Longitudinal Disorder of Superconducting Pancake Vortex Arrays

Christopher Fetsch

Advisor: W.J. Kossler

May 8, 2000

Abstract

The high T_c superconductors are of type II, that is, in an applied magnetic field the field penetrates in the form of vortices. Further, the extreme anisotropy of these materials leads to longitudinal disorder along the vortex itself. A means to calculate the internal magnetic field distributions including the effects of disorder has been developed and is presented. Three dimensional vortices are built up from two-dimensional pancake vortices, each located in a single plane in the crystalline structure of the material. By summing the magnetic fields associated with each pancake, an overall field distribution is obtained. Disorder is introduced by randomly displacing the pancakes, and the magnetic field probability distributions are presented for a range of applied fields and degrees of disorder.

Introduction

Superconductors have been known and studied since 1911, when Dutch physicist Kammerlingh Onnes discovered the property in mercury at a temperature of 4K. The basic idea behind superconductivity is that many materials can exhibit zero resistance to current flow if cooled to a certain critical temperature (T_c) that is unique to the material [1]. Superconductors are described by two fundamental parameters: the penetration depth (λ) is the length in which magnetic fields die off within a superconductor, and the coherence length (ξ) is a measure of the distance over which local fields within a material have an noticeable effect. From these we define the Ginzburg-Landau parameter ($\kappa = \frac{\lambda}{\xi}$), which allows us to group superconductors into two types: type I ($\kappa < \frac{1}{\sqrt{2}}$), and type II ($\kappa > \frac{1}{\sqrt{2}}$) [2].

In 1933, Walter Meissner and Robert Ochsenfeld discovered another characteristic of superconductors, dubbed the Meissner Effect. They found that the electric currents normally induced in a conductor by a magnetic field were actually able to repel the field in a superconducting material [2]. In other words, when cooled below its T_c, a superconductor in an applied field will expel the field and become a perfect diamagnet. Only when the field strength reaches some critical value does the field re-enter the material. Depending on the type of superconductor, the field can either re-enter almost immediately or more gradually. In a type I material, the field is completely expelled when it is less than the critical field ($H < H_c$). If the field is greater than H_c, the material goes out of the superconducting state and the field penetrates completely. In type II materials, once the field reaches an initial critical field H_{C1}, there is a period of transition called the mixed state, lasting until superconductivity is lost at the second critical value H_{C2}. In the mixed state, the field en-

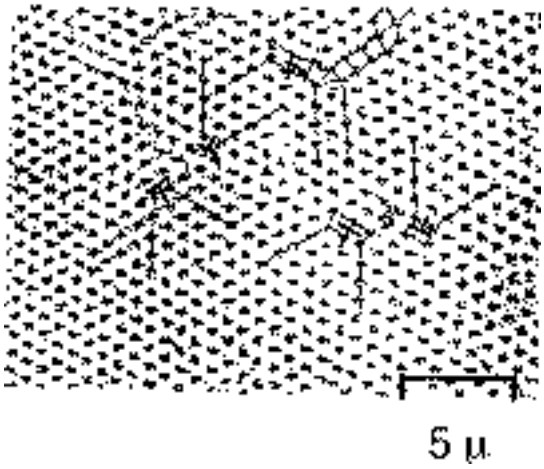

Figure 1: A FLL revealed by decoration with fine iron powder.

ters the sample in the form of *vortices*, each carrying a magnetic flux Φ_0 = hc/2e = 2.07 x $10^{-7} Gcm^2$. These vortices can be pictured as swirling tubes of electrical current at the center of which superconductivity is suppressed [2]. The properties of vortices are such that they usually arrange themselves at the corners of a triangular pattern known as a Flux Line Lattice (FLL) or Abrikosov lattice (Fig. 1). This arrangement minimizes the free energy of the system, with the vortices held apart by magnetic repulsion and arranged somewhat like drinking glasses on a cupboard shelf [3].

Theory

Calculations of the magnetic fields associated with a FLL can be performed using what is called the London theory, a much simpler alternative than the more comprehensive BCS theory [4, 2]. The fields for the vortices are actually calculated in three separate ways. The first is with a reciprocal lattice technique, the second is with three-dimensional vortices, and the third approach is with two-dimensional *pancake* vortices. To assemble the complete field

distribution, the the fields from the nearby three-dimensional vortices are subtracted from the reciprocal lattice result, and then the fields from nearby pancake vortices are added back in. This is accomplished by our computer programs described in a later section, but first we need to outline the theory for each approach.

Reciprocal Lattice Calculation

The first step in the development of the theory is the case of a single vortex in an *isotropic* superconductor. We can construct the free energy density of the superconductor by including a term for the energy density of the magnetic field ($\frac{b^2}{8\pi}$) and the kinetic energy density of the superconducting particles in the solid ($n_s \frac{1}{2} m v^2$, where n_s is the superconducting charge carrier density) [4]. Then, given the equation for current density:

$$j = n_s e v$$

where e is the charge per particle, we can solve for v and plug it into the kinetic energy density equation to obtain:

$$E_k = \frac{1}{2} \frac{m j^2}{n_s e^2}$$

From here we can use the Maxwell equation:

$$\nabla \times \mathbf{b} = \frac{4\pi}{c} \mathbf{j}$$

add in the magnetic field energy density, and derive the total free energy of the system [4]:

$$F = \int_V dV \frac{1}{8\pi} \left(b^2 + \frac{mc^2}{4\pi n_s e^2} |\nabla \times \mathbf{b}|^2 \right) = \frac{1}{8\pi} \int_V dV \left(b^2 + \lambda_L^2 |\nabla \times \mathbf{b}|^2 \right) \tag{1}$$

In brief, we can then perform a variational calculation of this result, minimizing the free energy and integrating by parts, and the result is an equation that allows the calculation of the magnetic fields b(x,y) in the mixed state of an isotropic superconductor:

$$\mathbf{b} + \lambda_L^2 \nabla \times (\nabla \times \mathbf{b}) = \Phi_0 \delta(\mathbf{r} - \mathbf{r}_v) \tag{2}$$

The materials relevant to this study, the newer, high temperature copper-oxide superconductors, are highly *anisotropic*. This means the current flows more easily in a direction parallel to the CuO_2 planes [6]. The next step in the process, therefore, is to introduce anisotropy into the theory by considering the masses of the superconducting charge carriers both within (M_1) and perpendicular to (M_3) the superconducting planes as components of an effective mass tensor [1]. These masses first must be normalized so that:

$$m_1 = \frac{M_1}{M_{av}}$$
$$m_3 = \frac{M_3}{M_{av}}$$
$$\text{where } M_{av} = (M_1^2 M_3)^{1/3}$$

We can make the following modification to the penetration depth in Eqs. 1 and 2:

$$\lambda_L^2 = \frac{mc^2}{4\pi n_s e^2} \rightarrow \lambda^2 m_{ij} = \frac{M_{av}c^2}{4\pi n_s e^3} m_{i,j} \tag{3}$$

Here m_{ij} are the components of the tensor and i and j are either x, y, or z [1]. From our new expression of the free energy density, the same minimization calculation is performed, and the field equations for the anisotropic case are:

$$b_k = \lambda^2 m_{i,j} \epsilon_{ist} \epsilon_{jkl} \left(\frac{\partial^2 b_s}{\partial x_t \partial x_l} \right) + \Phi_0 \delta_{zk} (\mathbf{r} - \mathbf{r}_v) \tag{4}$$

It remains to extend the theory to enable the calculation of the magnetic field from any point within a vortex lattice. The calculation involves a Fourier transform of the three vector components of equation 4 (b_x, b_y, b_z) into reciprocal lattice space, using the following relationship:

$$\begin{aligned} \mathbf{b}(\mathbf{r}) &= \sum_{\mathbf{G}} \mathbf{b}(\mathbf{G}) e^{i\mathbf{G}\cdot\mathbf{r}} \\ \mathbf{b}(\mathbf{G}) &= \frac{B}{\Phi_0} \int \mathbf{b}(\mathbf{r}) e^{-i\mathbf{G}\cdot\mathbf{r}} d^3 r \end{aligned} \tag{5}$$

The $\mathbf{G}$'s are the reciprocal lattice vectors, and B is the average field strength over the FLL unit cell. We can now write the components of the field in terms of the reciprocal lattice

vectors [1]:

$$b_x(\mathbf{G}) = B\lambda^2 m_{xz} G_y^2/d$$

$$b_y(\mathbf{G}) = -B\lambda^2 m_{xz} G_x G_y/d$$

$$b_z(\mathbf{G}) = B(1+\lambda^2 m_{zz} G^2)/d$$

$$\text{where } d = (1+\lambda^2 m_1 G_x^2 + \lambda^2 m_{xx} G_y^2)(1+\lambda^2 m_{zz} G^2) - \lambda^4 m_{xx}^2 G^2 G_y^2 \tag{6}$$

Finally, one must perform the sum in Eq. 5 over the reciprocal lattice vectors for each component in Eq. 6.

As it turns out, we will be comparing against field distributions in a single crystal with the applied field parallel to the **c** (crystal) axis. Any complications associated with anisotropy, therefore, could have been avoided, but the original program described below utilizes the more general form.

Three Dimensional Vortices

In the isotropic case, the magnetic field from a single, three-dimensional vortex is:

$$\mathbf{b} = \frac{\Phi_0}{2\pi\lambda^2} K_0(\frac{r}{\lambda})\hat{z} \tag{7}$$

where K_0 is the zero order Bessel function of imaginary argument and$\hat{z}$ is a unit vector in the z direction[4]. The fields then can be built up from an array of such vortex fields. As explained above, this equation is still useable for the anisotropic case when the applied field is parallel to the **c** axis.

Pancake Fields

Using the anisotropic London theory, the microscopic magnetic fields within a FLL can be calculated numerically by means of a computer program, given the necessary input. The required parameters are the average magnetic field (B), the effective penetration depth (λ), the angle between B and the direction of the crystal structure (θ), and the anisotropy parameter (Γ) for the superconductor in question [1]. This anisotropy parameter, equal to m_3/m_1, varies greatly in the different copper-oxide superconductors. The London theory as developed here (Eqs. 2 and 6) applies well to YBCO ($YBa_2Cu_3O_7$), which has a Γ of 25. BSCCO ($Bi_2Sr_2Ca_1Cu_2O_{8+\delta}$) on the other hand, has a Γ of over 3000, requiring us to utilize a somewhat different model of the vortex structure. Clem [6] has emphasized the discreteness of the CuO_2 layers in introducing the theory of pancake vortices. The model describes a stack of these two-dimensional pancakes, one at each CuO plane in the crystal structure, as opposed to a continuous flux tube running through the material (as with three-dimensional vortices). The magnetic field produced by a single pancake vortex in an isolated superconducting layer is shown to be:

$$\mathbf{b} = \hat{\rho} b_\rho(\rho, z) + \hat{z} b_z(r, z) \tag{8}$$

where $\hat{r}$ and $\hat{z}$ are the unit vectors in cylindrical coordinates ($\hat{r} = \hat{x}\cos\phi + \hat{y}\sin\phi$) [6]. The field components for such a vortex in an infinite stack of superconducting layers is:

$$b_z(r,z) = \frac{\Phi_0}{2\pi\Lambda r} e^{-r/\lambda_{||}} \tag{9}$$

$$b_\rho(r,z) = \frac{\Phi_0}{2\pi\Lambda r}\left(\frac{z}{|z|}e^{-|z|/\lambda_{||}} - \frac{z}{r}e^{-r/\lambda_{||}}\right) \tag{10}$$

where Φ_0 is the magnetic flux quantum, Λ is the thin-film screening length, and $\lambda_{||}$ is the effective penetration depth ($\lambda_{||} = (s\Lambda/2)^{1/2}$, and s is the spacing between sheets). These fields can be summed to produce the magnetic field from a lattice of pancake vortices.

Obtaining the Magnetic Field Probability Distribution

Once a data set of field values is obtained, it is useful to determine the probability distribution of the values contained in the data. This step involves a numerical integration over the field contours:

$$n(M) = \int_A \delta((M(x,y) - M_0)dA \tag{11}$$

Here A is the area of the region in question (one unit of the FLL), M(x,y) is the field value at the position (x,y), M_0 is the value at some point on the contour line, and δ is the Dirac delta function. The resulting magnetic field probability distributions can then be plotted an analyzed.

Disorder

At low temperatures, the vortices in a superconductor form a static lattice, and all of the above theory applies. When higher temperatures are involved, however, individual vortices can shear, bend, and even jitter back and forth in the plane. The pancake model has been used to develop a theory of this phenomenon of vortex *disorder*. Disorder refers to the disruption of the normal equilibrium positions of the vortices in a flux line lattice. A greater heat energy, for example, causes the vortex lattice to *melt* in a sense and become a two-dimensional flux liquid [3].

Calculations

The portion of the project I have been involved in consists of using Fortran programs to calculate the field distributions, then subsequently modifying the calculations to accommodate disorder. We began with three primary programs, each corresponding to one of the three approaches for calculating the fields (see Appendix): hpxy.f performs the near pancake calculation, hspag.f calculates the near direct lattice (from the three-dimensional vortices, designated here as *spaghetti* fields), and hsum.f calculates the reciprocal lattice result. Before we could proceed, we needed to verify that these previously existing programs were working properly. In the near limit, hspag.f and hpxy.f should produce the same results, but the numbers were significantly off. The problem was traced to hpxy.f, but after we fixed it we encountered a similar setback with hsum.f. To resolve this, we inserted a more straightforward calculation of the reciprocal lattice result from Tinkham [7], creating a newer version

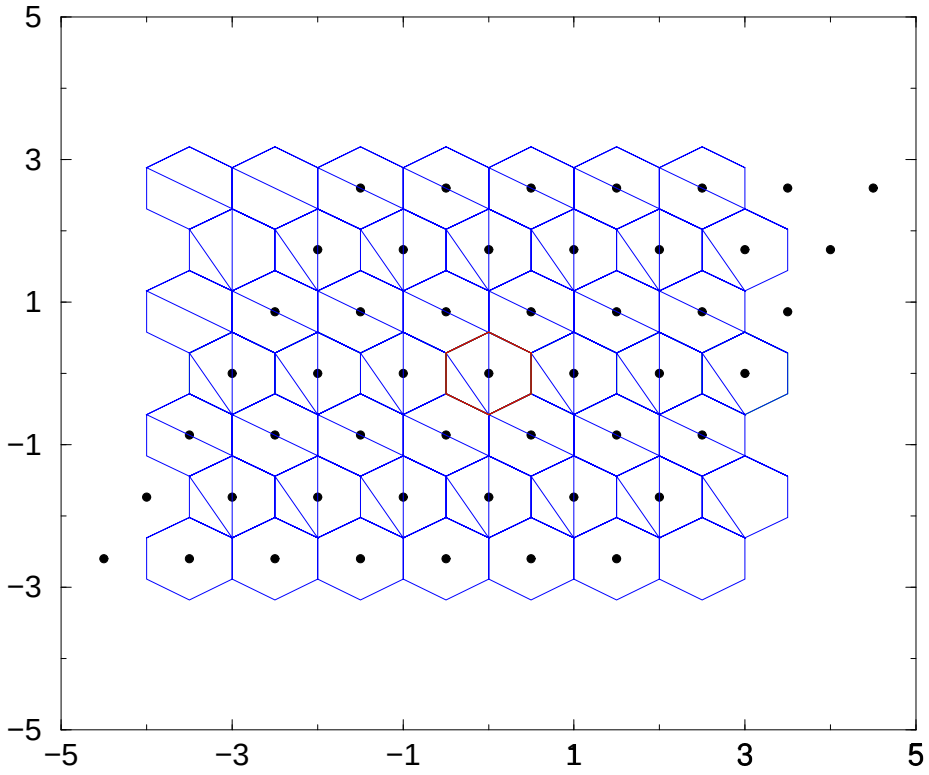

Figure 2: Wigner-Seitz cells.

of hsum.f.

After this initial debugging phase, we could finally begin focusing on disorder. The disorder consists of a series of random displacements in the x-y plane as one moves up or down in a stack of pancakes, so to include it in the pancake program we needed to utilize a random number generator. Since these displacements could wander quite far from the origin, the first step was to create a program that could take a set of coordinates and re-map them onto an equivalent position closer to the origin (that is, with the same position relative to a vortex). For this it was necessary to employ the concept of a Wigner-Seitz cell, defined as the region of space about a lattice point that is closer to that point than to any other lattice point [7]. We used the XMGR application to plot a grid of points and their surrounding hexagonal Wigner-Seitz cells as a visual aid (Fig. 2).

It was these hexagonal cells that were the basis for our re-mapping. No matter where the point had strayed, its position in any Wigner-Seitz cell was considered equivalent to the corresponding position in the origin cell. The next task, therefore, was to write a program that would take the coordinates of the end point of the random walk, find the nearest vortex to that point, and then subtract off the distance from that vortex to the origin. This program (xnearest.f) does so by comparing the squares of the distances from the point to the four possible nearest vortices (those surrounding the point). The xnearest subroutine then returns the coordinates of the re-mapped point, which would always be within the first Wigner-Seitz cell. We tested this program using the same gaussian probability distribution method of random number generation that would be used in the actual calculation of disorder. The random numbers were scaled so that on average they would be greater than the dimensions of a single Wigner-Seitz cell, so we would expect the resulting re-mapped points to be roughly evenly distributed within the origin cell. This was indeed the case (see figure 3), and we were then able to insert the pancake calculation into the program containing the random walk and re-mapping functions.

The product of this combination was the program tryout.f, which generates the array of field values predicted by the pancake theory and modified by the disorder calculation. This array can then be fed into another previously existing program, integp.f, which does the numerical integration described in equation 11 to calculate the magnetic field distributions. Once we were convinced that the tryout program was working correctly, we renamed it hpxy.f and reverted back to an older program (longdis.f) that called each of the three magnetic field programs (hxpy.f, hspag.f, and hsum.f). Finally, the complete picture could be assembled,

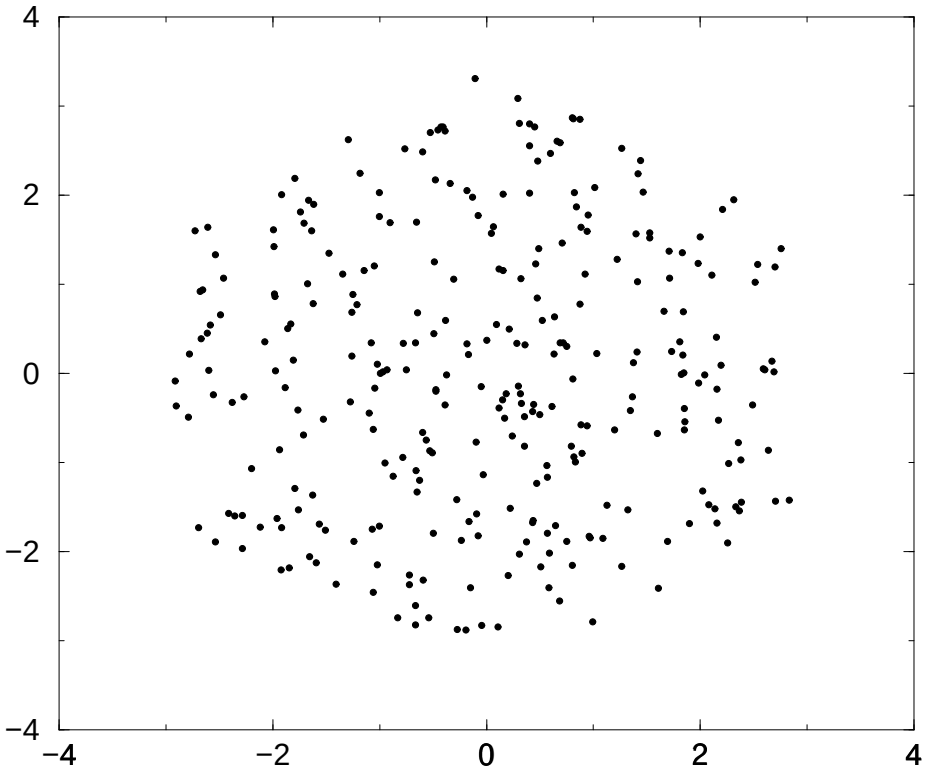

Figure 3: The test of the remapping subroutine (xnearest.f).

and integp.f could be used to finalize the task and produce actual theoretical data in the form of the field distributions n(b) as a function of magnetic field. The basic sequence of programs used to generate these data is summarized in figure 4.

Results

These distributions are subject to a number of parameters that are read in by the program. The first is the scale factor (sc) for the random walk calculation, in units of thousands of angstroms. The others include the applied field strength B, the number of pancake sheets summed over (nsh), the spread of pancakes in the x-y plane (npanxy), and the number of iterations of the pancake calculation being averaged (nave). For initial comparison, and to

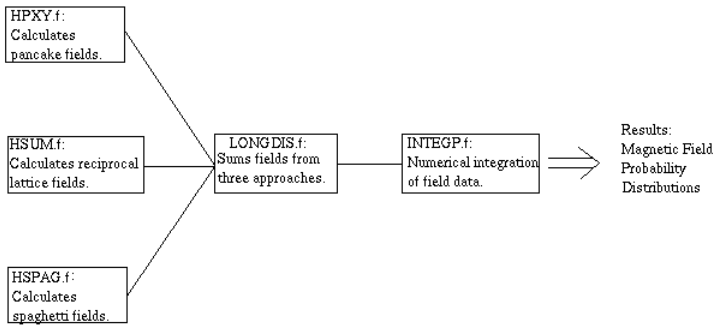

Figure 4: A flowchart of the main programs used to sum the magnetic fields and calculate the probability distributions.

see if the numbers were coming out as expected, we generated data for fields of 1000 G, 10000 G, and 60000 G, and for scale factors of 0, .02, .04, .05, .06, .08, .1, .14, .18, .2, .4, .6, .8, 1.0, 2.0, and 3.0. The expectation was that with greater disorder (higher sc), the distribution would become narrower and more like a straight peak instead of one with a gradual downslope (fig. 5). Also, with a higher field, we expect the effect of a given scale factor to be more pronounced. This result can be seen in the progression of figures 6, 7, and 8.

Conclusion

As expected, the field distribution narrows with increasing disorder, but we did encounter one obvious discrepancy. The average field was supposed to be roughly equal to the external applied field (B), but we found that with a higher field and increasing disorder the average

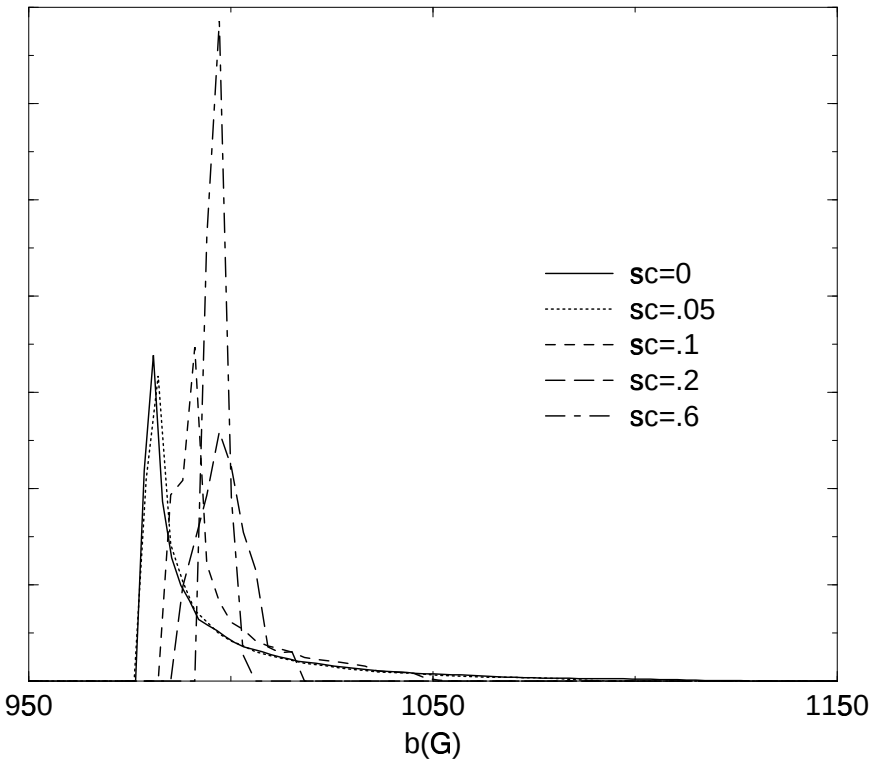

Figure 5: The probability distributions for 1 kG and variable step size (sc) in kÅ units.

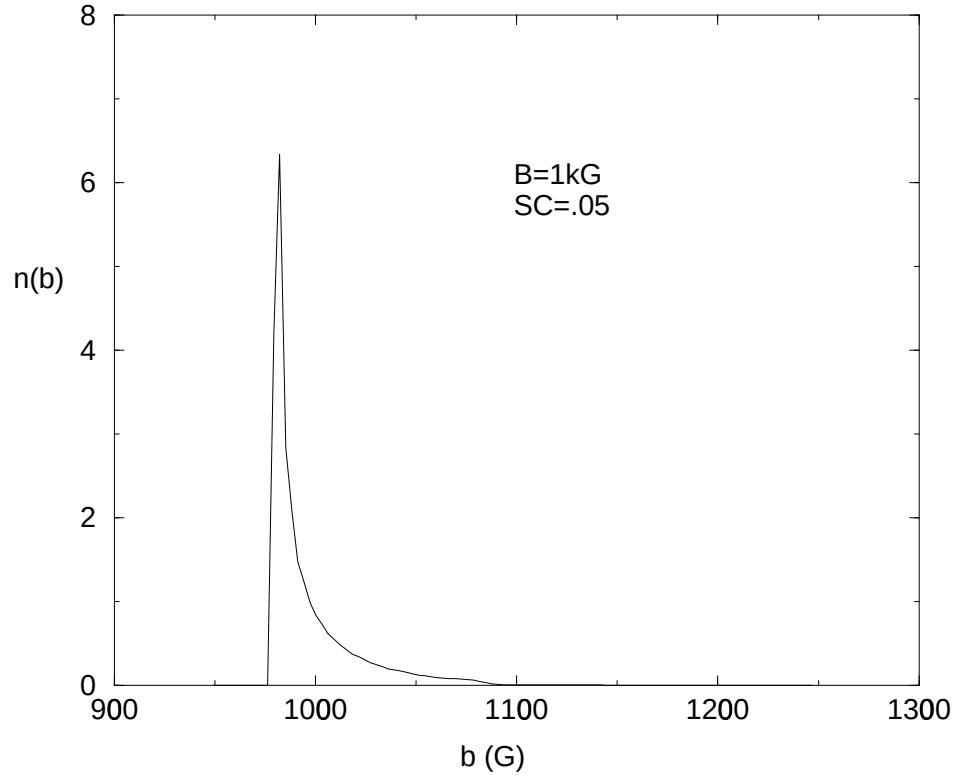

Figure 6: Probability distribution for sc=.05 and b=1kG

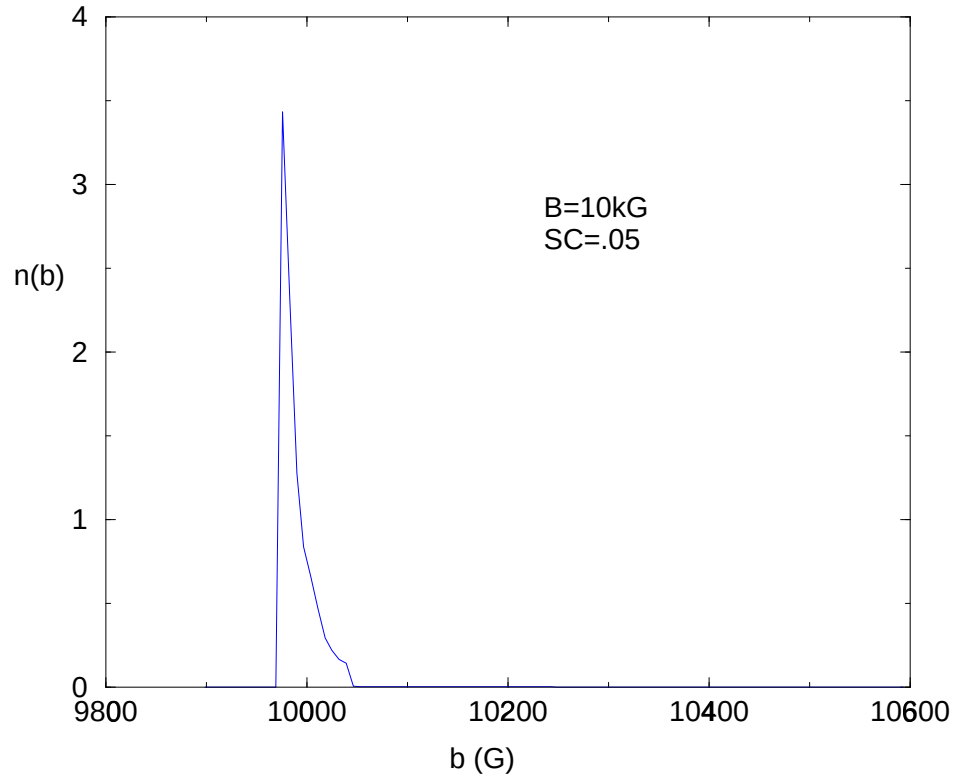

Figure 7: Probability distribution for sc=.05 and b=10kG

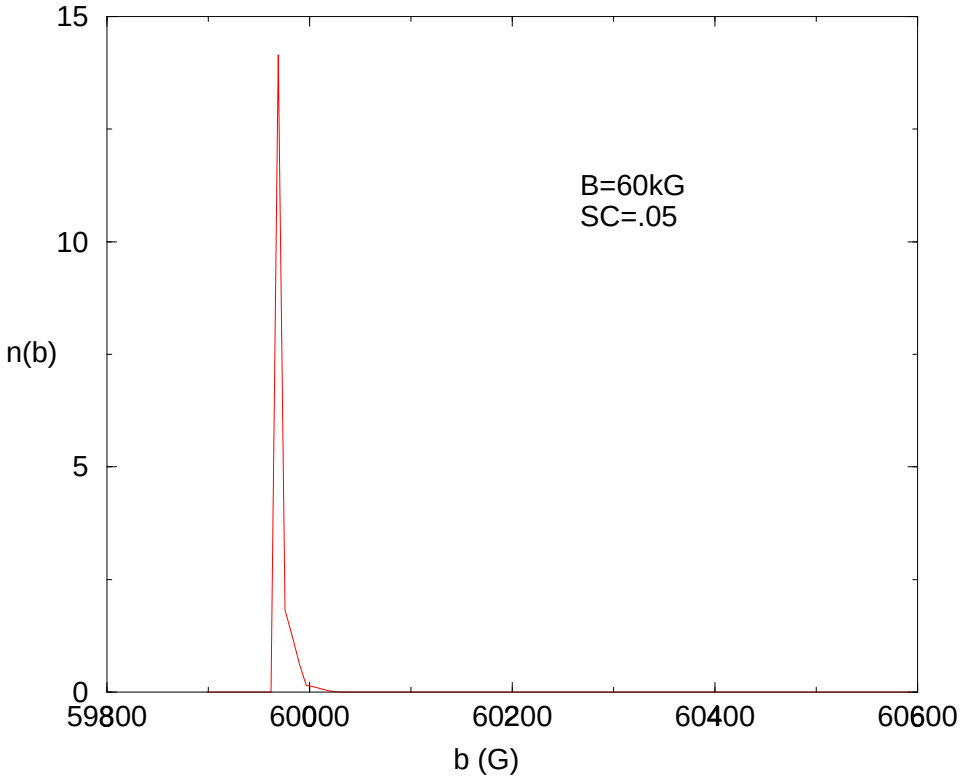

Figure 8: Probability distribution for sc=.05 and b=60kG

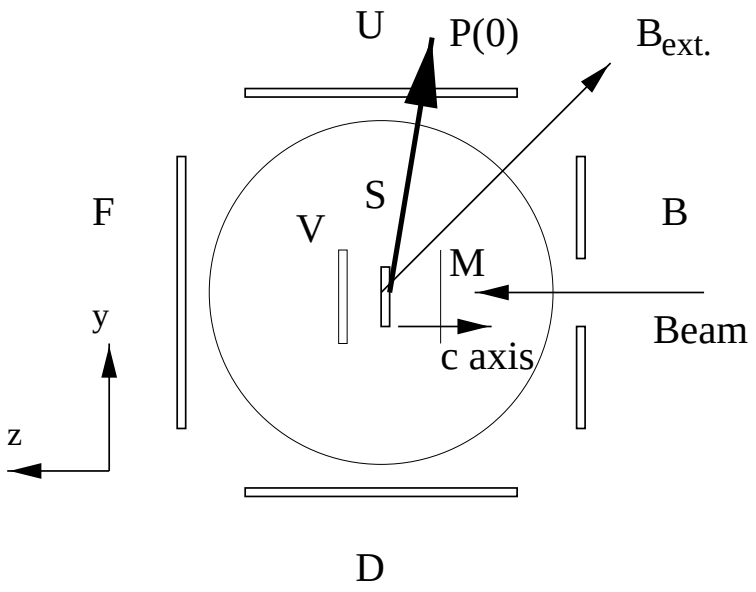

Figure 9: Schematic view of experimental setup for μSR [5]

shifted down slightly. This is believed to be the result of the average pancake position being further away from the origin with increasing disorder.

We hope these results will help us better understand the phenomenon of disorder in superconductors, but most of the conclusions have yet to be made. Some inconsistencies need to be addressed by further study, but the initial findings are promising and generally agree with the expected outcome. The theoretical calculations will eventually be used to compare with experimental data from the TRIUMF lab in British Colombia, Canada, which conducts muon spin rotation spectroscopy (μSR) experiments. In μSR, a surface beam of muons is fired at a superconducting target, and the resulting positrons are detected by the μSR apparatus (fig. 9). Since muons decay preferentially in the direction of their spin, and the spin itself is meanwhile rotating (precessing) around the direction of the local magnetic field, these studies can be used to determine the microscopic magnetic fields inside the target [5]. With the numerical models we have been building, the theory of disorder of pancake

vortices in superconductors can be tested against the experimental results to better our understanding of these materials.

Appendix: Fortran Code

```
c**********************************************************************
      subroutine hpxy(B,xlab,sc,nsh,npanxy,hpan,INDEX)
      dimension hpan(0:INDEX,0:INDEX)
      parameter (pi=3.14159265,phi0=2.07e3,s=.03081/2.)
c     s is peculiar to BSCCO.
ccccccccccccccccccccccccccccccc
      save
      z=0.
      ss=s*s
      xl=(2./s)*xlab**2
      a=sqrt(phi0*2/(B*sqrt(3.)))
      xmax=a/2.
      ymax=sqrt(3.)*a/4.
      ax1=a
      ax2=a/2
      ay2=a*sqrt(3.)/2.
      dx=xmax/index
      dy=ymax/index
      pref=phi0/(2.*pi*xl)
ccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c All the above is set up
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      print*,'Away we go!'
      do i=0,index
         x=i*dx
         do j=0,index    ! These are the x and y positions of the muon.
            y=j*dy
            do k=-npanxy,npanxy          ! Pancake positions.
               do l=-npanxy,npanxy
                  xmp0=x-(k*ax1+l*ax2)
                  xs=(xmp0)**2
                  ymp0=y-l*ay2
                  ys=(ymp0)**2
                  xpys=xs+ys
                  do m=-nsh,nsh
                     zs=(s*m)**2
                     r=sqrt(xpys+zs)
                     hpan(i,j)=hpan(i,j)+exp(-r/xlab)/r
                  enddo
```

```
         enddo
      enddo
      hpan(i,j)=pref*hpan(i,j)
   enddo
enddo
return
end
```

```
      subroutine hsumc(B,xlab,hsum,INDEX)
      parameter(phi=2.07e3,pi=3.142592654,IG=50)
      dimension hsum(0:INDEX,0:INDEX),hq(-IG:IG,-IG:IG)
      dimension qx(-IG:IG,-IG:IG),qy(-IG:IG,-IG:IG)
      complex ci
      save
      ad=sqrt(phi*2./(B*sqrt(3.)))
      xlabs=xlab**2
      tpioa=2*pi/ad
      tpioaor3=tpioa/sqrt(3.)
      pr2=tpioaor3*2
c First get the Q^2
      do i=-ig,ig
         do j=-ig,ig
            qx(i,j)=i*tpioa
            qy(i,j)=-i*tpioaor3 + pr2*j
            qs=qx(i,j)**2+qy(i,j)**2
            hq(i,j)=B/(1.+xlabs*qs)
         enddo
      enddo
      ci=(0,1)
      dx=ad/(2*INDEX)
      dy=ad*sqrt(3.)/(4.*INDEX)
      do i=0,INDEX
         do j=0,INDEX
            rx=dx*i
            ry=dy*j
            do l=-ig,ig
               do m=-ig,ig
      hsum(i,j)=hsum(i,j)+hq(l,m)*exp(ci*(rx*qx(l,m)+ry*qy(l,m)))
               enddo
            enddo
         enddo
      enddo
      end
```

```
c**********************************************************************
      subroutine hspags(B,xlab,hspag,INDEX,npanxy)
      parameter (pi=3.14159265,phi0=2.07e3)
      dimension hspag(0:INDEX,0:INDEX)
ccccccccccccccccccccccccccccccccc
      save
      z=0.
      ss=s*s
      xl=(2./s)*xlab**2
      a=sqrt(phi0*2/(B*sqrt(3.)))
      print*,'a=',a
      xmax=a/2.
      ymax=sqrt(3.)*a/4.
      ax1=a
      ax2=a/2
      ay2=a*sqrt(3.)/2.
      prefspag=phi0/(2.*pi*xlab**2)
      print*,'prefspag=',prefspag,' =?'
      dx=xmax/index
      dy=ymax/index
c Next calculate the near spaghetti.
c      npanxy=10
      do i=0,index
         x=i*dx
         do j=0,index
            y=j*dy
            hspag(i,j)=0.
            do k=-npanxy,npanxy
                  do l=-npanxy,npanxy
                        xx=x-(k*ax1+l*ax2)
                        yy=y-l*ay2
                        r=sqrt(xx*xx+yy*yy)+.0001
                        hspag(i,j)=hspag(i,j)+bessk0(r/xlab)
                  enddo
            enddo
            hspag(i,j)=prefspag*hspag(i,j)
         enddo
      enddo
      return
      end
```

```
c*****program INTEGP.f**********
      parameter(index=9,len=100)
      dimension hsum(0:INDEX,0:INDEX),fldnow(LEN),field(LEN),
     1  hsumz(0:INDEX,0:INDEX)
      nr=(index+1)**2
      open(9,file="field_array")
      do k=1, (index+1)**2
         read(9,*)i, j, hsum(i,j)
      enddo
      close(9)

      dx=1.
      dy=1.
      call integ(hsum,hsumz,index,dx,dy,fldnow,field,len,theta,nflds)
      open(9,file="testfile")
      fdndw=0.
      wt=0.
      do i=1,len
         fdndw=fdndw +field(i)*fldnow(i)
         wt=wt+fldnow(i)
         write(9,*)field(i),fldnow(i)
      enddo
      bave=fdndw/wt
      print*,'baverage= ',bave
      close(9)

      end
c**************************************************************************
      subroutine integ(hsum,hsumz,INDEX,dx,dy,fldnow,field,LEN,theta,
     1  nflds)
c...This routine will run through field values and sum for the line-shape
c...at each one.  Field values are assumed to vary linearly between successive
c...points on the grid, and also across diagonals.  Each contribution to the
c...sum is calculated by dividing the length of the field line through half
c...of the rectangle by the gradient.  Updated 2/93 for the calculation of
c...dn/db*<bperp^2/b^2> -- hence the inclusion of the hsumz grid to calc
c...bperp.  Again 5/27/93 for bx dn/db & by dn/db.
c**************************************************************************
      save
      parameter(LEM=100)
      dimension hsum(0:INDEX,0:INDEX),fldnow(LEN),field(LEN),
     1  hsumz(0:INDEX,0:INDEX)
      do i=1,len
         fldnow(i)=0.
```

```
      enddo

      dstot=sqrt(dx**2+dy**2)


c...Find max and min field values.
      do 4 i=0,INDEX
      do 4 j=0,INDEX
      if((i.eq.0).and.(j.eq.0))then
      fmax=hsum(0,0)
      fmin=hsum(0,0)
      endif
      if(abs(hsum(i,j)) .lt. fmin) fmin=hsum(i,j)
      if(abs(hsum(i,j)) .gt. fmax) fmax=hsum(i,j)
4     continue

c...Need to automatically choose the limits for the lineshape calculation.
c...Fstart is easy, one below its value is sufficient to give a first zero.
c...To find the end value, typical distributions were analyzed and it was
c...found that values from fmin up to a certain % of the range of fields held
c...all of the necesary information.  The percentage increases as the angle
c...increases.
      print*,'In integ, fmin & fmax= ',fmin,fmax,' = ?'
      read*,fmin,fmax
      itheta=nint(theta/1.571*19.)
      fstart=fmin-1.
      fend = fmax
      print*,'Min and max field values are ',fmin,fmax


change force nflds to 100 8/25/96
      nflds=LEM
      if(nflds .gt. LEN)nflds=LEN

      df=(fend-fstart)/real(nflds)
      print*,'start,end,df= ',fstart,fend,df

c...Loop over field values.
      do 10 ifield=1,nflds

      field(ifield)=fstart + (ifield-1)*df

c...Loop over the grid.
      do 20 j=0,INDEX-1
```

```
      do 30 i=0,INDEX-1

      xd=-1.
      yd=-1.
      xe=-1.
      ye=-1.

c...Check if the field is in the lower right triangle.
      if((field(ifield) .lt. amax1(hsum(i,j),hsum(i+1,j),
     1 hsum(i+1,j+1))).and.(field(ifield) .ge. amin1(hsum(i,j),
     2 hsum(i+1,j),hsum(i+1,j+1))))then

c...Calculate the gradient for the lower right triangle.
      dbdx=(hsum(i+1,j)-hsum(i,j))/dx
      dbdy=(hsum(i+1,j+1)-hsum(i+1,j))/dy
      gradb=sqrt(dbdx**2+dbdy**2)
c         print*,'dbdx,dbdy,gradb= ',dbdx,dbdy,gradb

c...Calc these for other contrib.
      dbdxz=(hsumz(i+1,j)-hsumz(i,j))/dx
      dbdyz=(hsumz(i+1,j+1)-hsumz(i+1,j))/dy
      dbdxyz=(hsumz(i+1,j+1)-hsumz(i,j))/dstot

c...Find the intersection points on the lower right triangle.
c...x-axis bottom.
      if((field(ifield) .lt. amax1(hsum(i,j),hsum(i+1,j))).and.
     1 (field(ifield) .ge. amin1(hsum(i,j),hsum(i+1,j)))) then
      xe=(field(ifield)-hsum(i,j))/dbdx
      endif

c...y-axis right.
      if((field(ifield) .lt. amax1(hsum(i+1,j),hsum(i+1,j+1))).and.
     1 (field(ifield) .ge. amin1(hsum(i+1,j),hsum(i+1,j+1))))then
      ye=(field(ifield)-hsum(i+1,j))/dbdy
      endif

c...diagonal up-right to low-left.
      if((field(ifield) .lt. amax1(hsum(i,j),hsum(i+1,j+1))).and.
     1 (field(ifield) .ge. amin1(hsum(i,j),hsum(i+1,j+1))))then
      dbdds=(hsum(i+1,j+1)-hsum(i,j))/dstot
      deltas=(field(ifield)-hsum(i,j))/dbdds
      fract=deltas/dstot
      xd=fract*dx
      yd=fract*dy
```

```
      endif

c...Only two of the above three should be satisfied.
      if((xe .gt. 0.).and.(ye .gt. 0.).and.(xd .gt. 0.)) then
      print*,'Problem at i,j= ',i,j,' lower right triangle.'
      goto 40
      endif

      if(xe .lt. 0.)then
      dlen=sqrt((dx-xd)**2 + (yd-ye)**2)
      bperp2=((hsum(i+1,j)+dbdy*ye)**2-(hsumz(i+1,j)+dbdyz*ye)**2 +
     1 (hsum(i,j)+dbdds*deltas)**2-(hsumz(i,j)+dbdxyz*deltas)**2)/2.
c      otherc=(hsumz(i+1,j)+dbdyz*ye+hsumz(i,j)+dbdxyz*deltas)/2.
c      print*,'bp2xe= ',bperp2

      elseif(ye .lt. 0.)then
      dlen=sqrt((xd-xe)**2 + yd**2)
c          print*,'dlen= ',dlen
      bperp2=((hsum(i,j)+dbdx*xe)**2-(hsumz(i,j)+dbdxz*xe)**2 +
     1 (hsum(i,j)+dbdds*deltas)**2-(hsumz(i,j)+dbdxyz*deltas)**2)/2.
c      otherc=(hsumz(i,j)+dbdxz*xe+hsumz(i,j)+dbdxyz*deltas)/2.
c      print*,'hsum(i,j),dbdx,xe,hsumz(i,j),dbdxz,dbdds,deltas,dbdxyz= ',
c     1 hsum(i,j),dbdx,xe,hsumz(i,j),dbdxz,dbdds,deltas,dbdxyz
c      print*,'bp2ye= ',bperp2

      else
      dlen=sqrt((dx-xe)**2 + ye**2)
      bperp2=((hsum(i+1,j)+dbdy*ye)**2-(hsumz(i+1,j)+dbdyz*ye)**2 +
     1 (hsum(i,j)+dbdx*xe)**2-(hsumz(i,j)+dbdxz*xe)**2)/2.
c      otherc=(hsumz(i+1,j)+dbdyz*ye+hsumz(i,j)+dbdxz*xe)/2.
c      print*,'bp2diag= ',bperp2

      endif

c...Calc the contrib. to the sum.
      cont=dlen/gradb
cj      cont=dlen/gradb*bperp2
c      cont=dlen/gradb*otherc
      fldnow(ifield)=fldnow(ifield)+cont
c         print*,'fldnow, cont= ',fldnow(ifield),cont

40    xe=-1.
      ye=-1.
c...Keep xd and yd in case are needed below.
```

```
      endif

c...Now for the upper left triangle.
      if((field(ifield) .lt. amax1(hsum(i,j),hsum(i,j+1),
     1 hsum(i+1,j+1))).and.(field(ifield) .ge. amin1(hsum(i,j),
     2 hsum(i,j+1),hsum(i+1,j+1)))) then

c...Calc gradient up here.
      dbdx=(hsum(i+1,j+1)-hsum(i,j+1))/dx
      dbdy=(hsum(i,j+1)-hsum(i,j))/dy
      gradb=sqrt(dbdx**2 + dbdy**2)
c      print*,'dbdx2,dbdy2,gradb2= ',dbdx,dbdy,gradb

c...Calc these for othercont.
      dbdxz=(hsumz(i+1,j+1)-hsumz(i,j+1))/dx
      dbdyz=(hsumz(i,j+1)-hsumz(i,j))/dy
      dbdxyz=(hsumz(i+1,j+1)-hsumz(i,j))/dstot

c...Top.
      if((field(ifield) .lt. amax1(hsum(i,j+1),hsum(i+1,j+1))).and.
     1 (field(ifield) .ge. amin1(hsum(i,j+1),hsum(i+1,j+1))))then
      xe=(field(ifield)-hsum(i,j+1))/dbdx
      endif

c...Left side.
      if((field(ifield) .lt. amax1(hsum(i,j),hsum(i,j+1))).and.
     1 (field(ifield) .ge. amin1(hsum(i,j),hsum(i,j+1))))then
      ye=(field(ifield)-hsum(i,j))/dbdy
      endif

c...Diag. done above: if there is a crossing, xd and yd are calculated.
      if((xe .gt. 0.).and.(ye .gt. 0.).and.(xd .gt. 0.))then
      print*,'Problem at i,j= ',i,j,' upper left triangle.'
      goto 30
      endif

      if(xe .lt. 0.)then
      dlen=sqrt(xd**2 + (yd-ye)**2)
      bperp2=((hsum(i,j)+dbdy*ye)**2-(hsumz(i,j)+dbdyz*ye)**2 +
     1 (hsum(i,j)+dbdds*deltas)**2-(hsumz(i,j)+dbdxyz*deltas)**2)/2.
c      otherc=(hsumz(i,j)+dbdyz*ye+hsumz(i,j)+dbdxyz*deltas)/2.
c      print*,'bp2xe2= ',bperp2
```

```
      elseif(ye .lt. 0.)then
      dlen=sqrt((xd-xe)**2 + (dy-yd)**2)
      bperp2=((hsum(i,j+1)+dbdx*xe)**2-(hsumz(i,j+1)+dbdxz*xe)**2 +
     1 (hsum(i,j)+dbdds*deltas)**2-(hsumz(i,j)+dbdxyz*deltas)**2)/2.
c      otherc=(hsumz(i,j+1)+dbdxz*xe + hsumz(i,j)+dbdxyz*deltas)/2.
c      print*,'bp2ye2= ',bperp2

      else
      dlen=sqrt(xe**2 + (dy-ye)**2)
      bperp2=((hsum(i,j+1)+dbdx*xe)**2-(hsumz(i,j+1)+dbdxz*xe)**2 +
     1 (hsum(i,j)+dbdy*ye)**2-(hsumz(i,j)+dbdyz*ye)**2)/2.
c      otherc=(hsumz(i,j+1)+dbdxz*xe + hsumz(i,j)+dbdyz*ye)/2.
c      print*,'bp2diag2= ',bperp2

      endif

c...Calc contrib. for this.
      cont=dlen/gradb
cj      cont=dlen/gradb*bperp2
c      cont=dlen/gradb*otherc
c          print*,'2, fldnow,cont= ',fldnow(ifield),cont
      fldnow(ifield)=fldnow(ifield)+cont
      endif

30    continue
20    continue
10    continue

      return
      end
```

```
c*****program LONGDIS.f********
      parameter(INDEX=9)
      dimension hsum(0:index,0:index),hpan(0:index,0:index)
      dimension hspag(0:index,0:index)
      save
      nave=1
      do i=0,index
         do j=0,index
            hsum(i,j)=0.
            hspag(i,j)=0.
            hpan(i,j)=0.
         enddo
      enddo
      B=70.
      sc=0.
      xlab=2.
      nsh=1000
      npanxy=4
      print*,'B,xlab,sc,nsh,npanxy,nave=?',B,xlab,sc,nsh,npanxy,nave
      read*,B,xlab, sc,nsh,npanxy,nave
      print*,'B,xlab,sc,nsh,npanxy,nave=',B,xlab,sc,nsh,npanxy,nave
      call hspags(B,xlab,hspag(0,0),INDEX,npanxy) ! The near direct lattice
      call hsumc(B,xlab,hsum(0,0),INDEX)
      call hpxy(B,xlab,sc,nsh,npanxy,nave,hpan(0,0),INDEX) ! Near Pancakes
      print*,'hpan=',(hpan(0,j),j=1,9)
      print*,'hspag=',(hspag(0,j),j=1,9)
      do i=0,INDEX
         do j=0,INDEX
            hsum(i,j)=hsum(i,j)+hpan(i,j)-hspag(i,j)
         enddo
      enddo
               open(9,file="field_array")
         do i=0, index
            do j=0,index
               write(9,*)i, j, hsum(i,j)
            enddo
         enddo
         close(9)
      end
```

References

[1] A. J. Greer and W. J. Kossler. *Low Magnetic Fields in Anisotropic Superconductors.* Springer Verlag, Berlin, 1995.

[2] Michael Tinkham. *Introduction to Superconductivity.* McGraw-Hill, 1995.

[3] Frank J. Owens and Jr. Charles P. Poole. *The New Superconductors.* Plenum Press, 1996.

[4] P. G. de Gennes. *Superconductivity of Metals and Alloys.* Addison Wesley, 1989.

[5] W. J. Kossler et al. Transparency of the **ab** planes of $Bi_2Sr_2CaCu_2O_{8+\delta}$. *Phys. Rev. Letters*, 80:592–595, 1998.

[6] John R. Clem. Two-dimensional vorices in a stack of thin superconducting films: A model for high-temperature superconducting multilayers. *Phys. Rev. B*, 43:7837–7836, 1991.

[7] Neil W. Ashcroft and N. David Mermin. *Solid State Physics.* W. B. Saunders Co. Philadelphia, PA, 1976.