------|:--------|:----|:--------|

2020-03-21 00:01|FT.1101 |FT1102 | ... |TT-1699 | CV-1101 | CV-1102 | ... |CV-1617 |

\* AI Columns: start at 2 to 283

\* AO Columns: start at 284 to 333

\* number of Rows: 2881

The data is store it in the following

\* all\_data: contains all data

\* AI\_data : sub-set of all\_data colum 1--283

\* AO\_data : sub-set of all\_data columns 1 and 284 to 333

```{r 2.1-read-data, warning= warning\_flag, echo = echo\_flag}

 # +------------------------------------------------------------------------

 # | Read data

 # | proj\_folder == defined in CDL\_Setup.CSV

 # | B-Consolidate-Data == folder name of consolidated data

 # | data\_file == file name of consolidated data

 # +------------------------------------------------------------------------

 csv\_file = paste(proj\_folder,"/","B-Consolidate-Data/",data\_file,sep="")

 # +------------------------------------------------------------------------

 # | store all data in the "all\_data"

 # +------------------------------------------------------------------------

 all\_data <<- read.csv(csv\_file,sep=",")

 # +------------------------------------------------------------------------

 # | Replace the first column name to "TimeStamp"

 # +------------------------------------------------------------------------

 colnames(all\_data)[1] <- "TimeStamp"

 # +------------------------------------------------------------------------

 # | Assign all\_n\_rows to number of rows in all\_data

 # | Assign all\_n\_cols to number of columns in all\_data

 # +------------------------------------------------------------------------

 all\_n\_rows <<- nrow(all\_data)

 all\_n\_cols <<- ncol(all\_data)

 # +------------------------------------------------------------------------

 # | extract AI data

 # +------------------------------------------------------------------------

 AI\_data <- all\_data[,c(1,first\_col\_AI:last\_col\_AI)]

 AI\_n\_rows <<- nrow(AI\_data)

 AI\_n\_cols <<- ncol(AI\_data)

 # +------------------------------------------------------------------------

 # | extract AO data

 # +------------------------------------------------------------------------

 AO\_data <- all\_data[,c(1,first\_col\_AO:last\_col\_AO)]

 AO\_n\_rows <<- nrow(AO\_data)

 AO\_n\_cols <<- ncol(AO\_data)

```

The data is reduced to

What | value

:-----------------------------------|:-----------------

All Data Number of Rows | `r all\_n\_rows`

All Data Number of Columns | `r all\_n\_cols`

First Row (from Setup) | `r first\_row`

Last Row (from Setup) | `r last\_row`

AI Data Number of Rows | `r AI\_n\_rows`

AI Data Number of Columns | `r AI\_n\_cols`

AO Data Number of Rows | `r AO\_n\_rows`

AO Data Number of Columns | `r AO\_n\_cols`

## Define Result Folder

The following folders or files are at the top folder

```{r 2.2-create sub-folder,warning= warning\_flag, echo = echo\_flag}

 # +------------------------------------------------------------------------

 # | define a subfolder name

 # +------------------------------------------------------------------------

 subfolder <- paste("/D-Oscillation-Analysis","/",case\_name, "-Analysis","/", sep = "")

 # +------------------------------------------------------------------------

 # | Create sub folder

 # +------------------------------------------------------------------------

 dir.create(file.path(proj\_folder, subfolder), showWarnings = FALSE)

 # +------------------------------------------------------------------------

 # | list files in case there is any

 # +------------------------------------------------------------------------

 list.files(path = ".")

 # +------------------------------------------------------------------------

 # | delete all existing files using unlink function

 # +------------------------------------------------------------------------

 unlink(paste(proj\_folder,subfolder,"/\*",sep=""),recursive = TRUE)

```

```{r 2.3 -write files,warning= warning\_flag, echo = echo\_flag}

 # +------------------------------------------------------------------------

 # | save input data for future use

 # | tempfile == folder + subfolder + filename

 # | write.csv write the data to tempfile with row.name set to TRUE

 # +------------------------------------------------------------------------

 tempfile = paste(proj\_folder,subfolder,"01-input-data.csv",sep="")

 write.csv(all\_data, tempfile, row.names=T)

 tempfile = paste(proj\_folder,subfolder,"02-input-data-summary.csv",sep="")

 write.csv(summary(all\_data), tempfile, row.names=T)

```

The following files are stored in 'r subfolder`

What | value

:-------------------------|:-----------------

Project Folder | `r proj\_folder`

sub Folder | `r subfolder`

all data | 01-input-data.csv

all data Summary | 02-input-data-summary.csv

# Build Frequencies

```{r 3.1 build frequency, warning= warning\_flag, echo = echo\_flag}

 #

 # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

 # |Build\_Frequency - [1] Build empty freqsummarydata

 # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

 #

 numberofrows = all\_n\_rows

 n\_acv <- as.numeric(n\_acv)

 nhalf = n\_acv / 2

 freqdata <- matrix(nrow=numberofrows,ncol=n\_acv+1)

 freqfile <- paste(proj\_folder,subfolder,"03-freq.csv",sep="")

 fftdata <- matrix(nrow=numberofrows,ncol=nhalf+1)

 fftfile = paste(proj\_folder,subfolder,"04-fft.csv",sep="")

 freqsummarydata <- data.frame( "tagname" = character(),

 "frq" = integer(),

 "amp" = numeric(),stringsAsFactors=FALSE)

 freqsummaryfile = paste(proj\_folder,subfolder,"05-freq-summary.csv",sep="")

 ctrlsummarydata <- NULL

 ctrlsummarydata <- data.frame( "tagname" = character(),

 "cova-index" = numeric(),

 "fft-index" = numeric(),

 "ctrl-index" = numeric(),stringsAsFactors=FALSE)

 ctrlsummaryfile = paste(proj\_folder,subfolder,"06-ctrl-index.csv",sep="")

 clustersummaryfile = paste(proj\_folder,subfolder,"07-cluster.csv",sep="")

```

```{r 3.1.1 frequecy analysis}

 freqvec <- data.frame(t(freqdata))

 colnames(freqvec) <- freqvec[1,]

 freqvec <- freqvec[-1, ]

 freqvec = as.data.frame(sapply(freqvec, as.numeric))

 #freqcor <- correlate(freqvec)

 #ncolumns <- 20 # ncol(freqvec)

 #chart.Correlation(freqvec[,1:ncolumns], histogram=TRUE, pch=19)

```

```{r 3.2 ,echo=FALSE}

 #

 # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

 # |Build\_Frequency - [2] Process each tag

 # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

 #

 firstcol <- 2

 lastcol <- last\_col\_AO

 nrows <- all\_n\_rows

 for (iloop in firstcol:lastcol) {

 i <- as.numeric(iloop)

 tagname = colnames(all\_data)[i]

 pv <- all\_data[,i]

 #

 # check for non zero sd

 #

 pv.mean <- mean(pv)

 pv.sd <- sd(pv)

 if (pv.sd == 0 ){next}

 #

 # calculate erroe

 #

 error <- pv - pv.mean

 #

 # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

 # |Build\_Frequency -[3] start plots

 # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

 #

 #

 par(mfrow=c(1,1))

 jpeg(paste(proj\_folder,subfolder,tagname,".png",sep=""),width=11.0,height=8.0,units="in",res=1200)

 par(fig = c(0.0, 0.7, 0.55, 1.0))

 #

 # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

 # |Build\_Frequency - [3.1] pv and average +/- SD

 # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

 # +~~~~~~~~~~~~~+~~~~~~~~+

 # | Pv | |

 # +~~~~~~~~~~~~~+~~~~~~~~+

 # | | |

 # +~~~~~~~~~~~~~+~~~~~~~~+

 # | | |

 # +~~~~~~~~~~~~~+~~~~~~~~+

 #

 mainlabel=paste("File: ",data\_file," TagName: ",tagname,sep="")

 plot(pv,type = "l", ylab=tagname, xaxt ='n', xlab="", main=mainlabel)

 timelab = paste("Begin: ",all\_data[1,1]," End: ",all\_data[nrow(all\_data),1])

 mtext(timelab,side=1)

 abline(h=pv.mean, col="red",lty=4)

 pvupsd <- pv.mean+pv.sd

 abline(h=pvupsd, col="green",lty=3)

 pvdnsd <- pv.mean-pv.sd

 abline(h=pvdnsd, col="green",lty=3)

 #

 # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

 # |Build\_Frequency - [3.2] plot historgram

 # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

 # +~~~~~~~~~~~~~+~~~~~~~~+

 # | | hist(H)|

 # +~~~~~~~~~~~~~+~~~~~~~~+

 # | | |

 # +~~~~~~~~~~~~~+~~~~~~~~+

 # | | |

 # +~~~~~~~~~~~~~+~~~~~~~~+

 #

 par(fig = c(0.60, 1.0, 0.55, 1.0), new=TRUE)

 if (length(error)>0)

 x<-error

 if (length(x)>0)

 {

 hist(x,freq=F,main="",ylim =c(0.0 , 1.0), ylab="" , axes=FALSE)

 title(main="Density plot of Pv-Pv(average)",line=-1,cex.main=0.75)

 axis(4)

 box()

 rug(x)

 mn <- mean(x)

 stdev <- sd(x)

 curve(dnorm(x, mean = mn, sd= stdev), add=TRUE, col="red", lty="dotted", xaxt="n")

 abline(v=mean(x),col="blue")

 abline(v=mn+2\*stdev,col="green", lty="dotted")

 abline(v=mn-2\*stdev,col="green", lty="dotted")

 mtext(paste("mean ", round(mean(x),2), "; sd ", round(sd(x),2), "; N ", length(x),sep=""), side=1, cex=.75)

 } # fi

 #

 # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

 # |Build\_Frequency - [3.3] plot error

 # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

 # +~~~~~~~~~~~~~+~~~~~~~~+

 # | | |

 # +~~~~~~~~~~~~~+~~~~~~~~+

 # | error | |

 # +~~~~~~~~~~~~~+~~~~~~~~+

 # | | |

 # +~~~~~~~~~~~~~+~~~~~~~~+

 #

 #

 par(fig = c(0.0, 0.7, 0.30, 0.75), new=TRUE)

 plot(error,type = "l", ylab="Pv-Average(Pv)",xlab="")

 abline(h=0, col="red",lty=4)

 abline(h=pv.sd, col="green",lty=3)

 abline(h=-pv.sd, col="green",lty=3)

 #

 # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

 # |Build\_Frequency - [3.4] histogram of error

 # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

 #

 # +~~~~~~~~~~~~~+~~~~~~~~+

 # | | |

 # +~~~~~~~~~~~~~+~~~~~~~~+

 # | | Hist(V)|

 # +~~~~~~~~~~~~~+~~~~~~~~+

 # | | |

 # +~~~~~~~~~~~~~+~~~~~~~~+

 #

 par(fig = c(0.60, 1.0, 0.30, 0.75), new=TRUE)

 h <- hist(error, breaks=16, plot=F)

 bplt <- barplot(h$counts, horiz=TRUE )

 title(main="Histogram of Pv-Pv(average)",line=-1,cex.main=0.75)

 text(x= h$counts+max(h$counts)\*0.06, y= bplt, labels=as.character(h$counts), xpd=TRUE,cex =0.5)

 #

 # multerror == vector of 1.0 and will be set to zero for SD = 0

 #

 multerror <- rep(1,nrows)

 for (j in 1:nrows)

 {

 begin1 <- j - nhalf

 begin2 <- j + nhalf

 if (begin1 < 0) {begin1 = 0}

 if (begin2 > nrows) {begin2 = nrows}

 tempSd <- sd(error[begin1:begin2])

 tempMean <- mean(error[begin1:begin2])

 if (tempSd == 0) {multerror[j]= 0.0}

 if (abs(tempMean)/tempSd > 100 ) { multerror[j]= 0.0}

 #cat(i,"Sd:", tempSd," Mean:", tempMean, "r:" , abs(tempMean)/tempSd, "multerror:", multerror[i])

 #cat("\n")

 } # end of for (j in 1:nrows)

 newerror = error \* multerror

 #

 # build auto-correlation

 #

 npoints <- nrows - n\_acv

 baseerror <- newerror[1:npoints]

 covvector <<- rep(0,n\_acv)

 for (j in 1:n\_acv)

 {

 iend <- j+npoints-1

 runvect <- newerror[j:iend]

 covvector[j] <- cor(runvect,baseerror)

 #cat(" j:", j," iend:", iend," n:", iend-j," nn:",npoints, " CV:", covvector[j])

 #cat("\n")

 } # end of for (j in 1:n\_acv)

 #

 # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

 # |Build\_Frequency - [3.4] plot cov-vector

 # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

 # +~~~~~~~~~~~~~+~~~~~~~~+

 # | | |

 # +~~~~~~~~~~~~~+~~~~~~~~+

 # | | |

 # +~~~~~~~~~~~~~+~~~~~~~~+

 # | cov-vector | |

 # +~~~~~~~~~~~~~+~~~~~~~~+

 par(fig = c(0.0, 0.7, 0.0, 0.45), new=TRUE)

 plot(covvector, type="l",ylab="Auto-Correlation of Error")

 #

 #

 #

 peakcc <- findpeaks(covvector,thresh=0)

 if (sum(is.na(peakcc)) == 0)

 {

 peakccm1 <- peakcc -1

 ## points(peakcc,covvector[peakccm1],col=c("red", "blue", "green"))

 for (k in 1:length(peakcc))

 {

 dx <- peakcc[k] - peakcc[k-1]

 #cat("k:", k, " peak:", peakcc[k], " CovVector:", covvector[peakcc[k]]," dx:", dx ,"\n")

 }

 }

 #

 # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

 # |Build\_Frequency - [3.5] plot fft cov-vector

 # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

 #

 # perform the FFT. In this case the number of points (N) will be equal to 1024.

 # Output will be the individual components of the FFT.

 fourierComponents<- fft(covvector,inverse = FALSE)

 # get the absolute value of the coefficients

 fourierCoefficients = abs(fourierComponents);

 # Normalize coefficients fig 5 here N = 1024 samples so N/2 = 512

 # normalizedFourierComponents = fourierCoefficients / (nhalf);

 normalizedFourierComponents = fourierCoefficients #/ (nhalf);

 fftvector<<-normalizedFourierComponents[1:nhalf]

 peakfft=findPeaks(fftvector)

 # +~~~~~~~~~~~~~+~~~~~~~~+

 # | | |

 # +~~~~~~~~~~~~~+~~~~~~~~+

 # | | |

 # +~~~~~~~~~~~~~+~~~~~~~~+

 # | | fft |

 # +~~~~~~~~~~~~~+~~~~~~~~+

 par(fig = c(0.6, 1.0, 0.00, 0.45), new=TRUE)

 slowness<-sum(covvector^2)

 logslowness<-log(slowness)

 ocilation<-sum(fftvector^2)

 logocilation<-log(ocilation/2)

 ctrlindex=1.5\*logocilation+logslowness

 fftxlab=paste("ctrl index 1.5\*o+s=", round(ctrlindex,2), " s=", round(logslowness,2), " o=", round(logocilation,2),sep="")

 cat(i,tagname,logslowness,logocilation, ctrlindex,"\n")

 ctrlsummarydata[nrow(ctrlsummarydata)+1,] <-c(tagname,

 logslowness,

 logocilation,

 ctrlindex)

 fftplt <-plot(fftvector,type='l',ylab="", axes=FALSE)

 axis(4)

 box()

 title(main="FFT of Covariance",line=-1,cex.main=0.75, xlab=fftxlab)

 numberofpeak = min(length(peakfft),4)

 if (length(peakfft) > 1)

 { #there is data

 peakfftm1 <- peakfft -1

 points(peakfft[1:numberofpeak],fftvector[peakfftm1[1:numberofpeak]],col=c("red", "blue", "green"))

 text(x=fftvector[peakfftm1[1:numberofpeak]]+max(fftvector[peakfftm1[1:numberofpeak]])\*0.06, y= fftplt, labels=as.character(peakfft), xpd=TRUE,cex =0.5)

 for (k in 1:numberofpeak)

 {

 freqsummarydata[nrow(freqsummarydata)+1,] <-c(tagname,peakfft[k],fftvector[peakfft[k]])

 #cat(i, firstcol, lastCol, tagname, peakfft[k], "\n")

 } # end of for

 } # end of if

 #

 #

 #

 #

 irow = i - firstcol +1

 freqdata[irow,]<-c(tagname,covvector)

 fftdata[irow,]<-c(tagname,fftvector)

 dev.off()

 cat(iloop,tagname, "\n" )

}

# end of for (i in firstCol:lastCol)

 write.csv (freqdata,freqfile,row.names=T)

 write.csv( fftdata,fftfile,row.names=T)

 write.table( freqsummarydata,freqsummaryfile,sep=",",row.names=F)

 write.table( ctrlsummarydata,ctrlsummaryfile,sep=",",row.names=F)

```

# Cluster Analysis

```{r 5.1 - BUILD CLUSTERS}

 #

 # [1] Initialize

 #

 clusterreport<- NULL

 clusterdata <- NULL

 #

 # [2] Load data

 #

 clusterdata <- as.data.frame( freqsummarydata)

 clusterdata$frq <-as.integer(clusterdata$frq)

 clusterdata$amp <-as.double(clusterdata$amp)

 str(clusterdata)

 #

 # [3] only extract AMP>10

 #

 df <- NULL

 df <-clusterdata[clusterdata$amp > 10,]

 #

 # [4] remove NA

 #

 df <- na.omit(df)

 #

 # select only max amp

 #

 df <-df %>%

 group\_by(tagname) %>%

 slice(which.max(amp))

 #

 # [5] Dissimilarity matrix

 #

 d <- dist(df, method = "euclidean")

 #

 # [6] Ward's method

 #

 hc <- hclust(d, method = "ward.D2" )

 #

 # [7] Plot

 #

 par(mfrow=c(1,1))

 jpeg(paste(proj\_folder,subfolder,"08-Cluster-Dendrogram.png",sep=""),width=11.0,height=8.0,units="in",res=1200)

 par(fig = c(0.0, 1.0, 0.0, 1.0))

 plot(hc, cex = 0.1, ylab = "Freq")

 rect.hclust(hc, k = 4, border = 2:5)

 dev.off()

```

```{r 5.1.1 - build all}

 #

 # [8] Cut tree into 4 groups

 #

 sub\_grp <- cutree(hc, k = 4)

 # Number of members in each cluster

 table(sub\_grp)

 split\_df <-split(df, sub\_grp)

 k = 0

 clusterreport <- NULL

 for (i in 1:4) {

 s1 <- NULL

 s1 <-data.frame(split\_df[[i]])

 cluster\_folder = paste(proj\_folder,subfolder,"\\cluster\_",i ,sep="")

 dir.create(file.path(cluster\_folder), showWarnings = FALSE)

 cat("cluster:",i, "folder :" , cluster\_folder," N row s:",nrow(s1), "\n")

 for (j in 1:nrow(s1) ){

 k = k + 1

 currentfile <-paste(proj\_folder,subfolder,s1[j,1],".png" ,sep="")

 newlocation <-paste(cluster\_folder,"\\",s1[j,1],".png" ,sep="")

 file.copy(from=currentfile, to=newlocation,

 overwrite = TRUE, recursive = FALSE,

 copy.mode = TRUE)

 #cat("k:",k," i:",i," j:",j," Tag:",s1[j,1],"\n")

 clusterreport$clusternumber[k] <- i

 clusterreport$tagname[k] <- s1[j,1]

 clusterreport$frq[k] <- s1[j,2]

 clusterreport$amp[k] <- s1[j,3]

 cat("k:",k," i:",i," j:",j," Tag:",clusterreport$tagname[k] ,"\n")

 }

 }

 write.table( clusterreport,clustersummaryfile,sep=",",row.names=F)

```

```{r 5 corraltion analysis}

 #

 # get tags

 selectedtags <- clusterreport$tagname[which(clusterreport$clusternumber==1)]

 # create an index

 index <- match(selectedtags,names(all\_data))

 # extract data

 cordata <- all\_data[,index]

 cormat <- correlate(cordata)

d2 <- cordata %>%

 as.matrix %>%

 cor %>%

 as.data.frame %>%

 rownames\_to\_column(var = 'var1') %>%

 gather(var2, value, -var1)

d3 <- filter(d2, (value >= .9 & value !=1 ))

 # create an index

 index\_9 <- match(d3$var1,names(all\_data))

 index\_9 <-na.omit(index\_9)

 cordata\_9 <- all\_data[,index\_9]

 #chart.Correlation(cordata\_9[,1:ncol(cordata\_9)], histogram=TRUE, pch=19)

```

```{r 6 - control index clusters}

 head(ctrlsummarydata)

 tail(ctrlsummarydata)

 ctrlsummarydata$ctrl.index <- as.numeric(ctrlsummarydata$ctrl.index)

 ci\_min <-min(ctrlsummarydata$ctrl.index)

 ci\_max <-max(ctrlsummarydata$ctrl.index)

 ci\_range <- ci\_max - ci\_min

 cat(ci\_min,ci\_max,ci\_range,"\n")

 ctrlsummarydata$order <- as.integer((as.numeric(ctrlsummarydata$ctrl.index) - ci\_min)\*4/ci\_range)+1

 k<-0

 for (i in 1:4) {

 folder\_text\_b <- format(round((ci\_range/4)\*(i-1)+ci\_min, 2), nsmall = 2)

 folder\_text\_e <- format(round((ci\_range/4)\*(i)+ci\_min, 2), nsmall = 2)

 s1 <- NULL

 s1 <-split\_df[[i]]

 cluster\_folder = paste(proj\_folder,subfolder,"\\CntlIndex\_Cluster\_",i,sep="")

 dir.create(file.path(cluster\_folder), showWarnings = FALSE)

 cat("CI\_cluster:",i, "folder :" , cluster\_folder, "\n")

 for (j in 1:nrow(ctrlsummarydata) ){

 if (ctrlsummarydata$order[j] == i) {

 k = k + 1

 sel\_tagname <- ctrlsummarydata$tagname[j]

 currentfile <-paste(proj\_folder,subfolder,sel\_tagname,".png" ,sep="")

 newlocation <-paste(cluster\_folder,"\\",sel\_tagname,".png" ,sep="")

 file.copy(from=currentfile, to=newlocation,

 overwrite = TRUE, recursive = FALSE,

 copy.mode = TRUE)

 cat("k:",k," i:",i," j:",j," Tag:",sel\_tagname,ctrlsummarydata$order[j],"\n")

 }

 }

 }

```

\newline

 N | Name | Comment

----|---------------|---------------------------

\newline

```{r 9.1- execution Time}

end\_time <- Sys.time()

cat(format(Sys.time(),usetz = TRUE))

cat(paste("Program Execution Time :", format(end\_time-start\_time) ,sep=""), sep="\n")

```

## References

The following are used as reference in this document:

\* Ref\_01

\* Ref\_02