Stanislav Bartoň
Visualisation of the electromagnetic vector fields

In: Jan Chleboun and Pavel Kůs and Jan Papež and Miroslav Rozložník and Karel Segeth and Jakub Šístek (eds.): Programs and Algorithms of Numerical Mathematics, Proceedings of Seminar. Jablonec nad Nisou, June 19-24, 2022. Institute of Mathematics CAS, Prague, 2023. pp. 7–14.

Persistent URL: http://dml.cz/dmlcz/703183

Programs and Algorithms of Numerical Mathematics 21
J. Chleboun, P. Kůs, J. Papež, M. Rozložník, K. Segeth, J. Šístek (Eds.)
Institute of Mathematics CAS, Prague 2023

VISUALISATION OF THE ELECTROMAGNETIC VECTOR FIELDS

Stanislav Bartoň

Opole University of Technology
Faculty of Electrical Engineering, Automatic Control and Informatics
Prószkowska 76 Street, 45-758 Opole, Poland
s.barton@po.edu.pl

Abstract: Modern computer algebra software can be used to visualize vector fields. One of the most used is the Maple program. This program is used to visualize two and three-dimensional vector fields. The possibilities of plotting direction vectors, lines of force, equipotential curves and the method of colouring the surface area for two-dimensional cases are shown step by step. For three-dimensional arrays, these methods are applied to various slices of three-dimensional space, such as a plane or a cylindrical surface. Finally, the temporal evolution of the vector fields is illustrated by animations based on the above methods. In contrast to the publication [2], which deals only with the problem of colouring vector fields, the present paper makes a completely comprehensive study of the problem, including the representation of vectors in a predefined network, the computation of the shape of power lines, and the animation of time changes, including the animation of the coloured vector fields.

Keywords: vector field, visualisation, coloring, line of force, animation, Maple

MSC: 65D18, 65L06, 26B15, 34K28, 78A30

1. Introduction

Visualising vector fields is a handy tool that allows an easy and obvious presentation of their basic properties. All the visualisations presented in this article are made using Maple. Because it is a very complex problem it is solved step by step. In some parts, the algorithms presented could be shortened, but at the cost of lower clarity.

The calculation of the magnetic induction vector is quite complex, and the analytical relationship can only be derived for the elementary shapes of the conductor - a line segment or a circular arc. For more complex conductor shapes, a numerical solution is required because a complicated integral must be calculated along the curve that corresponds to the conductor shape.

DOI: 10.21136/panm.2022.01

2. Magnetic field inside the stator of a three-phase electric motor

The magnetic induction vector inside the motor is given by the vector sum of the magnetic inductions that form the individual phase coils of the stator. Suppose we observe the magnetic induction in the longitudinal plane of symmetry of the stator. In that case, it is possible to neglect the influence of the parts of the coils that are wound parallel to the stator bases. The resulting magnetic field is then given by the sum of the magnetic fields formed by the six straight conductors evenly spaced around the stator circumference.

To simplify the calculation, assume that the length of the stator is much larger than its radius, which we choose as the unit of length. We choose a coordinate system with the origin at the centre of the stator axis, the x axis oriented horizontally, the y axis oriented vertically and the z axis identical to the stator axis.

The magnetic induction vector $\vec{B}$ of a linear conductor is calculated according to the Biot-Savart law

$$\vec{B} = \frac{\mu\, J}{4\pi} \int_C d\vec{w} \times \frac{\vec{p} - \vec{r}_w}{|\vec{p} - \vec{r}_w|^3}$$

where μ =permeability, J =current flowing through the conductor, $d\vec{w}$ =current element of the conductor, $\vec{p}$ =position vector of the observation point and $\vec{r}_w$ =position vector of the current element of the conductor $d\vec{w}$.

For a very long wires, $L \to \infty$, in X_i, Y_i coordinates, parallel to the z axis, the observation point $\vec{p} = [x, y, 0]$ and choosing the currents $\frac{2\,\mu\,|J_i|}{4\pi} = 1$, $\vec{B}_i$ can be expressed as:

$$\vec{B}_i(p) = \left[\frac{J_i\,(Y_i - y)}{(X_i - x)^2 + (Y_i - y)^2}, \frac{J_i\,(x - X_i)}{(X_i - x)^2 + (Y_i - y)^2}, 0 \right].$$

The simplifying condition for the absolute magnitude of the current combined with the choice of the length unit does not affect the shape of the resulting plots because they are only multiplicative constants.

The currents flowing through the single-phase coils of a three-phase electric motor are offset from each other by one-third of a phase. Each coil has two parts parallel to the stator axis. The first part sends a current in the positive direction of the z-axis and the second part flows back from the negative direction of the z-axis. Thus, six alternating currents with a gradual phase shift of one-sixth of the period flow through the stator,

$$J_i = \sin\left(\omega\, t + \frac{i\, 2\pi}{3}\right), i = 1..6.$$

If we choose one period of the alternating current as the time unit, i.e. $\omega = 2\pi$, this multiplicative constant will not appear in the derived expressions. This option does not affect the visualization or animation of the magnetic field.

The conductors forming the coils are located on the coordinates:

$$X = \left[r, \frac{r}{2}, -\frac{r}{2}, -r, -\frac{r}{2}, \frac{r}{2} \right], \quad Y = \left[0, \frac{r\sqrt{3}}{2}, \frac{r\sqrt{3}}{2}, 0, -\frac{r\sqrt{3}}{2}, -\frac{r\sqrt{3}}{2} \right].$$

If we restrict the study of the magnetic induction vector to the plane perpendicular to the stator symmetry axis, $z \equiv 0$, the resulting magnetic induction vector is

$$\vec{B}_f(x, y, t) = \sum_{i=1}^{6} \vec{B}_i.$$

The function that describes the magnetic induction vector depending on the coordinates x, y and time t we derive in Maple as follows:

```
> restart; with(plots): with(LinearAlgebra):
> b:=[J[i]*(Y[i]-y)/((X[i]-x)^2+(Y[i]-y)^2),
      J[i]*(x-X[i])/((X[i]-x)^2+(Y[i]-y)^2)]:
> X:=[seq(r2*cos(i*Pi/3),i=0..5)]: Y:=[seq(r2*sin(i*Pi/3),i=0..5)]:
> J:=[-seq(sin(2*Pi*t+i*Pi/3),i=0..5)]: Bf:=unapply(sum(b,i=1..6),x,y,t):
```

3. Mathematical foundations of vector field visualization

An ordinary differential equation `deq` gives the magnetic induction field line. This equation has no analytical solution, so it must be solved numerically. For accurate drawing, it is necessary to control `d`, the mutual distances of the points on the power line. We use Newton's iterative procedure to calculate the parameter step `p`, which ensures the desired distance between the last `x1, y1` and new end point `x, y` of the power line. The function `DP` gives the correction to the existing parameter value. `xp, yp` are the derivatives of x and y according to the parameter p, returned by the `NS` numerical procedure solving the `deq` differential equation. The intermediate numeric values of the parameter p are contained in the variable `q`.

```
> deq:={diff(x(p),p)=Bf(x(p),y(p),t)[1],diff(y(p),p)=Bf(x(p),y(p),t)[2],
        x(0)=x0,y(0)=y0}:
> DP:=(x1,y1,x,y,xp,yp,d)->-1/2*(x1^2-2*x1*x+x^2+y1^2-2*y1*y+y^2-d^2)
         /(-xp*x1+xp*x-yp*y1+yp*y):
```

3.1. Initial values of parameters for drawing power lines

The arrows representing the magnetic induction vectors are drawn for the points stored in the `AP` list. As an initial point for drawing the power lines, we choose $2\,$`n` $+ 1$ point `IP0` $\rightarrow$ `IP` lying on a line passing through the centre of the stator and perpendicular to the magnetic induction vector in the stator axis. We stop drawing the power line when its endpoint reaches the stator circumference, `R1` ≥ 1. The distances of the points on the power line are chosen as `d` $= 0.05$. We plot the power lines for 90-time instants of one revolution of the motor, `T`. The stator coils are wound around the stator axis on the radius `r2` $= 1.25$. In reality, the coils are wound on a smaller radius, but this choice will reduce the differences in the length of the arrows plotting the magnetic induction vectors near the inner circumference and the centre of the stator. This will make the magnetic field more homogeneous in the stator.

```
> AP:=[seq(seq([i/12,j/12],i=-12..12),j=-12..12)]:
> AP:=map(u->`if`(add(w^2,w=u)>1,NULL,u),AP): r2:=1.25: d:=0.05: n:=16:
> IP0:=[seq(i/n,i=-n..n)]: T:=[seq(i/90.,i=0..89)]:
```

3.2. Auxiliary variables used in the calculation

The meaning of the other variables used is as follows: `nt` = current value of the time loop step, `Bf00` = vector of magnetic induction in the stator axis, `alpha` perpendicular direction to `Bf00`, `t` = current value of time, `TXY` = list containing the completed power lines for the given time step, `ni` = current value of the initial value loop step, `x0, y0` = initial values for the numerical solution `NS` of the differential equation `deq`, `XY` = list of calculated points of the current force line, `Q` = list of numerical values of the formal parameter p for the points stored in `XY`, `dq` = correction of `q` for the relative distance of the power line points or for the location of the power line end points on the stator circumference, `X, Y` = current position of the power line point, `W` = magnetic induction vectors at `AP` grid points, `GLoF[nt], GLoA[nt]` = graphs for the time `t` showing the internal stator circuit and the power lines, and a graph showing the magnetic induction vectors.

3.3. Variables needed for colouring of vector fields

The colouring of the vector field is done in the grid that is stored in the `TP` variable as a list of rectangles. First, the magnetic induction vectors `BF` are calculated for the rectangle centre points `CP`. The absolute values `AB` and the arguments `Phi` are calculated for these vectors. For each time step, the maximum and minimum value of `AB` are stored in the list `MAB` and `mab`. The following procedure does the actual colouring of the vector field:

```
> read "TP.sav": CP:=map(u->add(w,w=u)/nops(u),TP):  MAB:=[]: mab:=[]:
```

3.4. Main computational loop

```
> for nt from 1 to nops(T) do;                                    # THE TIME LOOP
    t:=T[nt]: TXY:=[]: Bf00:=evalf(Bf(0.,0.,t)):
    alpha:=evalf(argument(Bf00[2]-Bf00[1]*I)):
    IP:=map(u->evalf([u*cos(alpha),u*sin(alpha)]),IP0):
    for ni from 1 to nops(IP) do:              # THE FORWARD, q=-0,02, NODE LOOP
      XY:=[IP[ni]]: x0,y0:=XY[-1][]: NS:=dsolve(deq,{x(p),y(p)},numeric);
      Q:=[0.]; q:=-0.02: R1:=0.:
      while R1<1.0 do:                          # THE NEW POWER LINE POINT LOOP
        dq:=1:
        while abs(dq)>1e-6 do;                             # THE d DISTANCE LOOP
          Xp,Yp:=evalf(Bf(XY[-1][],t))[]: X,Y:=map(u->rhs(u),NS(q)[2..3])[]:
          dq:=DP(XY[-1][],X,Y,Xp,Yp,d): dq:=dq/sqrt(4+dq^2): q:=q+dq:
        end do:
        XY:=[XY[],[X,Y]]; Q:=[Q[],q]: q:=2*Q[-1]-Q[-2]: R1:=sqrt(X^2+Y^2):
      end do:
      q:=Q[-1]: dq:=1:
```

```
      while abs(dq)>1e-6 do;                    # THE END POINT - PERIMETER LOOP
        X,Y:=map(u->rhs(u),NS(q))[2..3][]: Xp,Yp:=evalf(Bf(X,Y,t))[]:
        dq:=DP(0,0,X,Y,Xp,Yp,1): dq:=dq/sqrt(4+dq^2): q:=q+dq:
      end do:
      XY:=[XY[1..-2][],[X,Y]]: TXY:=[TXY[],XY]:
    end do:
    # for ni from 1 to nops(IP) do: ... end do: # THE BACKWARD, q=0.02, NODE LOOP
    # This loop repeats the calculation of the forward loop.
    # The only difference is the choice of the initial value of the parameter q.
    # Therefore, its listing is omitted
    GLoF[nt]:=display({plot([sin(f),cos(f),f=0..2*Pi],color=black,thickness=2),
      plot(TXY,color=red)}): W:=evalf(map(u->Bf(u[],t),AP)):
    GLoA[nt]:=arrow(zip((u,v)->[u,v/25],AP,W),shape=harpoon):
    GLIP[nt]:=plot(IP,style=point,symbol=circle,symbolsize=15,color=blue):
    BF:=map(u->evalf(Bf(u[],t)),CP): AB[nt]:=map(u->abs(u[1]+I*u[2]),BF):
    MAB:=[MAB[],max(AB[nt])]; mab:=[mab[],min(AB[nt])]:
    Phi[nt]:=map(u->argument(u[1]+I*u[2]),BF):
end do:
```

The resulting magnetic induction vector field is shown in Figure 1.

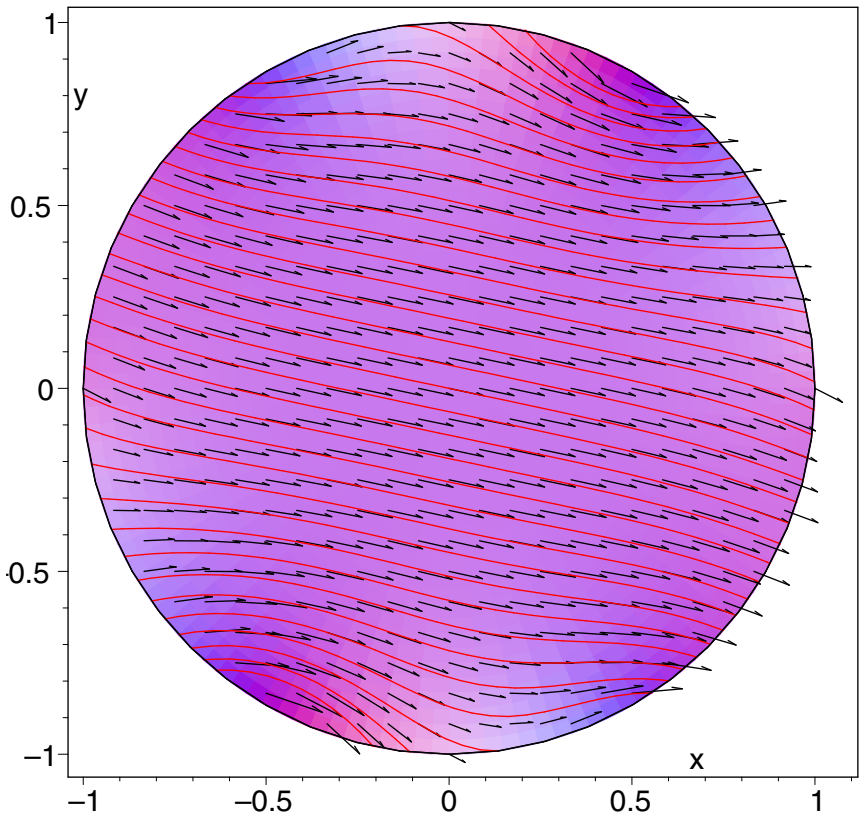

Figure 1: Electromagnetic induction inside stator, `nt` = 49

3.5. Colouring of the vector fields

The colouring of the vector field is done in the grid that is stored in the TP variable as a list of rectangles. First, the magnetic induction vectors BF are calculated for the rectangle centre points. The absolute values AB and the arguments Phi are calculated for these vectors.

To colour the vector field, three functions are needed to mix the fill color of the individual quadrilaterals from TP by selecting the shade of the red, green and blue components. The intensity of each colour component is controlled by the functions, R1, R2 and R3 which depend on the absolute value of the vector AB. The colour ratio depends on the direction of the vector, Phi.

```
> R1:=(av,phi)->evalf(1-(sin(phi)+1)*av/2):
> R2:=(av,phi)->evalf(1-(sin(phi+2*Pi/3)+1)*av/2):
> R3:=(av,phi)->evalf(1-(sin(phi+4*Pi/3)+1)*av/2):
```

Because Min is not equal to zero, it is convenient to choose 0.25 as the lowest intensity value, when the difference in hue is already noticeable. The upper limit of the intensity is 1.

To achieve the identical colouring of the vector fields for all time instants it is necessary to perform a linear transformation LT of the absolute values of the magnetic induction vector, AB. Therefore, the maximum and minimum values of the list MAB are saved in the variables Max and Min. This leads to a higher colour contrast in the resulting image.

```
> Max:=max(MAB): Min:=min(mab): Yh:=1: Yd:=0.25: k:=(Yh-Yd)/(Max-Min):
> q:=Yh-a*Max: LT:=unapply(k*u+q,u):
```

Colouring the vector field in Figure 1 for nt = 49 was done by the following procedure

```
> PAB:=zip((u,v)->[LT(u),v],AB[49],Phi[49]):
> display(zip((u,v)->polygonplot(u,color=COLOR(RGB,R1(v[]),R2(v[]),R3(v[]))),
    TP,PAB),style=patchnogrid,axes=boxed,scaling=constrained);
```

A reference vector field, see Figure 2, is required to properly evaluate the coloured vector field. This figure shows how the colour depends on the absolute magnitude of the vector and its direction. This means that from the system of polygons TP, it is necessary to remove those whose central point CP has a distance from the origin of the coordinate system less than 0.25, TP → TPr, CP → CPr. The remaining polygons are then coloured depending on the absolute size and the position vector argument of the centre point CPr.

```
> TPr:=zip((u,v)->`if`(u[1]^2+u[2]^2>Yd^2,v,NULL),CP,TP):
> CPr:=map(u->`if`(u[1]^2+u[2]^2>0.0625,u,NULL),CP):
> pab:=subs(Float(undefined)=0.,map(u->[abs(u[1]+I*u[2]),
      argument(u[1]+I*u[2])],CPr)):
> CW:=display(zip((u,v)>polygonplot(u,color=COLOR(RGB,R1(v[]),R2(v[]),R3(v[]))),
      TPr,pab),style=patchnogrid,axes=boxed,scaling=constrained):
```

```
> CWL:=plot([[Yd*cos(f),Yd*sin(f),f=0..2*Pi],[Yh*cos(f),Yh*sin(f),f=0..2*Pi]],
      color=[blue,red],thickness=2):
> CWTmin:=textplot([Yd*cos(Pi/4),Yd*sin(Pi/4),convert(round(Min*1000)*.001,
      string)],color=blue,align={LEFT,BELOW},font=[COURIER,BOLD,16]):
> CWTmax:=textplot([Yh*cos(Pi/4),Yh*sin(Pi/4),convert(round(Max*1000)*.001,
      string)],color=red,align={RIGHT,ABOVE},font=[COURIER,BOLD,16]):
> display({CW,CWL,CWTmin,CWTmax}):
```

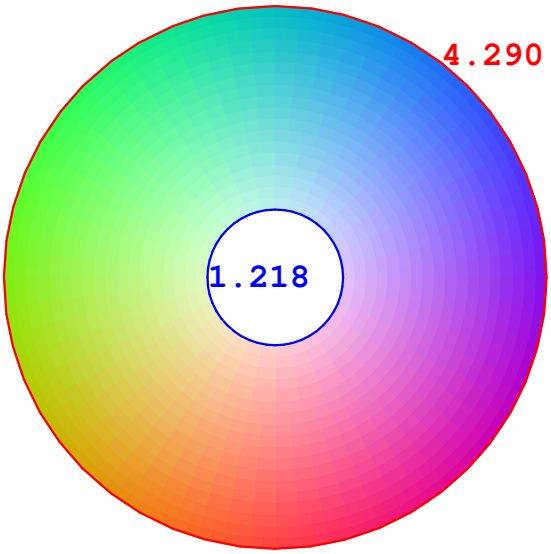

Figure 2: Reference vector field

3.6. Animation

The animation is created by displaying the images sequentially. It is possible to display several images simultaneously at each step of the animation. In the case of the magnetic induction vector visualisation, this will be three sub-images. The arrows indicating the magnitude and direction of the magnetic induction vector AA, the magnetic field lines AL, and the coloured vector field ACF. However, to fit the vector field, it is necessary to perform a linear transformation of the absolute values of the magnetic induction vector AB from the interval $\langle \mathtt{Min}, \mathtt{Max} \rangle$ to $\langle 0.25, 1 \rangle$.

```
> for nt from 1 to nops(T) do:
    PAB:=zip((u,v)->[LT(u),v],AB[nt],Phi[nt]):
    CF[nt]:=display(zip((u,v)->polygonplot(u,color=COLOR(RGB,R1(v[]),
           R2(v[]),R3(v[]))),TP,PAB),style=patchnogrid);
  end do:
> ACF:=display([seq(CF[i],i=1..nops(T))],insequence=true):
> AL:=display([seq(GLoF[i],i=1..90)],insequence=true,axes=boxed,
      scaling=constrained):
```

```
> AA:=display([seq(GLoA[i],i=1..90)],insequence=true,axes=boxed,
      scaling=constrained):
> TACF:=display({AA,AL,ACF}): TACF;
```

Creating the final animation and preparing its display is extremely computationally intensive and can take imately twenty minutes. It takes the same amount of time to free the operating memory after the animation is finished directly in the Maple environment. Therefore, it is preferable to save the final animation as an animated file in `gif` format. It takes less time to create than a Maple animation and can then be displayed at any time independently of Maple. However, it cannot be controlled as Maple allows.

```
> interface(plotdevice=gif,plotoutput="TACF.gif",plotoptions=
    `width=1000,height=1000`):
> TACF;
```

4. Conclusion

The procedures presented here for visualising vector fields offer superior and easy to understand results. However, they must be used judiciously. Firstly, the goal of the visualisation must be defined, and the whole procedure must be subordinated to this. This means deciding whether the visualisation is 2D or 3D, whether to use only arrows to indicate the size and direction of the vectors or whether to use force line drawing, as well as the decision to use area colouring or animation.

In the case of rendering 3D graphs, it must be remembered that graphs containing more complex vector fields are difficult to see. Therefore, it is preferable to display these fields as two-dimensional arrays using several planar slices through the viewed space.

A fundamental issue for drawing power lines is the appropriate choice of their starting points. Force lines may be concentrated in one part of the graph if the initial points are chosen inappropriately, and the resulting display may give a misleading impression. The second issue is the choice of the correct step length – the distance between the points forming the power line. Too small a step length increases the demands on computational capacity. Conversely, too large a step can lead to general detail distortion and loss of scientific accuracy. Therefore, it is always up to the user to tune the procedures published here for individual purposes.

References

[1] Gander, W., Gander, M. J., Kwok, F.: *Scientific Computing. An Introduction using Maple and Matlab.* Springer, 2014.

[2] Bartoň, S.: Color visualisation of the Vector Field. In: M. Ross (Ed.) *5th International Congress on Industrial Applied Mathematics.* UoT Sydney, 2003.

[3] Walker, J.: *Fundamentals of Physics.* John Willey, 2014.

[4] Urone, P., Hinrichs, R.: *College Physics.* OpenStax, 2017.