Radiosonde Calibration Example. The files needed are

RadioData.dat - radiosonde mixing ratio data
RH.dat - radiosonde RH data
TempK.dat - radiosonde Temperatures in K
RadioHeights.dat - radiosonde heights
CompositeData.dat - uncalibrated lidar mixing ratio
SumErrorProfTest.dat - lidar random errors
LidarHeights.dat - lidar heights.

Submitted as supplemental data for "Correction technique for raman water vapor lidar aignal dependent bias and suitability for water vapor trend monitoring in the upper troposphere".

```
In[1]:= Needs["PlotLegends`"];
```

Mathematica routines needed for the example to function. GetRadioCalFactorRegress is the last routine defined in this section.

```
In[2]:= GetLastPositionLess[InputList_, Value_] := Module[{i},
          i = 1;
          While[
              InputList[[i]] < Value, If[i == Length[InputList], i++; Break[]]; i++];
              i - 1]
```

```
In[3]:= GetFirstPositionGreater[InputList_, Value_] := Module[{i},
          i = 1;
          While[
              InputList[[i]] ≤ Value, If[i == Length[InputList], Break[]]; i++;];
              i]
```

```
In[4]:= RunningSum[InputList_, PointsToSum_] := Module[{Nmax, i, j, OutList = {}},
          Nmax = IntegerPart[Length[InputList] / PointsToSum];
          i = 1; j = PointsToSum;
          Do[
              OutList = {OutList, Plus @@ Take[InputList, {i, j}]};
              i += PointsToSum; j += PointsToSum;,
              {Nmax}];
          If[Length[InputList[[1]]] > 1,
      Partition[Flatten[OutList], Length[InputList[[1]]]], Flatten[OutList]]]
```

```
In[5]:= MyMedian[list1_] := Block[{s, n}, s = Sort[list1];

      n = Length[list1]; 1/2 (s〚n/2〛 + s〚n/2 + 1〛)] /; EvenQ[Length[list1]]
```

Least Median of Squares. From Shaw - Tigg.

```
In[6]:= lmsfita[data_, x_] := Module[{l, v, i, j, w, h, a, sl, dx}, l = Length[data];
          v = Flatten[Table[{data⟦i⟧, data⟦j⟧}, {i, 1, l - 1}, {j, i + 1, l}], 1];
          w = Union[(Module[{dx}, dx = #1⟦1, 1⟧ - #1⟦2, 1⟧; If[dx ≠ 0,
                      #1⟦1, 2⟧ - #1⟦2, 2⟧    #1⟦1, 1⟧ #1⟦2, 2⟧ - #1⟦1, 2⟧ #1⟦2, 1⟧
                     {-------------------, -----------------------------------------}, {∞, #1⟦1, 1⟧}]] &) /@ v];
                              dx                              dx
          h = Function[a, {MyMedian[(If[a⟦1⟧ ≠ ∞, (#1⟦2⟧ - a⟦1⟧ #1⟦1⟧ - a⟦2⟧)^2, (a⟦2⟧ - #1⟦1⟧)^2] &) /@
                 data], a⟦1⟧, a⟦2⟧}];
          sl = Sort[h /@ w]⟦1⟧; If[sl⟦2⟧ ≠ ∞, sl⟦2⟧ x + sl⟦3⟧, Print["x = ", sl⟦3⟧]]]
```

Adaptive Radiosonde Caibration Routine. The algorithmn works as follows:

1) interpolate data to height grid approximating the sonde vertical resolution
2) Select initial population of points based on RSquare of linear least squares regressions of subsets of data
3) Delete duplicate points
4) presum interpolated data to approximate lidar vertical resolution
5) Perform Least Median of Squares regression
6) Remove outliers based on percentage devation from PredictedResponse
7) If sufficient number of points, Use these points to calculate the final calibration value by taking the Mean ratio of the points. Also perform linear regression on final points.
8) If not sufficient points, decrease Rsquare and AcceptancePercentage values and repeat

```
In[7]:= GetRadioCalFactorRegress[RadioMRData_, RadioRHData_,
         RadioTData_, RadioHeights_, LidarData_, LidarDataErrs_, LidarHeights_,
         MinCalAlt_, MaxCalAlt_, MinRSquareIni_, RegressionIntervalKM_,
         AcceptancePercentageIni_, MinNumberRegressionPoints_, MinRH_, MinTempC_] :=
        Module[{MinRadioBin, MaxRadioBin, MinLidarBin, MaxLidarBin, SubsetRadioHeights,
          SubsetRadioMRData, SubsetRadioRHData, SubsetRadioTData, SubsetLidarHeights,
          SubsetLidarData, MeanDeltaH, HeightsForInterp, HeightsForInterpNew, HeightsForInterpNew2,
          DataR, DataL, RadioMRFunc, RadioRHFunc, RadioTFunc, LidarFunc, InterpRadioMRData,
          InterpRadioRHData, InterpRadioTData, InterpLidarData, LidarResolution,
          MeanSondeResolution, PresumNumber, SondeMRDataNew, SondeRHDataNew, SondeTDataNew,
          SondeDataNew2, LidarDataNew, LidarDataNew2, SelectedData, RegressionData,
          ThisRegressionData, FinalResolution, NumToRegress, NumberOfRegressions, ThisRegression,
          ThisRSquare, PointsToSave, SinglePointsToSave, PointsToRegress, linfit, lmslinfit,
          PredictedLMSResponse, LMSResiduals, LMSRatios, GoodPoints1, FinalRegressionPoints,
          InitialLinFit, FinalLinFit, InitialDataPoints, FinalDataPoints, InitialCalVal,
          FinalCalVal, ThisMinRSquare, ThisAcceptancePercentage, GoodPoints, lm, x, i},

         MinRadioBin = GetFirstPositionGreater[RadioHeights, MinCalAlt];
         MaxRadioBin = GetLastPositionLess[RadioHeights, MaxCalAlt];

         MinLidarBin = GetFirstPositionGreater[LidarHeights, MinCalAlt];
         MaxLidarBin = GetLastPositionLess[LidarHeights, MaxCalAlt];

         SubsetRadioHeights = Take[RadioHeights, {MinRadioBin - 1, MaxRadioBin + 1}];
         SubsetRadioMRData = Take[RadioMRData, {MinRadioBin - 1, MaxRadioBin + 1}];
```

```
SubsetRadioRHData = Take[RadioRHData, {MinRadioBin - 1, MaxRadioBin + 1}];
SubsetRadioTData = Take[RadioTData, {MinRadioBin - 1, MaxRadioBin + 1}];

SubsetLidarHeights = Take[LidarHeights, {MinLidarBin - 1, MaxLidarBin + 1}];
SubsetLidarData = Take[LidarData, {MinLidarBin - 1, MaxLidarBin + 1}];

RadioMRFunc =
 Interpolation[Thread[{SubsetRadioHeights, SubsetRadioMRData}], InterpolationOrder -> 1];
RadioRHFunc = Interpolation[Thread[{SubsetRadioHeights, SubsetRadioRHData}],
 InterpolationOrder -> 1];
RadioTFunc = Interpolation[Thread[{SubsetRadioHeights, SubsetRadioTData}],
 InterpolationOrder -> 1];

DataL = Transpose[{SubsetLidarHeights, SubsetLidarData}];
LidarFunc = Interpolation[DataL, InterpolationOrder -> 1];

(* determine which dataset has higher vertical resolution. Develop height grid using
mean resolution of that dataset and put all data on the same height grid *)

LidarResolution = (LidarHeights[[2]] - LidarHeights[[1]]);
MeanSondeResolution =
 (SubsetRadioHeights[[-1]] - SubsetRadioHeights[[1]]) / Length[SubsetRadioHeights];
MeanDeltaH = Min[LidarResolution, MeanSondeResolution];
HeightsForInterp = Table[SubsetRadioHeights[[2]] + (i - 1) MeanDeltaH,
 {i, 1, IntegerPart[(MaxCalAlt - MinCalAlt) / MeanDeltaH] - 1}];


InterpRadioMRData = RadioMRFunc[HeightsForInterp];
InterpRadioRHData = RadioRHFunc[HeightsForInterp];
InterpRadioTData = RadioTFunc[HeightsForInterp];
InterpLidarData = LidarFunc[HeightsForInterp];

(* presum data to approximate the resolution of the lidar data *)

PresumNumber = Max[IntegerPart[LidarResolution / MeanDeltaH ], 1];
HeightsForInterpNew = RunningSum[HeightsForInterp, PresumNumber] / PresumNumber;
SondeMRDataNew = RunningSum[InterpRadioMRData, PresumNumber] / PresumNumber;
SondeRHDataNew = RunningSum[InterpRadioRHData, PresumNumber] / PresumNumber;
SondeTDataNew = RunningSum[InterpRadioTData, PresumNumber] / PresumNumber;
LidarDataNew = RunningSum[InterpLidarData, PresumNumber] / PresumNumber;

(* select the data according to RH and T thresholds if desired *)

SelectedData = Select[Thread[{SondeMRDataNew, SondeRHDataNew, SondeTDataNew,
 LidarDataNew, HeightsForInterpNew}], #[[2]] > MinRH && #[[3]] > MinTempC &];
LidarDataNew2 = Transpose[SelectedData][[4]];
SondeDataNew2 = Transpose[SelectedData][[1]];
HeightsForInterpNew2 = Transpose[SelectedData][[5]];

(* initial linear fit and mean cal constant *)

InitialLinFit =
LinearModelFit[Thread[{LidarDataNew2, SondeDataNew2}], x, x]["BestFitParameters"];
InitialCalVal = Mean[SondeDataNew2 / LidarDataNew2];
InitialDataPoints = {LidarDataNew2, SondeDataNew2, HeightsForInterpNew2};

(* Determine how many points in each regression and how many total regressions *)

FinalResolution = HeightsForInterpNew[[2]] - HeightsForInterpNew[[1]];
NumToRegress = IntegerPart[RegressionIntervalKM / FinalResolution];
(* number of points to use in each regression *)
NumberOfRegressions = Length[LidarDataNew2] - NumToRegress + 1;
(* total number of regressions in the While loop *)
(* Print[NumToRegress," ",NumberOfRegressions];*)

(* select points within user defined region based on R-
square. Retain a point if it is in any regression interval that
qualifies. Maintain height at which points occur. Eliminate duplicate points. *)

RegressionData = Thread[{LidarDataNew2, SondeDataNew2, HeightsForInterpNew2}];
PointsToSave = {};

(* the outer while loop is to adjust the values of MinRSquare and
AcceptancePercentage if insufficient number of good points are found *)

ThisMinRSquare = MinRSquareIni;
ThisAcceptancePercentage = AcceptancePercentageIni;
GoodPoints = {};
While[Length[GoodPoints] ≤ MinNumberRegressionPoints,

(* the inner while loop selects by Rsquare and filters data using LMS regression *)

i = 1;
While[i ≤ NumberOfRegressions,
ThisRegressionData = Take[RegressionData, {i, i + NumToRegress - 1}];
PointsToRegress = Transpose[Drop[Transpose[ThisRegressionData], -1]];
 (* the Transpose[Drop[Transpose operation selects just the ordered pairs *)
 lm = LinearModelFit[PointsToRegress, x, x];
 ThisRSquare = lm["RSquared"];
 If[ThisRSquare ≥ ThisMinRSquare, PointsToSave = Append[PointsToSave, ThisRegressionData]];
 (* the heights data are included here *)
 i++;
 ];
SinglePointsToSave = DeleteDuplicates[Flatten[PointsToSave, 1]];
 (* the heights data are included here *)

(* Use Least median of squares on this population of points. Use
AcceptancePercentage to eliminate outliers from the LMS fit fit *)
If[Length[SinglePointsToSave] ≥ MinNumberRegressionPoints,

PointsToRegress = Transpose[Drop[Transpose[SinglePointsToSave], -1]];
{linfit, lmslinfit} = {Fit[PointsToRegress, {1, x}, x], lmsfita[PointsToRegress, x]};

PredictedLMSResponse = lmslinfit /. x → Transpose[PointsToRegress][[1]];
LMSRatios = Transpose[PointsToRegress][[2]] / PredictedLMSResponse;

GoodPoints = Select[Thread[{LMSRatios, SinglePointsToSave}],
 (#[[1]] > 1 - ThisAcceptancePercentage && #[[1]] < 1 + ThisAcceptancePercentage ) &];
];

ThisMinRSquare = ThisMinRSquare - 0.0125;
ThisAcceptancePercentage = ThisAcceptancePercentage + 0.005;

];

FinalRegressionPoints = Transpose[GoodPoints][[2]];
PointsToRegress = Transpose[Drop[Transpose[FinalRegressionPoints], -1]];
FinalLinFit = LinearModelFit[PointsToRegress, x, x]["BestFitParameters"];
FinalCalVal = Mean[Transpose[PointsToRegress][[2]] / Transpose[PointsToRegress][[1]]];
FinalDataPoints = Transpose[FinalRegressionPoints];

 {InitialCalVal, InitialDataPoints, InitialLinFit, FinalCalVal, FinalDataPoints,
 FinalLinFit, lmslinfit, ThisMinRSquare + 0.0125, ThisAcceptancePercentage - 0.005}
 ]
```

Set the directory to where the input data files are located and read the data.

```
In[8]:= SetDirectory["C:\\Data\\Processed\\UWO_2012\\ALVICE\\CalExample\\"];
```

```
In[9]:= RadioData = << "RadioData.dat";
        RH = << "RH.dat";
        TempK = << "TempK.dat";
        RadioHeights = << "RadioHeights.dat";
        CompositeData = << "CompositeData.dat";
        CompositeError = << "SumErrorProfTest.dat";
        LidarHeights = << "LidarHeights.dat";
```

Execute the adaptive routine for the case of a single radiosonde - lidar comparison.

```
In[16]:= {InitialCalVal, InitialDataPoints, InitialLinFit, FinalCalVal, FinalDataPoints,
             FinalLinFit, lmslinfit, MinRSquareFinal, AcceptancePercentageFinal} =
           GetRadioCalFactorRegress[RadioData, RH, TempK - 273.15, RadioHeights,
             LidarData = CompositeData, LidarDataErrs = CompositeError, LidarHeights,
             MinCalAlt = 2.275, MaxCalAlt = 5.275, MinRSquare = 0.95, RegressionIntervalKM = 0.5,
             AcceptancePercentage = 0.03, MinRegressPts = 30, MinRH = 5, MinTempC = -50];
```

Package the data and plot.

```
SondeProfilePoints = Thread[{InitialDataPoints[[2]], InitialDataPoints[[3]]}];
LidarProfilePoints = Thread[{FinalCalVal InitialDataPoints[[1]], InitialDataPoints[[3]]}];

p1r = ListPlot[{SondeProfilePoints, LidarProfilePoints}, Joined → False,
   Axes → False, Frame → True, FrameLabel → {"Mixing Ratio (g/kg)", "Altitude (km)"},
   PlotRange → {{0.9 Min[FinalCalVal FinalDataPoints[[1]]],
      1.2 Max[FinalCalVal FinalDataPoints[[1]]]}, {MinCalAlt, MaxCalAlt}},
   PlotStyle → {Directive[PointSize[Medium], Black], Directive[PointSize[Medium], Red]},
   PlotMarkers → Automatic, PlotLegend → {"Sonde", "Lidar"}, LegendPosition → {0.3, 0.3},
   LegendTextSpace → 10, LegendSpacing → 1, LegendSize → {0.3, 0.2},
   LegendShadow → False, BaseStyle → {FontSize → 14, FontFamily → "Helvetica"}];

p1ra = Show[p1r, Epilog →
    Inset[Framed[Style["Before selection", 12], Background → LightYellow], {-0.3, 0.45}]];
SondeFinalProfilePoints = Thread[{FinalDataPoints[[2]], FinalDataPoints[[3]]}];
LidarFinalProfilePoints = Thread[{FinalCalVal FinalDataPoints[[1]], FinalDataPoints[[3]]}];
```

```
In[23]:= p2r = ListPlot[{SondeFinalProfilePoints, LidarFinalProfilePoints}, Joined → False,
   Axes → False, Frame → True, FrameLabel → {"Mixing Ratio (g/kg)", "Altitude (km)"},
   PlotRange → {{0.9 Min[FinalCalVal FinalDataPoints[[1]]],
      1.2 Max[FinalCalVal FinalDataPoints[[1]]]}, {MinCalAlt, MaxCalAlt}},
   PlotStyle → {Directive[PointSize[Medium], Black], Directive[PointSize[Medium], Red]},
   PlotMarkers → Automatic, PlotLegend → {"Sonde", "Lidar"}, LegendPosition → {0.3, 0.3},
   LegendTextSpace → 10, LegendSpacing → 1, LegendSize → {0.3, 0.2},
   LegendShadow → False, BaseStyle → {FontSize → 14, FontFamily → "Helvetica"}];
```

```
In[24]:= p2ra = Show[p2r, Epilog →
    Inset[Framed[Style["After selection", 12], Background → LightYellow], {-0.3, 0.45}]];
InitialRegressionPairs = Thread[{InitialDataPoints[[1]], InitialDataPoints[[2]]}];
FinalRegressionPairs = Thread[{FinalDataPoints[[1]], FinalDataPoints[[2]]}];
```

```
In[27]:= p4r = Plot[{InitialLinFit[[1]] + InitialLinFit[[2]] x, FinalLinFit[[1]] + FinalLinFit[[2]] x},
   {x, 0, 1.1 Max[InitialDataPoints[[1]]]},
   PlotRange → {{0, 1.1 Max[InitialDataPoints[[1]]]}, {0, 1.1 Max[InitialDataPoints[[2]]]}},
   Axes → False, Frame → True, FrameLabel → {"Lidar (uncal)", "Sonde (g/kg)"},
   PlotLegend → {"Initial Regress", "Final Regress"}, LegendPosition → {0.4, -0.2},
   LegendTextSpace → 10, LegendSpacing → 0.2, LegendSize → {0.4, 0.15},
   LegendShadow → False, BaseStyle → {FontSize → 14, FontFamily → "Helvetica"},
   Evaluate[Epilog → {Black, PointSize[0.02], Point /@ InitialRegressionPairs,
      Red, PointSize[0.02], Point /@ FinalRegressionPairs,
      Inset[Framed[Style["Calini = " <> ToString[InitialCalVal] <> "\nCalfin= " <> ToString[
            FinalCalVal] <> "\nMinRsq = " <> ToString[MinRSquareFinal] <> "\nAcc%= " <>
          ToString[AcceptancePercentageFinal], 12], Background → LightYellow],
       {0.2 Max[InitialDataPoints[[1]]], 0.9 Max[InitialDataPoints[[2]]]}]}]];
```

Export the graphic to see the results.

```
In[28]:= Export["RadioCalAnalysisTest.gif", GraphicsGrid[{{p1ra, p2ra}, {p4r}},
   Spacings → {-50, 10}], ImageSize → 1000, ImageResolution → 100];
```