LINFF: An IDL-widget program for force-free coronal magnetic fields.

Version-1: 28.03.2003
Version-2: 20.12.2005
Version-3: 29.12.2006
Version-4: 22.09.2008
Version-5: 20.12.2011

Thomas Wiegelmann (wiegelmann@mps.mpg.de)
Max-Planck-Institut für Sonnensystemforschung
Max-Planck-Strasse 2, 37191 Katlenburg-Lindau, Germany

December 20, 2011

Abstract

This documentation describes an IDL-program to compute and display linear force free magnetic fields. All basic features are controlled with help of the mouse. Version-1 of this program was written 28.03.2003 an contained the following features.:

- Tools for loading and saving FITS-files (e.g. magnetograms from MDI on SOHO) and IDL-savefiles.
- 2D-Tools to display magnetograms and to grab (active) regions.
- 3D-Tools to compute and display magnetic field lines in 3D.
- Tools to print 2D and 3D images as EPS-files and JPEG-files.

Version-2 was written 20.12.2005 and contains additional features:

- Tools to work with vectormagnetograph data and to read 3D coronal magnetic field data provided from a non-linear force-free extrapolation code.
- Tools to display coronal images from e.g. EIT.
- A program to compute full sphere potential magnetic fields from synoptic magnetograms.
- A rudimentary version of a stereoscopy program.

Version-3 was written 29.12.2006 and contains additional features:

- A multigrid-like version of the cartesian non-linear force-free optimization code and corresponding IDL-tools to prepare the data.
- A rudimentary version of a spherical non-linear force-free optimization code and corresponding IDL-tools. The code includes also the possibility to compute magnetohydrostatic equilibria (under development).

Version-4 was written 22.09.2008 and contains additional features:

- A C-version of the preprocessing routine. This is faster as the IDL-version.
- Plotting magnetic field-lines can be done now also directly on the 2D magnetogram. (Field lines are calculated in 3D and projected on magnetogram)

Version-5 was written 20.12.2011 and a chapter about SDO has been added.

We still have to describe a lot of things in more detail:

- Explain all changes done since last documentation.
- Merge previous version with modifications done by Li, Julia, Tilaye, e.g.:
    - Stereoscopy (Li)

- Vector potential (Julia)
- NLFFF spherical (Tilaye)
- New cartesian NLFFF with error matrix (Thomas)
- Many other things too.

We describe also a non-linear force-free extrapolation code written in parallelized C in this manual.

Chapter 1

Basic version

The first version of this program has been written to use line-of sight magnetograms (e.g. from SOHO/MDI, Kitt Peak) for the computation of potential and linear force-free coronal magnetic fields. This linear force-free tool gave the program it's name **LINFF** and is still the core of the software packet described here. In this chapter we describe the basic features of the core program **LINFF_ BASIC.PRO**. The file **GLOBALS.PRO** contains the global variables (IDL COMMON blocks) and the buttons are controlled with **LINFF_ BUTTONS.PRO**.

1.1 Requirements

The program is written in IDL. The program has been tested with IDL version 5.5 under Microsoft windows and IDL version 5.4 under Unix. To work with **FITS** files (e.g. from MDI on Soho *http://soi.stanford.edu/data/*) the procedures **READFITS** and **WRITEFITS** are required (e.g. from the ASTRO-IDL library *http://idlastro.gsfc.nasa.gov/contents.html*) [1]. The program also uses the program **COLORBAR** from David Fanning *http://www.dfanning.com/*. For Version-2 Solarsoft-IDL is advisable, although most features work without Solarsoft. The computation of non-linear force-free fields is done with a C-program, usually running on a different (faster, parallelized)computer than the IDL-program.

1.2 Installation and Start

The file **LINFF.PRO** is a kind of master program. The basic features of the code are collected in **LINFF _ BASIC.pro**. Start the program under IDL with **.r LINFF**. An IDL Widget opens. If some error occurs, it is sometimes helpful to compile the program again with **.com LINFF**.

[1] You can activate FITS by **.COM FITS _ TOOLS**. Note that you must do so before compiling and running the widget.

1.3 How to use the program?

The menu point **[Help → Basic Help]** provides some basic help:

1. Load magnetogram as FITS-file [File→Open... .FITS]
2. Grab an (active) region [2D-Tools→Grab Region]
3. Reduce resolution [2D-Tools→Reduce Resolution]
4. Setup 3D-Box [3D-Tools→setup 3D-box]
5. Compute magnetic field lines.
6. Use mouse to select start points.
7. Print images [Print→Print into eps-file]

Here we provide some more details regarding this points.

1.3.1 Load magnetogram as FITS-file [File→Open... .FITS]

First one needs to load a line of sight magnetogram. FITS-files (The usual format for MDI-magnetograms) are loaded with **File→Open... .FITS]**. Alternative it is also possible to restore IDL-savefiles with **[File→Restore IDL savefile]**. The IDL-savefile has the extension **.dat** or **.sav** and contains only a 2D FLOAT-ARRAY called **Bz**. The **File** menu also provides tools to save magnetograms as IDL-savefiles [File→Save Magnetogram as IDL savefile] or FITS-files [File→Save Magnetogram as FITS-file].

1.3.2 Grab an (active) region [2D-Tools→Grab Region]

The menu **2D-Tools** deals with displaying and manipulating magnetograms as 2D-data. **[2D-Tools→Bz(x,y,0)]** just displays the current magnetogram. With **[2D-Tools→Grab Region]** it is possible to grab any rectangular area from the original magnetogram. To do so one needs to click with the left mouse-button twice on the magnetogram. Once for the lower left and once for the upper right corner of the selected area [2]. The minimum and maximum in x and y (measured in pixel) are displayed in a pop-up window and it is still possible to correct these values here. With **[2D-Tools→Reduce Resolution]** the resolution of the magnetogram is reduced by a factor of two. A reduced resolution will speed up the magnetic field line calculation in 3D significantly. **[2D-Tools→Restore original magnetogram]** restores the original magnetogram.

[2] Actually one can choose any two significant corners of the area and the lower left and upper right corner are calculated automatically.

1.3.3 Setup 3D-Box [3D-Tools→setup 3D-box]

[3D-Tools→setup 3D-box] sets up a 3D-box with the magnetogram colorcoded on the bottom. To compute magnetic field lines just click on this magnetogram with the LEFT MOUSE BUTTON [3]. The postion of the cursor (in pixel) and the corresponding magnetic field strength (in Gauss) are displayed in a pop-up-window. The pop-up-window also contains a slider to choose a value of α for a force-free reconstruction (Default is a potential field reconstruction with $\alpha = 0.$). Click the RIGHT MOUSE BUTTON if you want to leave the field line computation routine [4]. It is possible to continue the field line computations with **[3D-Tools→compute field lines]**.

1.3.4 Print images [Print→Print into eps-file]

Both 2D-images and 3D-images are printed in .EPS or .JPEG format with **[Print→Print in eps-file]** or **[Print→Print in jpeg-file]** respectively. Version 2 supports also png-files. Technical the images are read out of the Z-Buffer first and then saved to the file. For EPS-files this can use to significant loose of quality. To produce high quality EPS-figures better use EPS-tools: 1.) EPS_ OPEN, 2.) create images, 3.)EPS_CLOSE

1.3.5 Some hints for advanced users

You can change some option in the file **LINFF.PRO** in *options* = ...

- **no_help=0**: The program displays **pop-up information** with some help. While this is useful for beginners, it might become annoying after a while. To switch of this features set the variable **no_help=1**.
- **maxbz=500**: Set this value to the maximum magnetic field strength (in Gauss) you want to display. The variable is used for the color coding in 2D and 3D.
- **color_map=4**: Colormap to load with IDL-routine loadct. (0 is black and white)
- **ax1=30 and az1=-30**: View angle for the 3D-graphics. Play around with this values to choose the preferred viewing angle. (This can be changed in **LINFF _ BASIC** in the routine **set_options**).
- **dt0=0.5**: Stepsize for the field line integration with a 4th order Runge-Kutta algorithm. A smaller value results in a more accurate field line computation and longer computing time. (This can be changed in **LINFF _ BASIC** in the routine **set_options**).
- **maxsteps=600**: Maximum number of integration steps for a single field line. The field line integration stops when the field line reaches a boundary of the computational box or when the number of integration steps reaches maxsteps.
- To organize the data we can specify folders for different kind of data, e.g.

[3] For high resolution magnetograms the computation can take some time. Please be patient.

[4] It is necessary to leave the field-line routine with RIGHT MOUSE BUTTON before you can do anything else like printing an image or loading a new magnetogram

- **dir _ mdi** : (full disc) magnetograms
- **dir _ syn** : synoptic magnetograms
- **dir _ eit** : coronal images from e.g. EIT, SUMER, TRACE
- **dir _ mf** : 3D magnetic fields in spherical coordinates

1.3.6 If Errors occur

If IDL displays Error messages it is often a good idea to recompile the program with **.COM LINFF**.

1.4 Physical background and basic equations

Due to the low average plasma β the structure of the corona is determined by the coronal magnetic field. Knowledge of the structure of the coronal magnetic field is therefore of prime importance to understand the physical processes in the solar corona. At the present time there is no general method available which allows the direct and accurate measurement of the magnetic field at an arbitrary point in the corona. We therefore have to extrapolate the coronal magnetic field from measurements taken at photospheric or chromospheric level. If we want to do this we have to make assumptions about the current density in the corona. The low average plasma β allows us to assume that to lowest order the magnetic field is force-free, i.e. the current density is aligned with the magnetic field. Linear force free magnetic fields have to obey the equations

$$\begin{aligned} \nabla \times \mathbf{B} &= \alpha \mathbf{B} \qquad (1.1) \\ \nabla \cdot \mathbf{B} &= 0, \qquad (1.2) \end{aligned}$$

where α is a constant [5]. We use the method of Seehafer [1978] for calculating the linear force-free field [6] for a given magnetogram (e.g. MDI on SOHO) and a given value of α. This method gives the components of the magnetic field in terms of a Fourier series. The observed magnetogram which covers a rectangular region extending from 0 to L_x in x and 0 to L_y in y is artificially extended onto a rectangular region covering $-L_x$ to L_x and $-L_y$ to L_y by taking an antisymmetric mirror image of the original magnetogram in the extended region, i.e.

$$\begin{aligned} B_z(-x, y) &= -B_z(x, y) \\ B_z(x, -y) &= -B_z(x, y). \end{aligned}$$

The expression for the magnetic field is given by

$$B_x = \sum_{m,n=1}^{\infty} \frac{C_{mn}}{\lambda_{mn}} \exp\left(-r_{mn} z\right) \cdot \left[\alpha \frac{\pi n}{L_y} \sin\left(\frac{\pi m x}{L_x}\right) \cos\left(\frac{\pi n y}{L_y}\right) - \right.$$

[5] In general α can be a function of space for (nonlinear) force free fields and has to obey $\mathbf{B} \cdot \nabla \alpha = 0$. Nonlinear force free fields need photospheric vectormagnetograms as input and the computation is way more complicated as for linear force free fields.

[6] We can also use the method to calculate potential fields by choosing $\alpha = 0$.

$$-r_{mn}\frac{\pi m}{L_x}\cos\left(\frac{\pi m x}{L_x}\right)\sin\left(\frac{\pi n y}{L_y}\right)\Bigg] \tag{1.3}$$

$$B_y = -\sum_{m,n=1}^{\infty}\frac{C_{mn}}{\lambda_{mn}}\exp\left(-r_{mn}z\right)\cdot\left[\alpha\frac{\pi m}{L_x}\cos\left(\frac{\pi m x}{L_x}\right)\sin\left(\frac{\pi n y}{L_y}\right)+\right.$$

$$\left.+r_{mn}\frac{\pi n}{L_y}\sin\left(\frac{\pi m x}{L_x}\right)\cos\left(\frac{\pi n y}{L_y}\right)\right] \tag{1.4}$$

$$B_z = \sum_{m,n=1}^{\infty}C_{mn}\exp\left(-r_{mn}z\right)\cdot\sin\left(\frac{\pi m x}{L_x}\right)\sin\left(\frac{\pi n y}{L_y}\right) \tag{1.5}$$

with $\lambda_{mn} = \pi^2(m^2/L_x^2 + n^2/L_y^2)$ and $r_{mn} = \sqrt{\lambda_{mn} - \alpha^2}$.

The coefficients C_{mn} are obtained by comparing Equation (1.5) for $z = 0$ with a FFT of the magnetogram data. The numerical method has to cut-off the Fourier series at some maximum values for m_{max} and n_{max}. The maximum value of α^2 for given L_x and L_y is

$$\alpha^2_{\text{max}} = \pi^2\left(\frac{1}{L_x^2}+\frac{1}{L_y^2}\right).$$

To normalise α we choose the harmonic mean L of L_x and L_y defined by

$$\frac{1}{L^2} = \frac{1}{2}\left(\frac{1}{L_x^2}+\frac{1}{L_y^2}\right).$$

For $L_x = L_y$ we have $L = L_x = L_y$. With this normalisation the values of α fall into the range $-\sqrt{2}\pi < \alpha < \sqrt{2}\pi$.

Magnetic field lines are calculated with help of the equation

$$\frac{\partial \mathbf{r}}{\partial \tau} = \mathbf{B} \tag{1.6}$$

and we solve this equation with help of a fourth order Runge-Kutta scheme with step-size control.

Chapter 2

Additional Tools

2.1 VECMAG

The vector magnetogram tools can be activated with: **[Add-Tools→Realize VECMAG]**. If this tool is not needed anymore it can be destroyed by **[Add-Tools→Destroy VECMAG]**. The source files are collected in **LINFF _ VECMAG**.

These tools handle vector magnetograms (B_x, B_y, B_z) on the photosphere.

- Magnetograms can be loaded with **[File→Restore IDL savefile]**. The IDL-savefile has the extension **.dat** or **.sav** and contains three 2D FLOAT-ARRAYS called **Bx, By, Bz**.

- Alternatively one can load vectormagnetograms in FITS-format with **[FITS (Tokyo)]**. This tool has been used to work with vectormagnetograms from the SFT (Solar Flare Telescope) in Tokyo. Take care: Sometimes Bx and Bz are interchanged in SFT-data and there are scaling factors involved.

- The resolution of (vector) magnetograms can be reduced by **[2D-Tools→Reduce Resolution]**.

- With **[2D-Tools→Grab Region]** it is possible to grab any rectangular area from the original magnetogram.

- Display the different magnetic field components with the buttons **Bx, By, Bz** or all together with **VecMag**.

- The button **Aly** checks how good the vector magnetogram agrees with the assumption of a force-free field, the so called Aly-criteria Aly [1989]. All values should be small compared to one for good data, but in reality they are often not.

- Vectormagnetograms are used as photospheric boundary conditions for a nonlinear force-free modelling of the coronal magnetic field as described in chapter 3 and Wiegelmann [2004]. Usually (if the Aly-criteria are not fulfilled) we have to preprocess the data, as described below and in Wiegelmann *et al.* [2006].

- **QUAD Bz** (Quick and dirty) makes the magnetogram larger by introducing a boundary-layer of nd points around the magnetogram. Specify nd in the IDL command window, e.g. $nd = 16$. Bz is homogeneous in the boundary layer and so chosen that the whole magnetogram is flux balanced. Bx and By are computed as potential fields. (It is much better to choose a vectormagnetogram which is flux balanced already. This procedure is just a quick way to deal with vector magnetograms with a limited field of view.)

- **PrePro B** preprocesses the vector magnetogram as described in Wiegelmann *et al.* [2006]. The routine takes care about noise and inconsistencies in the measured data. The coronal magnetic field is assumed to be force-free, but the photospheric magnetic field not. As result of the preprocessing the data are consistent with the force-free assumption and are as close as possible to the measured magnetogram within the measurement error. You might notice that the values in **Aly** are much smaller (better) than the original measured vector magnetograms.

- **Save2C** saves several files which are used for the computation of nonlinear force-free coronal magnetic fields in 3D as described in chapter 3. Please specify the location of these files by setting the variable *folder* in the IDL command window and specify also the extension of the box in z-direction, e.g. $nz = 64$.

- Compute the non-linear force-free magnetic field in a $nx \times ny \times nz$ box as described in chapter 3. As result you get two binary files, **B0.bin** contains the potential field and **Bout.bin** the nonlinear force-free field.

- Load **Bout.bin** into the IDL-widget with **Load B3D (BIN)** or **B3d.BIN-boundary**. The latter one omits the *nd*-pixel boundary layer. The button **Swap=0** or **Swap=1** swaps between different binary formats (usually Unix has a different BIN-format than Windows) of the BIN-files.

- One can check the quality of the reconstruction with **calc L**. This can take quite long for large boxes.

- Display a 3D-box with **3D-Box**. The Bz component of the magnetogram is color coded on the bottom of the box.

- Start the magnetic field line tracer with **f-line (Box)**. Just click on the magnetogram with the LEFT MOUSE BUTTON to compute magnetic field lines. The position of the cursor (in pixel) and the corresponding magnetic field strength of B_z (in Gauss) are displayed. [1] Click the RIGHT MOUSE BUTTON if you want to leave the field line computation routine.

There are some other buttons in the **Vecmag** tools, but they are of minor importance and usually not needed. Useful might be the option to load and save the 3D magnetic field in IDL-save-file format with **Load B3D (IDL)** and **Save B3D (IDL)**. **Global Alpha** computes an average value of α from the vector magnetogram. **Cont** shows the continuum component of the SFT data (only for data loaded with **FITS (Tokyo)**). The

[1] The pop-up window also shows a slider for alpha, but this has no function here!!

ASCII buttons load or save the magnetogram as ASCII (not properly installed yet). **Norm B** normalizes the 3D magnetic field and **Expand** increases the box resolution by a factor of two.

2.2 EIT/SUMER

The EIT/SUMER tools (which can and have been also used for TRACE) can be activated with: **[Add-Tools→Realize EIT/SUMER]**. If this tool is not needed anymore it can be destroyed by **[Add-Tools→Destroy EIT/SUMER]**. The source files are collected in **LINFF _ SUMER**.

Usually this tool is used together with the **VECMAG** tools. The **EIT/SUMER** tools are used to investigate the relationship between coronal magnetic fields and coronal plasma structures.

Usually one will first load a magnetogram and grep and (active) region as described in chapter 1 before using the plasma/image tools described here. The program has been used so far with data from SOHO and Kitt Peak (using the FITS-headers). To use other instruments (STEREO, TRACE, SOLAR-B) probably changes in the program are needed. In the following we assume that a magnetogram has been loaded and an appropriate region chosen.

- Load an image (e.g. from SOHO/EIT) with **EIT-FITS**. The image must be taken (at least approximately) to the same time and viewpoint as the (full disk) magnetogram.

- **EIT-Area** grabs the same region as the magnetogram, e.g. an active region.

- **Show EIT** displays the EIT image and **SOBEL(EIT)** applies an edge-enhancing method (Sobel-function) to the image.

- **Setup Loop _ struc** sets up a structure to store 3D magnetic loops. The last computed 3D field lines (as described in chapter 1 and 2.1) can be stored in this structure with **Add Loop**. One can remove a loop from the structure with **Remove Loop**.

- The stored loops can be save to disc with **Save Loops** and loops can be loaded with **Load Loops**.

- The 3D Loops can be shown in 3D with **Field-lines 3D** (activate **3D-Box** first) and projected onto the magnetogram with **Field-lines 2D**.

- **Field-lines** projects all stored field lines onto the EIT-image. Please press **Show EIT** or **SOBEL(EIT)** first. **B-Contours** outlines colour coded the photospheric magnetic field (Bz) superimposed to the EIT-image.

- We can compute magnetic field lines directly onto the EIT-image by **B-EIT (lin)** (linear force-free fields) or **B-EIT (nonlin)** (nonlinear force-free fields). The latter requires that the magnetic field (a 3D magnetic field not just a magnetogram) has been loaded before with **Load B3D (BIN)**, as described in chapter 2.1. In the following we describe **B-EIT (lin)**. (Let us remark that **B-EIT (lin)** works similar, but you cannot choose a value of ALPHA here, of course.)

  - Chose a value for the force-free parameter on the slider **ALPHA**. α is normalized with the average box length of the magnetogram.
  - Click **B-EIT (lin)**.
  - Move with the mouse over the image. The corresponding magnetic field strengths is displayed.
  - Click left mouse-button to compute a magnetic field line with the current cursor position as foot point. The field line becomes projected onto the EIT-image (it may take a few seconds or longer) and stored within the loop structure mentioned earlier.
  - To leave the field line tracer press the right mouse button.

- Column three contains tools for tracing features with the mouse (by hand). We hope to establish automatic feature tracking methods soon.
  - **setup hand** setups another structure to store these hand trace loops.
  - **EIT Trace** activates the tracer. By holding the left mouse button, one can follow structures on the image (red dots or a red line is shown).
  - Click right mouse button to stop the tracing. You are asked if you want to store the traced structure.
  - **Loops EIT** projects the hand traced loops onto the EIT image and **Loops MDI** onto a magnetogram and **Loops MDI-3D** on the bottom of a 3D box.
- The last column **Load P**, **Load P-b**, **P** has been created to deal with the plasma pressure. We have used this until now only for test data and a further development is needed.

Some progress in automatic feature recognition with the help of magnetic field information has been done in Wiegelmann *et al.* [2005a] and some software exists in the file *LINFF _ SUMER*, but the programs not controlled with mouse.

2.3 AL-Loops

This tool has been created mainly for a project of comparing observed loops in the chromosphere with extrapolated field lines as published in Wiegelmann *et al.* [2005b] and is probably not needed by other users. The source files are collected in **LINFF _ LOOPAL**.

2.4 Sphere and CH-Tools

The tool can be activated with: **[Add-Tools→Realize Sphere+CH-Tools]**. If this tool is not needed anymore it can be destroyed by **[Add-Tools→Destroy Sphere+CH-Tools]**. The source files are collected in **LINFF _ CH**.

The spherical tools are used to compute a global source surface potential field from synoptic magnetic field data. The CH-tools (CH coronal hole) have been mainly created for statistical investigations of coronal holes as published in Wiegelmann and Solanki [2004]. The data used for this project can be loaded with **[Projects→Load Project Coronal Holes]**, but probably nobody else wants to use this special data set again.

2.4.1 Basic equations

We have to solve:

$$(\nabla \times \vec{B}) = 0 \tag{2.1}$$

$$\nabla \cdot \vec{B} = 0 \tag{2.2}$$

We represent the magnetic field with help of a scalar potential:

$$\vec{B} = \nabla \Phi \tag{2.3}$$

$$\Delta \Phi = 0 \tag{2.4}$$

In spherical (r, θ, ϕ) coordinates (2.4) has the solution:

$$\Phi(r, \theta, \phi) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} \left[A_{lm}\ r^{l} + B_{lm}\ r^{-(l+1)} \right]\ Y_{lm}(\theta, \phi) \tag{2.5}$$

where Y_{lm} are Spherical Harmonics and A_{lm} and B_{lm} are coefficient which we get from boundary conditions.

2.4.2 Short remark

Any function $g(\theta, \phi)$ can be represented as

$$g(\theta, \phi) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} C_{lm} Y_{lm}(\theta, \phi) \tag{2.6}$$

$$C_{lm} = \int_0^{2\pi} \int_0^{\pi} Y^*_{lm}(\theta, \phi)\ g(\theta, \phi)\ \sin(\theta)\, d\theta d\phi \tag{2.7}$$

where $Y^*_{lm} = (-1)^m\ Y_{l,-m}$.

2.4.3 Boundary conditions

On the photosphere we prescribe von Neumann boundary conditions $B_r(r_0, \theta, \phi) = \frac{\partial \Phi}{\partial r}$ and we apply (2.7) to calculate C_{lm} for $g(\theta, \phi) = B_r(r_0, \theta, \phi)$ where we get $B_r(r_0, \theta, \phi)$ from synoptic MDI-data. The full radial magnetic field is given by:

$$B_r(r, \theta, \phi) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} \left[A_{lm}\ l\, r^{(l-1)} - B_{lm}\ (l+1)\ r^{-(l+2)} \right]\ Y_{lm}(\theta, \phi) \tag{2.8}$$

The values of A_{lm} and B_{lm} are not completely determined with C_{lm}, but additional boundary conditions have to be imposed. Here we only discuss two possibilities.

Boundary condition I: $B_r(r \to \infty) = 0$

This condition leads to $A_{lm} = 0$ for $l > 0$ and

$$B_{lm} = -\frac{C_{lm}\, r_0^{l+2}}{(l+1)}$$

A_{00} can be chosen completely free and gives only a constant to Φ which does not influence the magnetic field.

Boundary condition II: $\vec{B} = B_r\, \vec{e}_r$ **for** $r \geq r_1$

These boundary conditions correspond to the *Source surface* model, which is a kind of standard model for global potential fields. Here

$$B_\theta = \frac{1}{r} \frac{\partial \Phi(r, \theta, \phi)}{\partial \theta} \tag{2.9}$$

$$B_\phi = \frac{1}{r\, \sin(\theta)} \frac{\partial \Phi(r, \theta, \phi)}{\partial \phi} \tag{2.10}$$

are supposed to vanish for $r \geq r_1$ Consequently the potential only depends on the radial component for $r \geq r_1$, $\Phi = \Phi(r)$ and all coefficients of Y_{lm} (but Y_{00}) have to vanish here. Together with the photospheric boundary condition (2.7) we get two equation to calculate A_{lm} and B_{lm} for $l \geq 1$:

$$A_{lm}\, l\, r_0^{(l-1)} - B_{lm}\, (l+1)\, r_0^{-(l+2)} = C_{lm} \tag{2.11}$$

$$A_{lm}\, r_1^{l} + B_{lm}\, r_1^{-(l+1)} = 0 \tag{2.12}$$

which leads to:

$$A_{lm} = \frac{C_{lm}\, {r_0}^{2+l}}{{r_1}^{1+2\,l} + l\, ({r_0}^{1+2\,l} + {r_1}^{1+2\,l})} \tag{2.13}$$

$$B_{lm} = -\left(\frac{C_{lm}\, {r_0}^{2+l}\, {r_1}^{1+2\,l}}{{r_1}^{1+2\,l} + l\, ({r_0}^{1+2\,l} + {r_1}^{1+2\,l})}\right) \tag{2.14}$$

The distance of the source surface sphere r_1 might be chosen in a way that we can fit some additional constrains, e.g. observations of radiating loops or helmet streamer. We might compare the magnetic field and observed plasma structures similar as for the cartesian linear force free case and minimize with respect to r_1 here.

2.4.4 Remarks

All components of $\vec{B}$ can be calculated analytically from Φ. IDL has spherical harmonics as built in functions. This make programming easy, but IDL is quite slow. For practical reason the code might be rewritten in C, which is much faster and allows parallelization. Actually we need to write a program to calculate spherical harmonics in C first and teach C to handle complex numbers. Both is straight forward and just needs some time.

In principle we can compute nonlinear force-free fields, similar as with our cartesian nonlinear force-free optimization (see chapter 3 and Wiegelmann [2004]. An extension to spherical coordinates might be straight forward. For longer structures, e.g. helmet streamer, forces have to be included because $\beta \approx 1$ in the higher corona. This has to be implemented.

2.4.5 How to use the program?

- Load Synoptic magnetogram with **Load Syn**.
- Compute global 3D magnetic field with **calc new B**. This can take several hours depending on the resolution of the 3D box in (R, Θ, Φ). The resolution can be changed in the file *LINFF _ CH.PRO* in the procedure *calc _ B*. Default is $nt = 37$ and $nr = 26$. One can also change the number of spherical harmonics in the procedure *Calc _ Clm*. Default is $maxl = 12$. As higher *maxl* as longer the computation takes and as better fine structures are resolved.
- The final 3D magnetic field is saved in the file *B1.mf*.
- The Sun can be visualized in 3D by **Create Sun** . Choose an appropriate viewing angle from the slider in the range $0 \ldots 360$. The colour coding shows the magnetic field strength on the solar surface.
- Compute a magnetic field line with **calc B-line**. Θ, Φ and the magnetic field strength on the solar surface are displayed. Click left mouse button to compute a magnetic field-line. Click right mouse button to leave the program.
- The button **Field lines** computes many field lines.

2.4.6 New features in Version 3

The program has been extended and is now also used to display nonlinear force-free (or even non-force-free) magnetic fields in spherical geometry. The corresponding tools are collected in **sphere _ ff _ tools**. The 3D magnetic field can be stored and loaded in **.sbin** files and the field lines are displayed as described above. The nonlinear force-free code in spherical geometry is described in Wiegelmann [2007]. The code has been tested with the help of a semi-analytic solution, but not been applied to real data until now.

2.5 STEREO Tools

The tools can be activated with: **[Add-Tools→Realize STEREO Tools**. If this tool is not needed anymore it can be destroyed by **[Add-Tools→Destroy STEREO Tools]**. The source files are collected in **STEREO _ EVENTS**. Development of this tool is still in progress. A manual will be written later. The basic idea of magnetic stereoscopy has been published in Wiegelmann and Neukirch [2006] and applied to test cases. Li Feng applied the tools to real data from SOHO, TRACE and STEREO/SECCHI and added a lot of image processing features, which require Solar-Soft-IDL. The new tools have still to be included in this widget and be described in this manual.

Chapter 3

Non-linear force-free magnetic fields: optimization code.

This program is written in C and based on an optimization principle. The program is not part of the IDL-widget programm packet, but the IDL-tools are used for preprocessing of vectormagnetograph data (which are then used for the computation of non-linear force-free fields in the C-program described here) and for the analysis of the reconstructed 3D magnetic field. The interface between IDL and C is done via ASCII-files and Binary-files, as the C-Programm will usually run on a different (faster) computer as IDL.

3.1 Theory

Force-free coronal magnetic fields have to obey the equations

$$\begin{aligned} (\nabla \times \mathbf{B}) \times \mathbf{B} &= \mathbf{0}, && (3.1)\\ \nabla \cdot \mathbf{B} &= 0. && (3.2) \end{aligned}$$

We define the functional

$$L = \int_V \left[w_a(x,y,z)\, B^{-2}\, |(\nabla \times \mathbf{B}) \times \mathbf{B}|^2 + w_b(x,y,z)\, |\nabla \cdot \mathbf{B}|^2 \right] d^3x, \qquad (3.3)$$

where w_a and w_b are weighting functions. It is obvious that (for $w_a, w_b > 0$) the force-free equations (3.1-3.2) are fulfilled when L is equal zero. We minimize the functional (3.3) with an iterative scheme:

$$\frac{\partial \mathbf{B}}{\partial t} = \mu \tilde{\mathbf{F}}, \qquad (3.4)$$

which (for $\mu > 0$) ensures that L is monotonically decreasing. See Wiegelmann [2004] for the definition of $\tilde{\mathbf{F}}$ for the case of one weighting function ($w_a = w_b = w$). The method has been developed by Wheatland *et al.* [2000] for the case without weighting function. Our code reduces to this approach for the choice $w_a = w_b \equiv 1$ in the entire box. We use this choice if the boundary conditions for all six boundaries of a computational box are given (e.g. Test Case I and Wiegelmann and Neukirch [2003]). The weighting functions are useful if only the bottom boundary data are known. In this case we introduce a buffer boundary of several grid points towards the lateral and top boundary of the computational box. The weighting functions are chosen constant in the inner, physical domain and drop to 0 with a cosinus-profile in the buffer boundary towards the lateral and top boundary of the computational box (see Wiegelmann [2004] for details). In most cases the weighting functions w_a and w_b are chosen identically ($w_a = w_b = w$), which gives the same weighting to the force-free and solenoidal condition [1].

The method works as follows:

1. Compute start equilibrium (e.g. a potential field) in the computational box.
2. Replace the bottom boundary (or all six boundaries) with the vector magnetogramm.
3. Minimize the functional (3.3) with the help of Eq. (3.4). The continuous form of (3.4) guaranties a monotonically decreasing L. This is as well ensured in the discretized form if the iteration step dt is sufficiently small. The code checks if $L(t + dt) < L(t)$ after each time step. If the condition is not fulfilled, the iteration step is repeated with dt reduced by a factor of 2. After each successful iteration step we increase dt slowly by a factor of 1.01 to allow the time step to become as large as possible with respect to the stability condition.
4. The iteration stops if L becomes stationary. Stationarity is assumed if $\frac{\partial L}{\partial t}/L < 1.0 \cdot 10^{-4}$ for 100 consecutive iteration steps.

The program is written in C and has been parallelized with OpenMP. The bottom (or any other) boundary is not changed during the iteration. The code has been applied to vector magnetograph data from the German Vacuum Tower Telescope (VTT) in Wiegelmann *et al.* [2005b].

3.2 How to use the code

- Compile the program with

  make -f Makefile.helios

  Maybe you need to modify the Makefile. On UNIX system you might get a lot of warning messages about *invalid white space character in directive,* which you can ignore. It just happens, because I wrote the source code on a Windows computer.

- The code needs two files as input:

[1] We did some tests for $w_a \neq w_b$.

1. **grid.ini**
2. **allboundaries.dat**

The files have to be consistent which each other. **grid.ini** contains the dimensions of the 3D computational box (nx,ny,nz) and the size of the boundary layer nd. The variable mu is not used anymore. It has previous been used to specify the integration time step, but this is now computed automatically. **allboundaries.dat** contains the boundary data, usually the measured and preprocessed photospheric magnetic field vector, but the file can additional contain also the magnetic field vector on the other five boundaries of the computational box. This is useful in particular for performance test. (For real magnetograms we have of course only the photospheric data.)

- The code has several options to run. The name of the code is **relax2** and you run it as

  relax2 param1 param2
  where param1 is a number, which specifies special options of the code and param2 the maximum number of iteration steps. It is save to give here a high number, e.g. $param2 = 10000$. [2] The main options for param1 are:

  - **relax2 22 10000** : Compute first a potential field (from B_z in **allboundaries.dat**) and save it as **B0.bin**. Then replace the photospheric boundary with the data of **allboundaries.dat** and minimize Eq. 3.3. The iteration stops if $\frac{\partial L}{\partial t}/L < 1.0 \cdot 10^{-4}$ for 100 consecutive iteration steps or if the iteration time steps becomes too small or if the maximum number of iteration steps as defined in **param2** is reached. The final, non-linear force-free field is saved in **Bout.bin**. **Bout.bin** and **B0.bin** have the same structure, but **B0.bin** just contains the potential field. (param2=22) is the standard option.
  - **relax2 23 0** : Compute a potential field (from B_z in **allboundaries.dat**) and save it as **B0.bin**.
  - **relax2 20 10000** : **B0.bin** must exist!! Replace the photospheric boundary with the data of **allboundaries.dat** and minimize Eq. 3.3. **relax2 23 0** and after **relax2 20 10000** does exactly the same as **relax2 22 10000** .
  - **relax2 0 10000**: **B0.bin** must exist!! Replace all 6 boundaries of the computational box with the data of **allboundaries.dat** and minimize Eq. 3.3. This option is NOT possible for real vector magnetograph data, because here the lateral and top boundaries are unknown.

- The most important output of the code is **Bout.bin**, which contains the final non-linear force-free magnetic field. The code also outputs **step.log** (it contains only a single number) and **prot.log**, which contains (for every 10 iteration steps) t

[2] The weighting functions (w_a, w_b) are calculated in the code. You can, however, also read these functions from the binary-files **wa.bin, wb.bin** by choosing $param2 = 987654$.

(iteration step), L (Eq.3.3), L_i (like L, but excluding the boundary layer of *nd* points) and $\delta L/L$. The values are also written on the screen.

- You can use the IDL program **read_relax.pro** to read **Bout.bin**, **step.log** and **prot.log**. You need to specify in the program the folder where these files are. Please note that the file **grid.ini** must exist in the same folder. The program plots also the evolution of L in logarithmic scale. The final 3D magnetic field components are written into the IDL-variables *b3dx,b3dy,b3dz,b3dabs*. The size of the fields is nx, ny, nz as specified in **grid.ini**. The photospheric magnetic field data are written into the variables *bx,by,bz*, which have the dimension nx, ny and agree with **allboundaries.dat**.

3.3 How to start, step by step

3.3.1 Example I

1. Compile the program with **make -f Makefile.helios** or with **make -f Makefile.stereo**. You might need to modify the Makefiles on other computers or compilers.
2. cp grid_nlfff1.ini grid.ini
3. cp allboundaries_nlfff1.dat allboundaries.dat
4. **relax2 23 0 ⇒ B0.bin**
5. **relax2 0 10000 ⇒ Bout.bin, step.log, prot.log**
6. (In IDL:) **read_relax3.pro**

This corresponds to Case I in Schrijver *et al.* [2006].

3.3.2 Example II

1. cp grid_nlfff2.ini grid.ini
2. cp allboundaries_nlfff2.dat allboundaries.dat
3. **relax2 22 10000 ⇒ B0.bin,Bout.bin, step.log, prot.log**
4. (In IDL:) **read_relax3.pro**

This corresponds to Case II in Schrijver *et al.* [2006]. In both cases you can display magnetic field lines in IDL with:

- Click button **Load B3D (IDL)**
  and choose **Bout.sav** to read the 3D-NLFFF file into the widget.
- Click button **3D-Box** to create a 3D-view and **f-line (Box)** to compute 3D-field lines by clicking onto the magnetogram for loop-footpoints.

3.4 Preprocessing

The preprocessing routine has been described in detail in Wiegelmann *et al.* [2006].

The reason why preprocessing of measured photospheric vector magnetograms is necessary is the following. A problem is that measurements of the photospheric magnetic vector field contain inconsistencies and noise. In particular the transversal components (say Bx and By) of current vector magnetographs have their uncertainties. Furthermore the magnetic field in the photosphere is not necessary force-free and often not consistent with the assumption of a force-free field above. We develop a preprocessing procedure to drive the observed non force-free data towards suitable boundary conditions for a force-free extrapolation.

To do so we minimize a 2D functional:

$$L = \mu_1 L_1 + \mu_2 L_2 + \mu_3 L_3 + \mu_4 L_4 \tag{3.5}$$

where

$$\begin{aligned}
L_1 &= \left[\left(\sum_p B_x B_z\right)^2 + \left(\sum_p B_y B_z\right)^2 + \left(\sum_p B_z^2 - B_x^2 - B_y^2\right)^2\right] \\
L_2 &= \left[\left(\sum_p x\left(B_z^2 - B_x^2 - B_y^2\right)\right)^2 + \left(\sum_p y\left(B_z^2 - B_x^2 - B_y^2\right)\right)^2\right. \\
&\quad \left. + \left(\sum_p y B_x B_z - x B_y B_z\right)^2\right] \\
L_3 &= \left[\sum_p (B_x - B_{xobs})^2 + \sum_p \left(B_y - B_{yobs}\right)^2 + \sum_p (B_z - B_{zobs})^2\right] \\
L_4 &= \left[\sum_p (\Delta B_x)^2 + \left(\Delta B_y\right)^2 + (\Delta B_z)^2\right]
\end{aligned} \tag{3.6}$$

The surface integrals are here replaced by a summation $\sum_p$ over all grid nodes p of the bottom surface grid and the differentiation in the smoothing term is achieved by the usual 5-point stencil for the 2D-Laplace operator. Each constraint L_n is weighted by a yet undetermined factor μ_n. The first term (n=1) corresponds to the force-balance condition, the next (n=2) to the torque-free condition. The following term (n=3) ensures that the optimized boundary condition agrees with the measured photospheric data and the last terms (n=4) controls the smoothing. The 2D-Laplace operator is designated by Δ.

The aim of our preprocessing procedure is to minimize L so that all terms L_n if possible are made small simultaneously. A strategy on how to find the optimal yet undefined parameters μ_n is described in Wiegelmann *et al.* [2006].

As result of the preprocessing we get a data set which is consistent with the assumption of a force-free magnetic field in the corona but also as close as possible to the measured data within the noise level. [3]

[3] For application to data the optimal parameter set $\mu_1 \ldots \mu_4$ has to be found. It is usually save to use

3.5 Multigrid-like version of the optimization code

- The method is not full multigrid, but computes the solution on different grids only once, e.g., something like $50^3,\ 100^3,\ 200^3$.
- The main idea is to get a better (than potential field) start equilibrium on the full resolution box.
- Solution of smaller grids are interpolated onto larger grids as initial state for the magnetic field in the computational domain of the next larger box.
- Preprocessing is possible on each grid. This is necessary for application to real vectormagnetogram data, which contain noise and inconsistencies.

3.5.1 How to use the multigrid-like code?

The multigrid-like code uses first IDL to deal (and preprocess) the data and then (in C) the code as described above, a rebin-procedure called **rebin.c** and a shell script called **multigrid**.

- Load a vectormagnetogram as described above.
- Take care that the size in x and y is consistent with the intended multigrid level (*nx* and *ny* must be integer multiples of 4 for 3-level multigrid.)
- Click on button **Multigrid**.
- Several inputs are required now in the main IDL-window:
  - Maximum height *nz* in pixel.
  - Size of the boundary layer *nd*. Something like 32 is a good choice for large ($nx, ny > 200$) magnetograms. Please note that *nz* and *nd* must be multiples of 4 for 3-level multigrid.
  - Number of multigrid levels. Currently 3 is a kind of standard.
  - Preprocessing of the data on each grid (yes or no). For real data it is important to choose **Y** to get meaningful results. The preprocessing deals with noise and inconsistencies of the magnetogram.
  - Finally confirm that you really want to prepare the data for output in the shown folder.
- As output one gets several (3 for 3-level multigrid) files called **grid1-3.ini** and **allboundaries1-3.ini**. The preprocessing can take several minutes, however. The files contain the grid and input data for the nonlinear force-free code.
- Now use the command shell:

$\mu_1 = \mu_2 = 1$. For Aads-case (see Example-III) and data from the Solar-Flare-Telescope we used $\mu_3 = 0.001$ and $\mu_4 = 0.01$. Other instruments require other parameters, in particular these parameter-set is NOT optimal for SOLIS-data (μ_4 should be higher).

- **multigrid 3** starts the code. The code runs several times on different grids as explained earlier.
- Back in IDL: load the computed nonlinear force-free field called **Bout.bin** with the button **Load B3D (BIN)** or **B3D.BIN-boundary**. The latter choice does not include the *nd* pixel boundary layer.
- Alternative: Use the IDL-program **read_relax3** to convert *Bout.bin* to the IDL sav-file *Bout.sav*. You are asked if you want to include the boundary layer. If you choose *N* the structure in *Bout.sav* contains only the center region.

3.5.2 Example III (Multigrid+Preprocessing-example): Aads case

This might be a good test case for testing the code including multigrid and the preprocessing routine. By replacing the artificial magnetogram with a real vectormagnetogram this example can be used for application to data. Here we use the multigrid like version of the code. As example we describe Aads case [4].

1. Compile the program with **make -f Makefile.helios** or with **make -f Makefile.stereo** [5]
2. Compile **rebin.c** with **cc -o rebin rebin.c -lm** or **gcc -o rebin rebin.c -lm**
3. (In IDL:) **[Add-Tools→Realize VECMAG]**
4. (In IDL:) **[File→Restore IDL savefile]** : Choose the file *test_mag_aad.sav*.
5. (The buttons **Bx, By, Bz** display the components of the vector magnetogram. With (In IDL:) **Aly** one can check how well force-free consistency criteria are fulfilled. This step is not necessary, but it might be helful to look at the magnetogram.)
6. (In IDL:) Click button **Multigrid**. Answer the questions in the IDL-text window:

   - Input max height nz: **256**
   - Input boundary layer nd: **32**
   - Input number of Multi grids (2-4): **3**
   - Preprocessing on each grid y or n? **y**
   - Start Multigrid y or n? **y**

   This creates six files grid123.ini and allboundaries123.dat, which are used by the NLFFF-C-code. via the Shell-script **multigrd**
7. (Under Unix:) multigrid 3
   This executes the NLFFF-code several times on 3 grids and takes a while.

[4] Aads case refers to a force-free equilibrium computed by Aad van Ballegooijen, which has been used as a recent test-case within the NLFFF-consortium (Metcalf *et al.* [2008]).

[5] It might be necessary to adjust the makefile to other computers or compilers.

8. (In IDL:) **.r read_relax3**
   This creates an IDL-sav-file called **Bout.sav** which contains the NLFFF- 3D magnetic field (One has to decide if the *nd* boundary layer should be included or not).

9. Click button **Load B3D (IDL)**
   and choose **Bout.sav** to read the 3D-NLFFF file into the widget. It is also possible to load the C-file **Bout.bin** directly with **Load B3D (BIN)** or **B3D.BIN-boundary**. The latter removes the boundary layer of *nd* grid-points.

10. Click button **3D-Box** to create a 3D-view and **f-line (Box)** to compute field lines (one has to click onto the magnetogram to start field-line integration on particular footpoints.). Since *Version-4* one can use the filed-line plotting via **f-line (Box)** also on the 2D-views.

3.5.3 Preprocessing in C and Hα-preprocessing

(New since Version-4)

The preprocessing code (originally written in IDL) has been written in C for this aim and is contained in the file **prepro.c**. This can be used together with the multigrid-like nlfff-code:

- Compile the preprocessing code with: **cc -o prepro prepro.c -lm**

- (In IDL): Click button **Multigrid**
  choose **n** when asked *Preprocessing on each grid y or n?*

- (under Unix): **multiprepro 3**
  Preprocesses the magnetogram on all grids.

- **multigrid 3** computes the 3D-NLFFF field.

The C-code is faster as the IDL-code and the result might me somewhat different for the following reasons: In IDL we gave a maximum time step of 2000 (to limit computing time), but in C we allow to compute until the functional is minimized. The C-code has an automatic step-size control and the IDL code not. The parameters μ_3 and μ_4 have to be specified in the file **multiprepro**. Please note that one has to find the optimal parameters by an experimental study for each new instrument separately. The default values are used for the SFT-vector magnetograms. SOLIS and SDO/HMI will require other paramters.

The preprocessing might contain an additional term, the angle of the photospheric magnetic field vector with chromospheric Hα-images. First tests with artificial data look promising Wiegelmann *et al.* [2008], but more tests with data are necessary before this tool can be used as standard software.

3.6 Spherical implementation of the Optimization code

We can as well compute nonlinear force-free fields in spherical geometry. First tests have been published in Wiegelmann [2007]. Some IDL-tools are collected in **sphere_ff_tools**. The code is planned for use with synoptic vector magnetograms, which we still not have (SOLIS is assumed to provide them). An extension of the code towards the inclusion of plasma pressure and gravity is planned. The code has been applied only to test-data until now.

3.6.1 How does the spherical code work?

- The code contains the files: **spheremhs.c, sopt.c, sbfield.c, sglobals.h**.
- Compile the code with **Make**.
- in IDL: Create a semi-analytic test case, e.g., with **sphere_low2.pro**.
- That program creates the files **BTN.sbin** and **sgrid.ini**
- Compute a corresponding potential (for a potential field) with **Calc_Clm, bz** and **sphere_pot** which outputs the file **u.sbin** containing the spherical potential.
- In Command shell: **spheremhs 1 0** which computes a potential field as the gradient of the potential **u.sbin** and save the result in **B0.sbin**.
- Specify which kind of boundary conditions (case 1 to case 4) as published in Wiegelmann [2007]) you want to use in **sparam.ini**.
- **sparam.ini** contains also other variables, which are planned for use in the magnetohydrostatic version of the code. They should be all equal 0 here.
- Only for case 1: specify in **sgrid.ini** how close you go to the poles with the variable *nd*.
- One can test, how could the analytic equilibrium **BTN.sbin** is with the command: **spheremhs**
- Compute a spherical nonlinear force-free field with: **spheremhs 10 50000**.
- The resulting field is stored in **Bout.sbin**. Load this file into IDL with the button **Load sbin**.
- **sprot.log** contains some information about the optimzation process. One can display these information in IDL with **read_sphere3**.

3.6.2 Computing spherical magnetohydrostatic equilibria.

We made first tests of including pressure and gravity into the code. First one needs to compute a semi-analytic MHS-equilibrium, e.g. the one described in Wiegelmann and Inhester [2006], which contains plasma pressure but no gravity. This can be done with the IDL-program **sphere _ mhd**. The program outputs the magnetic field as described above and additional the plasma pressure in **Ndensity.sbin**. The pressure is loaded in the code if the variable $mhs = 1$ in the file **sparam.ini**. An extension of the code to compute magnet-hydrostatic-equilibria in spherical geometry has been done in Wiegelmann *et al.* [2007]. The spherical codes have been applied so far only to test-cases with semi-analytic equilibria. *This has been changed by now, December 2011, and Tilaye Tadesse applied the spherical NLFFF-code to SOLIS. As future work on NLFFF is mainly devoted to SDO/HMI we describe the newest version of this code in the SDO-section.*

3.7 SDO-Tools

Since the first vector magnetograms have become available to us since August 2011 we started to shape our Tools for application to SDO. This affects IDL-widget-tool as well as the cartesian and spherical NLFFF-codes. Most tools described here have been developed and tested in preparation of our first SDO/HMI paper Wiegelmann *et al.* [2012].

The tools can be activated with: **[Add-Tools→Realize SDO Tools**. If this tool is not needed anymore it can be destroyed by **[Add-Tools→Destroy SDO Tools]**. The source files are collected in **sdo _ tools**. Some auxiliary programs have been developed, too, e.g. **figure _ tools _ sdo** and **new _ mag**. Some programs for spherical NLFFF with SDO are collected in **sdo _ sphere _ tools** and **LINFF _ PARTS**.

We use Widget _ Combobox for multi-functional buttons. In the left column (column 1) we created buttons to load and save different file formats:

Load files

- **[Load SDO/HMI+AIA]:** Load SDO/AIA-images in all 8 wavelength and SDO/HMI line-of-sight magnetograms. These data-sets are usually downloaded and preprocessed via Solar-Soft as described in the SDO data analysis guide [DeRosa & Slater 2011, http://www.lmsal.com/sdouserguide.html]. We prepared the program **sdo _ example3** for easy download of SDO-data.

- **[Load AR vecmag] and [Load FD vecmag]:** Read FITS-files for HMI-vector-magnetograms for active regions and the full disk, respectively. **Only for uncompressed fits so far. To do: write program to deal with compressed fits-files.**

- **[Load B3D(BIN)]:** Loads the result of cartesian NLFFF-code. Program is identical as described in section 2.1.

- **[Load Sphere-3D psbin]:** Loads the result of spherical NLFFF-code. This tool is still under development.
- **[Load SDO _ INFO]:** To load complex data-sets. One has to create the corresponding *.infosav* files, under development.
- Some more buttons exist, but I to lazy to describe everything.

Save files

In principle similar as loading-files, but we not support all file-formats, as we probably not need to save magnetograms as fits (can add this later if needed). This section saves also files to the disk (as specified in folder **options.dir NLFFF**) which are used by the NLFFF-codes written in C:

- **[NLFFF Standard]:** Saves all files the cartesian NLFFF-code needs into the folder defined in **options.dir _ nlfff**. The standard parameter set corresponds to case E as described in Wiegelmann *et al.* [2012].
- **[NLFFF Advanced]:** This saves the same files as the previous, but one has to answer a number of questions regarding vertical box-size, boundary-layer, number of multi-scale, grids, Lagrangian parameter and mask-function. Some variations of parameters have been studied in Table 1 in Wiegelmann *et al.* [2012].
- **[Sp-NLFFF Standard]:** and **[Sp-NLFFF Advanced]:** store all files necessary for the spherical NLFFF-code, either a standard-set for the full disk or a set after answering many questions (one question is if full disk or only a selected active region should be computed). **still under development**

Buttons in Column 2-4

- Buttons in column 2 are to show full disk images from SDO/AIA and HMI, to select a particular area (for example an active region) and tools to aline HMI-active-region vector magnetograms with full disk AIA/HMI with a correlation analysis **[Align HMI, Bz]**.
- Columns 3 and 4 contain buttons to show the components of the photospheric magnetic field vector and AIA-images for the selected FOV. It is also possible to overplot both. Column 5 is used for computing magnetic field lines. One has to load the 3D-NLFFF-field first with **[Load B3D(BIN)]**. These file is created by the NLFFF-code in C and we describe in the next section how to run the code.
- **[3D-Box]:** This opens a 3D-view of the cartesian NLFFF-box and displays B_z on the bottom boundary.
- **[comp.field-line]:** Can be only used if a 3D-NLFFF-filed has been loaded. Left-mouse-click on the magnetogram and a field line is computed from the corresponding position on the magnetogram. Right-mouse-click to leave the program. This tool can be used both in 2D and 3D view.

- Column 5 contains also buttons to store magnetic loops in memory and to display the stored loops in 3D and 2D projection.

3.7.1 How to make NLFFF-extrapolations from HMI-AR vector-magnetograms?

Installation and use of the cartesian NLFFF-code

The code is written in C and is usually running on a UNIX or Linux computer, also compilation under Windows is possible, too. All necessary source-files are collected **C _ code _ nlfff.zip**.

- Unpack these files, e.g. with **unzip C _ code _ nlfff.zip** in Unix or Linux, or whatever program in windows.
- Install the code with **installNLFFF** under Linux and most Unix-system, or **installNLFFF _ helios** on our local parallel computer called Helios. If something goes wrong one has probably to change compiler options or something else.
- Copy all compiled files and batch-files (or for simplicity just all files) to the folder where you want to run the NLFFF-code. Should be the same folder as specified in IDL in the structure **options.dir NLFFF**.
- Before running the code one has to save (from the IDL-widget) all files the code needs with **[NLFFF Standard]** or **[NLFFF Advanced]**.
- Run the code with **nlfffcode**. This will usually take several hours, (see Table 1 in Wiegelmann *et al.* [2012])
- Finally the NLFFF-codes creates the 3D-NLFFF-field in **Bout.bin** and a potential field in **B0.bin**. These files can be loaded into the IDL-widget as described above (only 1 file can be loaded at one time, not both together).

Disclaimer

The program is provided *as-is*, without any warranty. In no event will the authors be held liable for any damages arising from the use of this software.

Acknowledgements

The program is written in IDL and uses the IDL-procedures **COLORBAR** from David Fanning, **READFITS** from W.B. Landsman and **WRITEFITS** from Jim Wofford. I thank Dr. Andreas Kopp and Borut Podlipnik for useful discussions regarding IDL and Dr. Bernd Inhester, Prof. Thomas Neukirch, Dr. Peng Ruan, Dr. Li Feng, Dr. Julia Thalmann and Dr. Tilaye Tadesse and my colleagues from the NLFFF-consortium and the SDO/HMI-team for many scientific discussions. I acknowledge financial support by an EC Marie-Curie-Fellowship at the university St. Andrews, and by DLR-grant 50 OC 0007 and DLR-grant 50 OC 0501 at the MPS in Katlenburg-Lindau, Germany.

Bibliography

J. J. Aly. On the reconstruction of the nonlinear force-free coronal magnetic field from boundary data. *Sol. Phys.*, 120:19–48, 1989.

T. R. Metcalf, M. L. Derosa, C. J. Schrijver, G. Barnes, A. A. van Ballegooijen, T. Wiegelmann, M. S. Wheatland, G. Valori, and J. M. McTiernan. Nonlinear Force-Free Modeling of Coronal Magnetic Fields. II. Modeling a Filament Arcade and Simulated Chromospheric and Photospheric Vector Fields. *Sol. Phys.*, 247:269–299, February 2008.

C. J. Schrijver, M. L. Derosa, T. R. Metcalf, Y. Liu, J. McTiernan, S. Régnier, G. Valori, M. S. Wheatland, and T. Wiegelmann. Nonlinear Force-Free Modeling of Coronal Magnetic Fields Part I: A Quantitative Comparison of Methods. *Sol. Phys.*, 235:161–190, May 2006.

N. Seehafer. Determination of constant alpha force-free solar magnetic fields from magnetograph data. *Sol. Phys.*, 58:215–223, July 1978.

M. S. Wheatland, P. A. Sturrock, and G. Roumeliotis. An Optimization Approach to Reconstructing Force-free Fields. *ApJ*, 540:1150–1155, September 2000.

T. Wiegelmann and B. Inhester. Magnetic Stereoscopy. *Sol. Phys.*, 236:25–40, June 2006.

T. Wiegelmann and T. Neukirch. Computing nonlinear force free coronal magnetic fields. *Nonlinear Processes in Geophysics*, 10:313–322, April 2003.

T. Wiegelmann and T. Neukirch. An optimization principle for the computation of MHD equilibria in the solar corona. *A&A*, 457:1053–1058, October 2006.

T. Wiegelmann and S. K. Solanki. Similarities and Differences between Coronal Holes and the Quiet Sun: Are Loop Statistics the Key? *Sol. Phys.*, 225:227–247, December 2004.

T. Wiegelmann, B. Inhester, A. Lagg, and S. K. Solanki. How To Use Magnetic Field Information For Coronal Loop Identification. *Sol. Phys.*, 228:67–78, May 2005.

T. Wiegelmann, A. Lagg, S. K. Solanki, B. Inhester, and J. Woch. Comparing magnetic field extrapolations with measurements of magnetic loops. *A&A*, 433:701–705, April 2005.

T. Wiegelmann, B. Inhester, and T. Sakurai. Preprocessing of vector magnetograph data for a nonlinear force-free magnetic field reconstruction. *Sol. Phys.*, 233:215–232, February 2006.

T. Wiegelmann, T. Neukirch, P. Ruan, and B. Inhester. Optimization approach for the computation of magnetohydrostatic coronal equilibria in spherical geometry. *A&A*, 475:701–706, November 2007.

T. Wiegelmann, J. K. Thalmann, C. J. Schrijver, M. L. Derosa, and T. R. Metcalf. Can We Improve the Preprocessing of Photospheric Vector Magnetograms by the Inclusion of Chromospheric Observations? *Sol. Phys.*, 247:249–267, February 2008.

T. Wiegelmann. Optimization code with weighting function for the reconstruction of coronal magnetic fields. *Sol. Phys.*, 219:87–108, January 2004.

T. Wiegelmann. Computing Nonlinear Force-Free Coronal Magnetic Fields in Spherical Geometry. *Sol. Phys.*, 240:227–239, February 2007.

T. Wiegelmann, J. K. Thalmann, B. Inhester, T. Tadesse, X. Sun, J. T. Hoeksema. How to optimize nonlinear force-free coronal magnetic field extrapolations from SDO/HMI vector magnetograms?. *Sol. Phys.*, submitted.