Computational Physics

Project 4: Laplace's Equation

Due: February 26th 2018

George Davila[1]*

1 Department of Physics, University of Central Florida, 4111 Libra Drive, Physical Sciences Bldg. 430, 32826, Orlando, United States

Abstract: Here we discuss project #4. We review the context of the problem, its relation to real physical systems, and the methodology by which we tackled its solution. All figures are appended at the end of the paper (pages 7 - 14).

1. Introduction

Here we use iterative methods to find solutions to Laplace's equation. Laplace's equation appears throughout many different sectors of physics. Most predominantly in electromagnetism. But it can also be used to things such as gravitational and fluid potentials. Moreover it appears as a term in many other important equations, for example the heat equation [4]

$$\frac{\partial u}{\partial t} - \alpha \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} \right) = 0 \tag{1}$$

So that if the field u is time-independent it simply reduces to Laplace's equation. And of course a Laplacian term similarly appears with the Schrdinger equation, which also reduces to Laplace's equation under the right circumstances. Already the importance of Laplace's equation and its generalizations are apparent, and of course such a brief discussion hardly does it any justice.

Laplace's equation is primarily dependent on a given set of conditions, usually with some set of constraints on the solutions at the boundary or within the interior. Both the cases we will discuss below vary in the constraints applied. While we mainly speak of electrostatic potential here, as noted above, Laplace's equation and the cases herein could also be described in terms of fluid potentials.

* *E-mail: GDavila@knights.ucf.edu*

Furthermore, from a purely computational perspective, the scenarios described herein certainly provide a good test of the iterative schemes used. Not only do we deal with second order derivatives, but we also apply strict boundary conditions that must be satisfied.

Here we ddeal with a 40×40 box with or without 'plates' withing its interior. We describe the specifics of this in §4.

2. Analytics

We begin with the equation

$$\frac{\partial^2 V}{\partial x^2} + \frac{\partial^2 V}{\partial y^2} + \frac{\partial^2 V}{\partial z^2} = \frac{\rho}{\epsilon_0} = 0 \tag{2}$$

Where

- V is the electrostatic potential
- $\rho = 0$ is the (net) electrostatic charge density, taken here to be zero
- $\epsilon_0 = 8.85418782 \times 10^{-12} Fm^{-1}$, is the vacuum permittivity
- θ is the angle and $\theta_0 \equiv \theta(t = 0)$

So clearly this is just Laplace's equation so that we may also write it as $\nabla^2 V = \Delta V = 0$. We concern ourselves with two related but distinct physical scenarios:

1. **Unplated** with $V = \pm 1$ at boundaries.
2. **Plated** with the constraint that $V = 0$ at the boundaries, so this is not simply a smaller version of the unplated case.

Clearly for the **Unplated** case we simply have the linear relation

$$V = -\int \vec{E} \cot d\vec{x} = \frac{\rho}{\epsilon_0} x + V_0 \tag{3}$$

Where we assume constant Electric field $\vec{E}$ and constant surface charge density ρ (of the boundaries). And V_0 is just a constant dependent on our choice of coordinates. If we choose coordinates such that $x = 0$ at a point where $V = 0$ (the center in these examples) then $V_0 = 0$. This is simply the same as the voltage between two capacitance plates.

But for the **Plated** case, we essentially have each plate interacting with 3 others.

Remark 2.1.
Lets identify some things that should happen in the plated case:

- Potential on boundary of box $j = 0, 40$ should be zero
- Potential at center of box should be zero
- Potential lines traveling through the plates between the boundaries should increase/decrease very nearly linearly
- Potential lines that don't hit the plates should have a wave-like form with peaks at the x-value of the plates

3. Discrete Approximation

To a rough approximation we have that

$$\frac{\partial V}{\partial x} \approx \frac{\Delta V}{\Delta x} = \frac{V(i+1,j,k) - V(i,j,k)}{\Delta x} \tag{4}$$
$$\frac{\partial V}{\partial y} \approx \frac{\Delta V}{\Delta y} = \frac{V(i,j+1,k) - V(i,j,k)}{\Delta y} \tag{5}$$
$$\frac{\partial V}{\partial z} \approx \frac{\Delta V}{\Delta z} = \frac{V(i,j,k+1) - V(i,j,k)}{\Delta z} \tag{6}$$

Where i is the step for the x coordinate, j for the y, and k for the z. These approximations clearly hold best when V changes only very slowly and when V is monotonic (at least in the applicable regime). These relationships are derived from the definition of derivatives themselves, but rather than a step size $h \to 0$ we use an appropriately small step size. Further we expect that electrostatic potentials obeying Eq. (2) should change fairly smoothly and slowly (relative to the step size we seek to implement). Obviously if we only used very few steps we would get a significant amount of error.

Now we can write the approximation to the second derivatives

$$\frac{\partial^2 V}{\partial x^2} \approx \frac{V(i+1,j,k) - 2V(i,j,k) + V(i-1,j,k)}{(\Delta x)^2} \tag{7}$$
$$\frac{\partial^2 V}{\partial y^2} \approx \frac{V(i,j+1,k) - 2V(i,j,k) + V(i,j-1,k)}{(\Delta y)^2} \tag{8}$$
$$\frac{\partial^2 V}{\partial z^2} \approx \frac{V(i,j,k+1) - 2V(i,j,k) + V(i,j,k-1)}{(\Delta z)^2} \tag{9}$$

And the indices are arbitrary, so we could make any choice for $n \in \mathbb{Z}$ such that $i \leftrightarrow i+n$, here we just have that $n = 0$ for the center step. From Eq (2) we then have

$$\frac{V(i+1,j,k) - 2V(i,j,k) + V(i-1,j,k)}{(\Delta x)^2} + \frac{V(i,j+1,k) - 2V(i,j,k) + V(i,j-1,k)}{(\Delta y)^2} + \frac{V(i,j,k+1) - 2V(i,j,k) + V(i,j,k-1)}{(\Delta z)^2} = 0 \tag{10}$$

We now assume isotropic steps such that $\Delta x = \Delta y = \Delta z$, giving that

$$V(i,j,k) = \frac{1}{6}\left[V(i+1,j,k) + V(i-1,j,k) + V(i,j+1,k) + V(i,j-1,k) + V(i,j,k+1) + V(i,j,k-1)\right] \quad (11)$$

So each potential is the sum of its nearest neighbors in each direction. For lower dimensional potentials this simply reduces to

$$V(i,j) = \frac{1}{4}\left[V(i+1,j) + V(i-1,j) + V(i,j+1) + V(i,j-1)\right] \text{ (2-dimensional)} \quad (12)$$

$$V(i) = \frac{1}{2}\left[V(i+1) + V(i-1)\right] \text{ (1-dimensional)} \quad (13)$$

And we set the potential at the boundaries to be some given value. For the code *laplace*2.*f* we model a 2-dimensional potential and set $V = -1, +1$ at the opposite sides.

4. Computations

The codes used are shown in the Figures appended.

Work with a 40×40 lattice with each node representing different points in space. Do 1000 iterations for both. These codes therefore used the 2D potential approximation

$$V(i,j) = \frac{1}{4}\left[V(i+1,j) + V(i-1,j) + V(i,j+1) + V(i,j-1)\right] \quad (14)$$

The physical scenarios described previously are associated with very particular parameters

1. (Unplated) 40×40 lattice with potentials at boundaries such that $V = -1$ at $i = 1$ and $V = +1$ at $i = 40$
2. (Plated) 40×40 lattice with potentials inside boundaries such that $V = -1$ at $i = 15$, $j \in [10, 30]$ and $V = +1$ at $i = 25$, $j \in [10, 30]$. ***But*** we also have the constraint that the poential is zero $V = 0$ at the boundaries, so this is not simply a smaller version of the unplated case.

We append the cases of 500 iterations as the last 2 figures. As we can see, these are essentially the same as their—technically more accurate—1000 iteration counterparts. These plots are stylized differently, however, since they were generated in *Excel*.

4.1. Unplated

We obtain a simple linear plot (Fig. 3) between the two boundaries. Clearly this agrees with our analytic expression Eq. (3). It is such that if we start at any point on the boundaries $i = 1, 40$ and move towards the

other side we will follow this same potential line. Moving up or down in the j direction will, however, not change the potential. A 3D plot is given in Fig. 4.

Remark 4.1.
We concern ourselves with 3 plots of varying iteration:

- 200 iterations **(does NOT compile)**
- 500 iterations **(does compile)**, plotted below in Fig. 7
- 1000 iterations **(does compile)**, plotted below in Fig. 4

4.2. Plated

For the case of 1D motion along the lattice (Fig. 5) between the boundaries in the plated case, we have quite different behavior. We can see that the potentials peek at the plates and are 0 at the center and the boundaries. Fig. 5 illustrates this well, so it is included to supplement the 3D plot. Beginning at a specific j (series $\#j$) it show the potential as we move left and right along x-axis. And going from one series to another corresponds to vertical motion. A 3D plot is given in Fig. 6 and shows the same behavior.

Remark 4.2.
We concern ourselves with 3 plots of varying iteration:

- 200 iterations **(does NOT compile)**
- 500 iterations **(does compile)**, plotted below in Fig. 8
- 1000 iterations **(does compile)**, plotted below in Fig. 6

Clearly these computations satisfy the conditions of Remark 2.1 very well.

5. Conclusions

These agree well with what we would expect from analytic calculation and simple intuition. The unplated and plated simulations satisfy Eq. (3) and Remark 2.1 respectively. As we can see the 3D potentials generated by iterative methods in Fig. 4 and Fig. 6 not only accurately portray what's going on. But the fact that they're discrete isn't even immediately apparent. As such these are very good approximations.

References

[1] S. Stolbov, Computational Physics Class Notes

[2] B. Ydri, *Computational Physics: An Introduction to Monte Carlo Simulations of Matrix Field Theory*, arXiv:1506.02567 [hep-lat] (June 2015)

[3] K. Anagnostopoulos, *Computational Physics Volume 2: A Practical Introduction to Computational Physics and Scientific Computing* pg. 91 (Aug. 2014)

[4] Weisstein, Eric W. *Heat Conduction Equation.* From MathWorld–A Wolfram Web Resource.

Figure 1. One of the codes used, entitled *laplace.f*, that we wrote in class.

student12@teaching:~/proj4

```
      program laplace
      real v(40,40),v1(40,40)
      print*,"Enter the number of iterations"
      read(5,*) ni
      open(7,file="potential")
111   format(40(2x,e12.5))
      v=0.0
      do j=1,40
        v(1,j)=-1.0
        v(40,j)=1.0
        v1(1,j)=-1.0
        v1(40,j)=1.0
      enddo
      do i=2,39
        v(i,1)=v(i-1,1)+0.051282
        v(i,40)=v(i,1)
        v1(i,1)=v(i,1)
        v1(i,40)=v(i,40)
      enddo
      do ii=1,ni ! iterations
        call update(v,v1,40)
        call update(v1,v,40)
        dv=0.0
        do i=2,39 ! error
          do j=2,39
            err=abs(v1(i,j)-v(i,j))
            if (err.gt.dv) then
              dv=err
            endif
          enddo
        enddo ! error
        if(dv.lt.1.0e-5) then
          goto 55
        endif
      enddo !iterations
55    continue
      if(dv.gt.1.0e-5) then
        print*, "Potential didn't converge!!"
        print*, "dv=",dv
      endif
      do j=1,40
        write(7,111) (v(i,j),i=1,40)
      enddo
      end

      subroutine update(v,v1,n)
      real v(n,n),v1(n,n)
      do i=2,n-1
        do j=2,n-1
          v1(i,j)=0.25*(v(i+1,j)+v(i-1,j)+v(i,j+1)+v(i,j-1))
        enddo
      enddo
      return
      end
```

Figure 2. One of the codes used, entitled *laplace2.f*, that we wrote in class. Can see that we added the *plates* subroutine.

student12@teaching:~/proj4

```
      program laplace
      real v(40,40),v1(40,40)
      print*,"Enter the number of iterations"
      read(5,*) ni
      open(7,file="lap2potential")
111   format(40(2x,e12.5))
      v=0.0
      v1=0.0
         do j=1,40
         v(40,j)=1.0
         v1(1,j)=-1.0
         v1(40,j)=1.0
       enddo
       do i=2,39
         v(i,1)=v(i-1,1)+0.051282
         v(i,40)=v(i,1)
         v1(i,1)=v(i,1)
         v1(i,40)=v(i,40)
       enddo
      call plates(v,40)
      call plates(v1,40)
      do ii=1,ni ! iterations
        call update(v,v1,40)
        call plates(v1,40)
        call update(v1,v,40)
        call plates(v,40)
        dv=0.0
        do i=2,39 ! error
          do j=2,39
            err=abs(v1(i,j)-v(i,j))
            if (err.gt.dv) then
              dv=err
            endif
          enddo
        enddo ! error
        if(dv.lt.1.0e-5) then
          goto 55
        endif
      enddo !iterations
 55   continue
      if(dv.gt.1.0e-5) then
        print*, "Potential didn't converge!!"
        print*, "dv=",dv
      endif
      do j=1,40
        write(7,111) (v(i,j),i=1,40)
      enddo
      end

      subroutine plates(v,n)
      real v(n,n)
      do j=10,30
        v(15,j)=-1.0
        v(25,j)=1.0
      enddo
      return
      end
      subroutine update(v,v1,n)
      real v(n,n),v1(n,n)
      do i=2,n-1
        do j=2,n-1
          v1(i,j)=0.25*(v(i+1,j)+v(i-1,j)+v(i,j+1)+v(i,j-1))
        enddo
      enddo
      return
      end
```

Figure 3. **Unplated potential plot for motion along the x-axis.**

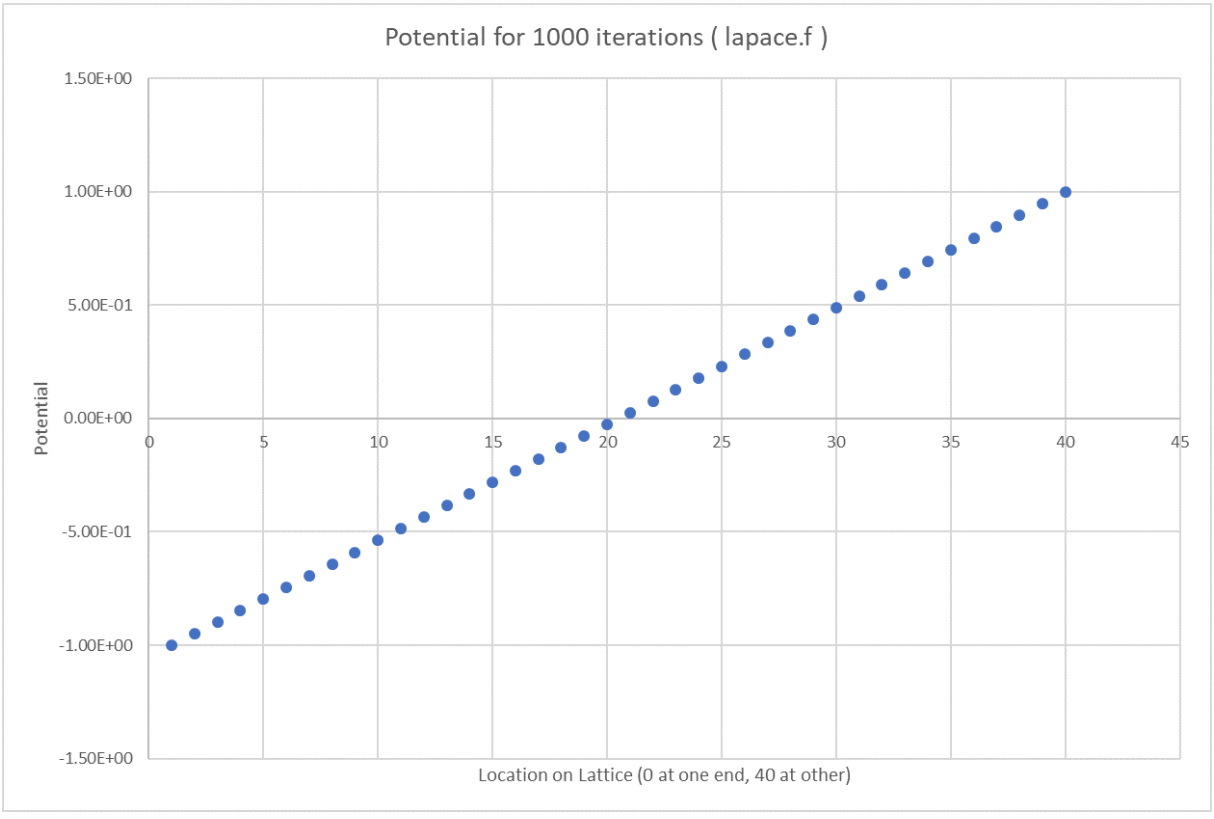

Figure 4. **Unplated potential 3D plot. 1000 iterations.**

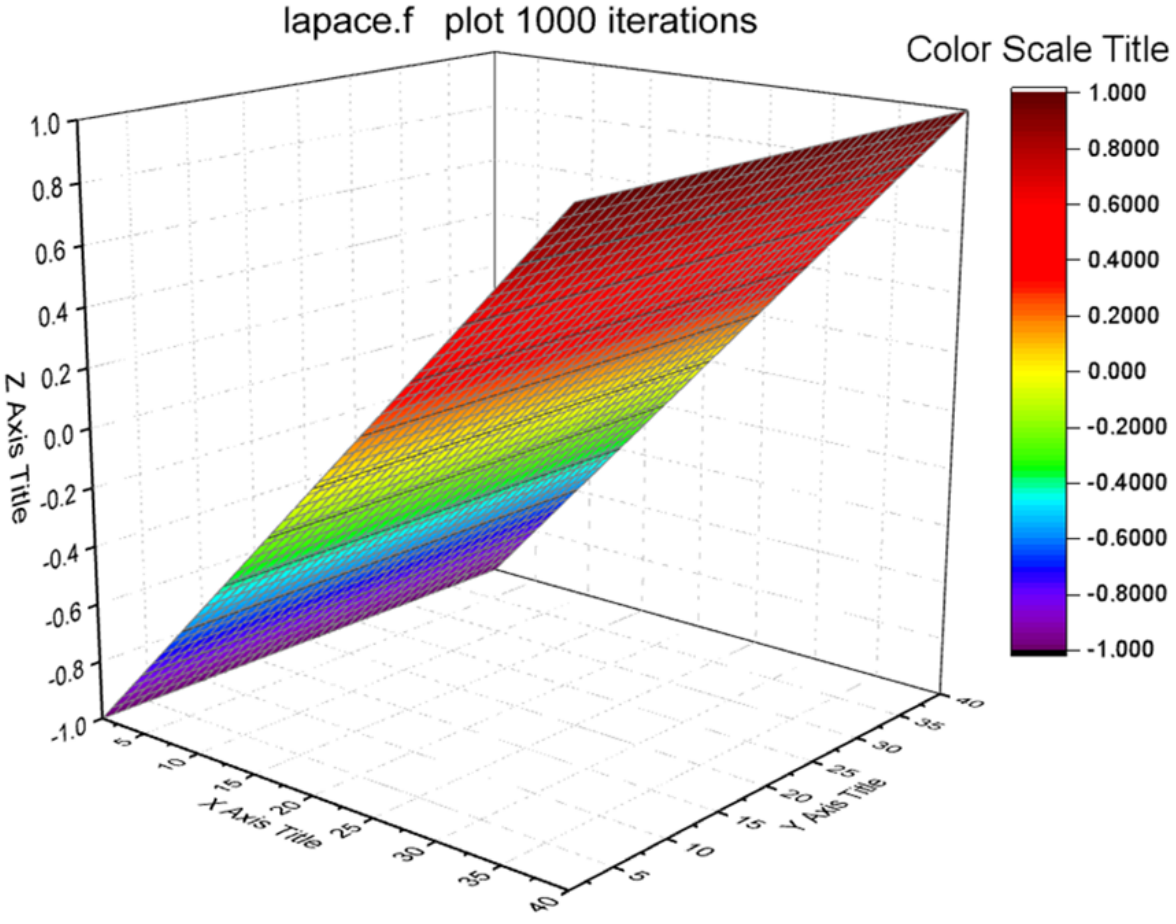

Figure 5. Plated potential plot for motion along the x-axis. Series #j corresponds to the value of j we begin at. That is, the plot for Series #j corresponds to the potential line going through all points with y coordinate j. This plot is included since it allows us to see what's happening in the 3D version (Fig. 6) more explicitly.

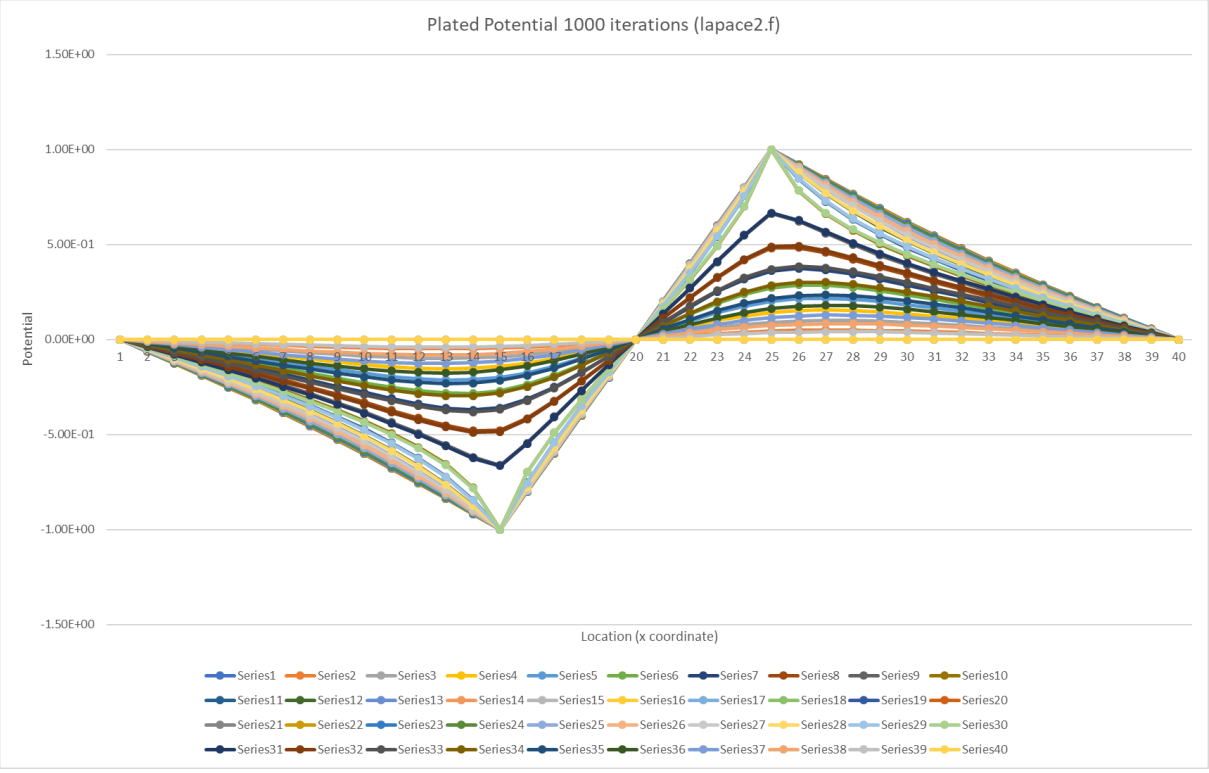

Figure 6. **Plated potential 3D plot. 1000 iterations.**

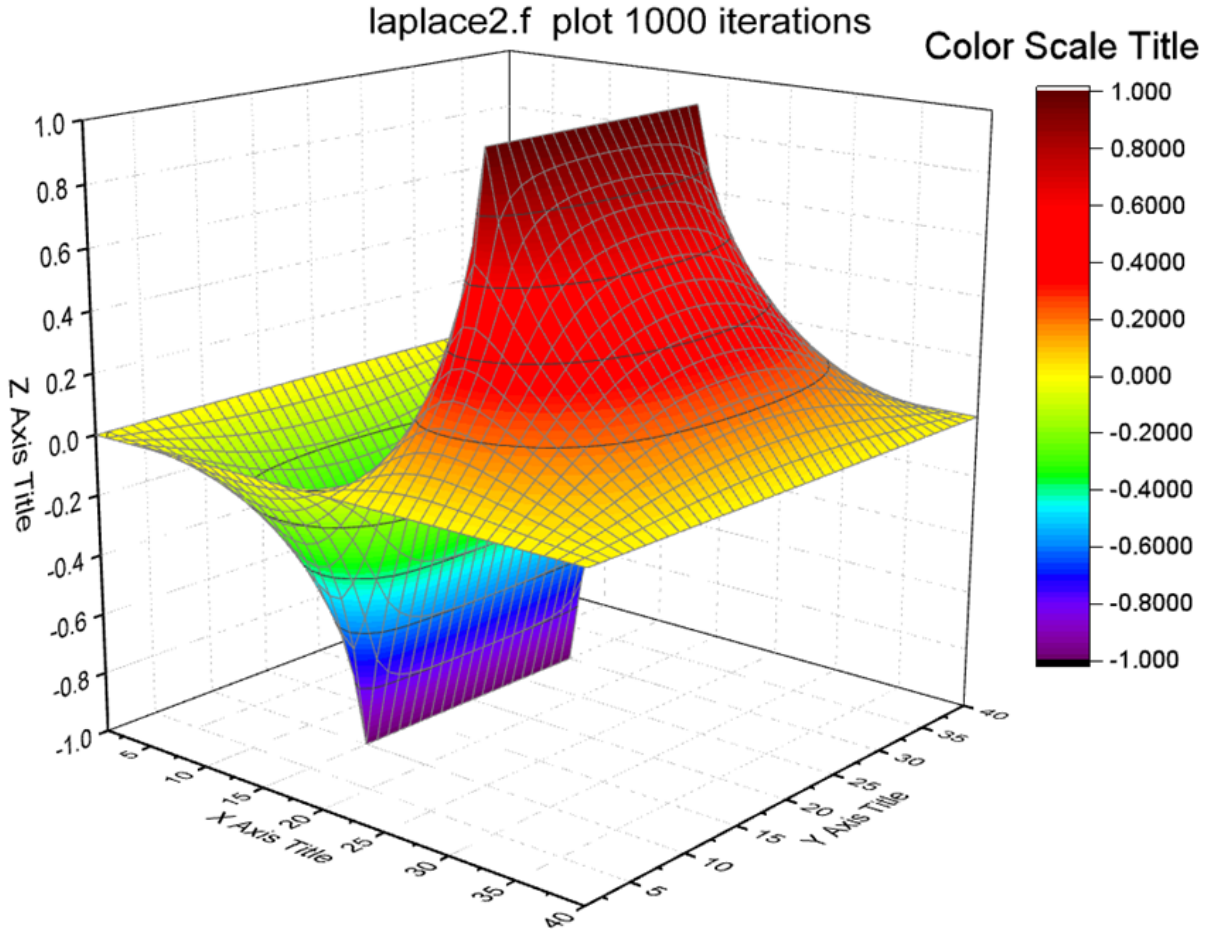

Figure 7. **Unplated potential 3D plot. 500 iterations.**

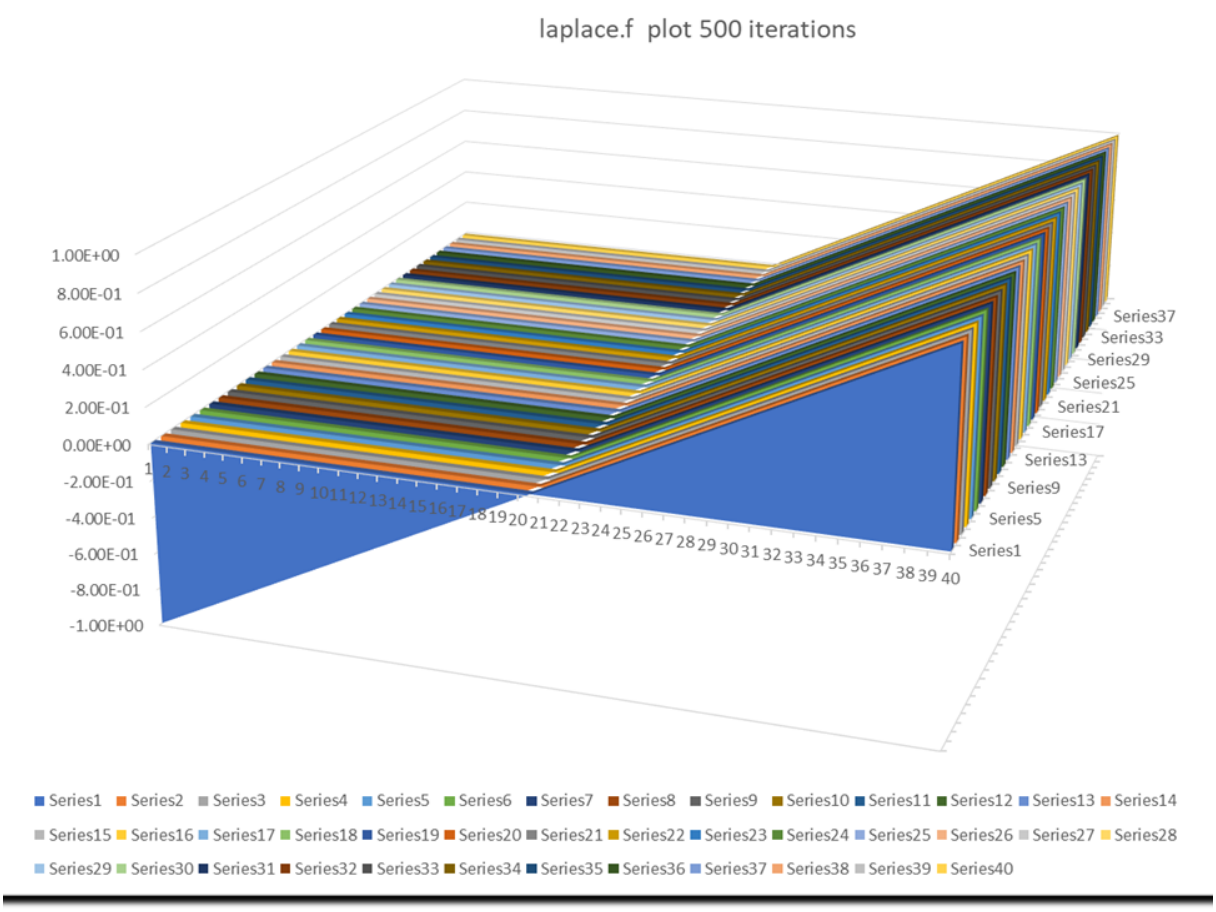

Figure 8. **Plated potential 3D plot. 500 iterations.**