# Lecture 36

# Computational Electromagnetics, Numerical Methods

Due to the advent of digital computers and the blinding speed at which computations can be done, numerical methods to seek solutions of Maxwell's equations have become vastly popular. Massively parallel digital computers now can compute at breakneck speed of tera\peta\exa-flops throughputs [219], where FLOPS stands for "floating operations per second". They have also spawn terms that we have not previously heard of (see also Figure 36.1).

**Computer performance**

| Name | Unit | Value |
|---|---|---|
| kiloFLOPS | kFLOPS | $10^3$ |
| megaFLOPS | MFLOPS | $10^6$ |
| gigaFLOPS | GFLOPS | $10^9$ |
| teraFLOPS | TFLOPS | $10^{12}$ |
| petaFLOPS | PFLOPS | $10^{15}$ |
| exaFLOPS | EFLOPS | $10^{18}$ |
| zettaFLOPS | ZFLOPS | $10^{21}$ |
| yottaFLOPS | YFLOPS | $10^{24}$ |

Figure 36.1: Nomenclature for measuring the speed of modern day computers (courtesy of Wikipedia [219]).

We repeat a quote from Freeman Dyson—"Technology is a gift of God. After the gift of life it is perhaps the greatest of God's gifts. It is the mother of civilizations, of arts and of sciences." The spurr for computer advancement is due to the second world war. During then, men went to war while women stayed back to work as computers, doing laborious numerical computations manually (see Figure 36.2 [220]): The need for a faster computer is obvious. Unfortunately, in the last half century or so, we have been using a large part of the gift of technology to destroy God's greatest gift, life, in warfare!

Figure 36.2: A woman working as a computer shortly after the second world war (courtesy of Wikipedia [220]).

## 36.1 Computational Electromagnetics, Numerical Methods

Due to the high fidelity of Maxwell's equations in describing electromagnetic physics in nature, and they have been validated to high accuracy (see Section 1.1), often time, a numerical solution obtained by solving Maxwell's equations is more reliable than laboratory experiments. This field is also known as ***computational electromagnetics***. Numerical methods exploit the blinding speed of modern digital computers to perform calculations, and hence to solve large system of equations.

Computational electromagnetics consists mainly of two classes of numerical solvers: one that solves differential equations directly: the differential-equation solvers; and one that solves integral equations: the integral equation solvers. Both these classes of equations are derived from Maxwell's equations.

## 36.2 Examples of Differential Equations

An example of differential equations written in terms of sources are the scalar wave equation:

$$(\nabla^2 + k^2(\mathbf{r}))\,\phi(\mathbf{r}) = Q(\mathbf{r}), \tag{36.2.1}$$

An example of vector differential equation for vector electromagnetic field is

$$\nabla \times \overline{\boldsymbol{\mu}}^{-1} \cdot \nabla \times \mathbf{E}(\mathbf{r}) - \omega^2 \overline{\boldsymbol{\epsilon}}(\mathbf{r}) \cdot \mathbf{E}(\mathbf{r}) = i\omega \mathbf{J}(\mathbf{r}) \tag{36.2.2}$$

These equations are linear equations, but for inhomogeneous media where $k^2(\mathbf{r})$ and $\epsilon(\mathbf{r})$ are functions of position vector $\mathbf{r}$, generally, they do not have closed form solutions. They have one commonality, i.e., they can be abstractly written as

$$\mathscr{L} f = g \tag{36.2.3}$$

where $\mathscr{L}$ is the differential operator which is linear, and $f$ is the unknown, and $g$ is the driving source. Differential equations, or partial differential equations, as mentioned before, have to be solved with boundary conditions. Otherwise, there is no unique solution to these equations.

In the case of the scalar wave equation (36.2.1), $\mathscr{L} = (\nabla^2 + k^2)$ is a differential operator. In the case of the electromagnetic vector wave equation (36.2.2), $\mathscr{L} = (\nabla \times \overline{\boldsymbol{\mu}}^{-1} \cdot \nabla \times) - \omega^2 \overline{\boldsymbol{\epsilon}} \cdot$. Furthermore, $f$ will be $\phi(\mathbf{r})$ for the scalar wave equation (36.2.1), while it will be $\mathbf{E}(\mathbf{r})$ in the case of vector wave equation for an electromagnetic system (36.2.2). The $g$ on the right-hand side can represent $Q$ in (36.2.1) or $i\omega\mathbf{J}(\mathbf{r})$ in (36.2.2).

## 36.3 Examples of Integral Equations

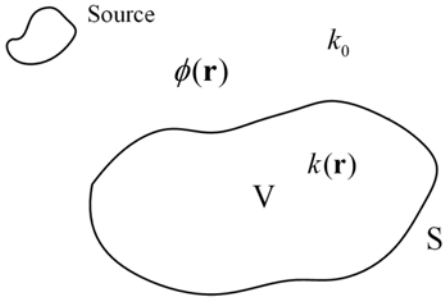


Figure 36.3: Geometry for the derivation of the volume-integral equation for scalar waves. The wavenumber $k(\mathbf{r})$ is assumed to be inhomogeneous and hence, a function of position $\mathbf{r}$ inside the scatterer, but a constant $k_0$ outside the scatterer.

### 36.3.1 Volume Integral Equation

This course is replete with PDE's, but we have not come across too many integral equations as yet. In integral equations, the unknown is embedded in the integral. The simplest integral equation to derive is the volume integral equation. Hence, we shall first derive the volume integral equation for the scalar wave case. In this case, the pertinent scalar wave equation is

$$[\nabla^2 + k^2(\mathbf{r})]\phi(\mathbf{r}) = Q(\mathbf{r}), \tag{36.3.1}$$

where $k^2(\mathbf{r})$ represents an inhomogeneous medium over a finite domain $V$, and $k^2 = k_0^2$, which is constant outside $V$ (see Figure 36.3). Next, we define a Green's function satisfying

$$[\nabla^2 + k_0^2]g(\mathbf{r}, \mathbf{r}') = -\delta(\mathbf{r} - \mathbf{r}'). \tag{36.3.2}$$

Then, (36.3.1) can be rewritten as

$$[\nabla^2 + k_0^2]\phi(\mathbf{r}) = Q(\mathbf{r}) - [k^2(\mathbf{r}) - k_0^2]\phi(\mathbf{r}). \tag{36.3.3}$$

Note that the right-hand side of (36.3.3) can be considered an equivalent source. Since the Green's function corresponding to the differential operator on the left-hand side of (36.3.3) is known, by the principle of linear superposition, we can write the formal solution to (36.3.3) as

$$\phi(\mathbf{r}) = -\int\limits_{V_s} dV' g(\mathbf{r}, \mathbf{r}')Q(\mathbf{r}') + \int\limits_{V} dV' g(\mathbf{r}, \mathbf{r}')[k^2(\mathbf{r}') - k_0^2]\phi(\mathbf{r}'). \tag{36.3.4}$$

The first term on the right-hand side is just the field due to the source in the absence of the inhomogeneity or the scatterer, and hence, is the incident field. The second term is a

volume integral over the space where $k^2(\mathbf{r}') - k_0^2 \neq 0$, or inside the inhomogeneous scatterer. Therefore, (36.3.4) becomes

$$\phi(\mathbf{r}) = \phi_{inc}(\mathbf{r}) + \int_V dV' g(\mathbf{r}, \mathbf{r}')[k^2(\mathbf{r}') - k_0^2]\phi(\mathbf{r}'). \tag{36.3.5}$$

In the above equation, if the total field $\phi(\mathbf{r}')$ inside the volume $V$ is known, then $\phi(\mathbf{r})$ can be calculated everywhere. But $\phi(\mathbf{r})$ is unknown at this point. To solve for $\phi(\mathbf{r})$, an integral equation has to be formulated for $\phi(\mathbf{r})$. To this end, we imposed (36.3.5) for $\mathbf{r}$ in $V$. Then, $\phi(\mathbf{r})$ on the left-hand side and on the right-hand side are the same unknown defined over the same domain. Consequently, (36.3.5) becomes the desired integral equation after rearrangement as

$$\phi_{inc}(\mathbf{r}) = \phi(\mathbf{r}) - \int_V dV' g(\mathbf{r}, \mathbf{r}')[k^2(\mathbf{r}') - k_b^2]\phi(\mathbf{r}'), \quad \mathbf{r} \in V. \tag{36.3.6}$$

In the above, the unknown $\phi(\mathbf{r})$ is defined over a volume $V$, over which the integration is performed, and hence the name, volume integral equation. Alternatively, the above can be rewritten as

$$\phi_{inc}(\mathbf{r}) = \phi(\mathbf{r}) - \mathcal{G}(\mathbf{r}, \mathbf{r}')\phi(\mathbf{r}'), \quad \mathbf{r} \in V, \tag{36.3.7}$$

where $\mathcal{G}$ is the integral operator in (36.3.6).[1] It is also a **Fredholm integral equation** of the second kind because the unknown is both inside and outside the integral operator. In the above, integration over repeated variable $\mathbf{r}'$ is implied. Nevertheless, it can be written as

$$\mathscr{L} f = g \tag{36.3.8}$$

where $\mathscr{L}$ is a linear operator, while $f$ represents the unknown function $\phi(\mathbf{r})$ and $g$ is the known function $\phi_{inc}(\mathbf{r})$.

[1] Sometimes, this is called the kernel of the integral equation.

### 36.3.2 Surface Integral Equation

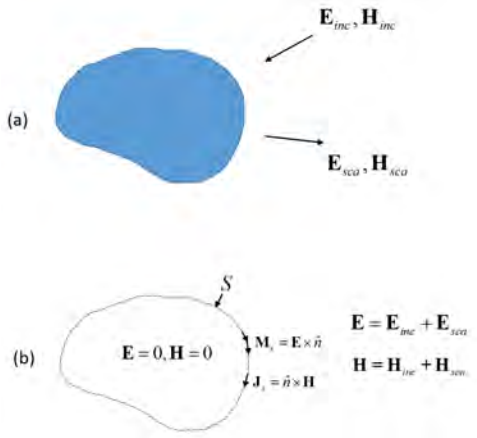


Figure 36.4: Geometry for the derivation of the surface-integral equation for vector electromagnetics waves. (a) The original electromagnetics scattering problem. (b) The equivalent electromagnetics problem by invoking equivalence principle.

The surface integral equation method is rather popular in a number of applications, because it employs a homogeneous-medium Green's function which is simple in form, and the unknowns reside on a surface rather than in a volume.[2]

The surface integral equation for vector electromagnetic field can be derived using the equivalence theorem also called the Love's equivalence theorem [182]. Given a scattering problem shown in Figure 36.4(a), it can be replaced by an equivalence problem as shown in Figure 36.4(b). One can verify this by performing a Gedanken experiment as we have done for the other equivalence problems discussed in Section 31.1.

In this figure, the total field outside the scatterer, $\mathbf{E} = \mathbf{E}_{inc} + \mathbf{E}_{sca}$ and $\mathbf{H} = \mathbf{H}_{inc} + \mathbf{H}_{sca}$. The impressed equivalence currents are given by $\mathbf{M}_s = \mathbf{E} \times \hat{n}$, and $\mathbf{J}_s = \hat{n} \times \mathbf{H}$. These impressed currents, together generate the scattered fields outside the scatterer, while they generate zero field inside the scatterer! One can verify that this is the case by performing Gedanken experiments.

As such, the scattered fields outside the scatterer can be found from the radiation of the impressed currents $\mathbf{M}_s$ and $\mathbf{J}_s$. Notice that these currents are radiating via the free-space Green's function because the scatterer has been removed in this equivalence problem. Now that if the scatterer is a PEC, then the tangential component of the total electric field is zero on the PEC surface. Therefore, $\mathbf{M}_s = 0$ and only $\mathbf{J}_s$ is radiating via the free-space Green's function.

Note that this equivalence problem is very different from that of an impressed currents on the PEC scatterer as discussed in Section 31.2. There, only the magnetic surface current

[2]These are sometimes called boundary integral equations method [221, 222].

is radiating in the presence of the PEC, and the Green's function is that of a current source radiating in the presence of the PEC scatterer, and it is not the free-space Green's function.

Now we can write the fields outside the scatterer using (31.4.17)

$$\mathbf{E}_{sca}(\mathbf{r}) = \frac{1}{i\omega\epsilon}\nabla\times\nabla\times\oint_S dS'\, g(\mathbf{r}-\mathbf{r}')\,\hat{n}'\times\mathbf{H}(\mathbf{r}') = \frac{1}{i\omega\epsilon}\nabla\times\nabla\times\oint_S dS'\, g(\mathbf{r}-\mathbf{r}')\,\mathbf{J}_s(\mathbf{r}') \quad (36.3.9)$$

In the above, we have swapped $\mathbf{r}'$ and $\mathbf{r}$ compared to (31.4.17). Also, we have kept only the electric current $\mathbf{J}_s(\mathbf{r})$ due to $\hat{n}\times\mathbf{H}(\mathbf{r})$. If we impose the boundary condition that the tangential component of the total electric field is zero, then we arrive at $\hat{n}\times\mathbf{E}_{sca} = -\hat{n}\times\mathbf{E}_{inc}$

$$-\hat{n}\times\mathbf{E}_{inc}(\mathbf{r}) = \hat{n}\times\frac{1}{i\omega\epsilon}\nabla\times\nabla\times\oint_S dS'\, g(\mathbf{r}-\mathbf{r}')\,\mathbf{J}_s(\mathbf{r}'). \qquad \mathbf{r}\in S \quad (36.3.10)$$

In the above, $\hat{n}\times\mathbf{E}_{inc}(\mathbf{r})$ is known on the left hand side on the scatterer's surface, while the right-hand side has embedded in it the unknown surface current $\mathbf{J}_s(\mathbf{r}) = \hat{n}\times\mathbf{H}(\mathbf{r})$ on the surface the scatter. Therefore, the above is an integral equation for the unknown surface current $\mathbf{J}_s$. It can be written as a form of $\mathscr{L}f = g$ just like other linear operator equations.

## 36.4 Function as a Vector

Several linear operator equations have been derived in the previous sections. They are all of the form

$$\mathscr{L}f = g \quad (36.4.1)$$

In the above, $f$ is a functional vector which is the analogy of the vector $\mathbf{f}$ in matrix theory or linear algebra. In linear algebra, the vector $\mathbf{f}$ is of length $N$ in an $N$ dimensional space. It can be indexed by a set of countable index, say $i$, and we can described such a vector with $N$ numbers such as $f_i, i = 1,\ldots,N$ explicitly. This is shown in Figure 36.5(a).

A function $f(x)$, however, can be thought of as being indexed by $x$ in the 1D case. However, the index in this case is a continuum, and countably infinite. Hence, it corresponds to a vector of infinite dimension and it lives in an infinite dimensional space.[3]

To make such functions economical in storage, for instance, we replace the function $f(x)$ by its sampled values at $N$ locations, such that $f(x_i), \quad i = 1,\ldots,N$. Then the values of the function in between the stored points $f(x_i)$ can be obtained by interpolation.[4] Therefore, a function vector $f(x)$, even though it is infinite dimensional, can be approximated by a finite length vector, $\mathbf{f}$. This concept is illustrated in Figure 36.5(b) and (c). This concept can be generalized to a function of 3D space $f(\mathbf{r})$. If $\mathbf{r}$ is sampled over a 3D volume, it can provide an index to a vector $f_i = f(\mathbf{r}_i)$, and hence, $f(\mathbf{r})$ can be thought of as a vector as well.

[3]When these functions are square integrable implying finite "energy", these infinite dimensional spaces are called Hilbert spaces.

[4]This is in fact how special functions like $\sin(x)$, $\cos(x)$, $\exp(x)$, $J_n(x)$, $N_n(x)$, etc, are computed and stored in modern computers.

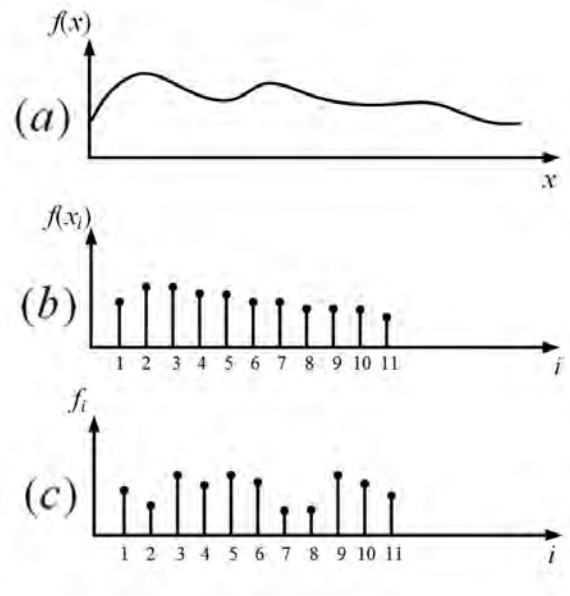


Figure 36.5: A function can be thought of as a vector. (a) A continuum function $f(x)$ plotted as a function of $x$. (b) A digitally sampled values of the same function. (c) When stored in a computer, it will be stored as an array vector.

## 36.5 Operator as a Map

### 36.5.1 Domain and Range Spaces

An operator like $\mathscr{L}$ above can be thought of as a map or a transformation. It maps a function $f$ defined in a Hilbert space $V$ to $g$ defined in another Hilbert space $W$. Mathematically, this is written as

$$\mathscr{L} : V \to W \tag{36.5.1}$$

indicating that $\mathscr{L}$ is a map of vectors in the space $V$ to vectors in the space $W$. Here, $V$ is also called the ***domain space*** (or domain) of $\mathscr{L}$ while $W$ is the ***range space*** (or range) of $\mathscr{L}$.

## 36.6 Approximating Operator Equations with Matrix Equations

### 36.6.1 Subspace Projection Methods

One main task of numerical method is first to approximate an operator equation $\mathscr{L}f = g$ by a matrix equation $\overline{\mathbf{L}} \cdot \mathbf{f} = \mathbf{g}$. To achieve the above, we first let

$$f \cong \sum_{n=1}^{N} a_n f_n = g \tag{36.6.1}$$

In the above, $f_n, n, \ldots, N$ are known functions called basis functions. Now, $a_n$'s are the new unknowns to be sought. Also the above is an approximation, and the accuracy of the approximation depends very much on the original function $f$. A set of very popular basis functions are functions that form a piece-wise linear interpolation of the function from its nodes. These basis functions are shown in Figure 36.6.

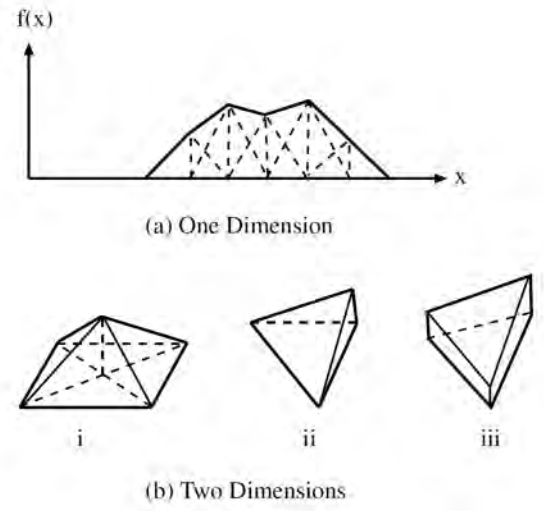


Figure 36.6: Examples of basis function in (a) one dimension, (b) two dimension. Each of these functions are define over a finite domain. Hence, they are also called sub-domain basis functions. They can be thought of as interpolatory functions.

Upon substituting (36.6.1) into (36.4.1), we obtain

$$\sum_{n=1}^{N} a_n \mathscr{L} f_n = g \tag{36.6.2}$$

Then, upon multiplying (36.6.2) by $w_m$ and integrating over the space that $w_m(\mathbf{r})$ is defined,

then we have

$$\sum_{n=1}^{N} a_n \langle w_m, \mathscr{L} f_n \rangle = \langle w_m, g \rangle, m = 1, \ldots, N \tag{36.6.3}$$

In the above, the inner product is defined as

$$\langle f_1, f_2 \rangle = \int d\mathbf{r} f_1(\mathbf{r}) f_2(\mathbf{r}) \tag{36.6.4}$$

where the integration is over the support of the functions, or the space over which the functions are defined.[5] For PDEs these functions are defined over a 3D coordinate space, while in SIEs, these functions are defined over a surface. In a 1D problems, these functions are defined over a 1D coordinate space.

### 36.6.2 Dual Spaces

The functions $w_m, m = 1, \ldots, N$ is known as the weighting functions or testing functions. The testing functions should be chosen so that they can approximate well a function that lives in the range space $W$ of the operator $\mathscr{L}$. Such set of testing functions lives in the ***dual space*** of the range space. For example, if $f_r$ lives in the range space of the operator $\mathscr{L}$, the set of function $f_d$, such that the inner product $\langle f_d, f_r \rangle$ exists, forms the dual space of $W$.

### 36.6.3 Matrix and Vector Representations

The above equation (36.6.3) is a matrix equation of the form

$$\overline{\mathbf{L}} \cdot \mathbf{a} = \mathbf{g} \tag{36.6.5}$$

where

$$\begin{aligned} \left[\overline{\mathbf{L}}\right]_{mn} &= \langle w_m, \mathscr{L} f_n \rangle \\ [\mathbf{a}]_n = a_n, [\mathbf{g}]_m &= \langle w_m, g \rangle \end{aligned} \tag{36.6.6}$$

What has effectively happened here is that given an operator $\mathscr{L}$ that maps a function that lives in an infinite dimensional Hilbert space $V$, to another function that lives in another infinite dimensional Hilbert space $W$, via the operator equation $\mathscr{L} f = g$, we have approximated the Hilbert spaces with finite dimensional spaces (subspaces), and finally, obtain a finite dimensional matrix equation that is the representation of the original infinite dimensional operator equation. This is the spirit of the subspace projection method.

In the above, $\overline{\mathbf{L}}$ is the matrix representation of the operator $\mathscr{L}$ in the subspaces, and $\mathbf{a}$ and $\mathbf{g}$ are the vector representations of $f$ and $g$, respectively, in their respective subspaces.

When such a method is applied to integral equations, it is usually called the method of moments (MOM). (Surface integral equations are also called boundary integral equations (BIEs) in other fields [222].) When finite discrete basis are used to represent the surface unknowns, it is also called the boundary element method (BEM) [223]. But when this method is applied to solve PDEs, it is called the finite element method (FEM) [224–227], which is a rather popular method due to its simplicity.

[5] This is known as the reaction inner product [35, 50, 131]. As oppose to most math and physics literature, the energy inner product is used [131] where $\langle f_1, f_2 \rangle = \int d\mathbf{r} f_1^*(\mathbf{r}) f_2(\mathbf{r})$.

### 36.6.4 Mesh Generation

In order to approximate a function defined on an arbitrary shaped surface or volume by a finite sum of basis functions, it is best to mesh (tessellate or discretize) the surface and volume by meshes. In 2D, all shapes can be tessellated by unions of triangles, while a 3D volume can be meshed (tessellated) by unions of tetrahedrons. Such meshes are used not only in CEM, but in other fields such as solid mechanics. Hence, there are many "solid modeling" commercial software available to generate sophisticated meshes.

When a surface is curved, or of arbitrary shape, it can be meshed by union of triangles as shown in Figure 36.7. When a volume is of arbitrary shape or a volume is around an arbitrary shape object, it can be meshed by tetrahedrons as shown in Figure 36.8. Then basis functions as used in (36.6.1) are defined to interpolate the field between nodal values or values defined on the edges of a triangle or a tetrahedron.

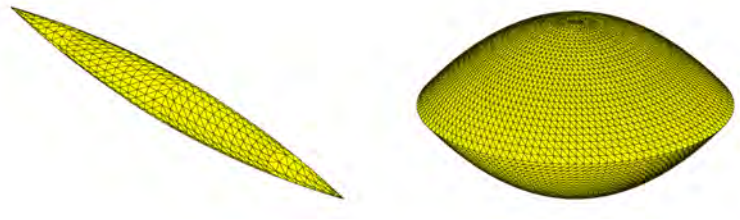

Figure 36.7: An arbitrary surface can be meshed by a union of triangles.

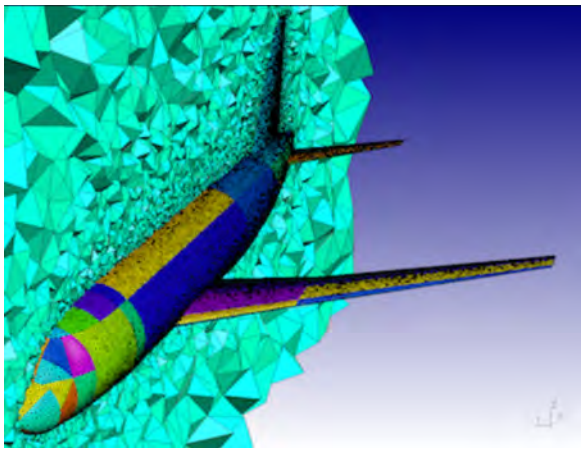

Figure 36.8: A volume region can be meshed by a union of tetrahedra. But the surface of the aircraft is meshed with a union of trianlges (courtesy of gmsh.info).

### 36.6.5 Differential Equation Solvers versus Integral Equation Solvers

As have been shown, the two classes of numerical solvers for Maxwell's equations consist of differential equation solvers and integral equation solvers. Differential equation solvers are generally easier to implement. As shall be shown in the next lecture, they can also be easily implemented using finite difference solver. The unknowns in a differential equation solver are the fields. The fields permeate all of space, and hence, the unknowns are volumetrically distributed. When the fields are digitized by representing them by their point values in space, they require a large number of unknowns to represent. The plus side is that the matrix system associated with a differential equation solver is usually sparse, requiring less storage and less time to solve.

As has been shown, integral equation solvers are formulated using Green's functions. That is integral equations are derived from Maxwell's equations using Green's function, where the unknowns now are surface sources such as surface electric and magnetic currents. Therefore, the unknowns are generally smaller, living only on the surface of a scatterer (or they occupy a smaller part of space). Hence, they can be approximated by a smaller set of unknowns. Thus, the matrix systems generally are smaller. Once the currents are found, then the fields they generate can also be computed.

Since the derivation of integral equations requires the use of Green's functions, they are in general singular when $\mathbf{r} = \mathbf{r}'$, or when the observation point (observation point) $\mathbf{r}$ and the source point $\mathbf{r}'$ coincide. Care has to be taken to discretize the integral equations. Moreover, a Green's function connects every current source point on the surface of a scatterer with every other source points yielding a dense matrix system. But fast methods have been developed to solve such dense matrix systems [9].

## 36.7 Solving Matrix Equation by Optimization

Given a matrix equation, there are many ways to seek its solution. The simplest way is to find the inverse of the matrix operator by direct inversions (e.g., using Gaussian elimination [228] or Kramer's rule [229]) have computational complexity[6] of $O(N^3)$, and requiring storage of $O(N^2)$. Due to the poor computational and memory complexity of direct inversion, when $N$ is large, other methods have to be sought.

To this end, it is better to convert the solving of a matrix equation into an optimization problem. These methods can be designed so that a much larger system can be solved with an existing resource of a digital computer. Optimization problem results in finding the stationary point of a functional.[7] First, we will figure out how to find such a functional.

Consider a matrix equation given by

$$\overline{\mathbf{L}} \cdot \mathbf{f} = \mathbf{g} \tag{36.7.1}$$

[6]The scaling of computer time with respect to the number of unknowns (degrees of freedom) is known in the computer parlance as computational complexity.

[7]Functional is usually defined as a function of a function [35, 49]. Here, we include a function of a vector to be a functional as well.

For simplicity, we consider $\overline{\mathbf{L}}$ as a symmetric matrix.[8] Then the corresponding functional is

$$I = \mathbf{f}^t \cdot \overline{\mathbf{L}} \cdot \mathbf{f} - 2\mathbf{f}^t \cdot \mathbf{g} \tag{36.7.2}$$

Such a functional is called a quadratic functional because it is analogous to $I = Lx^2 - 2xg$, which is quadratic, in its simplest 1D rendition.

Taking the first variation with respect to $\mathbf{f}$, namely, we let $\mathbf{f} = \mathbf{f}_o + \delta\mathbf{f}$. Then we substitute this into the above, and collect the leading order and first order terms. Then we find the first order approximation of the functional $I$ as

$$\delta I = \delta\mathbf{f}^t \cdot \overline{\mathbf{L}} \cdot \mathbf{f}_o + \mathbf{f}_o^t \cdot \overline{\mathbf{L}} \cdot \delta\mathbf{f} - 2\delta\mathbf{f}^t \cdot \mathbf{g} \tag{36.7.3}$$

If $\overline{\mathbf{L}}$ is symmetric, the first two terms are the same, and the above becomes

$$\delta I = 2\delta\mathbf{f}^t \cdot \overline{\mathbf{L}} \cdot \mathbf{f}_o - 2\delta\mathbf{f}^t \cdot \mathbf{g} \tag{36.7.4}$$

For $\mathbf{f}_o$ to be the optimal point or the stationary point, then its first variation has to be zero, or that $\delta I = 0$. Thus we conclude that at the optimal point (or the stationary point),

$$\overline{\mathbf{L}} \cdot \mathbf{f}_o = \mathbf{g} \tag{36.7.5}$$

Hence, the optimal point to the functional $I$ in (36.7.2) is the solution to (36.7.1) or (36.7.5).

### 36.7.1 Gradient of a Functional

The above method, when applied to an infinite dimensional Hilbert space problem, is called variational method, but the main ideas are similar. The wonderful idea about such a method is that instead of doing direct inversion of a matrix system (which is expensive), one can search for the optimal point or stationary point of the quadratic functional using gradient search or gradient descent methods or some optimization method.

It turns out that the gradient of a quadratic functional can be found quite easily. Also it is cheaper to compute the gradient of a functional than to find the inverse of a matrix operator. To do this, it is better to write out functional using index (or indicial, or Einstein) notation [230]. In this notation, the functional first variation $\delta I$ in (36.7.4) becomes

$$\delta I = 2\delta f_j L_{ij} f_i - 2\delta f_j g_j \tag{36.7.6}$$

Also, in this notation, the summation symbol is dropped, and summations over repeated indices are implied. In the above, we neglect to distinguish between $\mathbf{f}_o$ and $\mathbf{f}$. It is implied that $\mathbf{f}$ represents the optimal point. In this notation, it is easier to see what a functional derivative is. We can differentiate the above with respect to $f_j$ easily to arrive at

$$\frac{\partial I}{\partial f_j} = 2L_{ij} f_i - 2g_j \tag{36.7.7}$$

[8] Functional for the asymmetric case can be found in *Chew, Waves and Fields in Inhomogeneous Media*, Chapter 5 [35].

Notice that the remaining equation has one index $j$ remaining in index notation, meaning that it is a vector equation. We can reconstitute the above using our more familiar matrix notation that

$$\frac{\delta I}{\delta \mathbf{f}} = \nabla_{\mathbf{f}} I = 2\overline{\mathbf{L}} \cdot \mathbf{f} - 2\mathbf{g} \tag{36.7.8}$$

The left-hand side is a notation for the functional derivative or the gradient of a functional in a multi-dimensional space whichi is a vector obviated by indicial notation. And the right-hand side is the expression for calculating this gradient. One needs only to perform a matrix-vector product to find this gradient. Hence, the computational complexity of finding this gradient is $O(N^2)$ at worst if $\overline{\mathbf{L}}$ is a dense matrix, and as low as $O(N)$ if $\overline{\mathbf{L}}$ is a sparse matrix.[9] In a gradient search method, such a gradient is calculated repeatedly until the optimal point is found. Such methods are called iterative methods.

If the optimal point can be found in $N_{\text{iter}}$ iterations, then the CPU time scales as $N_{iter}N^{\alpha}$ where $1 < \alpha < 2$. There is a clever gradient search algorithm, called the ***conjugate gradient method*** that can find the exact optimal point in $N_{\text{iter}} = N$ in exact arithmetics. But exact solution is not needed in an optimal solution: an approximate solution suffices. In many gradient search solutions, to obtain an approximate solution where the error is acceptable, $N_{\text{iter}} \ll N$. The total solution time or solve time which is $N_{iter}N^{\alpha} \ll NN^{\alpha} \ll N^3$, resulting in great savings in computer time, especially if $\alpha = 1$. This is the case for FEM [227], [231], [232], [233], [226], and fast multipole algorithm [234], [235].

What is more important is that this method does not require the storage of the matrix $\overline{\mathbf{L}}$, but a computer code that produces the vector $\mathbf{g}_o = \overline{\mathbf{L}} \cdot \mathbf{f}$ as an output, with $\mathbf{f}$ as an input. Both $\mathbf{f}$ and $\mathbf{g}_o$ require only $O(N)$ memory storage. Such methods are called matrix-free methods. Even when $\overline{\mathbf{L}}$ is a dense matrix, which is the case if it is the matrix representation of matrix representation of some Green's function, fast methods now exist to perform the dense matrix-vector product in $O(N \log N)$ operations.[10]

The value $I$ is also called the cost function, and its minimum is sought in the seeking of the solution by gradient search methods. Detail discussion of these methods is given in [236]. Figure 36.9 shows the contour plot of a cost function in 2D. When the condition number[11] of the matrix $\overline{\mathbf{L}}$ is large (implying that the matrix is ill-conditioned), the contour plot will resemble a deep valley. And hence, the gradient search method will tend to zig-zag along the way as it finds the optimal solution. Therefore, convergence is slow for matrices with large condition numbers

[9] This is the case for many differential equation solvers such as finite-element method or finite-difference method.

[10] Chew et al, *Fast and Efficient Algorithms in CEM* [9].

[11] This is the ratio of the largest eigenvalue of the matrix to its smallest eigenvalue.

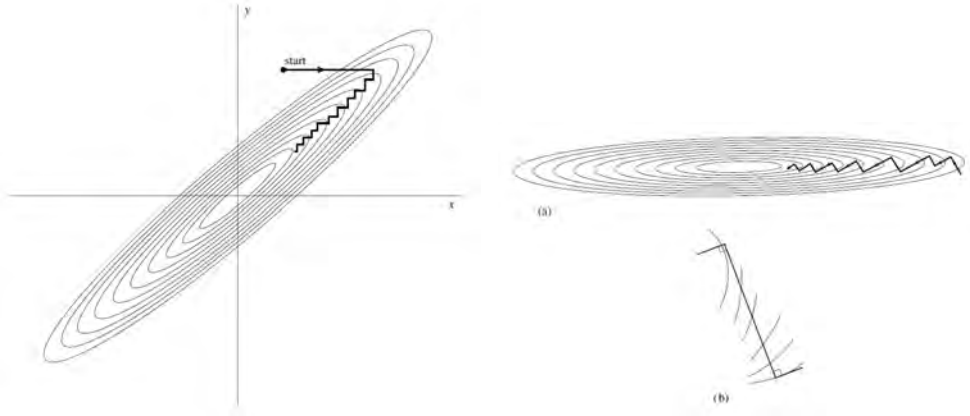


Figure 36.9: Plot of a 2D cost function, $I(x, y)$ for an ill-conditioned system (courtesy of Numerical Recipe [236]). A higher dimensional plot of this cost function will be difficult.

Figure 36.10 shows a cartoon picture in 2D of the histories of different search paths from a machine-learning example where a cost functional similar to $I$ has to be minimized. Finding the optimal point or the minimum point of a general functional is still a hot topic of research: it is important in artificial intelligence.

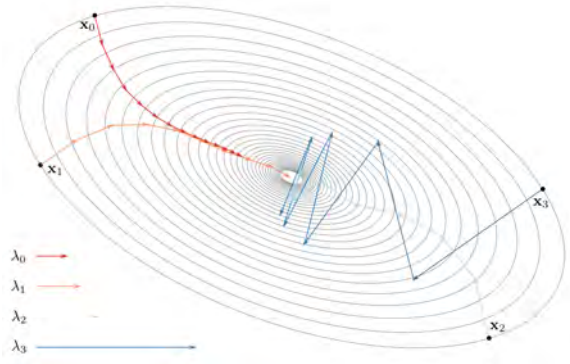


Figure 36.10: Gradient search or gradient descent method is finding an optimal point (courtesy of Y. Ioannou: https://blog.yani.io/sgd/).