T.C
BEYKENT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR ANABİLİM DALI
BİLGİ TEKNOLOJİLERİ BÖLÜMÜ

2D ELEKTROSTATİK LAPLACE DENKLEMİNİN SONLU ELEMANLAR YÖNTEMİYLE ÇÖZÜMÜ İÇİN BİR PROGRAMLAMA UYGULAMASI

(Yüksek Lisans Tezi)

Tezi Hazırlayan: Berna SÜLÜ

Tez Danışmanı : **Yrd. Doç. Dr. Turhan KARAGÜLER**

İstanbul 2009

İÇİNDEKİLER **Sayfa No**

EKLER

EK – A input.txt yapısı

EK – B C-Dilinde kodlanmış örnek program

1) Input.txt'yi oluşturan kod

2) Solver

EK – C MATLAB örnek program

Yemin Metni : Sunduğum Yüksek Lisans Projesi /Yüksek Lisans Tezimi, Akademik Etik İkelerine bağlı kalarak, hiç kimseden akademik ilkelere aykırı bir yardım almaksızın bizzat kendimin hazırladığına and içerim.

T.C.
BEYKENT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ MÜDÜRLÜĞÜ
TEZLİ YÜKSEK LİSANS TEZ SINAV TUTANAĞI

23/09/2009

Enstitümüz Matematik-Bilgisayar Anabilim Dalı Bilgi Teknolojileri Bilim Dalı yüksek lisans öğrencilerinden 050862016 numaralı **Berna SÜLÜ'** *ye "Beykent Üniversitesi Lisansüstü Eğitim - Öğretim Yönetmeliği*'nin ilgili maddesine göre hazırlayarak, Enstitümüze teslim ettiği ***"2D Elektrostatik Laplace Denkleminin Sonlu Elemanlar Yöntemiyle Çözümü İçin Bir Programlama Uygulaması"*** *tezini*, Yönetim Kurulumuzun 30.04.2009 tarih ve 2009/5 sayılı toplantısında seçilen ve Fakülte binasında toplanan biz jüri üyeleri huzurunda, ilgili yönetmeliğin *(c) bendi gereğince* aday tarafından savunulmuş ve sonuçta adayın tezi hakkında **oyçokluğu /** ~~***oybirliği***~~ ile ***Kabul / Red veya Düzeltme*** kararı verilmiştir.

İşbu tutanak Enstitü Müdürlüğü'ne sunulmak üzere tarafımızdan düzenlenmiştir.

DANIŞMAN
Yrd. Doç. Dr. Turhan KARAGÜLER

ÜYE
Prof. Dr. Mahir RESULOV

ÜYE
Prof. Dr. Esat HAMZAOĞLU

Özet

Sonlu Elemanlar Yöntemi; mühendislik problemlerinin sayısal yöntemlerle çözümünde en popüler olan yaklaşımlarından biridir. Özellikle doğrusal olmayan sınır ve yüzeyleri modellemede tercih edilir. Ancak bu yöntemin uygulanması ve programlanması göreceli olarak zordur. Çoğunlukla paket programlar sayesinde yöntemin problemlere uygulanması sağlanır. Burada en önemli programlamama yükü otomatik ağ(mesh) oluşturmada görülür. Sonlu elemanlar yöntemi, sık olmasa da, sadece ilgili probleme özgü olarak geliştirilen modellemelerde de kullanılır. Bunda son yıllarda hızla gelişen ve değişen programlama araçlarının sunduğu olanakların önemli rolü vardır.

Bu çalışmada 2D Laplace denklemi ile temsil edilen dielektrik bölgeli kapasitör problemi sonlu elemanlar yöntemi ile iki ayrı programlama ortamı kullanılarak modellenmiştir. Bu araçlar göreceli olarak yeni sayılabilecek Script dil örneği MATLAB ve klasik yapısal programlama dili örneği C Dili programlama ortamlarıdır. Sonlu elemanlar yöntemi öngörülen problem için diskritize edilmiş ve yöntemin diğer adımları sırasıyla uygulanarak potansiyel dağılım ve elektrik alan vektörleri elde edilmiştir. Programlama araçları gerek performans gerekse olanakları yönünden karşılaştırılarak sonlu elemanlar yöntemini kendi problemlerine uygulamayı düşünen araştırmacılara program geliştirme ortamı yönünde yol gösterme hedeflenmiştir.

Anahtar Kelimeler : Sonlu Elemanlar Yöntemi, Elektrostatik Laplace Denklemi, Programlama Teknikleri

Abstract

Finite Element is one of the most popular approaches for solving the problems of engineering and science by means of numerical methods. They are particularly significant for geometries having non-smooth surfaces and borders. However the application and coding of it is relatively complicated therefore mostly the method is applied to the problems by using software packages which provides the most difficult part of FE programming that is automatic mesh generation. On the other hand, instead of using mostly expensive and extensive packages, sometimes simple programs addressing specific problems of interest are preferred to employ for FE modelling as well. This has been made possible by the facilities and easiness provided by new generation programming environments.

In this work, a dielectric capacitor problem represented by a 2D Laplace equation is modelled and coded by using two different programming tools. These tools are an example of scripting languages, MATLAB, and structured programming C. In order to obtain the potential distribution and electric field vectors, first the problem is discritized accordingly with FE method and later conventional steps of the method are applied. The programming tools are compared from both performance and facilities point of views. The work is aimed at potential FE programmers to choose the best programming environment for their applications.

Keywords: Finite Element Method, Electrostatic Laplace Equation, Programming Techniques

Şekiller Listesi **Sayfa No**

BÖLÜM 1

1. GİRİŞ

Mühendislik ve Temel Bilimlerde karşılaşılan birçok problem adi (ordinary) ve/veya kısmi (partial) diferansiyel denklemlerle ifade edilirler. Bu denklemlerin çoğu zaman analitik çözümünü elde etmek oldukça zor hatta olanaksızdır. Ancak günümüzde bilgisayarlar yardımıyla bu tür problemler için sayısal çözümler elde edilebilmektedir. Bilgisayarların ilk döneminde önce sayısal yöntemler tanıtıldı. Sonlu Elemanlar Yöntemi (Finite Element Method-FE), Sonlu Farklar Yöntemi (Finite Difference Method) ile birlikte bu yöntemlerin en popüler örnekleridir. Daha sonra bu yöntemlere, özellikle Sonlu Elemanlar Yöntemine, dayalı Bilgisayar Destekli Tasarım (Computer Aided Design-CAD) yazılım paketleri geliştirildi. Tüm bu yazılımların en önemli işlevi, problem geometrisinin bilgisayara aktarılmış hali olarak tanımlanabilecek olan mesh (ağ) oluşumunun gerçeklenmesi ve bu meshi esas alan Çözücünün (Solver) otomatik olarak çalıştırılabilmesidir. Ancak bu tür bir paket programın hazırlanması, özellikle karmaşık grafik ara yüzleri dikkate alındığında, oldukça kapsamlı ve ileri bir programlama becerisi gerektirecektir. Bu nedenle, günümüz programlama ortamlarının sağladığı kolaylıklarda göz önünde tutularak, paket yazılımların yanı sıra, basit ve spesifik problemlere yönelik olarak çok daha hızlı geliştirilebilecek, prototip programlar da kullanılmaktadır.

Bu tez kapsamında bir örnek probleme yönelik sonlu elemanlar prototip program iki ayrı programla ortamı kullanılarak geliştirilmiştir. Programlama ortamlarının farklı seçilmesi ve karşılaştırılması ile özellikle sonlu elemanlar ile programlamada en uygun yazılım geliştirme ortamının seçilmesi hedeflenmiştir.

İkinci bölümde modelleme amacıyla seçilen Elektromanyetizmada kullanılan iki boyutlu (2D) düzlem kapasitör problemi tanıtılmış olup, problemi temsil eden Laplace denklemi Maxwell denklemlerinden elde edilmiştir. Üçüncü bölümde Laplace denklemimin çözümünde kullanılabilecek sayısal yöntemler kısaca tanıtılmış olup, bu yöntemlerden tezde modelleme amacıyla programlanan sonlu elemanlar yöntemi ayrıntılı olarak ele alınmıştır. Dördüncü bölümde programlama ortamı olarak seçilen Yapısal programlama (structured programming) örneği C-dili ve Script programlama örneği

MATLAB ile programlama uygulamaları karşılaştırmalı olarak verilmiştir. Program sonuçları ayrıca tartışılmıştır.

BÖLÜM 2

2. ÖRNEK PROBLEM, LAPLACE DENKLEMİ

Matematikte Laplace Denklemi, ilk defa Pierre – Simon Laplace tarafından ortaya çıkarıldığı için aynı isimle adlandırılmış bir kısmi diferansiyel denklemdir. Laplace Denklemi elektromanyetizma, astronomi, akışkanlar dinamiği gibi bilim alanlarında sıklıkla karşılaşılır ve çözümleri ile özellikle elektrik ve yer çekim potansiyeli ile akışkan potansiyelinin davranışı açıklanmaya çalışılır. Laplace denkleminin çözümlerinin genel teorisi aynı zamanda potansiyel teorisi olarak da bilinmektedir.

Laplace denklemi ile modellenen problemde, denklem sistemi oldukça küçük bir alana indirgenmiş olur. Problem bölgesini Laplace denklemi temsil eder. Laplace denklemi sadece problem alanını küçültmede değil ayrıca çok sayıda değişken ile uğraşmak yerine azaltılmış veri sayısı ile de çözüme kolaylık getirir. Denklem sistemi, sadece sınır düğümlerindeki bilinmeyenlerden oluşur. Buna karşılık, denklemin sayısal yöntemle dönüştürülmesi ile elde edilen sistem matrisi doludur (dense) ve simetrik değildir

Bu çalışmada elektrostatik alan için elde edilen Laplace denklemi örnek problem olarak seçilmiştir. Öncelikle Elektromanyetizma için Laplace denklemini elde edelim.

Bilindiği üzere elektromanyetizmanın tüm problemleri aşağıda verilen Maxwell denklemleri ile tam olarak ifade edilmektedir [1].

$$Curl\vec{E} = \frac{\partial \vec{B}}{\partial t} \tag{2.1}$$

$$Curl\vec{H} = \vec{J} + \frac{\partial \vec{D}}{\partial t} \tag{2.2}$$

$$div\vec{B} = 0 \tag{2.3}$$

$$div\vec{D} = \rho \tag{2.4}$$

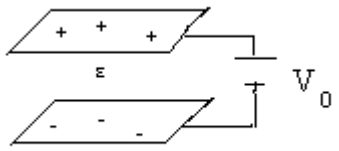
Bu denklemlerde, $\vec{B}[T]$: manyetik alan yoğunluğu, $\vec{H}[A/m]$: manyetik alan şiddeti, $\vec{E}[V/m]$: elektriksel alan, $\vec{D}[C/m^2]$: deplasman akı yoğunluğu, $\vec{J}[A/m^2]$: yüzeysel akım yoğunluğu, $\rho[C/m^3]$: hacimsel yük yoğunluğu olup aşağıdaki bünye denklemleri ile birbiriyle ilişkilidirler.

$$\vec{B} = \mu \vec{H} \tag{2.5}$$

$$\vec{D} = \varepsilon \vec{E} \tag{2.6}$$

$$\vec{J} = \sigma \vec{E} \tag{2.7}$$

Test problemi olarak aşağıda şekil 2.1 de verilen elektrostatik düzlemsel kapasitör seçildi. Bu seçimin nedeni, özellikle bu tez çalışması, konunun programlama ve sayısal yöntem boyutunu öne çıkarmayı hedeflemesinden, problemin basit ve iyi bilinen bir türe aitliğinin uygun bir tercih olmasıdır. Düzenekte iki eşdeğer düzlemsel iletken levha aralarına bir dielektrik malzeme konulmuş ve bir DC gerilim kaynağına bağlanmıştır. Problem ara bölgedeki elektrik alan ve potansiyel dağılımını bulma olarak tanımlanır.



Şekil 2.1 Düzlemsel Dielektrik Kapasitör

Öncelikle problem elektrostatik duruma ait olduğundan, zamana bağlılık olmayacak böylece elektriksel ve manyetik alanlar birbirinden bağımsızlaşacaktır. Bunun sonucu olarak karmaşık (2.1) ve (2.2) denklemleri yerine kısmen daha basit olan (2.3) ve (2.4) kullanılarak problemin istenen yönetici (governing) denklemi elde edilecektir. Bizim problemimiz elektrostatik durumla ilgili olduğundan doğaldır ki 2.4 denklemi başlangıç noktası olacak ve (2.6) bünye denklemi ile birlikte ele alınarak aşağıdaki denklem elde edilecektir.

$$div\, \varepsilon \vec{E} = \rho \tag{2.8}$$

Yukarıdaki denklemde ε ortamın dielektrik geçirgenliği olup $\varepsilon = \varepsilon_0 \varepsilon_r$ ile ifade edilir, ε_r ise ortamın dielektrik sabitidir ve doğrudan dielektrik malzemenin türünü belirler. Genelde elektromanyetizmada elektriksel ve manyetik alan vektörlerinin yerine potansiyel eşdeğerleri kullanılır. Pratikte genel olarak potansiyeller, denklemlerdeki alan büyüklükleri yerine tercih edilirler. Önce potansiyel ifadesini elde edip sonra alan büyüklünü bulma sıkça başvurulan bir yöntemdir. Bunda vektör büyüklük olan alan yerine çoğunlukla skaler büyüklük olan potansiyelin kullanılması bilinmeyen sayısını azaltacağından önemlidir. Ayrıca sayısal yöntemlerde sınır koşullarının uygulanması sırasında potansiyel ifadeleri süreklilik dikkate alındığında oldukça pratiklik ve kolaylık sağlayabilmektedirler. Elektrostatik durumda, aşağıda verildiği gibi, elektrik alanı bir potansiyelin gradyantı olarak ifade edebiliriz (zamanla değişen durumda ise, elektriksel alan korunan alan olmayacağından elektriksel alanın potansiyelin gradyantı olarak ifadesi mümkün değildir).

$$\vec{E} = -\,gradV \tag{2.9}$$

Elektriksel alan yerine elektrik potansiyelin V kullanılması ile birlikte (2.8) denklemi (2.10) denklemine dönüşür.

$$div\varepsilon\,(-\,gradV) = \rho \tag{2.10}$$

Yukarıdaki (2.10) denklemi Poisson denklemi olarak bilinir ve elektrostatiğin en genel durumunu temsil eder. Laplace denklemi ise Poisson denkleminin sağ tarafsız özel halini yansıtır ve pratikte ortamda serbest yük olmama durumuna denk düşer. Bizim örnek problemde modellenecek bölge iletkenler arası serbest yüksüz bölge olduğundan Laplace denklemi uygulanacak sayısal yöntem için asıl denklem olacaktır.

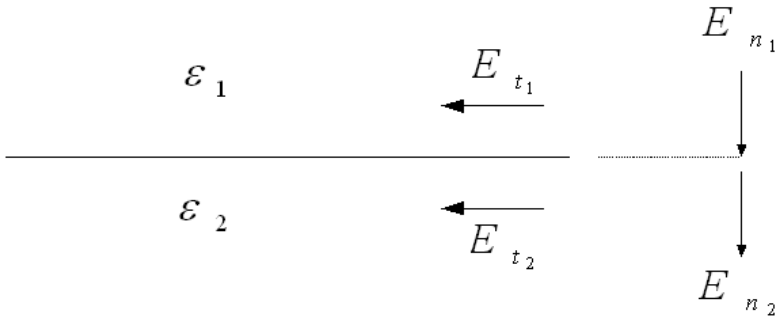
$$div\varepsilon\,(-\,gradV) = 0 \tag{2.11}$$

Bu arada problemin kaç boyutlu olacağının belirlenmesi gerekir. En genel halde zamanla değişim söz konusu değilse 3-boyutlu gösterim kullanılır. Ancak yine genel bir uygulama olarak probleme özgü bazı gerçekçi kabuller yapılarak problemin boyut sayısı düşürülebilir. Bu durum karmaşık geometrili problemlerde bilinmeyen sayısını azaltması

açısından önemlidir. Test problemde kapasitör iletken yüzey doğrultusunda elektrik alan için değişim olmaması kabulü yapılabileceğinden problemi gerçek 3-boyut yerine 2-boyutlu olarak modellemek olanaklıdır. Bu durumda diverjans ve gradyant işlemleri Kartezyen koordinat sisteminde potansiyel ifadesine uygulanırsa, Laplace denkleminin açık formu elde edilir.

$$\frac{\partial}{\partial x}\varepsilon\frac{\partial V}{\partial x}+\frac{\partial}{\partial y}\varepsilon\frac{\partial V}{\partial y}=0 \tag{2.12}$$

Yukarıdaki (2.12) denklemi çözümü programlanacak olan denklem sistemidir ve uygun sınır koşulları yardımıyla sayısal olarak çözülebilecektir. Sınır koşulları genel olarak elektrik alana ilişkin tanımlanır ve temel denklemdeki değişken üzerinden uygulanır.



Şekil 2.2) Sınır Koşullarının gösterimi

E_t : Teğetsel bileşen ve E_n normal bileşen olarak verilmiş olsun. Sınır koşulları iki bölgeyi ayıran arakesit üzerinde

$$E_{t_1} = E_{t_2} \tag{2.13}$$

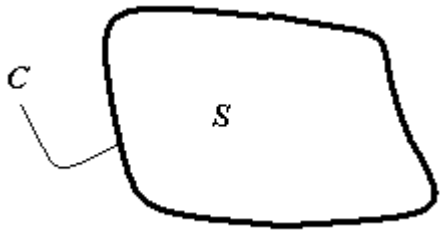
$$E_{n_1} - E_{n_2} = \rho \tag{2.14}$$

Elektrik alan yerine V elektrik potansiyeli kullanıldığında, serbest yükün olmadığı dikkate alınırsa, iç bölgede farklı sınır koşulları için herhangi bir ek işlem yapmaya gerek olmayacaktır.

BÖLÜM 3

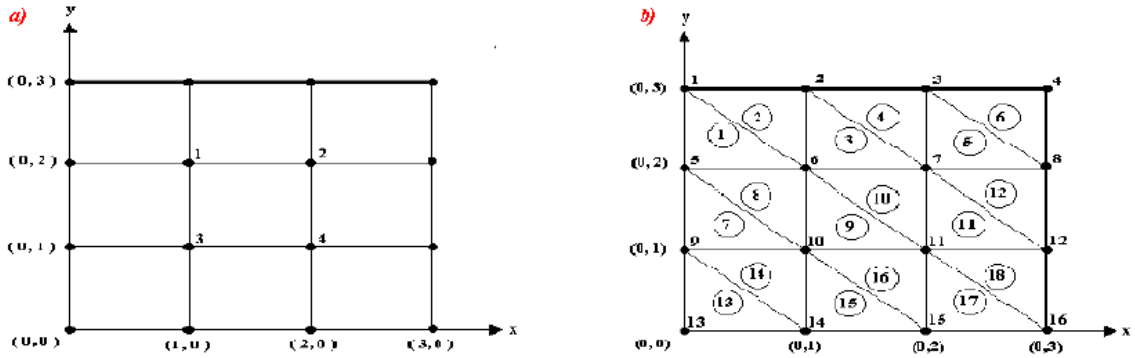
3. Sayısal Yöntemler

Basit geometriye sahip problemlerde, eğer ilave kabuller yapılarak 1 boyuta indirgenme sağlanabilirse, analitik çözüm yöntemleri kullanılarak problemin tam çözümü elde edilebilir. Ancak, tezimizin konusunu oluşturan (2.12) Laplace denklemi de dahil, 2 veya daha yüksek boyutlu problemlerde, sayısal çözüm yöntemleri kullanılarak önce cebirsel denklemlere dönüştürülmek yoluyla çözüm elde edilebilecektir. Sayısal çözüm yöntemlerindeki temel yaklaşım, alan denklemlerinin veya bir eşdeğer integral formülasyonunun bir lineer denklem sistemine indirgenmesidir. Bu yöntemler, plakalarımız arasındaki S bölgesinde yapılan yaklaşımlar ve sadece C sınırı üzerinde yapılan yaklaşımlar olmak üzere iki sınıfta incelenebilir [Şekil 3.1]. Sonlu farklar (finite difference) ve sonlu elemanlar (finite elements) yöntemleri birinci sınıfta, sınır elemanları (boundary elements) yöntemi ise ikinci sınıfta değerlendirilmektedir. Bu yöntemler günümüze kadar mühendislikte ve fizikte uygulama alanı olan elektromanyetizma, titreşim, stabilite, akışkanlar mekaniği, sürekli ortam mekaniği, sıvı veya termal etkilere maruz yapıların analizi gibi pek çok probleme başarıyla uygulanmıştır.



Şekil 3.1) Bir düzlemde C (sınır) ile S(yüzey) gösterilişi

Şekil 3.2'de dayandıkları ağ şematik olarak gösterilen sonlu elemanlar ve sonlu farklar yöntemleri, yaygın olarak kullanılmaktadırlar ve aşağıdaki alt bölümlerde kısaca ele alınıp tanıtılmışlardır.



Şekil 3.2 a) Sonlu Farklar Yöntemi b)Sonlu Elemanlar Yöntemi

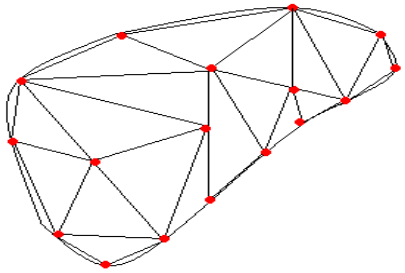
3.1) Sonlu Elemanlar Yöntemi

Sonlu Elemanlar Yöntemi'nin amacı, verilen diferansiyel denklemi ve sınır değer koşullarını yaklaşık olarak sağlayan aynı zamanda da bilinen basit fonksiyonlar aracılığı ile ifade edilebilen çözüm fonksiyonları bulmaktır. Esas olarak yöntem, bütün bir aralıkta yaklaşık çözüm bulmaya çalışmaktansa, tanım aralığını "sonlu eleman" olarak adlandırılan belirli parçalara bölmek, her bir parça için yaklaşık çözüm bulmak ve sonra her bir eleman için bulunan yaklaşık çözümleri belirli kurallar dahilinde birleştirerek tüm aralıkta yaklaşık bir çözüm elde etmek amacı güder.

Sonlu Elemanlar Yöntemi matematikçilerin yanı sıra mühendislerin de özellikle ilgilendiği alan olmuştur. Sonlu Elemanlar Yöntemi, katı mekaniği, sıvı mekaniği, ısı transferi, akustik, dinamik sistemler, elektromanyetizma, biyomekanik, vs. gibi alanlarda, daha çok,

- sınır koşullarının karmaşık olduğu,
- geometrisi düzgün olmayan

problemler üzerinde kullanılmaktadır.



Şekil 3.3) Bölgenin Sonlu Elemanlara Bölünmesi

Sonlu Elemanlar Yöntemi'nde çözüm bölgesi, şekil fonksiyonları (shape functions) önceden bilinen birçok elemana, örneğin 2 boyutlu uygulamalarda üçgen, dörtgen gibi, bölünür. Elemanlar "düğüm" (node) adı verilen noktalarda tekrar birleştirilirler (Şekil 3.2) ve mesh oluşturulur. Bu şekilde cebrik bir denklem takımı elde edilmesi hedeflenir. İncelenen probleme bağlı olarak bu şekilde yüzlerce hatta binlerce denklem elde edilir. Bu denklem takımının çözümü ise bilgisayar kullanımını zorunlu kılmaktadır. Sonlu Elemanlar Yöntemi'nde, temel yaklaşım sürekli fonksiyonları bölgesel sürekli fonksiyonlar (genellikle polinomlar) ile temsil etmektir. Bunun anlamı bir eleman içerisinde hesaplanması istenen büyüklüğün değeri o elemanın düğümlerindeki değerler kullanılarak interpolasyon ile bulunur. Bu nedenle Sonlu Elemanlar Yöntemi'nde bilinmeyen ve hesaplanması istenen değerler düğümlerindeki değerlerdir. Bir varyasyonel prensip (örneğin enerjinin minimum olması prensibi) veya başka benzer yaklaşım kullanılarak büyüklük alanının düğümlerindeki değerleri için bir denklem takımı elde edilir.

Sonlu Elemanlar Yöntemi ilk olarak yapı analizinde kullanılmaya başlandı. İlk çalışmalar Hrenikoff (1941) ve McHenry(1943) tarafından geliştirilen yarı analitik analiz metotları olsa da, metodun başlangıcı olarak Courant'ın (1943) çalışmaları gösterilir [2]. Courant bu çalışmasında bölgesel sürekli lineer yaklaşım kullanarak burulma problemleri için çözüm yöntemi önermiştir. Turner ve diğerleri (1956) bir üçgen eleman için Stiffness matrisini oluşturmuştur [3]. "Sonlu Elemanlar" terimi ilk defa Clough (1960) tarafından telaffuz edilmiştir [4]. Yapı alanı dışındaki problemlerin Sonlu Elemanlar Yöntemiyle çözümü 1960'lı yıllarda başlamıştır. Örneğin Zienkiewicz ve Cheung (1965) Sonlu Elemanlar Yöntemi ile Poisson denklemini çözmüştür. Yöntemin 3-boyutlu problemlere uygulanması 2-boyutlu teoriden sonra kolayca gerçekleştirmiştir. (örneğin Argyis (1964)) Araştırıcılar 1960'lı yılların başında non-lineer problemlerle ilgilenmeye başladılar ve Sonlu Elemanlar Yöntemi'ni non-lineer problemlere de uyguladılar.Sonlu Elemanlar Yöntemi ile stabilite analizi ise ilk Martin (1965) tarafından tartışılmıştır. Statik problemlerin yanı sıra dinamik problemlerde Sonlu Elemanlar Yöntemi'yle incelenmeye başlanmıştır. (Zienkiewicz (1967) [5]. Sonlu Elemanlar Yöntemi geliştirilerek ısı transferi, akışkanlar mekaniği, elektromanyetizma ve diğer birçok benzer alana uygulanmıştır.

Genel amaçlı sonlu elemanlar paket programları 1970'li yıllardan itibaren ortaya çıkmaya başlamıştır. 1980'li yılların sonlarına doğru ise artık paket programlar mikro

bilgisayarlarda kullanılmaya başlandı. 1990 yıllarının ortaları itibariyle Sonlu Elemanlar Yöntemi ve uygulamaları ile ilgili yaklaşık 40,000 makale ve kitap yayınlaşmış olup, günümüzde bu sayının ikiye katlandığı tahmin edilmektedir.

3.2) Sonlu Farklar Yöntemi

Sonlu Farklar Yöntemi, aynen Sonlu Elemanlar Yönteminde olduğu gibi, matematikte diferansiyel denklemlerin çözümünde ve özellikle de sınır değerlerinin bulunmasında kullanılır. Sonlu Farklar Yöntemi, çeşitli diferansiyel denklemlerin kapalı çözümleri elde edilemediğinde, kullanılan sayısal ve yaklaşık yöntemlerdendir.

Sonlu Farklar Yöntemi'nde, herhangi bir koordinat yönünde kısaltılmış bir Taylor serisinin açılımından faydalanarak, fark operatörü çözüm bölgesi üzerine yerleştirilmiş bir dörtgensel ızgaranın her noktasında ayrıştırılır ve uygulanır. Denklem sistemleri iterasyonla ya da doğrudan çözülür. Fonksiyonda, değişkene farklı değerleri verilerek, bu değerlere karşılık gelen birden çok çözüm değerleri bulunabilir. Dolayısıyla asıl istenilen değişkenin değerine karşılık gelen fonksiyon değeri analitik fonksiyonlar için mümkündür. Ancak fonksiyonun çözümünde verilen değer bilinen gözlem veya deneye dayalı verilerin olması durumunda çözülmüş olur. Yine de bilinen değerlerin arasında veya dışındaki noktalarda değerlerine ihtiyaç duyulabilir. Bu işlem bir ara değer bulma problemidir. Ara değer bulma işleminde sonlu fark tabloları da sıkça kullanılır. Sonlu Farklar Yöntemi, sürekli bir sistem olarak tanımlanan sınır değer problemini, düğüm noktası olarak adlandırılan N tane noktaya bölerek kesikli bir sistem haline dönüştürür. Böylelikle diferansiyel denklem eşitliği olarak tanımlanan sınır değer problemi bir grup çözülmesi gereken cebirsel eşitlik haline gelir.

Bu yöntemin dezavantajları, problem geometrisinin kabaca modellenmesi ve özellikle açık alan problemlerinde bilinmeyenlerin çok sayıda olması olarak özetlenebilir.

3.3) Sınır Elemanlar Yöntemi

Sınır Elemanları Yöntemi, probleme esas denklemlerinin çözümü için farklı bir yaklaşım kullanır. Bu yöntemde, ilk önce problemi tanımlayan kısmi diferansiyel denklem bir sınır integral denklemine dönüştürülür. Böylece sınırın sürekli eğrisindeki değişimler çözümlenmiş olur.

Sonlu Elemanlar ve Sonlu Farklar Yönteminde düğüm noktası sayısı arttıkça elde edilen çözümlerin hassasiyetinin arttığı bilinmektedir. Bununla birlikte, düğüm noktası sayısının arttırılması, gerekli olan bilgisayar kapasitesini ve hesap süresini de aynı oranda arttıracaktır. Ancak pek çok problemde gerçek değere yakın hassas sonuçlar fiziksel anlamda ancak birkaç özel noktanın değerinin bilinmesini gerektirmektedir. Sonlu Elemanlar Yöntemi'nde çözüm için yaklaşık bir fonksiyon seçilerek çözüme başlanır. Ancak seçilen interpolasyon fonksiyonları yerel düzeyde olup, elemanlar için geçerlidir. Sonlu Elemanlar Yöntemi ile çözüm bölgesi çok fazla elemana ayrılarak yeter hassasiyette sonuçlar elde etmek mümkündür. Özellikle plak veya kabuk elemanların hassas çözümleri ancak yüksek sayıda elemana bölünerek sağlanır. Elemanın çok fazla bölgeye ayrılarak (mesh generation) çözüme ulaşılması durumunda ise gerekli olan hesaplayıcı kapasitesi ve harcanan zaman artacaktır. Bununla birlikte yeter yaklaşıkta sonuçlar yani gerçek değere çok yakın sonuçlar mühendislik uygulamalarında çoğunlukla bir veya birkaç özel noktada gerekmektedir

3.4) Sayısal Yöntem Seçimi

Sonlu Elemanlar Yöntemi tüm farklı şekildeki elemanlar için rahatlıkla kullanılabilirken, Sonlu Farklar Yöntemi genellikle dikdörtgensel bir yapıya sahip yüzeyler için düşünülmektedir. Sonlu Elemanlar, boyutları ve şekillerinin esnekliği nedeniyle, verilen bir cismi gerçeğe yakın temsil edebilir, bu nedenle karmaşık şekilli bir geometride çok daha güvenilir olabileceğinden tercih edilecektir. Sonlu Farklar Yöntemi'nde problem geometrisi, koordinat eksenlerine paralel eşit aralıklarla yerleştirilmiş düğüm noktaları ile temsil edilirler. Sonlu Elemanlar Yöntemi'nde ise elemanların eşit büyüklükte olması veya koordinat eksenlerine paralel konumda olması gerekmemektedir. Bu özelliği ile Sonlu Elemanlar Yöntemi karmaşık geometrilerin çözümünde kullanılabilir. Çözüm bölgesinde farklı geometrik şekilli elemanlar olabilir. Çok bağlantılı bölgeler (yani bir veya çok delikli cisimler) veya köşeleri olan bölgeler zorluk çekilmeksizin incelenebilir. Ayrıca, Sonlu Elemanlar Yönteminde, değişik malzeme veya geometrik özellikleri bulunan problemler ek bir zorluk göstermez. Geometri ve malzemenin yapısındaki bozukluklar, değişken özellikler (zamana bağlı) malzeme özellikleri kolaylıkla göz önüne alınabilir. Sonlu Elemanlar Yöntemi'nde sınır şartları kolayca uygulanır.

Sonlu Elemanlar Yöntemini temsil amaçlı eklenen şekil 3.3 te kırmızı noktalar düğüm olarak adlandırılır. Her üç düğümün oluşturduğu üçgensel bölge ise eleman olarak adlandırılır. Problem bölgesi ise bu türden sonlu elemanlara bölünerek yaklaşık çözüm elde edilir.

Bu tezde programlama yönünden daha kapsamlı bir algoritmaya dayanması ve programlama açısından daha karmaşık ve ilginç olması gibi faktörler göz önünde bulundurarak Sonlu Elemanlar Yöntemi seçilmiş ve Laplace denklemine uygulanmıştır. Bu yöntemin çözümde kullanılan ayrıştırma metotlarından biri seçilerek çözüm elde edilecektir.

3.5) Sonlu Elemanlara Ayrıştırma Yöntemleri

Sonlu Elemanlar Yöntemi'nde yaklaşık çözüm, parçalı polinomlar olarak tanımlanan sonlu sayıdaki temel fonksiyonlarının lineer kombinasyonu olarak ifade edilir. Yaklaşık çözüm fonksiyonu olarak hesaplanır. Sonlu Elemanlar Yöntemi dahilindeki başlıca yöntemler Varyasyonel - Rayleigh-Ritz Yöntemi [6] ve Residual (Galerkin) Yöntemi'dir [7].

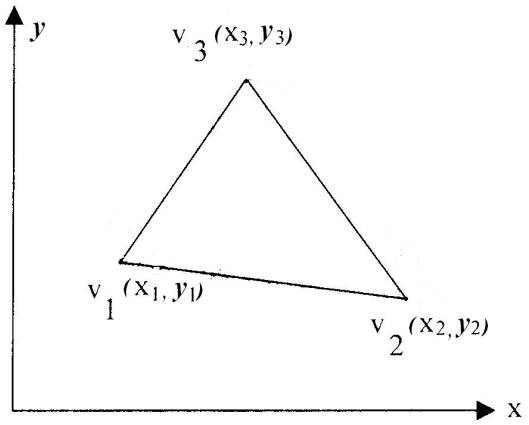
i) Varyasyonel Yöntem - Rayleigh – Ritz Yöntemi

Bu yöntemde, örneğin varyasyonel formülasyon halinde problem bölgesi eleman olarak isimlendirilen çok küçük alt alanlara bölünür. Elemanlar tüm bölgeye düzenli olarak dağıtılabilir veya bölgenin çeşitli kısımlarında yoğunlaştırılabilir. Eleman geometrileri ve bilinmeyenler, düğüm değerlerini içeren polinomlarla ifade edilir. Her elemanın içindeki enerji minimize edilerek Laplace denklemi için çözüm bulunur. Bu metot değişimler hesabına (variational calculus) ve uygun fonksiyonellerin oluşturulmasına dayanır. Verilen her diferansiyel denklem eşitliği için farklı fonksiyoneller oluşturulması gerekir. Bu zorluğa ilave olarak problemin çözümünü doğrudan bulmak yerine enerjinin azaltılmasına yönelik bir çözüm olması da dikkate alınmalıdır. Bu yaklaşık yöntemin basitleştirmedeki başarısı en önemli avantajlarından biridir. Genel olarak en iyi sonucu verilen problemin diferansiyel metotlara dönüştürülmesiyle elde edilebilir.

Varyasyonel metodun avantajı uygulamada basitlik ve yaklaşık yöntem üzerinde teorik garantisinin bulunmasıdır. Dezavantajı ise teorisinin çoğu zaman karmaşık olması ve fonksiyonelin bulunamadığı durumlarla karşılaşılmasıdır.

ii) Ağırlıklı Residü Yöntemi (Weighted Residual Method):

Diferansiyel eşitlikleri ayrık probleme dönüştürülmesinde kullanılan metotlar sınıfıdır. Prensipte fonksiyon uzayına metodun uygulanmasının anlamı eşitliği daha zayıf bir formülasyona dönüştürmektir. En tipik olanı, fonksiyon uzayını temel fonksiyonlardan oluşan sonlu kümeye karakterize ederken kullanılan kısıtlamalardır. Bu metot Rus matematikçi Boris Galerkin tarafından inşa edilmiştir. Bu yöntemde yaklaşık çözümü veren bir ifadenin var olduğu kabul edilir ve bu ifade taban (basis) fonksiyonları (aslında katsayıları bilinmeyen bir polinomdur) yardımıyla elde edilir. Çözümdeki katsayılarının bulunması için gerekli lineer denklem sistemi oluşturma esasına dayanır. İlk olarak yapılması gereken düğüm noktalarını (eşit aralıklı olması şart değildir) seçmektir. Taban fonksiyonlarının oluşturulmasında çoğunlukla lineer ifadeler kullanılmakla beraber yüksek mertebeden diferansiyel denklemlerin çözümleri için parabolik, hiperbolik, vb. ifadeler de kullanılabilir. Seçim tamamlandıktan sonra yönetim denkleminin (governing equation) bilinmeyen büyüklüğünün örneğin potansiyel V yerine yaklaşık çözümü ifade eden V_a kullanması halinde sağ taraf sıfırdan farklı olacaktır, yani R ile gösterilen residü olacaktır. Bu residünün ağırlık fonksiyonları ile çarpılmış ifadesinin toplam yüzey (veya hacmi) boyunca integralini sıfır yapma istenilen cebirsel denklemleri dolayısıyla çözümü bulmayı sağlayacaktır. Burada karar verilmesi gereken şey ağırlık fonksiyonlarının ne olacağıdır ki, Collacataion, Sub-Domain, Galerkin gibi yöntemler burada ortaya çıkmaktadır. Bu yöntemler arasında Galerkin yöntemi sonlu elemanlar için varyasyonel yöntemlerin alternatifidir ve basitçe ağırlık fonksiyonu olarak taban yada şekil fonksiyonu olarak da bilinen yaklaşık fonksiyonu kullanır.



Şekil 3.4) Üçgensel Bölgenin 2-boyutlu koordinat üzerinde gösterilişi

Residual metot çözülmesi gereken fiziksel eşitliğe uygulanabilirlik Varyasyonel Yöntem ile kıyaslandığında daha basit ve uygulamada kolaylık açısından daha avantajlıdır. Modellemede 2-boyutlu üçgen elemanlar kullanıldı. Bir üçgensel elemanın lineer fonksiyonel ifadesi şu şekildedir:

$$V(x,y) = a_1 + a_2 x + a_3 y \tag{3.1}$$

Burada V elektriksel potansiyeldir. Ancak başka herhangi bir büyüklük de olabilirdi. Herhangi bir üçgendeki her bir düğüm için V aşağıdaki şekilde biçimlenecektir.

$$V_1 = a_1 + a_2 x_1 + a_3 y_1, \qquad V_2 = a_1 + a_2 x_2 + a_3 y_2, \qquad V_3 = a_1 + a_2 x_3 + a_3 y_3$$

Yukarıdaki bu eşitliklerden yola çıkarak D determinantı göstermek üzere a katsayılarını matris olarak gösterecek olursak:

$$D = \begin{vmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{vmatrix} \text{ olmak kaydıyla,}$$

$$a_1 = \frac{1}{D}\begin{vmatrix} V_1 & x_1 & y_1 \\ V_2 & x_2 & y_2 \\ V_3 & x_3 & y_3 \end{vmatrix} \qquad a_2 = \frac{1}{D}\begin{vmatrix} 1 & V_1 & y_1 \\ 1 & V_2 & y_2 \\ 1 & V_3 & y_3 \end{vmatrix} \qquad a_3 = \frac{1}{D}\begin{vmatrix} 1 & x_1 & V_1 \\ 1 & x_2 & V_2 \\ 1 & x_3 & V_3 \end{vmatrix}$$

olarak hesaplanır. Eğer

$p_1 = x_2 y_3 - x_3 y_2$, $q_1 = y_2 - y_3$ ve $r_1 = x_3 - x_2$

$p_2 = x_1 y_3 - x_3 y_1$, $q_2 = y_1 - y_3$ ve $r_2 = x_3 - x_1$

$p_3 = x_3 y_1 - x_1 y_3$, $q_3 = y_1 - y_2$ ve $r_3 = x_2 - x_1$

olarak ifade edilirse, eşitliğimiz aşağıdaki gibi formülüze edilebilir.

$$V(x,y) = \sum_{i=1}^{3} \frac{1}{D}\left(p_i + q_i x + r_i y\right) \tag{3.2}$$

Bir üçgen eleman için bulunan bu potansiyel ifadesinden elektriksel alan E kolayca elde edilebilir.

$$\vec{E} = -gradV = \vec{i}E_x + \vec{j}E_y = -\vec{i}\left(\frac{\partial V}{\partial x}\right) - \vec{j}\left(\frac{\partial V}{\partial y}\right) \quad \text{dolayısıyla}$$

$$\vec{E} = \vec{i}\left(\frac{1}{D}\right)(q_1V_1 + q_2V_2 + q_3V_3) - \vec{j}\left(\frac{1}{D}\right)(r_1V_1 + r_2V_2 + r_3V_3) \qquad (3.3)$$

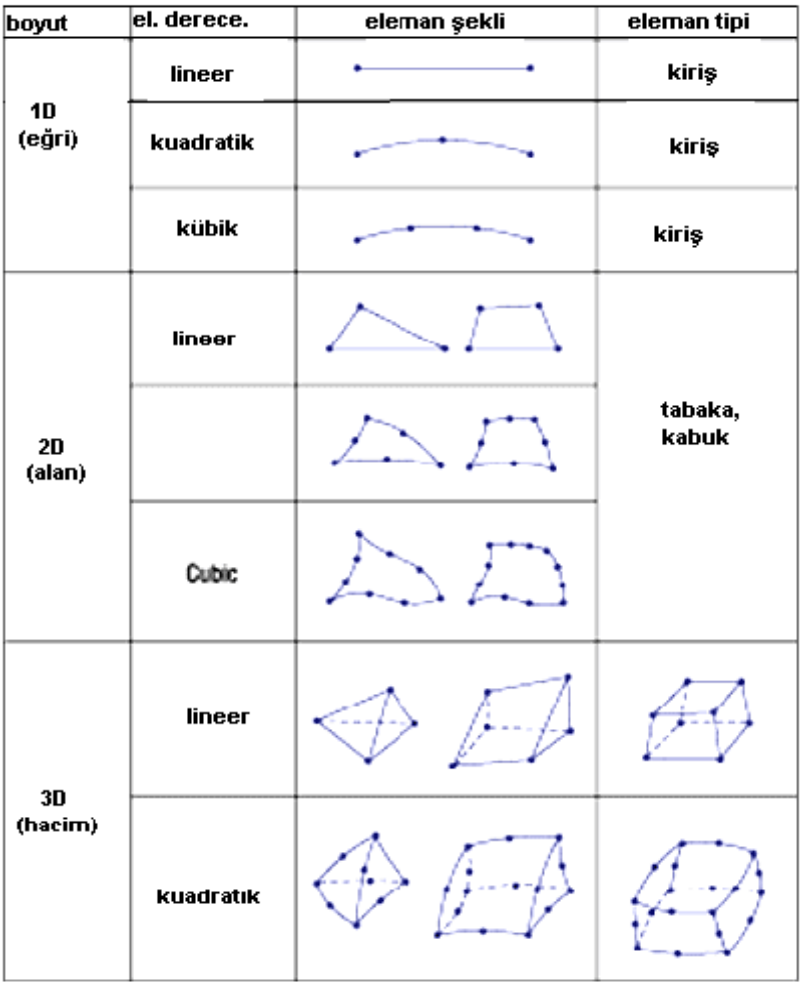
Şekil (shape) fonksiyonları kullanılarak genelleştirmeye gidileceğinden,

$$N_1(x,y) = \left(\frac{1}{D}\right)(p_1 + q_1x + r_1y), \qquad N_2(x,y) = \left(\frac{1}{D}\right)(p_2 + q_2x + r_2y),$$

$$N_3(x,y) = \left(\frac{1}{D}\right)(p_3 + q_3x + r_3y) \qquad \text{olacaktır.} \qquad (3.4)$$

Burada N ‘ler şekil ya da temel (basis) fonksiyonlar olarak adlandırılırlar.

Sonlu elamanlar yöntemi için bu çalışmada bölge üçgen elemanlara bölünmesinden dolayı elde edilen şekil fonksiyonları sadece üçgen elemanlar için çıkarılmıştır. Unutmamak gerekir ki, aşağıda şekilde verildiği gibi 1-boyutlu, 2-boyutlu ve 3-boyutlu çeşitli tipte elemanlar probleme bağlı olarak kullanılabilir.



Şekil 3.5) 1-boyutlu, 2-boyutlu ve 3-boyutlu Sonlu Eleman Örnekleri

3.6) Laplace Denklemi için Sonlu Eleman Denklemleri

Residual Yöntem öncelikle residünün elde edilmesini gerektirir. Bu amaçla, nümerik yöntemlerde yaklaşık sonuç elde edilebileceğinden yola çıkarak, (2.14) Laplace denkleminde kullanılan V potansiyel ifadesi yaklaşık V_a ile değiştirilirse, denklemin sağ tarafı sıfırdan farklı, bir residü, olacaktır.

$$div\varepsilon\,(-\,grad V_a) = R \tag{3.5}$$

Burada V_a üçgen bir eleman için

$$V_a = \sum_{i=1}^{3} N_i V_i \tag{3.6}$$

elde edilir. Ağırlıklı Residü (Weighted residue) yöntemi, residü ifadesinin yüzey boyunca ağırlıklı olarak integralinin sıfırlanarak parametrelerin bulunmasına dayandığından,

$$\int_{\Omega} W R \, d\Omega = 0 \tag{3.7}$$

bilinmeyenlerin bulunmasını sağlayacaktır. Burada, Ω bölge yüzeyini W ise ağırlık (weighting) fonksiyonunu ifade etmektedir. Galerkin yöntemi ise ağırlık fonksiyonu olarak şekil fonksiyonlarını kullanır. Bu durumda Galerkin formülü aşağıdaki gibi elde edilir.

$$\int_{\Omega} N\left[div(\varepsilon \, grad V_a)\right] d\Omega = 0 \tag{3.8}$$

Diverjans teoremi yardımıyla (3.7) eşitliğinin sol tarafı

$$\int_{\Omega} N\left[div(\varepsilon \, grad V_a)\right] d\Omega = \oint_{c} N \varepsilon \, grad V_a \, dl + \int_{\Omega} \varepsilon \, grad V_a \; grad N \, d\Omega \tag{3.9}$$

olarak ifade edilebilir. Yukarıdaki eşitliğin sağ tarafının ilk terimi direkt olarak sınır koşulları ile ilişkilidir ve dış sınırlar hariç çoğunlukla dikkate alınmaz. İkinci terim ise çözüm matrisini oluşturacak asıl denklem işlevi görecektir. (3.9) denklemi orijinal ikinci dereceden birinci dereceye düşürüldüğü için zayıf form olarak adlandırılır.

Bölgedeki toplam düğüm sayısını K olarak ifade edersek,

$$\sum_{k=1,K} \int_{\Omega k} \left[\varepsilon \, grad V_a . \; grad N_k\right] d\Omega = 0 \tag{3.10}$$

elde edilir.

Yukarıdaki (3.10) denkleminin matris formuna dönüştürülebilmesi için V_a yerine sadece genelleme açısından V kullanılırsa ve (3.2) tekrar dikkate alınırsa,

$$V(x,y) = \frac{1}{D}(p_1 + q_1 x + r_1 y)V_1 + \frac{1}{D}(p_2 + q_2 x + r_2 y)V_2 + \frac{1}{D}(p_3 + q_3 x + r_3 y)V_3$$

elde edilir, ve buradan

$$gradV = \vec{i}\,\frac{1}{D}\left(q_1V_1 + q_2V_2 + q_3V_3\right) + \vec{j}\,\frac{1}{D}\left(r_1V_1 + r_2V_2 + r_3V_3\right) \qquad (3.11)$$

elde edilir.

Şekil fonksiyonları için gradient operatörü uygulanırsa

$grad\,N_1 = \vec{i}\,\frac{1}{D}q_1 + \vec{j}\,\frac{1}{D}r_1$, $grad\,N_2 = \vec{i}\,\frac{1}{D}q_2 + \vec{j}\,\frac{1}{D}r_2$ ve

$grad\,N_3 = \vec{i}\,\frac{1}{D}q_3 + \vec{j}\,\frac{1}{D}r_3$ elde edilir.

Buradan bir üçgen eleman için matris formu eğer $\alpha = \frac{\varepsilon}{2D}$ ise aşağıdaki gibi elde edilir.

$$[S] = \alpha * \begin{vmatrix} q_1*q_1 + r_1*r_1 & q_1*q_2 + r_1*r_2 & q_1*q_3 + r_1*r_3 \\ q_1*q_2 + r_1*r_2 & q_2*q_2 + r_2*r_2 & q_2*q_3 + r_2*r_3 \\ q_1*q_3 + r_1*r_3 & q_2*q_3 + r_2*r_3 & q_3*q_3 + r_3*r_3 \end{vmatrix} \begin{vmatrix} V_1 \\ V_2 \\ V_3 \end{vmatrix} \qquad (3.12)$$

Tüm elemanlar için tek tek elde edilebilecek olan yukarıdaki gösterim, eleman birleştirme operasyonu yardımıyla ortak düğüm numarası taşıyan elemanlar arasında uygun satır ve sütun element düzeyinden global düzeye genişletilerek Stiffness matris olarak bilinen matris elde edilir.

Çözümden bir önceki aşama sınır koşullarının uygulanmasıdır. Örnek uygulamada kaynak yük olmadığından elektrik alan dış sınırlardaki Dirichlet sınır koşulundan (sabit potansiyel değer) ortaya çıkacaktır. Sabit potansiyel değerler bu sınırlarda zorlayarak (imposing) uygulanırlar. Bir başka ifadeyle değeri sabit olarak bilinen bu düğüm potansiyelleri bilinmeyenler dışına alınarak, kalan düğüm değerleri için çözüme gidilir. Bu noktada iteratif veya eleme yöntemlerinden biri seçilerek son aşama gerçekleştirilir. Bu tezde bilinmeyen sayısının azlığı dikkate alınarak Gauss Eliminasyon yöntemi ile sonuç bulunmuştur.

BÖLÜM 4

4. Model ve Programlama

Sayısal yöntemler yoluyla herhangi fiziksel bir problemin çözümünde model oluşturma ve programlama oldukça önemlidir. Bu bölümde çözümde kullanılan yöntemin genel modelleme süreci ve özelde problemde kullanılan model verilmiştir. Ayrıca çözümde kullanılan programlama tekniği özellikle dil açısından ele alınmış ve tartışılmıştır.

4.1) Modelleme

Genel olarak CAD sistemlerinin dolayısıyla Sonlu Elemanlar Yöntemi'nin de uygulanması üç ayrı aşaması vardır. Bu aşamalar:

1. Prepocessing (ön işlem):

Bu aşamada, çözüm öncesi problemin geometrisinin belirlenmesi ve gerekli input verilerinin hazırlanması yapılır. Kısaca maddeleyecek olursak, preprocessing adımı aşağıdaki işlemleri kapsar.

-Problem geometrisinin ve koordinatlarının belirlenmesi.

-Geometrinin sonlu elemanlara bölünmesi ve elemanlar ve düğümlerin (nodes) elde edilmesi.

-Elemanın fiziksel özelliklerine uygun şekil fonksiyonlarının seçilmesi.

-Eleman denkleminin oluşturulması.

-Elemanların birleştirilerek global matrisinin oluşturulması.

-Sınır şartları, başlangıç şartları ve kaynak ifadelerinin uygulanması.

2. Çözüm (solution) :

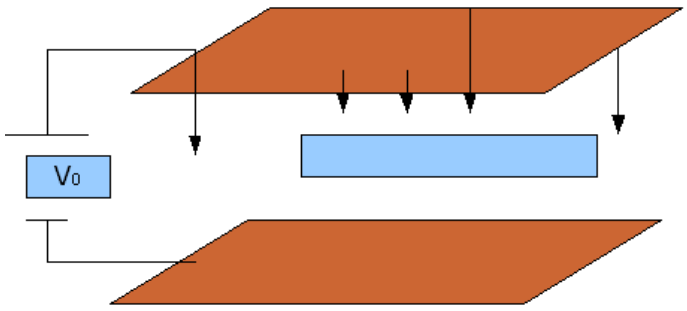
Bu safhada, diferansiyel denklemlerden cebirsel denklemlere (genellikle matris formunda) dönüşüm sonrası bilinmeyenleri bulmak için denklem sisteminin çözümü hedeflenir. Bu işlem sonunda tüm düğümlerde aranan büyüklüğün değerler bulunur. İterasyon veya direkt çözüm yöntemleri, bilinmeyenleri bulmak için kullanılır.

3. Post-processing (çözüm sonrası):

Düğüm noktalarında değerleri bulunan büyüklüklerin grafiksel ve görsel gösterimi sonuçların değerlendirilmesini kolaylaşlaştırır. Özellikle farklı renk tonlarıyla gösterilmiş alan dağılımları çok daha tanımlayıcıdır ve post-processing aşaması bu ve benzeri gösterimleri olanaklı kılar. Ayrıca temel büyüklüğün yanısıra, başka yardımcı büyüklük değerleri de bu aşamada hesaplanıp görselleştirilir.

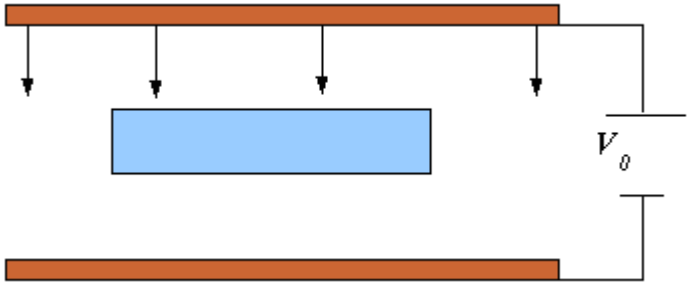
4.2) Örnek Problemin Modeli

Bu çalışmada örnek alınan problemin fiziksel modeli iki düzlemsel iletken levha ve arasında yeralan hava ve/veya dielektrik malzemeden oluşmaktaydı. Levhalardan birine V_0 =100[V] diğerine ise 0 [V] sabit potansiyel değerleri bir üreteç aracılığıyla uygulanacaktır. İletken levhalar arasında birinci halde sadece hava olduğu, ikinci durumda ise daha genel, havanın yanısıra dielektrik bir malzemenin de olduğu modeller kullanılmıştır. Aşağıda verilen şekil 4.1, genel dielektrikli modeli şematik olarak göstermektedir. Şekilde 3-boyutlu gösterim kullanılmıştır. Ancak iletken levhaların yüzeylerinin oldukça geniş, aralarındaki mesafenenin bununla göreceli olarak küçük olduğu durumda geometrik boyutu 3-boyut'tan 2-boyut'a indirgemekle, levhalar arasında kalan bölgedeki potansiyel ve alan dağılımlarında önemli bir değişiklik olmayacağı düşünüldüğünden dolayı problemi 2-boyutlu olarak çözmek yeterli olacaktır.



Şekil 4.1) Örnek Problemin Gösterilişi

Bir başka ifadeyle, bu problem, levha yüzeyinde potansiyelin değişmiyeceğini varsayarak 2-boyutlu bir problem olarak tasvir edilebilir. Şekil (4.2) boyutu 2-boyutluya indirgenmiş ve Sonlu Elemanlar Yöntemi'nin uygulanmış olduğu modeli göstermektedir.



Şekil 4.2) Örnek Problemin 2-boyutlu gösterilişi

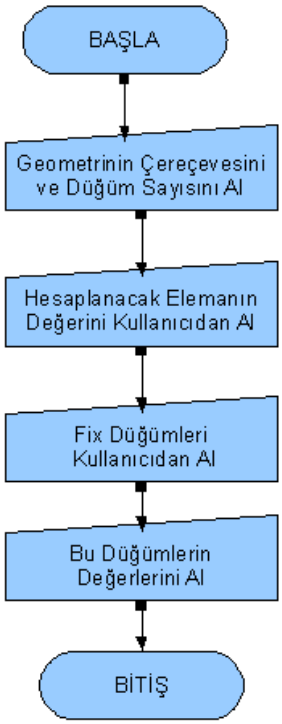
Modelde iletken levhalar arasında kalan bölgedeki potansiyel dağılımı ve ondan yola çıkarak elektrik alan bulunacaktır.

Problemin genel hatlarını belirledikten sonra önceki bölümlerde verilen formüllerle birlikte programın nasıl gerçekleşeceği incelenecektir.

4.3) Programlama

Problemin çözümü sayısal yöntem yoluyla yani bir bilgisayar program yardımıyla gerçekleştireleceğinden dolayı, bu program için temel verilerin oluşturulması ve programa input olarak aktarılması gerekmektedir.

Aşağıda şekil 4.3'te verilen diagramda program akış şeması gösterilmiştir. Buna göre, 2-boyutlu düzlemde x ve y eksenlerindeki düğüm sayısı kullanıcıdan istenir. Bu düğüm sayısı seçimi madde üzerindeki potansiyel dağılımın hassaslığının belirlenmesi açısından önemlidir. Eğer x ve y eksenlerindeki düğüm sayısı çok fazla seçilirse bazı potansiyel değerlerdeki değişimler anlaşılamayacak kadar birbirine yakın, eğer düğüm sayısı az seçilirse düğümler arasındaki potansiyel değişim problemin çözümüne hizmet etmeyecek kadar fazla olacaktır.



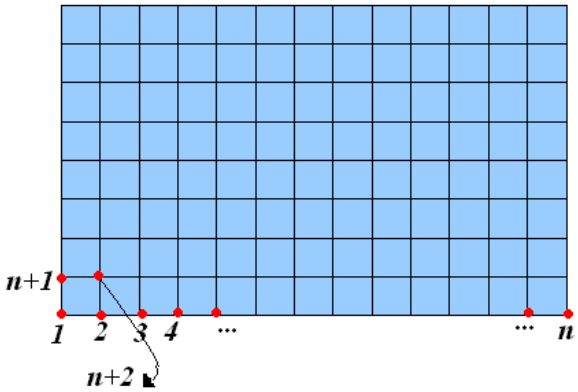
Şekil 4.3) Input.txt dosyası için akış diyagramı

Düğüm sayısı belirlendikten sonra programımız plakalar arası uzunluğu da kullanıcıdan alır. Böylece kullanıcı uzunluk ile düğüm arasındaki bağlantıyı oluşturabilecek esnekliğe sahip olacakken problem de gerçek oranlarda test yapmaya müsait olacaktır.

x ve y koordinatları belirlendikten sonra analiz edilecek problem yüzeyinin bulunduğu sonlu elemanların düğüm (node) numaraları kullanıcıdan alınarak problemin oluştuğu bölge belirlenmiş olur. Daha sonra malzeme özellikleri belirlenir. Mesela elastik bir analiz için isotropic malzeme özeliklerinden elastikiyet modülü, Poison oranı ve yoğunluk gibi özellikler girilir. Burada tez konusu olan problemin gereği iki levha arasına yerleştirilmiş olan dielektrik maddelerin dielektrik katsayıları programa giriş olarak verilmesi gerekmektedir. Bunlar problemin bulunduğu yüzeyin diğer bölgeden farklı olan özelliğinin alınması olarak değerlendirilebilir.

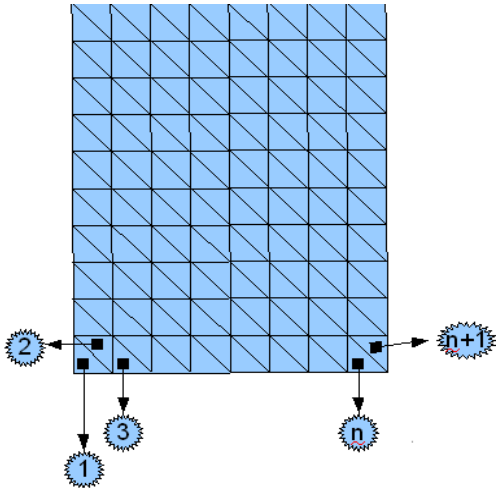
Tüm bu verileri bir input dosyasında toplamak yararlı olacaktır. Kullanıcıdan alınan x ve y eksenlerindeki düğüm sayılarına göre alan belirlenir. Bu işlemde tüm düğümler numaralandırılır. Numaralandırma işlemi standart olarak Şekil 4.4 daki gibi yapılır. Tüm

düğümler aşağıdan başlanarak soldan sağa doğru numaralandırılır. Bu Sonlu Elemanlar Yönteminin programlanması için önemlidir.



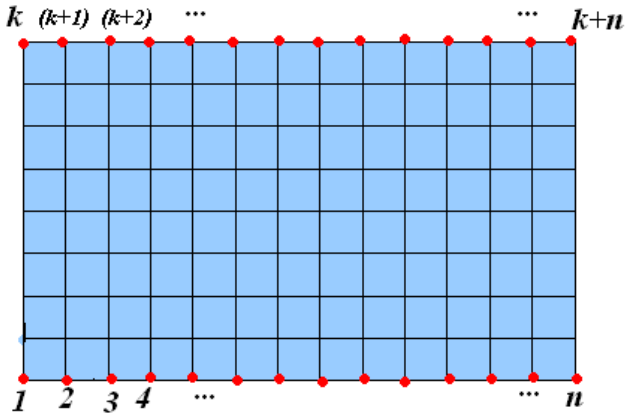
Şekil 4.4) Düğümlerin numaralandırılması

Numaralandırma işlemi bittikten sonra tüm alan üçgensel elemanlara bölünür. Elemanlar numaralandırılırken saat yönünün tersine doğru numaralandırılacaktır. Bu gösteriliş Şekil 4.5 de belirtilmiştir. Her üç düğüm arasında kalan bölge eleman olarak adlandırılmaktadır.



Şekil 4.5) Eleman numaralandırılması

Elemanların belirlenmesinin ardından kullanıcıdan potansiyel değerleri daha sonradan belirlenecek olan plakaların bulunduğu düğümler istenecektir. Şekil 4.6 de plakaların bulunduğu düğümler örnek olarak gösterilmiştir.



Şekil 4.6) Sınırların belirlenmesi

Potansiyel farklılık yaratan plakaların bulunduğu düğümlerin seçiminin ardından bu plakaların potansiyel değerleri kullanıcıdan istenir. Bu adımda programa modelin sınır bölgelerindeki koşullar girilir. Ayrıca varsa model üzerindeki diğer etkiler belirtilir. Mesela katı mekaniği için destek noktaları ve yük uygulama noktaları belirtilir. Veya ısı problemleri için sınırlardaki sıcaklılar veya ısı akıları belirtilir. Biz belirli bir problem için bu uygulamayı tasarladığımız için program kodu yalnızca sınır bölgedeki potansiyel değerlerin alınmasını içermektedir. Çalışılan yüzeyin sınırlarında potansiyel değerleri değişmektedir. Problemin oluşmasının nedeni de zaten budur. Dolayısıyla kod içerisinde bu sınır bölgedeki potansiyellerin talep edilmesi gerekmektedir. Potansiyelleri belirlenmiş olan sınırlarda artık mevcut problem yüzeyinin durumunu analiz edebiliriz. Geriye plakalar arasındaki, eğer varsa, dielektrik maddenin bulunduğu elemanların seçim kalır. Kullanıcıdan maddenin bulunduğu eleman numaraları istenir. Maddenin dielektrik sabiti de kullanıcıdan istendikten sonra problemin tüm veriler alınmış ve input dosyası oluşturulmuş olur.

Eleman özellikleri belirlendikten sonra model otomatik olarak sonlu elemanlara ayrılır. Böylece problemin mesh üretimi tamamlanmış olur. Burada önemli olan seçilen eleman kullanılarak modelin nasıl daha iyi küçük parçalara bölüneceğidir.

Şekil 4.7) Çözücü (solver) akış şeması

Mesh üretimi için öncelikle problemin çözümünde kullanılacak olan matrisin temizlenmesi gerekmektedir. Tüm düğümlerdeki bilinmeyen potansiyel enerji değerleri, matrisin sağ tarafı ve sınır değerleri temizlenir. Her bir düğüm için Stiffness matrisi de sıfıra atanır.

İkinci adımda daha önce verileri tutulan dosyadan gerekli bilgiler alınır. Input dosyası içeriği zaten düğüm sayısı, eleman numarası belirli düzende yazılmıştı. Burada toplam düğüm, eleman sayısı gibi bilgiler alınacaktır. Ardından tüm düğümlerin koordinatları okunarak, sınır koşulları ve diğer tüm veriler input dosyasından programdaki değişkenlere aktarılır.

Stiffness matrisin oluşumu için daha önce vermiş olduğumuz eşitliğin verilerinin elde edilmesi gerekmektedir. Elemanların tümü için elemanları oluşturan düğümler bulunup bu düğümlerdeki koordinatlar okunur. Böylece matris formunu oluşturan veriler yerlerine yazılıp hesaplama yapılabilir.

$$r_1 = x_2 - x_3 \quad , r_2 = x_1 - x_3 \quad , r_3 = x_1 - x_3$$

değerleri bu düğümlerin x-ekseninden ;

$$q_1 = y_2 - y_3 \quad , q_2 = y_1 - y_3 \quad , q_3 = y_1 - y_3$$

değerleri düğümlerin y-ekseninden elde edilecektir.

$$D = \begin{vmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{vmatrix}$$ formülü bilgisayar dilinde

$$D = (x_2 * y_3) + (x_1 * y_2) + (x_3 * y_1) - (x_1 * y_3) - (x_3 * y_2) - (x_2 * y_1)$$

olacaktır. Bu determinant da hesaplanmasından sonra, sıra hava aralığı bölge malzemelerinin geçirgenlik değerlerinin işlenmesine gelir. Bu bilgi yine input dosyamızda kullanıcıdan alınan değerle belirlenmişti. Katsayı hesaplama işlemi de $coeff = \frac{Perm}{D}$ formülü ile elde edilirki, burda, Perm maddenin dielektrik geçirgenliğini ve D yukarıda hesaplanan determinantı gösterir. Artık Stifness matrisini hesaplamak için gerekli tüm verileri elde edilmiş olur.

Bir sonraki adımda sınır koşulları alınır. Sınır koşulları Stifness matrisin katsayısı ile ilişkili bir değer alacağından Stifness matrisin sınırdaki değerleri temizlenir.

Yeniden matriste eleme uygulanarak mevcut maddedeki potansiyel değerler her bir düğüm için hesaplanmış olur.

4.4) Model için Programlama Araçları ve Dilleri

Bir önceki bölüm de (4.3) işlemleri bilgisayar aracılığı ile yapabilmek için, iki ayrı dil ortamı kullanıldı. Burda iki ayrı dille kodlamanın nedeni ise, Sonlu Elemanlar Yöntemi için yaygın olan iki yaklaşımın karşılaştırılabilmesi ve en uygun ortamın önerilmesidir. Bir yapısal dil (C) ve bir de script dili (MATLAB) programlama araçları olarak seçildi. Her iki ortam da oldukça yaygın kullanımı olan ortamlardır.

Bunlardan MATLAB yüksek seviyeli bir teknik programlama ortamıdır. MATLAB aracılığıyla algoritmalar oluşturulabilir ve verilen bir problemin çözümünü kolayca elde etmeye yarayan araçlara sayesinde sonuçlar elde edilip görselleştirilebilir. Görselleştirme ve grafiksel uygulamalara olanak tanıması MATLAB'ın en güçlü özellikleridir. Ayrıca bir diğer üstün yanı da kodların oldukça kısa yazılabilmesidir. Geleneksel programlama araçlarına göre kimi zaman 1/4 oranında kod satırı yeterli olabilmektedir. İçerdiği "toolbox" adı verilen paketler aracılığıyla sayısal işaret işleme, kontrol tasarımı, test ve ölçüm, finansal modelleme ve analiz, haberleşme gibi birçok alanda kullanılabilir [8]. Ancak MATLAB daha çok mühendislik uygulamaları için oluşturulmuş bir yazılım paketi olduğundan bu çalışma için de kullanmaya ve seçenek ortam olarak tanımlamaya oldukça elverişlidir. MATLAB öğrenilmesi basittir. Çünkü mühendislik bilimlerinde gerekli matematiksel formüller ve basit programcılık bilgileri ile problemlere çözüm bulmayı kolaylaştırır. MATLAB çok geniş ve güçlü özelliklere sahip bir hesap makinesi olarak da düşünülebilir. Bu nedenlerden dolayı özellikle son yıllarda bir çok Sonlu Elemanlar Yöntemini kullanan programlar için MATLAB ortamı önde gelen tercihtir.

MATLAB ilgili öne sürülebilecek olumsuzluklara gelince, ücretli ve göreceli olarak pahalı olmasıdır. Ayrıca en önemli zayıflığı ise script dili olmasından kaynaklı çalışma süresinin (run-time) yüksek olmasıdır. Yazılan kodların çalıştırılabilmesi için genellikle programın kendisinin kurulu olmasına ihtiyaç duyulması da bir diğer olumsuzluğu olarak gösterilebilir.

Sonlu Elemanlar Yöntemi'nin uygulanmasında ilk yıllardan 80'lerin ortalarına kadar FORTRAN dili kullanılan en yaygın programlama aracıdır. Bu anlaşılabilir bir durumdur. Özellikle temel bilimler ve mühendislik alanlarında FORTRAN, bir miktar BASIC ve PASCAL uygulamalarının dışında hemen hemen tek seçenekti. Ancak C dilinin 70'lerin ortalarından itibaren kullanılmaya ve 80'ler birlikte yaygınlaşmaya başlaması trendi C diline doğru döndürmüştür. C dili bilindiği gibi yapısal programlama dilleri arasında yüksek seviyeli dil olarak makinaya en yakın dildir [9]. Bunda özellikle işaretcileri kullanmak yoluyla programların çalışma performanslarında inanılmaz iyileştirmeler sunması en büyük avantajı olarak değerlendirilebilir. Sonlu elemanlar uygulamaları için geliştirilen programlar, uygulandıkları alana bağlı olarak genellikle yüksek seviyede CPU ve bellek kullanımına gereksinim duyar. Bu nedenle herhangi bir programlama dili bu kaynakları ne denli etkin kullanabilirse Sonlu Elemanlar Yöntemi için o denli tercih

nedenidir. C dili 80'ler itibarıyla sonlu elemanlar alanındaki programlama uygulamaları açısından bu performans üstünlüğü nedeniyle zamanla FORTRAN dilinin yerini almıştır. Bilinen ve aşağıda verilen klasik C dili avantajlarıyla birlikte 90'lara gelindiğinde C ve sonrasında C++ ilk tercih edilen program geliştirme ortamı olmuştur.

- Yüksek performanslıdır.
- Platform bağımsızlığı ve taşınabilirlik özelliği en üst düzeydedir.
- Çoğunlukla derleyicileri ve editörleri ücretsiz edinilebilir.
- Lisans sorunları yoktur.
- En iyi tasarlanmış yapısal programlama dilidir ve kompak program yazımına elverişlidir.
- Oldukça gelişmiş ve yaygın kullanılan hazır kütüphaneleri vardır.

Tüm bu verilen nedenler C dilinin tercih edilmesinde önemli bir yer tutar. Bu arada bazı dezavantajlardan da söz edilebilir. Kod yazımı, kompaktlığından ve işaretci (pointer) değişkenlerden dolayı diğer dillere göre göreceli zordur. Özellikle grafiksel uygulamalar için farklı kaynaklarca geliştirilmiş kütüphanelere ihtiyaç duyabilir. Daha etkin program yazımı için bir IDE'ye (Integrated Developmen Environment) gerek duyar ki, çoğu zaman bu IDE'ler ücrete tabiidir.

MATLAB ile karşılaştırıldığında C dili performans üstünlüğü konusunda tartışmasız öndedir. MATLAB birçok klasik algoritmayı tek bir komutta sunabilmektedir. Matematik temelli düşünüldüğünden matris ve dizi işlemleri son derece kısa yolla pratik olarak kullanılmaktadır. Bu nedenle matematiksel hesaplamaların bilgisayarda yapılması C dilinde olduğundan daha pratik ve daha az kod yazılarak gerçekleştirilebilmektedir. Böylece problemin bilgisayara uyarlanmasından çok problemin kendisine yoğunlaşmayı olanaklı kılar. Ayrıca pre-processing ve post-processing aşamalarında duyulan grafiksel destekler konusunda MATLAB rakipsiz olup C programcılarının kabusu sayılan grafiksel uygulamalar konusunda herhangi bir yardımcı kaynağa ihtiyaç göstermez.

Bu tez kapsamında çözüm için ekte verilen kaynak kodlar bölgenin sonlu elemanlara bölünmesi yoluyla mesh üretimini sabit boyutlu üçgenler olmak kaydıyla

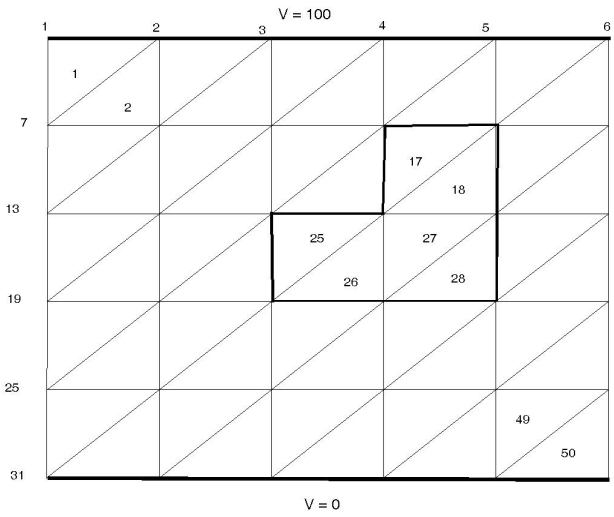
otomatik olarak gerçekleştirmektedir. Programlar her iki MATLAB ve C ortamı aracılığıyla geliştirilmiş olup, verilen iş akış şemasına uygun olarak çalıştırılmaktadırlar.

BÖLÜM 5

5. Program Sonuçları ve Değerlendirme

Bu çalışmada potansiyel ve elektriksel alan dağılımına ilişkin sonuçlar aynı model için gerek MATLAB gerekse C dili ortamlarında geliştirilen programlardan alınmıştır.

Programlarda şekil 5.1 te verilen model esas alınmıştır. Bu modele göre hava aralığı ve dilektrik bölge eşit aralıklarla 6*6 toplam 36 düğümden ve bu düğümlerin 3'erli birleştirilmesi ile oluşan 50 üçgen elemandan oluşturulmuştur. Doğaldır ki düğüm sayısını ve dolayısıyla eleman sayısını belirleme yöntemin sonuçlarının gerçeğe yakın olması açısından önemlidir. Her zaman olmamakla birlikte elamanların küçük, nod sayısının fazla seçilmesi yöntemin hassaslığını arttırır. Bu çalışmada kısmen kolay sayılabilecek bir örnek uygulama seçildiğinden ve problem test amaçlı olarak kullanıldığından az sayıda düğüm ve eleman sayıları yeterli görülmüştür.



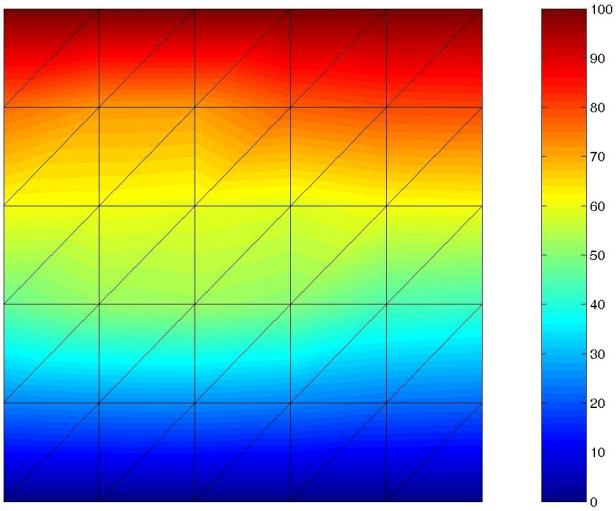
Şekil 5.1) Programlanan Model

Yukarıda verilen modelde kalın çizgilerle belirtilen kenarlardaki 1,2,3,4,5,6 ve 31,32,33,34,35,36 nolu düğümlere sabit potansiyel değerler (Dirichlet koşulu gereği),

100V ve 0V atanmış olup, 17,18,25,26,27,28 nolu elemanlarda dielektrik bölge, geri kalanlarda ise hava bölgesi tanımlanmıştır.

Bu arada hava aralığı, eğer metrik sistem düşünülürse 10 [m] gibi gerçekci olmayacak bir değerde seçilmiştir. Ancak bu değere 10 [m] demek yerine şekilden de anlaşılacağı ve farklı birimde sonuçlar değişmiyeceğinden, 10 birim olarak almak daha pratik olacaktır. Dileketrik malzemenin dilektrik katsayısı $\varepsilon_r = 5$, havanın ise $\varepsilon_r = 1$ olarak programda alınmıştır. Bu program için gerekli olan input.txt dosyası Ek bölümde verilmiştir.

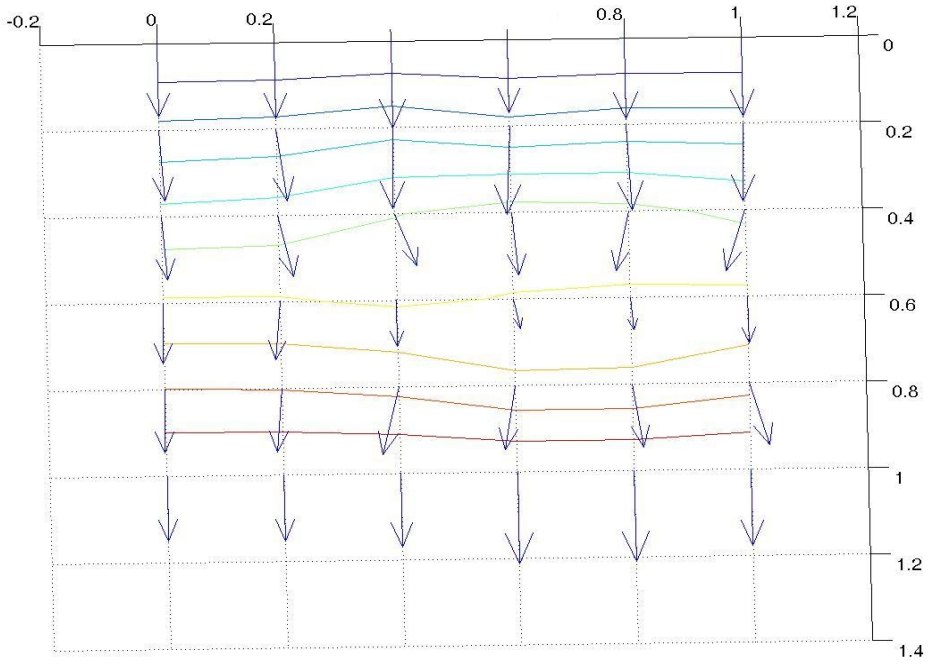
Program akış şemasına uygun olarak uygulanan prosedürler sonucu Dirichlet Kenarlar dışındaki düğümler için potansiyel değerler bulunmuştur. Beklenildiği gibi her iki programlama ortamından hemen hemen aynı potansiyel değerler elde edilmiştir. Örneğin 10 numaralı düğümde potansiyel değeri 71.45 [V] olarak bulunmuştur. Tüm düğümlerdeki potansiyel değerleri şekil 5.4'te bir tür bar chart gibi kabul edilebilecek bir gösterim ile verilmiştir. Bu potansiyel değerler her bir eleman (element) için heaplanarak şekil 5.22'te gösterildiği üzere renklendirilmişlerdir. Buna göre kırmzıdan maviye doğru azalan bir dağılım söz konusudur.



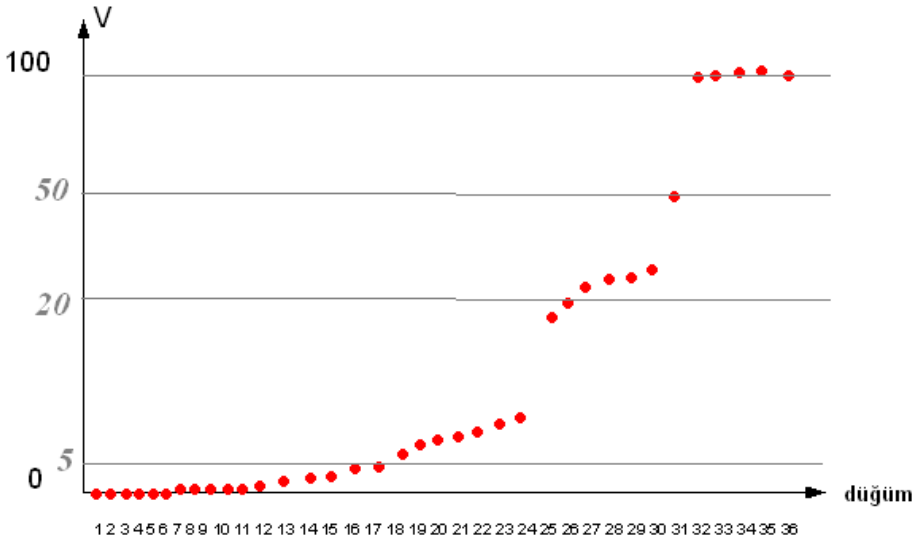
Şekil 5.2) İletkenler Arası Potansiyel Dağılım Sonucu

Ayrıca potansiyel değerlerinin yanısıra elektrik alan vektörleri de her bir sonlu elaman için elde edilebilir. Örneğin dielektrik bölgenin içinde, 17 nolu üçgen elemanda elektrik alan

vektörünün mutlak değeri (magnitude) 5.40 olarak bulunmuşken, dielektriğin hemen dışında yer alan 16 nolu elemanda bu değer 9.44 olarak hesaplanmıştır. Elektrik alan vektörleri aşağıda şekil 5.3'te gösterilmiştir. Buna göre özellikle dilektrik bölge civarında alan vektörlerinde hafif sapmalar göze çarpmaktadır.



Şekil 5.3) Elektrik Alan Vektörleri ve Eş-potansiyel çizgiler



Şekil 5.4) Düğümlerdeki potansiyel değerler

5.1) Değerlendirme ve Sonuç

Aynı değerlerde sonuç üreten iki program, ek bölümde verilen kodları ve performans özellikleri itibarıyla oldukça farklıdırlar. Tam da genel beklentilere uygun olarak C diliyle yazılan kod, MATLAB koduna göre oldukça uzundur. Örneğin boş ve yorum satırlarını çıkardıktan sonra C kodu yaklaşık 200 satırı bulurken, MATLAB kodu yaklaşık 60 satır civarında satırla yazılmıştır. Bir script dili olmasına karşın, bu durum, program geliştirmede MATLAB'ın oldukça etkin bir ortam olduğunun kanıtıdır. Buna karşın performansları göz önüne alındığında, C dili ile yazılan program hemen hemen CPU time olarak saptanamayacak bir hızla çalışmışken MATLAB programı cputime komutuyla 0.0375 CPU zamanı ile çalışmıştır.

Asında bu yukarıda verilen sonuçlar, Sonlu Elemanlar programlamasındaki ölçütleri de ortaya koymaktadır. Buna göre eğer performansın gerçekten en önemli parametre olduğu uygulamalarda, sonlu elemanlar programcısının fazla bir seçeneği olmayıp C veya C++ ile kodu yazmalıdır. Burdaki uygulamada olduğu gibi eğer performans yerine görsellik ve kolay kod yazımı söz konusu ise MATLAB mevcut seçenekler arasında en doğru olanıdır.

Programcılar için sadece MATLAB ve C seçeneği dışında da Sonlu Elemanlar Yöntemi uygulamasında alternatif programlama ortamları da mevcuttur. Son yıllarda örneğin Java ile geliştirilen sonlu elemanlarla çözüm kodlarına literatürde rastlanılmaktadır. Belki yakın gelecekte web programlamada göz önüne alınınca ve Java 3-boyutlu grafik paketinin de etkinleştirlmesi sonucu Java dili de ciddi bir seçenek olarak sonlu elemanlar programcılarının kullanımında ortaya çıkacaktır.

6. Kaynaklar

[1] Cheng, D.K. Field and Wave Electromagnetics. Addison-Wesley, 2001

[2] R. L. Courant, "Variational methods for the solution of problems of equilibrium and vibration", *Bulletin of the American Mathematical Society*, vol. *49,* pp. *1-23,* 1943.

[3] M.J. Turner, R.W. Clough, H.C. Martin and L.C. Topp, "Stiffness and Deflection Analysis of Complex Structures"*,* J*ournal of Auronautical Science*, vol. *23,* pp. *507-510,* 1956.

[4] R.W. Clough, "The finite method in plane stress analysis", *Proc. The 2.nd ASCE conf. on electronic computation*, Pittsburgh, Pennsylvania

[5] O. C. Zienkiewicz, "*The Finite Element Method*", McGraw-Hill, 1967

[6] W. Ritz, "Über eine neue methode zur losung gevissen Variations" – probleme der mathematischen physik", *J.Reine Angew. Math.* ", 135, pp. 1-61, 1909

[7] B.G. Galerkin, "Series solution of some problems of elastic Equilibrium of rods and plates", *Vestn. Inzh. Tech.*, 19, pp.897-908, 1915

[8] A Biran, M. Breiner, "*MATLAB 6 for Engineers*", Prentice-Hall, 2002.

[9] B. Kernighan, D. Ritchie, "*The C Programming Language"*, Prentice Hall, 1978.

EK - A
input.txt yapısı

36 ”Toplam Eleman Sayısı”
50 “Toplam Düğüm Sayısı”
2 “Toplam Bölge Sayısı”
2 “Birbirinden Farklı Bölge Sayısı”

Elemanların Düğümleri			Bölge No.	Eleman No.
1	8	2	1	1
1	7	8	1	2
2	9	3	1	3
2	8	9	1	4
3	10	4	1	5
3	9	10	1	6
4	11	5	1	7
4	10	11	1	8
5	12	6	1	9
5	11	12	1	10
7	14	8	1	11
7	13	14	1	12
8	15	9	1	13
8	14	15	1	14
9	16	10	1	15
9	15	16	1	16
10	17	11	2	17
10	16	17	2	18
11	18	12	1	19
11	17	18	1	20
13	20	14	1	21
13	18	20	1	22
14	21	15	1	23
14	20	21	1	24
15	22	16	2	25
15	21	22	2	26
16	23	17	2	27
16	22	23	2	28
17	24	18	1	29
17	23	24	1	30
19	26	20	1	31
19	25	26	1	32
20	27	21	1	33
20	26	27	1	34
21	28	22	1	35
21	27	28	1	36
22	29	23	1	37
22	28	29	1	38
23	30	24	1	39
23	29	30	1	40
25	32	26	1	41
25	31	32	1	42
26	33	27	1	43
26	32	33	1	44
27	34	28	1	45
27	33	34	1	46
28	35	29	1	47
28	34	35	1	48
29	36	30	1	49
29	35	36	1	50

Düğümlerin Koordinat Değerleri

x	y
0.	10.
2.	10.

4.	10.
6.	10.
8.	10.
10.	10.
0.	8.
2.	8.
4.	8.
6.	8.
8.	8.
10.	8.
0.	6.
2.	6.
4.	6.
6.	6.
8.	6.
10.	6.
0.	4.
2.	4.
4.	4.
6.	4.
8.	4.
10.	4.
0.	2.
2.	2.
4.	2.
6.	2.
8.	2.
10.	2.
0.	0.
2.	0.
4.	0.
6.	0.
8.	0.
10.	0.

100	“Üst Plakanın Potansiyel Değeri”
1 2 3 4 5 6	“Üst Plakanın Sabit Potansiyelli Düğümleri”
0	“Alt Plakanın Potansiyel Değeri”
31 32 33 34 35 36	“Alt Plakanın Sabit Potansiyelli Düğümleri”

<u>Bölge Özellikleri</u>

Bölge ID	ε_r
1	1.
2	5.

EK – B :
C-Dilinde kodlanmış örnek program
1) Input.txt'yi oluşturan kod

```
#include <stdio.h>
#include <stdlib.h>

#define MAX_BUFFER_LEN    1024

int main(int argc, char *argv[])
{
   int i;
   int j;
   int xeks;
   int yeks;
   int ToplamNodeSayisi;
   int ToplamElemanSayisi;
   int node[MAX_BUFFER_LEN];
   int HesaplanacakNodeSayisi;
   int ElemanBolgesi;
   int ElemanSayisi;
   int xLong;
   int yLong;
   int twice;
   int ElemanCounter;
   float xKoor;
   float yKoor;
   char ch;
   char tmpStr[10];
   char buffer[MAX_BUFFER_LEN];
   FILE *fp;

 ch = 0x00;
 i  = 0;

 printf ("x - eksenindeki node sayisi :\n");
 scanf ("%d" , &xeks);
 printf ("y - eksenindeki node sayisi :\n" );
 scanf ("%d",&yeks);
 printf ("x-ekseni uzunlugu :\n");
 scanf ("%d", &xLong);
 printf ("y-ekseni uzunlugu :\n");
 scanf ("%d", &yLong);
 printf ("kac eleman girisi yapilacak?");
 scanf ("%d", &ElemanCounter);


 ToplamNodeSayisi       = xeks * yeks;
 HesaplanacakNodeSayisi = xeks * (yeks - 1);
 ToplamElemanSayisi     = 0;

 while (i<ElemanCounter){
  printf ("elemanlar:\n");
  scanf  ("%d" , &node[i]);
  fflush(stdin);
  sprintf (tmpStr , "%x" ,node[i]);
  if (tmpStr[0] <= 0x30 || tmpStr[0] > 0x39){
     printf ("we'll prepare file:\n");
     node[i] = 0x00;
     break;
  }

  i++;
  node[i] = 0x00;
 }
```

```
  // prepare input file
  ElemanSayisi = 0;
  fp=fopen("C:\\input.txt","w");
  memset (buffer , '\x00' , MAX_BUFFER_LEN);

  for (i = 1 ; i <= HesaplanacakNodeSayisi ; i++){
     if ( (i % xeks ) == 0){
       continue;
     }
     ToplamElemanSayisi++;
     ToplamElemanSayisi++;
  }

  sprintf (buffer ,"%d %d %d %d" ,ToplamNodeSayisi, ToplamElemanSayisi , 2, 2);//buradaki ilk 2 condition diger 2
material sayisidir!!!
  fwrite (buffer , 1 , sizeof(buffer) , fp);

  for (i = 1 ; i <= HesaplanacakNodeSayisi ; i++){
     if ( (i % xeks ) == 0){
       continue;
     }
     ElemanBolgesi = 1;
     for (j = 0 ; j < ToplamNodeSayisi ; j++){
        if (node[j] == ElemanSayisi+1){//i){
          ElemanBolgesi = 2;
        }
     }
     memset (buffer , '\x00' , MAX_BUFFER_LEN);
     sprintf (buffer , "%d %d %d %d %d",i ,i + xeks + 1 , i + 1 , ElemanBolgesi , ElemanSayisi+1);

     ElemanSayisi++;
     fwrite (buffer , 1 , sizeof(buffer) , fp);

     ElemanBolgesi = 1;
     for (j = 0 ; j < ToplamNodeSayisi ; j++){
        if (node[j] == ElemanSayisi+1){//i){
          ElemanBolgesi = 2;
        }
     }
     memset (buffer , '\x00' , MAX_BUFFER_LEN);
     sprintf (buffer , "%d %d %d %d %d",i ,i + xeks , i + xeks + 1 , ElemanBolgesi , ElemanSayisi+1);

     ElemanSayisi++;
     fwrite (buffer , 1 , sizeof(buffer) , fp );

  }

  for (xKoor = 0 ; xKoor <= xLong ; xKoor += (float)(xLong / (xeks - 1))){
     for (yKoor = 0 ; yKoor <= yLong ; yKoor += (float)(yLong / (yeks - 1))){
        memset (buffer , '\x00' , MAX_BUFFER_LEN);
        sprintf (buffer , "%f %f" , xKoor , yKoor);
        fwrite (buffer , 1 , sizeof (buffer) , fp);
     }
  }

  twice = 0;
  while (twice < 2) {
     i = 0;
     while (1){
       printf ("fix nodelar :\n");
       scanf  ("%d" , &node[i]);
       fflush(stdin);

       sprintf (tmpStr , "%d" ,node[i]);
       if (tmpStr[0] <= 0x30 || tmpStr[0] > 0x39){// 0x1B || tmpStr[0] == 'c'){
         //printf ("we'll prepare file:\n");
         node[i] = 0x00;
```

```
        break;
      }
      if (i >= xeks - 1){
        i++;
        break;
      }

      i++;
      node[i] = 0x00;
    }

    printf ("bu nodelarin degeri :\n");
    scanf  ("%d" , &node[i]);
    fflush(stdin);

    memset  (buffer , '\x00' , MAX_BUFFER_LEN);
    sprintf (buffer ,"%d" ,node[i]);
    fwrite  (buffer , 1 , sizeof (buffer) , fp);

    for (j = 0 ; j < i ; j++){
       sprintf (tmpStr , "%d" , node[j]);//(j+1)
       if (j == 0){
         strcpy (buffer ,tmpStr);
       }
       else {
         strcat (buffer , tmpStr);
       }
       strcat (buffer , " ");
    }
    fwrite  (buffer , 1 , sizeof (buffer) , fp);
    twice++;
  }

  getch ();
  fclose (fp);

  //system("PAUSE");
  //return 0;
}
```

2)Solver

```
#include <iostream>
#include <stdio.h>
#include <math.h>

using namespace std;
/* Degisken bilgileri

  NNO:      Nod sayisi
  NEL:      Eleman sayisi
  NCON:     Potansilleri bilinen sinir sayisi
  NMAT:     dielectrik malzeme sayisi
  KTRI(NEL,3): Her bir eleman icin node sayilari
  MAT(NEL):   her eleman icin malzeme tipi
  PERM(NMAT): Her eleman icin Permeability degeri
  X(NN0):    Nod  x coordinatlari
  Y(NN0):     Nod y coordinatlari
  VI(NCON,20): Imposed potansiyel degerleri
  SS(NNO,NNO): Stiffness Matrisi
  VV(NNO):    Bilinmiyen potansiyel matrisi
  VDR(NNO):  Sag taraf matrisi
*/
```

```
void sifirlama();
void stiffness();
void sinir( );
void gauss();
void output();
void grad(int,int, int );

//Global Degiskenler
int KTRI[200][3], MAT[200],NOCC[10][20];
float X[150],Y[150],PERM[10],VI[10];
double SS[150][150],VV[150],VDR[150];
int ELEM[200], NMATERIAL[10];
//parametreler
int NNO;
int NEL;
int NCON;
int NMAT;
//Hesaplanan Degiskenler
int n1,n2,n3;
double q1,q2,q3,r1,r2,r3;
double EX,EY;

FILE *fp,*fp2;

int main( )
{
 sifirlama( );

 input( );

 stiffness();

 sinir( );

 gauss();

 output();

 fclose(fp2);
 system("PAUSE");
 return 0;
}


void sifirlama()
{
 int i,j;
 for(i=0; i<=NNO; i++)
   {
    VV[i]=0;
    VDR[i]=0;
    for (j=0;j<=NNO; j++)
      SS[i][j]=0;
   }
   for(i=0; i<=NCON; i++)
     for(j=1; j<=20; j++)
      NOCC[i][j]=0;
}
void input()
{
 int i,j;
 FILE *fp;
 fp=fopen("input.txt","r");

 fscanf(fp, "%d %d %d %d",&NNO,&NEL,&NCON,&NMAT);
```

```
  //Mesh yapisi
  for(i=1;i<=NEL;i++)
    fscanf(fp,"%d%d%d%d%d",&KTRI[i][1],&KTRI[i][2],&KTRI[i][3],&MAT[i],&ELEM[i]);

  // koordinatlar
  for(i=1;i<=NNO;i++)
    fscanf(fp,"%f%f",&X[i],&Y[i]);

// sinir kosullari
  for(i=1;i<=NCON;i++){
    fscanf(fp,"%f",&VI[i]);
    for (j=1;j<=6; j++)
      fscanf(fp, "%d",&NOCC[i][j]);

  }

//malzeme dielektrik degerleri
  for(i=1;i<=NMAT; i++)
       fscanf(fp,"%d %f",&NMATERIAL[i],&PERM[i]);

    fclose(fp);
}

void stiffness()
{
  int i,j,k;
  int kk,jj;
  int naux[4];
  double s[4][4];
  double xperm;
  double det,coeff;
  int nm;

fp2=fopen("output.txt", "w");

  for(i=1; i<=NEL; i++)
  {
    n1=KTRI[i][1];
    n2=KTRI[i][2];
    n3=KTRI[i][3];
    nm=MAT[i];

    q1=Y[n2]-Y[n3];
    q2=Y[n3]-Y[n1];
    q3=Y[n1]-Y[n2];
    r1=X[n3]-X[n2];
    r2=X[n1]-X[n3];
    r3=X[n2]-X[n1];

    xperm=PERM[nm];

    det=X[n2]*Y[n3]+X[n1]*Y[n2]+X[n3]*Y[n1]-X[n1]*Y[n3]-X[n3]*Y[n2]-X[n2]*Y[n1];
    coeff=xperm/(2*det);

    s[1][1]=coeff*(q1*q1+r1*r1);
    s[1][2]=coeff*(q1*q2+r1*r2);
    s[1][3]=coeff*(q1*q3+r1*r3);
    s[2][1]=s[1][2];
```

```
   s[2][2]=coeff*(q2*q2+r2*r2);
   s[2][3]=coeff*(q2*q3+r2*r3);
   s[3][1]=s[1][3];
   s[3][2]=s[2][3];
   s[3][3]=coeff*(q3*q3+r3*r3);

 // s[3][3] eleman matrisinin SS[NNO][NNO] genel matrisin icine aktarimi
   naux[1]=n1;
   naux[2]=n2;
   naux[3]=n3;
   printf("naux[1]=%d  naux[2]=%d  naux[3]=%d\n",naux[1],naux[2],naux[3]);
   for(k=1; k<=3; k++)
   {
     kk=naux[k];
     for(j=1; j<=3; j++)
       {
        jj=naux[j];
        SS[kk][jj]=SS[kk][jj]+s[k][j];

       }
   }

 }

 }

 void sinir()
 {
   int i,j,k;
   int NOX;

   for(i=1; i<=NCON; i++)
  {
     for (j=1; j<=10; j++)
    {
       NOX=NOCC[i][j];
       if(NOX==0)  goto out1;

       for(k=1; k<=NNO; k++)
           SS[NOX][k]=0;

       SS[NOX][NOX]=1;
  // imposed potentiseller sag tarafa aktariliyor
       VDR[NOX]=VI[i];

    }
   out1: printf("*\n");
 }
}

void gauss()
{
  int i, m, j;
  int nn;
  double fact;
```

```
 double sum;

  nn=NNO-1;
   for(i=1; i<=nn; i++){
   for (m=i+1; m<=NNO; m++){
    fact=SS[m][i]/SS[i][i] ;
    VDR[m]=VDR[m]-VDR[i]*fact;
    for(j=i+1; j<=NNO; j++)   SS[m][j]=SS[m][j]-SS[i][j]*fact;
    }
   }
    VV[NNO]=VDR[NNO]/SS[NNO][NNO] ;

    for (i=nn; i>=1; i--)
    {
       sum=0;
       for(j=i+1; j<=NNO; j++)  sum=sum+SS[i][j]*VV[j];

       VV[i]=(VDR[i]-sum)/SS[i][i] ;
    }

}

void grad(int N1, int N2, int N3)
{
    double Q1,Q2,Q3,R1,R2,R3;
    double DET;


    Q1=Y[N2]-Y[N3];
    Q2=Y[N3]-Y[N1];
    Q3=Y[N1]-Y[N2];

    R1=X[N3]-X[N2];
    R2=X[N1]-X[N3];
    R3=X[N2]-X[N1];

    DET=X[N2]*Y[N3]+X[N1]*Y[N2]+X[N3]*Y[N1]-X[N1]*Y[N3]-X[N3]*Y[N2]-X[N2]*Y[N1];
    EX=-(Q1*VV[N1]+Q2*VV[N2]+Q3*VV[N3])/DET;
    EY=-(R1*VV[N1]+R2*VV[N2]+R3*VV[N3])/DET;
}
void output()
{
    int i;
    int n1,n2,n3;
    double EM;
   for(i=1; i<=NNO; i++)
    printf("Node: %d \t Potential:%f\n",i,VV[i]);

//elemanlardaki elektrik alan degerleri ciktisi

   for(i=1; i<=NEL; i++)
   {
    n1=KTRI[i][1];
    n2=KTRI[i][2];
    n3=KTRI[i][3];
    grad(n1,n2,n3);

    EM=sqrt(EX*EX+EY*EY);
    printf("Element %d \t EM=%f\n",i,EM);
    }
}
```

EK – C :

MATLAB örnek program

```
%2D Electrostatik Laplace Denkleminin FE yontemiyle cozumu
%Ucgen elemanlar kullanildi
% ND:   Nod sayisi
% NE:   Eleman sayisi
% NP:   fixed nod sayisi
% NDP(I):   fixed potansiyelnodlari
% VAL(I):   Fixed nodlarda bilinen potansiyel degerleri
% NL(I,J):  Eleman nod numaralari
% CE(I,J):  Eleman katsayilar matrisi
% C(I,J):   Global katsayi matrisi  (stiffness matris)
% B(I)=     Sag taraf matrisi
% X(I),Y(I): Her bir nod icin global koordinatlar
% XL(J), YL(J): Her bir node icin lokal koordinatlar
% V(I): Potansiyeller
%.....................................................
%  Ilk adim: input data
%.....................................
clear
input('input dosyasini gir:')

eval('load elements3_Square.dat; elements3_Square(:,1)=[]; ','elements3_Square=[];');
%.....................................
% Ikinci adim: Her bir eleman icin matris elemanlarini hesapla ve genel matrise yerlestir
%.....................................................
B=zeros(ND,1);
C=zeros(ND,ND);
time1=cputime;
for I =1:NE
   %Lokal koordinatlari ata
   K=NL(I,[1:3]);
   XL=X(K);
   YL=Y(K);

   P=zeros(3,1);
   Q=zeros(3,1);
   P(1)=YL(2)-YL(3);
   P(2)=YL(3)-YL(1);
   P(3)=YL(1)-YL(2);
   Q(1)=XL(3)-XL(2);
   Q(2)=XL(1)-XL(3);
   Q(3)=XL(2)-XL(1);

   AREA=0.5*abs( P(2)*Q(3) - Q(2)*P(3));
%Eleman I icin katsayiyi hesapla
   CE=(P*P'+Q*Q')/(4*AREA);
%stiffness matrisi global olarak olustur
   for J=1:3
      IR=NL(I,J);
      IFLAG1=0;
%satirin fixed potansiyel satiri olup olmadigini kontrol et
     for K=1:NP
      if(IR==NDP(K))
         C(IR,IR)=1.0;
         B(IR)=VAL(K);
         IFLAG1=1;
      end
     end

 if(IFLAG1==0)
  for L=1:3
      IC=NL(I,L);
      IFLAG2=0;
%sutunun fixed potansiyel sutunu olup olmadigini kontrol et
   for K=1:NP
    if(IC==NDP(K))
```

```
        B(IR)=B(IR)-CE(J,L)*VAL(K);
        IFLAG2=1;
      end
    end
    if(IFLAG2==0)
      C(IR,IC)=C(IR,IC)+CE(J,L);
    end
   end %for L=1:3
  end  %for if IFLAG1==0
  end %for J=1:3
end  %for I=1:NE

%*****************************************************************
%  Ucuncu adim - Solver
%*****************************************************************
V=inv(C)*B;
input('name of input data:')
V=VP'
%V, contour ve vektor cizimleri icin, yeniden duzenleniyor
k=0;
VV=zeros(6,6);
for i=1:6
    for j=1:6
        k=k+1;
        VV(i,j)=V(k);
    end
end
deltaTime=cputime-time1;
disp(deltaTime);
[Ex,Ey]=gradient(VV,0.2,0.2);
xx=X(1:6);
yy=Y(1:6:36);
[x,y]=meshgrid(xx,yy);

DET=X[N2]*Y[N3]+X[N1]*Y[N2]+X[N3]*Y[N1]-X[N1]*Y[N3]-X[N3]*Y[N2]-X[N2]*Y[N1];
     EX=-(Q1*VV[N1]+Q2*VV[N2]+Q3*VV[N3])/DET;
     EY=-(R1*VV[N1]+R2*VV[N2]+R3*VV[N3])/DET;

%*****************************************************************
%   Dorduncu adim - Sonuclar ve gosterim
%*****************************************************************

V=ones(1,36);
trisurf(elements3_Square,X,Y,V)
view(40,10);
hold on
contour(x,y,VV);
hold on
quiver(x,y,Ex,Ey)
hold off
title('Solution of the Problem')
```

Berna SÜLÜ

Doğum Tarihi : 12 Mayıs 1980
e-mail : bernaslu@gmail.com

Eğitim Bilgileri :

Eğitimi			
Yüksek Lisans	Beykent Üniversitesi	Bilgi Teknolojileri	2006-2009
Lisans	İstanbul Üniversitesi	Matematik	1998-2004
Lise	Çanakkale Milli Piyango Anadolu Lisesi	Fen-Matematik	1991-1998

Çalıştığı Kurumlar :

Ağustos 2009 – Halen : Türk Hava Yolları
GÖREV : Uzman Yazılımcı – DCS
Kullandığı Yazılım Dili : Assembly

Temmuz 2008 – Şubat 2009 : Verisoft
GÖREV : POS yazılım Uzmanı
Kullandığı Yazılım Dili :ANSI C
Edinimler : EMV,ISO 8583

Haziran 2006 – Temmuz 2008 : Ceren Bilgisayar
GÖREV : Analist ve Yazılmcı
Kullandığı Yazılım Dili : c/c++
Geliştirme Ortamı : Debian Tabanlı Linux üzerinde Eclipse,QT