Automated Calibration of a Five-color High-speed Radiometer

by Alejandro Perdomo and Willard Casey Uhlig
NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer’s or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.
Automated Calibration of a Five-color High-speed Radiometer

Alejandro Perdomo and Willard Casey Uhlig
Weapons and Materials Research Directorate, ARL
Radiation pyrometry or radiometry correlates the radiation emitted by a body to its temperature using Planck’s radiation law. Radiometers are ideal for high-speed events in which direct optical observation can be obtained. Multichannel radiometers eliminate the requirement of a measured emissivity and allow the attainment of measurement uncertainty as well as temperature. However, calibration of such systems is crucial, and because large amounts of data are required, calibration can be a time-consuming process. Here we present a LabTalk script that automatically calibrates the data and determines certain values such as the frequency and amplitudes. The calibration process includes finding the ratios of light emitted at different wavelengths and comparing the natural log of those ratios to the inverse of the temperatures. The script is for the program OriginLab, a data analysis and graphing software that created LabTalk specifically for OriginLab, and is similar to the programming language C.
## Contents

List of Figures ................................................................. iv

Acknowledgments ............................................................. v

1. Introduction ................................................................. 1
   1.1 Five-color Radiometry .................................................. 1
   1.2 Blackbody Theory ....................................................... 2
   1.3 Calibration Needed .................................................... 3

2. Calibration ................................................................. 3
   2.1 Smoothing Data .......................................................... 3
   2.2 Finding Needed Values ................................................ 5
   2.3 Extracting Positive Data ............................................. 6
   2.4 Extracting Negative Data ............................................ 7
   2.5 Calculating Ratios ..................................................... 8

3. Conclusion ................................................................. 9

4. References ................................................................. 10

Appendix. Calibration Program .......................................... 11

Distribution List ............................................................ 16
List of Figures

Figure 1. Block diagram of the five-channel radiometer assembly. ...............................................2
Figure 2. Diagram of a blackbody source. ......................................................................................3
Figure 3. Radiometer output voltage vs. time at five different wavelengths. .................................4
Figure 4. Smoothed radiometer output voltage vs. time. ................................................................5
Figure 5. Graphical description showing that the difference of x-intercept 3 and 1 is also the distance between peaks. .................................................................6
Figure 6. The positive data from channel 5 of the radiometer (a) and the extracted data (b) .......7
Figure 7. Extracted negative data for all wavelengths. ...................................................................8
Figure 8. The part of the script that generates the ten ratios............................................................9
Acknowledgments

I would like to acknowledge the mentorship of Dr. Casey Uhlig and sincerely thank him for all he has taught, the encouragement he has given, and the time he spent helping me.

I would also like to thank everyone involved in handling the Science and Engineering Apprenticeship Program (SEAP) for making this experience possible.
INTENTIONALLY LEFT BLANK.
1. Introduction

A shaped charge is an explosive charge shaped to focus the effect of the explosive’s energy (1). Shaped charges can form explosively formed penetrators or shaped-charge jets, depending on the liner shape. These hypervelocity projectiles use kinetic energy to penetrate armor (1). The temperatures of these projectiles need to be measured to verify shock physics codes like the Arbitrary Lagrangian Eulerian General Research Application (ALEGRA) and make sure simulations accurately predict the temperature of these projectiles. Special methods must be used to obtain the temperatures of high-speed objects like shaped-charge jets. One such method is five-color radiometry.

1.1 Five-color Radiometry

Five-color radiometry measures the light emitted by an object at five different wavelengths and uses ratios of the measured intensities and blackbody theory to determine the temperature of the object emitting the light (2). The light from the source or object passes through the lens into a beamsplitter array that splits the beam into six equal beams. One beam of light goes to a video camera, the rest continue to bandpass filters. Each bandpass filter only lets in one specific wavelength of light, allowing the radiometer to measure different wavelengths independently. Next, the light passes through photodetectors that gather the signal, and then the signal travels through amplifiers to boost the signal, which is output to a data collection device. A diagram of the radiometer is shown in figure 1. Emitted light intensities of multiple wavelengths are measured by the radiometer to help eliminate error by comparing the different measurements. Emissivity of an object is the intensity of light emitted by an object divided by the intensity of light emitted by a blackbody (2). Using the ratio of the emitted light intensities of two wavelengths removes the need for measuring the emissivity of the object. This assumes that the emissivity at the two wavelengths is approximately equal (2).
1.2 Blackbody Theory

A blackbody is a theoretical object that absorbs and radiates energy perfectly (3). A blackbody source is like a small furnace with an opening in it to observe a hollow space that is heated to a chosen temperature (4). A blackbody source is shown in figure 2. The blackbody source is set to a series of known temperatures and its emitted intensity is recorded by a radiometer. The calibration process includes finding the ratios of light intensity at different wavelengths. Since the natural log of the ratios is proportional to the inverse of the temperature (5), they provide a data reference to calculate the temperature from data collected on objects in an experiment.
1.3 Calibration Needed

The recorded output voltage of the radiometer must be calibrated to obtain the ratios needed to estimate the temperature of objects. The calibration process is lengthy due to the large amount of data recorded. Therefore, a script was created that automatically calibrates the radiometer data at the click of a button. This script was written in LabTalk, a programming language similar to C, made specifically for the data analysis and graphing software OriginLab. The light from the blackbody source is sent through an optical chopper to get an AC signal, the amplitude of this AC signal is the light intensity that must be measured. To get the amplitudes, the peaks of the output voltage on each channel (or wavelength) need to be extracted; most of the script is devoted to this.

2. Calibration

2.1 Smoothing Data

Using the script described below, the raw data are first smoothed by an adjacent averaging method. The adjacent averaging moves down the rows of data and creates a moving average of 50 points. An example of the smoothing process is shown in figures 3 and 4. After the data are smoothed, the program looks for the x-intercepts of the data by searching for a point that is positive and the following point is negative, or when a point is negative and the next point is
positive. Then the x value and row number of each x-intercept is recorded into a dataset. The data must be smoothed because the raw data have a significant amount of noise. If not smoothed, the program may find more x-intercepts than actually exist. The smoothed data are only used to acquire the x-intercepts; they are then discarded and the remainder of the work is performed on the raw data.

Figure 3. Radiometer output voltage vs. time at five different wavelengths.
2.2 Finding Needed Values

The script allows the user to choose the number of points to be extracted from each peak for the final analysis via a user dialogue box. Thus, the program can extract a small amount of data if there are not many points or if the frequency is high, or larger amounts based on the conditions of the data acquisition. The x-intercept data are used to find several values that are needed for the data extraction formula. The distance between each peak is calculated by subtracting the number of points to the third x-intercept by the number of points to the first x-intercept. This yields the number of points per wavelength of the optical chopper or, in other words, between any spot on a peak and the corresponding spot on the peak next to it, as shown in figure 5. The center of the peak or trough is found by the midpoints of the x-intercepts. Half of the data points requested by the user are extracted before the center of the peak/trough and half after. This is carried out for each peak and trough.
2.3 Extracting Positive Data

To extract data from the positive peaks of one of the channels, we use the following expression:

```c
for (m=1; m<NumOfRow; m++)
{
    if((rmod(m-(start1)), peakdistance)<user) &&
        (rmod(m-(start1)), peakdistance)>0))
    {
        s[xyz] = a[m]; //puts data into dataset
        b[xyz] = b[m];
        c[xyz] = c[m];
        d[xyz] = d[m];
        e[xyz] = e[m];
        f[xyz] = f[m];
        xyz++;
    }
}
```

In the above code, `rmod` finds the remainder, `m` is the current row, `start1` is the first row at which extraction must start for the positive data, `peakdistance` is the number of points between each peak, and “user” is the number of points the user requests via the dialog box. This expression loops for every row of data, and if true, copies the data into a dataset. The dataset is then copied into a worksheet. An example of the positive data extraction process is shown in figures 6a and b. The mean of all the extracted positive data in the worksheet is then taken, giving the positive peak amplitude of the wave.
2.4 Extracting Negative Data

The same steps used to extract the crests are used to extract the points from the troughs, using this expression:

```c
for(s=1;s<NumOfRow;s++)
{
    if((rmod((s-(start2))),(peakdistance))<usec) &
        (rmod((s-(start2)),(peakdistance))>=0))
    {
        aaaa2[xy] = aa[s];
        bbbb2[xy] = bb[s];
        cccc2[xy] = cc[s];
        dddd2[xy] = dd[s];
        eeee2[xy] = ee[s];
        ffff2[xy] = ff[s];
        xy++;
    }
}
```

Here, rmod finds the remainder, s is the current row, and start2 is the point at which extraction must start for the negative data. The mean of all the extracted negative data gives the average of the troughs. A graph of the extracted negative data for all wavelengths is shown in figure 7. The mean of the extracted troughs is the negative peak amplitude of the wave.
2.5 Calculating Ratios

The absolute values of the two peak amplitudes are added, giving the average distance from crest to trough or the peak-to-peak amplitude of the light intensity for a given wavelength (determined by the notch filter in the radiometer on that channel). Then, the extraction process is repeated for each channel until all the amplitudes are calculated. Ten possible ratios are produced from the peak-to-peak amplitudes of each of the wavelengths. The part of the script that calculates ratios is shown in figure 8. The entire process of smoothing data, finding x-intercepts, extracting data, and generating ratios is repeated for each worksheet in the OriginLab window. (Each worksheet contains the data for a different temperature set point from the blackbody source.) This allows the program to acquire the 10 ratios for multiple temperatures on a single run. The natural log of each ratio is then automatically filled into a worksheet with the name of the worksheet they came from. This natural log can then be plotted as a function of the inverse of the temperature, and a linear fit to that data yields the proportionality constants required to complete calibration. Now that calibration is complete, an arbitrary object can be observed with the radiometer and acquire 10 simultaneous surface temperature measurements.
3. Conclusion

Five-color radiometry can be used to find the temperature of high-speed objects for verification of shock physics codes and simulations. However, high-speed radiometry methods used to find temperature require a vast amount of data processing to complete. A script has been successfully developed that automatically calibrates the radiometer data and produces several useful values for finding temperature. This saves significant time with calibrations so that more time can be spent interpreting data instead of preparing it.
4. References


Appendix. Calibration Program

Attached here is the Calibration program in its entirety.

Alejandro Perdomo 7/18/12 Program finds frequency, means, amplitudes, and ratios.

define getfreq
{
%W = %H;
//records name of worksheet into %W//
win = r %H thedata;
%V = %H;
//makes active window name equal %V//
%K = layer.name$;
//records page name//
range ww = [%V]%K!;
set %V -er ww.maxrows;
//deletes empty rows//
ww.AutoAddRows=0;
range FaF = [%V]%K!Col(F);
int NumOfRows = ww.nRows;
int nn = numofrows;
for(p=1;p<numofrows;p++)
 //loops for every row//
{
if(FaF[1]<0 && FaF[nn]<0)
  //if the first and last number in column F is negative//
  
  nn=nn-30;
  set %V -er nn; //deletes last 30 rows//
}
if(FaF[1]>0 && FaF[nn]>0)
  //if the first and last number in column F is positive//
  
  nn=nn-30;
  set %V -er nn; //deletes last 30 rows//
}
smooth iy:=Col(B) method:=aav npts:=50 oy:=Col(Smooth1);
  //smooths data with adjacent averaging by 50 pts//
smooth iy:=Col(C) method:=aav npts:=50 oy:=Col(Smooth2);
smooth iy:=Col(D) method:=aav npts:=50 oy:=Col(Smooth3);
smooth iy:=Col(E) method:=aav npts:=50 oy:=Col(Smooth4);
smooth iy:=Col(F) method:=aav npts:=50 oy:=Col(Smooth5);
Range KK = [%V]%K!Col(K);
int numofrow = ww.nrows;
Dataset TT;
Dataset RR;
int x = 0;
for(n=1;n<NumOfRow;n++)
  //loops for every row//
  
  if(KK[n]>=0 && KK[n+1]<0)
    //finds x intercept//
    
    x++;
    //adds to x intercept counter//
    TT[x] = col(a)[n]; //puts time of x intercept into dataset TT//
    RR[x] = n; //puts row of x intercept into dataset RR//
}
if(KK[n]<0 && KK[n+1]>0)
  //finds x intercept//
x++;  //adds to x intercept counter/

TT[x] = col(a)[n];  //puts time of x intercept into dataset TT/
RR[x] = n;  //puts row of x intercept into dataset RR/

}  

}  

del col(g);  //deletes smooth columns/
del col(h);
del col(i);
del col(j);
del col(k);
int c = 0;  //declares a couple of variables/
double T1;
double T2;
double DeltaT;
double Frequency;
c = x/2;  //cycles = number of x intercepts/2/
T1 = TT[1];  //time 1 = time of first x intercept/
T2 = TT[x];  //time 2 = time of last x intercept/
DeltaT = T2-T1;  //DeltaT = Time2-Time1/

Frequency = c/DeltaT;  //Frequency = cycles/DeltaT/

}  

define extract1  
{
int user;
getnumber (# of points wanted) user
(Enter the amount of points wanted from each peak);
int user1 = user/2;  //gets number of points through a user dialog box/
double kit = RR[1];  //kit = row of first x intercept/
double kat = RR[2];  //kat = row of second x intercept/
int kitkat = (kat-kit)/2;  //kitkat = amount of rows from kit to center of peak/
double center1 = kit + kitkat;  //center1 = row of center of peak/
double start1 = center1-user1;  //start1 = where you want to begin extracting from each peak/
double tic = RR[2];  //tic = row of second x intercept/
double tac = RR[3];  //tac = row of third x intercept/
int tictac = (tac-tic)/2;  //tictac = amount of rows from tic to center of peak/
double center2 = tic + tictac;  //center2 = row of center of peak/
double start2 = center2-user1;  //start2 = where you want to begin extracting from each peak/
double peakdistance = RR[3]-RR[1];  //xint3-xint1 = distance between peaks/

win -a %V;  //makes %V the active window/
worksheet -da extracted1;  //copies active worksheet to new worksheet/
dataset aaaa;  //declares a ton of variables/
dataset bbbb;
dataset cccc;
dataset dddd;
dataset eeee;
dataset ffff;
range aaa = [extracted1]%K!col(a);
range bbb = [extracted1]%K!col(b);
range ccc = [extracted1]%K!col(c);
range ddd = [extracted1]%K!col(d);
range eee = [extracted1]%K!col(e);
range fff = [extracted1]%K!col(f);
range aa = [%V]%K!col(a);
range bb = [%V]%K!col(b);
range cc = [%V]%K!col(c);
range dd = [%V]%K!col(d);
range ee = [%V]%K!col(e);
range ff = [%V]%K!col(f);
int xyz = 1;
for(m=1;m<NumOfRow;m++) //loops for every row//
{
    if((rmod((m-(start1)),(peakdistance))<user) &&
    (rmod((m-(start1)),(peakdistance))>=0)) //looks if the data is
    a peak//
    {
        aaaa[xyz] = aa[m]; //puts data into dataset//
        bbbb[xyz] = bb[m];
        cccc[xyz] = cc[m];
        dddd[xyz] = dd[m];
        eeee[xyz] = ee[m];
        ffff[xyz] = ff[m];
        xyz++;
    }
}
copy -x aaaa aaa; //copies dataset into
worksheet//
copy -x bbbb bbb;
copy -x cccc ccc;
copy -x dddd ddd;
copy -x eeee eee;
copy -x ffff fff;
sum(bbb); //gets means
of the columns//
double mean1 = sum.mean;
sum(ccc);
double mean3 = sum.mean;
sum(ddd);
double mean5 = sum.mean;
sum(eee);
double mean7 = sum.mean;
sum(fff);
double mean9 = sum.mean;
}
define extract2
{
    win -a %V; //makes %V the
active window//
    worksheet -da extracted2; //copies active worksheet to new
worksheet//
dataset aaaa2; //declares a lot of
variables//
dataset bbbb2;
dataset cccc2;
dataset dddd2;
dataset eeee2;
dataset ffff2;
range aaa2 = [extracted2]%K!col(a);
range bbb2 = [extracted2]%K!col(b);
range ccc2 = [extracted2]%K!col(c);
range ddd2 = [extracted2]%K!col(d);
range eee2 = [extracted2]%K!col(e);
range fff2 = [extracted2]%K!col(f);
```c
int xy = 1;
for(s=1;s<NumOfRow;s++)
{
    if((rmod((s-(start2)),(peakdistance))<user) &&
      (rmod((s-(start2)),(peakdistance))>=0))
        //looks if the data is
        //a peak/
        {
            aaaa2[xy] = aa[s]; //puts data into dataset/
            bbbb2[xy] = bb[s];
            cccc2[xy] = cc[s];
            dddd2[xy] = dd[s];
            eeee2[xy] = ee[s];
            ffff2[xy] = ff[s];
            xy++;
        }
}
copy -x aaaa2 aaa2; //copies dataset into
worksheet//
copy -x bbbb2 bbb2;
copy -x cccc2 ccc2;
copy -x dddd2 ddd2;
copy -x eeee2 eee2;
copy -x ffff2 fff2;
sum(bbb2); //gets means
of columns//
    double mean2 = sum.mean;
    sum(ccc2);
    double mean4 = sum.mean;
    sum(ddd2);
    double mean6 = sum.mean;
    sum(eee2);
    double mean8 = sum.mean;
    sum(fff2);
    double mean10 = sum.mean;
}
define therest
{
    double amplitude1 = abs(mean1)+abs(mean2); //gets amplitudes of columns by adding abs of means/
    double amplitude2 = abs(mean3)+abs(mean4);
    double amplitude3 = abs(mean5)+abs(mean6);
    double amplitude4 = abs(mean7)+abs(mean8);
    double amplitude5 = abs(mean9)+abs(mean10);
    double ratio12 = amplitude1/amplitude2; //gets ratios by dividing amplitudes/
    double ratio13 = amplitude1/amplitude3;
    double ratio14 = amplitude1/amplitude4;
    double ratio15 = amplitude1/amplitude5;
    double ratio23 = amplitude2/amplitude3;
    double ratio24 = amplitude2/amplitude4;
    double ratio25 = amplitude2/amplitude5;
    double ratio34 = amplitude3/amplitude4;
    double ratio35 = amplitude3/amplitude5;
    double ratio45 = amplitude4/amplitude5;
}
window -cd extracted1; //deletes extracted data pages//
window -cd extracted2;
win -r %V %W; //sets worksheet name
back to %W//
//if on first worksheet//
if(jr==1)
{
    win -t wks origin ratios; //makes new worksheet called ratios//
    worksheet -a (9); //adds columns//
}
else
{
    win -a ratios; //makes ratios the
```
active window/

}  
cell(jr,2) = $(ln(ratio12));  //fills worksheet/
cell(jr,3) = $(ln(ratio13));
cell(jr,4) = $(ln(ratio14));
cell(jr,5) = $(ln(ratio15));
cell(jr,6) = $(ln(ratio23));
cell(jr,7) = $(ln(ratio24));
cell(jr,8) = $(ln(ratio25));
cell(jr,9) = $(ln(ratio34));
cell(jr,10) = $(ln(ratio35));
cell(jr,11) = $(ln(ratio45));
cell(jr,1) = "%K";

} doc -e W 
{  //for every workbook/
    string stringy$ = %H;
    stringy.Delete(13);
    //if name has 13 characters delete 13th character
    %j = stringy$;
    %U = A;
    %T = %j%U;
    //%T = original name with A added to the end
    win -r %H %T;
    //rename workbook to
    %T/
};
doc -cw;
int jr = 1;
doc -e W 
{
    //for every workbook/
    getfreq;
    //macros/
    extract1;
    extract2;
    therest;
    jr++;
    //add to counter
    if(jr > count) break;
};