Magnetic field distribution calculation via iterated circular current loops

Lorenzo V. Dumancas
Department of Physics, The College of Wooster, Wooster OH 44691 USA
Work done in the Magnet Science and Technology division of NHMFL, under supervision of Dr. A. Gavrilin

Abstract

It was the aim of this project to create a user-friendly program that calculates the magnetic field distribution along a path given editable parameters of a wire-wound magnet. The program was to be integrated into the NHMFL Intranet in order to aid with magnetic design. The main technique used was that of superposing a series of iterated circular current loops. Field distribution calculation is handled by a Fortran subroutine. A Java GUI was implemented in order that the program be user-friendly

Background

The ability to model magnetic field distribution along paths is important as one of the first steps in the magnetic design process.
Magnetic field distribution is calculated primarily in three ways.

Explored Calculation Methods

- Brown-Flax Method
  -Treats all winding as one unit, also known as "smearing" the winding sections.

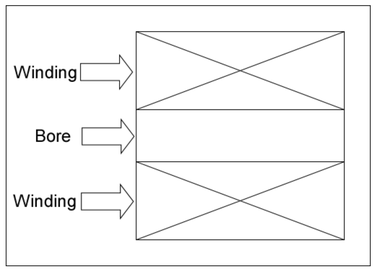

Fig. 1. Brown-Flax smeared winding sections

- Garrett's Method [1]
  -Similar to Brown-Flax Method but instead breaks winding into multiple sections.

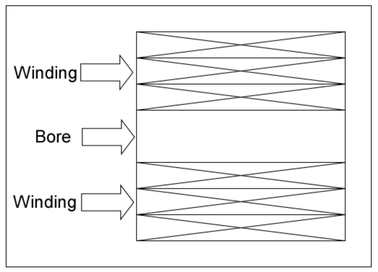

Fig. 2. Garrett Partitioned winding sections

- Superposition Method [2]
  -Calculates magnetic field using individual current filaments that comprise wire-wound magnet.
  -Primary method used in program.

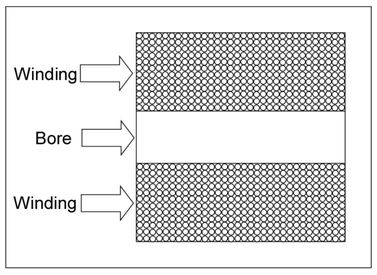

Fig. 3. Individual filaments of superposition method

Relevant Equations

- Vector potential of current filament loop given by A_φ [2].

$$A_\varphi(r,\theta) = \frac{\mu_0 Ia}{4\pi}\int_0^{2\pi} \frac{\cos\varphi'\, d\varphi'}{\sqrt{a^2+r^2-2ar\sin\theta\cos\varphi'}}$$
$$= \frac{\mu_0}{4\pi}\frac{4Ia}{\sqrt{a^2+r^2+2ar\sin\theta}}\left[\frac{(2-k^2)K(k^2)-2E(k^2)}{k^2}\right]$$

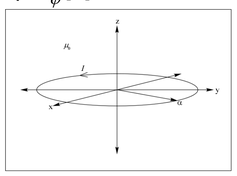

Fig. 4. Circular current filament geometry

- r, θ, and φ are the usual spherical coordinates.
- K and E are complete elliptic integrals of the first and second kind.
- The argument of the elliptic integrals k^2 is given by

$$k^2 = \frac{4ar\sin\theta}{a^2+r^2+2ar\sin\theta}.$$

- A static magnetic field generated by a constant current can be calculated via the equations below

$$B_r = \frac{1}{r\sin\theta}\frac{\partial}{\partial\theta}(\sin\theta A_\varphi),$$
$$B_\theta = -\frac{1}{r}\frac{\partial}{\partial r}(rA_\varphi),$$
$$B_\varphi = 0.$$

Program

The magnetic field calculation occurs in a Fortran subroutine written by Dr. A. Gavrilin separate from the Java GUI because this method allows for a more efficient overall program.

- Since Java is more system intensive and must run in the JVM it does not lend itself as well as Fortran does to computationally intensive numerical calculation.
- However, Java has exceptional built in tools for constructing user-friendly GUIs as well as several excellent libraries for graphing.
- In order to take advantage of Java's graphical tools while maintaining the efficiency of Fortran we simply order the Java portion of the program to call a Fortran subroutine which handles the main computation.
- Three primary methods of interoperability between Fortran and Java were explored.
  -Java and Fortran can be interfaced by using C code alongside the Java Native Interface (JNI) (outdated) [3].
  -The C code can be disregarded if instead one uses Java Native Architecture (JNA) to connect Java to a Fortran DLL file [4].
  -Both JNI and JNA were eventually set aside in favor of the Java Runtime class.
    - Allows Java to invoke a Fortran executable directly and wait for its completion.

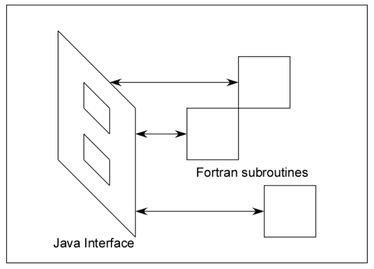

Fig. 5. Java front-end with Fortran subroutines

Fortran Code

- Below is the loop responsible for calculating the fields generated by the current filaments. Minor edits were written into the original Fortran code to provide for interoperability with Java.

```
      do 2 m=1,nop
        do 4 n=1,nos
          br=0.d0
          bz=0.d0
          notn=not(n)
            do 3 k=1,nol(n)
              rr=dble(r(n,k)/rp(m))
              rri=1.d0/rr
              rsq=dsqrt(rr)
              rm=dble(r(n,k)*rp(m))
                do 7 i=1,notn
                  zc=dble(zp(m)-z(n,i))
                  kappa=dsqrt(4d0*rm/(zc*zc+(dble(r(n,k)+rp(m)))**2))
                  if(kappa.gt.0.99999996666) kappa=0.99999996666
                  kappa2=kappa*kappa
                  kappa3=1.d0-kappa2
                  Kint=ELLK(kappa3)
                  Eint=ELLE(kappa3)
                  bz=bz+kappa/rp(m)*rsq*(rri*Kint+Eint/kappa3*
                  *(0.5d0*kappa2*(1.d0+rri)-rri))
                  br=br+kappa*zc/rm*rsq*(0.5d0*(1.d0+1.d0/kappa3)*Eint-Kint)
                continue
            continue
          mfr(n,m)=1.0d-007*cur(n)*br
          mfz(n,m)=1.0d-007*cur(n)*bz
        continue
        mfrr(m)=0.d0
        mfzz(m)=0.d0
        do 9 n=1,nos
          mfzz(m)=mfzz(m)+mfz(n,m)
          mfrr(m)=mfrr(m)+mfr(n,m)
        continue
        mf(m)=dsqrt(mfzz(m)**2+mfrr(m)**2)
      continue
```

Fig. 5. Magnetic field calculation code fragment

Java Front-End with Sample Parameters

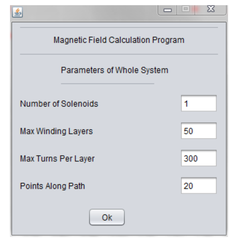

Fig. 6. Window that controls the number of solenoids as well as the number of data points to calculate

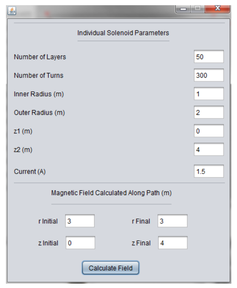

Fig. 7. Window that controls the parameters of each solenoid as well as the path along which the field is calculated

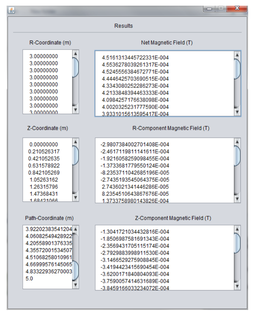

Fig. 6. Window that displays the results of the calculations

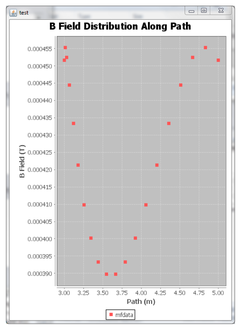

Fig. 7. Plot window, created using the jfreechart-1.0.14 and jcommon-1.0.17 libraries

- The program is capable of calculating fields for magnets of up to N solenoids.
- Currently the program is accessible within the Magnet Lab Intranet from the folder Z:\Lab-Wide Use\FieldCalc

Future Work

- Capability to run within a web browser.
- Plotting over a surface rather than just a path.
- More user options in regards to inputting data.

Acknowledgements

- I would like to thank Dr. Andrey Gavrilin and Iain Dixon for their input and assistance.
- I would also like to thank Matthew Pouliotte and Peter Jensen for their help with integrating the program with the intranet.
- This REU Program was funded by NSF Grant DMR-0654118.

References

[1] A. I. Rusinov, High Precision Computation of Solenoid Magnetic Fields by Garrett's Methods. *IEEE Transactions On Magnetics*. **30** 2685-2687 (1994)
[2] J. Simpson, J. Lane, C. Immer, and R. Youngquist, Simple Analytic Expressions for the Magnetic Field of a Circular Current Loop. NASA Technical Reports Server, accessed July 16, 2012, http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20010038494_2001057024.pdf
[3] Calling Fortran and C from Java. Csharp, accessed July 17, 2012, http://csharp.com/javacfort.html
[4] Putting a Java Interface on your C, C++, or Fortran Code. UCLA Math, accessed July 17, 2012, http://www.math.ucla.edu/~anderson/JAVAclass/JavaInterface/JavaInterface.html