```vb
'================================================================================
'
' Authors:        Asbjørn Hagset Amundsen,
'                 Kristian Andre Thomassen Vada
'
' Date:           14.04.2019
' Version:        1.3.0
'
' Description:    Import and processing of input data, calculate values,
'                 write relevant dataset for analyzis to data worksheet,
'                 write relevant results to worksheets,
'                 create trend charts.
'
' Changelog:      chart fix, added x-axis auto scaling
'                 Added error handling for no input data
' Todo:
'
'================================================================================

'public variables that must be accessible from other methods
Public arrInput As Variant '[date, A+, R+, A-, R-]
Public arrDelta() As Variant '[dt, kVAdt, kWdt], dt = decimal number representing 24-hour intervals
Public arrPloss() As Variant   '[kWdt, kWh] pLoss = p0 + pk * (kvaLoad / sn) ^ 2 / 1000
Public Tt As Variant    'time of use, Loss [hours per datarange]
Public Tb As Variant    'time of use, Load [hours per datarange]
Public noSamples As Long 'number of data samples
Public dt As Variant    'average sample time [hours]


Sub mainProgram() 'init main program when "calculate" button is pressed
'program runs by calling methods under step by step.

  'Selects the dynamic area range of input data,
  'writes data to arrInput array so it easely can be processed further.
  makeInputArr

  'calculates delta values by using arrInput array and stores result in arrDelta array.
  makeDeltaArr

  'calculate power loss for each interval by using arrDelta and stores result in arrPloss array.
  makePlossArr

  'calculates time of use, Loss [hours per datarange]
  Tt_calc

  'calculates time of use, Load [hours per datarange]
  Tb_calc

  'prints relevant results and info to worksheets
  printToSheet

End Sub


Function printToSheet()

  'declare local variables
  Dim ws As Worksheet
```

```
'write data to sheet "Result"

    'select sheet object
    Set ws = Sheets("Result")

    With ws
        'first and last date for samples
        .Cells(2, 2).Value = arrInput(LBound(arrInput, 1), 1)
        .Cells(3, 2).Value = arrInput(UBound(arrInput, 1), 1)

        'number of data samples
        .Cells(4, 2).Value = noSamples

        'sample time in minutes
        .Cells(5, 2).Value = dt * 60

        'time of use, loss
        .Cells(2, 6).Value = Tt
        'maximum recorded loss, dPmax
        .Cells(3, 6).Value = maxArrCol(arrPloss, 1)

        'time of use, load
        .Cells(4, 6).Value = Tb
        'maximum recorded load, dPbmax
        .Cells(5, 6).Value = maxArrCol(arrDelta, 2)

        'sum loss, Ploss kWdt
        .Cells(6, 6).Value = sumArrCol(arrPloss, 1)

        'sum loss, Ploss kWh
        .Cells(7, 6).Value = sumArrCol(arrPloss, 2)

        'degree of utilization, average value in pct
        .Cells(8, 6).Value = ((sumArrCol(arrDelta, 2) / (noSamples - 1))) / Sheets("Setup").Range("B3").Value * 100

        'degree of utilization, peak value in pct
        .Cells(9, 6).Value = (maxArrCol(arrDelta, 2)) / Sheets("Setup").Range("B3").Value * 100
    End With


'write data to sheet "Data"

    'select sheet object
    Set ws = Sheets("Data")

        'declare local variables
        Dim Destination As Range

        'call methods
        clearDS4 'clear old data in sheet

        'write date column from arrInput array to sheet
        Set Destination = ws.Range("A2")
        Destination.Resize(UBound(arrInput, 1) - 1, UBound(arrInput, 2)).Value = arrInput
        ws.Range("A:A").NumberFormat = "dd.mm.yyyy hh:mm"  'set cell format to make readable.

        'write arrDelta array to sheet
        Set Destination = ws.Range("B2")
    Destination.Resize(UBound(arrDelta, 1), UBound(arrDelta, 2)).Value = arrDelta
    ws.Range("B:B").NumberFormat = "hh:mm:ss" 'set cell format to make readable.

    'write arrPloss array to sheet
    Set Destination = ws.Range("E2")
    Destination.Resize(UBound(arrPloss, 1), UBound(arrPloss, 2)).Value = arrPloss

End Function


Function makeInputArr()
'reads input data to arrInput,
'[date, A+, R+, A-, R-]

  'declare local variables
  Dim startCell As Range, lastRow As Long, lastCol As Long, ws As Worksheet

  'set objects
  Set ws = Sheet1
  Set startCell = Range("A2")
  ws.Range("A:A").NumberFormat = "dd.mm.yyyy hh:mm" 'set cell format to make readable.

  'select last row and column
  lastRow = ws.Cells(Rows.Count, "A").End(xlUp).Row
  lastCol = ws.Cells(1, Columns.Count).End(xlToLeft).Column

  If lastRow > 2 Then 'if input data present
    'select dynamic range of data
    ws.Range(startCell, ws.Cells(lastRow, lastCol)).Select

    'copy selected data to array
    arrInput = Selection.Value

    noSamples = lastRow - 1 'number of samples = last row - first row

  Else ' If no input data show error message

    MsgBox "Error! Please add input data!"

  End If

End Function

Sub makeDeltaArr()
'calculates time, kVA, and kW between sample points,
'stores values in arrDelta array,
'[dt, kVAdt, kWdt], dt = 24-hour intervals

  'declare local variables
  Dim r, c, dts, kVAh, kWh As Variant, n As Long, ws As Worksheet

  'get setup variables from sheet
  Set ws = Sheets("Setup")
  n = ws.Range("B6").Value 'measurement transformer scaling factor

  'resize array to fit samples
  ReDim arrDelta(LBound(arrInput, 1) To (UBound(arrInput, 1) - 1), 1 To 3)

  'loop rows
```

```
    For r = LBound(arrInput, 1) To (UBound(arrInput, 1) - 1)

        'loop columns
        For c = LBound(arrInput, 2) To UBound(arrInput, 2) 'outer array = 1, inner array = 2 [[,]]

            'calc delta dates and add to array column 1
            If c = 1 Then
                arrDelta(r, c) = (arrInput(r + 1, c) - arrInput(r, c)) 'shows dt in date format, deciaml 24-hour intervals
                dts = arrDelta(r, 1) * 24 'shows dt in deciaml hours per sample

            'calc kVAdt and add to array column 2
            ElseIf c = 2 Then
                kVAh = ((Sqr((arrInput(r + 1, c) - arrInput(r, c)) ^ 2 + (arrInput(r + 1, c + 1) - arrInput(r, c + 1)) ^ 2)) *
n) 'throughput per hour
                arrDelta(r, c) = kVAh / dts 'throughput per delta time period

            'calc kWdt and add to array column 3
                kWh = (arrInput(r + 1, c) - arrInput(r, c)) * n 'throughput per hour
                arrDelta(r, c + 1) = kWh / dts 'throughput per delta time period

            End If

        Next
    Next

    'average sample time, value in hours
    dt = (sumArrCol(arrDelta, 1) * 24) / (noSamples - 1)

End Sub


Sub makePlossArr()
'calculates loss per delta time, and per hour,
'stores values in arrPloss,[kWdt, kWh]

    'declare local variables
    Dim r As Double

    'resize array to fit samples
    ReDim arrPloss(LBound(arrDelta, 1) To UBound(arrDelta, 1), 1 To 2)

    'loop rows
    For r = LBound(arrDelta, 1) To (UBound(arrDelta, 1))

        'calc loss and add to array.
        arrPloss(r, 1) = calcPloss(arrDelta(r, 2)) 'loss per delta time period dt
        arrPloss(r, 2) = calcPloss(arrDelta(r, 2)) * (dt / 1) 'loss per hour

    Next

End Sub


Function calcPloss(kvaLoad As Variant) As Double

    'declare local variables
    Dim Sn, p0, pk As Double, ws As Worksheet

    'get setup variables from sheet
    Set ws = Sheets("Setup")
    Sn = ws.Range("B3").Value
    p0 = ws.Range("B4").Value
    pk = ws.Range("B5").Value

    'calculate and return value
    calcPloss = (p0 + pk * (kvaLoad / Sn) ^ 2) / 1000 'returns kWdt

End Function


Function sumArrCol(arr As Variant, col As Long)
'summing column(col) in array(arr) and returns value

    With Application.WorksheetFunction

        sumArrCol = .Sum(.Index(arr, 0, col))

    End With

End Function


Function maxArrCol(arr As Variant, col As Long)
'finds max value from column(col) in array(arr) and returns value

    With Application.WorksheetFunction

        maxArrCol = .Max(.Index(arr, 0, col))

    End With

End Function


Function Tt_calc()
'time of use, Loss [hours per dataset]

    'calculate
    Tt = (sumArrCol(arrPloss, 1) * (1 / dt)) / maxArrCol(arrPloss, 1) '(SumLoss[kWdt]*(1hour/dthour))[kWh] / PeakLoss[kWdt]

End Function


Function Tb_calc()
'time of use, Load [hours per dataset]

    'calculate
    Tb = (sumArrCol(arrDelta, 3) * (1 / dt)) / maxArrCol(arrDelta, 3) '(SumLoad[kWdt]*(1hour/dthour))[kWh] / PeakLoad[kWdt]

End Function


Sub clearDS1()
'clear sheet data

    Set ws = Sheets("Input Data")
    ws.Range("A2", ws.Cells.End(xlDown).End(xlToRight)).ClearContents 'clears formulas, keeps formatting
End Sub


Sub clearDS4()
'clear sheet data

    Set ws = Sheets("Data")
    ws.Range("A2", ws.Cells.End(xlDown).End(xlToRight)).ClearContents 'clears formulas, keeps formatting
    ws.Range("G2", ws.Cells.End(xlDown).End(xlToRight)).ClearContents 'clears formulas, keeps formatting

End Sub


Sub drawCharts() 'runs program sequence when "Generate Charts" button is pressed
'runs by calling methods under step by step.

    'deletes all existing charts
    DeleteAllCharts

    'create histogram chart
    cHist

    'create kVA chart
    ckVAdt

    'create Loss chart
    cLoss

End Sub


Sub cHist()
'create histogram chart

    'declare local variables
    Dim dtc As Variant, lastRow As Long

    'clear old data area before generating new data
    lastRow = Sheets("Data").Range("G" & Rows.Count).End(xlUp).Row

    If lastRow > 1 Then 'clear old data if data present
    Sheets("Data").Range("G2:I" & lastRow).ClearContents
    End If

    'call method to make new histogram data
    makeHistData

    'read length of histogram data area
    lastRow = Sheets("Data").Range("G" & Rows.Count).End(xlUp).Row

    'generate chart
    Charts.Add2

    With ActiveChart 'set chart properties
        .Name = "c.Hist"
        .SetSourceData Source:=Sheets("Data").Range("G2:H" & lastRow)
        .FullSeriesCollection(1).ApplyDataLabels
        .HasTitle = True
        .ChartTitle.Text = "Histogram of Transformer Load"
        .Axes(xlCategory).HasTitle = True
        .Axes(xlCategory).AxisTitle.Caption = "Load kVA %"
        .Axes(xlValue).HasTitle = True
        .Axes(xlValue).AxisTitle.Caption = "Frequency"
        .Axes(xlCategory).TickLabels.Orientation = -45
        .Move After:=Sheets(Sheets.Count)
On Error GoTo jumpHere  'jumps if no legend object
        .Legend.Delete
jumpHere:
   End With

   Sheets("Data").Select 'return to sheet

End Sub


Sub ckVAdt()

    'declare local variables
    Dim dtc As Variant, lastRow As Integer, data As Range
    'get data range
    lastRow = Sheets("Data").Range("C" & Rows.Count).End(xlUp).Row
    Set data = Sheets("Data").Range("C2:C" & lastRow)

    'generate chart
    Charts.Add2

    With ActiveChart 'set chart properties
        .Name = "c.kVA"
        .ChartType = xlLine
        .SetSourceData Source:=Sheets("Data").Columns("C:C")
        .HasTitle = True
        .ChartTitle.Text = "Transformer Load"
        .Axes(xlCategory).HasTitle = True
        .Axes(xlCategory).AxisTitle.Caption = "Sample"
        .Axes(xlValue).HasTitle = True
        .Axes(xlValue).AxisTitle.Caption = "kVA"
        .Move After:=Sheets(Sheets.Count)
        .Axes(xlValue).TickLabels.NumberFormat = "0"
        On Error GoTo jumpHere  'jumps if no legend object
        .Legend.Delete
jumpHere:
   End With

   Sheets("Data").Select 'return to sheet

End Sub


Sub cLoss()

    'declare local variables
    Dim dtc As Variant, lastRow As Integer, data As Range
    'get data range
    lastRow = Sheets("Data").Range("E" & Rows.Count).End(xlUp).Row
    Set data = Sheets("Data").Range("E2:E" & lastRow)

    'generate chart
    Charts.Add2

    With ActiveChart 'set chart properties
        .Name = "c.Loss"
        .ChartType = xlLine
        .SetSourceData Source:=Sheets("Data").Columns("E:E")
        .HasTitle = True
        .ChartTitle.Text = "Transformer Loss"
        .Axes(xlCategory).HasTitle = True
        .Axes(xlCategory).AxisTitle.Caption = "Sample"
        .Axes(xlValue).HasTitle = True
        .Axes(xlValue).AxisTitle.Caption = "kW"
        .Move After:=Sheets(Sheets.Count)
        .Axes(xlValue).TickLabels.NumberFormat = "0,0"
        On Error GoTo jumpHere  'jumps if no legend object
        .Legend.Delete
jumpHere:
   End With

   Sheets("Data").Select 'return to sheet

End Sub


Sub makeHistData()

    'declare local variables
    Dim noBins, lastRow, n As Integer
    Dim arrData, arrBinsPct, arrBins, binSizePct As Variant
    Dim pctMaxLim, Sn As Double
    Dim ws As Worksheet

    'get setup variables from sheet
    pctMaxLim = Sheets("Setup").Range("G3").Value
    noBins = Sheets("Setup").Range("G4").Value
    Sn = Sheets("Setup").Range("B3").Value
    Set ws = Sheets("Data")

    'get data range
    lastRow = ws.Range("C" & Rows.Count).End(xlUp).Row
    Set arrData = ws.Range("C2:C" & lastRow)

    'size of each chart bar in pct
    binSizePct = (pctMaxLim / noBins)

    'make intervals into which you want to group the data values,
    'store intervals in arrBins array. bins refer to chart bars intervals
    'first cell in array
    ReDim arrBinsPct(1 To noBins, 1 To 1) 'scale array to fit data
    arrBinsPct(1, 1) = "0 - " & binSizePct & " %" 'make interval text string
    ReDim arrBins(1 To noBins, 1 To 1) 'scale array to fit data
    arrBins(1, 1) = (binSizePct / 100) * Sn 'make interval limits
    'the rest cells
    For n = 2 To UBound(arrBinsPct)
        arrBinsPct(n, 1) = CStr(Round((binSizePct * (n - 1)) + 1, 0)) & " - " & CStr(Round(binSizePct * n, 0)) & " %"
'make interval text string
        arrBins(n, 1) = binSizePct * n / 100 * Sn 'make interval limits
    Next
    'write arrBins array to data sheet
    Range("G2").Resize(UBound(arrBinsPct, 1)) = arrBinsPct
    Range("I2").Resize(UBound(arrBins, 1)) = arrBins

    'make histogram frequency data and write to data sheet
    freq = WorksheetFunction.Frequency(arrData, arrBins)
    Range("H2").Resize(UBound(freq, 1) - 1) = freq

End Sub


Sub DeleteAllCharts()
'delete all charts in ThisWorkbook

    Dim chrt

    Application.DisplayAlerts = False

    For Each chrt In ThisWorkbook.Charts
        chrt.Delete
    Next chrt

    Application.DisplayAlerts = True

End Sub
```