Finite Element 2D System for Electromagnetic and Thermal Field Analysis

EFCAD

Electromagnetic Fields Computer Aided Design

MANUAL

(version 8.0 – 2008)

developed by

J.P. Assumpção Bastos, N. Sadowski

GRUCAD – Grupo de Concepção e Análise de Dispositivos Eletromagnéticos
UFSC - Universidade Federal de Santa Catarina

and

LEEI – Laboratoire d'Electrotechnique et d' Electronique Industrielle
ENSEEIHT

TABLE OF CONTENTS

1. Introduction

EFCAD is a computational package developed to solve the 2D partial differential equations (as Laplace's and Poisson's equations) related to thermal and electromagnetic phenomena by using the FEM (Finite Element Method).

The first version of **EFCAD** was operational in 1986, and this manual refers to the eighth version.

Lately, many modifications and extensions were made in **EFCAD** in order to improve its performance and to add more options for the user. The package, **written in FORTRAN, works as a C program** through the use of a translation procedure. Because of this, the access to memory has been extended and the speed has been increased considerably in comparison to the older FORTRAN versions.

There are three main sections in **EFCAD**, namely, the pre-processor, the solution modules (solvers), and the post-processor. In addition, there are two programs for management of thermal and electromagnetic properties files.

EFCAD has been used successfully for many years and the system is a robust package. However, the system is not perfect and some problems may arise as it is being applied. If this happens, **we would like to be informed** in order to be able to correct any problems, or to incorporate appropriate modifications and extensions to improve the package.

In this manual, we will first describe the general functioning of the system and then, the different programs included in the package. At the Appendix section, the basic equations and boundary conditions are presented, without detailing the theory. We **strongly recommend the reading** of this Appendix, since useful information is provided. If the reader is interested on the theoretical bases of the method, we recommend the book "Electromagnetics and Calculation of Fields", by N.Ida and J.P.A. Bastos (see Reference section) where this package is intensively used for electromagnetic applications.

The development of **EFCAD** is a collaboration result. The research group of the *Laboratoire d'Electrotechnique et Electronique Industrielle de Toulouse* has been a main partner of the GRUCAD and, particularly, the theoretical and computation implementation of the versions of **EFCAD** related to electrical rotating machines were performed in cooperation with this group. Also, all the computational support (graphical libraries, compilators and link-editors) have been constantly furnished and improved by the LEEI team. The GOPAC, of the Universidade Federal de Minas Gerais, Brazil has developed and furnished the ICCG routines for linear matrix system solution (see Reference section).

Finally, we point out that the use of this system has been considered by most of the users as simple, friendly, fast and efficient. If any need and help is necessary, contact the GRUCAD or the LEEI.

Remark on the didactical (free) version of EFCAD: The didactical version of

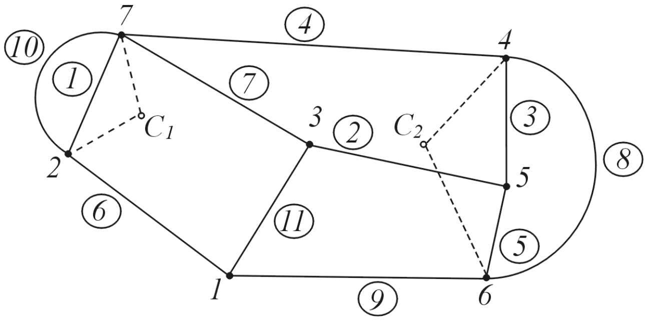
EFCAD can be obtained as a free package, which can be **downloaded** from the site www.grucad.ufsc.br, or contacting directly the GRUCAD. All the programs are **identical** to those of the full version, excepting that the solvers, which are **limited to 600 nodes or unknowns**.

Suggestion for using EFCAD: because many dialogues with the programs are performed as UNIX or LINUX commands, we suggest to open a MS-PROMPT DOS Window to interact with **EFCAD**.

2. The Drawing Description

This section describes the method of drawing the solution domain and its placement in a file. The drawing is related to the geometry of the study domain and does not contain any physical information. The related file is a **.pre** type and the drawing program EFD normally creates it, as it will be seen soon.

The next figure shows an example of a domain.



Example of a domain drawing

The drawing can have only two types of elements: **line segments** and **circular arcs**. The points can be **randomly numbered**. Two points, in any order, define a segment. For example, element 9 in Figure 1 can be described by:

1 6 0 or: 6 1 0

The number **"zero"** indicates that it is a **line segment**. The **elements** of the drawing can also be displayed in the file **at any order**, numbered randomly. Element **10** is a circular arc and is defined in a counterclockwise fashion; that is, element **10** is defined as:

7 2 2

This means that the arc **starts** at point **7** and **ends** at point **2**. The third number, which is **2,** means that the center of the circle is given by the second set of center coordinates.

With these definitions, the drawing in the figure above is defined by the following data set:

```
1000.
2     7     0
3     5     0
5     4     0
7     4     0
6     5     0
2     1     0
3     7     0
6     4     1
6     1     0
7     2     2
```

```
3       1       0
F
x1        y1
x2        y2
x3        y3
.
.
.
x7        y7
F
xc1   yc1
xc2   yc2
F
```

This set of data provides information as follows:

- The first line gives a **scale factor**. In this example, the value is **1000**., meaning that the coordinates are in **meters/1000.=millimeters**. If the description is **in meters**, the scale factor should be **1**.

- **Second part**: This contains the elements (line segments and arcs) that make up the domain, as explained previously. The points are given as integer numbers and the order of the elements is arbitrary. The letter **F** indicates the end of the elements listing.

- **Next are the coordinates** of the points that define the elements above. In this example, these are given in millimeters (real numbers). The letter **F** again indicates the end of the record.

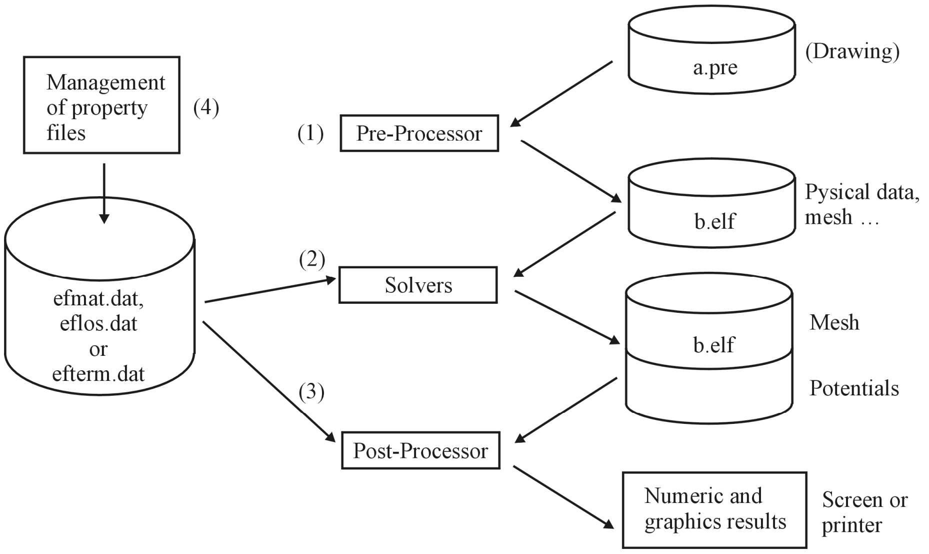
- **The next section gives the coordinates of the circle centers** for the arcs. In this case there are two arcs and their center coordinates are also given in millimeters. The letter **F** indicates the end of the record.

This file is always given an extension **.pre** by the pre-processor, which writes it as an ASCII, free formatted file (i.e. the numbers are separated by blanks or commas).

Because the structure of this file is very simple, **the user may develop its own drawing generation program,** directly adapted to his/her specific application.

3. General Diagram of EFCAD

The general diagram of EFCAD is shown in the figure below.



General diagram of EFCAD

Block 1 represents the **pre-processor section** of **EFCAD**. The programs in this section will be described shortly. The main goal of the pre-processor is to read the drawing file (**a.pre** in the example in the figure above) and, **after receiving** information that defines the physical problem (materials, sources and boundary conditions data**), to generate the mesh** necessary for the application of the finite element method to the problem.

The files created by the pre-processor are **written as coded ASCII** files and are therefore very difficult to understand. Their extension is **.elf.** In the figure above, the file name is **b.elf**. One drawing (such as **a.pre**) can generate many **.elf** files depending on the user's needs.

Block 2 shows the **solvers section** of the package. In this section, the FEM is applied. The file containing the mesh data and other information, **b.elf**, is read by one of the solvers. After the solution, the primary result: potential at the nodes, is written in file **b.elf**, following the mesh information, without modifying the existing entry information. To obtain the solution, an additional file containing the electromagnetic (or thermal) properties of the materials is also used. This file is called **efmat.dat** (or **efterm.dat**). For iron losses evaluation the data of file **eflos.dat** are needed.

Block 3 is related to the **post-processor program**, which will read the file **b.elf** (containing the solution) and the property file **efmat.dat**. This sector will transform the potentials into actual fields and provide a complete graphic and numerical treatment of the FEM calculation. The results can be displayed on the monitor and/or printed.

Block 4 is made of **three programs that manage the material properties** files **efmat.dat** and **eflos.dat** for electromagnetic applications or **efterm.dat** for thermal problems. In this section, the user enters the material properties normally required for solution of his/her own problems.

Some Remarks:

- After calculation in the solution section (block 2, and eventually passing through the post-processor, block 3) the pre-processor can read again a **.elf** file. This means that the user is able to change, for example a material or source etc., without changing all the definition previously made. However, it is **necessary to re-mesh** the domain.

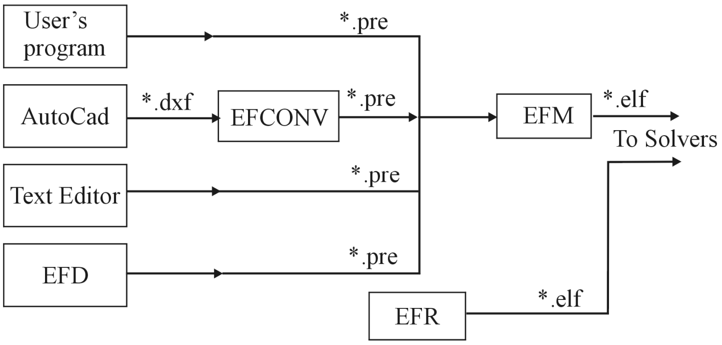
- All programs in **EFCAD** have names beginning with the prefix **EF**, as, for example, the solver **EFCS** that performs static finite element calculations.

- The pre- and post-processors are unique, and they can interact with any of the solvers.

- In any solver as well as in **EFGN**, the user can **type <Enter>** in response to the prompt: "File name?". The **last file** created by the pre-processor, which matches the required extension **.elf** is used by the solver.

3.1. The Pre-Processor Section.

The pre-processor (first block of the previous diagram) is shown in the next figure.



The pre-processor section diagram.

The **main goal** of using a simple description of the drawing (section 2), recorded as a readable ASCII file **.pre**, **is to leave the user completely free** to generate this file as needed. Since it is almost impossible to fill all the needs of all the users in terms of drawing, we have decided to proceed in this way. The user can write his/her own program to generate the drawing **.pre** according to his/her needs. For example, it may become necessary to calculate a sequence of *contactors* for which, only some dimensions change. The user can then write a small program which supplies the necessary parameters to generate the corresponding drawings **.pre**.

It is also possible to write this file directly in any text editor. Another possibility is the use of commercial drawing package such as AutoCad[1]. In the latter case, the drawing is

[1] **Some** versions of AutoCad.

written in standard format as a file **.dxf**. Program **EFCONV**, which is an integral part of **EFCAD**, will then grab this file and convert the line segments and arc segments in **.dxf** to the format required by **.pre**.

Program **EFD** is the code provided in **EFCAD** for the purpose of creating the drawing. This program allows the user to generate the line segments and circular arcs necessary for drawing, in a simple and user friendly environment. This module, more than any other in **EFCAD** has been constantly changed and improved in order to fill the needs of different users. For this reason we recommend the following:

1. That the users ask for and use only the last version of **EFD.**
2. That the users suggest any modifications and extensions he/she feels are necessary for his/her needs and are of general interest.

Program **EFM**[2] performs many tasks. The first is to offer drawing possibilities, allowing the user to make some final modifications and adjustments to file **.pre**. Then, as an internal procedure, the program continues with defining the regions that make up the solution domain. The third step is to offer the user the possibility of defining materials and sources in the regions, as well as boundary conditions, and others. Finally, the last step is the automatic mesh generation and creation of file **.elf**. In its last version**, three methods of meshing are available** in **EFM**, adding robustness and flexibility to it.

Program **EFR** is a useful program in describing a problem in which the physical domain is regular, and can be described by a regular mesh, such as the meshes used in finite difference methods. It is extremely simple and fast and, therefore, is highly recommended whenever it is possible to describe the domain in this way. The finite elements generated are **quadrilateral elements** and these provide **very accurate results**. **EFR** creates a **.elf** file directly, which can then be used by the solvers. If desired, **EFR** can also generate a **.pre** file based on the main lines defining the region.

Program **MESH** is another program in **EFCAD** pre-processing environment. It enables the graphical identification of mesh nodes and elements and can be useful in some applications.

3.2. The Solvers Section

The solvers section is shown in the next figure.

Only **.elf** files can be read by the solvers. The user chooses the solution module according to his/her physical problem. After performing the calculation, the solvers add to the same file (**.elf**), a **“flag”** variable indicating **which solver was chosen** and the corresponding **solution** (potentials at the nodes of the mesh).

All solvers have names starting with **EFC** which mean that Finite Element Calculations are performed. The file **efmat.dat**, which contains the electromagnetic material

[2] Some EFCAD packages incorporate also program EFMA concerning meshing slightly different of EFM.

properties is read by **EFCS, EFCC, EFCJ, EFCT, EFCF, EFCR, EFCR360, EFCM, EFCM360, EFCG, EFCSL, EFCMF, EFCMF360** and **EFCV**. File **efterm.dat**, which contains thermal properties is read by programs **EFCTS** and **EFCTT**. The iron losses data file **eflos.dat** is used by solvers **EFCR, EFCR360, EFCM, EFCM360, EFCB, EFCMF** and **EFCMF360**.

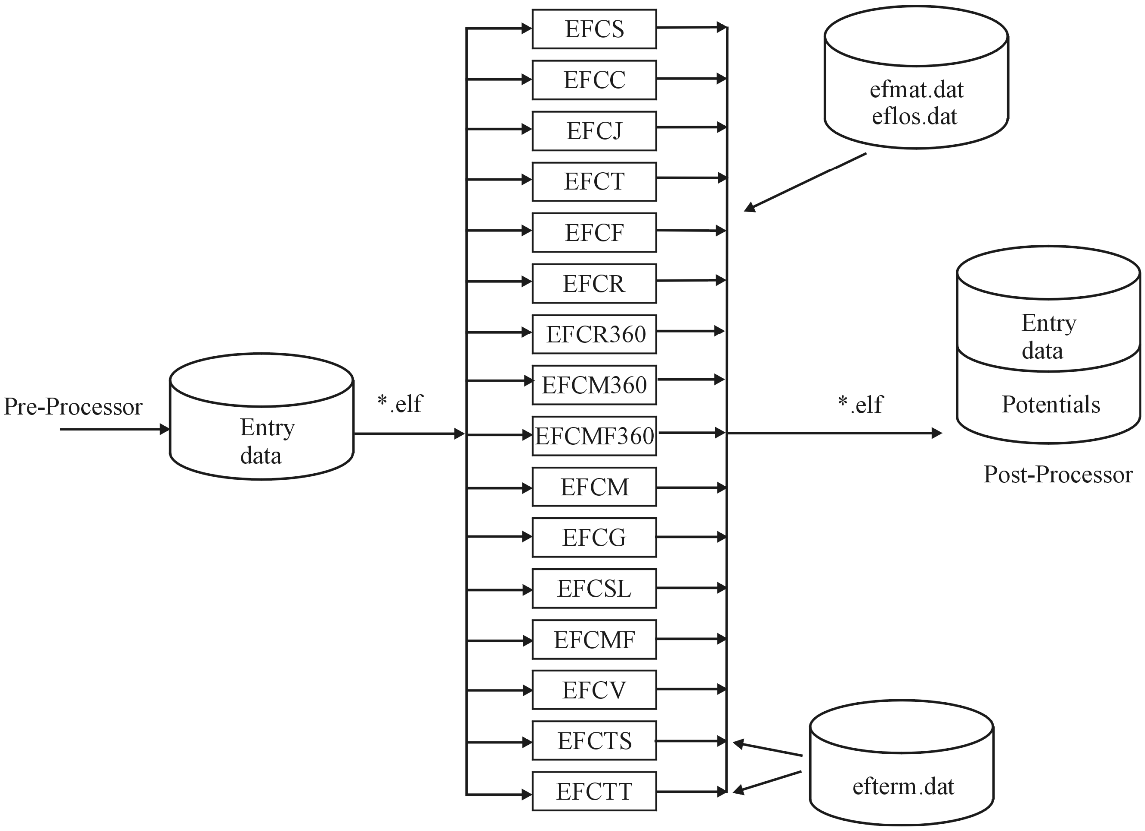


Diagram of the solution section

The solvers are:

EFCS: For **static** electromagnetic calculations.

EFCC: For electromagnetic calculations using the **complex formulation**. This module can be used for sinusoidal excitation, including eddy currents, but with linear materials only.

EFCJ: For transient electromagnetic problems using a step by step timing procedure. Non-linearity and eddy currents can be treated. The **sources** are **current densities J**, defined by the user.

EFCT: Similar to **EFCJ** but the **sources are voltage fed**. In this case, the electric circuits related to the sources are taken into account simultaneously with the solution. The unknowns are the potentials at nodes **as well as currents established in coils**.

EFCF: **For solving nonlinear, time stepping, eddy currents cases fed with voltage defined in the sources and** thick conductors relied**. Thick conductors are conducting material where eddy current are induced.**

EFCR: Calculation for Static Cases with rotation for electrical machines **with (anti) periodic boundary conditions**. Allows the calculation of **flux**, **inductances**, **torque**, **f.e.m.**, **inductions** and **iron losses** as a function of rotor position.

EFCR360: performs the same calculations as **EFCR** for electrical machines that must be simulated **without using (anti) periodicity boundary conditions**. In this case, the whole machine is calculated.

EFCM: Similar to **EFCT** but incorporating rotation for voltage fed electrical machines. Allows calculation of **currents**, **torque**, **rotor speed** and **iron and eddy current losses**.

EFCM360: makes the same calculations than **EFCM** for electrical machines that must be simulated **without using (anti) periodicity boundary conditions**. In this case the whole machine is calculated.

EFCG: Special solver to squirrel-cage induction motors; **EFCG** is similar to **EFCM** but considers the electrical circuit of the squirrel-cage. Allows calculation of currents in stator windings and rotor bars as well as **torque**, **rotor speed** and **iron and eddy current losses**.

EFCSL: For static problems as **EFCS** but incorporates the possibility of solving simultaneously cases for several sources. Calculates automatically **flux and inductances**.

EFCMF: Solving nonlinear, time stepping, eddy currents cases fed with voltage defined in the sources, connection between thick conductors and taking into account rotor movement. As **EFCM** allows calculation of **currents**, **torque**, **rotor speed** and **iron and eddy current losses**.

EFCMF360: performs the same calculations as **EFCMF** for electrical machines that must be simulated **without using (anti) periodicity boundary conditions**. In this case, the whole machine is calculated.

EFCV: Electromagnetic calculations in which parts of the structure **move at constant velocity.** Eddy currents and nonlinear materials are treated in this module.

EFCTS: For **static thermal** calculations.

EFCTT: For **transient thermal calculations**. This module is particularly useful when the temperatures in the structure are changing.

3.3. The Post-Processor Section

The diagram of the post-processor is now shown.

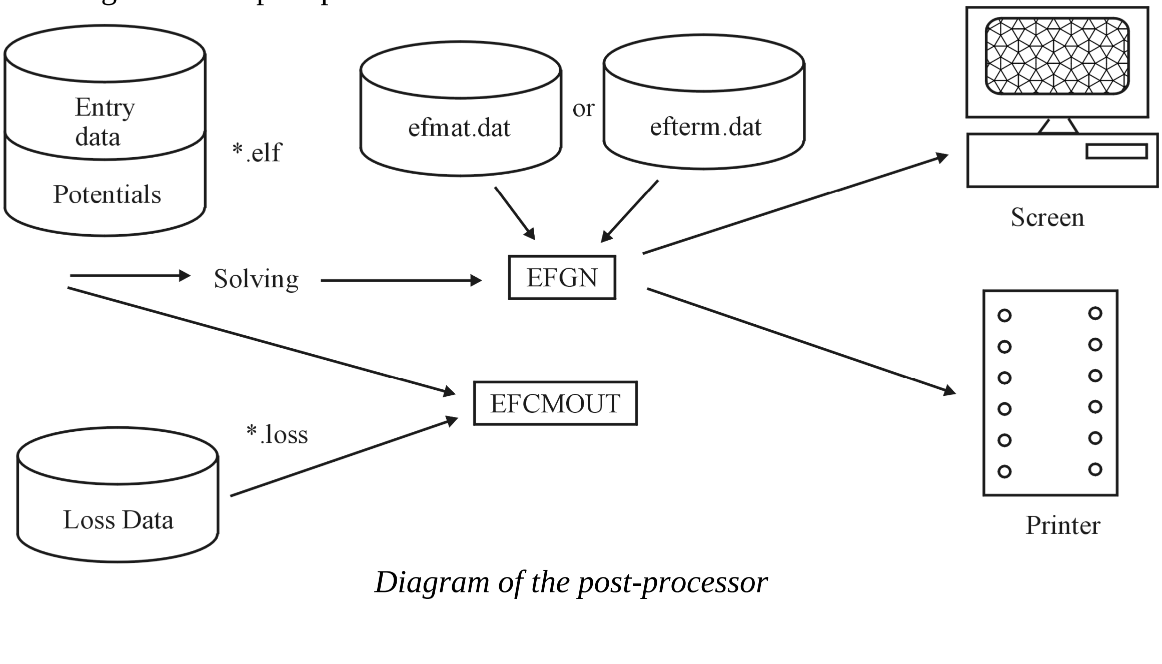


Diagram of the post-processor

Program **EFGN** is a general module performing all the post-processing operations. File **.elf**, with the potentials (or temperatures) as calculated by any of the solvers is read by **EFGN**. Based on the data read, **EFGN** provides both graphic and numeric results in a simple, friendly way.

EFGN uses the file of electromagnetic or thermal properties (**efmat.dat** or **efterm.dat**) depending on the nature of the physical problem**. EFGN** presents the results on the screen or on a printer, according to the user's needs.

EFCMOUT is another post-processor of **EFCAD** allowing investigation of iron losses in electrical machines. **EFCMOUT** reads a file **.los** generated by the solvers **EFCR**, **EFCR360**, **EFCM**, **EFCM360**, **EFCG**, **EFCMF** and **EFCMF360**.

3.4. Management of the Properties Files

Management of the material properties files is performed by the following diagram.

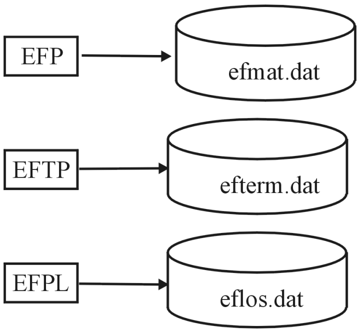


Diagram of the material properties file management.

To **define the materials** inside the regions of the solution domain **we use numbers**. The numbers relate the data inserted in files **efmat.dat** and **eflos.dat** for electromagnetic applications or **efterm.dat** for thermal applications.

The management of these files is made through program **EFP** (electromagnetic properties), **EFPL** (iron losses properties) or program **EFTP** (thermal properties). **The most convenient way to work with the material properties** is to fill the files **efmat.dat**, **eflos.dat** and **efterm.dat** with all materials one expects to commonly use. If this is done, preparing and solving a particular problem does not require additional manipulation of files **.dat**.

In the pre-processor step, the reading of files **.dat** is not necessary. The user only needs to identify the particular region with a number which represents the material with properties he/she needs. The solution and post-processing modules read the property file and associate the material number with the material property.

4. Description of The Programs

In the description of the **EFCAD** programs that follows, it is assumed the user is familiar with the concepts of electromagnetics. The description is therefore limited to the technical aspects and usage of the programs.

The major interaction between the user and **EFCAD** is obviously during the pre-processor step, in which the drawing and physical information are furnished by the user. It is advisable that the user prepares the data in terms of geometry and physical aspects before typing it.

A well-prepared and properly defined case is normally easy to enter and will shorten the time needed to insert the data. On the other hand, a badly or incompletely prepared problem will result in the need for corrections and may result in errors, not to mention the time loss.

Some remarks:

- For all modules, when the user is asked for a file name (**.pre or .elf**) the user has to type the file name **without** the extension.

- In any solver and in **EFGN**, if **the user types <Enter>** following a prompt for a file name, **the last file generated** by the pre-processor will be **used by the program**.

- Using the **"menu"** the user can either **click inside the box** related to the desired item or simply **type on the keyboard the upper case letter** appearing in the corresponding message. In this case, the **cursor cannot be located on the menu** space.

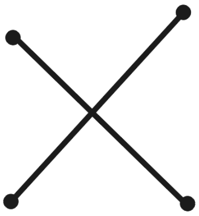
4.1. THE PRE-PROCESSORS

EFD

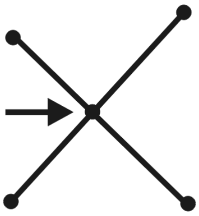
4.1.1. EFD - Drawing of the Domain.

This program is responsible for drawing of the structure. It has been the object of many changes and extensions in order to fill the different needs of users, which constantly ask for new features and options. **EFD** creates a file **.pre**, which is ASCII, formatted (see section 2) and contains the description of the geometry of the solution domain.

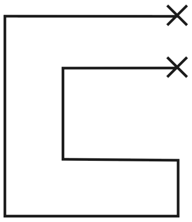
In normal use, **EFD** displays the drawing on the monitor. The elements (line segments and arcs), points, and the center of circles are displayed in different colors. The situations shown below should be avoided, since they are incompatible with the regular use of FEM and will cause difficulties/errors in subsequent steps.



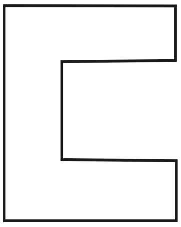
a. Incorrect: Segments crossing each other



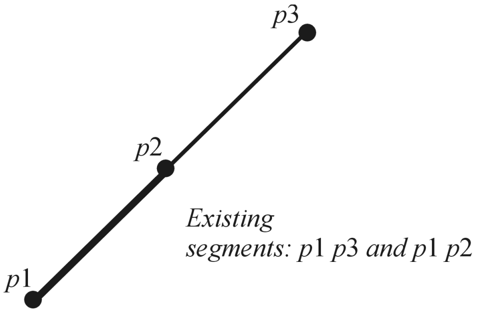
b. Correct: crossing point is defined



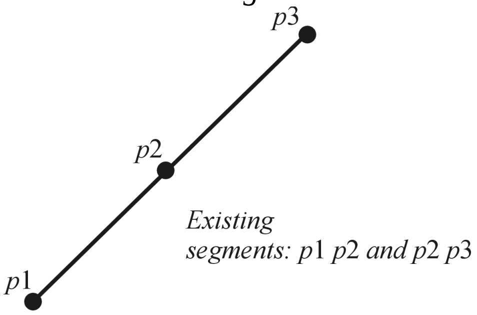
a. Incorrect: Disconnected points



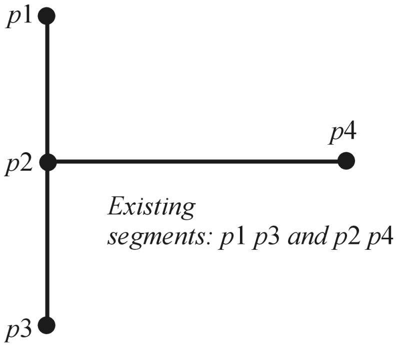
b. Correct: Connected points by inserting a segment



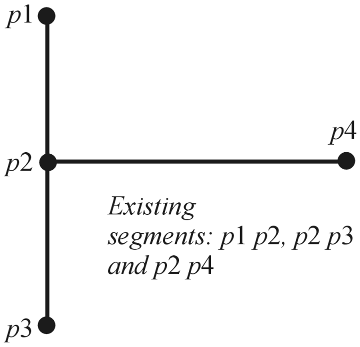
a. Incorrect: Superposed segments



b. Correct: Successive segments



a. Incorrect: Segment is disconnected



b. Correct: All segments connected

Program **EFD** saves the drawing data onto the disk as it is being entered. Because of this, if the program aborts at any stage, practically all data entered prior to the incident is saved in the corresponding file **.pre.**

Most of the options in **EFD** are displayed in the "menu" bar on the top of the screen. They are self explanatory and easy to use. The steps and commands of **EFD** are:

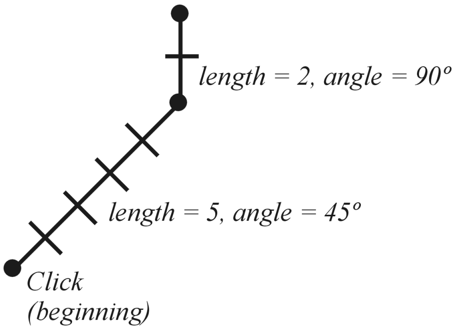
Initial commands: First, the user is asked if this is a new case or an existing case in need of modifications. If the user indicates it is a new case, the user has to enter the **scale** (meters/Scale, e.g. 1000. if the unit to be used is millimeter) and the limits of the drawing (which can be larger or smaller than the limits of the solution domain). When a line segment or arc exceeds the limits, the program adjusts the limits accordingly. The user is then asked for the **name of the new file.** If the problem **is an old case**, the name of the file must be entered and the user has to specify the the file name in which the modified file will be stored (it can be the **same,** and the old file will be **overwritten** with the modified data).

If the user indicates a new case, a point corresponding to the minimum coordinates of the limits will appear on the monitor. If the case is not new, the old drawing will be displayed.

a. Command 'Segment' (message: "Segment").

Draws the line segments. First, it is **required to click** on the origin of the segment. Then the user **types the length and angle** (in degrees) of the segment (at the bottom of the window). The angle refers to the trigonometric orientation of the segment, assuming that the origin point is at the center of the trigonometric circle. As a segment is created, the **final point becomes** the origin of the following segment, to ensure connectivity of the segments. To **finish a sequence, the user types "E"** when the message ***"Length, Angle (degrees);[Type E for Exit] =*** " for the next segment appears on the monitor.

Example:



Using command "Segment"

b. Command 'New point' (message: "Newpt")

A sequence of points can be **inserted** in the drawing **through the keyboard**. The points can be inserted in Cartesian coordinates or Polar coordinates. The corresponding messages are "***Coords New Point (Type E to Exit) X,Y=*** " and "***Coord New Point (Type E to Exit) R, Theta =*** " respectively. As the coordinates are typed, the point appears on the drawing. The sequence **is finished by typing E.**

c. Command 'Graphic Point' (message: "grfPt")

Setting this option, the user creates new points **only by clicking the mouse** at the location where new points are needed. This command is very useful when the exact location of the point is not important. The **option is canceled** when another **command is set.**

d. Command 'Join' (message: "Join")

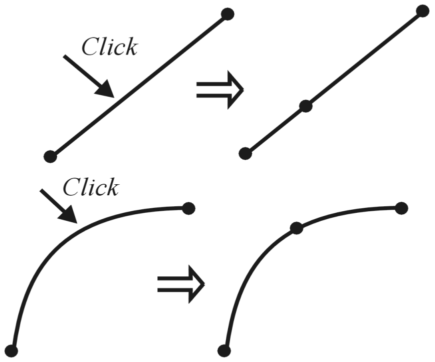
This command creates segments by **clicking on two points** defining the new segment. The option is **canceled** when a **new command is set.**

e. Command 'Division' (message: "Division")

This option is used when it is necessary to **divide a line segment or arc into two**

parts. The mouse is clicked at the location (or very close) where the division is desired. A new point is created at that location and the original element is divided in two. The command is **canceled** when **a new command is set**.

Example:



*Element (segment or arc) are divided in two part***.**

f. Command 'Zoom' (message: "Zoom")

The 'Zoom' command is used to magnify a part of the domain, in particular where visual inspection is difficult. After **setting the 'Zoom' command**, the user clicks on the **two points defining the window** to be magnified.

h. Command 'Continue' (message: "Continue")

Clicking this command, following 'Zoom', or any other, the drawing **reverts back** to the original limits.

Important Note: Commands **"Zoom" and "Continue" are used in the way** described here **throughout the EFCAD package.**

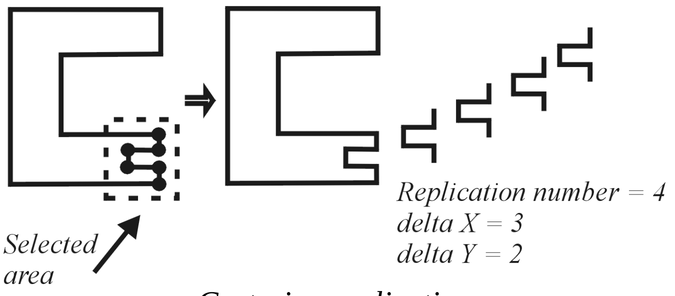
i. Command 'Elimination' (message: "eLim")

This command is set when it is **necessary to eliminate segments or arcs**. After setting the command, a sub-menu appears, giving the choice between the direct use of the **mouse** or a message for **deleting** (erasing) the last element entered (message is ***"elim Last"***). If none is clicked, the **mouse** becomes the default choice. In this option, clicking the mouse on the required segment or arc eliminates a line segment or arc. Click exactly on the segment (eventually making a "zoom") to avoid ambiguous results. If "**elimination last**" is selected, the last entered element (line segment or arc) is erased, just by **clicking on the message itself**. This option is particularly useful when an element is erroneously entered. In such an instance, the segment can be eliminated before proceeding.

j. Command 'Replication' (message: "Replic")

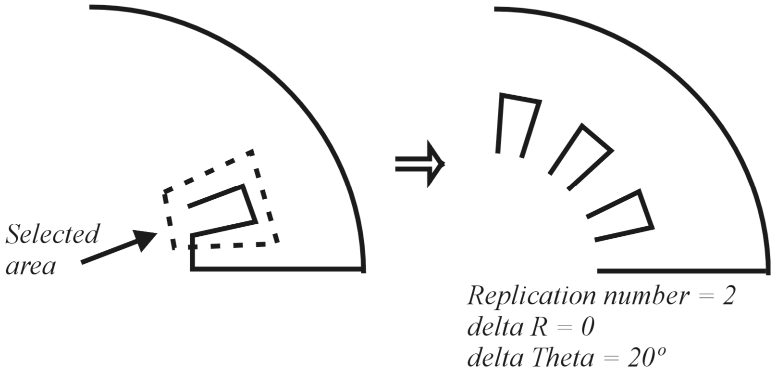
This option is set when it is **necessary to replicate** elements in a geometric sequence. This becomes very useful when a geometry is made of repetitive sections such as in electric machines, where, for example, a slot can be defined and then replicated as many times as necessary. When this option is set, a **sub-menu** appears and the user can choose between **"Cartesian" and "Polar**" replication. The user must **first select the window** in which the elements to be reproduced are. This **is done by clicking twice on the mouse** on the appropriate locations for defining the window section . Then, the number of replications and the desired displacement are entered from the keyboard, at the bottom of the screen.

Example of Cartesian Replication:



Cartesian replication

Example of Polar Replication:



Polar Replication

In the case of polar replication the **point (0,0)** (existing or not inside the domain) is assumed **to be the center of the polar circle.** For example, if the user is working with an electric machine, the point (0,0) should be chosen as the rotation axis of the machine.

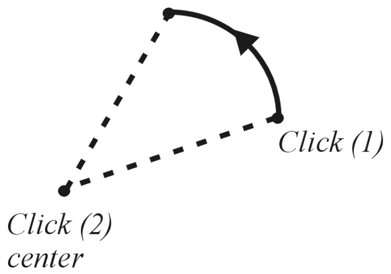
k. Command 'Arc' (message: "Arc")

EFD offers different ways to create arcs. When this option is set, a sub-menu appears on the screen with the following options:

k1. Point, Angle, Center (message: "pt/Ang/cen")

Arcs are always defined in the counterclockwise direction. In the case of this option, the user first clicks the mouse **on the origin point** of the arcs. Then the **angle** that the arc spans is **typed through the keyboard,** and finally the **user types the coordinates of the center** of circle or **clicks the mouse** on the center of the circle. By using the mouse, the user assumes the center of circle is in the drawing area. Typing the coordinates of the center does not require that the center be in the drawing area.

Example:

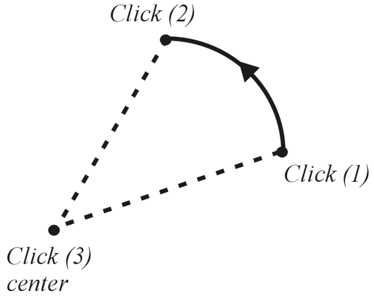


Creating an arc defined by initial point, angle and center

k2. Two points and Center (message: "2 pTs/cen")

The arc is created by clicking on the **two points defining the ends** of the arc and then by clicking on the **center of the circle** (or entering the coordinates of the center through the keyboard).

Example:



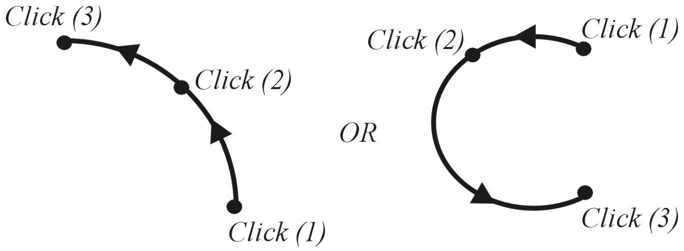
Creating an arc by using two points and the center

In this case, **it is necessary** that the **distances** between the center and the two points are **equal or very close**, otherwise the operation is aborted.

k3. 3 Points (message: "3 Pts")

The user clicks on three existing points that define the arc. Observe the difference when clicking on the points in two distinct ways.

Example:

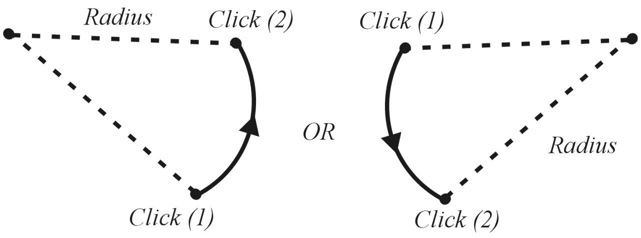


Creating an arc defined by the three points; note the two possible ways

k4. Two points and radius (message: "2 pts/rD")

The user has to click on two points defining the arc and type the length of the radius.

Example:



Creating an arc by using two points and the radius length

Note: The center is always located on the side of the arc such that the sequence "initial point of arc, final point of arc, and center of circle" are in the counterclockwise rotation. If the radius is too small (smaller than half the distance between the two points) the operation is aborted.

k5. Two points and center at the Origin (message: "2 pts/Or")

The user click **on the two points** (counterclockwise) defining the arc and the **center is considered automatically at the origin (0,0);** it is very useful, for example, when defining arcs for electrical rotating machines.

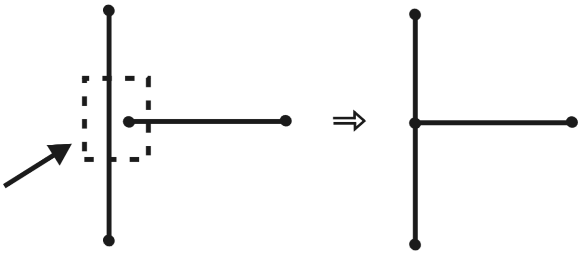
k6. Circle (message: "ciRcle")

A circle is created by the **addition of two arcs**. The first spans from 0 to 180 degrees and the second from 180 to 360 degrees. To use this option, the user **indicates the center of the circle** (by clicking the mouse or entering the coordinates through the keyboard) and then **entering the length of the radius**.

l. Command 'Verification/Cleaning' (message: "Vrf/clean")

Selecting this command accomplishes many tasks:

- Points that are not used are erased.
- Disconnected elements will be drawn in a different color.
- Crossing elements will be drawn with a different color.
- If two segments are superposed, one of them will be eliminated.
- Duplicate elements are eliminated.
  - The operations below will be performed:



Connecting segments are automatically set

If the distance between the disconnected point and the valid element is very small, the point is placed on the segment and the segment divided into two valid segments.

It is always advisable to click this command before exiting the drawing stage in order to detect and correct errors at this stage rather than detect them later on.

m. Command 'End' (message: "End")

Typing this command completes the drawing. The file is stored in the format **.pre** described above (section 2) indicating the number of segments, arcs, points, and centers of circles stored.

EFM

4.1.2. EFM - Physical Definition and Meshing

The program **EFM** has been the subject of many improvements these last years. Some are internal for adjusting, correcting and improving algorithms and procedures. Others are related to new features and possibilities. Now, the user can choose one of three meshing methods. The first one (see reference section), based on three operations (closing **acute angles**, eliminating **concavities** and using node distances to the polygon **barycenter** as criterion for creating new nodes) as well as the second (creating a **“cloud”** of new nodes and generating the elements by the **Delaunay method**) were developed at the **GRUCAD**. The third one, is a program called “Triangle” developed by Richard Shewchuk (see Reference section) and available as a public domain program. This program (also based on the method of Delaunay) is included in **EFM** and it is called automatically generating the meshes at the different regions. The whole functioning of **EFM** remains the same and just before meshing, a sub-menu proposing this choice appears in the screen. Also, now, the user **firstly furnishes the name of the file (.elf)** where the problem and mesh will be stored. Therefore, if a trouble occurs during the case treatment, the entered data is kept in the file and the user can re-starts **EFM** without losing the data previously typed.

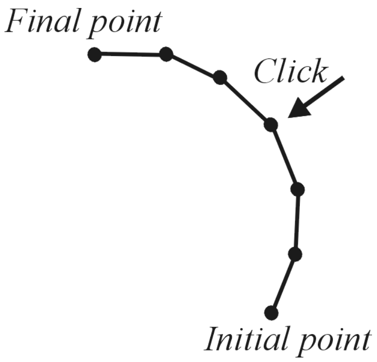
EFM discretization section can also **create quadrilateral elements**, whenever a rectangular region (or close to rectangular) is detected. For more details, read the option “discretization” below.

Program **EFM** has three main steps:
- **Reading the drawing file .pre**, including the option of making modifications to the drawing.
- **Definition** of materials, sources, and boundary conditions for the given case.
- **Automatic mesh** generation.

EFM can read a file **.pre** for a new case or a file **.elf** to proceed with modifications in a case already treated by **EFCAD**. When the program asks for the file name with extension **.pre**, the user can simply type <Enter> and the question will refer to a file with extension **.elf**. In this case, **EFM** goes directly to the second step mentioned above.

In the first step, after the name of file **.pre** has been given, the drawing of the structure is displayed on the monitor. The user is now able to perform some modifications to the drawing using most of the commands described in section 4.1.1 (for the program **EFD**). However, there is one command particular to **EFM** which is displayed in the menu as ***"Any".*** This option is related to the arcs of circles. These are transformed into line segments after division of the arcs by **EFM**. If the **number of divisions does not**

match the user's particular needs, the user can click the **mouse on this option and then click on one of the points of the arc,** as shown in the figure.



Modifying the number of segments defining an arc

The program writes on the bottom of the screen the number of divisions of the arc and it **will ask for the number of divisions desired by the user**. Following input by the user, the arc is now divided and re-displayed on the monitor to show the new division.

All other commands possible in this first part of **EFM** are similar to the commands described in section 4.1.1.

When the drawing is ready, the user can click on ***"Exit".*** The program asks if the user **wants to store the (modified) drawing** in another file **.pre**. If the answer is positive, the name of the new file is required. This file name **can be the original file name**, in which case the **old drawing will be overwritten**. If the **user does not want** to store the new drawing he/she can **simply type <Enter>**.

When this part is finished, **EFM** will find automatically the adjacent regions forming the whole domain.

Now the second step of **EFM** starts by drawing the domain and a new menu appears at the top of the screen. The commands in this menu are:

a. Materials (message: "Mater")

By default, all regions are filled with **material No. 1** (a good idea is to give material No. 1 in file **efmat.dat** the properties of air). By setting this option, the **number of the material** is provided by the user and **by clicking the mouse inside the selected region**, the region is **filled with this material**.

b. Thermal and current sources (message: "J-curr" and "Q-therm")

The user types the **number** of source (the **actual values**, in MKSA units will be **supplied to the solver later**). The user clicks the mouse inside the regions that form the source. If an **incorrect source** or region are specified, the **user can define a source number "zero"** and click inside the region that needs to be corrected.

c. Electromagnetic boundary conditions (message: "electrm bC")

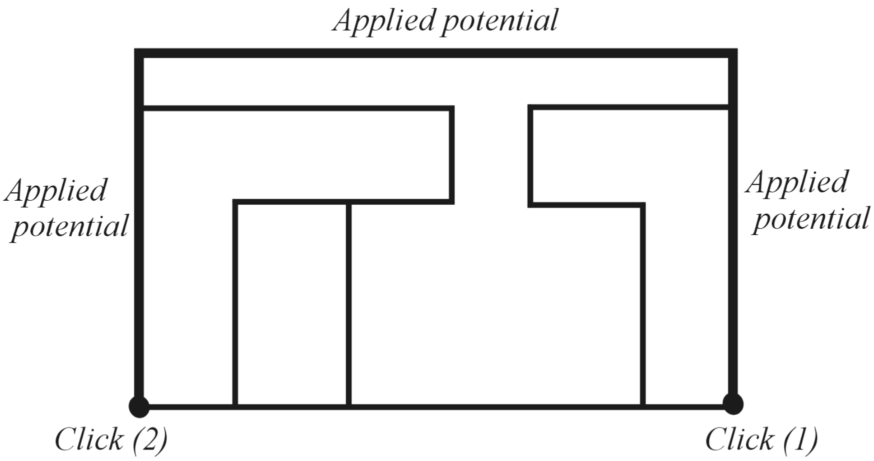
With this command the user can apply Dirichlet boundary conditions (that is, to apply fixed potentials on the boundary). A sub menu appears with the following commands:

c1. Total Dirichlet on the Boundary (message: "totDirch")

By clicking the mouse on the option, **all segments of the boundary** will have fixed a **potential equal zero**. This option is a **very simple way of defining** this particular condition in problems in which the **vector potential is used** for solution and which have no flux crossing its boundaries. (**Note**: applied (imposed) potentials or Dirichlet boundary conditions are two expressions for the same condition).

c2. Dirichlet - 2 points (message: "dirch/2 Pt")

In this case the user enters the value of the applied potential and **clicks the mouse on the two points defining the partial boundary** on which the potential is applied. **All segments** of the boundary included **between the two points (in counterclockwise sense**) will assume the value of this potential, as shown in the figure below:



Applying potentials (Dirichlet condition) at all the segments of the boundary between the two points indicated.

c3. Dirichlet by Segment. (message: "dirch/Segm")

This follows the same procedure as above but instead of clicking on two points defining the boundary, **the user clicks the mouse on the segments** on which the potential is to be applied. It is advisable to click the mouse in a point close to the middle of the segment, in order to avoid confusion with other segments.

c4. Elimination by Segment (message: "eLim segm")

When a segment has an applied potential and **this condition has to be removed**, the user can set this option and click on the segment on which he/she wishes to remove the applied boundary condition. This can be repeated and stays in effect until another option is set.

c5. Total elimination (message: "Tot elim")

By setting this option, **all Dirichlet boundary conditions** will be **removed**.

When defining applied (Dirichlet) boundary conditions **inside the domain** (as opposed to its boundaries) the option **"Dirichlet by segment"** is the only way to achieve this. The segment on which the boundary condition is applied must be the border between two different materials since, otherwise, when the mesh will be "smoothed" the internal nodes (not on the boundary of the domain) will be displaced causing incorrect results. If the line is not the border between two different materials, the user must define a **different material number** and assign to this number the same properties (electromagnetic properties in **efmat.dat** or thermal properties in **efterm.dat**) as those of the neighboring material.

d. Thermal Boundary Condition (message: "Therm bc")

The options are the same as above, but with a difference: when **choosing any "Dirichlet"** option, the following question will appear at the bottom of the screen:
"Imposed Temperature (Dirichlet) or Convection (D/C)?"

If the answer is **"D"** for Dirichlet, the user then enters the value of the applied temperature (centigrade degree). If the boundary is a **convection/radiation boundary**, the program will ask for the **external temperature** and the **coefficient H** that appears in the convection equation (see Appendix, section A10).

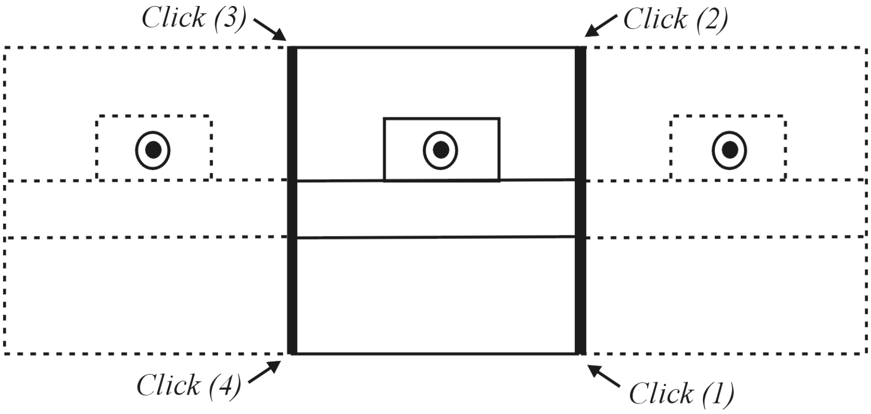
Specifications of the boundaries that have these conditions follows the procedure outlined above. Boundaries with thermal boundary conditions are drawn **with dashed lines.**

e. Periodicity (message: "Period")

When this command is set, a sub-menu appears with the following options:

e1. Periodicity (message: "Period")

In this case the user has to indicate two sequences of segments which are periodical, as in the figure below:



Indicating the two periodical boundaries

The sequence for indicating the periodical boundaries is: **click on the two** limiting points for the first boundary and then, **click on the two** other points limiting the boundary which is periodical to the first one. **EFM accept**s these indications **only if the number of segments of the two sequences** are equal and if their dimensions are the same so as to match the physical assumption of periodicity.

e2. Anti-periodicity (message: "Anti-per")

Anti periodicity occurs under similar condition of “periodicity” but when the **sources** in an adjacent region, **are inverted** compared to the sources in the region itself. To indicate the segment sequences the procedure is the same as above.

e3. Elimination of periodicity (message: "Tot elim")

One clicks on this option and all previous indications of periodicity or anti-periodicity have to be canceled.

f. Discretization (message: "Discret")

The mesh is generated by setting this command. As already indicated, this new version offers three methods of discretization. Therefore a sub-menu appears with following messages:

f1. Acute angles method (message: “Ac/co/ba”) ; in this case a meshing method, developed at the GRUCAD, is based on closing acute angles, breaking the concavities and using the barycenter as criterion for creating new nodes is used. It is a very fast method and it is recommended to apply it as first attempt to mesh.

f2. Cloud method (message: “cloud”) ; this method, also developed at GRUCAD, creates a “cloud” of nodes inside the regions and elements are generated based on the method of Delaunay.

f3. Using Triangle (message: “Triangle”); Triangle is a mesher available as public domain software. It uses the method of Delaunay and it is a very efficient and robust mesher. For more information about it, see the reference section.

After choosing the mesher, **another sub-menu appears** giving the options from ***"very coarse"*** to ***"very fine"*** meshes. The user chooses one of these mesh densities by **clicking on this menu** or typing on the **keyboard the number** corresponding to the desired mesh density (these numbers are indicated in the menu). The ***"normal"*** density is usually adequate for almost all practical applications. Following this, the drawing of the structure (in red) is displayed on the screen, allowing the user the possibility of **adding or subtracting divisions on the segments** (messages: ***"Add"*** or "***subtRact***"). If, for example, there is a region that is considered to be important for the practical analysis, the user **can opt for adding divisions** on the liens surrounding this region. The mesh will then be denser in this part of the domain. To perform this operation, the user **clicks on the menu (“*Add*”)** and **then on the segment**. For **each click** on the segment, the number of divisions **increases (or diminishes is “*subtRact*” is set) by one**. It is possible to do this sequentially in one or many segments. The option is canceled when another option is chosen. After all additions are entered, the user should select the option ***"Mesh"*** to generate the mesh.

If a **region is approximately rectangular** in shape (four angles close to **90 degrees** and the opposite **sides have the same number of divisions**), quadrilateral elements will be generated in this particular region. This type of elements provides **very accurate solutions** and therefore, it is advisable to use them whenever possible, by defining rectangular regions with the same number of divisions on opposite sides.

After generating the mesh, a **"smoothing" technique** is applied in order to improve the quality factor of the elements. This factor is written on the bottom of the screen and the **closer it is to "1" the better the mesh**.

After meshing, the principal menu appears again and if the user wishes to try a new mesh density, the same option ***("Discret"***) has to be selected again.

g. Others (message: "Others")

Clicking on this option some additional possibilities are displayed in a sub-menu. These are:

g1. Insert/Suppress (message: "ins/Supp")

Selecting this option, the user will be able **to insert or delete nodes in the mesh**. The mesh will appear, and the user has to click on ***"Insert"*** or ***"Suppress"***. In the first case, a **node is inserted wherever the mouse is clicked.** The user can continue inserting nodes one after the other. If the ***"Suppress***" option is selected, nodes are selected for deletion by clicking on them. Again, nodes **can be deleted one after the other**. Nodes **cannot be deleted or inserted** in regions **with quadrilateral elements nor** at the borders between **different materials and/or sources**.

g2. Previous division (message: "preVdiv")

This option displays on the screen the divided segments as it has been in the previous meshing. This option is **useful when the user proceeds with division changes** in the segments **and after meshing he/she notices that at some places, the segments should also be divided more finely.** In the other hand, if the user set the option ***"Discret"*** all the previous changes made on the segments divisions **would be lost** and **EFM** would **adopt the standard** division the mesh density chosen.

g3. Listing (message: "Listing")

If a **listing of the regions is required**, the user can select this option. The numerical screen appears and a listing of region data is displayed, showing the number of the material, the number of the source inside the region, and the segments forming the polygons related to the region.

g4. Displacement (message: "Displc")

This option is selected when it **becomes necessary to displace part**s of the domain. First the points to be displaced have to be selected using a **Cartesian** or **Polar** indication as described in **EFD**. In the case of **EFM**, the window can be defined graphically (by clicking the mouse twice) or numerically (by typing the coordinates of two limiting points of the window). After selecting the window, the user indicates if the displacement is a **Cartesian or a Polar** displacement. Even though not frequently encountered, the displacement **can be Polar for a window defined in Cartesian** coordinates and vice versa. After selecting the type of displacement, the user types at the bottom of the screen the value of **"delta X, Y"** or **"delta R, Theta"** for the Cartesian or Polar option, respectively. The modified domain is then drawn. The user must confirm the modification by clicking ***"OK"*** or canceling it by clicking ***"Cancel".***

h. listing file Efmat.dat (message: "eFmat.dat")

Using this option, a numerical display of the file **efmat.dat** appears on the screen. This option is useful when, for example, the user, before inserting the materials inside the region, wants to be sure of the correct number of the medium.

i. End (message: "End")

This option is normally accepted if the discretization has been done. It is extremely important to note that for any modification of materials, sources, boundary conditions and periodicity conditions, the **mesh has to be generated again** so that the elements and nodes of the mesh match the new conditions. However, if the user desires to finish the test, **EFM** will allow him/her to do so, after confirming this wish. All the physical conditions will be retained, but the file will not contain the mesh.

When ***"End"*** is clicked, the nodes of the mesh are renumbered to minimize the bandwidth of the matrix in the solver. A message relating to this renumbering is displayed. Some additional verification on the mesh is made by **EFM** and the data will be stored.

Important Remark: *another mesh generator* **EFMA** *is also available. It presents the same commands that* **EFM** *but incorporates different mesh generation techniques as Expert Meshing as well as a mesh generator based on the* **Acute angles method** *previously presented* **but** *here* **particulary adapted to mesh electrical machines.**

EFR

4.1.3. EFR. Regular Mesh Pre-Processor

This program is intended to generate, in a very simple way, a case in which the geometric description can be accomplished by a **regular mesh**. The finite element mesh generated through this program has only **quadrilateral elements,** which normally provide **very accurate results**. It is therefore advisable to use **EFR** whenever possible. Cases generated by **EFR** can be read directly by the solvers, **without the need** for processing by **EFM**.

Initially the program **asks if it is a new case or an old one**. If it is an old case, the name of the **.elf** file has to be typed. The scale factor and the type of grid, previously defined, are indicated. The definition of grids is described below.

The first step in **EFR** is to insert the grid of finite elements, if it is a new case. There are **two types of grids**: Cartesian and Polar.

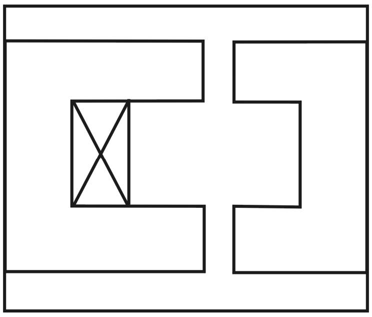
The procedures and commands in **EFR** are now described.

a. Insertion of grids.

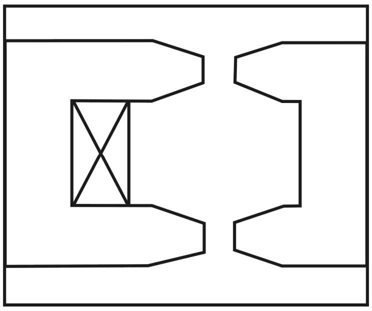
Initially the type of coordinates (Cartesian or Polar) and scale factors are required. Answering this question the user types the key **<Enter>** for **Cartesian** coordinates. Notice that **very often**, in **EFCAD**, the **most common options** can be inserted by typing **the key <Enter>** (rather than a specific letter) for facilitating and speeding the dialogue.

a1. Cartesian grids.

First we point out that the example in the figure *a* can be defined by **EFR** since the **lines** describing the structure **are parallel to the axes *Ox* and *Oy***.



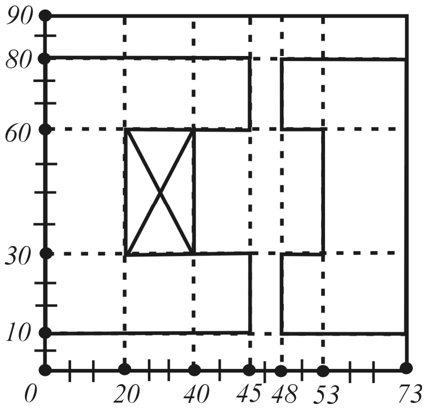
a - **EFR** *can be used.*



b – **EFR** *can not be used*

Figure *b* shows a similar case in which the iron contour **lines are not parallel** to the axes and therefore the geometry **cannot be defined** by **EFR**.

Suppose that the structure in the figure *a* is discretized using the grid shown in the figure *c*, which also shows the dimensions of the structure.



c - defining the rectangular grid

Note that in the *x* direction, the abscissae defining the structure are 0., 20., 40., 45., 48., 53., and 73.. For each interval between two abscissas, **we have to impose a number of subdivisions** in order to create the grid; for example, between 0. and 20. there are 3 divisions and between 45. and 48. (the airgap) we impose two divisions.

The way to insert the grid is, initially, to provide **the number of divisions** (integer numbers) , **finishing** the sequence with a **zero**, which signifies that all data were entered. In the example above we type:

3 (for the section between 0. and 20.)
3 (for the section between 20. and 40.)
2 (for the section between 40. and 45.)
2 (for the section between 45. and 48.)
2 (for the section between 48. and 53.)
3 (for the section between 53. and 73.)
0 (to **stop** the sequence)

EFR will now ask for **seven values** of the abscissae. In response we type:

0.
20.
40.
45.
48.
53.
73.

The **same operation will be repeated for the *Oy* direction**, which for the problem described here is:

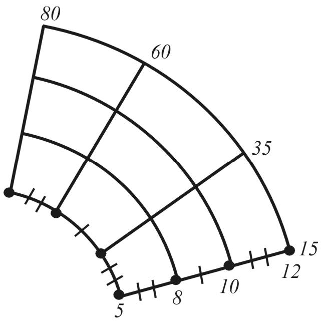
2 (section between 0 and 10)
3 (section between 10 and 30)
4 (section between 30 and 60)
3 (section between 60 and 80)
2 (section between 80 and 90)
0 (to stop the sequence)

Followed by the 6 ordinates:

0.
10.
30.
60.
80.
90.

a2. Polar Coordinates

Entering polar grids requires operations similar to those described above. Figure below shows an example of a polar grid.



a Polar case and grid

In the radial direction, we type the **number of divisions**

3 (between radius 5 and 8)
2 (between radius 8 and 10)
3 (between radius 10 and 12)
0 (to stop the sequence)

and the **four values** of the radii are:

5.
8.
10.
12.

In the **Theta (tangential) direction** we type:

3 (between angle 15 and 35 degrees)
2 (between angle 35 and 60 degrees)
3 (between angle 60 and 80 degrees)
0 (**to stop** the sequence)

and the **four angle values** (in **degrees** and using polar (trigonometric) orientation) are:

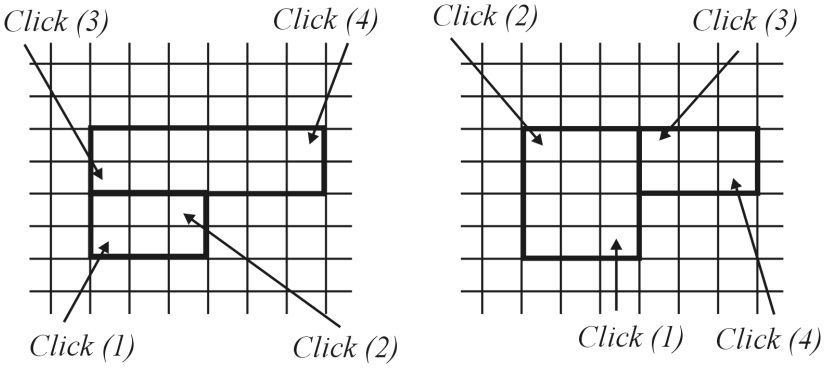
15.
35.
60.
80.

It is important to note that **it is not possible to insert the value zero for the radius**, since the quadrilateral elements would be deformed at this point.

b. Material (message: "Mater")

After the grid is displayed on the screen, the user selects the menu in ***"Mater"***. At the **bottom of the screen**, he/she enters (types) the number of the material. It is similar to **EFM** procedures.

The material will be inserted in the domain **by rectangles** (in the case of Cartesian coordinates) as shown in the Figure.



Clicking: two different and equivalent ways to insert a material.

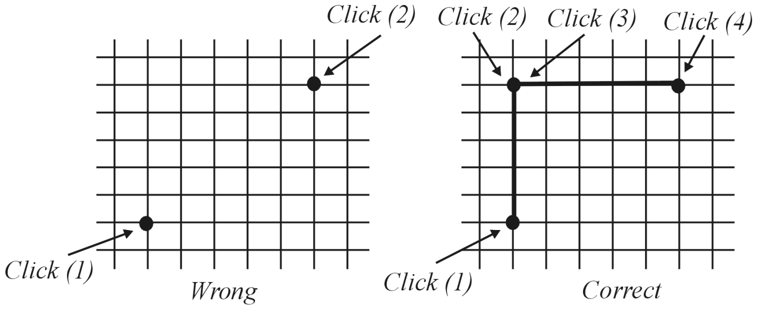
The mouse is clicked in the extreme corner **elements** defining the region to be selected **rather than the corner nodes**. The **default material is material No. 1**. If a mistake is made, or **it becomes necessary to make changes** to the material distribution, the user **can redefine the material** in the region in which the change is needed by defining the new material in the same way as described (or using material No. 1 for the default).

c. Heating and Current Sources (Message: "Q-term" and "J-curr")

The procedure **to insert sources** is exactly the same as described above. The **number of the source is typed by the user at the bottom of the screen** and the insertion is made as for materials. If any **source is incorrectly entered**, the user can **define a source number zero** and insert it where necessary.

d. Electromagnetic and Thermal Boundary Conditions (Messages: "electrm bC" and Therm bc)

The procedure to input potential (Dirichlet) is very similar to that described in **EFM, and only the EFR particular aspects** of this inserting will be **described here.** The major difference is that the applied potential lines (on the boundary or inside the domain) **must be made of a set of lines,** connecting **one node to another**, without crossing over elements. Figure below shows this method of insertion.

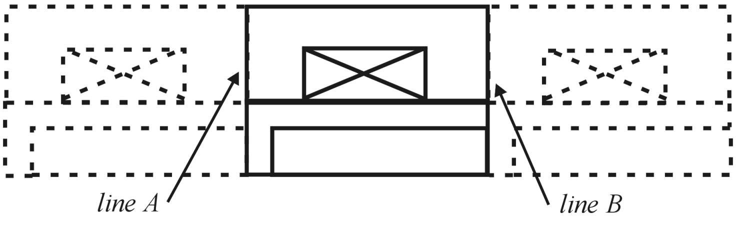


Wrong way to insert boundary cond. *Correct way*

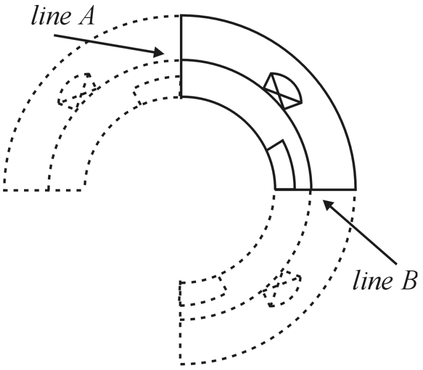
The elimination of boundary conditions and insertion of thermal boundary conditions are similar to those described in **EFM.**

e. Periodicity and Anti-Periodicity (message: "Perid(x/t)" and "Anti-per")

The periodicity and anti-periodicity conditions in **EFR** are **only admitted** if they are specified between two limiting lines **parallel to the Oy axis** (for Cartesian coordinates) or between two limiting **radial lines** (for Polar coordinates) as shown in the following figures.



(anti) periodicity in Cartesian coordinates



(anti) periodicity in Polar coordinates

The periodicity and anti-periodicity condition is established between the lines A and B. These lines are identified by **EFR automatically.** The user only **needs to click on the "periodicity" or "anti-periodicity" option** in the menu. These options are displayed as a sub-menu for **electromagnetic boundary** conditions.

h. listing file Efmat.dat (message: "eFmat.dat")

Using this option, a numerical display of the file **efmat.dat** appears on the screen. This option is useful when, for example, the user, before inserting the materials inside the region, wants to be sure of the correct number of the medium.

f. End (message: "End")

When entering this option, the user is required to give a file name in which the problem description is stored. It is always a **.elf** file. If the file **already exists**, it will be **overwritten** by the new data.

EFR can also create a file **.pre** (if the user wishes it) which contains the drawing of the structure. The lines outlining the blocks of materials and sources as well as the external contour will then be shown in file **.pre**.

The files **.elf** created by **EFR** are compatible with program **EFM**. Only the **boundary conditions have to be redefined** again in **EFM** and, obviously, the discretization must be done in **EFM.**

EFCONV

4.1.4. EFCONV – Converting files created in AutoCad to .pre EFCAD files

This program converts the drawing files **.dxf**, made by professional drawing packages (such as AutoCad and VersaCad) into files **.pre** as defined in section 2. When calling **EFCONV**, only the names of the original **.dxf** file and the new **. pre** file are needed.

Using AutoCad to generate the drawing (for example), is quite useful if the user is familiar with this particular software and with the finite element method. It is however important to **take into account the remarks made section 4.1.1** regarding, for example, the connection of points, crossed and superposed segments, etc. In our experience it is not very effective to use a drawing file, **which is intended primarily for display purposes**, to generate the file **.pre**. A good procedure is to generate the file

.dxf while being aware of the necessary conditions needed to properly generate the regions and the mesh, as required by the finite element method.

MESH

4.1.5. MESH – Identification of Mesh Elements and Nodes

This program enables the graphical identification of mesh nodes and elements. **MESH** asks for a **.elf** file generated by pre-processors **EFM** or **EFR**. With the menu appearing at the top of the screen three operations can be performed:

a. *Nodes visualization (message: "Point")*
Using this option, the node given by the user is shown;

b. *Elements visualization (message: "Mesh el.")*
The user types an element number and **MESH** shows this element in the mesh;

c. *Indication of mesh element number (message: "Mouse")*
Using the mouse the user click on the mesh presented on the screen and the program will inform its number.

4.2. THE SOLVERS EFC's - (General Introduction)

The solvers are named **EFCxx**. The letter **C** stands for "calculation" and is followed by another letter (or two letters) in order to define what the solver is supposed to perform. **The user decides which solver** has to be used accordingly with the **case under analysis**.

Using **EFCs**, the most **common options** can be **answered** by just **typing <Enter>** to speed the dialogue between the software and the user.

With the material numbers defined in the elements, the electromagnetic properties are read from file **efmat.dat** and the iron losses properties from file **eflos.dat**. When using the solvers for **thermal cases**, the properties will be read from the file **efterm.dat**.

After furnishing the data for the solvers **EFC's**, the Finite Element calculation is performed. When finished, a "flag" variable defining which solver was employed as well as the results (potentials at the nodes) are written in the same file **.elf** after the entry data read by the solver. If it is a case with a set of potentials (as the step by step case), the results (potentials for each time step) are written successively.

Remark: *an important remark is that current sources (windings or relied conducting regions) can not touch **(even partially)** periodic or anti-periodic boundary conditions.*

EFCS

4.2.1. EFCS – Calculation for Static Cases

EFCS is the finite element solver for electromagnetic static solutions. The **descriptions below are valid for all solvers except when it is indicated**.

Name of the file: If the user wishes to solve the **last case defined by the pre-processor**, it is not necessary to type the file name for the case; **just type <Enter>**. The name of the last created file will be opened.

Mesh drawing: The program will ask if drawing of the mesh is required. **If not, press <Enter>;** this option is useful when the user is not sure about the file name of the case under analysis.

Type of potential: When **no sources are detected**, **EFCS** requires **specifying if the case uses vector or scalar** potential formulation. **Having source(s),** necessarily the electromagnetic problem in **EFCAD uses magnetic vector potential**, which is automatically detected by the program.

Coordinate System: The program also asks if the problem is Cartesian or axisymmetric. If the case uses **Cartesian coordinates, press <Enter>.**

Linearity: Next the program asks if the problem is linear or not. **If linear, press <Enter>**. If **non-linear**, the user **must supply the number of iterations**, which is the next question. The **Newton-Raphson method** is applied and under normal conditions, it **converges quickly** to a very accurate result. Note however that the **first five iterations** are **performed using the "successive approximation"** method. It is not advisable to set a high number of iterations since the convergence using the Newton-Raphson method is fast. If **convergence is not achieved** in up to 20 iterations, the **program switches to the "successive approximation**" method **automatically**, starting from the beginning. This usually converges to an accurate result (however slower than the Newton-Raphson method). The convergence criterion is based on the relative error, which must smaller than 0.001 for all nodes. The **solver automatically sets** this criterion. During the iteration, the sum of the potentials at the nodes and the average error found for the iteration appear on the monitor. However, because computers are now very fast, it is difficult to follow these messages in the screen.

Current Sources: When current sources are detected, **EFCS** indicates their number (which were inserted in the pre-processor step) and **requests the current density value** in the corresponding sources. Although **EFCAD** is based on the MKSA units, the value of **J has to be provided in A/mm^2**, for convenience.

EFCC

4.2.2. EFCC – Using complex formulation for linear time harmonic cases

This solver uses the complex time harmonic formulation for sinusoidal sources and under linear conditions. The vector potential is employed and eddy currents are calculated.

Except for the considerations related to the linearity (which is implicitly assumed), all other considerations for **EFCS** at section 4.2.1. are valid for this program as well.

EFCC requires, in a straight dialogue, the **number of frequencies** and **their values**, which should be **entered successively, and separated by commas**. For the sources, the values of **J** and the **Phase** have to be provided to **EFCC**.

The solution will be performed **successively** according to the **number and values of frequencies entered.** These solutions are then added to file **.elf** corresponding to the cases solved.

EFCJ

4.2.3. EFCJ – Solving nonlinear, time stepping, eddy currents cases fed with density current defined in the sources

This module solves transient cases, including nonlinear materials (if these exist) eddy currents, and uses a step-by-step time procedure. **EFCJ** assumes that sources are current-fed and that the values of **J** are imposed data to be furnished. Vector potential is used here.

The data entry topics already discussed in section 4.2.1 are valid for **EFCJ**. However, the **initial and final time** of the simulation, as well as the **time step** are also required by the program. The program also asks for **an interval for writing** on file **.elf**. If this **interval is zero**, **all time steps** will be written into the file. If the interval is **"1"** (for example), for **each two time steps** one will be written. It is useful when the simulation is very long it is not advisable to write all the potential solutions since then file **.elf** will become very large. Notice that it is **maybe necessary, for accuracy purposes**, to use a very small time step, but, normally, **the user does not need** all the solution values for his/her analysis.

The user is also **required to specify how J varies** as a function of time. If it **is sinusoidal, the amplitude (**$J_{\max}$=A/mm^2**), frequency (**f**) and phase (**φ**)** of the source must be entered; the current density in calculation will then be considered as $J_{\max}\sin(2\pi f+\varphi)$. If **J** varies in a different way, the user **supplies the pairs (t,J)** describing the temporal behavior of **J**. The sequence of points (t,J) **is completed with a (0,0).** During the simulation, the values of **J** will be interpolated between given points.

Finally the program asks if the user wishes to create a file for **Eddy Current Losses**. These losses are concerned with **eddy currents** and are calculated in **all materials with an non-null electrical conductivity**. The total losses are stocked in a **.des** file. This file can be read by the software **DSN** developped by the researches of LEEI – Laboratoire d' Electrotechnique et d'Electronique Industrielle de Toulouse.

EFCT

4.2.4. EFCT– Solving nonlinear, time stepping, eddy currents cases fed with voltage defined in the sources

EFCT is similar to **EFCJ** but the coils are voltage excited. In this case, the currents established in the coils **are also unknown** and are solved simultaneously.

Consider the following electric circuit equation(s) for a **motor** operation

$$RI + L\frac{dI}{dt} + n\frac{d\Phi}{dt} = V$$

The **values of coils resistance** *R* and the additional **inductances** *L* (end-windings inductance) are also required by **EFCT**. The number of **turns** *n* in the sources and the **structure depth** (in meters) must also be provided by the user. In the case of an axisymmetric device the depth is automatically considered in the formulation.

The temporal behavior of *V(t)* is entered exactly as *J(t)* is entered in **EFCJ**. Finally, two additional files are automatically created by **EFCT** to describe *V(t)* entered during the step by step procedure as well as the **calculated values** of *I(t)* in the coils. If the data file is named, for example, **aa.elf**, the two new files will be named **aat.plt** and **aac.plt** for the **voltage and current respectively**. These files **will be rea**d in the post-processor step.

This solver asks if the user wishes to create two additional files (for "current and voltage"). Type **yes** if you want to stock the exciting voltage and the calculated current waveforms in a **.des** format file. As **EFCJ**, this solver also asks if the user wishes to calculate **eddy current losses** in conducting parts and the name of the corresponding **.des** file.

EFCV

4.2.5. EFCV – Solving eddy current created by conducting materials with velocity (movement)

This solver is used for vector potential applications in which there are conducting materials **moving at constant velocity v.** The eddy currents calculated from vector product between the quantities **v by B**. The velocity of the materials **is read from** file **efmat.dat**. The sources are fed by current. For the other data entry and remarks, see description of the **EFCS** above.

EFCF

4.2.6. EFCF– Solving nonlinear, time stepping, eddy currents cases fed with voltage defined in the sources and relied thick conductors

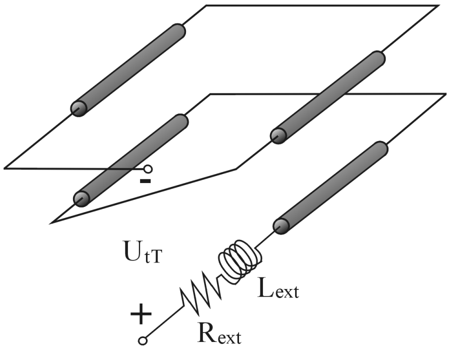
EFCF is similar to **EFCT** (where the coils are voltage excited) but, contrarily to the latter, conducting regions (or **thick conductors having eddy currents)**, can be serial or parallel relied. With **EFCF only cartesian** cases can be calculated and the data entry topics already discussed in section 4.2.4 are valid for **EFCF**.

Two different types of conductors can be presented in the device:

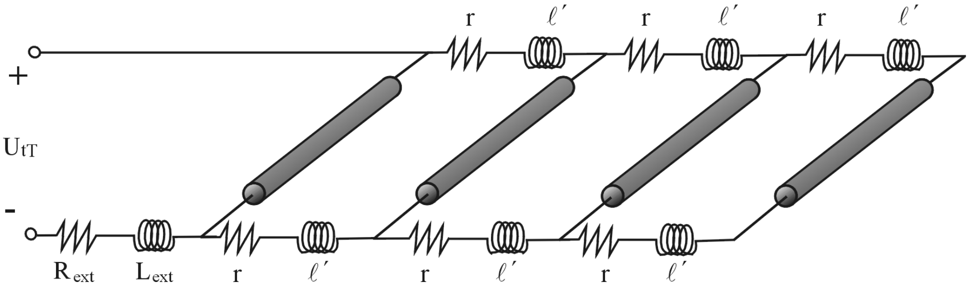
a) **thin conductors**: conductors in which skin effect is not presented;
b) **thick conductors**: skin effect is considered

EFCF recognizes a thick conductor when a **source number** is detected (it was considered as a source in the pre-processing stage) but **also a non-null electrical conductivity**. **Thick conductors must be numbered firstly of the thick conductors** coils in pre-processing. Another important point: **thick conductors cannot be (even partially)** on periodic or anti-periodic boundary lines.

Serial and parallel connection of thick conductors can be fed by means of an applied voltage U_{tT} in series with a R_{ext}, L_{ext} circuit as shown in figures below. If the conductors are parallel connected, the resistance *r* and inductance ℓ' values of the interconnection are also required and considered in calculation. Given small or large values for the interconnection resistance allow simulating short-circuited or non-connected thick conductors.



Thick conductors serial connected



Thick conductors parallel connected

For serial connection of thick conductors the sources numbers associated to them must be given sequentially with alternating + and – signs. For instance, let us suppose that the thick conductors associated to source numbers 1, 2, 3 and 4 are serial connected. In this case, in pre-processors **EFM** or **EFMA,** these sources must be given as +1, -2, +3, -4 or –1, +2, -3, +4. If the sources have all the same sign, for instance +1, +2, +3, +4 calculation will be made supposing parallel connection.

As for thin conductors, the questions are the same as the ones required by **EFCT** but here, the currents **.des** file will contain the total current in the thick conductors and the currents in the thin conductor windings. The user also supplies another **.des** file name for the induced voltages in the thick conductors.

As for **EFCJ** and **EFCT** the program also asks if the user wishes to create a **.des** file for **Eddy Current Losses**. These losses are related to the **eddy currents** and they are calculated for **all materials with a non-null electrical conductivity.**

EFCR

4.2.7. EFCR – Calculation for Static Cases with rotation

EFCR is similar to **EFCS** but allows studying electromagnetic structures with a **rotating part as, for instance, electrical machines**. Rotation is automatically taken into account using a moving band with **quadrilateral elements** located in the air gap. The rotation is always applied to the inner part of the machine and this rotation is always performed in the counterclockwise direction. **EFCR** needs a machine domain delimitated by (anti) periodicity boundary conditions (see pre-processor programs **EFM**, **EFMA** and **EFR**).

For defining the moving band, it is necessary to insert it (in pre-processing stage with EFM or EFMA) as the material number 25. This material 25 must be defined in module EFP as air (permeability μ_0). The moving band should be regularly meshed with quadrilateral elements. This can be made by imposing divisions in the mesh generation modules **EFM** or **EFMA** using the option **ANY**. However it is possible to choose a rotation step different of the mesh step (tolerating a finite element distortion located inside the moving band).

The field sources may be permanent magnet and/or currents. The magnetization of the magnets may be radial/tangential or may be given in terms of cartesian components in module **EFP**. The currents are imposed but they may vary as indicated by the user without any restriction terms of position: sinusoidal, rectangular, or any other waveform giving the curve by points. It is also possible to enter a current density curve indicating the fundamental amplitude and phase as well as the harmonics rank and respective amplitude and phase. This curve can be furnished from the keyboard or by means of a ***.dea** file. This file, is an ASCII data file with the following structure:

1,npoints
1,j(1),phase(1)
rank(2),j(2),phase(2)

⋮

rank(npoints),j(npoints),phase(npoints)

where *npoints* is the number of points; *,j(1),phase(1)* are the fundamental amplitude and phase and *rank(i),j(i)* and *phase(i)* are, respectively, the harmonic *i* rank, amplitude and phase.

The calculation, corresponding to a succession of static states, is possible for linear or non-linear cases.

The results are **flux**, self and mutual **inductances**, **e.m.f**., **torque** (by different methods), **flux density in different points** (regions) of the structure and **iron losses** estimation for all the positions. With exception to iron losses, the other calculated quantities above are stocked in **.des** files. In the case of **flux density**, the calculation is performed for different materials defined by the user in **EFM** or **EFMA** pre-processors. The corresponding materials regions should be sufficiently small to act as “probe” regions. For **iron losses,** it should be noticed that, according to the mathematical and physical formulations used in **EFCAD**, the calculation **must be made through a complete electrical period**. The three loss components are considered: **hysteresis**, **eddy current** and **anomalous**. Iron Losses coefficients (**Watts/kg**) are read from file **eflos.dat**. The iron losses results are stocked in a **.los** file. This file will be read by **EFCMOUT**.

If the calculation of torque is related to a case where the rotation step does not match the mesh step (creating distorted elements), it is necessary to create a **second band** in the airgap with the same regular mesh as the moving band. Attention is to be paid when using these two bands: **the internal one must be** defined as the moving band. The torque will be automatically calculated (by Maxwell’ s stress tensor) inside the second (external) band where there are no distorted finite elements). This method gives better results compared to calculations performed with only a moving band.

Attention:

a) *the machine coordinates must have as reference the point (x=0,y=0), i.e., the center of the machine*;
b) *the domain must be drawn in counterclockwise direction from a horizontal line starting at the point (x=0,y=0).*

EFCR360

4.2.8. EFCR360 – Calculation for Static Cases with Rotation

EFCR360 allows calculations for a whole machine (domain takes 360 degrees).

Here the movement is also taken into account using quadrilateral finite elements in the airgap. This module, except for the whole domain, is identical to **EFCR** and we recommend the reading of instruction at the **EFCR** section.

Attention:

a) *the machine coordinates must have as reference the point (x=0,y=0), i.e., the*

center of the machine;
b) *the domain must be drawn in counterclockwise direction from a horizontal line starting at the point (x=0,y=0).*

EFCM

4.2.9. EFCM– Solving nonlinear, time stepping, eddy currents cases fed with voltage defined in the sources and taking into account rotor movement

This solver can be understood as an extension of **EFCT,** adding here the movement. Please read also the information given above for **EFCT** and **EFCR** since the rotation is performed in **EFCM** also with a moving band. Notice that with the moving band method, the machine structure **must be** defined using (anti) periodic boundary conditions.

In **EFCM** the winding voltages can be sinusoidal or given by a set of points. In the latter, if the voltages are periodic (but not having sinusoidal shape) the user furnishes the points for a single cycle and the number of cycles. It is also possible to furnish the points of the single cycle by means of a file with extension **.dea**. This file, is an ASCII data file with the following structure:

```
1,npoints+1
t(1),v(1)
t(2),v(2)
    ⋮
t(npoints),v(npoints)
0,0
```

where *npoints* is the number of voltage points and *t(i),v(i)* are, respectively, the time and voltage values for point *i*. Notice that the end of the file must contain *0,0*.
Similarly to **EFCT**, the **values of coil resistances** *R* and the additional **inductances** *L* (end-windings inductance) are also required by **EFCM**. The number of **turns** *n* in the sources and the **structure depth** (in meters) must also be provided by the user as well as the **number of symmetrical sections** needed to represent the whole machine. For instance, if the calculation domain of the machine is 45 degrees, the number of symmetrical sections is 8.

The calculated winding currents and the imposed voltages are also stocked in **.des** files. In **EFCM** the windings can be independently relied or *Y*-connected (with or without neutral wire) or in *triangle*. For *Y without neutral wire* and *triangle* connections, the voltages to be furnished are the **phase-to-phase voltages**. For *Y with neutral wire* the voltages are the **phase-to-neutral voltages**. In the case of *triangle* connection, the user furnishes also the name of **.des** file where the line currents will be stocked.

EFCM can also calculate the **e.m.f** induced in the machine windings (**phase-to-phase e.m.f for a triangle connection and phase-to-neutral e.m.f. for a star connection**). If the user wishes such a result a **.des** file to stock these voltages must be furnished. Similarly to **EFCR**, eddy current losses in conducting materials and magnetic induction on different regions (materials) are made by **EFCM** and stocked in a file to be read by the software **DSN**.

The iron losses are stocked in a **.los** file in the same way of **EFCR** (please, see above).

EFCM also asks if the user wishes to create a **.des** file for **Eddy Current Losses** in all conducting parts.

The rotation angle with **EFCM** can be **imposed** (the rotor speed is given by the user) or **calculated** by means of coupling the following mechanical equation:

$$J_m \frac{d}{dt}\omega = T_e - T_L$$

where J_m is the rotor inertia, ω is its speed, and T_e and T_L are, respectively, the electromagnetic and the load torques. The load torque T_L curve as a function of time must be given by a set of points. For either imposed and calculated movement the electromagnetic torque is calculated but only in the last case the user furnishes a **.des** file to stock the rotor speed.

Attention:

a) the machine coordinates must have as reference the point (x=0,y=0), i.e., the center of the machine;

c) the domain must be drawn in counterclockwise direction from a horizontal line starting at the point (x=0,y=0).

EFCM360

4.2.10. EFCM360 – Solving nonlinear, time stepping, eddy currents cases fed with voltage defined in the sources and taking into account rotor movement

EFCM360 allows calculations for a whole machine (domain takes 360 degrees).

Here the movement is also taken into account using quadrilateral finite elements in the airgap. This module, except for the whole domain, is identical to **EFCM** and we recommend the reading of instruction at the **EFCM** section.

Attention:

a) *the machine coordinates must have as reference the point (x=0,y=0), i.e., the center of the machine*;

b) *the domain must be drawn in counterclockwise direction from a horizontal line starting at the point (x=0,y=0).*

EFCG

4.2.11. EFCG – Solving nonlinear, time stepping, eddy currents cases fed with voltage defined in the sources and taking into account rotor movement; solver for asynchronous motors

The instructions for using **EFCG** are identical to those of **EFCM**. **However** for **EFCG** each rotor bar **must** be filled with a different material number at the pre-processor step. All rotor bars **must** have the same shape and dimensions and the machine structure **must be** defined using (anti) periodic boundary conditions.

Contrarily to **EFCM**, where all the conducting materials are connected and short-circuited (2D approach), here the **bar-end** and **inter-bar** impedances of the short-circuit rings are considered in the calculation. The additional questions of **EFCG** concern the bars of the squirrel-cage rotor:

Number of rotor bars in FE domain: give the number of bars
Bar 1 Material: enter the material number of bar 1
Bar 2 Material: enter the material number of bar 2
⋮
Bar n Material: enter the material number of bar n
Bar-end resistance: give the bar-end resistance value
Bar-end inductance: give the bar-end inductance value
Short-circuit ring resistance (between two bars): enter the short-circuit resistance value
Short-circuit ring resistance (between two bars): enter the short-circuit inductance value
Periodicity factor: enter 1 if the calculation domain is periodic or –1 if anti-periodic
Bar currents file name: give the **.des** file name for the total current in the rotor bars

EFCG also asks if the user wishes to create a **.des** file for **Eddy Current Losses** in all conducting parts.

Attention:

a) *the machine coordinates must have as reference the point (x=0,y=0), i.e., the center of the machine*;
b) *the domain must be drawn in counterclockwise direction from a horizontal line starting at the point (x=0,y=0).*
c) *Rotor bars cannot touch* ***(even partially)*** *periodic or anti-periodic boundary conditions*.

EFCMF

4.2.12. EFCMF– Solving nonlinear, time stepping, eddy currents cases fed with voltage defined in the sources, connection between thick conductors and taking into account rotor movement

EFCMF is similar to **EFCF** (see above) but adding movement. As for the calculations with **EFCR**, **EFCG** and **EFCM** the machine structure **must be** defined using (anti) periodic boundary conditions. Please, read the instructions given previously for **EFCF** and **EFCM**.

The *triangle* or *Y* connections concern only the thin conductor windings. As for **EFCF** the currents **.des** file will contain **firstly** the total current in the thick conductors and **after** the currents in the thin conductors windings. The **e.m.f.** file will contain only the voltages induced in the thin conductors windings.

The calculated iron losses are stocked in a **.los** file as for **EFCR** (see above). **Eddy Current Losses** for the conducting parts can also be calculated.

Attention:

a) *the machine coordinates must have as reference the point (x=0,y=0), i.e., the center of the machine*;

b) *the domain must be drawn in counterclockwise direction from a horizontal line starting at the point (x=0,y=0).*

EFCMF360

4.2.13. EFCMF360 – Solving nonlinear, time stepping, eddy currents cases fed with voltage defined in the sources, connection between thick conductors and taking into account rotor movement

EFCMF360 allows calculations for a whole machine (domain takes 360 degrees). with the movement also taken into account using quadrilateral finite elements in the airgap. This module, except for the whole domain, is identical to **EFCMF** and we recommend the reading of instruction at the **EFCMF EFCF** and **EFCM** sections.

Attention:

a) *the machine coordinates must have as reference the point (x=0,y=0), i.e., the center of the machine*;

b) *the domain must be drawn in counterclockwise direction from a horizontal line starting at the point (x=0,y=0).*

EFCSL

4.2.14. EFCSL – Static case with several sources

As **EFCS, EFCSL** is a solver for static problems but it incorporates the possibility to solve simultaneously **several sources**.

It is possible to obtain results considering each source (current or permanent magnet) independently or combining several sources. Two types of simulation can be performed:

1. *Linear case*

For this case the user furnishes the number of the sources for each configuration The matrix solution considers the sources separately and add them as required by the user.

Self and mutual fluxes for the different sources configurations are automatically calculated and stocked in a file **.sol**. This file keeps the same name given for the currents file.

2. *Non-linear case*

In this case the calculation is performed for the all sources together.

The total flux linking the sources configurations is automatically calculated and stocked in the **.sol** file.

EFCTS

4.2.15. EFCTS – Solving thermal static cases

This module solves static thermal problems. The program has the similar characteristics of **EFCS**, except that the property file used is **efterm.dat**. The values for the **exciting sources must be provided in W/m^3 divided by 10^6** so that they are in accordance to the values of **J** given in A/mm^2. The solution is always **nonlinear** since the thermal conductivity varies with temperature. However, the convergence is normally **very fast, since** this nonlinearity is not strong (thermal characteristics have small variation with the temperature). The "successive approximation" method is used for the nonlinear iteration.

EFCTT

4.2.16. EFCTT – Solving transient thermal cases

This solver treats transient thermal problems. The module works similarly to **EFCJ** but the curve **Q(t),** which is the time dependent heating source, is given by points **(t,Q)** (units are: second and **W/m^3 divided by 10^6**) in the same way that **J(t)** is entered in **EFCJ**. The property file for this program is **efterm.dat**. This solver is **very useful** when one wishes to observe the **evolution of the temperature with the time**, while the solver **EFCTS** furnishes the temperature **at steady state**.

4.3. THE POST-PROCESSORS

EFGN

4.3.1. EFGN – General Graphical-Numerical Post-Processor

EFGN is the general post-processor in **EFCAD**. All solvers generate results that are treated and displayed either graphically or numerically by **EFGN.**

After entering the **.elf** file name (or <Enter> if the last file created is treated), **EFGN** displays the scale **factor used, number of nodes, number of elements, type of problem** (related to the corresponding solver), the **type of geometry** (Cartesian or axisymmetric), the **linearity condition** (linear or nonlinear) and **type of potential used**.

While **EFCTS** and **EFCTT** always use a scalar potential, **EFCC, EFCJ, EFCT**, **EFCF**, **EFCR**, **EFCR360**, **EFCM**, **EFCM360**, **EFCG**, **EFCMF**, **EFCSL** and **EFCV** use a vector potential.

For static electromagnetic solutions, (solved by **EFCS**) there are three distinct cases: a) There is a **current source** and a **vector potential** is used as the variable, b) There is **no source** and a **vector potential** is used and c) There is **no source** and a **scalar potential** is used. When **EFGN** detects no sources in the case produced by **EFCS**, it asks if the problem is ***electrostatic*** or ***magnetostatic*** so that it can proceed with the correct calculation of quantities and appropriate unities.

If there are **quadrilateral finite elements**, **EFGN** converts them **into triangles** by inserting new nodes at the barycenter of the elements. A message **"Expanded mesh"** gives the new number of nodes and elements, used exclusively for the post-processing.

EFGN has two displays: the first, called **"Graphical"** furnishes **mostly graphical results** (although numerical ones are also shown) and a second one called **"Numerical"** which allows the user to obtain **mostly numerical quantities** (although it often uses graphical means to obtain it).

When using the **Graphical Display,** the user **select the options he/she wants** and the program performs the drawings simultaneously. It can useful, however many drawings together can bring some confusion. The **best way** to observe graphical results is to **click on the option *"cLean"*** (for cleaning the screen) and choose the specific option(s) wanted; after doing so, the user **clicks on the option *"Draw"*** and the corresponding graphics will be displayed.

After listing the initial messages, by pressing **<Enter>, EFGN** displays the graphical display and shows a drawing named ***"Normal",*** in which the structure is shown **with 27 equipotential lines**. Sources and materials are shown in different colors. Permanent magnets and conducting materials are also shown in different, distinct colors. The minimum and maximum values of the equipotential lines are shown at the bottom of the screen.

For using this **graphical window**, the user **has to click the mouse on the chosen option(s)** or typing the **upper case letter** indicated in the messages of the menu, while the mouse can not be located over the menu.

The commands in **EFGN** are:

a. Normal Drawing (Message: "Normal")

Choosing this option, the user will see the structure and equipotential lines in the form described above.

b. Color shading (message: "sHading")

When this option is chosen, the structure is displayed and the required value is displayed **in a range of colors** representing the intensity of the respective quantity. After clicking on this option a sub-menu will be displayed; when treating **electromagnetics cases** these options are:

b1. *"Potent"* : The potentials will be displayed with colors

b2. *"Fields"* : the magnetic inductions or electric field (for magnetic or electrostatic cases, respectively) will be shown

b3. *"Eddy curr"*: if the problem has (eddy) induced currents, their density will be displayed

For thermal cases, the options appearing will be:

b4. *"Potent":* temperatures will be displayed

b5. *"Fields"*: the gradient of temperatures is shown

c. Equiline (message: "eQuiline")

The user may use this option to draw equipotentials, either fields, eddy currents and temperatures (if it is a thermal case). In this option, the **user can define the number of equipotential lines** and their separate values by typing them. If the user chooses the "default" option, **EFGN** draws 27 equipotentials between the minimum and maximum and minimum values of the required quantity.

d. Vectors (message: "Vectors")

EFGN draws **arrows** showing the direction of the fields. The **size of the arrow** is proportional to the magnitude of the field at the point where the arrow is being displayed. For **magnetic problems**, the arrow is related to the **magnetic induction** and for electrostatic cases (as well as thermal cases) the arrows are related to the **electrical field (**or the gradient of temperature**).**

e. Mesh and Structure (message: "Mesh/dom")

This option offers a sub-menu where the **user chooses between the mesh and the structure** for drawing. If the mesh is selected, the user can choose between **"normal"** or **"compact"** size of the mesh elements.

f. Printing (message: "Print")

This option is chosen when a print of the drawing is required. Color or black and white prints may be made, provided the appropriate printer is available. It creates a standard **.ps** file containing the drawing that can be printed by any printer with the appropriate software for **.ps** files. The name of the file is asked; it has to be furnished without the extension **.ps** .

g. Cleaning (message: "cLean")

It is important to note that the options selected by the user, accumulate on the screen. At some point, the screen must be cleaned. Clicking this command does this.

h. Drawing (message: "Draw")

After cleaning the screen (using command ***"cLean"*** above), the user can select which drawing(s) is/are required by clicking on the corresponding option in the menu. After this selection, the user must click on ***"Draw"*** to display the drawing on the screen.

i. Zooming (message: "Zoom")

Having a drawing displayed, the user may need to inspect a section of the drawing in more detail. Selecting this option does this. Clicking in two points defines the zoom window.

j. Anti-Zooming (message: "Continue")

Clicking this option **cancels zooming** and the drawing is displayed with the initial geometric limits.

k. Switching to Numerical Display (message: "numer/Exit")

This option switchs the **graphical display to the numerical** one, as defined above. It displays the "normal" drawing and the menu corresponding to numerical results. The messages below are related to the numerical display.

l. Exploration of Results by Points (message: "by Points")

This option furnishes the numerical value of fields at the bottom of the screen, as defined by the user. This can be done **using the mouse or the keyboard**. A sub-menu appears and the user chooses the way of defining the point. The results can also be expressed in **Cartesian or Polar coordinates**. These choices are also part of the sub-menu. Using the mouse the user has to click it on the point where the quantity is required. If the **keyboard is used** instead, the user **will type the coordinates** of the required point. When this is done, the permeability (or the permissivity) and magnetic **induction** or **electric field** or **temperature** (for magnetic or electrostatic or thermal cases, respectively) will appear at the bottom of the screen, together with the number of the selected point (a sequence of points can be selected by the user), **displayed over the structure**. Choosing the option ***"Print"*** in the sub-menu, a drawing of the structure with the number of the points will be printed, followed by the quantities calculated at the points.

For cases in **complex variables**, both the **real and imaginary** values are displayed. The display is in two steps. After the first part of the value is displayed, **the user must click the mouse again to display** the second part.

j. Curves of Quantities (message: "Curves")

To use this option, the **user defines a line** (usually by two points) on which the values are needed (**magnetic induction** or **electric field** or **temperatures** for magnetic or electrostatic or thermal calculations, respectively). The results will be displayed on a graph with the abscissa axis for **the length of the line** and the ordinate displaying the value of induction (or electric field or temperature). This line can be defined by the **mouse** or by **typing the coordinates** of the points (when the position of the line must be very precisely defined) through the keyboard. The line can be in **Cartesian coordinates** (and in this case, **the line between the two points is a straight** line) or Polar coordinates (an **arc, centered at point (0,0)** will be drawn). The messages are: ***"mouse X/y", "mouse R/tg", "Keyb x/Y" and "keyb r/Tg".*** The options using Polar coordinates are very useful if, for example, the magnetic induction in the airgap of a machine is needed. After a line is defined, the user clicks on option *"calcuLate"* and, **after providing the title** of the plot, the curve will be displayed.

k. Forces (message: "forceS")

Selection of this option allows the user to calculate forces. These are **calculated using Maxwell's stress tensor** in magnetic structure. To define the line (or lines) needed to apply this concept, the user proceeds as above. Then, after clicking ***"calcuLate",*** the magnetic force density is plotted; typing or clicking on ***"exit"*** the numerical values of the forces are written on the screen. If it is an **axisymmetric case, these results** are given in **Newton**; if the structure is **Cartesian**, the forces are given in **Newton/meter**, since the user must multiply it by the depth of the structure to obtain the actual force.

Also, **Laplace's force** acting on a conductor by the classical **expression (BIL)** (message : ***"biL/source"***) can be calculated. In this case, the **user must define the region** (by clicking inside it) where the source of current is placed and furnish the current density.

Laplace's force (same expression as above) can be calculated on materials having eddy currents, since external fields will create forces on the conducting materials having eddy currents (message is "***bil/eddY***"). In this case the user indicates **only** the corresponding region, since the eddy current density is calculated by the programs (**EFCC, EFCJ, EFCT**, **EFCF**, **EFCM**, **EFCM360**, **EFCG**, **EFCMF**, **EFCMF360** or **EFCV**). An arrow (or two for **EFCC**, showing real and imaginary parts) showing the direction of the force is drawn, as its magnitude appears in the bottom of the screen.

l. Flux Crossing a Line (message: "Flux")

As above, the user has to define the line and click on "***calcuLate***". The value of the flux crossing the line is then displayed on the bottom of the screen. For the unity, see the remark above.

m. Calculation of Inductance (message: "Induct")

After clicking on this option, **EFGN** will ask for the **number of turns** in the coil (only **one coil can be activated** at a time) and the inductance is calculated and displayed at the bottom of the screen.

n. Calculation of Magnetic Energy (message: "Mag energy")

Supplies the values of magnetic energy and co-energy in the structure. Both are displayed on the screen.

o. Torque Calculation (message: "tOrque")

It calculates the torque, when the **rotation axis is the point (0,0).** Useful option when we are dealing with electrical rotating machines. **EFGN** asks for defining a line along with the forces will be evaluated, which is normally the airgap line.

p. Additional calculation (message:“externAl”)

This option allows customizing **EFGN**. In this way these steps must be followed:

a) write (in Fortran) a file **nomfich.f** representing the calculations to be performed according the user’s need. The instructions to be employed are listed in the file **person.f** furnished with EFCAD;
b) tape **efgnmk nomfich** (without extension **.f**).

The customized version of **EFGN** named **EFGNP** will be created in the directory of the user. The user may check these procedures before customizing his/her **EFGN** version by using the command **efgnmk person.f**.

Example: suppose that **EFCAD** is installed in **c:\efcad** and a customized **EFGNP** is written in the directory **c:\custom**. In this directory two files must be created: **efgnmk.bat** and **makefile** with the following contents:

a) **efgnmk.bat**:

```
cls
@echo off
@if not exist %1.f goto stoppp
f2c -w %1.f
@if not exist %1.c goto stop
@if exist efgnp.exe del efgnp.exe

g77 -o efgnp.exe c:\efcad\efgn11.o %1.c c:\efcad\libefgn.a -lgks -
luser32 -lgdi32

@if not exist efgnp.exe  goto stopp
@echo ----------------------------------------------
@echo Bravo : votre efgn personnalise (EFGNP) a ete cree !
@echo ----------------------------------------------
@echo off
del %1.c
@goto fin
:stop
@echo ------------------------------------------------------------
@echo erreur de compilation dans %1.f :  Revoyez votre Fortran !
@echo ------------------------------------------------------------
@goto fin
:stopp
@echo -------------------------------------------
@echo il manque efgn11.o et/ou une biliotheque !
@echo -------------------------------------------
@echo ----------------------------------------------
@echo Attention : l'ancien efgnp.exe a ete efface !
@echo ----------------------------------------------
@goto fin
:stoppp
@echo --------------------------------------------------
@echo votre module fortran %1.f n'est pas accessible !
@echo --------------------------------------------------
:fin
```

b) **makefile**:

```
.SUFFIXES :
.SUFFIXES : .c .o .f
%.o : %.f
S = C:
CC = gcc
FC = g77
.f.c:
      f2c $<
.c.o:
      gcc -c -O -I$(S)/gcc-2.95.2/i386-mingw32/include $<
.f.o:
      g77 -c $<
SFEXE = .exe
LIBS = $(S)/gcc-2.95.2/i386-mingw32/lib/curgks.o -L$(S)/gcc-
2.95.2/i386-mingw32/lib -lgks -luser32 -lgdi32

efcad : efgnp$(SFEXE)

efgnp$(SFEXE) : /efcad/efgn11.o person.o
      $(FC) -o efgnp$(SFEXE) /efcad/efgn11.o person.o /efcad/libefgn.a
$(LIBS)
```

Notice that **EFGNP** can also be created by using the regular command **makefile** but, in this case, some troubles cannot be shown.

q. Displaying Voltage and Current Curves (message: "curRcurve" and "volTcurve")

This option is only available when the finite element calculation are performed by **EFCT** or **EFCF** (which creates files **.plt**: see section 4.2.4). When this option is selected, the values of ***V(t)*** and ***I(t)*** are listed numerically and then the curve requested is plotted.

r. Switching to Graphical Display (message: "graph/eXit")

Using this option the menu goes back to the graphical menu.

s. ***"End"*** is the option used to stop the program at any stage in **EFGN**.

Some remarks:

- Some options are only displayed when appropriate. For example, "magnetic energy" is not available for thermal problems or electrostatic calculations.
- For cases using complex formulations most of the messages have also an imaginary part.
- For problems having more than one set of potentials (cases calculated by **EFCC, EFCJ, EFCT**, **EFCF**, **EFCR**, **EFCR360**, **EFCM**, **EFCM360**, **EFCG**, **EFCMF**, **EFCMF360**, **EFCSL** and **EFCTT**) the user has to choose which case will be treated by **EFGN**. In this situation, when clicking on the option ***"graph/Exit"*** or ***"numer/Exit",*** **EFGN** shows a menu allowing the user to go to the ***"Numerical", "Graphical"*** *or* ***"new caSe".*** This last option can be used to go to a **new step** or **a new frequency** (if the complex formulation was used to generate the results).
- In the results given by **EFCT, EFCF, EFCR, EFCR360, EFCM, EFCM360, EFCG,**

EFCMF, EFCMF360, EFCSL and **EFCTT** the number of symmetrical sections and de depth of the device are considered **but EFGN does not take into account these parameters**.

EFCMOUT

4.3.2. EFCMOUT – Iron losses visualization

EFCMOUT is a post-processor to read **.los** files related to iron losses and generated by the solvers **EFCR**, **EFCM**, **EFCR360**, **EFCM360** and **EFCMF**. **EFCMOUT** displays the calculated iron losses density as well as the calculated losses values in the different materials as defined by the user.

4.4. THE MANAGEMENT OF PROPERTIES FILES

EFP

4.4.1. EFP – Management of the electric/magnetic properties file *efmat.dat*

This program manages the electromagnetic property file **efmat.dat** which is used in the electromagnetic solvers and by **EFGN**. Its use is very simple, not requiring long explanations. Most of the dialogues are displayed as simple questions without giving raise to ambiguous answers. The electromagnetic material properties are:

- ***perm***: the relative permeability or permittivity used in linear cases.
- ***Bcx, Bcy, Bcr, Bct***: the remnant induction of the permanent magnet in the direction Ox, Oy, radial and tangential (in *Tesla*).
- ***cond***: the conductivity (in $(\Omega.m)^{-1}$).
- ***velx, vely***: the velocity in the direction *Ox* and *Oy* (in *m/s*).

There are three types of materials:
- LIN - Linear materials
- SAT - For nonlinear materials (using the B(H) saturation curve).
- MAG - Permanent magnets.

The options in **EFP** are:

a. **Creation of a new file**: a new empty file **efmat.dat** will be created (the old file, if it exists can be **overwritten**).
b. **Listing of the materials**: a listing of the materials is displayed in two steps. First, **only the type and description** of the materials is shown. In the **second step, the material properties are listed.**

c. Modifications: the possibility of changing any property of the materials is allowed under this option. It is also possible to modify the curve B(H), excluding, including and discarding points of the curve.

d. Inclusion of a new material: it displays the possible free numbers for a new material.
e. Elimination of a material: the full description of the material is displayed before the user confirms its elimination.

f. Display the B(H) curve: lists the material properties of the material and displays the B(H) curve graphically.

A **practical rule** to follow, is to define material **"*1*" as air**, with relative permittivity and relative permeability equal to ***1***. Because the default material in the pre-processor step is also "*1*", it is very convenient.

EFTP

4.4.2. EFTP – Management of the thermal properties file *efterm.dat*

This module is the manager of the thermal properties file **efterm.dat.** It is very similar to **EFP** (described in section 4.4.1). The file **efterm.dat** is read by the thermal solvers **EFCTS** and **EFCTT**, and by **EFGN.**

The thermal properties of materials are:

- ***K*** - thermal conductivity.
- ***Kv*** - the variation of thermal conductivity.
- ***τ*** - the exponent defining the thermal conductivity variation.
- ***C*** - The thermal capacity
- ***Emis*** - Emissivity

The first three quantities are related to the equation of thermal conductivity as a function of temperature.

The options in **EFTP** are similar to the options described in **EFP**. These are:
a. Creation of a new file
b. Listing of materials
c. Modifications of material properties
d. Inclusion of new materials
e. Elimination of materials

EFPL

4.4.3. EFPL – Management of the iron losses parameters file *eflos.dat*

This module is the manager of the iron losses parameters file **eflos.dat.** It is very similar to **EFP** and **EFTP** (described in the above sections). The solvers **EFCR, EFCR360, EFCM, EFCM360, EFCG, EFCMF** and **EFCMF360** read the file **eflos.dat**.

The iron losses parameters to be furnished to **EFPL** are (see section A11):

- ***Ceddy***: eddy current losses coefficient (*Joules/kg*);
- ***Canom***: anomalous losses coefficient (*Joules/kg*);
- *Chyst*: hysteresis losses coefficient (*Joules/kg*);
- *Alfa*: hysteresis losses exponent;
- *Dens*: iron losses density (*kg/m^3*).

Another information concerns r_H, the relationship between purely rotating and purely alternating inductions (see section A11):

$$r_H = C_1 B^6 + C_2 B^5 + C_3 B^4 + C_4 B^3 + C_5 B^2 + C_6 B + C_7$$

where *B* is the magnetic induction. The coefficients $C_1 \cdots C_7$ must be furnished to **EFPL** as the seven r_H coefficients.

The options in **EFPL** are similar to the options described in **EFP** and **EFTP**. These are:
a. Creation of a new file
b. Listing of materials
c. Modifications of material properties
d. Inclusion of new materials
e. Elimination of materials

Appendix

A1. Introduction

In this appendix we present physical situations and examples that can be treated by **EFCAD**. Reading of this section is useful in preparation for use of **EFCAD** so that the various options and configurations solvable with this package may be understood.

For more in depth information on the topics discussed below we recommend reading the following text: *N. Ida and J.P.A. Bastos, Electromagnetics and Calculation of Fields,* indicated in the Reference section.

The quantities used in this text are:

H - Magnetic field intensity (*A/m*)
B - Magnetic flux density or induction (*T*)
E - Electric field intensity (*V/m*)
D - Electric flux density or induction (C/m^2)
J - Current density (A/m^2)
ρ - Charge density (C/m^3)
μ - Magnetic permeability (*H/m*)
ε - Electric permittivity (*F/m*)
σ - Electric conductivity $(\Omega m)^{-1}$
V - Scalar electric potential (*V*)
V_m - Scalar magnetic potential (*A.turns*)
A - Vector magnetic potential (*wb/m*)
T - Vector electric potential (*A/m*)

Maxwell's equations for low frequency applications are:

$$rot\mathbf{H} = \mathbf{J} \tag{1}$$

$$div\mathbf{B} = 0 \tag{2}$$

$$rot\mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \tag{3}$$

$$div\mathbf{D} = \rho \tag{4}$$

The constitutive relations are:

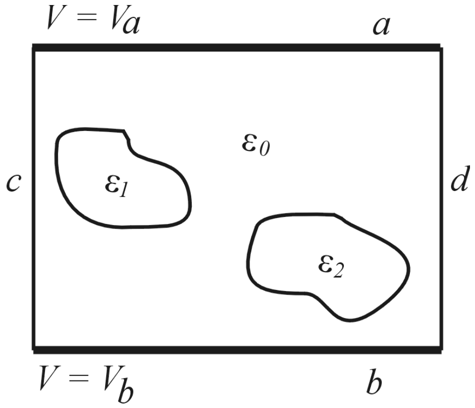
$$\mathbf{B} = \mu\mathbf{H} \tag{5}$$

$$\mathbf{D} = \varepsilon\mathbf{E} \tag{6}$$

$$\mathbf{J} = \sigma\mathbf{E} \tag{7}$$

A2. Electrostatic Fields - Dielectric materials - Scalar potential

The figure shows a domain with a dielectric material denoted by the permittivities ε. The two boundary lines **a** and **b** are related to the **imposed potentials** V_a and V_b. It is called **"Dirichlet boundary condition".**



Domain for electrostatic case – dielectric materials

The finite element method, as applied in **EFCAD**, works with potentials rather than directly with the fields. The fields are calculated in the post-processor **EFGN** from the potentials. For this problem, the scalar potential V (in Volts) is used as the unknown variable. V is related to **E** through:

$$\mathbf{E} = -gradV \qquad (8)$$

Maxwell's equation for this case is:

$$div\mathbf{D} = 0$$

since the fields are generated by the potential difference V_a–V_b rather than by charge density inside the domain. Using the relation:

$$\boldsymbol{D} = \varepsilon\boldsymbol{E}$$

and the equation above, we get:

$$div(\varepsilon\mathbf{E}) = 0$$

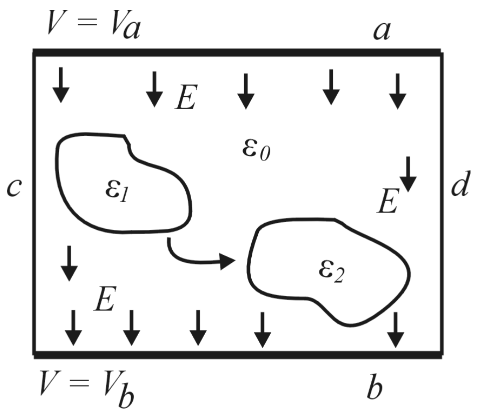
$$div(\varepsilon(-gradV)) = 0$$

which is Laplace's equation:

$$\frac{\partial}{\partial x}\varepsilon\frac{\partial V}{\partial x} + \frac{\partial}{\partial y}\varepsilon\frac{\partial V}{\partial y} = 0 \qquad (9)$$

This is the equation solved by the finite element method. Once this equation is solved, the post-processor is able to calculate the fields at any point of the domain. Using the **scalar potential, the boundary condition** will be related to the field as follows:

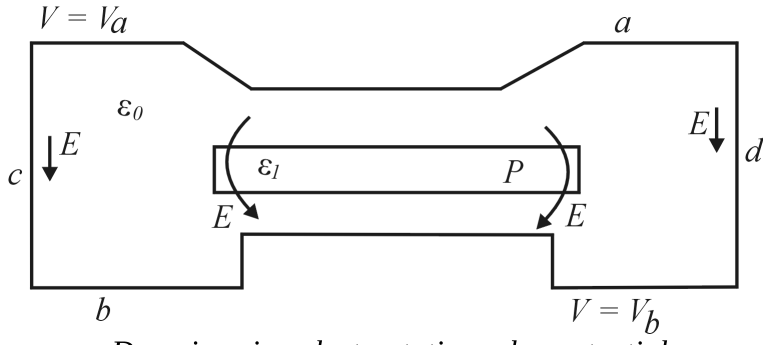
- On the surfaces on which **constant values of** V are applied (line **a** and line **b** in next figure) the field **E will be perpendicular to the equipotentials** V_a and V_b as shown in the figure.



Electric field at the boundary lines

On the remaining surfaces, (**c** and **d** in the figure above) where the user specifies no condition, the **field is parallel** to the surfaces. This condition is known as **"Neumann boundary condition".** These two conditions, Dirichlet and Neumann conditions, always refer to boundaries on which potentials are applied and on which no conditions are applied, respectively. The behavior of potentials and boundary conditions described above is **similar in all problems solved by scalar potentials**.

As an example, consider the domain shown in the figure below.

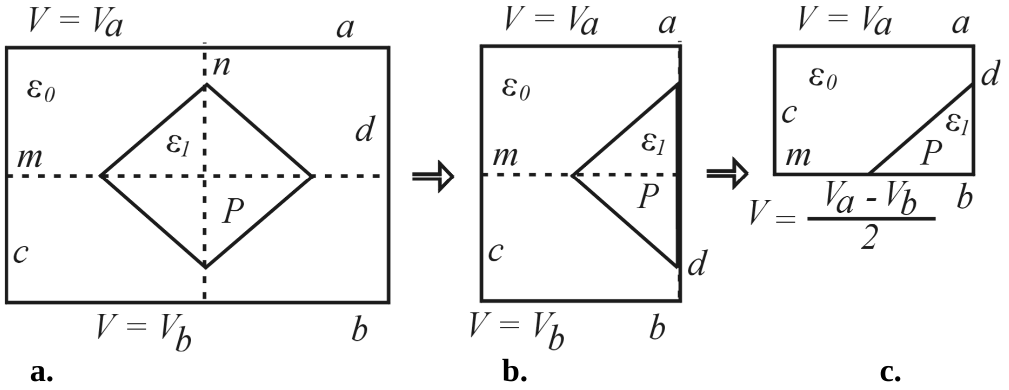


Domain using electrostatic scalar potential

On lines **a** and **b**, the potentials are V_a and V_b. A piece of different material, *P*, with permittivity ε_1 is embedded in the material of permittivity ε_0.

The geometry of the structure is easily entered using, for example, the program **EFD**. However the question is: **where to locate** the lines ***c*** and ***d*** ? One notices that the geometric variation around the edges of the piece ***P*** and the shapes of the lines ***a*** and ***b*** generate a perturbation of the field **E**. From the discussion above, the field lines must be parallel to lines ***c*** and ***d*.** Thus, if lines ***c*** and ***d*** are defined too close to the piece ***P***, we are forcing **E** to be vertical in a region where we know it is not vertical. The **solution is to move** the lines ***c*** and ***d*** further away as shown in figure above. In this case, the **numerical solution will match the physical situation**, and the field will be properly calculated.

Another situation where the Neumann condition can be used is shown in the figure **a** below.



Physical situations where, by symmetry, the domain can be reduced

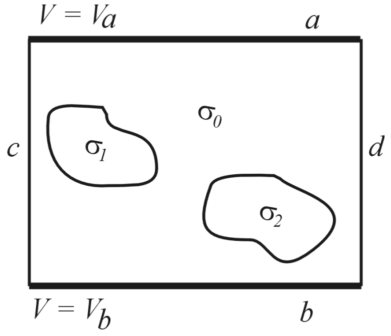
Noting that there are **two symmetry lines** in the domain, the problem can be reduced into one quarter of the original geometry. First, on line ***n***, the field is **parallel** to the symmetry line. Therefore, this line can be defined as a **Neumann condition**, as shown in the figure **b**.

Furthermore, because the field is **perpendicular** to line ***m*** (because of the second symmetry) the potential on this line will be $(V_a – V_b)/2$ and **this line can be redefined** as line ***b***, with applied potential $(V_a – V_b)/2$.

The **correct application** of boundary conditions is **critical**. The geometry of the device, dimensions, material properties and values of sources are **known directly** from the application. However, the choice of lines with **applied or implied boundary** conditions requires **some reflection and common sense**. In the examples above, it was pointed out how to correctly define the boundary conditions. The example in the last figure shows how it is sometimes **possible to reduce the domain** so that the solution becomes **faster** and the associated **file .elf** is **smaller**.

A3. Electrostatic Fields - Conductive Materials - Scalar Potentials

The physical configuration is shown in the figure below, where two potentials V_a and V_b are applied on boundaries ***a*** and ***b***. The domain contains only conducting materials.



Domain for electrostatic scalar potential – conductive materials

The equation to be used here is obtain from Eq. (1):

$$div(rot\mathbf{H}) = div\mathbf{J}$$

Because the divergence of the curl of any vector is identically zero, we will work with the electric continuity equation:

$$div\mathbf{J} = 0 \tag{10}$$

Defining the electric scalar potential as in the previous example by:

$$\mathbf{E} = -gradV$$

and Eq. (7) **J**=σ**E**, we obtain:

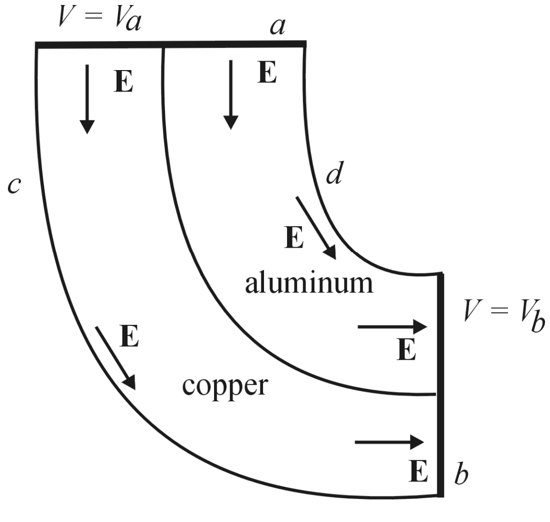
$$div(-\sigma gradV) = 0$$

or:

$$\frac{\partial}{\partial x}\sigma\frac{\partial V}{\partial x} + \frac{\partial}{\partial y}\sigma\frac{\partial V}{\partial y} = 0 \tag{11}$$

Looking at the Laplace's equation (in Eq. (9) or (11)) we note that σ and ε have a similar role. Here we also **point out the necessary and important fact** that in file **efmat.dat**, for the solution with materials as described here, the value of **conductivity must be inserted at the location of the permittivity (or permeability)** since the conductivity here plays the same role of the permittivity or permeability for electrostatic (with dielectric materials) or magnetic equations.

As a practical example, consider the domain of the figure below.



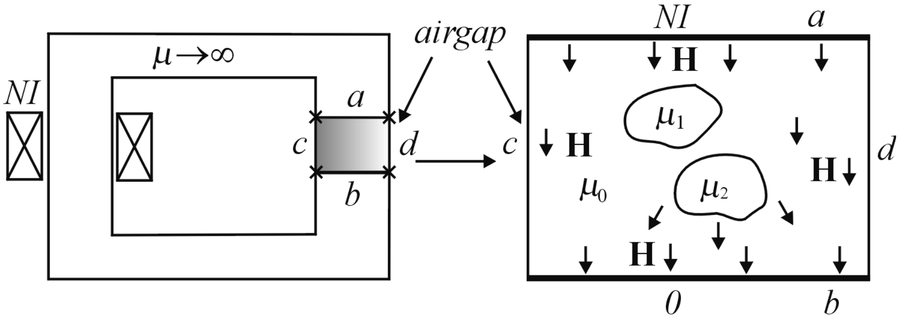
Domain for electric scalar potential – conductive materials

Two conducting materials (aluminum and copper) form a body that is subjected to a potential difference V_a–V_b. Lines ***a*** and ***b*** are the **Dirichlet boundaries** on which **E** (and therefore **J**=σ**E**) is **perpendicular**. The lines ***c*** and ***d*** are taken as **Neumann boundary** conditions since **J** must be **parallel** to these lines.

A4. Magnetostatic Fields - Scalar Potential

Firstly, we point out that in this formulation it is not possible to consider current sources in the solution domain.

In the figure **a** below, there is a magnetic circuit, made of an exciting coil, magnetic material and airgap.



a – *domain for magnetic scalar potential*

Assuming that **the permeability of iron is very high**, the magnetic field in this material can be neglected. This means that the magnetomotive force (*mmf*) *NI* appears between lines ***a*** and ***b***. The fields on lines ***c*** and ***d*** remain parallel to these lines (see figure above). This last assumption **depends very much** on the shape of the airgap (if the boundary length ***c***<<***a*** this assumption is good), and on the distribution and location of magnetic materials inside the airgap. Assuming this situation, **we can study only** the domain in the airgap. The basic Maxwell equation is:

$$div\mathbf{B} = 0$$

Defining the magnetic scalar potential V_m (units are A.t) as:

$$\mathbf{H} = -grad\,(Vm) \tag{12}$$

and using $\mathbf{B} = \mu\mathbf{H}$, we obtain the equation:

$$div(\mu gradVm) = 0$$

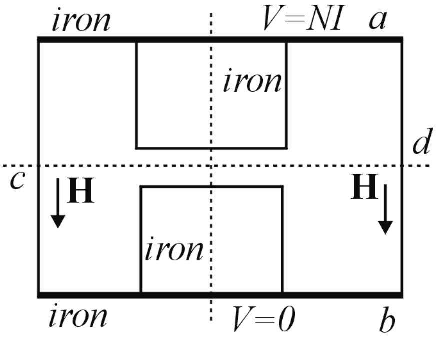
which can be written as Laplace's equation:

$$\frac{\partial}{\partial x}\mu\frac{\partial Vm}{\partial x} + \frac{\partial}{\partial y}\mu\frac{\partial Vm}{\partial y} = 0 \tag{13}$$

This is now solved using the finite element method.

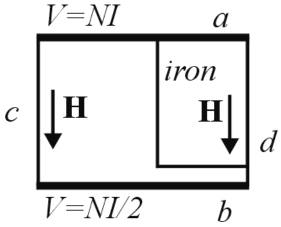
This case is again **very similar** to the first example, described in section A2. The only difference is that now we are solving for a **magnetostatic case**. This being a scalar potential problem, **all the remarks** made in section 2 regarding boundary conditions are applicable here as well.

As a practical example, suppose that the domain inside the airgap is as shown in the figure **b** below.



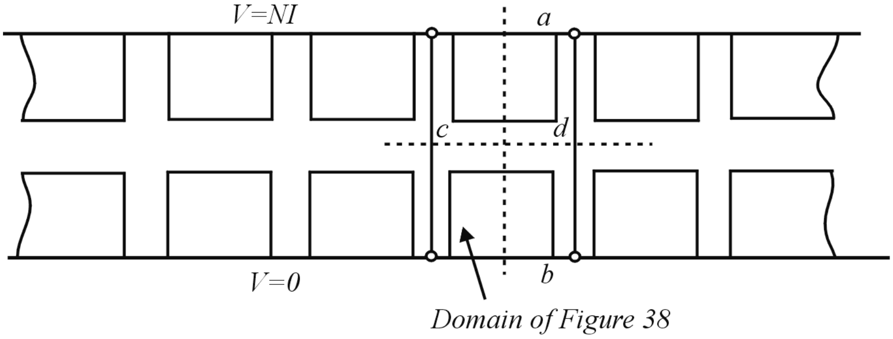
b – *domain for using magnetic scalar potential*

In this case, we may consider as a good approximation, that the fields on lines ***c*** and ***d*** are placed as shown in the figure **b** above. Because of **symmetries** (see dotted lines in this figure **b** above), the domain can be reduced to the case of the figure **c** below.



c – *reduced domain since there are symmetries in the whole geometry*

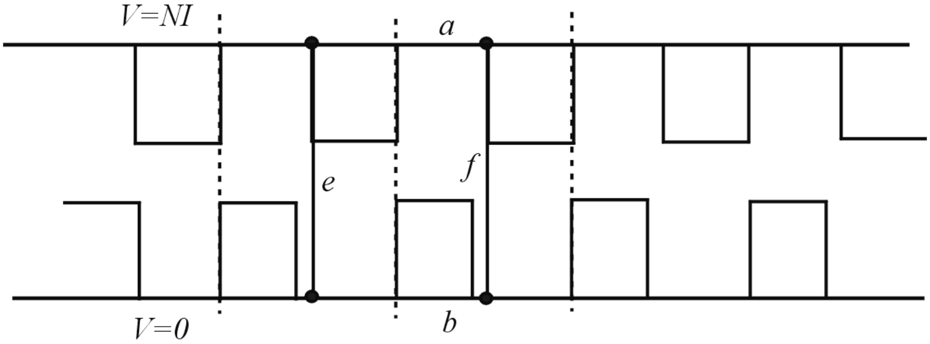
The study under consideration (figure **b**) **could also be a single section** of the case shown in the figure **d** below in which a repetition of the figure **b** is used.



d - *case under study can be considered as a replication of a single domain*

Because of the configuration of the iron teeth in the figure **d**, the Dirichlet (lines ***a*** and ***b***) and Neumann (lines ***c*** and ***d***) boundary conditions can be taken as shown in figures **b** and **c**.

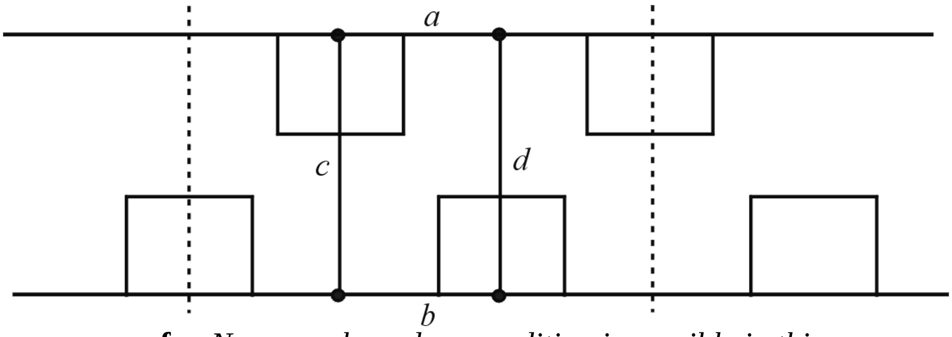
We take advantage of this example to introduce the **concept of periodicity**. Suppose that the lower iron teeth in the figure **d** are displaced as shown in the figure **e**.



e – *domain with periodicity .*

In this case it is practically **impossible to define two lines for Neumann** conditions on which the field would be parallel to the lines, because we do not know a-priori the shape of the field. However, we can state that the lines ***e*** and ***f*** are "periodical" lines since what happens to one will also happen to the other.

Therefore, defining the Dirichlet boundary condition on line ***a*** and ***b*** and the periodicity condition on lines ***e*** and ***f*,** it is possible to solve only the single domain of the figure **e**.



f. – *Neumann boundary condition is possible in this case*

Now we discuss the situation in the figure **f**. Because of the positioning of the teeth, it is possible to **define again a Neumann** condition on lines ***c*** and ***d***, since the fields will be parallel to these lines. There is no restriction in proceeding with the calculation for this case using the periodicity condition, however, using the Neumann condition on lines ***c*** and ***d*** (in the figure **f**) reduces the domain to *50%* of that using periodicity condition.

Finally, we remark that the formulation using magnetic **scalar potential is very useful for local studies** to cases as shown in this section that could be the "tooth shape" for a variable reluctance machine, for instance. Furthermore, this formulation **is very efficient** for nonlinear configurations.

A5. Magnetic Fields - Vector Potential.

Suppose that one wishes to calculate the whole structure shown in the previous figure, **including the exciting coil**. In this case, a different formulation has to be used since the one based on the **scalar magnetic potential does not allow** the inclusion of current sources in the domain.

We define here the vector magnetic potential **A** as:

$$\mathbf{B} = rot\,\mathbf{A} \tag{14}$$

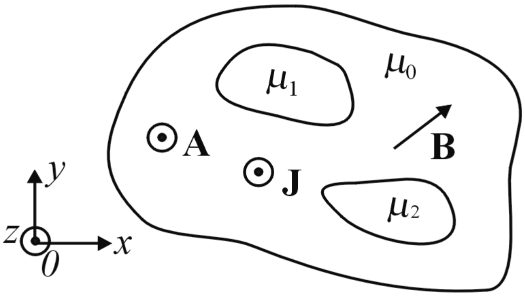
Using $\mathbf{B}=\mu\mathbf{H}$ and the Maxwell equation:

$$rot\,\mathbf{H} = \mathbf{J}$$

We obtain:

$$rot\frac{1}{\mu}\mathbf{B} = rot\frac{1}{\mu}rot\,\mathbf{A} = \mathbf{J} \tag{15}$$

In *2D* cases, the vector potential **A** and the current density **J** have only components perpendicular to the *2D* domain, as shown in the figure below. The induction **B** and the field **H** have components only in the directions *Ox* and *Oy*.



2D domain for applying the magnetic vector potential

Therefore, in order to solve Eq. (15) we must calculate $\mathbf{B} = rot\,\mathbf{A}$, which becomes:

$$\mathbf{B} = \hat{\mathbf{i}}\,B_x + \hat{\mathbf{j}}\,B_y = \hat{\mathbf{i}}\frac{\partial A}{\partial y} - \hat{\mathbf{j}}\frac{\partial A}{\partial x} \tag{16}$$

Noting that $\mathbf{A} = \hat{\mathbf{k}}\,A$, and $\hat{\mathbf{i}}, \hat{\mathbf{j}}, \hat{\mathbf{k}}$, are the unit vectors in the *Ox*, *Oy* and *Oz* directions respectively. The evaluation of:

$$rot\frac{1}{\mu}\mathbf{B} = rot\frac{1}{\mu}rot\,\mathbf{A} = \mathbf{J}$$

can be easily done as:

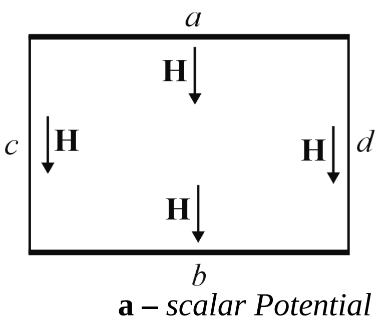
$$\det\begin{bmatrix} \hat{\mathbf{i}} & \hat{\mathbf{j}} & \hat{\mathbf{k}} \\ \dfrac{\partial}{\partial x} & \dfrac{\partial}{\partial y} & \dfrac{\partial}{\partial z} \\ \dfrac{v\partial A}{\partial y} & -\dfrac{v\partial A}{\partial x} & 0 \end{bmatrix} = \hat{\mathbf{k}}\,J$$

where v is the magnetic reluctivity $(v = 1/\mu)$, which gives:

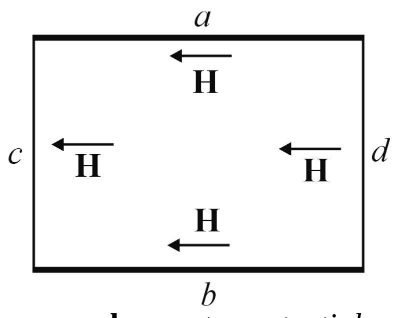
$$\frac{\partial}{\partial x}v\frac{\partial A}{\partial x} + \frac{\partial}{\partial y}v\frac{\partial A}{\partial y} = -J \tag{17}$$

which is the Poisson's equation related to this case.

It is **extremely important to understand** the boundary conditions for this formulation because **they are different compared to the scalar potential** case. Here, where the vector potential is imposed (**Dirichlet** boundary condition) the **field will be parallel to the boundary** (while with the scalar potential, the field is perpendicular to a Dirichlet boundary). On **Neumann boundaries**, where no condition is specified, the field is **perpendicular to the boundary** (while in the scalar potential case, the field is parallel). These situations are shown graphically in the figures below where lines ***a***, ***b*** are Dirichlet boundaries and ***c, d*** are Neumann boundaries for both cases.



a – *scalar Potential*



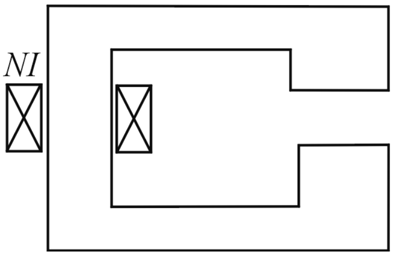
b – *vector potential.*

There are two additional facts of importance:

a) With the *2D* vector potential formulation discussed here, the magnetic **vector potential A** has a physical significance. **It is the magnetic flux** divided by the depth of the structure. Therefore, when obtaining the magnetic vector potential using the finite element method, we, in fact, obtain the flux/depth established in the solution domain by the sources. More precisely, the **difference between two values of** *A* in two parts of the domain **gives the magnetic flux integrated over the line** segment defined by the two points. Multiplying this difference by the depth of the structure gives the total flux in [*Wb*] through the corresponding surface.

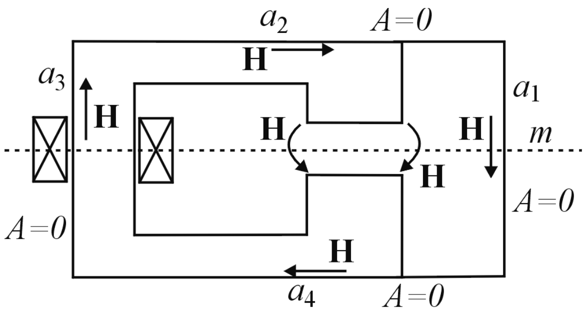
b) A vector equipotential line **is directly a line of magnetic field**.

Now we consider the structure of the next figure.

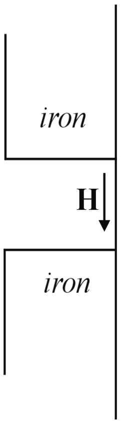


Structure to be analyzed by the vector potential

To **define** the domain, one has to take a **line** that is **not crossed by the flux**, or, a line where the **field is parallel**. On such lines, the vector **potential A=*0* is applied**. In this structure, this line is shown in the figure below.



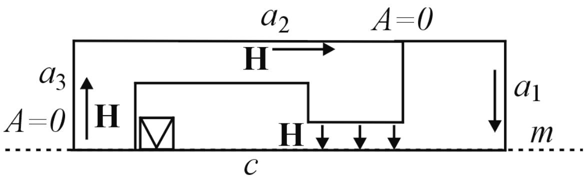
a – *lines where the flux is "retained"*



b – *wrong placement for the Dirichlet boundary condition*

Because of the high permeability of iron, the flux will be **"contained"** within the area **between lines *a_2*, *a_3*, and *a4***. However, the **line *a1*** had to be defined at a little distance from the airgap as shown in the figure **a** above since there is some **field spread** in this area (edge effect). If, on the other hand, we were to place line ***a1*** as shown in the figure **b**, the **solution would be wrong** since this defines an **incoherent physical configuration**. Thus, we impose the Dirichlet boundary conditions on the four lines ***a_1*, *a_2*, *a_3*,** and ***a_4*** by setting the magnetic vector potential to zero (**A**=0).

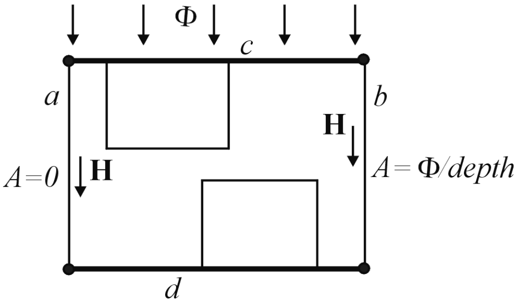
Suppose now that the **symmetry line exists** in the figure **a** (line ***m***) is considered. The magnetic field **is perpendicular** to this line. Therefore, a **Neumann boundary condition** can be applied on this line as shown in the figure **c** below.



c – *using the symmetry and applying the Neumann boundary condition*

On the boundary ***c*** we **do not impose any value** (Neumann condition) and the field will be **perpendicular to this line**, matching the physical situation. The other lines (***a_1*, *a_2***, and ***a_3***) **still have the same Dirichlet** boundary condition **A**=*0*. In this way, the domain is reduced to half its size, saving memory and computation time.

In the example above the fields are generated by the coil. However, there is a situation in which **the fields can be generated by a potential difference**, applied on the boundaries of the domain. Suppose that in the domain of the figure **d**, the **flux crossing** the domain has a **known value**. Consider also that the lines ***a*** and ***b*** retain this flux, meaning that the field **H is parallel** to these lines.

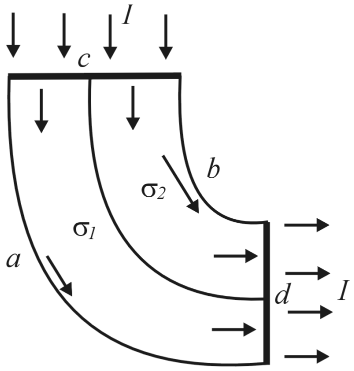


d – *vector potential application where the flux is known*

Using the concept of the magnetic vector potential, which is the **flux per depth**, it is easy to **apply the flux** in the domain. On line ***a,*** we impose the Dirichlet boundary condition *A=0.* On line ***b***, we also apply a Dirichlet boundary condition ***A=Φ/depth***. On lines ***c*** and ***d***, **no condition is applied** which makes these lines **Neumann boundary** conditions. Note the difference between this situation and the one presented previously. Here we are using a **vector potential, and the flux is known**. However, in section A4, a **scalar potential** was used and the **potential difference was known**. Depending on the problem to be solved, one of these two formulations may be more appropriate.

A6. Electric Fields - Vector Potential

The magnetic vector potential described above (section A5) has an **equivalent quantity** for the case of the electric field: **the vector electric potential T**. Suppose that in the figure below there are only conductive materials and the **current *I* passing through the domain** is **known**. We assume also that the current ***I*** is contained within lines ***a*** and ***b***.



Case to be studied by the electric vector potential

The appropriate Maxwell equation in this case is Eq. (3) with the time derivatives set to zero. Thus:

$$rot\mathbf{E} = 0$$

Defining **T** as:

$$\mathbf{J} = rot\,\mathbf{T}$$

and using:

$$\mathbf{E} = \mathbf{J}/\sigma$$

we obtain the expression:

$$rot\,\frac{1}{\sigma}\,rot\,\mathbf{T} = 0 \qquad (18)$$

As for ***A,*** we assume that *T* has only components on *z* direction and Eq. (18) becomes:

$$\frac{\partial}{\partial x}\frac{1}{\sigma}\frac{\partial T}{\partial x} + \frac{\partial}{\partial y}\frac{1}{\sigma}\frac{\partial T}{\partial y} = 0 \qquad (19)$$

which is equivalent to Eq. (17) but now it is written in terms of *T*. Note that the equivalent expressions for ***A*** and ***T*** (assuming that *J=0* for the *A* formulation) are:

$$rot\,\frac{1}{\mu}\,rot\,\mathbf{A} = \mathbf{J} \qquad rot\,\frac{1}{\sigma}\,rot\,\mathbf{T} = 0$$

$$\mathbf{B} = rot\,\mathbf{A} \qquad \mathbf{J} = rot\,\mathbf{T}$$

$$\phi = \int_S \mathbf{B}\cdot d\mathbf{s} \qquad I = \int_S \mathbf{J}\cdot d\mathbf{s}$$

$$\mathbf{A} = \frac{\Phi}{depth} \qquad \mathbf{T} = \frac{I}{depth}$$

Using the concepts described in the last example in section A5, the problem in the figure above may be solved by applying the potential ***T=0*** on line ***a*** and the potential ***T=I/depth*** on line ***b***. These lines **become Dirichlet boundary** conditions. On lines ***c*** and ***d***, on which **J** is perpendicular, are **left unspecified** (**Neumann** boundary conditions).

This formulation is **very useful** when we **know the current flowing** through the domain. A similar formulation, using the **scalar potential** was described in section A3 where the **known quantity was the potential difference.** The choice between the formulations depends, exclusively, on what is the external known condition: potential difference or current.

As in the case in section A3, ***σ* here plays the role of *μ*.** Therefore, in file **efmat.dat**, the value of ***σ*** has to be provided in the **place reserved for *μ*.** As a consequence, the user must remember that in the post processor, **EFGN,** the numerical quantities are correct but the **messages that appear on the screen** relate to the magnetic vector potential. Therefore, for the messages below, we have to affect the meanings indicated (see the corresponding equations above):

Message on the screen:	**Correct Meaning**
B (induction)	J (current density)
Flux (wb/m)	current (A/m)

Finally, the calculation of forces, inductances, and magnetic energy (available in the numerical menu of **EFGN**) are not applicable here.

A7. The Eddy Current Formulation in EFCAD

The formulations employed in the solvers **EFCJ, EFCT, EFCF, EFCM, EFCM360, EFCG, EFCMF** and **EFCV** are all based on the vector potential **A**, exactly as described in section A5. The boundary conditions also behave and are applied in the same way so there is little to remark here in this regard.

However, the equations on which these solvers are based take into account the eddy currents as follows. Using:

$$rot\mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$$

and

$$\mathbf{B} = rot\ \mathbf{A}$$

one writes:

$$rot\ \mathbf{E} = -\frac{\partial}{\partial t}\left(rot\ \mathbf{A}\right)$$

and:

$$\mathbf{E} + \frac{\partial \mathbf{A}}{\partial t} = grad\ \psi$$

Assuming that **E** is generated **only by the time rate of change of B**, and writing $\mathbf{J}_e = \sigma\mathbf{E}$, (meaning that $\mathbf{J}_e$ is the eddy current density) we obtain:

$$\mathbf{J}_e = -\sigma \frac{\partial \mathbf{A}}{\partial t}$$

The current density ***J*** in Eq. (1) takes into account all current densities, including source current densities and eddy current densities. Denoting the source current densities as $\boldsymbol{J_S}$ and eddy current densities as $\boldsymbol{J_e}$, we can write:

$$rot\, v\, rot\ A = \mathbf{J}_S + \mathbf{J}_e \qquad (20)$$

or:

$$\frac{\partial}{\partial x} v \frac{\partial A}{\partial x} + \frac{\partial}{\partial y} v \frac{\partial A}{\partial y} - \sigma \frac{\partial \mathbf{A}}{\partial t} = -J_s \qquad (21)$$

which can be written as

$$div\ \ v\, grad\ A - \sigma \frac{\partial A}{\partial t} + J_s = 0 \qquad (22)$$

where A is the component of **A** in the direction *Oz*.

A7.1. The complex formulation in EFCC

Using the complex formulation incorporated in **EFCC** we have:

$$A(t) = A^{*}e^{j\omega t} \qquad \text{for} \qquad A^{*} = A_0 e^{j\alpha} \ \text{ and } \qquad J_s(t) = J_s{}^{*}e^{j\omega t}$$

Noting that:

$$\frac{\partial A}{\partial t} = j\omega A^{*}e^{j\omega t}$$

Eq. (22) becomes:

$$div\ (v\,grad\ A^{*})e^{j\omega t} - \sigma j\omega A^{*}e^{j\omega t} + J_s{}^{*}e^{j\omega t} = 0 \qquad (23)$$

or, for linear, sinusoidal excitation we may write:

$$div\ (v\,grad\ A^{*}) - \sigma j\omega A^{*} + J_s{}^{*} = 0 \qquad (24)$$

A7.2. The Formulation in EFCJ

In this solver we consider non-linear applications and a step-by-step time in which:

$$\frac{\partial A}{\partial t} = \frac{A^{i} - A^{i-1}}{\Delta t}$$

where A^{i} represents the unknown at the time step "*i*" and A^{i-1} calculated at the previous time step "*i–1*". Δt is the time step. With this, Eq. (22) becomes:

$$div\ v\,grad\ A^{i} - \sigma\frac{A^{i}}{\Delta t} + \sigma\frac{A^{i-1}}{\Delta t} + J_s = 0 \qquad (24)$$

In this case, since the value A^{i-1} is known, this term is considered to be a known source for time step "*i*".

A7.3. The Formulation in EFCT, EFCM and EFCM360

This formulation is similar to that given above but the **source is voltage fed** rather than current fed. Using this method, *Js*, which is related to the current of the source coil(s) is also an unknown. In this case, we have to consider the electric circuit equation of the coil(s) which is, **for motor convention**:

$$V(t) = RI + L\frac{\partial I}{\partial t} + N\frac{\partial \Phi}{\partial t}$$

where *V(t)* is the voltage applied to the coil, *R* is the resistance of the coil and *L* is an additional inductance of the electric circuit (as “end effects” for example) and *N* the number of turns of the source(s). The electric circuit and the field equations are solved simultaneously and the system can be written as:

$$div\ v\,grad\ A^{i} - \sigma\frac{A^{i}}{\Delta t} + \sigma\frac{A^{i-1}}{\Delta t} + J_s(I) = 0 \qquad (25a)$$

$$RI + L\frac{\Delta I}{\Delta t} + N\frac{\Delta\Phi(A)}{\Delta t} = V(t) \qquad (25b)$$

Note that J_S **is a function of the current** and the **flux** ***Φ*** **is a function of the magnetic vector potential A.**

We point out that, using this solver, **we obtain** as solution the **potential *A*** at the nodes of the mesh (as is common in FEM) and the **current** *I(t)* established **in the coils**. This last quantity is available in the post-processor **EFGN** in its numerical menu and also in terms of files **.des** for some solvers.

A7.4. The Formulation for EFCV

When a conductive part in the domain moves at a relative velocity **v**, eddy currents are induced according to:

$$Je = \sigma \mathbf{v} B$$

The main equation now becomes:

$$div\ v\,grad\ A - \sigma \mathbf{v} B = J_S \qquad (26)$$

which is solved by this module. The velocity **v is read** in the property file **efmat.dat.**

A7.5. The Formulation for EFCF, EFCG, EFCMF and EFCMF360

With these solvers two types of conductors are considered:

a) **thin conductors** where the current is **uniformly distributed over their cross-section**. These voltage fed conductors are treated as for **EFCT** and equations (25) are also used here;

b) **thick conductors** where the current density is **not uniformly distributed over their cross-section**. The corresponding equations for these type of conductors are:

$$div\ v\,grad\ A\ \ - \sigma\frac{\partial}{\partial t}A + \sigma\frac{U}{\ell} = 0$$

$$U = R_m I_m + R_m \iint_S \sigma\frac{\partial}{\partial t}Ads$$

where U is the induced voltage on the thick conductor, I_m is the total current, and S, ℓ and R_m are, respectively, its surface, length and resistance.

With this formulation the conductors having eddy currents can be inter-connected or not. The difference between this formulation and classical 2D eddy currents one (used in **EFCJ**, **EFCT**, **EFCC**, **EFCM, EFCM360** and **EFCV)**, is that the conductors with eddy current are not intrinsically connected and short-circuited. Accordingly to the way they are relied (serial, parallel or in squirrel-cage) another electrical circuit equation is needed as, for instance:

$$C_1 U + C_2 I_m = E$$

The whole equations system for magnetic materials, thin conductors windings and thick conductors is then:

$$\begin{gathered}
div\ \ v\, grad\ A^i - \sigma \frac{A^i}{\Delta t} + \sigma \frac{A^{i-1}}{\Delta t} + J_s(I) + f_1(U_m) = 0 \\
RI + L\frac{\Delta I}{\Delta t} + N\frac{\Delta\Phi(A)}{\Delta t} = V(t) \\
C_1 U + C_2 I_m = E(t) \\
R_m I_m + R_m \sigma \frac{\Delta A}{\Delta t} - U_m = 0
\end{gathered}$$

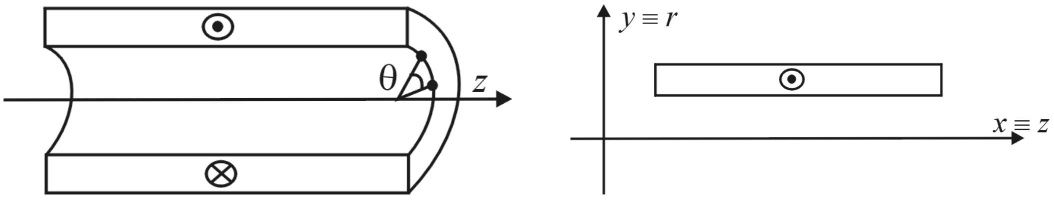
We point out that, using these solvers, **we obtain** as solution the **vector potential *A*** at the nodes of the mesh (as commonly done in FEA), the **current** *I(t)* established **in the thin conductors coils,** the **total current** $I_m(t)$ **in the thick conductors** with the corresponding **induced voltages** U_m. The calculated current and voltages are available in a **.des** formatted file.

A8. Axisymmetric Applications

As a 2D software, **EFCAD** assumes that the structure under study **does not have** any geometric **variation** in the direction **perpendicular to the cross-section** of the structure. However, it is possible to perform calculations for **three-dimensional structures** with **axial symmetry**. This means that the structure, when using **cylindrical coordinates**, does not have geometric variations in the “theta” direction.

As a simple example, consider the solenoid shown in the figure **a** in an axial cross-section. The method of entering this geometry in **EFCAD** is to place the source (current in the solenoid) as shown in the figure **b**. In the solver, the **user assumes** (by **answering** the appropriate question) that the case is axisymmetric; the solver will apply the corresponding equations.

In **EFCAD**, the **axial direction is** always the ***x*** **direction** and, obviously, the **radial direction is the *y* direction.** Using this procedure, the solver and **EFGN** will take into account the curvature of the device. If, on the other hand, the geometry in the figure **b** were not axisymmetric, it would constitute an infinitely long bar carrying a current, which is a different situation.



a – *axisymmetric case* **b –** *2D representation in* **EFCAD**

A9. Remark on File efmat.dat

The solvers in **EFCAD** and the post-processor **EFGN** take into account the fact that the formulation is a scalar or vector potential based. If for example, the problem is **magnetic**, the user enters the value of relative **permeability** in file **efmat.dat**, in the first field of the file, called **PERM**. If the solution uses a **vector potential**, the solver will proceed to calculate the **value** $1/\mu$ **as required** in Eqs. (15), (23), (24), and (25). In Eq. (13) the value of μ is used directly.

However, when the **electric field formulation** is used for conducting materials, the **equivalent quantity** in Eq. (11) (scalar potential) and Eq. (19) (vector electric potential) **are** σ and $1/\sigma$ **respectively**. To solve such cases, the **user must insert** the value of σ in the first field of file **efmat.dat**. **Note that** there is a field in this file for **conductivity** but this field is **only applicable to magnetic cases** with eddy currents.

A10. Thermal Calculations

The solvers **EFCTS** and **EFCTT** calculate static and transient thermal problems, respectively. The unknown value is the **temperature** that is considered as a **scalar potential.** The static thermal equation is:

$$div(\lambda gradT) = -q \qquad (27)$$

where:

λ is the thermal conductivity
T is the temperature (not to be confused with the vector electric potential)
q is the heating source.

Being a scalar potential, the boundary conditions **are similar** to those **presented** in section A2:
- **Dirichlet** boundary conditions for **applied temperatures**.
- **Neumann** boundary conditions if **no condition is to be specified**.

The **gradient of temperature** shows the **direction of the heat flux** and is similar to the electric field defined in section A2 as $\mathbf{E} = -gradV$.

There is also **an additional boundary condition** applicable to thermal problems, for cases in which **radiation and convection** occur.

Heat transfer to the ambient space **by convection** is expressed as:

$$q_c = HS(T - T_a)$$

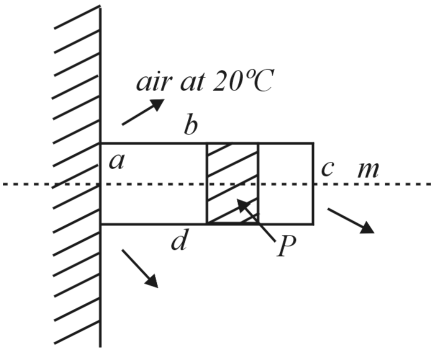
where *S* is the surface in contact with the ambient space, *H* is the coefficient of heat transfer (its units are *W/m^2.ºC*), *T* is the temperature of the body within the surface *S* and T_a is the ambient temperature.

Heat transfer **by radiation** is expressed as:

$$q_r = \varepsilon \gamma S(T^4 - T_a^4)$$

where ε is the emissivity of the body (0<ε<1), γ is the Stephan-Boltzmann constant (5.669x10^8 W/(m^2.ºK^4).
Convection and radiation transfer are included in the program **EFCTS** and **EFCTT**. The values of *H* and T_a are provided in the pre-processors **EFM** and **EFR**. The value of ε is stored in file **efterm.dat**.

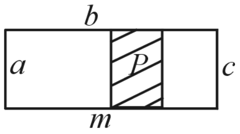
Figure **a** below shows an example of thermal calculation. A body is in contact with a **heated wall at *100ºC* (line a).** In part *P* there is a **current density** *J*, **responsible** for additional heating in the body. Except for the wall, all other surfaces (surfaces ***b, c, d***) of the body are in **contact with the air**, which is at *20ºC*. There is also **a symmetry**, defined by line ***m.***



a – *thermal case with Dirichlet, Neumann and convection/radiation boundary conditions.*

The domain for calculation is shown in the figure **b**. On boundary ***a***, we have a **Dirichlet** boundary condition with ***T=100ºC.*** On lines ***b*** and ***c***, we define a **radiation/convection** boundary condition with the ambient temperature at ***20ºC***. The **emissivity** of the body, which is **necessary for the radiation** condition, is read in file **efterm.dat**. The **coefficient** *H,* related to the **convection**, is **entered** (typical value for natural convection is ***10)*** in the **pre-processor(s)**. On boundary ***m***, **nothing is specified** and we have a **Neumann** boundary condition. This **means that the heat flux** (or the gradient of temperature) is parallel to this line because of the geometric symmetry.

The region *P*, which has an electric conductivity σ and a current density *J* is a source of heating in this example. The Joule effect produces the quantity $q=J^2/\sigma$ (*W/m^3*) shown in Eq. (27).



b – *corresponding domain entered in* **EFCAD.**

The **static calculation** will give the thermal situation in steady state, that is, the state of the body **after the temperatures in the whole domain have stabilized**.

The thermal equation taking into account the **transient state** (program **EFCTT**) is:

$$div(\lambda gradT) + q = c\frac{\partial T}{\partial t}$$

where *c* is the thermal capacity.

This equation, when discretized in time is similar to Eq. (24) and is:

$$\frac{\partial}{\partial x}\lambda\frac{\partial T^i}{\partial x} + \frac{\partial}{\partial y}\lambda\frac{\partial T^i}{\partial y} - c\frac{\partial T^i}{\Delta t} = -q^i - \frac{\partial T^{i-1}}{\Delta t}$$

where T^i is the unknown temperature at the i*th* time step and T^{i-1} is the temperature calculated at the previous time step. The heating source *q* can also be a function of time.

Applying this formulation to the example above we get the solution in terms of temperatures in the whole body at different time steps of the simulation. If it is continued long enough, the **final steps will provide** a solution **identical to the steady state** solution. The initial temperature of the body is taken by **EFCAD** as *20ºC,* which matches the major part of practical cases.

As already mentioned, **EFCTS** and **EFCTT** deal with non-linear calculations because the **thermal conductivity depends** on temperature as:

$$\lambda(T) = K + K_V e^{-(T/\tau)}$$

The values of *K*, K_V, and τ are stored in file **efterm.dat**. If one wishes to consider a **linear case**, **only** *K* has to be entered as a nonzero value in this file, which is managed by the program **EFTP**.

A11. Iron Losses Calculation

The solvers **EFCR, EFCR360, EFCM, EFCM360, EFCG, EFCMF** and **EFCMF360** calculate iron losses considering rotating and alternating induction as well as their harmonics. Losses separation into three components is considered:

- hysteresis losses p_H ;
- eddy current losses p_F ;
- anomalous losses p_E .

Considering these three components, the total iron losses density is:

$$p_{Tot} = p_H + p_F + p_E \qquad [\mathrm{W/m^3}]$$

or, writing explicitly the components:

$$p_{Tot} = \sum_{k=1}^{\infty}\left[1+R_{Bk}X\right]p_{Hk}^{a} + C_F\,\frac{1}{T}\int_0^T\left[\left(\frac{dB_r}{dt}\right)^2+\left(\frac{dB_t}{dt}\right)^2\right]dt + C_E\,\frac{1}{T}\int_0^T\left[\left(\frac{dB_r}{dt}\right)^2+\left(\frac{dB_t}{dt}\right)^2\right]^{\frac{3}{4}}dt$$

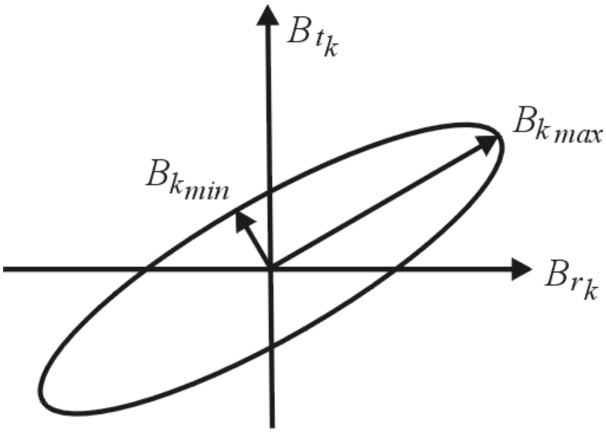
with

$$p_{Hk}^{a} = C_H B_{\max k}^{\alpha} kf$$

where

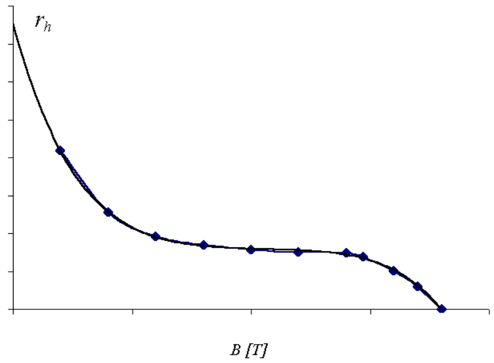
- T is the induction time period;
- B_r and B_t are the induction radial and tangential components;
- R_{Bk} and $B_{\max}$ are defined for each induction harmonic as shown in the next figure.

$$R_{Bk} = \frac{B_{k\,\min}}{B_{k\,\max}}$$



Locus B_r, B_t *for harmonic k*

- C_H, α are parameters associated to the hysteresis model;
- C_F is the eddy current losses coefficient;
- C_E is the anomalous losses coefficient;
- $X = (r_H - 1)$
- r_H is the relationship between the losses generated by purely rotating and alternating inductions with the same amplitude as shown in the following figure.



Ratio between rotating and alternating losses

In the iron losses evaluation performed by **EFCAD**, this curve is represented by means of a 6th order polynomial as follows:

$$r_H = C_1B^6 + C_2B^5 + C_3B^4 + C_4B^3 + C_5B^2 + C_6B + C_7$$

where B is the magnetic induction.

This formulation takes into account the real induction waveforms and can be used in step by step field calculations. But, for convenience, the parameters to be furnished to **EFCAD** (by means of file **eflos.dat** created by **EFPL**) are those related to a sinusoidal waveform (see section 4.3.3):

- $Ceddy = 2\pi^2 C_F$
- $Canom = 8.764C_E$
- $Chyst = C_H$
- $Alfa = \alpha$

References

N. Ida, J.P.A. Bastos,“Electromagnetics and Calculation of Fields”, Springer-Verlag, N.Y., USA, Second Edition 1997, ISBN 0-387-94877-5.

J.P.A. Bastos, N.Sadowski, “Electromagnetic Modeling by Finite Element Methods”, Marcel Dekker, N.Y., USA, 2003, ISBN 0-8247-4269-9.

Jonathan Shewchuk, Program Triangle, for 2Dmesh generation, see site: www.cs.berkeley.edu/~jrs/.

J.P.A. Bastos, R. Carlson, “An efficient automatic 2D mesh generation method”, COMPUMAG 1991, Sorrento, Italy, Proceedings, pp 83-86.

For personal contacts:

At GRUCAD – Florianópolis, Brazil (EFCAD continuing development)
Prof. João Pedro Assumpção Bastos : jpab@grucad.ufsc.br
Prof Nelson Sadowski : nelson@grucad.ufsc.br
Prof. Renato Carlson : rcarlson@ieee.org

At LEEI de Toulouse, France (expertise on device applications)
Dr. Michel Lajoie-Mazenc : lajoie@leei.enseeiht.fr

At LEEI de Toulouse, France (expertise on graphical and computer tools)
Dr. Jean Hector : hector@leei.enseeiht.fr

At University of Akron, USA (co-author of reference above)
Prof. Nathan Ida : ida@uakron.edu

At GOPAC / UFMG – Belo Horizonte, Brazil (computational and ICCG expertise)
Prof. Renato Mesquita : renato@cpdee.ufmg.br