1. INTRODUCTION

The technique of conformal mapping, i.e. the use of the mathematical properties of analytical complex functions of a complex variable was traditionally a basic staple of any engineering course or textbook in electromagnetic field theory (see for instance [1]), which have always included sections or even chapters on this subject. Indeed, applications in Electrostatics seem to be almost as old as the mathematical technique itself, with important problems like the edge effect in parallel plate conductors documented as early as 1923 [2]. The introduction of microwave printed transmission lines in the fifties fostered a revival of conformal mapping which was used in the first analysis of microstrip lines [3]. In the sixties the classical papers by Wheeler [4], Schneider [5] and epigones provided the standard formulas for microstrip lines, still widely used today. A good description of these procedures, involving the Schwarz-Christoffel transformation and elliptic functions can be found in the reference textbook by Collin [6], where conformal mapping is applied to the simpler case of the calculation of characteristic impedances of striplines. Later on, conformal mapping was combined by finite elements [7], and important problems in antenna theory, like the annular slot antenna fed by a coaxial line, were successfully tackled [8].

Today, conformal mapping remains an essential topic in applied mathematics. Full chapters are dedicated to it in most classical and current complex variable textbooks (see for instance [9]), among whose the trilogy of Henrici [10] stands as particularly useful for engineering applications. In addition, classical monographs like those by Bewley [11], Kober [12] and Nehari [13], fully dedicated to conformal mapping, remain an untapped source of scarcely explored mappings that could lead to unexpected applications.

However, in the last twenty years conformal mapping has disappeared from electromagnetic textbooks at the same rate that complex calculus has shrunk in electrical engineering curricula and that homemade or commercial versions of numerically intensive software tools have been introduced in university laboratories. Nowadays this trend could be rapidly reversed because, after a long exile in the real number realm, the most popular programming languages in engineering are back to the Argand plane, where they manipulate complex variables and implement matrix complex algebra operations with unprecedented easiness.

Hence, conformal mapping could become again a popular tool in electrical engineering, resurfacing as an useful pre-processing tool prior to the use of "brute force" numerical techniques, like finite differences or finite elements. Some signs of this software-driven resurgence can be found in recent publications like [14].

A final word should be said about the publication in 1997 of the excellent book by Tristam Needham "Visual Complex Analysis" [15], hailed as a new paradigm in the way of approaching the complex analysis. Indeed, this book changed forever the authors' perception...
of complex variables and was relevant to the genesis of the modest applications described in this paper. The reading of this exciting book, which also provides a very interesting bibliography, is unconditionally recommended to anyone interested in looking afresh to the complex variable realm.

2. ON THE CHOICE OF A PROGRAMMING LANGUAGE

This paper is strongly rooted in Matlab and all its ideas and predicates have been directly checked with it. While there are nowadays several programming languages able to accomplish the same tasks, Matlab is unchallenged in its versatility, since it can be easily used at all levels. Depending on the user's proficiency and personal slants, Matlab can offer many different services, ranging from a pocket supercomputer, to an object-oriented programming language and passing through powerful numerical analysis routines, specific engineering tools (for instance the FemLab finite element toolbox), virtual laboratories, comprehensive drawing facilities and friendly graphical user interfaces. Moreover, Matlab seems to be gaining a dominant position in engineering, as once did Fortran. Matlab manipulates naturally complex variables and matrices and its script lines look almost like standard mathematical formulas. Consequently, it is a natural language to explore conformal mapping, since it produces compact and transparent codes that can be easily transcribed in other programming languages. And if its matrix features are fully exploited, Matlab is not as slow as the common lore pretends.

In this paper, we would like to contribute to the reappraisal of conformal mapping by combining it with Matlab and showing graphically one less-known application of the theory, the calculation of 2D static Green's functions. The impressive graphical facilities of Matlab have been sparingly used, in order not to hide behind sophisticated graphical programming the basic mathematical lines in the codes. By doing that, the author hopes to reduce the risks of this paper becoming quickly obsolete, though it is true that nothing evolves faster than programming languages.

3. BASIC THEORY

Traditional wisdom suggests that a conformal mapping \( w = w(z) \) between two complex variables \( z = x + jy \) and \( w = u + jv \) may provide a successful strategy for the two-dimensional Laplace equation if it simplifies the shape and the boundaries of the domain where the equation is to be solved. As it is well known \[9\], the complex function \( w = w(z) \) is equivalent to a set of two real functions of two real variables:

\[
\begin{align*}
w = w(z) & \iff \{u = u(x, y) \quad ; \quad v = v(x, y)\} \\
\text{and if } w(z) \text{ is analytical, then the real functions } u, v \text{ fulfill the Cauchy-Riemann equations and are harmonic, i.e. they satisfy the Laplace equations } & \nabla^2_{xy} u = \nabla^2_{xy} v = 0, \text{ where the 2D Laplacian operator in cartesian coordinates is } \nabla^2_{xy} = \partial^2 / \partial x^2 + \partial^2 / \partial y^2. \\
\text{Now, let us consider a function } & F(u, v) \text{, where } u, v \text{ are the real and imaginary part of an analytical function } w(z). \text{ It is clear that } F \text{ is also a function of the coordinates } x, y \text{ and we can}
\end{align*}
\]

introduce formally the convenient notation: $F(u, v) = F[u(x, y), v(x, y)] = f(x, y)$. Now using the above mentioned properties of the harmonic functions $u$ and $v$, it can be shown by straightforward chain derivation that:

$$\nabla_{xy}^2 f = J_{uvxy} \nabla_{uv}^2 F,$$

(2)

where $\nabla_{uv}^2 = \partial^2 / \partial u^2 + \partial^2 / \partial v^2$ and $J_{uvxy}$ is the Jacobian [16] of the transformation $(x, y) \Leftrightarrow (u, v)$ associated to the conformal mapping:

$$J_{uvxy} = \begin{vmatrix} \frac{\partial u}{\partial x} & \frac{\partial u}{\partial y} \\ \frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} \end{vmatrix}$$

(3)

Hence, if $f$ satisfies the Laplace's equation in the $(x, y)$ plane, $F$ also satisfies it in the $(u, v)$ plane, at least in the points where the Jacobian is not zero:

$$\nabla_{xy}^2 f = 0 \Rightarrow \nabla_{uv}^2 F = 0$$

(4)

Summarizing, conformal transformations preserve Laplace's equation and this is the essential reason for the interest of using conformal mapping.

### 3. A SIMPLE MATLAB PLOTTING ROUTINE

With its very convenient handling of both complex quantities and matrices, a language like Matlab is the ideal programming tool to describe conformal mappings. In the appendix, a complete Matlab script is given to plot the curves in the $(x, y)$ plane that become, under the mapping $w = w(z)$, vertical and horizontal lines in the transformed $(u, v)$ plane.

The adopted strategy is to consider a set of points in a rectangular uniform mesh inside the rectangular domain $\{(x_1, x_2); (y_1, y_2)\}$ of the original $(x, y)$ plane. Then, a 2D complex array $z = x + jy$ containing the coordinates of these points is constructed, and the conformal mapping $w = w(z)$ is defined by a M-function.

Matlab finds the values $[u_1, u_2]$ and $[v_1, v_2]$ bounding the real and imaginary parts of $w = u + jv$ and defines the rectangular domain $\{[u_1, u_2]; [v_1, v_2]\}$. Inside this domain, a rectangular grid of vertical ($u = C$) and horizontal ($v = C$) lines is plotted and the corresponding curves $u(x, y) = C$ and $v(x, y) = C$ are also plotted in the original domain.

For instance, for studying the conformal mapping $w = w(z) = z^2$ in the rectangular domain $[[-2 < x < 2]; [-2 < y < 2]]$ discretized as an uniform grid of $100 \times 100$ points, the essential non-graphical instructions are given in boldface here below (see the complete script Conf_Map in the Appendix):

```matlab
% Define the x-y domain:
x1=-2; x2=2; y1=-2; y2=2;
% Make a grid of MxN points (x,y)
M=100; N=100; xv=linspace(x1,x2,M); yv=linspace(y1,y2,N);
[x,y]=meshgrid(xv,yv);
% Define the complex variable z=x+jy
z=x+i*y;
% Define the conformal mapping w(z)=u+jv
[w,u,v]=complexfunc(z,param);
```
% Find boundaries in the transformed plane
u1=min(min(u)); u2=max(max(u));
v1=min(min(v)); v2=max(max(v));

===============================================================================

% MatLab M-function "complexfunc" defining w(z) and providing its real and imaginary parts u,v.
% "param" is here an eventual function parameter
function [w,u,v] = complexfunc(z,param)
w=z.^2;
u=real(w);
v=imag(w);

In principle this simple script will work for any conformal mapping, although the original rectangular domain, the number of grid points and several plotting parameters may require some fine tuning for specific complex functions. Figures 1 and 2 show the graphical results for, respectively, the well-known mappings \( w = z^2 \) and \( w = \log\left(\frac{z+1}{z-1}\right) \).

**Figure 1. Results of CONF_MAP for \( w = z^2 \)**

**Figure 2. Results of CONF_MAP for \( w = \log\left(\frac{z+1}{z-1}\right) \)**

Particular care must be taken with multivalued functions. For instance, a classical example of the conformal mapping lore is the function \( w = \arccos(z) \), which transforms confocal ellipses and hyperbolas into straight lines. To follow in detail this mapping, we have also visualized how some specific points are transformed (the modifications in the Conf_Map script of Appendix are straightforward).

If we just define the mapping with the corresponding MatLab function \( \arccos \), the whole \( z \) plane is transformed into an infinite vertical strip of width \( \pi \) \((0 \leq u < \pi)\) and it is evident from the figure 3 that every ellipse is transformed into TWO segments at \( v = \pm v_0 \).

If we want every ellipse to be transformed into ONE segment \( v = v_0 \), we need to modify the natural MatLab definition of \( \arccos \), in order that the whole \( z \) plane be transformed into a semi-infinite strip of width \( 2\pi \) \((-\pi < u \leq \pi; v > 0)\). This is just obtained by post-processing all the \( w \)-points having a negative imaginary part. The absolute value of the imaginary part will be used but the real part will change its sign. In other words, we define the mathematical function \( \arccos \) by the three Matlab instructions: \( w=\arccos(z); \ u=u.*\text{sign}(v); \ v=\text{abs}(v) \).

The new result for the modified mapping is shown in figure 4.

**Figure 3. Results of CONF_MAP for a possible definition of \( w = \arccos(z) \):**

\[
w = \arccos(z) ;
\]

**Figure 4. Results of CONF_MAP for another definition of \( w = \arccos(z) \):**

\[
w = \arccos(z); \ u = u.*\text{sign}(v); \ v = \text{abs}(v);
\]

Finally, it is worth pointing out that concatenated conformal mappings (chain transformations) can be easily defined inside the M-function complexfunc, like for instance:

\[
s = \arccos(z) ; \ w = \exp(js) \Rightarrow w = \exp\left[j \arccos(z)\right]
\]

\[
s = \cosh(z) ; \ w = \exp(s) \Rightarrow w = \exp[\arccosh(z)]
\]
combinations which are useful to study finite width slots and strips as clearly shown in figures 5 and 6. Alternate expressions for these mappings are possible by using the logarithmic expressions of the arccos and arcosh functions, but again care must be exerted when selecting branches in the complex plane to obtain the desired mapping and this can depend on the definitions being used for multivalued functions like the logarithm and the square root. For instance, in Matlab we have:

\[ w = \exp\left[ j \arccos(z) \right] = z + j \sqrt{1-z^2} \neq z + \sqrt{z^2-1} \]

**Figure 5. Results of CONF_MAP for** \( w = \exp\left[ j \arccos(z) \right] : \quad w=\exp(j\arccos(z)); \)

**Figure 6. Results of CONF_MAP for** \( w = \exp\left[ \text{arcosh}(z) \right] : \quad w=\exp(\text{acosh}(z)); \)

### 4. POSSIBLE EXTENSIONS TO OTHER DIFFERENTIAL EQUATIONS

Let us now consider a more general partial differential equation (PDE), namely the inhomogeneous Helmholtz equation in two dimensions:

\[ \nabla^2 f + k^2 f(x,y) = g(x,y) \]  

which includes as particular cases the homogeneous Helmholtz equation \( (g = 0) \), the Poisson equation \( (k = 0) \) and the Laplace equation \( (g = 0 ; k = 0) \). Let us apply the conformal mapping (1) to the equation (5). According to the transformation of Laplacian operators, we obtain:

\[ \nabla^2_{uv} F(u,v) = G(u,v) \]  

where we have formally inverted the conformal mapping, expressed the variables \( x,y \) as functions of \( u,v \) and finally written:

\[
\begin{align*}
    f(x,y) &= f[x(u,v), y(u,v)] = F(u,v) \quad ; \quad g(x,y) = g[x(u,v), y(u,v)] = G(u,v)
\end{align*}
\]

In principle, the presence of the Jacobian in equation (6) is a drawback which could jeopardize any potential advantage of the conformal mapping strategy, since now the new equation is definitely more complicated. However, in addition to the Laplace equation, there is an interesting particular case of the Poisson equation where the Jacobian is not a hindrance. This situation arises when we look for the Green's function of a static problem and hence when the inhomogeneous (source) term in the Poisson equation is a Dirac's delta:

\[ \nabla^2 f(x,y) = \delta(x) \delta(y) \]  

In this case, the generalization of a well known Dirac's delta property [17] to the two-dimensional case gives the crucial result:

\[ \delta(u) \delta(v) = \delta(x) \delta(y) / \left| J_{uvxy} \right| \]  

This property is also related to the Change of Variable Theorem [16]. Therefore we have: \( G(u,v) = \left| J_{uvxy} \right| \delta(u) \delta(v) \), which shows that the Poisson equation for Green's functions (7) also transform into itself (assuming for the sake of simplicity a positive Jacobian) and can be written in the transformed domain as:

\[ \nabla^2_{uv} F(u,v) = \delta(u) \delta(v) \]
Finally, the Helmholtz equation for Green's functions

\[ \nabla^2_{xy} + k^2 \] f(x, y) = \delta(x) \delta(y) \quad (10)

becomes only slightly more complicated in the transformed domain:

\[ \nabla^2_{uv} + k^2 / J_{uvxy} \] F(u, v) = \delta(u) \delta(v) . \quad (11)

This corresponds to an "inhomogeneous" medium situation, where the wavenumber \( k \) would be a function of the coordinates. However, for simple Jacobians, an inhomogeneous medium of simple shape could be still be better than a homogenous medium but with a complicated shape.

The final answer will depend, among other issues, of the analytical or numerical techniques being used and should be carefully evaluated in every case.

5. CONFORMAL MAPPING AND STATIC GREEN'S FUNCTIONS

In section 3, the results were associated to classical Laplace problems where the "source" is either a set of finite conductors at a potential different from the reference value at infinite (figures 2, 3, 4) or a set of grounded conductors immersed in a constant electric field (figures 1, 5, 6). Now, following the conclusions of the previous section 4, we can try to consider point charges as sources and to study their effect on a set of grounded conductors. This is equivalent to solve analytically the Poisson's equation with a Dirac's delta source term and Dirichlet boundary conditions, and hence to obtain static Green's functions by using conformal mapping.

5.1 CHECKING THE PROCEDURE WITH SIMPLE SITUATIONS

To check the validity of the theoretical developments discussed in the previous section, we start with two well-known situations, the right angle wedge and the circular cylinder where analytical solutions are available with no reference to the conformal-mapping procedure.

5.1.1 Line source inside a grounded right angle wedge

Let's start with the problem of an infinite line source inside a dihedral right angle at zero potential (a metallic wedge). In the problem cross section (figure) this wedge occupies the positive parts of the \( x \) and \( y \) axes and the unit line source becomes the point \((x_0; y_0)\) as in figure 7.

\[ Figure \ 7. \ \text{The conformal mapping } w = z^2 \ \text{for a right-angle wedge} \]
This is a rather elementary situation. But, let us pretend a sheer electrostatic bluntness and ignore the well known fact that the solution of this problem can be straightforwardly obtained by using image theory (just add three images to the original source, the four sources forming a rectangle centered in the origin of coordinates). Suppose indeed, that our knowledge of image theory reduces to the case of a line source in presence of an infinite ground plane.

Then, we could use the conformal mapping: \[ w = z^2 \quad (u = x^2 - y^2 \quad ; \quad v = 2xy) \]
which transforms the right angle at zero potential into the horizontal plane \( v=0 \), and places the transformed source in the point \((u_0 = x_0^2 - y_0^2 ; v_0 = 2x_0y_0)\).

Now the 2D free space static Green's function, solution of equation (7) is: \[ f = -\log \rho , \]
where \( \rho \) is the source-observer distance. Therefore, using our "limited" knowledge of image theory in the transformed \((u, v)\) domain, we can add a single negative unit image at \((u = u_0; v = -v_0)\), accounting for the horizontal axis at zero potential and write directly the solution into the transformed plane as:

\[ f = \log \frac{\sqrt{(u-u_0)^2 + (v+v_0)^2}}{\sqrt{(u-u_0)^2 + (v-v_0)^2}} \quad (12) \]

It is a straightforward but tedious (I will even say painful) exercise, and therefore it is left as an exercise to the reader, to show that when in the above expression we replace \( u, v, u_0, v_0 \) by their expressions in function of \( x, y, x_0, y_0 \), we recover the four-term expression that could have directly obtained with "advanced" image theory:

\[ f = \log \frac{\sqrt{(x-x_0)^2 + (y+y_0)^2}}{\sqrt{(x-x_0)^2 + (y-y_0)^2}} \frac{\sqrt{(x+x_0)^2 + (y-y_0)^2}}{\sqrt{(x+x_0)^2 + (y+y_0)^2}} \quad (13) \]

There is perhaps a better way to convince the modern or the younger reader of the equivalence of expressions (12) and (13) and it consists in writing a short MatLab script. And the \textit{sine qua non} condition to be convincing is to be concise and to follow strictly and in a transparent way the corresponding mathematical reasoning. Therefore, the script must consist of the following five steps:
a) define arrays (matrices) \( x,y \) containing coordinates of observation points inside a rectangular domain; 
b) define the complex mapping; 
c) define the source point \( x_0, y_0 \); 
d) find the transformed coordinates \( u_0, v_0 \) of the source; 
e) write the formula for the Green's function in the transformed \( w \) plane and depict it as a function of the original coordinates;

Here are the corresponding Matlab lines, where "complexfunc" is the same M-function used in the previous script.

```matlab
% A) Defining a grid in the x-y plane
x1=0;   x2=10;   y1=0;   y2=10;   xv=linspace(x1,x2,100);
yv=linspace(y1,y2,100);   [x,y]=meshgrid(xv,yv);
% B) Introducing a conformal mapping \( w = w(z) \) with \( z = x+jy \) and \( w = u+jv \)
z=x+j*y;   [w,u,v] = complexfunc(z,0);
% C) Point source coordinates in x-y plane
x0=4;   y0=2;
% D) New coordinates of the source in the u-v plane
z0=x0+j*y0;   [w0,u0,v0] = complexfunc(z0,0);
% E) Image solution for the potential in the u-v plane
pot=-0.5*log((u-u0).^2+(v-v0).^2)+0.5*log((u-u0).^2+(v+v0).^2);
```

The full script (Poisson) is given in the Appendix. It looks much longer but the essential lines are those given above and the rest is essentially comments, checks and plotting instructions. Obviously, when the Green's function is available via another method (in this case "advanced image theory") it is useful to compute the alternate expression of the Green's function and to check that we obtain the same numerical results by computing the error between both expressions.

The results are graphically depicted in figure 8, which show quite convincingly that the procedure is correct. As a check, the difference between the conformal mapping result and the alternative image formulation with four sources in the original \((x,y)\) plane formula (corresponding to the parameter \( \text{error} \) in the script Poisson of Appendix) is within the numerical noise \(<10^{-15}\).

**Figure 8. Results of POISSON for** \( w = z^2 \): \( w = z^{^2} \).

### 5.1.2 A line charge facing a metallic grounded circular cylinder

This is another classical problem. We use the conformal mapping \( w = 1/z \) to transform the circular cylinder into a plane. Let's consider in the problem's cross section (figure 9) the cylinder represented by a circle of radius \( R \) with its center at \((R;0)\). The unit source will be at the point \((x_0; y_0 = 0)\).
Then the conformal mapping will transform the circle in the vertical line $u = 1/2R$, while the source point will be now $(u_0 = 1/x_0; v_0 = 0)$, as shown in figure 9. Therefore, the solution in the transformed plane will just call for one negative unit image at $(u = 1/R - u_0; v = 0)$ and we can write directly the solution into the transformed plane as:

$$f = \log \frac{(u + u_0 - 1/R)^2 + v^2}{(u - u_0)^2 + v^2}$$

(14)

Again, it is straightforward (?) to check that this solution, when transferred back to the original plane, is identical to the traditional solution which adds a negative unit image at $(x = x_0R/(x_0 - R); y = 0)$ and a constant $f_0 = -\log[R/(x_0 - R)]$

(Careful, this is not to be confused with the image solution of the sphere problem!)

The conformal mapping results are graphically depicted in figure 10, which show quite convincingly that the procedure is correct. As a check, the difference between the conformal mapping result and the alternative formulation replacing directly in the original $(x,y)$ plane the cylinder by an image source formula is again within the numerical noise ($\text{error} < 10^{-15}$)

5.2 MORE INVOLVED GREEN'S FUNCTIONS EXAMPLES

We move now to problems which lack an analytical solution by other elementary methods, and we consider several interesting examples where a line source is 1) inside parallel plates, 2) in the presence of an elliptical cylinder and 3) facing a slot in a ground plane.
5.2.1 A line charge inside two grounded parallel plates

This is a very important geometry, encountered directly or as an intermediate step in many technological and geometry-modelling problems. The traditional solution is obtained as a slowly convergent infinite series of images. However, it is scarcely pointed out that conformal mapping provides here again a closed-form analytical solution.

In the problem's cross-section, we consider two horizontal lines $y = 0$ and $y = d$ at zero potential and inside them a source at $(x = 0, y = y_0)$ (figure 11).

**Figure 11. The conformal mapping $w = \exp(\pi z/d)$ for a parallel-plate waveguide**

The conformal mapping $w = \exp(\pi z/d)$ will transform the two horizontal lines at zero potential into the $u$-axis ($v = 0$) and will put the source at $u_0 = \cos(\pi y_0/d)$; $v_0 = \sin(\pi y_0/d)$. Therefore, a single negative unit image at $(u = u_0, v = -v_0)$ should be enough and the solution is again given by:

$$f = \log \frac{(u-u_0)^2 + (v+v_0)^2}{(u-u_0)^2 + (v-v_0)^2}$$  \hspace{1cm} (16)

This expression can be, if needed, easily cast into the original $(x, y)$ plane by using the equations: $u = \exp(\pi x/d) \cos(\pi y/d)$; $v = \exp(\pi x/d) \sin(\pi y/d)$ and the above given values of $(u_0; v_0)$.

This closed form is to be compared with a traditional solution using an infinite set of images, given by:

$$f = \sum_{n=-\infty}^{n=+\infty} \log \frac{(x-x_0)^2 + (y+y_0+2nd)^2}{(x-x_0)^2 + (y-y_0+2nd)^2}$$  \hspace{1cm} (17)

The figure 12 shows the exact results obtained with a conformal mapping. Using the infinite series, the average relative error is 4.3%, 2.2% and 0.9% for respectively $n=10, 20, 50$. This slow convergence shows clearly the interest of conformal mapping.
5.2.2 A line charge parallel to an elliptical cylinder

The conformal mapping \( w = \arccos(z/d) \) must also be useful for the case of a line source inside or outside a grounded elliptical cylinder. In the cross section, the cylinder will be represented by an ellipse of semi-axes \( a \) and \( b \) (with \( d^2 = a^2 - b^2 \)), centred at the origin of coordinates. This problem has, obviously, no image solution. Let us start with a source in a position \( (x_0, y_0) \) outside the ellipse and in the first quadrant. The equation of the ellipse is \((x/a)^2 + (y/b)^2 = 1\), and from the inverse mapping \( z = d \cos(w) \), we can easily obtain the equation: \((x/d \cosh u)^2 + (y/\sinh u)^2 = 1\)

Therefore, we infer that the ellipse must become the two horizontal lines \( v = \pm v_e = \arccosh(a/d) \) and that the new source is located at \( u_0 + jv_0 = \arccos((x_0 + jy_0)/d) \) as depicted in figure 13. In this figure we have also included the findings of section 3, relative to the multivalued behaviour of the arccos function.

Since our original source is outside the ellipse, the transformed source is outside the region limited by the parallel lines \( v = v_e \). If we were facing a physical situation, the grounded infinite plane closer to the source would fully screen it and would make the second ground plane irrelevant. Hence, we would need to consider in the cross section (figure 13) only the grounded line which is closer to the source. Moreover, since Matlab will give us a single solution for the transformed source point \((u_0, v_0)\), we could be tempted (naively) to write as solution:
where
\[
f = \log \frac{\sqrt{(u-u_0)^2 + (v+v_0-2v_g)^2}}{\sqrt{(u-u_0)^2 + (v-v_0)^2}}
\]
(19)

But then, using our Matlab script Poisson, we would obtain the surprising result of fig. 14:

Figure 14. Incorrect results of POISSON for \( w = \arccos(z/d) \)

There are two mistakes in the implementation leading to this result and both are due to the fact that the function \( \arccos \) is multivalued and that every \( z \) point transforms into an infinite set of \( w \) points.

First, it is obvious from fig. 14 that our solution does not see the points outside the ellipse but on the lower half-plane \( (y < 0) \). We have mentioned that, although the ellipse is transformed into two horizontal lines, in the case of a source outside the ellipse, we will need only to take images respect to the horizontal line closer to the transformed source. Since Matlab defines the domain of the \( \arccos \) function as an infinite vertical strip \( 0 \leq u \leq \pi \), there won't be available results in the two quadrants "screened" by the second grounded horizontal line. But we have shown in section 3 how to solve this problem: we just define: \( w = \arccos(z) \Rightarrow [ w=\arccos(z); \ u=u.*\text{sign}(v); \ v=\text{abs}(v); ] \)

This will force the whole \( z \) plane to be transformed into a semi-infinite strip \( 0 \leq u \leq \pi; \ v \geq 0 \) and now all the calculation points are in the correct side respect to the source (fig. 13).

With this corrected definition of \( \arccos \), the equation (19) would provide values in the four quadrants of the \( z \)-plane but they would be still wrong. For we can use image theory only if the zero potential line is infinite. So, we need to consider the full range of real \( u \) values in figure 13. But this means that the point source at \((x_0;y_0)\) will be transformed into an infinite set of sources \( (u_0 + 2k\pi; \pm v_0) \) and we need to include the effect of all them, as well as the effect of the images sources respect to the nearest ground plane. Finally, the correct solution can be written as:

\[
f = \sum_{n=-\infty}^{n=+\infty} \log \frac{\sqrt{(u-u_0 + 2n\pi)^2 + (v+v_0 - 2v_g)^2}}{\sqrt{(u-u_0 + 2n\pi)^2 + (v-v_0)^2}}
\]
(20)

When these two modifications are implemented, we obtain correct results, as shown in figure 15, where the infinite sum has been truncated to \( n=20 \). Adding a new iteration produces a relative change in the values whose maximum value is 0.0002, so \( n=20 \) can be considered more than enough for most practical purposes.

Figure 15. Correct results of POISSON for \( w = \arccos(z/d) \)

The treatment of an internal source (line inside a grounded elliptical cylinder) is similar, but not identical to the external source problem and also leads to very interesting implementation problems. Following our usual policy, it is also left as an exercise for the committed reader.
5.2.3 A line charge parallel to a grounded metallic strip.

Obviously, the finite strip of width $2d$ is included as a degenerate case of the elliptic cylinder discussed in the previous section. Therefore the unique modification in the formula (20) is that now $v_e = 0$. Figure 16 shows the convincing analytical result.

Figure 16. A limiting case of fig. 15 where the ellipse becomes a strip

5.2.4 A line charge parallel to a slotted ground plane

An amazing application of the parallel-plate geometry is the problem of a line source facing a ground plane which includes a slot parallel to the line source. This is an interesting problem in Electromagnetic Compatibility, as it is the simplest model for the penetration of an electric field through a long slit in a metallic screen. Also it provides the starting point for computing characteristic impedances of slot lines. In the cross section the slotted plane is defined by the portions $x < -d$ and $x > d$ of the real axis and the source is at the point $(x_0; y_0)$ (figure 17).

Figure 17. The conformal mapping $w = \exp\left[j \arccos(z/d)\right]$ for a slotted ground plane

This problem has no trivial solution but the conformal mapping of the previous section 5.2.3: $s = p + jq = \arccos(z/d)$ with $d^2 = a^2 - b^2$ should do the job. Indeed this mapping transforms the slotted plane into the parallel vertical lines $p = 0$ and $p = \pi$ whereas the source becomes the inner point $p_0 + jq_0 = \arccos((x_0 + jy_0)/d)$. So, we have now the parallel plate geometry discussed in section 5.2.1 and we know that we must introduce a second conformal mapping $w = \exp(js)$.

The final composite transformation is: $w = \exp\left[j \arccos(z/d)\right] = (z/d) + j\sqrt{1-(z/d)^2}$ and the static Green's function has the remarkably simple expression

$$f = \log \frac{(u-u_0)^2 + (v+v_0)^2}{(u-u_0)^2 + (v-v_0)^2}$$

(18)

where, again, it wouldn't be difficult to introduce the original coordinates $(x, y)$.

The figure 18 shows a typical result with the line source above the edge of the infinite slot.
6. CONCLUSIONS
The above examples show conclusively the ability of conformal mapping to provide analytical expressions for two-dimensional Green's functions associated to Poisson's equation, i.e. electrostatic potentials created by infinite line sources parallel to metallic infinite cylinders of various shapes. The solution is naturally expressed in the transformed domain, but usually a direct expression in the original \((x,y)\) coordinates is easily obtained. In most cases, the proposed solutions are an order of magnitude simpler than those obtained by other approaches, like the infinite series of image theory.

The concatenated use of conformal mapping brings within practical reach many 2D shapes, including useful problems like the strip or the slot, which are of paramount relevance in computing the electrical parameters of printed transmission lines.

Increasingly more involved geometries can be afforded by the use of other sophisticated conformal mappings like implicit, iterated or Schwartz-Christoffel transformations. The conformal mappings dictionaries \([11-13]\) are full of examples waiting implementation. For instance, the static line source inside an infinite rectangular waveguide has a conformal mapping solution which is certainly more numerically efficient than the traditional double infinite series of images.

Also, conformal mapping can be advantageously used in some problems involving inhomogenous dielectrics and to reduce an open infinite domain to a closed finite one.

True, conformal mapping seems less useful for full-wave situations, since it doesn't apply directly to the Helmholtz equation. But it is well known that Laplace and Poisson equations are ubiquitous in some full-wave situations (for instance TEM modes) and even the Helmholtz equation could benefit of a conformal mapping for some specific geometries. In any case, the static Green's function can be used as a part of the full wave analysis (for instance in regularisation and extraction-of-the-singularity techniques) or for providing useful results in the limiting low frequency range, like the polarizabilities of small slots.

In addition, conformal mapping can be considered as an useful pre-processing geometrical tool when solving problems with numerical techniques, like finite differences and finite and boundary element methods.

Last but no least, conformal mapping is a clean technique, with an obvious aesthetical appeal, easily implemented in modern computing languages and leading to compact and user-friendly codes. In particular we hope to have convincingly demonstrated the natural affinity between a language like Matlab and conformal mapping. The implementation follows closely the mathematical formulas and, like in Fortran, we can fully work with complex equations. This means that the analytical extraction of the real and imaginary part of the function \(w(z)\) (the redoubtable \(u(x,y)\) and \(v(x,y)\) functions, recurrent nightmares of so many past undergraduates) is not necessary at all. But in addition, the matrix treatment make unnecessary in many situations to use the loops and control-flow statements that frequently clog computer code-writing.

The author includes routinely this material in a second-year undergraduate course in Electromagnetics. Since at this moment, students are getting acquainted in our School with Advanced Calculus and Matlab, the developments presented in this paper help the students to...
accept the today not so popular complex variable and partial differential equations subjects, while providing them with challenging examples to further develop their programming abilities. It is just a sad and regretful affair that the quaternions of William Rowan Hamilton and other similar mathematical ideas of past centuries never evolved into an efficient and powerful theory of hypercomplex numbers. This would have fulfilled one of the most cherished dreams of the electromagnetic aficionados: to compute three-dimensional Green's functions due to point sources with the same ease than the two-dimensional ones shown in this paper.

ACKNOWLEDGEMENTS

Thanks are given to Mr. Daniel Llorens del Rio (LEMA-EPFL, Switzerland), for the non MatLab artwork and for many useful suggestions.

REFERENCES

APPENDIX : THE MATLAB SCRIPTS (available in http://itopwww.epfl.ch/LEMA)

% CONF_MAP
% This MATLAB script studies the conformal mapping \( w = u+jv = w(z=x+jy) \) by considering first
% a rectangular domain \([x_1,x_2,y_1,y_2]\) in the \( z=x+jy \) plane.
% Then, the extreme values \([u_1,u_2,v_1,v_2]\) of \( u=\text{real}(w) \) and \( v=\text{imag}(w) \) are found.
% Finally a cartesian grid with equispaced values \( u_i, v_j \) is defined and plotted
% in the \( u+jv \) plane inside the rectangular domain \([u_1,u_2,v_1,v_2]\)
% and the curves \( u(x,y)=u_i \) and \( v(x,y)=v_j \) are also plotted in the \( z=x+jy \) plane.
% The conformal mappig is defined by the function \([w,u,v]=\text{complexfunc}(z)\)

% Juan R. Mosig, LEMA-EPFL-Switzerland, v1.3, June 2003

clear all

% Define the x-y domain:
x1=-2
x2=2
y1=-2
y2=2

% Make a grid of MxN points (x,y)
M=100
N=100
xv=linspace(x1,x2,M);
yv=linspace(y1,y2,N);
[x,y]=meshgrid(xv,yv);

% Define the complex variable \( z=x+jy \)
z=x+j*y;

% Define the conformal mapping \( w(z)=u+jv \)
[w,u,v]=complexfunc(z,0);

% PLOTS IN THE Z=X+JY PLANE
subplot(1,2,1)
nu=11;
nv=11;
u1=min(min(u));
u2=max(max(u));
un=linspace(u1,u2,nu);
[C,h]=contour(x,y,u,un);
set(h,'LineWidth',2);
axis equal
axis([x1,x2,y1,y2])
hold on
v1=min(min(v));
v2=max(max(v));
vn=linspace(v1,v2,nv);
[C,h]=contour(x,y,v,vn,'--');
set(h,'LineWidth',2);
hold off

% PLOTS IN THE W=U+JV PLANE:
subplot(1,2,2)
% POISSON: Test of conformal mapping for line sources.
% POISSON1: a line source facing a right-angle wedge.
% In the cross-section, the wedge is defined by the semi-axes x>0 and y>0
% and the source is at (x0,y0).
% This problem has a direct solution in term of images
% (we add 3 images to the original source).
% But it should be possible to use the conformal mapping w=z^2
% to transform the 90 degrees angle into the x-axis.
% Then the solution in the transformed plane requires only
% the most basic image procedure (effect of a ground plane).
% The conformal mappig is defined by the function [w,u,v]=complexfunc(z,param)

% Juan R. Mosig, LEMA-EPFL-Switzerland, v1.2, May 2003

clear all

% Defining a grid in the x-y plane
x1=0
x2=10
y1=0
y2=10
xv=linspace(x1,x2,100);
yv=linspace(y1,y2,100);
[x,y]=meshgrid(xv,yv);

% Introducing a conformal mapping w=w(z) with z=x+iy and w=u+jv
z=x+j*y;
[w,u,v] = complexfunc(z,0);

% Point source coordinates in x-y plane
x0=2
y0=2
% New coordinates of the source in the u-v plane
z0=x0+j*y0
[w0,u0,v0] = complexfunc(z0,0);

% Solution for the potential in the u-v plane
pot=-0.5*log((u-u0).^2+(v-v0).^2)+0.5*log((u-u0).^2+(v+v0).^2);

% Plotting the potential in the x-y plane
level=linspace(0,0.5,11);
contourf(x,y,pot,level)
axis equal
axis([x1,x2,y1,y2])
% CHECKING THE CONFORMAL MAPPING APPROACH
% Direct solution for the potential in the x-y plane with 3 images
potcheck=-log((x-x0).^2+(y-y0).^2)+log((x-x0).^2+(y+y0).^2)-log((x+x0).^2+(y+y0).^2)+log((x+x0).^2+(y-y0).^2);
potcheck=potcheck/2;

% ERROR (difference between POT and POTCHECK) should be zero
error=sum(sum(abs(pot-potcheck)))/sum(sum(abs(pot)))

% COMPLEXFUNC: M-function defining the conformal mapping and called by CONF_MAP and POISSON
% "complexfunc" defines a complex function W=W(Z)=U+JV
% PARAM is a parameter that can be included in the definition of the complex function W(Z)
% The function provides also as outputs U=real(W) and V=imag(W)
% Z,W, U and V are Matlab arrays of the same dimensions

function [w,u,v] = complexfunc(z, param)
w=z.^2;
u=real(w);
v=imag(w);

% Other possibilities used in this paper
% w=1./z;
% w=exp(pi*z/param);
% w=exp(j*acos(z/param));
% w=acos(z/param);

% u&v definitions to be used with the acos function:
% u=u.*sign(v);
% v=abs(v);
Figure 1. Results of CONF_MAP for $w = z^2$

$w = z^2; \ x_1 = -2; \ x_2 = 2; \ y_1 = -2; \ y_2 = 2$
Figure 2. Results of CONF_MAP for $w = \log\left(\frac{z+1}{z-1}\right)$

$w = \log((z+a)/(z-a))$; $a = 1$; $w = z^2$; $x1 = -2$; $x2 = 2$; $y1 = -2$; $y2 = 2$
Figure 3. Results of CONF_MAP for a possible definition of $w = \arccos(z)$

\[ w = \cos(z); \quad x_1 = -2; \quad x_2 = 2; \quad y_1 = -2; \quad y_2 = 2 \]
Figure 4. Results of CONF_MAP for another definition of  \( w = \arccos(z) \)

\[ w = \text{acos}(z); \quad u = u \cdot \text{sign}(v); \quad v = \text{abs}(v); \quad x_1 = -2; \quad x_2 = 2; \quad y_1 = -2; \quad y_2 = 2 \]
Figure 5. Results of CONF_MAP for \[ \text{\textit{w}} = \exp\left( j \arccos(z) \right) \]

\( w=\exp(j\arccos(z)); \quad x_1=-2; \quad x_2=2; \quad y_1=-2; \quad y_2=2 \)
Figure 6. Results of CONF_MAP for \( w = \exp[\text{arccosh}(z)] \)

\[ w = \exp(\text{acosh}(z)); \quad x_1 = -2; \quad x_2 = 2; \quad y_1 = -2; \quad y_2 = 2 \]
Figure 8. Results of POISSON for $w = z^2$

$w = z^2$; $x_0 = 2$; $y_0 = 2$; $x_1 = 0$; $x_2 = 10$; $y_1 = 0$; $y_2 = 10$
Figure 10. Results of POISSON for $w = 1/z$

$w=1/z; \ x_0=4; \ y_0=0; \ x_1=-2; \ x_2=6; \ y_1=-4; \ y_2=4$
Figure 12. Results of POISSON for \( w = \exp(\pi z / d) \)

\[
w = \exp(\pi^2 z / \text{param}); \quad x_0 = 0; \quad y_0 = 0.25; \quad x_1 = -1; \quad x_2 = 1; \quad y_1 = 0; \quad y_2 = 1
\]
Figure 14. Incorrect results of POISSON for \( w = \arccos(z/d) \)

\[ w = \arccos(z/\text{param}); \quad x_0 = 1; \quad y_0 = 2; \quad x_1 = -5; \quad x_2 = 5; \quad y_1 = -5; \quad y_2 = 5 \]
Figure 15. Correct results of POISSON for \( w = \arccos(z/d) \)

\[
\begin{align*}
    w &= \arccos(z/\text{param}); \\
    u &= u.*\text{sign}(v); \\
    v &= \text{abs}(v); \\
    x0 &= 1; \\
    y0 &= 2; \\
    x1 &= -5; \\
    x2 &= 5; \\
    y1 &= -5; \\
    y2 &= 5;
\end{align*}
\]
Figure 16. \textit{POISSON for a limiting case of $w = \arccos(z/d)$}

\begin{verbatim}
w = \arccos(z/\text{param}); \quad u = u.*\text{sign}(v); \quad v = \text{abs}(v); \quad x0 = 2; \quad y0 = 2; \quad x1 = -5; \quad x2 = 5; \quad y1 = -5; \quad y2 = 5
\end{verbatim}
Figure 18. Results of POISSON for \( w = \exp\left[j \arccos\left(\frac{z}{d}\right)\right] \)

\[ w = \exp(j \arccos(z/d)); \quad d=2; \quad x0=2; \quad y0=1; \quad x1=-5; \quad x2=5; \quad y1=-5; \quad y2=5 \]