Solutions to the Heat Equation for Circular Target Foil Heating by an Electron Beam for Uniform and Gaussian Beam Distributions

donald.jones@temple.edu

February 25, 2022

The purpose of this document is to provide a detailed solution to the heat equation for the specific situation of a circular Fe foil with an electron beam heat source at its center. However, the methods can be extended to more complex geometries where analytical solutions may not be possible. This technical note TargetHeating.tex, TargetHeating.pdf and the accompanying code FeFoilHeating.C can be found in the following Github repository:
https://github.com/jonesdc76/MollerPolarimetry/tree/master/TargetPolarization

1 Solving the Heat Equation for Conditions Specific to the Hall A Møller Polarimeter

To calculate the heating of the Møller polarimeter iron foil we start with the heat equation. Given the geometry of the Møller foil where we have a circular10 μm thick foil with a beam heat source located at the center, we can assume this has no azimuthal or z-dependence and we are left with only a radial dependence:

$$\rho C_p \frac{\partial T}{\partial t} = \kappa \nabla^2 T + \rho \alpha B_{\text{flux}} - \frac{2\sigma\epsilon}{\Delta z}\left(T^4 - T_0^4\right). \tag{1}$$

- $T(r,t)$ is the foil temperature in Kelvin,
- κ is the temperature dependent thermal conductivity of Fe which is approximately 0.8 W/(K cm) at room temperature (see Fig. 1),
- $\rho = 7.87$ g/cm^3 is the density of Fe,
- $\sigma = 5.67 \times 10^{-12}$ W/(K^4 cm^2) is the Stefan-Boltzmann constant,
- ϵ is the foil emissivity which depends on the polish and structure of the surface ranging from 0 (perfect polish) to 1 (perfect blackbody). Given the polish of the foil, something like 0.1 can be assumed.

- $T_0 = 294$ K, is the ambient temperature of the target ladder holding the foil at its boundary,
- $\Delta z = 10$ μm is the thickness of the foil,
- α is the collision stopping power for electrons in Fe. It is a function of electron energy and is 2.043 (MeV cm^2)/g=3.273$\times 10^{-13}$(J cm^2)/g for a 10 GeV electron using ESTAR. The ESTAR data along with a 5-degree polynomial fit used to calculate α as a function of energy is shown in Fig. 2. Care should be exercised when extrapolating outside the 1-10 GeV range.
- $C_p = 0.45$ J/(g K) is the specific heat of Fe and,
- $B_{\rm flux} = \frac{d^3N_e}{dsdt}$ is the flux density of the beam in $e^-/(\text{cm}^2\ \text{s})$.

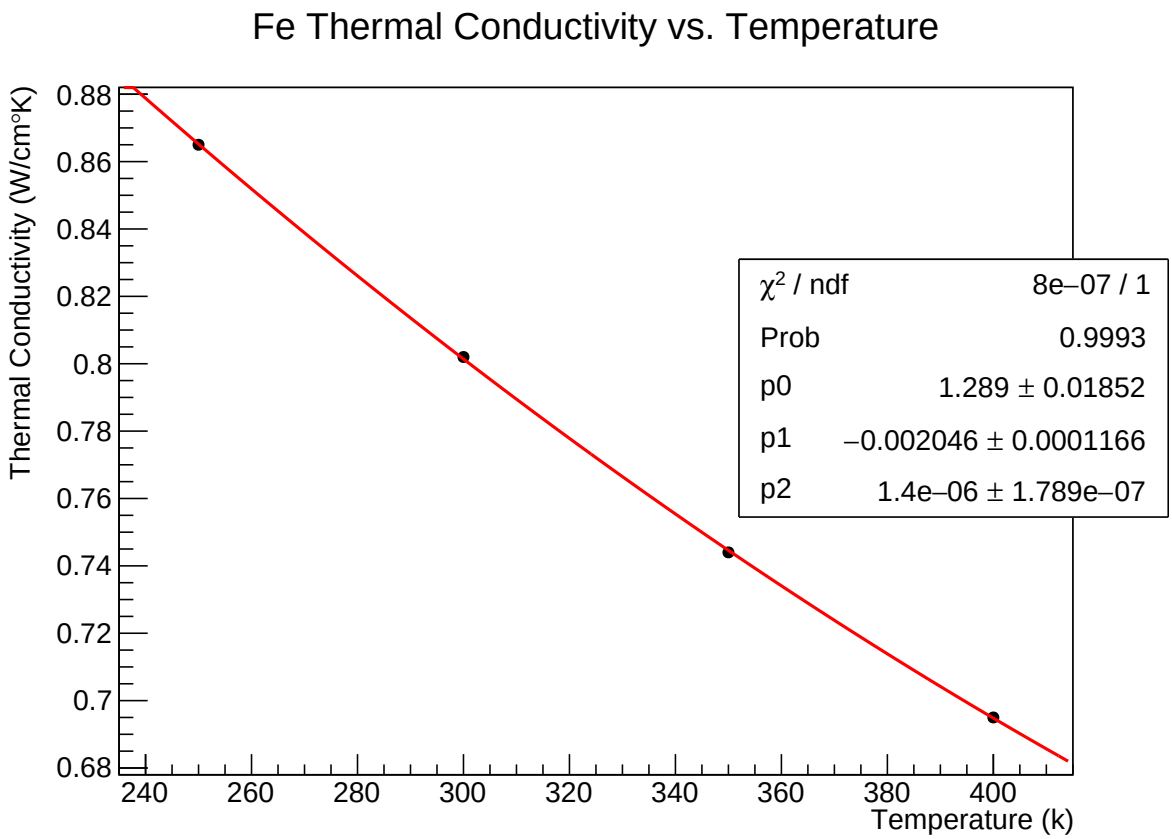

Figure 1: Fe thermal conductivityκ as a function of temperature. Data are from https://www.efunda.com/materials/elements/TC_Table.cfm?Element_ID=Fe and are fit to a 2nd degree polynomial.

In principle T and $B_{\rm flux}$ are functions of position and time. However, we are interested in the temperature of the steady state which is presumably reached quite rapidly when the beam turns on. Setting $\frac{\partial T}{\partial t} = 0$ simplifies Eq. 1. The expected heat load on a 10 μm thick Fe foil in the electron beam is about 12 mW/μA. If the temperature increase with beam inside the beam flux is of 30 degrees Celsius or less, over a beam radius of 1 mm, then the radiated energy in this circular area is 0.13 mW or about 1% of the heat load. In this case, we can safely neglect the radiative

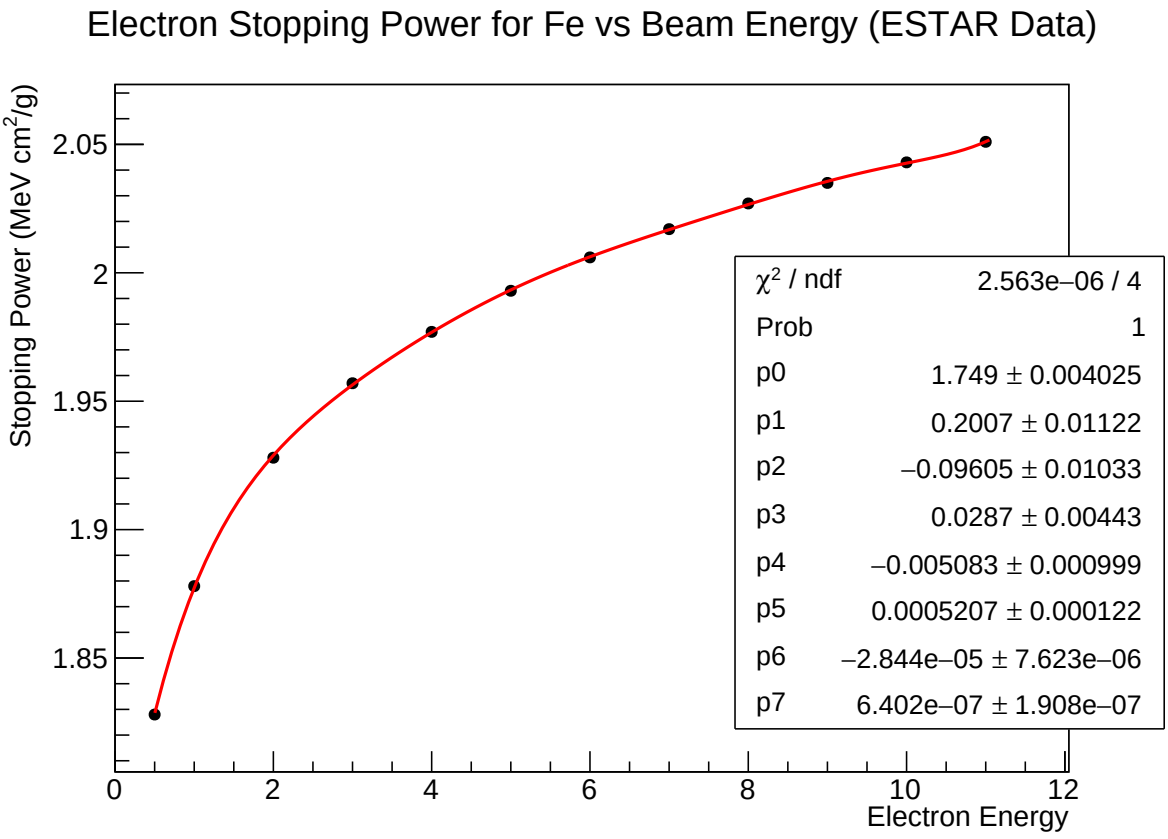

Figure 2: Stopping power for electrons as a function of energy in Fe. Data are from ESTAR and are fit to a 5-degree polynomial.

cooling term. If we end up with a temperature increase greater than 30 degrees, then we will have to revisit this assumption. Under these assumptions, Eq. 1 simplifies to

$$\kappa \nabla^2 T = -\rho \alpha B_{\text{flux}} \tag{2}$$

$$\frac{\kappa}{r}\frac{\partial}{\partial r}\left(r\frac{\partial T}{\partial r}\right) = -\rho \alpha B_{\text{flux}} \tag{3}$$

$$\frac{\partial}{\partial r}\left(r\frac{\partial T}{\partial r}\right) = -\frac{\rho \alpha}{\kappa} r B_{\text{flux}}. \tag{4}$$

1.1 Solving for a heating from a Gaussian profile beam spot

The Hall A Møller polarimeter, does not typically take rastered beam, and has an approximately Gaussian flux profile. We will consider the case of a circular Gaussian profile of 1 σ radius r_b. Therefore, the Gaussian profiled electron flux B_{flux} from a beam current I in amperes with a radius of r_b becomes

$$B_{\text{flux}} = \frac{I}{1.6 \times 10^{-19}\,(2\pi r_b^2)} e^{-r^2/2r_b^2}. \tag{5}$$

Inserting this density profile for the electron beam heat source into Eq. 4 gives

$$\frac{\partial}{\partial r}\left(r\frac{\partial T}{\partial r}\right) = -\gamma r e^{-r^2/2r_b^2}, \tag{6}$$

where $\gamma \equiv \frac{I\rho\alpha}{1.6\times10^{-19}\kappa\left(2\pi r_b^2\right)}$. Integrating both sides of Eq. 6 w.r.t. r gives

$$r\frac{\partial T}{\partial r} = r_b^2\gamma e^{-r^2/2r_b^2} + C, \tag{7}$$

$$\frac{\partial T}{\partial r} = \frac{r_b^2\gamma}{r}e^{-r^2/2r_b^2} + \frac{C}{r} \tag{8}$$

where C is a constant of integration to be determined from boundary conditions in the steady state. To determine C, the total heat load from the beam is given by $I\alpha\rho\Delta z/1.6\times10^{-19} = 16.1\Delta z$ W/(μA cm). The heat flow through the boundary is the product of the conductivity κ, the cross sectional area of the foil along the foil perimeter $2\pi R_{\rm foil}\Delta z$ and the temperature slope $\partial T/\partial r$, where length units are in cm. The perimeter of the foil at $R_{\rm foil}$ is assumed to be kept fixed at room temperature. The heat flow at the boundary has to equal the beam heat load in the steady state, so

$$\left(\kappa 2\pi R_{\rm foil}\Delta z\right)\frac{\partial T}{\partial r}|_{r=R_{\rm foil}} \approx -16.1\Delta z\left(\frac{\rm W}{\mu\rm A\ cm}\right) \approx \frac{\left(\kappa 2\pi R_{\rm foil}\Delta z\right)C}{R_{\rm foil}},$$

where the first term on the left side of Eq. 8) is not included since it is negligible at the boundary of the foil $R_{\rm foil}$. The negative sign comes from the direction of heat flow towards higher radius making the temperature decrease with increasing r.

$$C \approx -\frac{16.1}{2\pi\kappa} = -3.20\left(\frac{\rm K}{\mu\rm A}\right),$$

where the temperature dependent κ for Fe has been used (see Fig. 1). Now to find the temperature difference between the outside perimeter of the foil at $r = R_{\rm foil}$ and some $r < R_{\rm foil}$ integrate both sides from $R_{\rm foil}$ to r yielding

$$\Delta T = \int_{R_{\rm foil}}^{r}\left(\frac{r_b^2\gamma}{r'}e^{-r'^2/2r_b^2} + \frac{C}{r'}\right)dr'. \tag{9}$$

This can easily be integrated numerically as shown in Figures 3 and 4.

1.2 Solving for heating from a uniform circular distribution

In the case where a beam is rastered, the charge distribution can be considered to be uniform. Let's solve for the case of a uniform circular raster pattern of radius $r_{\rm rast}$ centered on the foil. In this case, the electron flux density B is given by

$$B = \frac{I\Theta(r_{\rm rast} - r)}{1.6\times10^{-19}\pi r_{\rm rast}^2}, \tag{10}$$

where $\Theta(r_{\rm rast} - r)$ is the Heaviside function which is unity for $r < r_{\rm rast}$ and zero for $r > r_{\rm rast}$. Inserting Eq. 10 into Eq. 4 yields

$$\frac{\partial}{\partial r}\left(r\frac{\partial T}{\partial r}\right) = -\frac{\rho\alpha}{\kappa}\frac{I\Theta(r_{\rm rast} - r)}{1.6\times10^{-19}\pi r_{\rm rast}^2}r \tag{11}$$

$$= -\gamma\Theta(r_{\rm rast} - r)r, \tag{12}$$

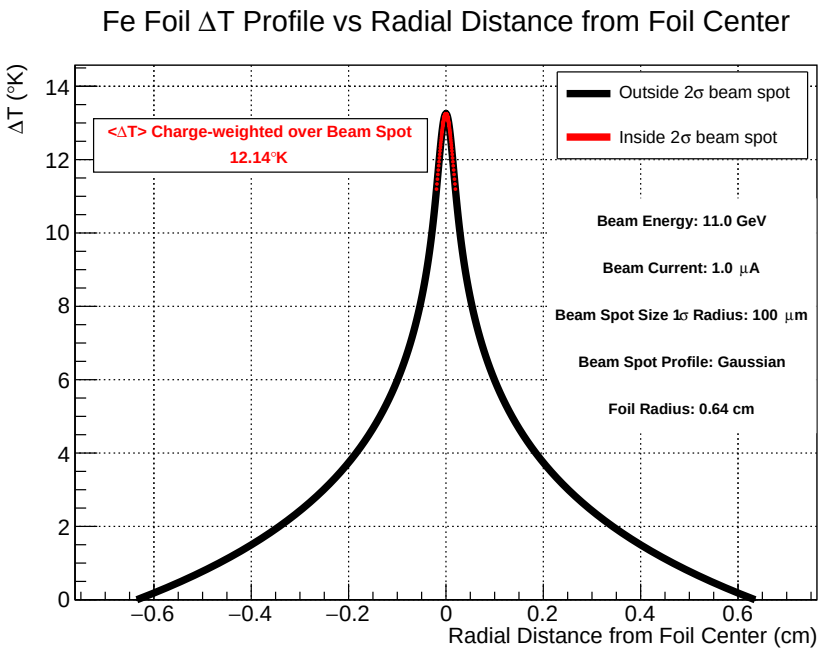

Figure 3: Fe foil ΔT profile from integrating Eq. 9 with beam spot size, and energy given.

where $\gamma \equiv \frac{\rho\alpha I}{1.602\times10^{-19}\kappa\pi r_{\text{rast}}^2}$. Integrating both sides with respect to r gives

$$r\frac{\partial T}{\partial r} = \begin{cases} -\frac{\gamma r^2}{2} + C, & r < r_{\text{rast}} \\ -\frac{\gamma r_{\text{rast}}^2}{2} + C, & r \geq r_{\text{rast}}. \end{cases}$$

This becomes

$$\frac{\partial T}{\partial r} = \begin{cases} -\frac{\gamma r^2}{2r} + \frac{C}{r}, & r < r_{\text{rast}} \\ -\frac{\gamma r_{\text{rast}}^2}{2r} + \frac{C}{r}, & r \geq r_{\text{rast}}. \end{cases}$$

Similar to before, the heat flow through the foil thickness at $r \geq r_{\text{rast}}$ has to equal the beam heat load in the steady state, so let's solve at $r = r_{\text{rast}}$:

$$(\kappa 2\pi r_{\text{rast}}\Delta z)\,\frac{\partial T}{\partial r}|_{r=R_{\text{foil}}} \approx \frac{\rho\alpha I}{1.602\times10^{-19}}\Delta z\left(\frac{\text{W}}{\mu\text{A cm}}\right) \approx \left(\frac{-\gamma r_{\text{rast}}^2}{2} + C\right)2\pi\kappa\Delta z.$$

Solving gives $C = 0$, so we now have

$$\frac{\partial T}{\partial r} = \begin{cases} -\frac{\gamma r^2}{2r}, & r < r_{\text{rast}} \\ -\frac{\gamma r_{\text{rast}}^2}{2r}, & r \geq r_{\text{rast}}. \end{cases} \tag{13}$$

Integrating both sides with respect to r in reverse direction from $r = R_{\text{foil}}$ to $r \leq r_{\text{rast}}$ gives ΔT

$$\Delta T = \begin{cases} -\frac{\gamma r_{\text{rast}}^2}{2}\int_{R_{\text{foil}}}^{r_{\text{rast}}}\frac{dr}{r} - \frac{\gamma}{2}\int_{r_{\text{rast}}}^{r} r'dr', & r < r_{\text{rast}} \\ -\frac{\gamma r_{\text{rast}}^2}{2}\int_{R_{\text{foil}}}^{r}\frac{dr'}{r'}, & r \geq r_{\text{rast}} \end{cases},$$

which can be piecewise solved analytically yielding

$$\Delta T = \begin{cases} \frac{\gamma r_{\text{rast}}^2}{2}\ln\left(\frac{R_{\text{foil}}}{r_{\text{rast}}}\right) + \frac{\gamma}{4}\left(r_{\text{rast}}^2 - r^2\right), & r < r_{\text{rast}} \\ \frac{\gamma r_{\text{rast}}^2}{2}\ln\left(\frac{R_{\text{foil}}}{r}\right), & r \geq r_{\text{rast}} \end{cases}. \tag{14}$$

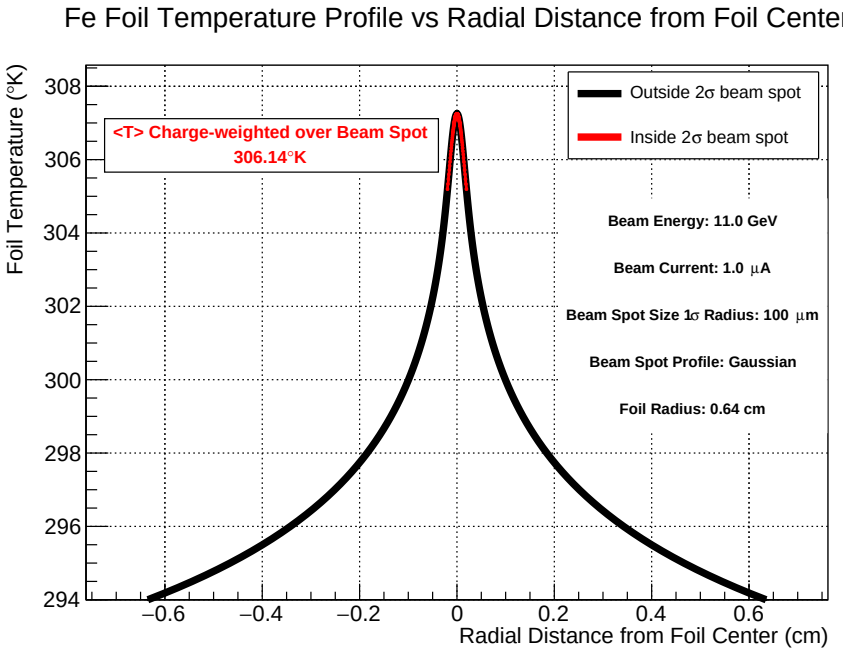

Figure 4: Fe foil temperature profile from integrating Eq. 9 with beam spot size, and energy given.

Figures 5 and 6 give plots of ΔT and T respectively for a uniformly rastered beam for the parameters given on the plots.

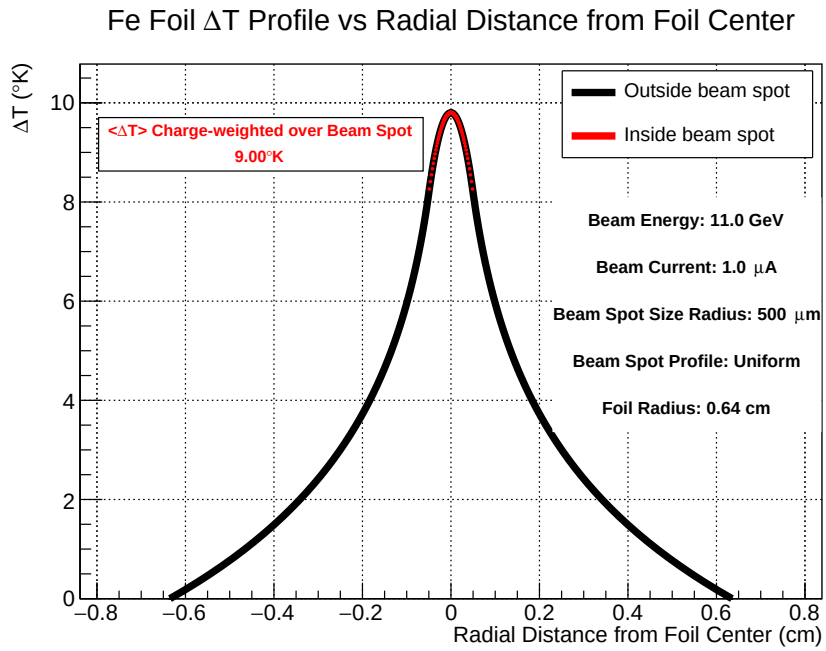

Figure 5: Fe foil ΔT profile from using Eq. 14 with beam spot size, and energy given.

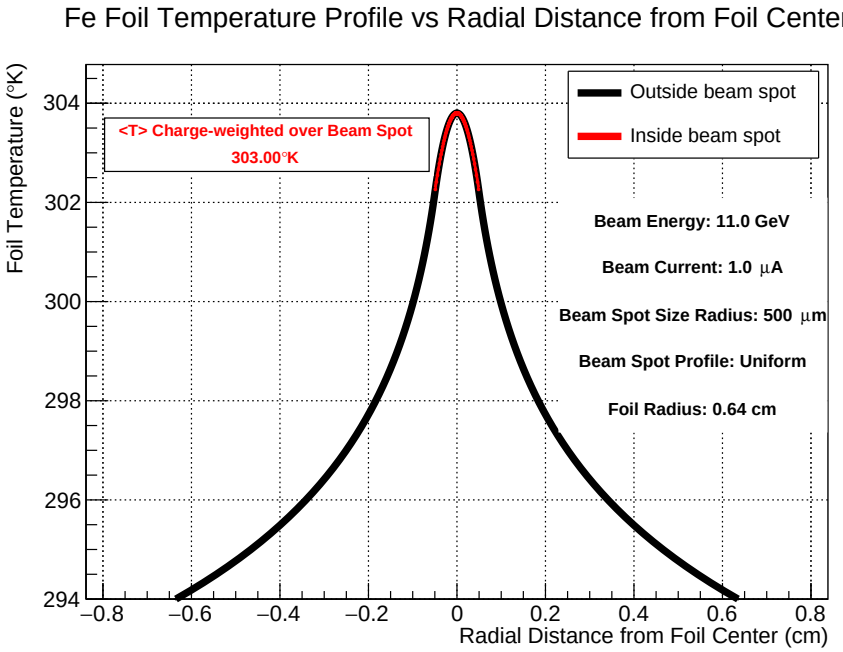

Figure 6: Fe foil temperature profile from integrating Eq. 14 with beam spot size, and energy given.

2 C++/ROOT Code for Numerically Integrating Eq. 9

The following ROOT macro uses Eq. 9 to calculate the foil heating for a circular Fe foil in a Gaussian profile electron beam. The latest version of this code is available at

```
#include "TF1.h"
#include <iostream>
#include "TGraph.h"
#include "TLegend.h"
#include "TAxis.h"
#include "TPad.h"
#include "TCanvas.h"
#include "TStyle.h"
#include "TPaveText.h"
#include "TString.h"

//////////////////
//Donald C. Jones//
//Nov. 2021    //
//////////////////

////////////////////////////////////////////////////////////////////////////////
//FeFoilHeating() calculates and graphs the temperature differnce in a thin circular
   //
//Fe foil between its edge held at a fixed temperature T0 and inside a circular //
//Gaussian-distributed or uniformly rastered electron beam.                    //
//                                                                             //
//Arguments:                                                                   //
//  beam_cur: beam current in Amperes                                          //
//  beam_r:  1 sigma beam spot size radius in cm (beam_r^2=sigma_x^2+sigma_y^2) //
//  beam_E:  beam energy in GeV                                                //
//  T0:      ambient (Hall) temperature in Kelvin taken as foil boundary temperature //
//  foil_r:  foil radius in cm (default is 1/2")                               //
//  uniform: uniform charge distribution? Otherwise, Gaussian assumed.         //
//                                                                             //
//Returns:                                                                     //
//the foil temperature difference in degrees K between T0 at the foil edge and the //
//temperature at the 1-sigma beam radius r_beam.                               //
//NOTE: it is helpful to recall that for a 2D circular Gaussian distribution the //
//     since the width is the quadrature sum of the x and y widths this is not the //
//     same as the 1-sigma width of the projected 1D distribution. The projected //
//     1D distribution width will be sqrt(2) smaller than the beam_r, the 2D 1-sigma
   //
//     width. The volume between r=0 and the n-sigma are as follows (where sigma //
//     is the quadrature sum of sigma_x and sigma_y):                          //
//     1 sigma = 39.35%, 2 sigma = 86.47%, 3 sigma = 98.89%, 4 sigma = 99.97%. //
////////////////////////////////////////////////////////////////////////////////

double FeFoilHeating(double beam_cur = 1e-6, double beam_r=15e-3, double beam_E = 11,
   double T0 = 294, double foil_r = 0.635, bool uniform = 0){
```

```
gStyle->SetStatY(0.7);
gStyle->SetStatH(0.2);
gStyle->SetOptFit(1111);
gStyle->SetTitleW(0.9);

bool save_plots = 1;
const double rho = 7.874;//density of Fe
const double sigma = 5.670e-12;//Stefan Boltzman constant W/(cm^2 K^4)
const double Cp = 0.45;//Fe specific heat capacity in J/(g K)
const double echarge = 1.602e-19;//Coulombs per electron
const double PI = 3.1415927;//pi obviously
const double foil_th = 0.001;//foil thickness in cm



//Use ESTAR data to estimate energy loss as a function of electron energy
//-------------------------------------------------------------------------------
TCanvas *c = new TCanvas("c","c",0,0,800,600);
double beam_en[10]={1,2,3,4,5,6,7,8,9,10};//beam energy in GeV
double stop_en[10]={1.878,1.928,1.957,1.977,1.993, //collision stopping power
         2.006,2.017,2.027,2.035,2.043};//in (MeV cm^2/g) using ESTAR
TGraph *grStop = new TGraph(10,beam_en,stop_en);
grStop->SetTitle("Electron Stopping Power for Fe vs Beam Energy (ESTAR Data)");
grStop->SetMarkerStyle(8);
grStop->Draw("ap");
grStop->GetXaxis()->SetTitle("Electron Energy");
grStop->GetYaxis()->SetTitle("Stopping Power (MeV cm^{2}/g)");
gPad->Update();
TF1 *fStop = new TF1("fStop","pol5",0,1);//use fit to give continuous function
grStop->Fit(fStop);
double alpha = echarge*fStop->Eval(beam_E)*1e6;//Collision stopping power in
    (Jcm^2/g)
cout<<"Stopping power "<<alpha<<" (J cm^2/g)"<<endl;
cout<<"Energy deposited in target: "<<alpha*rho*beam_cur/echarge*foil_th<<"
    W."<<endl;
if(save_plots)
  c->SaveAs("FeStoppingPower.pdf");



//Calculate the energy dependent thermal conductivity of Fe using data either from
//https://www.efunda.com/materials/elements/TC_Table.cfm?Element_ID=Fe
//or
//https://www.engineeringtoolbox.com/thermal-conductivity-metals-d_858.html
//-------------------------------------------------------------------------------
bool data_efunda = 1;
TCanvas *ct = new TCanvas("ct","ct",0,0,800,600);
double temp[4] = {250,300,350,400};
double cond[4] = {0.865,0.802,0.744,0.695};//www.efunda.com
TGraph *grC = new TGraph(4,temp,cond);
```

```
grC->SetTitle("Fe Thermal Conductivity vs. Temperature");
grC->SetMarkerStyle(8);
grC->Draw("ap");
grC->GetXaxis()->SetTitle("Temperature (k)");
grC->GetYaxis()->SetTitle("Thermal Conductivity (W/cm K)");
TF1 *fCond = new TF1("fCond","pol2",0,1);
grC->Fit(fCond);
gPad->Update();
if(!data_efunda)//www.engineeringtoolbox.com
  fCond = new TF1("fCond","1.13809-0.00111024*x",0,1);
double slope = uniform ? 19.5 : 17;
double guessTemp = T0+slope*beam_cur/1e-6;//starting guess for final foil
    temperature
double kappa = fCond->Eval(guessTemp);
cout<<"Conductivity at "<<guessTemp<<" K is "<<kappa<<endl;
if(save_plots)
  ct->SaveAs("FeThermalCond.pdf");



//Integral of f(r) gives delta T. Create the integrand f(r)
//-------------------------------------------------------------------------------
double gam = beam_cur/echarge*rho*alpha/kappa/PI/pow(beam_r,2)/(uniform ? 1.0 :
    2.0);
double C = -beam_cur/echarge*alpha*rho/2.0/PI/kappa;
TF1 *f = new TF1("f",Form("%e/x*exp(-x*x/%e)+%e/x",
          beam_r*beam_r*gam,2*beam_r*beam_r,C),0,foil_r);



//Improve thermal conductivity estimate using the calculated temperature.
//Temperature at 1.3*beam_r is a good estimate of the average temperature
//weighted by a Gaussian beam spot charge distribution. For a uniform distribution
//0.7*beam_r is a good estimate.
//-------------------------------------------------------------------------------
double r_est = (uniform ? 0.7 : 1.3)*beam_r;
if(uniform)
  guessTemp =
      gam*pow(beam_r,2)/2.0*log(foil_r/beam_r)+gam/4*(pow(beam_r,2)-pow(r_est,2))+T0;
else
  guessTemp = f->Integral(foil_r, r_est)+T0;
kappa = fCond->Eval(guessTemp);
gam = beam_cur/echarge*rho*alpha/kappa/PI/pow(beam_r,2)/(uniform ? 1.0 : 2.0);
C = -beam_cur/echarge*alpha*rho/2.0/PI/kappa;

cout<<"Conductivity re-calculated at "<<guessTemp<<" K is "<<kappa<<endl;
f = new
    TF1("f",Form("%e/x*exp(-x*x/%e)+%e/x",beam_r*beam_r*gam,2*beam_r*beam_r,C),0,foil_r);
```

```
//Graph resulting temperature profile by integrating f(r)dr. Make points red inside
//beam spot radius (2 sigma if Gaussian).
//--------------------------------------------------------------------------------
const int N=1000;
double r[N], T[N], dT[N],ri[N],Ti[N], dTi[N];
int n=0, ni=0;
double rp = foil_r;
double red_zone = uniform ? beam_r : 2*beam_r;
for(int i=0;i<N/2;++i){
  r[i]=rp;
  if(uniform){
    if(rp<red_zone)
 dT[i] = gam*pow(beam_r,2)/2.0*log(foil_r/beam_r)+gam/4.0*(pow(beam_r,2)-pow(rp,2));
    else
 dT[i] = gam*pow(beam_r,2)/2.0*log(foil_r/rp);
  }else{
    dT[i] = f->Integral(foil_r,rp);
  }
  T[i] = dT[i]+T0;
  if(rp<red_zone){
    ri[ni]=rp;
    Ti[ni]=T[i];
    dTi[ni]=dT[i];
    ++ni;
  }
  rp*=0.95;
  ++n;
  if(rp<0.00001)break;
}
for(int i=0;i<n;++i){
  r[i+n]=-r[n-i-1];
  dT[i+n] = dT[n-i-1];
  T[i+n] = T[n-i-1];
}
for(int i=0;i<ni;++i){
  ri[i+ni]=-ri[ni-i-1];
  dTi[i+ni] = dTi[ni-i-1];
  Ti[i+ni] = Ti[ni-i-1];
}
TCanvas *c1 = new TCanvas("c1","c1",0,0,800,600);
TGraph *grdT = new TGraph(2*n,r,dT);
grdT->SetMarkerStyle(8);
grdT->SetLineWidth(6);
grdT->SetMarkerSize(0.3);
grdT->Draw("acp");
grdT->SetTitle(Form("Fe Foil #DeltaT Profile vs Radial Distance from Foil Center"));
grdT->GetXaxis()->SetTitle("Radial Distance from Foil Center (cm)");
grdT->GetYaxis()->SetTitle("#DeltaT (K)");
TGraph *gridT = new TGraph(2*ni,ri,dTi);
```

```
gridT->SetMarkerStyle(8);
gridT->SetMarkerColor(kRed);
gridT->SetLineColor(kRed);
gridT->SetLineWidth(6);
gridT->SetMarkerSize(0.4);
gridT->Draw("samep");
TPaveText *pt = new TPaveText(0.61,0.35,0.899,0.7,"ndc");
pt->SetFillColor(0);
pt->SetShadowColor(0);
pt->SetBorderSize(0);
pt->AddText(Form("Beam Energy: %0.1f GeV",beam_E));
pt->AddText(Form("Beam Current: %0.1f #muA", beam_cur*1e6));
TString str = Form("Beam Spot Size 1#sigma Radius: %0.1f #mum",beam_r*1e4);
if(uniform)
  str = Form("Beam Spot Size Radius: %0.1f #mum",beam_r*1e4);
pt->AddText(str.Data());
pt->AddText((char*)(uniform ? "Beam Spot Profile: Uniform" : "Beam Spot Profile:
    Gaussian"));
pt->AddText(Form("Foil Radius: %0.2f cm",foil_r));
pt->Draw();
TLegend *lg = new TLegend(0.62,0.76,0.89,0.89);
if(uniform){
  lg->AddEntry(grdT,"Outside beam spot","lp");
  lg->AddEntry(gridT,"Inside beam spot","lp");
}else{
  lg->AddEntry(grdT,"Outside 2#sigma beam spot","lp");
  lg->AddEntry(gridT,"Inside 2#sigma beam spot","lp");
}
lg->Draw();
TCanvas *c2 = new TCanvas("c2","c2",0,0,800,600);
TGraph *gr = new TGraph(2*n,r,T);
gr->SetMarkerStyle(8);
gr->SetLineWidth(6);
gr->SetMarkerSize(0.3);
gr->Draw("acp");
gr->SetTitle(Form("Fe Foil Temperature Profile vs Radial Distance from Foil
    Center"));
gr->GetYaxis()->SetTitle("Foil Temperature (K)");
gr->GetXaxis()->SetTitle("Radial Distance from Foil Center (cm)");
gr->GetYaxis()->SetRangeUser(T0,T0+grdT->GetYaxis()->GetXmax());
TGraph *gri = new TGraph(2*ni,ri,Ti);
gri->SetMarkerStyle(8);
gri->SetMarkerColor(kRed);
gri->SetLineColor(kRed);
gri->SetLineWidth(2);
gri->SetMarkerSize(0.4);
gri->Draw("samecp");
lg->Draw();
pt->Draw();
```

```
//Integrate f(r) weighted by the beam charge distribution to find the average delta
    T
//-------------------------------------------------------------------------------
gStyle->SetOptFit(0);
TF1 *fGaus = new TF1("fGaus","[0]*exp(-x*x/(2*[1]*[1]))+[2]",-2*beam_r,2*beam_r);
fGaus->SetParameters((guessTemp-T0)/2.,2*beam_r,T0+(guessTemp-T0)/2.);
cout<<(guessTemp-T0)/2.<<endl;
fGaus->SetLineWidth(2);
fGaus->SetLineColor(kRed);
gr->Fit(fGaus,"r");
if(uniform){
  fGaus->SetRange(-beam_r,beam_r);
  //fGaus->FixParameter(0,fGaus->GetParameter(0));
  fGaus->FixParameter(2,fGaus->GetParameter(2));
  gr->Fit(fGaus,"r");
}
TString fstr = Form("x*exp(-x*x/2./%e)/%e",beam_r*beam_r,beam_r*beam_r);
if(uniform)fstr = Form("2*x/%e",beam_r*beam_r);
TString func = Form("(%e*exp(-x*x/(2*%e))+%e)*%s",
         fGaus->GetParameter(0),pow(fGaus->GetParameter(1),2),
         fGaus->GetParameter(2), fstr.Data());
TF1 *fAvgT = new TF1("fAvgT",func.Data(),0,1);
fAvgT->SetNpx(1000);
 //fAvgT->Draw();
if(uniform)
  cout<<"dT at 0.7 beam radius is "<<fGaus->Eval(beam_r*0.7)<<endl;
else
  cout<<"dT at 1.3 sigma is "<<f->Integral(foil_r,beam_r*1.3)<<endl;



//Return average temperature, weighted by the beam spot charge distribution.
//-------------------------------------------------------------------------------
c1->SetGrid();
c1->cd();
double avg = fAvgT->Integral(0, (uniform ? 1.0 : 10.0 ) * beam_r);
TPaveText *pt1 = new TPaveText(0.12,0.74,0.48,0.82,"ndc");
pt1->SetFillColor(0);
pt1->SetShadowColor(0);
//pt1->SetBorderSize(0);
pt1->SetTextColor(kRed);
pt1->AddText(Form("<#DeltaT> Charge-weighted over Beam Spot"));
pt1->AddText(Form("%0.2f K",avg-T0));
pt1->Draw();
gPad->Update();
if(save_plots)
  c1->SaveAs(Form("FeFoilHeatingdT%s.pdf",(char*)(uniform ? "Uniform":"")));
c2->SetGrid();
c2->cd();
```

```
  TPaveText *pt2 = new TPaveText(0.12,0.74,0.48,0.82,"ndc");
  pt2->SetFillColor(0);
  pt2->SetShadowColor(0);
  //pt2->SetBorderSize(0);
  pt2->SetTextColor(kRed);
  pt2->AddText(Form("<T> Charge-weighted over Beam Spot"));
  pt2->AddText(Form("%0.2f K",avg));
  pt2->Draw();
  if(save_plots)
    c2->SaveAs(Form("FeFoilHeatingT%s.pdf",(char*)(uniform ? "Uniform":"")));
  cout<<"Total correction to magnetization for Fe: "<<-0.0238*(avg-T0)<<"
      emu/g"<<endl;
  return avg;
}
```