PAMUKKALE ÜNİVERSİTESİ * FEN BİLİMLERİ ENSTİTÜSÜ
İNŞAAT MÜHENDİSLİĞİ ANABİLİM DALI

SONLU ELEMANLAR YÖNTEMİ İLE ÜÇ BOYUTLU YAPI ANALİZİ YAPAN BİR BİLGİSAYAR PROGRAMI

Pamukkale Üniversitesi Fen Bilimleri Enstitüsüne
"İnşaat Yüksek Mühendisi"

Ünvanı verilmesi için kabul edilen tezdir.

Tezin enstitüye verildiği tarih :
Tezin sözlü savunma tarihi :

Tezin danışmanı : Yrd. Doç. Dr. Hasan KAPLAN

Jüri Üyesi : Yrd. Doç. Dr. Abdurrahman ŞİMŞEK

Jüri Üyesi : Yrd. Doç. Dr. Muzaffer TOPÇU

Enstitü Müdürü : Prof.Dr.Hikmet RENDE

EYLÜL - 1996
DENİZLİ

Şevket Murat ŞENEL'in YÜKSEK LİSANS tezi olarak hazırladığı "Sonlu Elemanlar Yöntemi İle Üç Boyutlu Yapı Analizi Yapan Bir Bilgisayar Programı" başlıklı bu çalışma, jürimizce lisansüstü yönetmeliğinin ilgili maddeleri uyarınca değerlendirilerek kabul edilmiştir.

09 / 09 / 1996

Üye : Yrd.Doç.Dr. Hasan KAPLAN

Üye : Yrd.Doç.Dr. Abdurrahman ŞİMŞEK

Üye : Yrd.Doç.Dr. Muzaffer TOPÇU

Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 13.09.1996 gün ve 13/15 sayılı kararıyla onaylanmıştır.

Prof.Dr. Hikmet RENDE
Enstitü Müdürü

ÖNSÖZ

Bu tez çalışması Pamukkale Üniversitesi Fen Bilimleri Enstitüsü İnşaat Mühendisliği Anabilim Dalı Yüksek Lisans Programında Yapılmıştır.
Sonlu elemanlar yöntemi ile elde edilen formülasyon ve matrisleri kullanarak, çubuk sistemlerden oluşan üç boyutlu yapı sistemlerinin statik analizine imkan veren bir yazılım geliştirilmiştir. Görsel programlama tekniği kullanılarak geliştirilen yazılımda verilerin ve çıktıların "windows" ortamında izlenebilmesi sağlanmıştır.

Hazırlanan bu yazılım ile lisans ve lisans üstü seviyelerde matris metotları ile yapı sistemlerinin statik analizi konusuna kolaylıklar getirmek ve çözümleme metodunun anlaşılabilirliğine katkıda bulunmak hedeflenmiştir.

Bu Çalışmada beni yönlendiren tez danışmanım Y.Doç.Dr. Hasan KAPLAN'a, PAÜ Mühendislik Fakültesi İnşaat Mühendisliği Bölüm Başkanı Prof.Dr. İbrahim ALYANAK 'a, Y.Doç.Dr. Abdurrahman ŞİMŞEK 'e, Y.Doç.Dr. Muzaffer TOPÇU 'ya ve çalışma boyunca katkılarını esirgemeyen bütün mesai arkadaşlarıma teşekkür ederim.

ÖZGEÇMİŞ

1970 yılında Denizli'de doğdu. İlk ve orta öğrenimini Denizli'de tamamladı ve 1987 yılında Denizli lisesinden mezun oldu. 1993 yılında Orta Doğu Teknik Üniversitesi Mühendislik Fakültesi İnşaat Mühendisliği Bölümünü bitirdi. Mezuniyetten sonra bir sene serbest piyasada çalıştı. 1994 yılında Pamukkale Üniversitesi Mühendislik Fakültesi İnşaat Mühendisliği Bölümünde yüksek lisans çalışmasına başladı. 1995 yılında da aynı bölümde araştırma görevlisi olarak göreve başladı. Halen bu görevine devam etmektedir.

İÇİNDEKİLER **Sayfa**

ŞEKİL LİSTESİ

ÖZET

Son otuz yıldır özellikle inşaat mühendisliği alanında yaygın bir şekilde kullanılan sonlu elemanlar yöntemi, her türlü yapının tasarımına ve çözümüne imkan veren bir sayısal analiz yöntemidir.

Yapı mühendisleri tarafından karmaşık ve hesaplanması uzun problemlerin çözümünde geleneksel yöntemlerin yerine, her geçen gün daha sıkça kullanılan bu yöntemin teorisi üzerine son yıllarda önemli çalışmalar yapılmıştır. Yöntemin bilgisayar ile programlamaya uygun oluşu nedeni ile popülaritesi de artmıştır.

Bu çalışmada esas amaç, sonlu eleman formülasyonunu kullanarak çözümleme yapan bir yazılım geliştirerek, üç boyutlu yapıların statik analizini yapabilmektir. Hazırlanan yazılımın getirdiği en büyük kolaylık, diğer yazılımlardan farklı olarak çözümleme boyunca hesaplanan eleman ve sistem matrislerinin, kodlama tablosunun kullanıcı tarafından kontrol edilebilmesini sağlamaktır. Bu yolla kullanıcının, yöntemin teorisini anlayıp çözümleme işlemini adım adım gözleyebilmesi hedeflenmiştir.

Görsel programlama tekniği kullanılarak geliştirilen yazılım ile çeşitli yapı sistemlerinin çözümlemeleri yapılmıştır. Elde edilen sonuçlar SAP90 programının verdiği sonuçlar ile karşılaştırılmış ve tam bir uyumluluk olduğu görülmüştür.

ABSTRACT

Finite element method which is used especially in civil engineering area for thirty years is a numerical analysis method which can serve the design and solution of every kind of structure.

Important resarches are developed about the theory of this method by structural engineers who are prefer to use finite element method in the problems that are difficult and need much more time for solution. Because of applicability for computure programming, the popularity of method is inreased.

In this work it`s aimed that to achive statical analysis of three dimensional structures by improving a software which use finite element formulations. Different from other ones, by using this software user can control the code table, element and system matrices which are calculated during solution. By that way it`s aimed that, user can understand the theory of method and can control the solution procedure step by step.

Some of structural systems solved by the program which is prepared by using visual programming technic. Results are compared with the SAP90 solutions and it is seen that compatibility exist between the solutions of programs.

1. GİRİŞ

Yüzyılımızın ortalarına doğru bilgisayar ve elektronik hesap makinaları konusunda pek çok ilerlemeler sağlanmıştır. Daha karmaşık ve yoğun pek çok hesaplamanın çok daha doğru ve hızlı bir çözümü bu sayede mümkün hale gelmiştir. Bu ve benzeri pek çok kolaylıklar mühendislere ve araştırmacılara çok daha geniş alanlarda çalışma ve yeni yöntemler üzerinde düşünme imkanı vermiştir. Halihazırda var olan çözüm yöntemlerinin yaklaşık yöntemler oluşu ve bilgisayarla çözümleme için çok uygun olmayışı sebebi ile yeni metotlar üzerinde çalışılmıştır.

Sonlu elemanlar metodu bu araştırmalar neticesinde geliştirilmiş ve genellikle bütün mühendislik alanlarında bilgisayar ile çözümlemeye uygunluğu sebebi ile kullanılmış ve halen de kullanılmaktadır. Metot genel bir yaklaşım ile bütünü sonlu sayıda elemanlara bölmekte ve sistemin genel davranışını sistemi oluşturan elemanların davranışlarının ayrı ayrı süperpozisyonu ile açıklamaya idealize etmektedir. Görüldüğü gibi çözümün doğruluğu için modellemenin doğru yapılması, davranış biçiminin gerçeğe en yakın şekilde temsil edilmesi gerekmektedir. Sonlu elemanlar yönteminin uygulanışı içerdiği uygulamalar yönünden üç ana gurup olarak ele alınabilir. Bunlar matris ve lineer uygulamalar, seçilen sonlu elemen. formülasyonu ve statik ve dinamik analizde sonlu eleman denge denklemlerinin etkili bir biçimde çözümü için nümerik metotlar şeklinde özetlenebilir.

Doğru şekil fonksiyonlarının seçilmesinden sonra, sonlu elemanlar yöntemini kullanarak rijitlik ve yük matrislerinin formülasyonunu elde etmek mümkün olmuştur. Rijitlik matrisi yöntemi de elde edilen bu formülasyon ve matris uygulamalarının bir prosedür olarak ele alınması ve yapı sistemlerinin çözümüne uygulanmasından ibarettir. Yapı sisteminin genel şekli nasıl olursa olsun, yapıyı oluşturan bütün elemanların davranışları uygun şekil fonksiyonları ile temsil edilebildiği ve rijitlik ve yükleme matrisleri formüle edilebildiği için yapı siteminin çözümü, bu standart prosedürün uygulanması ile kolaylıkla yapılabilmektedir. Kolaylıkla anlaşılabileceği gibi böyle bir işlem bilgisayar ile çözümlemeye son derece uygundur.

Dünya üzerinde kırk seneyi aşkın bir süredir konu üzerinde pek çok çalışmalar yapılmıştır. SAP4, SAP80, SAP90, ETABS, SAFE, vb. programları sonlu eleman formülasyonu esas alınarak geliştirilmiş ve tüm dünyada yaygın olarak kullanılan yazılımlardır. Bu tür yazılımlar yardımı ile çubuk elemanların, kabukların, elastik zeminlerin modellemesi yapılabilmektedir. Her eleman için rijitlik matrislerini hesaplayan bu programlar, sistemin tamamını temsil eden genel rijitlik matrisini teşkil ederek yapı siteminin çözümünü yapmaktadırlar. Çoğunlukla ticari amaçlar gözetilerek hazırlanan bu yazılımlar ülkemizde de yaygın biçimde kullanılmaktadır. PROBİNA, STA4CAD, BABALIOĞLU, IDECAD bu tür yazılımlara örnek olarak gösterilebilir. Kullanıcı bu programlarda ihtiyacı olan kuvvet, moment ve deplasmanlara ulaşarak, çözümleme işlemini tamamlamaktadır. Fakat çözümleme işlemi sırasında program tarafından hesaplanan eleman ve sistem matrislerini görememektedir.

Bu kapsamda yapılan çalışmada benzer şekilde üç boyutlu çubuk sistemlerin çözümlemesine imkan veren bir yazılım geliştirilmiş, örnek yapı sistemlerinin çözümü yapılarak SAP90 sonuçları ile karşılaştırılmıştır. Programlama tekniği olarak dünya üzerinde henüz çok yeni olan "Görsel Programlama" tekniği kullanılmıştır. Bu teknik sayesinde "Windows" ortamında elektronik tablolara, menülere, mouse ve yazıcı kullanımına ulaşmak daha kolay bir biçimde mümkün olabilmektedir. Bu ve benzeri imkanlar kullanarak hazırlanan yazılım, SAP90 vb. yazılımlardan farklı olarak sistem ve eleman matrislerine, kodlama tablosuna doğrudan erişim sağladığı için kullanıcıya yöntemin bütün adımlarını görme imkanını sunmaktadır.

Bu ve benzeri yazılımların etkin ve yaygın olarak kullanımları sayesinde daha gerçekçi çözümlere ulaşmak mümkün olacaktır. Ayrıca bilgisayar üzerinde hesaplamanın verdiği avantajlardan faydalanacak olan kullanıcı bu doğru sonuçlara çok daha kısa sürede ulaşabilecektir. Benzer yazılımların lisans düzeyinde eğitim amacı ile kullanımı yoluyla piyasanın sorunlarına cevap verebilecek yeterlilikte mühendislerin yetiştirilmesi de mümkün olabilecektir.

2. SONLU ELEMANLAR YÖNTEMİ

2. 1. Giriş

Sonlu elemanlar yöntemi; malzeme özelliklerinin çeşitliliğini ve sınır koşullarının farklılığını güçlükle karşılaşmadan bağdaştırabilen, kullanımı kolay çok yönlü sayısal analiz yöntemlerinden biridir. İnşaat mühendisliği alanında bir çok problemi çözmede sonlu elemanlar yöntemi kullanılmaktadır. Ayrıca lineer ve non-lineer malzeme özelliklerini aynı anda problemde kullanılabilen, toplam ve efektif gerilme durumunu verecek şekilde formüle edilmiş bir analiz yöntemidir. Uygulayıcı mühendise büyük bir pratiklik sağlamakta, dikkat, bilgi ve tecrübe ile kullanıldığında kolaylıkla her probleme uygulanabilmektedir.

Sonlu elemanlar Yöntemi her türlü yapının tasarımında ve çözümünde, inşaat mühendisliği problemlerinde gittikçe artan bir şekilde kullanılmaktadır. Sayısal faktörleri çözüme katmaya olanak sağlayan, zemin-yapı etkileşimini de dikkate alan bir analiz yöntemidir. Ancak etkili bir çalışma ortamının sağlanması için pahalı teknik donanım gerektirmesi, her uygulamacının kolayca erişebileceği, kullanabileceği bir yöntem olmaması ve teorik güçlüklerinden dolayı günümüz için çok yaygın bir kullanımından söz edilememektedir.

Son yirmi yılda sonlu elemanlar yöntemi gerek teorik, gerekse uygulama alanında büyük gelişmeler göstermiştir. Matematiksel karakteristikleri tümüyle kurulmamış olan sonlu elemanlar yönteminin matematiksel temelini kurmak için son yıllarda önemli çalışmalar yapılmış ve çalışmalar bu konu üzerinde yoğunlaştırılarak geliştirilmiştir.

Yapı mühendisleri karmaşık problemlerin çözümünde geleneksel çözümleri kullanmak yerine sonlu elemanlar yönteminin faydalarını görerek bu yönteme gereken önemi vermiş ve yöntem her geçen gün daha fazla sayıda mühendis, uygulamacı tarafından kullanılır olmuştur.

Bu bölümde sonlu elemanlar yöntemi teorisinin kısa bir özeti verilerek yapı mühendisliğinde karşılaşılan problemlerin sonlu elemanlarla çözüm tekniği tanıtılacaktır.

Sonlu elemanlar metodunun ana prensipleri anlatılacak ve yöntemi daha iyi anlamak için gerekli olan önemli konulara değinilecektir.

2.2. Sonlu Elemanlar Yönteminin Tarihsel Gelişimi

Sonlu elemanlar kavramının fiziksel bir anlamı olduğu için yüzyıllardan beri çeşitli şekillerde kullanıla gelmiştir. Kavramın dayandığı ana fikir daima gerçek problemi daha basit olanı ile yer değiştirmektir. Basitleştirilmiş problem çözüldüğünde sonucun gerçek çözümü yeterli, tatmin edici yaklaşıklıkla yansıttığı düşünülür, sonlu elemanların mantığı bu varsayıma dayanmaktadır. Günümüzün modern sonlu elemanlar yöntemi eski zamanlardakinden çok daha karmaşık ve sofistike olmasına rağmen yine de ana mantık, basitleştirilmiş problemi diğeriyle yer değiştirip öyle çözmektir.

Sonlu elemanların ilk uygulaması geometri alanında olmuştur. İki bin yıl önce matematikçiler dairenin alanını ve çevresini hesaplamak için çalışmışlar ve tam çözüme ulaşmak için uzun süre beklemişlerdir. Daire örneğinde, daireyi düzenli poligonlara ayırıp basitleştirilmiş problemi çözmek, sonlu elemanlar yönteminin basit ama önemli karakteristiğini göstermektedir.

Antik kayıtlar, sonlu elemanlar yönteminin yüzlerce yıl önce kullanıldığını göstermektedir. " Ahmes papirüsü" MÖ. 1500 'de Mısırlı matematikçilerin π sayısı yerine $10^{1/2}$ 'yi kullandıklarını ve yine Mısırlıların MÖ. 1800 'de piramidin alanını ve kürenin alanını hesaplamada sonlu elemanlar yöntemini kullandığını göstermektedir. Arşimet 'in katıların hacmini hesaplamada yine sonlu elemanları kullandığı bilinmektedir.

1940 'larda uzay ve uçak mühendisliğinin önem kazanması, daha az malzeme daha mükemmel tasarım, en iyi, en güvenli , en yüksek hıza sahip uçakların imali sebebiyle sayısal yöntemler önem kazanmış ve yaygın olarak kullanılmaya başlanmıştır.

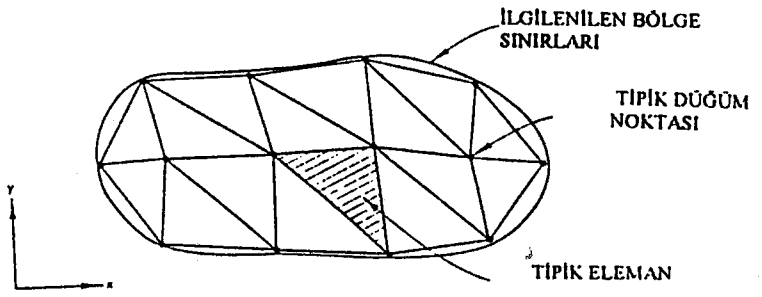
Metot, bilgisayarların gittikçe artan kullanımı sonucunda Langefors ve diğerleri tarafından tanıtılmış olan yapısal analizin matris formülasyonuna dayanmaktadır. 1956 'dan beri bilgisayarlardaki hızlı gelişme, matris yöntemlerinin ve sonlu eleman tekniğinin gelişimine yol açmış ve her geçen yıl sonlu eleman teknikleri ve matris metotlar üzerine yüzlerce bildiri ve yayınlar yayınlanmıştır. Bu konular ile ilgili seminer ve konferanslar düzenlenmiş, kitaplar basılmıştır.

2.3. Sonlu Elemanlar Yönteminin Genel Tanıtımı

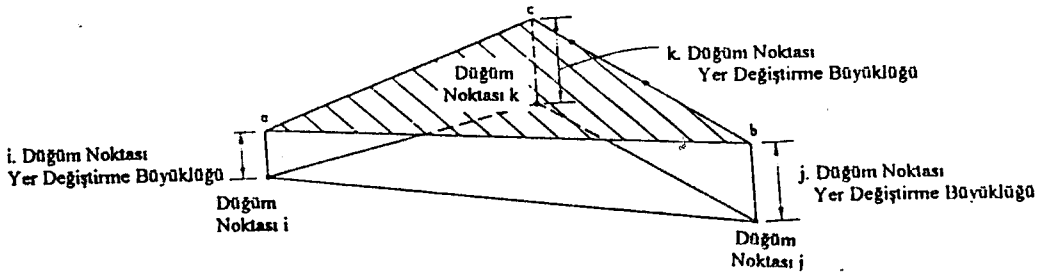
Teorisi ve uygulama alanı ile yapısal analizler için geliştirilmiş sonlu elemanlar yöntemi bugün uygulamalı bilimlerin, sınır ve başlangıç değer problemlerinin yaklaşık çözümü için tercih edilen genel bir metottur. Parça parça interpolasyon fonksiyonları yardımıyla çözümler, varyasyonel problemler veya alternatif olarak ağırlıklı artıklar (weighted residuals) yöntemleri mevcuttur. Esas olarak, "sınır veya başlangıç değer problemi" bölgesinde tanımlanan alt bölge "sonlu eleman" dır.

Sonlu elemanlar yönteminde dört ana konudan söz edilebilir. Bunlar, sonlu elemanlar teorisinin türetilmesi, gerçek problemi idealize ederek sonlu elemanlar problemine dönüştürmek, sonlu elemanlar teorisinin uygulaması için bilgisayar programlarının geliştirilmesi ve sonlu elemanlar çözümü için gerekli olan verilerin ve bilgilerin araştırılmasıdır. Başarılı bir sonlu elemanlar uygulaması için yukarıda sayılan her konu gereklidir. Bu yüzden verimli bir sonlu elemanlar çalışması için tüm bu konuları kapsayan koordineli bir çalışma programı gereklidir.

Özet olarak sonlu elemanlar yönteminin ana ilkesi "sonlu elemanlar" olarak adlandırılan alt bölgelerden oluşan bir cisim, bir yapı kullanmaktır, (Şekil 2.l). Bu elemanlar "düğüm noktaları" denilen kesişen eklemlerden oluşur. Her sonlu eleman için basit fonksiyon olarak gerçek yer değiştirmelerin dağılımları ve değişimleri seçilir ve bunlara "yer değiştirme fonksiyonları" veya "yer değiştirme modelleri" denir. Yer değiştirme modellerindeki bilinmeyen büyüklük, düğüm noktalarındaki yer değiştirmelerdir, (Şekil 2.2). Bir yer değiştirme modeli polinomlar, trigonometrik fonksiyonlar gibi çeşitli basit formüllerle ifade edilmektedir. Polinomlar, matematiksel işlemlerde kolaylık sağladığı için sonlu elemanlar uygulamalarında sıkça kullanılmaktadır.



Şekil 2.1 Üçgen elemanlar topluluğu ile ifade edilen iki boyutlu bölge



Şekil 2.2 Üçgen bir elemanın lineer yer değiştirme modeliyle çizilmiş üçüncü boyutunun izometrik görünüşü

Tüm cisim boyunca denge denklemleri, kesişen düğüm noktalarında korunan yer değiştirmelerin sürekliliği sayesinde, her elemanın denge denklemlerinin kombinasyonundan elde edilir. Bu denklemler sınır koşullarının bilinen değerleri ile modifiye edilir ve bilinmeyen yer değiştirmeleri bulmak için kullanılır. Bir çok problemde çözüm yer değiştirmeler yerine gerilmeler ve şekil değiştirmeler cinsinden istenir. Bu durum zaman zaman fazladan hesaplamalar, işlemler gerektirmektedir.

Sonlu elemanlar yönteminin teorisi iki bölümde incelenebilir. İlk bölüm "tekil elemanları, ikinci bölüm ise tüm cismi temsil eden "elemanlar topluluğunu" inceler. Her iki bölümün teorik ve pratik alandaki uygulamalarına ilerideki konularda değinilecektir. Sonlu elemanlar yönteminde takip edilen sırayı altı ana başlık altında toplayıp incelemek mümkündür. Aşağıda kısaca değinilecek olan bu altı adım yapı mekaniği uygulamaları için geliştirilmiş olmalarına rağmen diğer mühendislik dalları uygulamalarında da aynıdır.

1) Sürekli ortamın bölgelendirilmesi,
2) Yer değiştirme modelinin seçilmesi,
3) Sonlu elemanlar denklemlerinin türetilmesi,
4) Tüm bölge için sonlu eleman denklemlerinin topluluğu
5) Tüm bölge için sonlu eleman denklemlerinin çözümü,
6) Sonlu eleman sonuçlarının hesaplanması (gerilnıe, şekil değiştirme vb.)

Aşağıda sonlu elemanlar yönteminin bazı mühendislik alanlarındaki uygulamalarına dair örnekler verilmiştir.

Çalışma Alanı	Denge Problemleri	Öz Vektör Problemleri	Yayınım Problemleri
• Yapı Mühendisliği, Yapı Mekaniği ve Uzay Mühendisliği	• Kabuk, Kiriş ve Disklerin Analizi • Kompleks ve karmaşık yapıların analizi. • İki ve üç boyutlu gerilme analizi • Prizmatik bölümlerin burulması	• Yapıların stabilitesi • Yapıların titreşim modları ve doğal frekansları • Lineer viskoelastik sönümleme	• Gerilme dalgalarının yayılımı • Yapıların periodik olmayan yüklemelere karşı dinamik davranışı • Viskoelastik problemler
• Zemin Mekaniği, Temel mühendisliği ve Kaya Mekaniği	• Yapı ve kazı problemleri. • Şev stabilitesi problemleri • Zemin yapı etkileşimi. • Barajların, tünellerin analizi	• Zemin- yapı birleşiminin titreşim modları ve doğal frekansları	• Zemin ve kaya ortamlarda değişken hızlı akım durumu. • Zemin ve kaya ortamlarda gerilme dalgalarının yayınımı

Çalışma Alanı	Denge Problemleri	Öz Vektör Problemleri	Yayınım Problemleri
• Hidrodinamik, Hidrolik Mühendisliği ve su kaynakları	• Akışkanların potansiyel akım için çözümü. • Akışkanların viskoz akım için çözümü. • Akiferlerde ve poroziteli ortamlarda düzgün hızlı akış durumu • Barajların ve hidrolik yapıların analizi	• Salınımların doğal periodları ve titreşim modları • Rijit ve bükülebilir konteynerlerdeki akışların çarpması	• Kirlilik dağılım çakışmaları • Çökelti taşınımı • Düzensiz sıvı akımı • Dalga yayınımı • Akiferlerde ve boşluklu ortamlarda zamana bağlı sızma
• Nükleer Enerji Mühendisiği	• Nükleer reaktörlerin bulunduğu yapıların analizi		• Reaktörlerin bulunduğu yapıların dinamik analizi • Reaktör yapılarının termovisko elastik analizi • Reaktör yapılarındaki düzensiz sıcaklık dağılımlarının bulunması

2.4. Yöntemin Avantajları ve Sınırlamaları

2.4.1. Yöntemin Avantajları

Diğer sayısal yöntemler gibi sonlu elemanlar yöntemi de bölgelendirme prensibine dayanmaktadır. Varyasyon ve kalıcı yaklaşım yöntemi vasıtasıyla teknik, cismi yalnızca sürekli ortam gibi idealize etmekle kalmayıp, aynı zamanda sürekli ortam içinde her noktada yaklaşık çözüme ulaşan interpolasyonu da sağlamaktadır. Çözüm, sonlu sayıdaki ayrık düğüm noktalarında elde edilmesine rağmen, alan değişken modelinin formülasyonuyla cisimde tüm diğer yer değiştirmeleri de bulmaktadır. Varyasyonel ve kalıcı yaklaşımlardan farklı olarak, sonlu eleman metodunda ayrık alt bölgelerin kullanımı veya sonlu elemanların kullanımı karmaşık bir geometri gösteren süreklilikler için büyük kolaylık sağlamaktadır.

Sonlu elemanlar yöntemi, yapısal sürekliliklerin sonlu boyutlu ayrık elemanlardan oluşan fiktif bir sistemle yer değiştirilmesiyle çözümüne imkan vermektedir. Bu sistem çerçeve analizinde yaygın olarak kullanılan ve iyi bilinen yer değiştirme modeli vasıtasıyla analiz edilmektedir.

Tanımlanan sınır koşullarının sonlu elemanlar yöntemi denklem topluluğuna dahil edilişi oldukça kolaydır. Sınır koşulları değiştiğinde alan değişken modelinin değişmesine gerek yoktur. Ayıca metot, hiçbir özel tekniğe veya suni metotlara ihtiyaç duymamaktadır. Varyasyonel yaklaşımda yalnızca geometrik sınır koşulları belirlenmelidir.

Sonlu elemanlar metodu yalnız karmaşık geometri ve sınır koşulları tanımlamakla kalmaz, ek olarak diğer sayısal yöntemlerle modellenemeyen çeşitli karmaşık malzeme koşullarını, tiplerini de modelleyebilir. Anizotropik lineer olmayan, zamana bağımlı veya sıcaklığa bağımlı malzeme davranışları gibi katı mekaniğinin bünyesel özelliklerini de formüle edebilmektedir.

Yöntemle her türlü dış yükler modellenebilir. Diğer analitik metotlarla karşılaştırıldığında sınır koşullarının esnekliği avantajını oluşturmaktadır. Anizotropik malzemeleri, yapıları ve heterojenliği formülasyonu sayesinde dikkate almakta ve hesaba katmaktadır.

Sayısal analiz metotlarının uygulamasında zorluklara sebep olan homojen olmayan süreksizliklerin modellenmesinde, teknik, farklı elemanlara farklı malzeme özellikleri vererek kolaylıkla çözüme ulaşmaktadır. İstenilen değişken malzeme karakteristikleri temsil

edilirken, önceden seçilen polinom modeline uygun olacak şekilde, eleman içinde özelliklerin değişimini sağlamak mümkündür. Bünyesel parametrelerin sürekli veya süreksiz değişimleri, iki boyutlu bir cismin kalınlığı aynı anda verilebilmektedir.

Sonlu elemanlar yönteminin sistematik genelleştirilebilmesi onu çok çeşitli problemlere uygulanabilir güçlü ve kullanışlı bir metot haline getirmektedir. Sonuç olarak genel amaçlı, esnek, her probleme uyarlanabilir bilgisayar programları geliştirilebilmiştir. Metodun fiziksel anlamının olması, matematik temelinin gücü ve ifadesi sayesinde uygulamacıya ve mühendise kolaylık sağlamaktadır.

2.4.2. Yöntemin Sınırlamaları

Katı mekaniğindeki lineer olmayan malzemelerin sonlu eleman tekniği ile ifadesi analitik yöntemlerin gelişiminden geride kalmıştır. Sonlu eleman bilgisayar kodları oldukça büyük bilgisayar kapasitesine ve zamana ihtiyaç duymaktadır. Bu nedenle yöntemin kullanım alanı yüksek hızlı, belleği büyük bilgisayar kullanımıyla sınırlıdır. Sonlu eleman kodlarıyla oluşturulmuş diğer analizlerin tablo ve grafiklerini kullanarak metot, genel mühendislik problemlerine de dolaylı olarak uygulanabilmektedir.

Sonlu eleman tekniğinin kullanımının en yorucu olan kısmı süreklilikleri alt bölgelere ayırma işlemi ve bilgisayar için hata serbestliği olan veri girişi kısmıdır. Bu işlem bir dereceye kadar otomatik olmasına rağmen, bilgisayar programlarında bölgelendirme sırasında mühendislik yorumunun katılabilmesi için tümüyle otomatikleştirilmemiştir. Veri girişindeki hatalar, başlangıçta kabul edilebilir gözüken yanlışların ve fark edilmemiş sonuçların eldesine sebep olabilir. Bu nedenle mühendis veya uygulamacı böyle hataları arayarak kontrol etmelidir. Kontrollere ek olarak veri girişini sağlayan ve bölgelendirilmiş sürekliliğin çizimine imkan veren yardımcı ek işlemler tercih edilmektedir. Bu çizim veri girişinin çok hızlı bir şekilde mühendis veya programcı tarafından görsel kontroluna imkan vermektedir.

Son olarak, diğer sayısal analiz yöntemleri gibi sonlu elemanlar metodu da dikkatle yorumlanmalıdır. Kullanılan malzeme özelliklerinin sınırlamasına, sayısal hata ihtimallerine, formülasyonda yapılan kabullere dikkat edilmelidir. Sonlu eleman işlemi sonucunda oluşturulan çıkış bilgileri büyük yer kaplar, fakat bu veriler mühendislik yorumu ile desteklendiğinde önemlidir ve gereklidir.

2.5. Sonlu Elemanlar Yönteminin Matematiksel Temeli

Sonlu elemanlar yönteminin matematiksel temeli üzerine detaylı ve ayrıntılı bir çalışma bu tez konusunun ötesinde ve amacının dışındadır. Birçok mühendislik uygulaması için böyle ayrıntılar gereksiz ve önemsizdir. Ancak sonlu elemanlar tekniğinin ana mantığını anlamak açısından matematiksel temelinin ana hatlarını bilmek, öğrenmek yeterli olacaktır. Yöntemin matematiksel temelinin güçlü olması hem analiz tekniğini çeşitli mühendislik dallarına ve fizik matematiğine uygulamak, hem de yöntemin doğruluğunu test etmek açısından önemlidir.

Sonlu elemanlar tekniğinin gittikçe daha fazla kullanımıyla yöntemin matematiksel temeli üzerine çalışmalar ivme kazanmış ve gittikçe gelişmiştir. Direkt yöntemin kullanımının sonucunda, sonlu eleman yaklaşımı ve Ritz metodu vb. varyasyonel yöntemler arasındaki ilgi artmıştır.

Klasik Ritz yönteminde, sürekli ortamın tümünde yer değiştirmeyi fonksiyonlar grubu tanımlarken, sonlu elemanlar metodu her eleman için tekil yer değiştirmeyi sağlamaktadır. Elemanlardaki iç yer değiştirmeler düğüm noktası değerleri vasıtasıyla tanımlanmakta ve tüm yer değiştirmeler her biri bir elemanı kapsayan büyük sayıda, sürekli parça parça alanlardan ibaret farz edilmektedir. Ritz metodundaki gibi çözüm genellikle yaklaşıktır ve gerçek durumdan daha katı olan yapı yenilir. Bu durum yalnız komşu elemanlardaki iç yüzeylerde uyumluluk koşulları tatmin ediciyse geçerlidir.

Eğer uyumlu elemanlar kullanılırsa ve eleman boyutları yeterince küçültülürse, gerçek yer değiştirme modeline yeterince yaklaşılacaktır. Elemanlar için seçilen yer değiştirme modelinin, elemanlar içindeki sabit gerilme alanlarını üretebilecek şekilde seçilmesi gereklidir. Bu, elemanların boyutlarının küçüldükçe gerilmelerin her eleman boyunca hemen hemen sabit hale gelebilmesi için gereklidir.

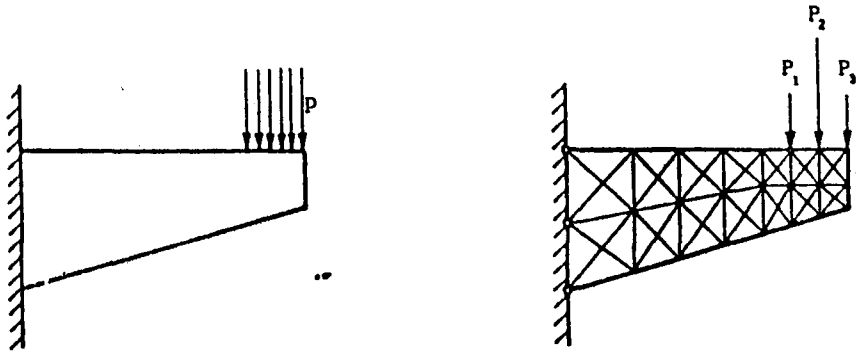
Pratik tecrübeler tümüyle uyumlu elemanlar gerekmediğini göstermektedir. Birçok araştırmacı sonlu elemanlar yönteminin yaklaşım karakteristiklerini katı mekaniğine uygulayarak, yöntemin teorik yönden temelini güçlendirmek için araştırmalar yapmış ve geliştirmiştir. Varyasyon yaklaşımı, sistemin iç şekil değiştirme enerjisinin karekökü olarak tanımlanan enerji formu kavramı kullanarak araştırılmıştır. Diğer yaklaşımlarda hatalar dizisi Taylor Serileri vasıtasıyla analiz edilmiştir. Fizik matematiğindeki problemlerin sınıflandırılması, genişletilmiş sonlu elemanlar teorisi Oden tarafından 1969 'da yapılmıştır.

3. SONLU ELEMANLAR YÖNTEMİNİN TEORİSİ

Modern sonlu eleman teorisi, bilinen başlangıcını yapısal analizin yer değiştirme (veya rijitlik) yönteminde yapmıştır. Başlangıç adımları sezgisel sebepler ve basit teorinin kombinasyonundan oluşmuştur. Daha sonra elastisitenin varyasyon prensiplerinden geliştirilen ilk çalışmalar yapılmıştır. Varyasyon prensiplerine dayanan sonlu elemanlar daha genel ve daha güçlü formülasyona sahiptir. Aynı zamanda varyasyon prensibine nazaran daha kazançlıdır. Bu yaklaşım kullanıldığında yöntemin tüm nitelikleri ortaya çıkmaktadır.

3.1. Sonlu Eleman Kavramı

Kabuklar, barajlar, levhalar vb. iki ve üç boyutlu gövdelerin gerilme ve yer değiştirmelerinin hesabı, diferansiyel denklem grubu içeren kapalı çözümler vasıtasıyla mümkündür. Ancak lineer olmayan, heterojen malzeme özellikleri, karmaşık yükleme modelleri, düzensiz ve karmaşık yapı geometrisi vb. durumlarda pratik tasarımlar bu tip kapalı çözümlerin dışına taşmaktadır ki bu durumda uygulamacı yaklaşık çözümleri kullanmak zorundadır. Bu koşullarda geleneksel yaklaşım; gerçek yapıyı, kapalı bir çözümün uygun olduğu idealize edilmiş, basitleştirilmiş model ile kurmak ve çözmektir. Diğer yaklaşım ise sonlu farklar veya diğer sayısal teknikleri kullanmaktır .



Şekil 3.1 Gerçek yapı ve sonlu eleman modeli (a) yapı (b) model

Sonlu eleman tekniğinde ilk adım, gerçek yapı yerine çözüm için uygun gerilmelerin ve yer değiştirmelerin arandığı basitleştirilmiş, idealize edilmiş sistemi kullanmaktır. Şekil 3.1(a) dış yüklere maruz bir konsolu, Şekil 3.1(b) ise konsolun sonlu elemanlara ayrılmış modelini göstermektedir.

İdealize edilmiş model prensipte özdeş, çerçevenin bilinen yer değiştirme metodu ile analiz edilmektedir. Yapı, dış yükler veya başlangıç şekil değiştirmeleri yüzünden deforme olunca komşu elemanlar arasındaki sürekliliğin, ortak düğüm noktalarında sürdürülmesi istenir. Düğüm noktalarının yer değiştirmeleri bilinmeyen olarak tanımlanır ve düğüm noktası kuvvetleri düğüm noktası yer değiştirmeleri vasıtasıyla gösterilir. Düğüm noktası yer değiştirmeleri ise düğüm noktalarındaki denge koşulları yardımıyla bulunmaktadır. Düğüm noktalarındaki bilinmeyen bağımsız yer değiştirme bileşenleri kinematik belirsizliğin (i) derecesine eşittir .

$$i=kn-1 \qquad (3.1)$$

n = toplam düğüm noktası sayısı

k = her düğüm noktasındaki bağımsız yer değiştirme bileşenleri sayısı (serbestlik derecesi)

1= sıfır olan veya değeri bilinen yer değiştirmelerinin sayısı

Sonlu elemanlar yöntemi iki ana problemin çözümünü gerektirir:

a) Eleman analizi

b) Sistem analizi

Eleman analizi şunları içerir:

1) Eleman içindeki yer değiştirmelerin düğüm noktası yer değiştirmeleri cinsinden tanımlayan fonksiyonların seçimi,

2) İlgili gerilmelerin türetilmesi,

3) Dağıtılan sınır gerilmelerine eşit fiktif düğüm noktası kuvvetlerinin türetilmesi.

Eleman analizi, elemana etki eden düğüm noktası kuvvetleri ve düğüm serbestlikleri arasında ilişki kurulmasından ibarettir. Bu ilişki eleman için rijitlik veya esneklik (flexibilite) matrisi cinsinden tanımlanmaktadır.

Sistem analizi; eleman denklemlerinin uygunluk ve süreklilik şartları da dikkate alınarak, sistem davranışını idare eden bir denklem oluşturulması ve çözülmesinden ibarettir. Sistem analizi, kullanılan eleman tipinden etkilenmeyecek şekilde formüle edilebilir. Sistem analizi için rijitlik matrisinin uygun olduğu herhangi eleman tipiyle çalışan bir program yazmak mümkündür. Böyle bir program farklı tipteki elemanların bileşiminden oluşan yapıları çözmede kullanılabilir. Ancak farklı tipteki elemanlar kullanıldığında ortak sınırlar boyunca yer değiştirme modellerinin uyumluluğuna dikkat edilmelidir.

3.2. Sürekli Ortamın Bölgelendirilmesi

Bölgelendirme prensibi, sürekli ortamın sonlu elemanlar olarak adlandırılan daha küçük sürekliliklere bölünmesi ilkesine dayanmaktadır. "Sürekli ortam" analiz edilecek fiziksel bir gövde, bir yapı veya cisimdir. "Bölgelendirme" verilen bir gövdenin sonlu elemanlardan oluşan alt sistemlere bölünmesidir. Bölgelendirme işlemi bir yaklaşıklık taşır. Sürekli ortam göz önüne alındığında, sonlu elemanlardan oluşan alt sistemlere bölünme işlemiyle doğru sonuca erişebilmek için, ayrık elemanların sayısı arttırılmalıdır.

Çoğu durumda hayali sınırlarla birbirinden ayrılan sonlu elemanlar arasındaki bağlar ve komşu elemanlar sınırsızdır. Bazen bölgelendirme zor olabilir. Bu zorluğun üstesinden gelmek için şu adımlar izlenebilir.

1) Süreklilikler, hayali çizgilerle ve yüzeylerle sonlu elemanlara ayrılır.

2) Elemanların sınırda, ayrık düğüm noktalarında temas ettiği farz edilir ve bu düğümlerin yer değiştirmeleri ve dönmeleri temel bilinmeyen parametredir.

3) Her sonlu elemanda yer değiştirme durumunu tanımlamak için düğüm noktası yer değiştirmeleri cinsinden fonksiyon grubu seçilir.

4) Yer değiştirme fonksiyonları, bir eleman içinde şekil değiştirme durumunu düğüm noktası yer değiştirmeleri cinsinden tanımlar. Bu şekil değiştirmeler, ilk şekil değiştirme ve malzeme bünye özellikleri ile birlikte eleman boyunca ve eleman sınırlarında gerilme durumunu tanımlar.

5) Sistem kuvvetleri düğüm noktalarında yoğunlaştırılır ve dengeyi sağlayan sınır gerilmeleri ve yükleri gösterir.

Sürekli ortam fiziksel bir cismi, yapıyı temsil eder ve kesindir. Örneğin, elastisitede sürekli ortam yalnızca deformasyon yapan cisim veya yapıdır. Ancak bazı durumlarda modellenecek sürekli ortamın kapsamı kesin olarak belirlenemez. Örnek olarak bir veya birden fazla boyutlu, yarı sonsuz veya oldukça büyük, orta boşluklu, poroziteli bir bölgede sızma problemi gösterilebilir. Böyle bir sürekli ortamın yalnızca önemli kısmı düşünülmeli ve bölgelendirilmelidir. Sonlu elemanlar analizinde herhangi bir büyük sürekli ortamın önemli olan kısmını içerecek, kapsayacak şekilde bir kısıtlamaya gerek vardır. Zemin içinden sızan su sürekli ortamı, düğüm noktalarındaki potansiyel su ise bilinmeyeni oluşturur. Yer gerilmeleri, potansiyel su, debi vb. nicelikler ikincil bilinmeyenleri oluşturmaktadır.

Gerilme-şekil değiştirme problemlerinin birincil bilinmeyenleri gerilmeler ve yer değiştirmeler olacak şekilde, problem gerilmeler cinsinden formüle edilmektedir. Sızma probleminde formülasyon, akım fonksiyonları veya akım fonksiyonları ve potansiyel su cinsinden olmaktadır.

Sonlu elemanlar yönteminin temel karakteristiği, sonlu elemanların birer birer analiz edilmesi ve çözülmesidir. Her elemanın fiziksel özellikleri belirlenir ve her birinin özellikleri ve rijitlik denklemleri yazılır. Eleman denklemleri tüm yapının denklemlerini elde etmek için toplanır.

3.2.1. Çeşitli Eleman Şekilleri

Bir boyutlu problemler için çubuk elemanlar, doğrusal veya eğrisel elemanlar kullanılır. İki boyutlu problemler için ise üçgenler, üçgenler grubu, dörtgenler ve diğer geometrik şekiller kullanılır. Üç boyutlu problemlerde ise üçgen prizma, dörtgen prizma veya dikdörtgenler prizması vb. tercih edilir. Sürekli ortamı alt bölgelere ayırma işlemi çoğunlukla otomatik olarak yapılmasına rağmen, sürekli ortamın mükemmel modellenmesi ve doğru sonucu elde etmek, etkin ve verimli çalışma için eleman tiplerinin mühendis tarafından önceden tasarlanması gerekmektedir.

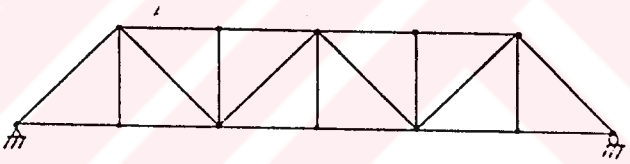
Sürekli bir ortamı alt bölgelere ayırma, bölgelendirme işlemi mühendisin yargısına kalmıştır. Mühendisin dikkat etmesi gereken ilk faktör, analizde kullanılacak olan temel elemanın şeklini, yerleşimini seçmektir. Seçim, tanımlanacak problemin geometrisine, şekline ve bağımsız uzay eksenlerinin sayısına göre değişir. Elemanların sınırları genelde düz

çizgilerle sınırlıdır, bazı özel durumlarda eğrisel sınırlardan söz edilebilirse de, basit çubuk elemanlar kullanmak daha avantajlıdır.

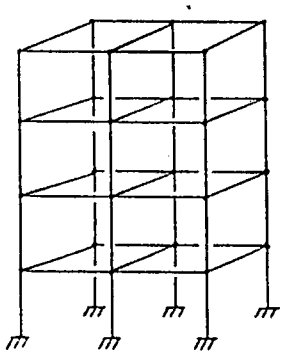
3.2.1.1. Bir Boyutlu Elemanlar

Geometri, malzeme özellikleri ve yer değiştirmeler gibi bağımlı değişkenler, bir bağımsız uzay ekseni cinsinden tanımlanabilirse bir boyutlu eleman seçmek uygun olur. Bu tip elemanlar Şekil 3.2 'de gösterildiği gibi çizgisel olarak idealize edilmiş makas ve çerçeve vb. yapılarda kullanılır. Ancak eksen boyunca sürekli olarak veya süreksiz olarak yapının geometrisinin ve özelliklerinin değiştiği durumlarda, kiriş ve çerçeve yapılarında bir boyutlu sonlu eleman kullanma yaklaşımı yanıltıcı olabilir. Bir boyutlu problemler basit ancak, sonlu elemanlar yöntemini anlamak ve geliştirmek için faydalı ve gerekli bir analizdir (Şekil 3.3).

Bir boyutlu eleman, düğüm noktaları ile son bulan düz bir çizgi ile temsil edilebilir, Şekil 3.4 'de 1 ve 2 no 'lu düğüm noktaları "dış düğüm noktaları" dır. Çünkü bitişik elemanlarla temas noktalarını oluşturmaktadır. Bazı uygulamalar için ek düğüm noktaları gereklidir, diğer elemanlarla hiçbir temasın olmadığı bu orta noktalara "iç düğüm noktaları veya ikincil düğüm noktaları" denir.

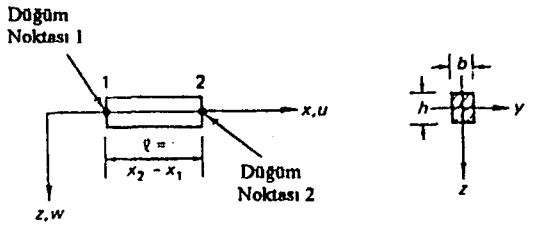


(a) Köprü Makası

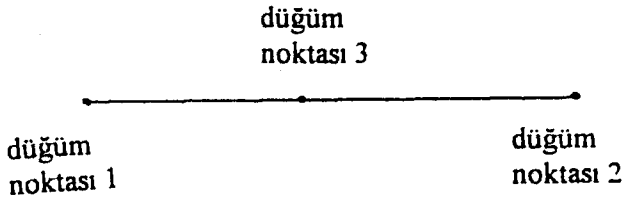


(b) Dört-katlı Bina Çerçevesi

Şekil 3.2 İdealize edilmiş çerçeve yapılar



Şekil 3.3 Kiriş eleman

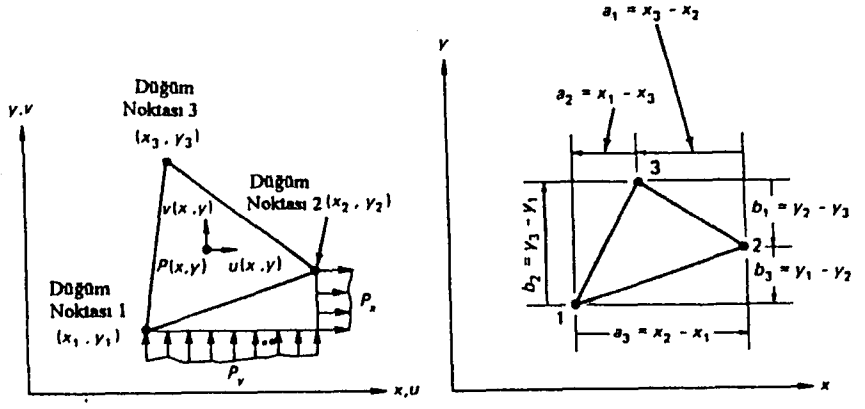


Şekil 3.4 Bir boyutlu eleman

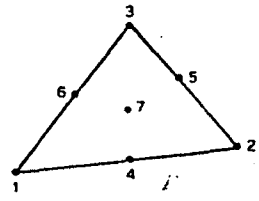
3.2.1.2. İki Boyutlu Elemanlar

Katı mekaniğinde bir çok problem yeteri kadar yaklaşıklıkla iki boyutlu formülasyonla idealize edilebilir. Bunlar düzlem şekil değiştirme, düzlem gerilme ve düzlem eğilme durumlarıdır. İki boyutlu problemlerin en basit formu olan üçgen eleman Şekil 3.5 ve Şekil 3.6(a) 'da görülmektedir.

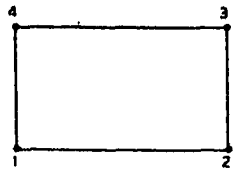
1, 2 ve 3 no 'lu köşe düğüm noktaları "birincil düğüm noktaları", 4, 5 ve 6 no 'lu düğüm noktaları ise "ikincil düğüm noktaları" dır, (Şekil 3.6.a). Bazı uygulamalar için, örnek olarak oturmaların önem kazandığı durumlarda ikincil düğüm noktaları gerekebilir. Bu durumda ya ikincil düğüm noktalarını da içeren elemanlar kullanılmakta ya da küçük boyutlu sayıca fazla elemanlar seçilmektedir.



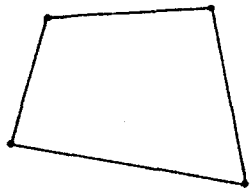
Şekil 3.5 Düzlemsel şekil değiştirme üçgeni



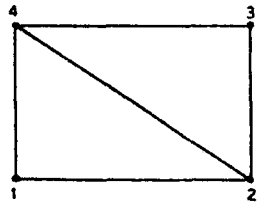
(a) Üçgen eleman



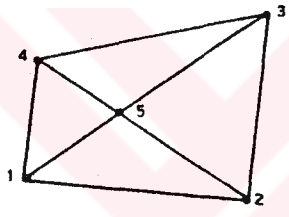
(b) Dikdörtgen eleman



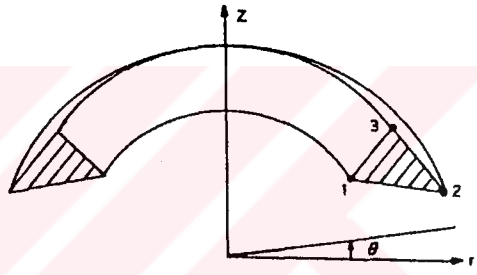
(c) Dörtgen eleman



(d) İki üçgenden oluşmuş dörtgen



(e) Dört üçgenden oluşmuş dörtgen



(f) Eksenelsimetrik üçgen halka eleman

Şekil 3.6 İki boyutlu elemanlar

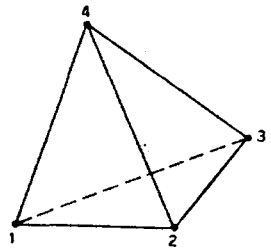
3.2.1.3. Eksenel Simetrik Elemanlar

Eksenel simetrik problemlerde, bir veya iki bağımsız değişken cinsinden temsil edilebilen üç boyutlu problemlerde dairesel halkalar, torodial elemanlar kullanılır. Silindirik koordinat sisteminde r, z ve θ cinsinden ifade edilir, tüm özellikler ve değişkenler θ koordinatından bağımsızdır (Şekil 3.6.f).

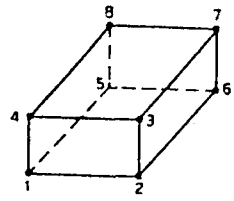
3.2.1.4. Üç Boyutlu Elemanlar

Üçgenlerden oluşan üçgen prizma, üç boyutlu problemlerde kullanılan temel sonlu eleman çeşididir. Dört dış düğüm noktasından oluşan üçgen prizma ve sekiz dış düğüm

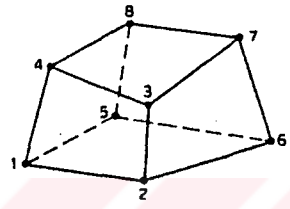
noktasından oluşan dikdörtgenler prizması kullanıldığı gibi ikincil düğüm noktalarını içeren elemanlar veya üçgen prizmalardan oluşan küp prizma da kullanılmaktadır (Şekil 3.6 ve Şekil 3.7).



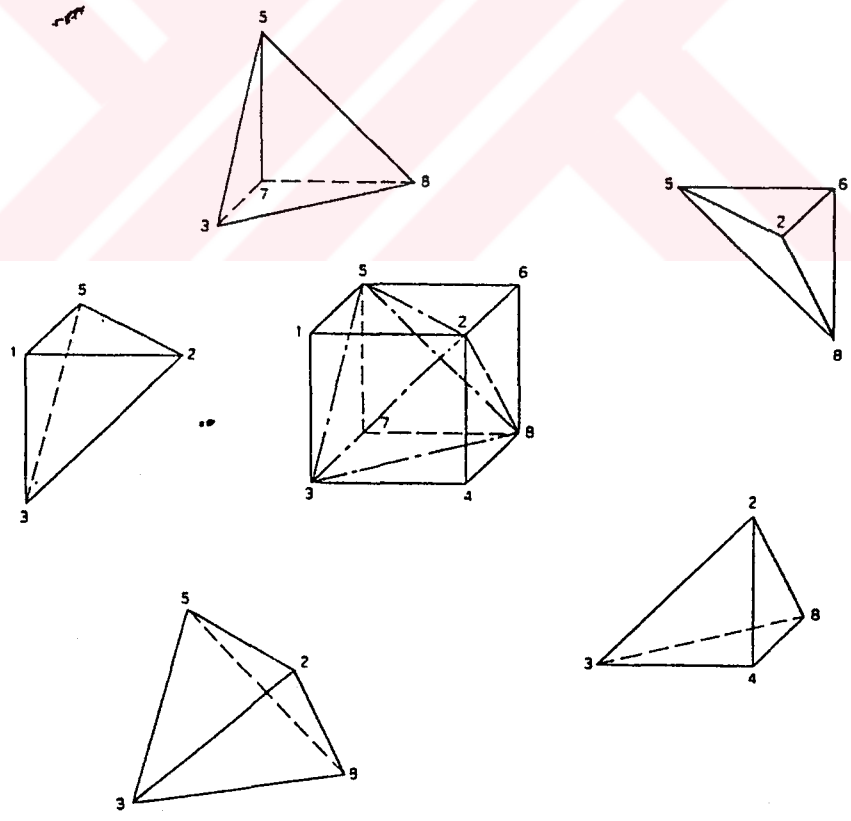
(a) Üçgen prizma



(b) Dikdörtgenler prizması



(c) Keyfi dörtgenler prizması



(d) Beş üçgen prizmadan ibaret küp

Şekil 3.7 Üç boyutlu elemanlar

3.2.2. Bir Yapının Bölgelendirilmesi

Alt bölgelere ayırma işlemi mühendislik yargısına göre değişir. Elemanların boyutları, şekilleri, numaralandırılması, konumu orijinal yapıya mümkün olduğunca benzer seçilmelidir. Bölgelendirme sırasında, yer değiştirme modelinin gerçek çözüme uyması için gövde yeterince küçük elemanlara ayrılmalıdır. Ancak çok küçük elemanlar seçilmesinin bilgisayara daha fazla zaman kaybettireceği unutulmamalıdır.

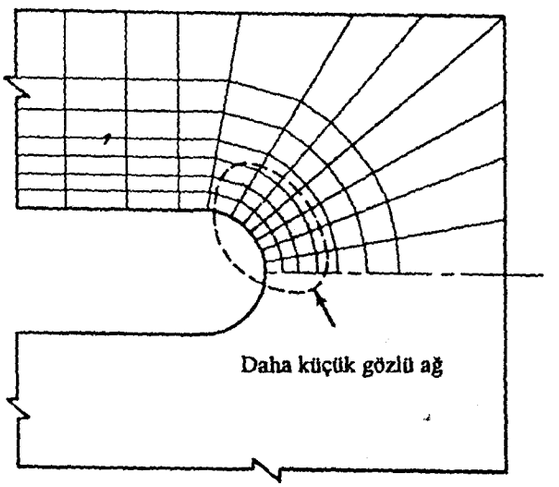
Bölgelendirme işlemi yapılırken ağın oluşturulmasında izlenen en kolay yol aynı boyuta ve şekle sahip elemanları kullanmaktır. Ancak çeşitli gerilme ve şekil değiştirme durumlarının oluşacağı yapılarda böyle bir ağ kullanılmaz. Bu duruma en çok gerilme yığılmalarının yer aldığı örneklerde rastlanır ve uygun bir çözüm için bu bölgelerde daha küçük elemanlar kullanılır.

Düzgün kenarlı elemanlar kullanıldığında eğrisel sınırlar içeren gövdeler yaklaşık olarak bölgelendirilir. Burada dikkat edilmesi gereken husus eğrisel sınırlara mümkün olduğunca yaklaşmak gerektiğidir. İzoparametrik elemanlar kullanarak böyle eğrisel sınırları modellemek mümkünse de karmaşık konfigürasyonlarda başarısız olunması mümkündür.

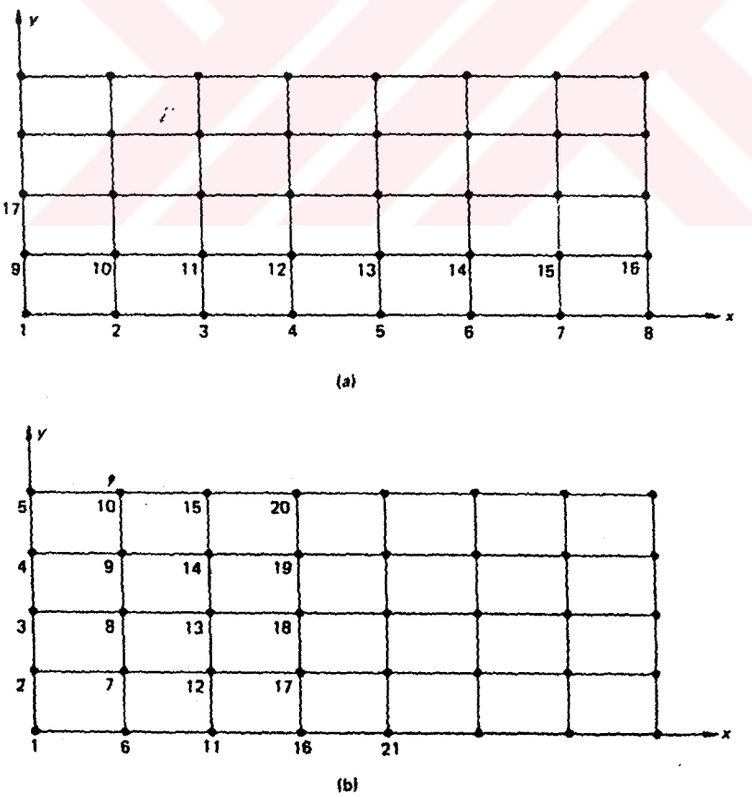
Sonlu elemanlar yöntemi ile çözülen bir çok problem karmaşık geometrilerinden dolayı kapalı form çözümleri içermez. Bu nedenle bölgelendirme esnasında geometrik yaklaşımlara gitme gereği vardır. Bu yaklaşımların mühendislik çözümlerinde faydalı olduğu ve sonuçların yorumlanmasında dikkatli olunması gerektiği bilinmektedir.

Bilgisayar programında işlemleri kolaylaştırmak için elemanları ve düğüm noktalarını sistematik bir şekilde numaralandırma önem kazanmaktadır. Numaralandırma işlemi bir kaç şekilde yapılabilir ve her birinin kendine göre avantaj ve dezavantajları vardır.

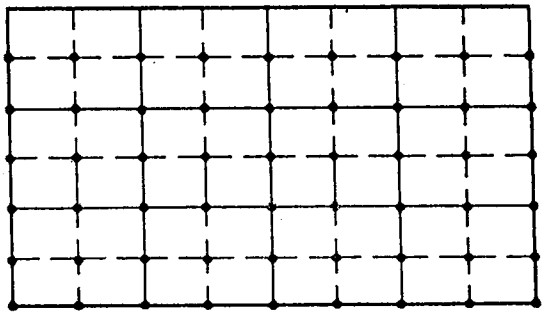
Çözüm zamanı ve tüm rijitlik matrisi için gerekli olan depolama aktivitesini minimuma indirgemek için bant genişliğini azaltmak gereklidir. Bant genişliğinin azaltılması iki şekilde yapılabilir. İlki yüksek dereceden model kullanımını gerektiren durumlardan kaçınarak mümkünse ikincil dış düğüm noktalarını kullanmaktır. Bu birincil düğüm noktalarında, yer değiştirmelerin türevlerini fazladan serbestlik derecesi olarak seçmekle yapılabilir. İkincisi düğüm noktaları için uygun bir numaralandırma sistemi kullanmak ve sistematik alt bölgeler hazırlamaktır. Numaralandırma esas olarak alınırsa, tüm rijitlik matrisinin bant genişliği tekil bir eleman için herhangi iki dış düğüm noktası arasındaki farka bağlıdır. Bant genişliği D değeri düşürülerek azaltılır.



Şekil 3.8 Gerilme yığılması örneği



Şekil 3.9 Bant genişliğini azaltmak için düğüm noktalarını numaralandırma



- = Önceki ağ ve azaltılmış ağdaki düğüm noktası çizgileri

- - = Azaltılmış ağdaki düğüm noktası çizgileri

Şekil 3.10 Dörtgen bölgenin azaltılabilir ağı

$$B=(D+1)f \tag{3.2}$$

B = bandın yarı genişliği

D = tüm eleman topluluğu için oluşan maksimim en geniş fark

f = her düğüm noktasındaki serbestlik derecesi sayısı

Bazı durumlarda minimum bant genişliği ile çalışmamak gerekebilir. Yanlış konumlandırılmış denklemler grubu veren, minimum bant genişliği içeren homojen olmayan problem tipleri örnek olarak gösterilebilir.

Yaklaşım esasını kanıtlamak için yeniden ağ düzenlenmesi aşamasında şu üç koşul sağlanmalıdır.

1) Cisimdeki her nokta ağın düzenlenmesinin herhangi bir aşamasında keyfi küçük elemanların içine dahil edilmelidir.

2) Tüm önceki sonlu eleman ağları daha hassas olan sonlu eleman ağları tarafından içerilmelidir.

3) Önceki ağlardaki düzenlenmiş ağ için form ve yer değiştirme model takımı aynı tutulmalıdır.

Genel olarak ilk iki koşulla tatmin olan alt bölgeler "azaltılabilir ağ" olarak adlandırılır. Şekil 3.10 dikdörtgen bir alandaki azaltılabilir ağ 'ın temel bir örneğini göstermektedir. Kalın çizgiler bir önceki ağın düğüm noktaları çizgilerini, kesikli çizgiler ise

azaltılmış ağ 'ın düğüm noktaları çizgilerini göstermektedir. Buradaki önemli, nokta önceki ağdaki düğüm noktalarının ve çizgilerinin yeni şebekede de aynen korunmasıdır. Bu nedenle bir önceki ağ, düzenlenmiş ağ 'ın basitleştirilmiş hali olarak düşünülmelidir.

Güç, öz vektör veya denge problemleri sonlu eleman yöntemiyle yapılırsa tekil çözümler yaklaşımı ispat etmek için yeterli olmayacaktır. Azaltılmış ağ kullanarak her iki çözümü karşılaştırmak gerekir. Eğer aradaki fark kabul edilebilir düzeyde ise problemin gerçek çözümüne ulaşmak için daha fazla alt bölgelere ayırma işlemine gerek yoktur.

3.2.2.1 Çok Büyük Yapıların Bölgelendirilmesi

Çok fazla sayıda sonlu eleman gerektiren ve çözümü uzun zaman alan, büyük bilgisayar kapasitesi ve belleği gerektiren problemlerde bu güçlüklerin üstesinden gelmek için ya büyükten küçüğe doğru giden çeşitli ebatlarda elemanlar, ya da alt sistemler kullanılmaktadır.

Büyükten küçüğe doğru giden boyutlarda elemanlar kullanılırken, önce sürekli ortam büyük elemanlardan oluşan alt bölgelere ayrılmakta ve bu bölgelerdeki yer değiştirme ve gerilme durumlarını çözmek için sonlu eleman analizi uygulanmaktadır. Sonra istenilen özel alan daha sık ve küçük elemanlı sonlu eleman ağına bölünerek ayrılmaktadır. İlk analizden elde edilen yükleme ve/veya yer değiştirme sonuçları veri olarak kullanılarak ikinci bir analiz yapılmaktadır.

Alt sistemlerde bölme işlemi genellikle uçak çerçeveleri, çok katlı bina çerçeveleri veya gemiler gibi büyük yapılara uygulanır. Bu yapıların her bir parçası alt sistemlere ayrılır ve bunlar çözülerek dış düğüm noktalarında birbirlerine bağlanarak daha karmaşık sistem modellenir ve çözülür. Her alt sistemin rijitlik matrisi;

1) Alt yapıları daha basit sonlu elemanlara ayırarak,

2) Basit eleman rijitliği ve alt yapı için baştan başa rijitlik matrisi topluluğunu hesaplayarak,

3) İç serbestlik derecelerini elemek için alt yapıların rijitliği yoğunlaştırarak elde edilir.

Analiz alt sistemlerin rijitliğini eklemekle ve baştan başa yapıya yüklemeleri uygulamakla sürdürülebilmektedir. Tekil alt yapıların detaylı çözümleri sonra ilgili yapının analizinden bulunan yükleme ve/veya yer değiştirme giriş verileri ile her biri için sonlu eleman analizi uygulayarak hesaplanmaktadır.

3.3. Alan Değişken Modelinin Seçimi

Sonlu elemanlar analizinde formülasyon için alan değişkeninin dağılımını gösteren bir model seçilmelidir. Gerilme şekil değiştirme analizinde alan değişkeni, yer değiştirme modelidir. Alan değişkeni olarak yer değiştirme fonksiyon modeli kullanıldığında, eleman düğüm noktalarında bilinmeyen yer değiştirmeler cinsinden tanımlanan polinomlarla çözüme gidilmektedir. Tahmin edilen yer değiştirme fonksiyonu veya modeli yer değiştirmelerin gerçek dağılımını takriben temsil etmektedir. En basit yer değiştirme modeli yaygın olarak kullanılan lineer polinomlardır. Şekil 2.2 'de taralı abc kısmı üzerinde ijk üçgen sonlu elemanı lineer polinom cinsinden tipik bir yer değiştirme modeli örneğidir. Her elemanda yer değiştirmenin gerçek değişkenini tümüyle temsil edebilen bir fonksiyon seçebilmek mümkün değildir. Yer değiştirme modelinin seçimini etkileyen üç tane birbiriyle ilgili faktör vardır.

1) Yer değiştirme modelinin tipi ve derecesi belirlenmelidir.

2) Modeli tanımlayacak yer değiştirme büyüklüğü seçilmelidir. Bu büyüklük genelde düğüm noktalarının yer değiştirmeleri olarak seçilir. Ancak, bu değerlerin bazıları veya tümü düğüm noktalarının türevini de içermelidir.

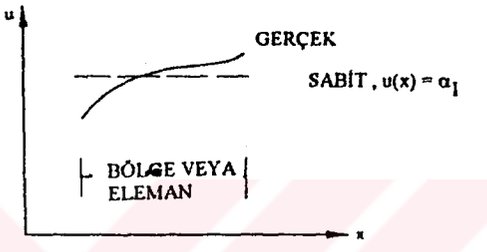
3) Model doğru çözüme yaklaşan sayısal sonuçları sağlamalıdır.

Yer Değiştirme Modellerinin Genelleştirilmiş Koordinat Formu

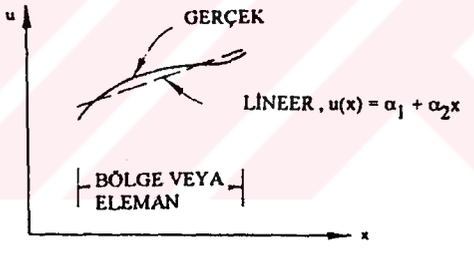
Bu modelin sıkça kullanılmasının iki önemli sebebi vardır. İlki, çeşitli elemanlar için istenilen denklemleri formüle etmek polinomlarla kolaydır. İkincisi, keyfi polinom takımları doğru çözüme kabul edilebilir yaklaşıklıkla varır. Yüksek dereceden polinomlar kullanarak doğru çözüme gittikçe daha fazla yaklaşılır. Bahsedilen yer değiştirme modelinin şematik gösterimi Şekil 3.11 'da verilmiştir.

$$u(x) = \alpha_1 + \alpha_2 x + \alpha_3 x^2 + \ldots + \alpha_{n+1} x^n \qquad (3.3a)$$

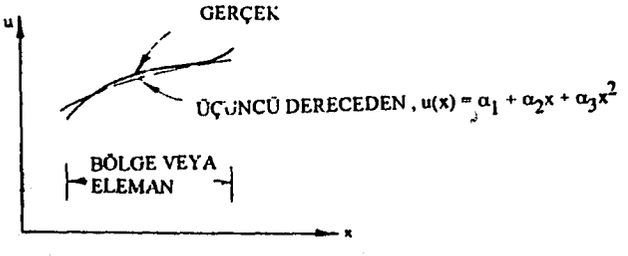
Polinomdaki terimlerin sayısı arttıkça çözüme o oranda yaklaşılır. (3.3a) eşitliğinde katsayılar, ağ , "genelleştirilmiş koordinat" veya "genelleştirilmiş yer değiştirme genişliği" olarak adlandırılır. Polinomdaki terimlerin sayısı yer değiştirme modelinin şeklini, genelleştirilmiş koordinatların büyüklüğü ise yer değiştirme modelinin genişliğini tanımlar. Genelleştirilmiş koordinatlar polinom genişliğini belirtmek için gerekli en küçük parametre sayısını temsil eder.



(a) Sabit (bir terimli) polinom



(b) Lineer (iki terimli) polinom



(c) Üçüncü dereceden (üç terimli) polinom

Şekil 3.11 Bir boyutlu polinom yaklaşımı

(3.3a) eşitliği matris formunda

$$u(x)=\{\Phi\}^T\{\alpha\}$$

$$\{\Phi\}^T=[1 \;\; x \;\; x^2 ..... \; x^n]$$

$$\{\alpha\}^T=[\alpha_1 \; \alpha_2 \; \alpha_3 .... \; \alpha_{n+1}] \qquad (3.3b)$$

şeklinde ifade edilir.

İki boyutlu yer değiştirme modelinde genel polinom

$$u(x,y)= [\alpha_1 + \alpha_2 x + \alpha_3 y + \alpha_4 x^2 + \alpha_5 xy + \alpha_6 y^2 + ... + \alpha_m y^n]$$

$$v(x,y)=[\alpha_{m+1} + \alpha_{m+2} x + \alpha_{m+3} y + \alpha_{m+4} x^2 + \alpha_{m+5} xy + \alpha_{m+6} y^2 + ... + \alpha_{2m} y^n] \qquad (3.4a)$$

olarak tanımlanır.

$$m = \sum_{i=1}^{n+1} i$$

(3.4a) eşitliği matris formunda şu şekilde ifade edilebilir.

$$\{u(x,y)\}=\begin{Bmatrix} u(x,y) \\ v(x,y) \end{Bmatrix}=[\Phi]\{\alpha\}=\begin{bmatrix} \{\Phi\}_1 & \{0\}^T \\ \{0\}^T & \{\Phi\}_1 \end{bmatrix}\{\alpha\}$$

$$\{\Phi_1\}^T=[1 \; x \; y \; z \; zx \; x^2 \; xy \; y^2 \; yz \; z^2 \; ... \; z^n] \qquad (3.4b)$$

$$\{\alpha\}^T=[\alpha_1 \; \alpha_2 \; \alpha_3 \; ..... \; \alpha_{3m}]$$

Son olarak üç boyutlu yer değiştirme modelinde

$$u(x,y,z)= [\alpha_1 + \alpha_2 x + \alpha_3 y + \alpha_4 z + \alpha_5 zx + ... + \alpha_m z^n]$$

$$v(x,y,z)=[\alpha_{m+1} + \alpha_{m+2} x + \alpha_{m+3} y + \alpha_{m+4} z + \alpha_{m+5} xz + ... + \alpha_{2m} z^n] \qquad (3.5a)$$

$$w(x,y,z)=[\alpha_{2m+1} + \alpha_{2m+2} x + \alpha_{2m+3} y + \alpha_{2m+4} z + \alpha_{2m+5} xz + ... + \alpha_{3m} z^n]$$

$$m=\sum_{i=1}^{n+1} i(n+2-i)$$

olarak ifade edilir. u, v ve w yer değiştirme bileşenleridir.(3.5a) eşitliği matris formunda şu şekilde yazılır.

$$\{u(x,y,z)\}=\begin{Bmatrix}u(x,y,z)\\v(x,y,z)\end{Bmatrix}=[\Phi]\{\alpha\}=\begin{bmatrix}\{\Phi_1\} & \{0\}^T\{0\}^T\\ \{0\}^T & \{\Phi_2\}\ \{0\}^T\\ \{0\}^T\ \{0\}^T & \{\Phi_3\}\end{bmatrix}\{\alpha\}$$

$\{\Phi_2\}^T=[1\ x\ y\ z\ zx\ x^2\ xy\ y^2\ yz\ z^2 \dots\ z^n]$ (3.5b)

$\{\alpha\}^T=[\alpha_1\ \alpha_2\ \alpha_3\ \dots\ \alpha_{3m}]$

Tüm bu genel polinom formlarının her biri doğrusal, ikinci dereceden, üçüncü dereceden veya daha yüksek dereceden olacak şekilde istenilen derecede kesilebilir. Örnek olarak, iki boyutlu durumda doğrusal model;

$$u=\alpha_1+\alpha_2 x+\alpha_3 y$$
$$v=\alpha_4+\alpha_5 x+\alpha_6 y \quad (3.5c)$$

olarak, ikinci dereceden model ise

$$u=\alpha_1+\alpha_2x+\alpha_3y+\alpha_4x^2+\alpha_5xy+\alpha_6y^2$$
$$v=\alpha_7+\alpha_8x+\alpha_9y+\alpha_{10}x^2+\alpha_{11}xy+\alpha_{12}y^2 \quad (3.5d)$$

olarak verilebilir.

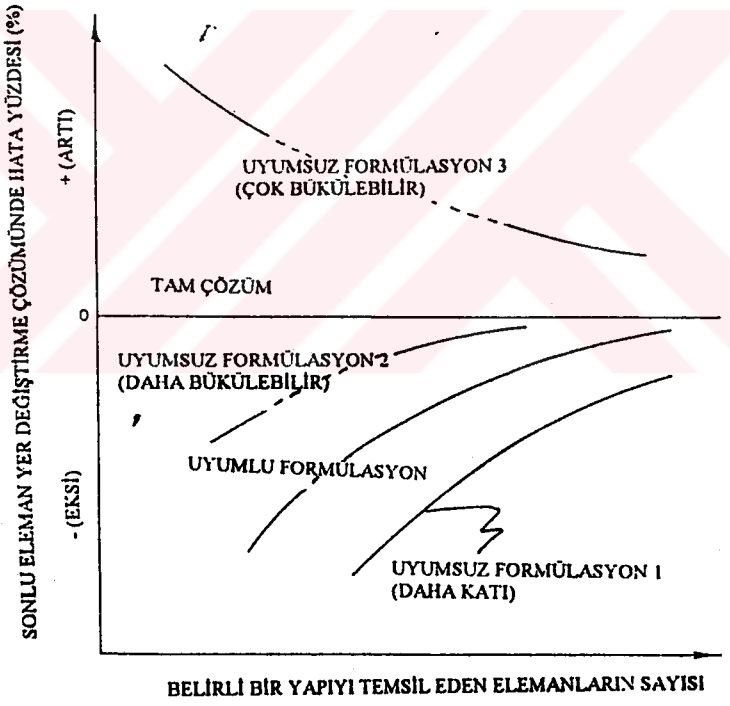
Sonlu elemanlar yöntemi için kesin durumlar altında yer değiştirme formülasyonunun, yapının gerçek rijitliğini sağladığı gösterilmiştir. Diğer bir deyimle verilen bir yer değiştirme modeli için, rijitlik katsayısının doğru çözüme diğerlerinden daha fazla yaklaştığı söylenebilmektedir. Bu yüzden, verilen bir yükleme altında temsil edilen yapı gerçek yapıdan daha az deformasyon yapar. Eğer sonlu eleman iyi tasarlanmışsa tahmini yer değiştirme gerçek çözümün altında kalır. Bu durumun gerçekleştirilebilmesi için şu üç koşul sağlanmalıdır.

1) Yer değiştirme modeli elemanlarda sürekli olmalı ve yer değiştirmelerin bitişik elemanlarda sürekliliği sağlanmalıdır. Sürekli polinom modellerinin seçimi tutarlı olmalı ve elemanlar arasında süreksizliklere, atlamalara, açıklıklara sebep olmaksızın bitişik elemanların deformasyon yapmaları sağlanmalıdır. Bir elemanın kenarı boyunca yer

değiştirmeler, o kenarı oluşturan düğüm noktalarının yer değiştirmelerine tabi değilse, iç elemanların uyumluluğundan söz edilemez.

2) Yer değiştirme modeli, elemanın rijit gövde yer değiştirmesini içermelidir. Rijit gövde deplasmanı bir elemanın katlanabileceği en temel deformasyondur. Örneğin, yer değiştirme modeli denkleminde (3.3a) α_1, sabit terimi rijit bir yer değiştirme sağlar.

3) Yer değiştirme modeli elemanın sabit şekil değiştirme durumlarını içermelidir. Bu koşulun gerekliliğini fiziksel olarak anlayabilmek için gittikçe daha küçük elemanlardan oluşan alt bölgeler içeren bir gövde düşünülürse, elemanlar sonsuz boyuta yaklaştıkça her elemandaki şekil değiştirmeler sabit değerlere yaklaşacaktır. Yaklaşık tahmin bu sabit şekil değiştirmeleri içermedikçe doğru çözüme ulaşılamaz. (3.4a) denklemlerindeki α_2, α_{m+3} değerleri bir elemandaki ε_x ve ε_y üniform şekil değiştirmeleri verir.



Şekil 3.12 Üç uyumsuz, hipotetik elemanla uyumlu bir elemanın şematik karşılaştırılması

Yer değiştirme modelleri çekme, düzlem şekil değiştirme ve üç boyutlu elastisiteye uygulandığında yukarıda verilen üç koşulu sağlamaktadır. Kiriş ve kabukların analizinde ilk iki koşulu sağlamak daha zordur. İlk koşulu sağlayan fakat tümüne uymayan elemanlar yaygın olarak ve başarıyla kullanılırlar. Uyumlu ve uyumsuz elemanların birbirleriyle karşılaştırılmaları Şekil 3.12 'de gösterilmiştir.

Polinom Takımının Seçimi

Yer değiştirme modeli polinom takımının seçiminde, uyumluluk ve bütünlük önemli olmasına rağmen tek başına yeterli değildir. Model dizisinin seçiminde dikkate alınması gereken husus, modelin lokal koordinat sisteminin yöneliminden bağımsız olması gerektiğidir. Modelin bu özelliğine "geometrik izotropi, uzaysal izotropi veya geometrik sabit" denir. Lineer polinom takımı için izotropi, sabit şekil değiştirme durumları kadar gereklidir. Yüksek dereceden modeller için lokal koordinat sisteminde değişiklik yapan yer değiştirme şekilleri istenmez. İzotropinin başarılı olması için bir yol, iki boyutlu polinomların değişken terimleri için Paskal üçgenini kullanmaktır.

1	sabit terim	
$x\ y$	lineer terimler	
$x^2\ \ xy\ \ y^2$	ikinci dereceden terimler	(3.6)
$x^3\ \ x^2y\ \ xy^2\ \ y^3$	üçüncü dereceden terimler	
$x^4\ \ x^3y\ \ x^2y^2\ \ xy^3\ \ y^4$	dördüncü dereceden terimler	
$x^5\ \ x^4y\ \ x^3y^2\ \ x^2y^3\ \ xy^4\ \ y^5$	beşinci dereceden terimler	

Yer değiştirme modeli polinom takımının seçiminde dikkat edilmesi gereken diğer nokta, bir eleman için genelleştirilmiş koordinatların toplam sayısının elemanın dış serbestlik derecesine, düğümlerin sayısına eşit veya daha büyük olması gerektiğidir. Genel kural, serbestlik kadar genelleştirilmiş koordinat kullanmaktır. Eleman rijitlik matrisini düzenlerken fazla sayıda genelleştirilmiş koordinat kullanmak mümkündür. Fazla sayıdaki koordinatlar iç düğüm noktalarıyla ilişkilendirilir ve eleman içindeki dengeyi artırır, fakat elemanlar arası dengeyi iyi kuramaz. Bu nedenle bir kaç fazla koordinatın varlığı tercih edilebilirse de bazı durumlarda zararlı olduğu unutulmamalıdır

Düğüm Noktalarının Serbestlik Derecesi

Sonlu elemanın deformasyonunu tümüyle belirleyebilmek için gerekli olan düğüm noktası yer değiştirmeleri, dönmeleri ve/veya şekil değiştirmeleri elemanın "serbestlik derecesini" verir. Serbestlik derecesi, her biri tekil bir düğüm noktasını tanımlayan ve fiziksel anlamı olan dönme ve şekil değiştirmeyi veren genelleştirilmiş koordinattan farklıdır. İç ve dış düğüm noktalarının serbestlik dereceleri birbirinden farklıdır. Tekil elemanın formülasyonunda bu farklılığı belirtmeye gerek yoktur. Ancak eleman topluluğu aşamasında bu fark önem kazanmaktadır. Fazla sayıda serbestlik derecesi içeren elemanlar "yüksek dereceden elemanlar" olarak adlandırılır. Serbestlik derecesi arttırıldıkça verilen boyuttaki bir eleman için eleman rijitliği daha esnek hale gelmektedir.

Serbestlik derecesi ve genelleştirilmiş koordinatlar yer değiştirme modeline uygulandığında, yer değiştirmeler düğüm noktalarında düğüm koordinatlarıyla değiştirilebilir. Örneğin, ilk modelin formüllerini (3.3b), (3.4b) ve (3.5b) kullanılarak

$$\{u\}=[\phi]\{\alpha\} \tag{3.7}$$

$$\{q\}=\begin{Bmatrix} u1 \\ u2 \\ \cdot \\ \cdot \\ v1 \\ v2 \end{Bmatrix}=\begin{Bmatrix} [\Phi_1] \\ [\Phi_2] \\ [\Phi_3] \\ \cdot \\ \cdot \\ \cdot \end{Bmatrix}\{\alpha\}=[A]\{\alpha\} \tag{3.8}$$

{q}=düğüm noktası yer değiştirmelerinin vektörü.

ve (3.8) formülünü kullanarak

$$\{\alpha\} = [A^{-1}]\{q\} \tag{3.9}$$

$[A^{-1}]$ = dönüştürülmüş yer değiştirme matrisi

[A] bir kare matris olduğu için genelleştirilmiş koordinatların toplam sayısı, iç serbestlik derecesi ve düğümlerin toplam sayısına eşittir. (3.9) denklemi (3.7) denkleminde yerine konulursa

$$\{u\}=[\phi][A^{-1}]\{q\} = [N]\{q\} \qquad (3.10)$$

denklemi bulunur. Eleman içinde herhangi bir P noktasında {u} yer değiştirmesi,{ q } düğüm noktası yer değiştirmesi vasıtasıyla tanımlanır.

Her zaman [A] matrisinin tersini elde etmek mümkün olmadığı için genelleştirilmiş koordinat, yer değiştirme modelini sınırlamaktadır. Bu güçlüğün üstesinden gelmek için yer değiştirme modelini temsil eden interpolasyon fonksiyonu kullanılmaktadır.

3.3.1. İnterpolasyon Modeli

Eğer interpolasyon fonksiyonu [N] matrisi içinden doğrudan seçilirse [A] matrisinin tersini elde etme işlemine gerek kalmaz. (3.7)-(3.10) eşitliklerinden açıkça görüldüğü gibi (3.10) eşitliğinde [N] matrisi direkt olarak kurulabilirse $[A]^{-1}$ matrisini hesaplama gerekliliğinden kurtulmak mümkündür. Gerçekten bir çok durumda interpolasyon fonksiyonu yer değiştirme modeli için temel kabul edilerek, bu tersini alma işleminden kaçınılabilir. [N] matrisi "interpolasyon fonksiyonu" veya "şekil fonksiyonundan" ibarettir. İnterpolasyon fonksiyonu tanımlı olduğu düğüm noktasında değeri "1" olan ve elemanın diğer düğüm noktalarında "0" değeri alan bir fonksiyondur.

{u} vektöründeki her yer değiştirme için {q} vektöründe temsil edilen her düğüm noktası serbestlik derecesinin yerini tutan interpolasyon fonksiyonuna ihtiyaç vardır. Bu işlem bir boyutlu ve üçgen elemanlar için çeşitli interpolasyon modelleri dizilerini uygulamakla gerçekleştirilebilir.

İnterpolasyon fonksiyonlarının doğal koordinat sistemleriyle tariflenmesi en uygun yoldur. İnterpolasyon fonksiyonları genelde yerel ve doğal sistemlerinin kullanımına dayanır. "Yerel koordinat sistemi" belirli özel bir elemana aittir. Tüm yapı veya gövde için tanımlanan koordinat sistemi "global sistemdir". "Doğal koordinat sistemi" ise büyüklüğü kesinlikle "1" değerini geçmeyen boyutsuz sayılar gurubundan oluşan eleman içindeki bir noktanın belirtilmesine olanak tanıyan yerel bir sistemdir. Böyle bir koordinat sistemi formülasyonu birleştirme ve genelleştirme ile kalmaz, aynı zamanda eleman rijitliğini hesaplamak için gereken eleman integrasyon işlemlerini elde etmeyi de amaçlar. Şekil 3.13 bir, iki ve üç boyutlu elemanların doğal koordinat sistemlerini göstermektedir.

Üçgen bir elemanın doğal koordinat sisteminde, Şekil 3.13(c), Lı bir noktayı tanımlamak için kullanılır ve bunlardan ikisi bağımsızdır.

$$\begin{Bmatrix}1\\x\\y\end{Bmatrix}=\begin{bmatrix}1 & 1 & 1\\x_1 & x_2 & x_3\\y_1 & y_2 & y_3\end{bmatrix}\begin{Bmatrix}L_1\\L_2\\L_3\end{Bmatrix} \qquad (3.11\text{ a})$$

A üçgenin toplam alanı ve A_1, A_2 ve A_3 üçgenin bileşenlerinin alanı ise (alan koordinatları) şöyle tanımlanır;

$L_1+L_2+L_3=1$ (3.11b)

$A_1+A_2+A_3 = A$ (3.11c)

$L_1 = A_1/A$, $L_2 = A_2/A$, $L_3 = A_3/A$ (3.11d)

(3.11 a) eşitliğinin tersi alınırsa

$$\begin{Bmatrix}L_1\\L_2\\L_3\end{Bmatrix}=1/2A\begin{bmatrix}2A_{23} & b1 & a1\\2A_{31} & b_2 & a_2\\2A_{12} & b_3 & a_3\end{bmatrix}\begin{Bmatrix}1\\x\\y\end{Bmatrix} \qquad (3.11\text{e})$$

yazılabilir. Burada

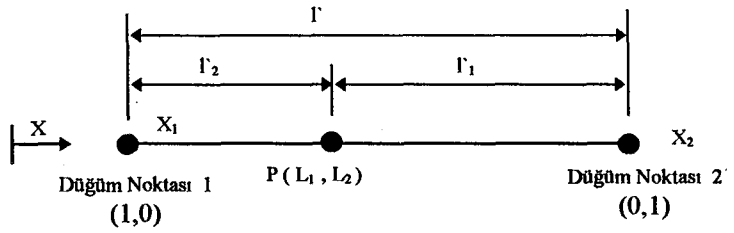
$a_1 = x_2-x_3$ $\qquad b_1 = y_2- y_3$

$a_2 = x_1-x_3$ $\qquad b_2 = y_3-y_1$ (3.11f)

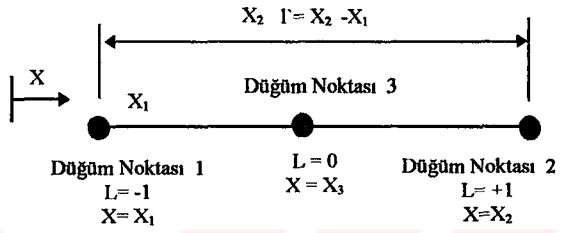
$a_3 = x_2-x_1$ $\qquad b_3 = y_1-y_2$

$2A = a_3b_2-a_2b_3 = a_1b_3-a_3b_1 = a_2b_1- a_1b_2$ (3.11g)

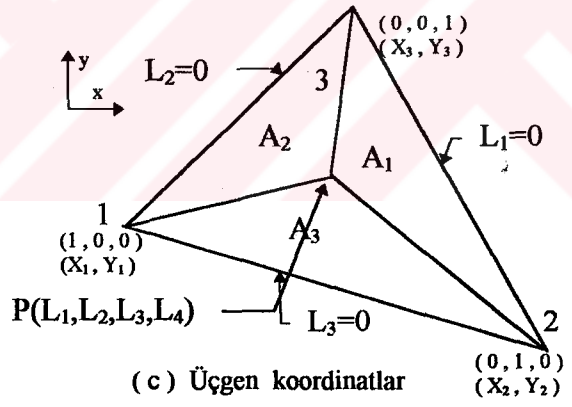
olarak gösterilir.



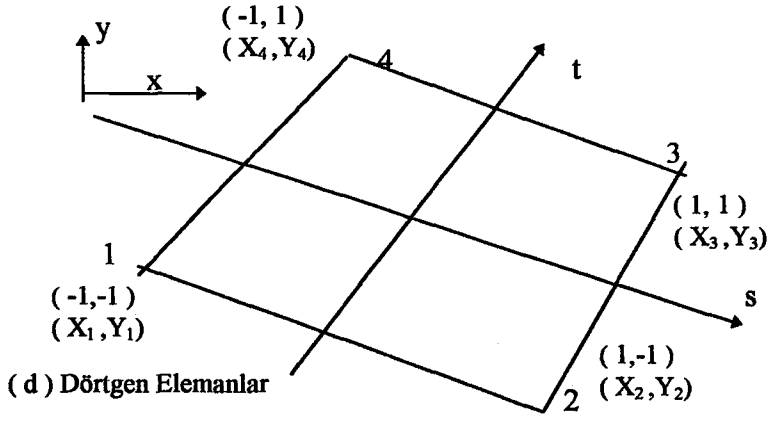
(a) Çizgi eleman için doğal koordinat



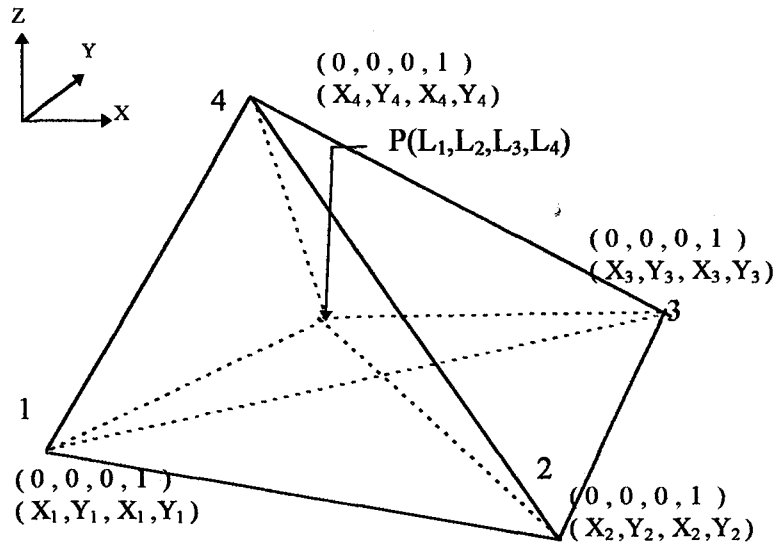
(b) Çizgi eleman için basit bir doğal koordinat



(c) Üçgen koordinatlar



(d) Dörtgen Elemanlar



(e) Tetrahedral koordinatlar

Şekil 3.13 Doğal Koordinat Sistemleri

3.4. Sonlu Eleman Denklemlerinin Elde Edilmesi

Bir sonlu elemanın özelliklerini belirleyen denklemlerin elde edilmesi için bir çok yöntem vardır. Bu yöntemlerin başında "Varyasyon Yöntemi" ve "Kalıcı Yöntem" gelir. Galerkin Yöntemi olarak da bilinen "Ağırlıklı Artıklar Yöntemi" son yıllarda en sık kullanılan metot olmuştur. Çünkü hem lineer hem de lineer olmayan denklemlerle oluşturulan problemlerin çözümü için en uygun ve genel yöntemdir. Eleman denklemlerine rehberlik eden formülasyon, matris notasyonunda şu şekilde gösterilmektedir.

$$[k]\{q\} = \{Q\} \qquad (3.12)$$

[k] = eleman özellikleri matrisi (yer değiştirme probleminde rijitlik matrisi, akım probleminde permeabilite matrisi)

{ Q } = düğüm noktasına etki eden kuvvet vektörü

{ q } = düğüm noktası yer değiştirme vektörü

Bu bölümde eleman özelliklerini hesaplayan genel metotlar anlatılmadan önce yapısal analizin temel tekniğinden türetilmiş olan "Direkt Metot" anlatılacaktır. Bir boyutlu eleman topluluğu ile temsil edilebilen çerçeve analizinde direkt metot, en az varyasyon yöntemi kadar güçlüdür. Daha karmaşık yapılarda ise varyasyon metodu bir çok yönden daha üstündür.

3.4.1. Direkt Yöntem

Direkt yöntem, sonlu eleman metodunun fiziksel yorumunu anlamaya yardımcı olan bir tekniktir. Bir boyutlu yapılarda verilen birim düğüm noktası yer değiştirmesi ile sınırlanan bir eleman düşünülsün. Bu yer değiştirmeleri sürdürmek için gerekli olan düğüm noktası kuvvetleri (reaksiyonları) değeri 1 olan düğüm noktası yer değiştirmesinin yerini tutan kolon matristir (rijitlik matrisinin elemanlarıdır). Örnek olarak dört düğüm noktası olan bir elemanda, serbestlik derecesi diğerlerinde "0" tutulurken üçüncünün değeri "1" olur. (sonuçta reaksiyon kuvvetleri, matrisin üçüncü kolonunun elemanlarıdır).

$$Q=\begin{Bmatrix}Q_1\\Q_2\\Q_3\\Q_4\end{Bmatrix}=\begin{bmatrix}k_{11} & k_{12} & k_{13} & k_{14}\\ k_{21} & k_{22} & k_{23} & k_{24}\\ k_{31} & k_{32} & k_{33} & k_{34}\\ k_{41} & k_{42} & k_{43} & k_{44}\end{bmatrix}\begin{Bmatrix}q_1\\q_2\\q_3\\q_4\end{Bmatrix}=[k]\{q\} \qquad (3.12)$$

{Q} = düğüm noktası yüklerinin vektörü

[k] = rijitlik matrisi ($q_1=q_2=q_4=0$ ise $q_3=1$ ve k_{13} 1 nolu düğüm noktasındaki reaksiyondur)

Daha karmaşık çok boyutlu yapılarda direkt metodun uygulanması o kadar kolay değildir. Çünkü sürekli bir yapıyı temsil etmek için çok boyutlu eleman topluluğu kullanmak gerekecektir ve çok boyutlu gövdeler için kullanılan, etki katsayılarına dayanan direkt tekniği uyumsuz elemanlar topluluğu ile sonuçlanacaktır. Diğer bir deyimle, boşluklar ve üst üste düşmeler elemanlar arasındaki ara yüzeylerde gelişecek ve uyumluluk yalnız düğüm noktalarında sürdürülebilecektir. Dahası böyle bir analiz düğüm noktalarında gerçekçi olmayan gerilme yığılmalarına sebep olacaktır. Böyle faktörleri karşılamak için bir yol tüm

elemanların aynı şekilde deforme olduğunu ve bu deformasyonların birbirine benzediğini düşünmektir.

Direkt metot formülasyonunda genel yaklaşım, bir elemandaki dağıtılmış yükleri eleman düğüm noktaları arasında bölünmüş olarak düşünmektir.

3.4.2. Varyasyon Prensibi

Varyasyon probleminin yaklaşık çözümüne en direkt varan yöntem Rayleigh-Ritz Metodudur. Bu metotta temel fonksiyonlar grubu seçilir ve sınır koşullarıyla sağlanan yaklaşık bir çözüm bulunur. Rayleigh-Ritz metodunun uygulamasından daha kolay ve bu metoda çok yakın olan diğer bir yöntem Galerkin Yöntemidir. Bu yöntem ileride kısaca anlatılacaktır.

Bu bölümde sonlu eleman denklemlerini türetmek için kullanılan katı mekaniğinin varyasyon teorileri hakkında kısa bilgi verilecektir. Bu teorilerin ispatları belirtilmemiştir, ancak teoriler mekaniğin iyi bilinen enerji prensiplerinden özellikle varyasyon formülasyonunun temeli olarak düşünülen virtüel iş denkleminden türetilebilirler. Bu teorilerin bir çoğu "minimum prensipler" olarak da adlandırılmaktadır. Çünkü fonksiyonun sabit değeri bir minimum olarak da gösterilebilmektedir. Varyasyonel Element Teorisinden önce matematiği kurulan Rayleigh-Ritz Metodu gibi benzer olarak, Galerkin Metodunun da varyasyon teorisinin diğer formlarıyla ilgili olduğu unutulmamalıdır. Gerçekte Rayleigh-Ritz Metodu veya onun uzantısı olan sonlu eleman yöntemi kullanıldığında virtüel iş prensibi gibi varyasyon prensibine gerek yoktur. Pratikte alternatif varyasyon yöntemleri direkt olarak (özel formlarının ilgili problemlerin yapısından elde edildiği) elastisite ve katı mekaniğinin teorisinden elde edilebilmektedir. Sonra elastisite problemleri için ek varyasyon formları kurmak kolaydır. Bu şekilde kurulan varyasyon prensibi virtüel iş prensibine tümüyle denktir. Ancak virtüel iş prensibinin uygulaması daha kolaydır. Sonlu eleman uygulamalarında en çok kullanılan varyasyon yöntemleri üç kategoriye ayrılabilir:

1) Virtüel yer değiştirme prensibi(minimum potansiyel enerji)

2) Virtüel gerilme prensibi(minimum komplementer enerji)

3) Reissner'in varyasyon prensibi

Lagrange çarpanları tekniği minimum komplementer enerji prensibinin uyarlanmış halidir ve eleman ara yüzeyleri boyunca uyumlu yer değiştirme fonksiyonları ve her eleman

içinde bir denge-gerilme alanı olduğunu farz eder. Reissner prensibinin veya geliştirilmiş komplementer enerji prensibinin özel problemlerde kullanılması daha faydalıdır.

Yük vektörü ve eleman rijitlik matrisini hesaplayan genelleştirilmiş metot, katı mekaniğinin varyasyon prensibine uygulanmasıdır. Statik problemlerde minimum potansiyel enerji prensibi kullanılır.

Yer değiştirme modeli elemanlar arası uyumu sağlayan sürekliliklerin veya her eleman için tek tek farz edilen varyasyon prensibindeki herhangi bir integralin, çeşitli yer değiştirme alt modellerinin toplama işlemi olarak düşünülür. Çünkü toplamın integrali tek tek integrallerin toplamına eşittir ve bu nedenle prensip tek tek bütün elemanlara uygulanabilir. Örnek olarak minimum potansiyel enerji düşünülürse, şekil değiştirme enerjisini elde etmek için bir integral alınırsa U, lineer elastik bir cisim için

$$2U = \int_V \int\int \{\varepsilon\}^T \{\sigma\} dV = \int\int\int \sum_{e=1}^{N} \{\varepsilon_e\}^T \{\sigma_e\} dV$$

$$= \sum_{e=1}^{N} \int_{Ve} \int\int \{\varepsilon_e\}^T \{\sigma_e\} dV_e \qquad (3.13)$$

e = eleman indisi

N = gövdeyi temsil etmek için kullanılan eleman toplam sayısı

Gerilmeler ve şekil değiştirmeler birbirlerine "gerilme-şekil değiştirme bağıntıları" ile bağlıdır. Basit bir gerilme-şekil değiştirme bağıntısı olarak Hooke kanununu sağlayan lineer elastik davranış denklemi örnek gösterilebilir. Hooke cismi kullanıldığında gerilme-şekil değiştirme ilişkisi şöyledir.

$$\{\sigma\} = [C]\{\varepsilon\} \qquad (3.14)$$

[C] = Elastik sabitler matrisi

Yer değiştirme modelinin, varyasyon fonksiyonu terimleri cinsinden sistemin potansiyel enerjisi, lineer elastik izotropik davranış gösteren malzemede ;

$$\Pi = 1/2 \int\int\int (\{q\}^T [B]^T [C]\{q\} - 2\{q\}^T [N]^T \{\overline{X}\}) dV + \int_S \int \{q\}^T [N]^T \{\overline{T}\} dS_1 \qquad (3.15)$$

şeklindedir. Burada;

$\{\overline{X}\}$ =gövde kuvvet vektörleri

$\{\overline{T}\}^T$=yüzey kuvvet vektörleri

V = eleman hacmi

S_1 = belirlenmiş kuvvetlerin yüzeyi

olarak ifade edilir. Varyasyon prensibi uygulanarak minimum potansiyel enerji;

$$\delta\prod = 0 \tag{3.16}$$

$$\{k\}\{q\}=\{Q\} \tag{3.17}$$

olarak yazılır. Rijitlik matrisi ve düğüm noktası yük vektörü

$$[k]=\int_V\int\int[B]^T[C][B]dV \tag{3.18a}$$

$$\{Q\}=\int_V\int\int[N]^T[\overline{X}]dV+\int_{Sy}\int[N]^T[\overline{T}]dS_1 \tag{3.18b}$$

ile ifade edilir. Yük vektörü ve eleman rijitlik matrisi için (3.18) eşitlikleriyle verilen formül integrasyonu gerektirmektedir. Kartezyen koordinatlarda sabit şekil değiştirmeli elemanlar için, [B] matrisi sabittir ve rijitliği elde etmek için gerekli integrasyonlar basit mertebelerdedir. Fakat yüksek dereceden modellerde yük karakteristikleri ve rijitlik için integraller polinomlar içermektedir. (3.18) denklemindeki rijitlik matrisinin sürekli ve simetrik olması da önemli bir noktadır. Yalnızca [C] elastik sabitler matrisi değil aynı zamanda $[B]^T[C][B]$ çarpımı da simetrik matristir.

3.4.3. Galerkin 'in Fark Yöntemi

Direkt yöntem ve varyasyon yaklaşımlarına ilave olarak sonlu eleman denklemleri "Fark Yöntemi" uygulanarak da formüle edilebilmektedir. Bu yaklaşımda varyasyonel fonksiyonlara gerek kalmaksızın diferansiyel denklemler doğrudan uygulanmaktadır. Yöntem lineer olmayan denklemlerle açıklanabilen problemlere uygulanabildiği için gün geçtikçe daha fazla kullanılmaktadır. Problem için diferansiyel denklem;

$$Lu=f \tag{3.19}$$

u = alan değişkeni
L = diferansiyel operatör
f = bilinen bir etki fonksiyonu

olarak yazılabilir. Fark R

$$R = Lu^*-f \tag{3.20}$$

u* = alan değişken modeli

olarak tanımlanır. Çeşitli fark yöntemleri farkı minimuma indirgeyen çeşitli tekniklere dayanmaktadır ve varyasyonel fonksiyonun uygun olmadığı problemler için faydalıdır.

3.5. Tüm Sistem İçin Sonlu Eleman Denklemler Topluluğu

Sistem için sonlu eleman denklemler topluluğunu kurmanın kuralları, farklı şebekeler ve çerçeveli yapıların matris analizinde kullanılan yöntemlerle ilgilidir. Sonlu eleman denklemler topluluğunun matematiksel ifadesinin türetilmesi için bir yol sonlu eleman yönteminin varyasyon kavramını kullanmaktır.

Bir önceki bölümde bir eleman için sonlu eleman denklemlerinin elde edilmesi anlatılmıştı. Bundan sonraki adım, tüm sistem için rijitliği ifade eden bu denklemleri birleştirerek toparlamaktır. Bu işlem her eleman için birer birer matris denklemlerini birbirine

eklemekten ibarettir. "Direkt Rijitlik Metodu" olarak adlandırılan bu yöntem, yapının fiziksel konumunun sürekliliğini sağlamak için uygulanır. Direkt rijitlik metodu sonlu eleman uygulamalarında cebrik denklemleri toplamak için hemen hemen her zaman uygulanmaktadır. Yöntemin popülaritesi, bilgisayarda depolanmasında ekonomi sağlaması ve bilgisayar için kodlama tekniğini kolaylaştırmasından dolayıdır.

Tüm gövde için rijitlik bağıntısı "global bağıntı" olarak adlandırılır ve

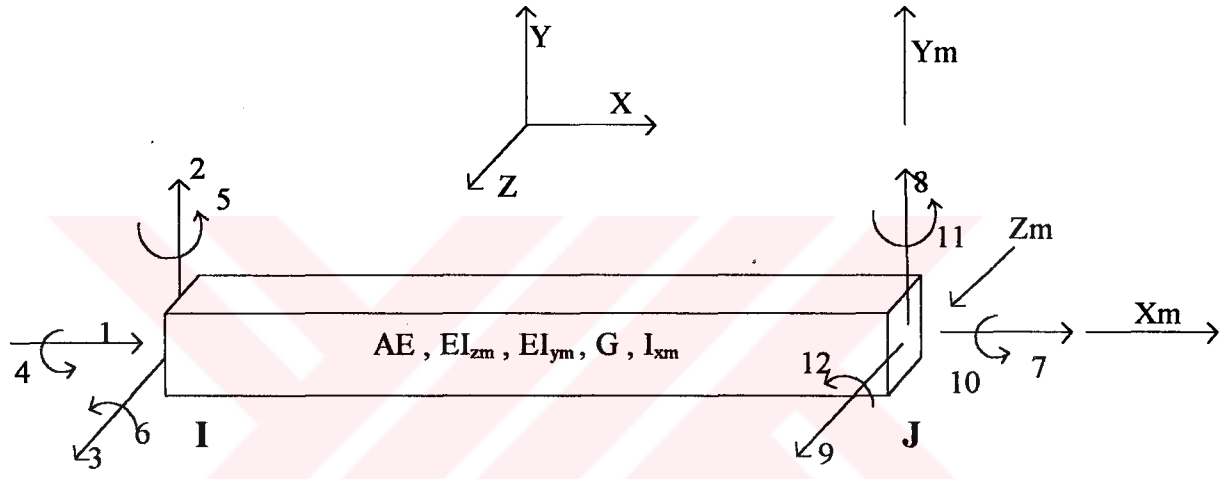
$$[K]\{d`\} = \{Q`\} \qquad (3.21)$$

[K] = global rijitlik matrisi
{d`} = global yer değiştirme vektörü
{Q`} = global yük vektörü
olarak ifade edilir.

Genellikle eleman rijitlik bağıntısı lokal koordinat sistemine göre kurulmaktadır. Böyle lokal bağıntıları global koordinat sisteminde yerine koymadan önce dönüştürmek gereklidir. Rijitlik matrisi yönteminin uygulanışı bundan sonraki bölümde "Rijitlik Matrisi Yöntemi" başlığı altında geniş olarak işlenmiştir.

4. RİJİTLİK MATRİSİ YÖNTEMİ

Yapılan çalışmada sonlu eleman formülasyonları kullanılarak uzay çerçeve elemanlar için elde edilen rijitlik matrisleri kullanılmıştır. Şekil 4.1 'de uzay çerçeve çubuğuna ait eleman eksenleri ve başlangıç (İ) ve son (J) düğümlerine ait hareket serbestileri gösterilmiştir.



Şekil 4.1 Uzay çerçeve elemanına ait hareket serbestileri ve global malzeme eksenleri

X_m ekseni elemanın ağırlık merkezinden geçen eksen ile çakışır. Y_m ve Z_m eksenleri ise X_m-Y_m ve X_m-Z_m gibi eğilme asal eksenlerine göre seçilirler. Şekil 4.1 'de de görüldüğü gibi her düğüm noktasında global eksenler doğrultusunda üç adet ötelenme ve global eksenler etrafında üç adet dönme olmak üzere toplam altı adet hareket serbestisi vardır. İki adet düğüm noktası ile tanımlanan bir uzay çerçeve çubuğu ile toplam oniki adet hareket serbestisi içermektedir.

Yapı statiğinde rijitlik geniş anlamı ile belirli bir doğrultuda birim deplasman temin edebilmek için taşıyıcı sisteme uygulanması gereken kuvvet demektir ve " k " ile gösterilir.

Kuvvetin tatbik edildiği yerdeki okun numarasını (i) ile ve birim deplasmanın bulunduğu yerdeki okun numarasını da (j) ile gösterirsek, bir taşıyıcı elemanın k_{ij} rijitlik katsayısını şu şekilde tarif edebiliriz.

k_{ij}: Taşıyıcı elemanın tarif edilmiş bütün serbestlik dereceleri doğrultusundaki deplasmanlar sıfir iken, yalnız " j " oku doğrultusunda birim bir deplasman meydana getirebilmek için " i " doğrultusunda elemana dıştan uygulanması gereken kuvvet.

k_{ii}: Taşıyıcı elemanın tarif edilmiş bütün serbestlik dereceleri doğrultusundaki deplasmanlar sıfir iken, yalnız " i " oku doğrultusunda birim bir deplasman meydana getirebilmek için gene " i " doğrultusunda elemana dıştan uygulanması gereken kuvvet.

Taşıyıcı elemanın geometrisi ve elastik parametreleri ile bu k_{ij} değerlerine, birim deplasmanlarla ilgili olmalarından dolayı rijitlik katsayıları denmektedir. Toplam serbestlik derecesi 12 olan bir uzay çerçeve elemanının rijitlik katsayılarını adedi 12^2 dir. Bu katsayılar tamamen bağımsız değildirler. Aralarında statik denge denklemlerini sağladıkları gibi, Betti-Maxwell'in karşıtlık prensibi gereğince de $k_{ij} = k_{ji}$ dir.

4.1. Rijitlik Matrisi ve Rijitlik Denklemi

Uzay çerçeve elemanının 12^2 adetteki rijitlik katsayılarını içeren 12 x 12 boyutlu matrisine rijitlik matrisi denir ve [k] ile gösterilir. Bir rijitlik matrisinin k_{ij} teriminin ilk indisi i, matristeki satır numarasını, ikinci indis j ise sütun numarasını gösterir.

Şekil 4.2 Yalnız eksenel kuvvet taşıyan bir düzlem kafes çubuğu

Şekil 4.2 'de basit bir örnek olarak ele alınan düzlem kafes çubuğunun uç kuvvetlerini P_1 ,P_2 ve uç deplasmanlarını da d_1 ,d_2 ile gösterelim. Düzlem kafes çubuğunun

birinci serbestlik doğrultusu tutulur ve ikinci doğrultuda birim deplasman düşünülürse, bu duruma karşılık gelen rijitlik katsayıları

$$k_{22}=AE / L \qquad \text{ve} \qquad k_{12}= -AE / L \qquad (4.1)$$

olarak bulunur. Aynı şekilde ikinci doğrultu hareketten mahrum bırakılır ve çubuğa birinci doğrultuda birim bir deplasman verilirse

$$k_{11}=AE / L \qquad \text{ve} \qquad k_{21}= -AE / L \qquad (4.2)$$

elde edilir. Burada k_{21} yalnızca birinci doğrultuda birim bir deplasman var iken, ikinci doğrultuda uygulanması gereken kuvvet demektir ve d= PL / AE formülasyonunda d=1 yazılarak elde edilir.

Elde edilen rijitlik katsayılarından faydalanarak uç kuvvetleri ve uç deplasmanları arasındaki şu bağıntıyı yazabiliriz.

$$P_1= k_{11}d_1 + k_{12}d_2 \qquad (4.3)$$
$$P_2= k_{21}d_1 + k_{22}d_2$$

matris notasyonu kullanılırsa (4.1) ve (4.2) yardımı ile

$$\begin{Bmatrix} P_1 \\ P_2 \end{Bmatrix} = \begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix} \begin{Bmatrix} d_1 \\ d_2 \end{Bmatrix} = \frac{AE}{L} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \begin{Bmatrix} d_1 \\ d_2 \end{Bmatrix} \qquad (4.4)$$

olur. Genel olarak aynı ifadeyi

$$\{ P \} = [k] \{ d \} \qquad (4.5)$$

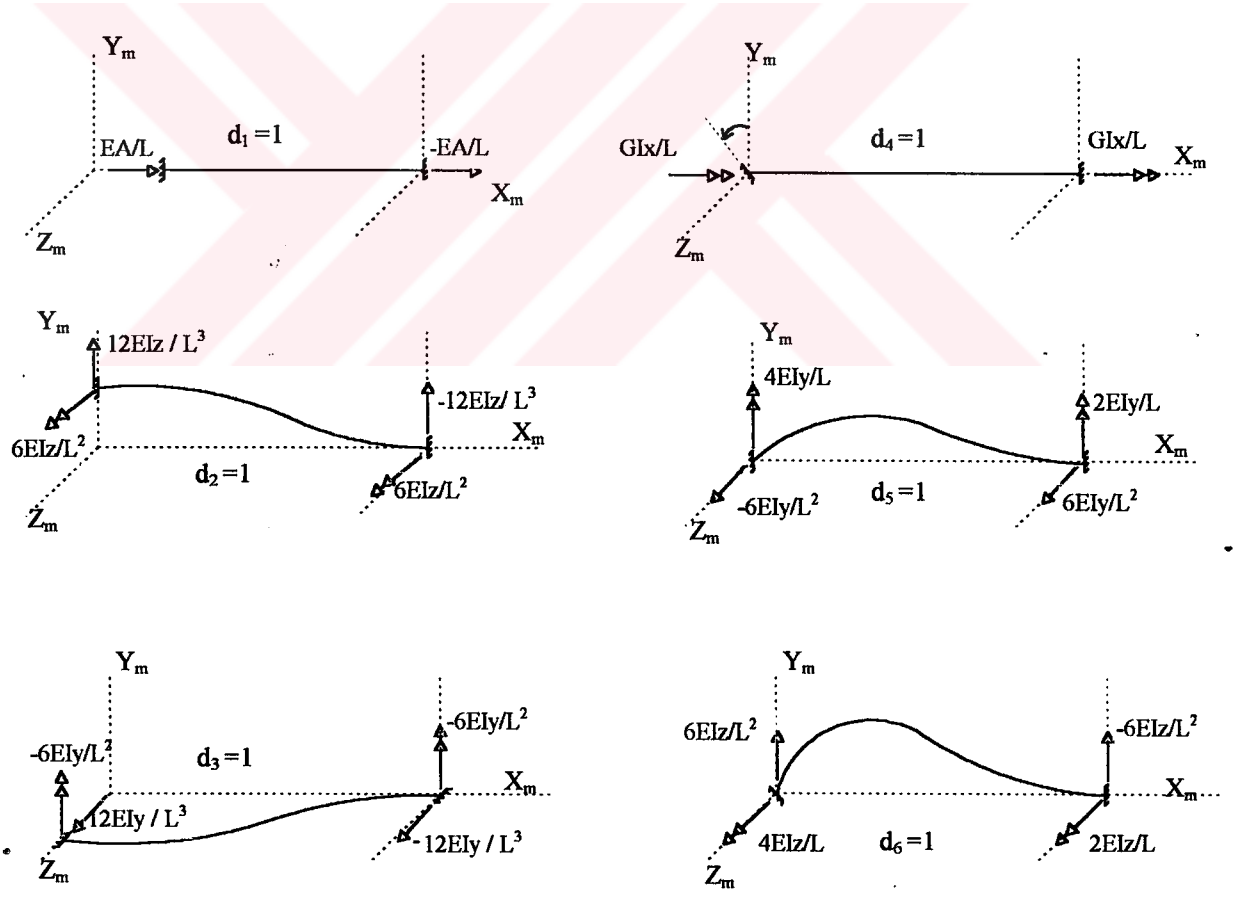
şeklinde yazabiliriz ki bu denkleme çubuğun rijitlik denklemi denir. Bu denklemde { P } = yük vektörü, [d] = deplasman vektörü, [k] = rijitlik matrisidir. İfadeden de anlaşılacağı gibi rijitlik matrisi çubuğun uç deplasmanlarını, karşılık gelen doğrultulardaki uç kuvvetlerine bağlayan birer parametre rolündedir.

Bir rijitlik denkleminden bahsederken, aynı zamanda o denklemin bağlı olduğu eksen takımını da tarif etmek gerekir. Çünkü her değişik eksen takımına yeni bir rijitlik matrisi karşılık gelir. İleride herhangi bir eksen takımına ait rijitlik matrisi biliniyorken, seçilen diğer

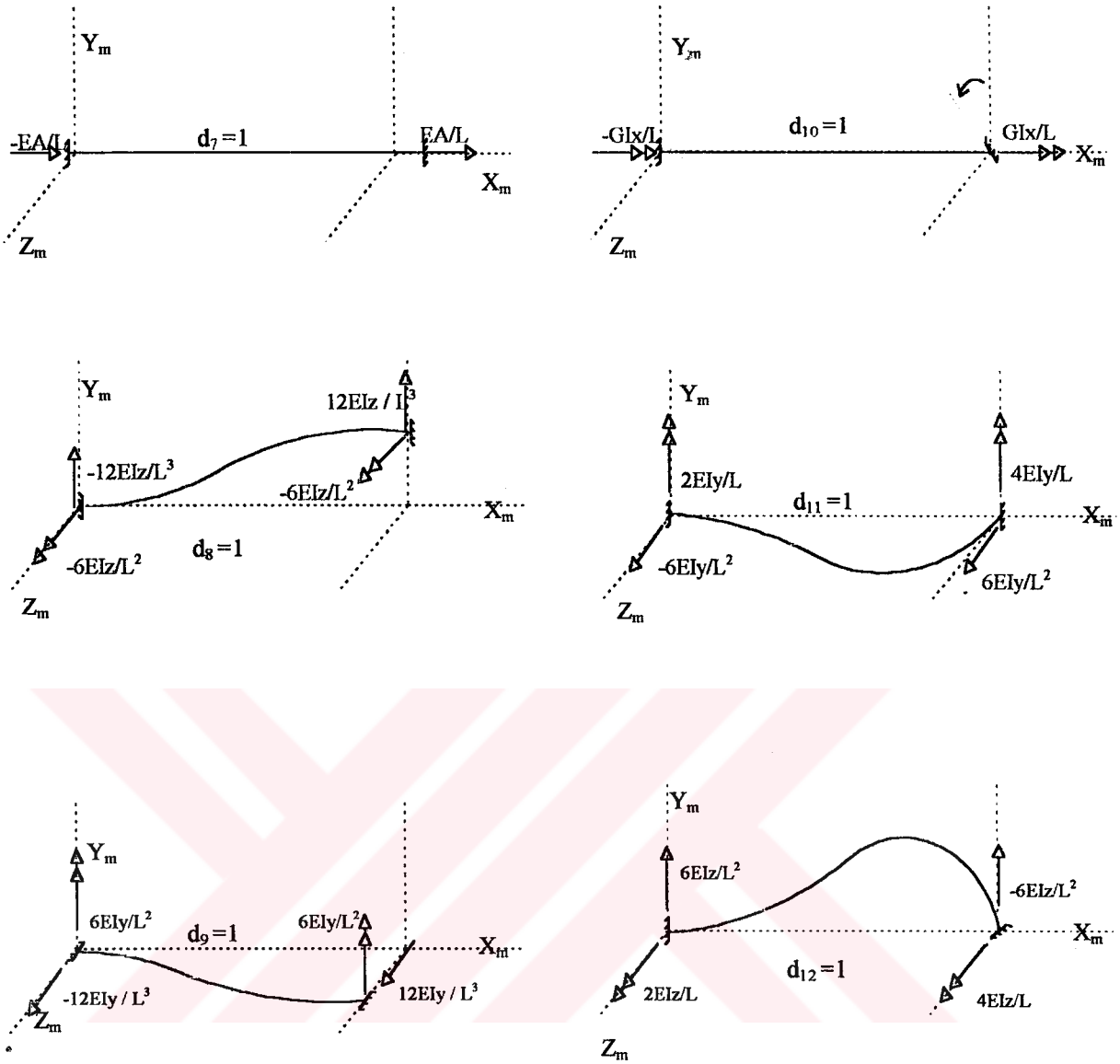
bir eksen takımına göre eşdeğer rijitlik matrisinin nasıl bulunacağından bahsedilecektir. Rijitlik denkleminde, unutulmaması gereken nokta, rijitlik katsayılarının elde edilmesinde kullanılan eksen takımı ile kuvvet ve deplasmanların bağlı olduğu eksen takımının aynı olması gerektiğidir.

4.2. Uzay Çerçeve Çubuğunun Rijitlik Katsayıları

Rijitlik katsayılarının tarifine göre k_{ij} terimini elde etmek için, j doğrultusunda birim bir deformasyon düşünmeli ve diğer deformasyonların sıfır olduğu bu hale ait, elastik eğrinin şeklini muhafaza etmek için gereken kuvvetler hesaplanmalıdır. Yani, tarif edilmiş bütün deformasyon doğrultuları teker teker ele alınmalı ve her seferinde bunlara birim deformasyon vererek uç kuvvetleri hesaplanmalıdır. Bütün bu işlemler uzay çerçeve çubuğu için yapılmış ve şekil 4.3 'de gösterilmiştir.



Şekil 4.3 Uzay çerçeve çubuğu için rijitlik katsayıları.



Şekil 4.3 (Devam) Uzay çerçeve çubuğu için rijitlik katsayıları.

Şekil 4.3 'te, uçlarında birim deformasyonlar bulunan bir uzay çerçeve çubuğunun uç kuvvetleri, her birim deformasyon için ayrı ayrı gösterilmiştir. Aslında bir çubuk, taşıyıcı sistemin yüklenmesinden sonra, birim deformasyonlara değil, $d_1,d_2,d_3,d_4,d_5,.....,d_{12}$ gibi sıfır olmayan sonlu uç deplasmanlarına maruz kalır. Bütün bu deformasyonların hepsinin aynı anda etkilerini içine alan $P_1,P_2,P_3,P_4,P_5,.....,P_{12}$ kuvvetlerini bulmak için, süperpozisyon kuralına başvurmak ve her deformasyon hali için oluşan kuvvetleri toplamak gerekir. Örneğin, d_3 deformasyonundan dolayı oluşan P_4 kuvvetini bulmak için, k_{43} ile d_3`ü çarpmak ve diğer deformasyonlardan dolayı oluşan P_4 tesirleri üzerine eklemek gerekir. Çünkü, k_{43} değeri, rijitlik katsayısını tarifine göre, 3 numaralı doğrultuda birim deformasyon varken, 4

numaralı doğrultudaki kuvveti verir. Her deformasyon hali için elde edilen rijitlik katsayıları bir matris içinde, o deformasyon numarasına karşılık gelen kolona yerleştirilirse, uzay çerçeve çubuğu için rijitlik matrisi şekil 4.4 'deki hali ile elde edilmiş olur.

Herhangi bir doğrultudaki kuvveti bulmak için bu yöntem, bütün deformasyonlara tatbik edilir ve aynı doğrultudaki kuvvetlerin bileşenleri süperpoze edilirse, uzay çerçeve sistemin çubuk kuvvetleri için rijitlik denklemi

$$P_1 = k_{11}d_1 + k_{12}d_2 + k_{13}d_3 + \ldots\ldots\ldots\ldots + k_{1\,12}d_{12} + f_1$$
$$P_2 = k_{21}d_1 + k_{22}d_2 + k_{23}d_3 + \ldots\ldots\ldots\ldots + k_{2\,12}d_{12} + f_2$$
$$P_3 = k_{31}d_1 + k_{32}d_2 + k_{33}d_3 + \ldots\ldots\ldots\ldots + k_{3\,12}d_{12} + f_3 \qquad (4.6)$$
$$\cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot$$
$$P_{12} = k_{12\,1}d_1 + k_{12\,2}d_2 + k_{12\,3}d_3 + \ldots\ldots\ldots\ldots + k_{12\ 12}d_{12} + f_{12}$$

şeklinde elde edilir.

Görüldüğü gibi rijitlik katsayıları, çubuğun uç deformasyonlarını, uçlardaki kesit tesirlerine bağlayan, malzemenin geometrisi ve elastik özellikleri ile ilgili çok kullanışlı sayılardır.(4.6) da verilen rijitlik denklemi matris formunda şu şekilde yazılabilir.

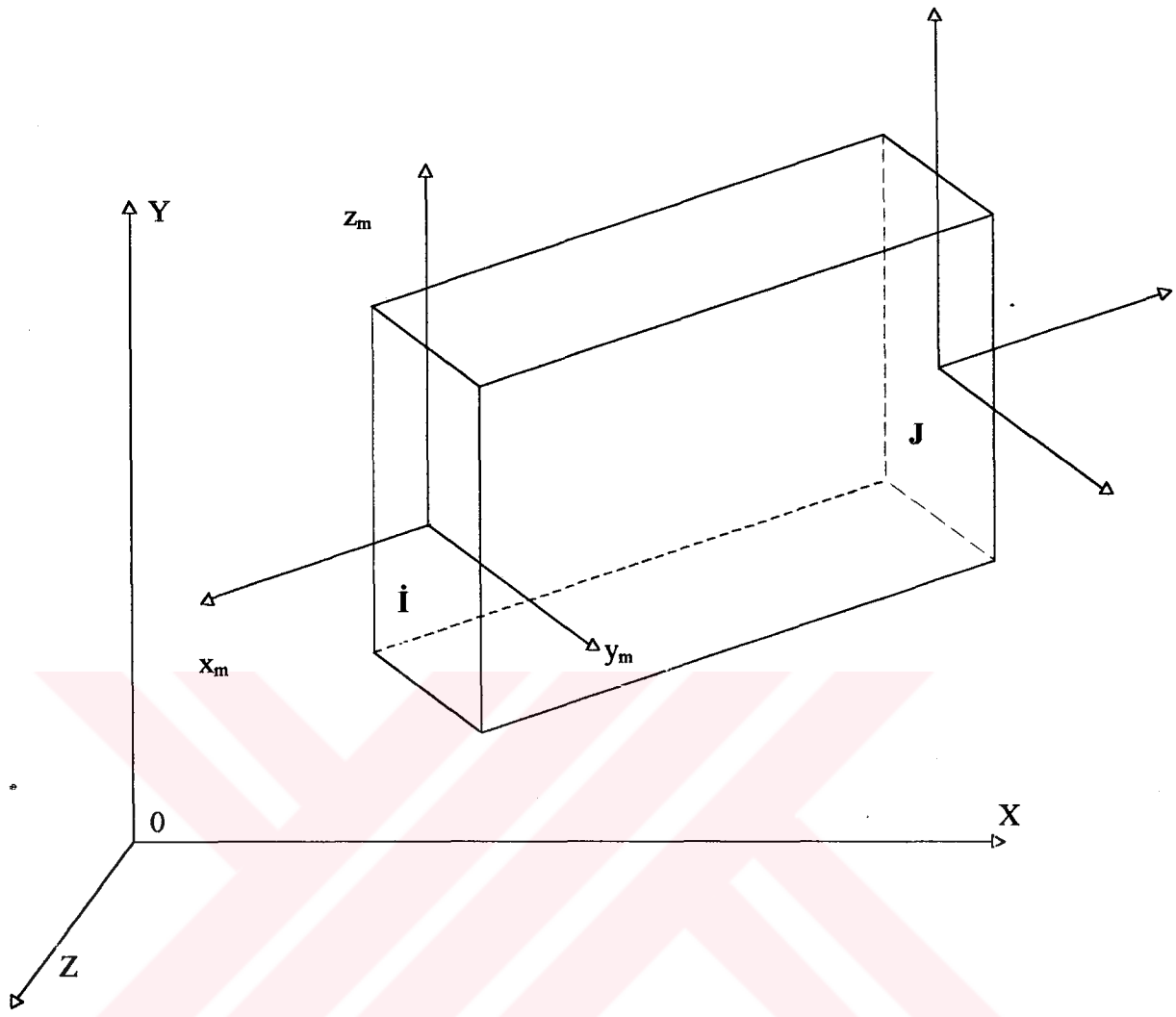
$$\begin{Bmatrix} p_1 \\ p_2 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ p_{11} \\ p_{12} \end{Bmatrix} = \begin{bmatrix} k_{11} & k_{12} & k_{13} & \cdots & k_{1\,12} \\ k_{21} & k_{22} & k_{23} & \cdots & k_{2\,12} \\ \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ k_{11\,1} & k_{22} & k_{23} & \cdots & k_{2\,12} \\ k_{12\,1} & k_{22} & k_{23} & \cdots & k_{2\,12} \end{bmatrix} \begin{Bmatrix} d_1 \\ d_2 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ d_{11} \\ d_{12} \end{Bmatrix} + \begin{Bmatrix} f_1 \\ f_2 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ f_{11} \\ f_{12} \end{Bmatrix} \qquad (4.7)$$

Burada {P}=uç kuvvetleri vektörü, {d}= uç deplasmanları vektörü, {f}=ankastrelik uç kuvvetleri vektörü, [k]= çubuk rijitlik matrisidir. Sisteme etki eden dış yükler (düğüm yükleri haricinde) biliniyor ise, bu tesirlerin düğümlere olan etkileri uygun formüllerle hesaplanarak, {f} ankastrelik uç kuvvetleri vektörü hesaplanır. Bunun yanında, uç deplasmanları,{d} bilinince, [k] rijitlik matrisi ile {d} vektörünün çarpımından, aranan {p} uç kuvvetleri vektörü hesaplanır. Çubuk uç kuvvetlerinin tam değerinin hesabı için elde edilen bu çarpıma {f} vektörü ilave edilmelidir.

$$
k_m = \begin{array}{c}
\begin{array}{cccccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \end{array} \\
\begin{bmatrix}
\frac{EA}{L} & 0 & 0 & 0 & 0 & 0 & \frac{-EA}{L} & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{12EIz}{L^3} & 0 & 0 & 0 & \frac{6EIz}{L^2} & 0 & \frac{-12EIz}{L^3} & 0 & 0 & 0 & \frac{6EIz}{L^2} \\
0 & 0 & \frac{12EIy}{L^3} & 0 & \frac{-6EIy}{L^2} & 0 & 0 & 0 & \frac{-12EIy}{L^3} & 0 & \frac{-6EIy}{L^2} & 0 \\
0 & 0 & 0 & \frac{GIx}{L} & 0 & 0 & 0 & 0 & 0 & \frac{-GIx}{L} & 0 & 0 \\
0 & 0 & \frac{-6EIy}{L^2} & 0 & \frac{4EIy}{L} & 0 & 0 & 0 & \frac{6EIy}{L^2} & 0 & \frac{2EIy}{L} & 0 \\
0 & \frac{6EIz}{L^2} & 0 & 0 & 0 & \frac{4EIz}{L} & 0 & \frac{-6EIz}{L^2} & 0 & 0 & 0 & \frac{2EIz}{L} \\
\frac{-EA}{L} & 0 & 0 & 0 & 0 & 0 & \frac{EA}{L} & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{-12EIz}{L^3} & 0 & 0 & 0 & \frac{-6EIz}{L^2} & 0 & \frac{12EIz}{L^3} & 0 & 0 & 0 & \frac{-6EIz}{L^2} \\
0 & 0 & \frac{-12EIy}{L^3} & 0 & \frac{6EIy}{L^2} & 0 & 0 & 0 & \frac{12EIy}{L^3} & 0 & \frac{6EIy}{L^2} & 0 \\
0 & 0 & 0 & \frac{-GIx}{L} & 0 & 0 & 0 & 0 & 0 & \frac{GIx}{L} & 0 & 0 \\
0 & 0 & \frac{-6EIy}{L^2} & 0 & \frac{2EIy}{L} & 0 & 0 & 0 & \frac{6EIy}{L^2} & 0 & \frac{4EIy}{L} & 0 \\
0 & \frac{6EIz}{L^2} & 0 & 0 & 0 & \frac{2EIz}{L} & 0 & \frac{-6EIz}{L^2} & 0 & 0 & 0 & \frac{4EIz}{L}
\end{bmatrix}
\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \\ 11 \\ 12 \end{array}
\end{array}
$$

Şekil 4.4 Uzay çerçeve çubuğu için eleman koordinatlarında rijitlik matrisi.

4.3. Çubukların Bağlı Olduğu Eksen Takımları



Şekil 4.5 Uzay çerçeve çubuğu eksen takımları.

4.3.1. Lokal Eksen Takımı

Çubuğun kendi ekseni ile bu eksene dik en kesitteki asal atalet momenti eksenlerinden oluşan ve sağ el vida kuralına uyan x_m, y_m, z_m koordinat takımına çubuk lokal eksen takımı denir. Çubuğu en kesitteki ağırlık merkezlerini birleştiren boyuna eksen x_m, büyük ve küçük atalet eksenleri de y_m ve z_m ile gösterilir, (Şekil 4.5).

Uzay bir çerçeve çubuğunda hangi atalet ekseninin y_m veya z_m olarak seçileceğine dair bir kural yoktur. Fakat seçim yapıldıktan sonra, doğal olarak hesaplarda seçilen atalet eksenine karşılık gelen atalet değerleri kullanılmalıdır.

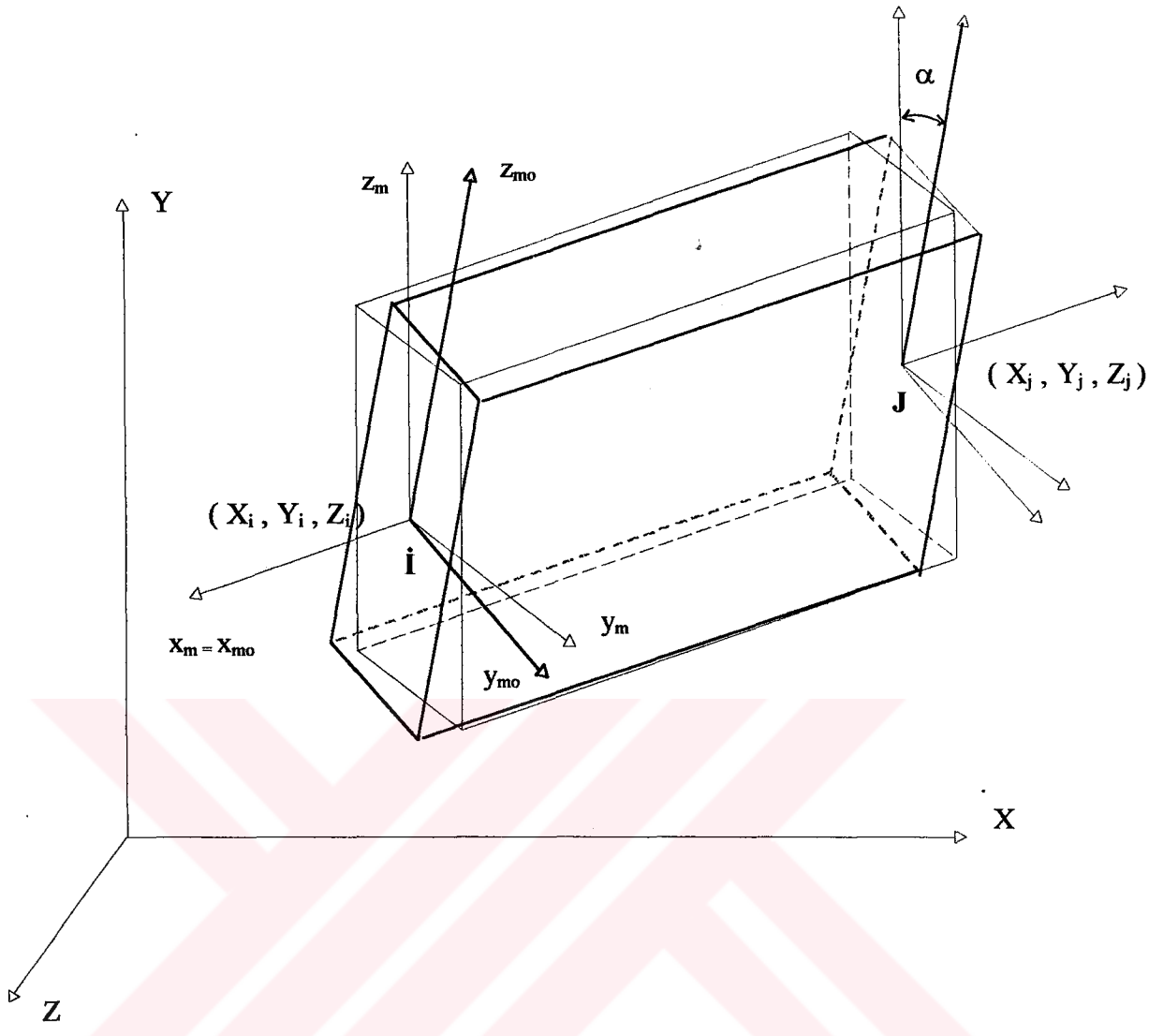
Çubuk lokal eksenleri çubuğun ötelenmesi ile ötelenir ve çubuğun dönmesi ile dönerler. Sistemi oluşturan bir çubuğun eksen takımı ile diğer bir çubuğun eksen takımı arasında hiç bir beraberlik yoktur. Çubuklar sistem içinde farklı farklı konumlarda bulunabilirler.

4.3.2. Global Eksen Takımı

Taşıyıcı sistemin denge denklemlerini yazabilmek ve bir düğüm noktasında birleşen çubukların uç kuvvet ve deplasmanlarını tek bir koordinat eksenine döndürebilmek için bütün sisteme ait olmak üzere, sağ el vida kaidesine uygun olarak seçilen, karşılıklı birbirine dik X, Y ve Z eksenlerine global eksen takımı denir. Global eksenlerin doğrultuları tamamen çubuk eksenlerinden bağımsızdır ve çubuğun konumu ile ilgileri yoktur. Bu eksenler analizi yapılan taşıyıcı sitemin üzerine çizilmiş gibidir.

Bir çubuğun asal atalet eksenleri olan y_m ve z_m eksenlerinin en kesit düzlemi içindeki yerini belirleyebilmek için, en kesit içinde, dönmeden sabit duran yeni bir eksen takımına ihtiyaç vardır. Çünkü y_m ve z_m eksenleri en kesit düzlemi içinde, çubuk boy ekseni olan x_m etrafında 360^o dönerek herhangi bir konumu işgal edebilirler. Bu özel konumu gösteren eksenler x_{mo}, y_{mo} ve z_{mo} ile şekil 4.6 'da gösterilmiştir.

Hangi eksen takımında çalışılırsa çalışılsın, koordinat eksenlerinin pozitif yönlerinde elde edilmiş olan uç kuvvetleri ve uç ötelenmeleri pozitif olur. Momentlerin ve dönmelerin işaretleri sağ el vida kuralına göre belirlenir.



Şekil 4.6 Uzay çerçeve çubuğu eksen takımları (Özel konum)

4.4. Çubuk Transformasyon Matrisi

Daha önce de belirtildiği gibi, uzay çerçeve elemanında, bir düğüm noktasında üç ötelenme ve üç dönme olmak üzere toplam altı hareket serbestisi vardır. Dönme vektörleri ile ötelenme vektörleri birbirleri ile ilgili değildir. Aynı şekilde kuvvet vektörleri de moment vektörlerinden bağımsızdır.

Kuvvet vektörlerinin transformasyonu için denklem 4.8 'deki gibi

$$\begin{Bmatrix} P_1 \\ P_2 \\ P_3 \end{Bmatrix}_{x_m\ y_m\ z_m} = [t] \begin{Bmatrix} P_1 \\ P_2 \\ P_3 \end{Bmatrix}_{XYZ} \qquad (4.8)$$

ve momentlerin transformasyonu aynı şekilde denklem 4.9 'daki gibidir.

$$\begin{Bmatrix} P_4 \\ P_5 \\ P_6 \end{Bmatrix}_{x_m\ y_m\ z_m} = [t] \begin{Bmatrix} P_4 \\ P_5 \\ P_6 \end{Bmatrix}_{XYZ} \qquad (4.9)$$

Bu iki transformasyon işlemi tek bir denklem takımı halinde yazılmak istenirse, köşegen dışı bloklar sıfır olmak üzere 4.10 ifadesi elde edilir.

$$\begin{Bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_5 \\ P_6 \end{Bmatrix}_{x_m\ y_m\ z_m} = \begin{bmatrix} [t] & [0] \\ [0] & [t] \end{bmatrix} \begin{Bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_5 \\ P_6 \end{Bmatrix}_{XYZ} \qquad (4.10)$$

Bir uzay çerçeve çubuğunun uçlarındaki on iki adet hareket serbestisinin hepsinin birden, toplu halde transformasyonunu sağlayabilmek için

$$\underset{(12\,x\,1)}{\{P\}_{x_m\ y_m\ z_m}} = \underset{(12\,x\,12)}{[T]}\ \underset{(12\,x\,1)}{\{P\}_{XYZ}} \qquad (4.11)$$

ifadesi kullanılır. Burada [T] ` ye çubuk transformasyon matrisi denir. Transformasyon matrisi, köşegen boyunca, [t] doğrultu kosinüsleri matrislerini içerir. Uzay çerçeve çubuğu için transformasyon matrisi [T], aşağıdaki gibidir.

$$[T] = \begin{bmatrix} [t] & [0] & [0] & [0] \\ [0] & [t] & [0] & [0] \\ [0] & [0] & [t] & [0] \\ [0] & [0] & [0] & [t] \end{bmatrix}_{12x12} \qquad (4.12)$$

[t] doğrultu kosinüsü matrisinin genel ifadesi, şekil 4.6 'da gösterilen ve x_m etrafında oluşan α dönmesini de içerecek şekilde aşağıda belirtilmiştir. Burada L uzay çerçeve çubuğunun boyu olmak üzere, Cx, Cy ve Cz ifadeleri (4.14) bağıntısındaki gibidir.

$$[t] = \begin{bmatrix} Cx & Cy & Cz \\ \dfrac{-CxCy\cos\alpha - Cx\sin\alpha}{\sqrt{Cx^2+Cz^2}} & \sqrt{Cx^2+Cz^2}\cos\alpha & \dfrac{-CyCz\cos\alpha + Cx\sin\alpha}{\sqrt{Cx^2+Cz^2}} \\ \dfrac{CxCy\sin\alpha - Cz\cos\alpha}{\sqrt{Cx^2+Cz^2}} & -\sqrt{Cx^2+Cz^2}\sin\alpha & \dfrac{CyCz\sin\alpha + Cx\cos\alpha}{\sqrt{Cx^2+Cz^2}} \end{bmatrix} \qquad (4.13)$$

$$Cx = \frac{Xj - Xi}{L}$$
$$Cy = \frac{Yj - Yi}{L} \qquad (4.14)$$
$$Cz = \frac{Zj - Zi}{L}$$

4.5. Üçlü Çarpım Transformasyon Formülleri

Bir düğüm noktasında birleşen çubukların rijitlik terimlerini toplayabilmek için, o çubukların rijitlik matrislerinin ortak bir eksen takımına göre yazılmış olması gerekir. Yani, çubuk rijitliklerini süperpoze edebilmek için, bütün rijitlik matrisleri hep aynı doğrultulardaki serbestlik derecelerine karşılık gelmelidir. Halbuki şekil 4.4 'de elde edilen rijitlik matrisi sadece çubuk eksenlerine göredir ve bir noktada birleşen çubukların lokal çubuk eksenleri birbirlerinden farklı doğrultulardadır. Bu durumda, çubuk uçlarında farklı doğrultularda oluşan deformasyon ve kuvvet vektörlerini ortak bir eksen takımına transforme etmek gerekir. Dolayısı ile problem lokal çubuk eksenlerine göre bilinen rijitlik matrislerinin bu ortak eksen takımına nasıl aktarılacağıdır.

Bu probleme çözüm getirebilmek için enerji yönteminden faydalanmak gerekir. Bir çubuğun uç kuvvetlerinin karşılık gelen deplasmanlar üzerinde yaptıkları harici işin toplamı sabittir, eksen takımı ne olursa olsun değişmez.. Dolayısı ile global eksenlerde verilen çubuk kuvvetlerinin, kendi doğrultularındaki yaptığı deplasmanlar üzerinde yaptığı iş, çubuk eksenlerine göre verilmiş eşdeğer kuvvet takımının karşılık gelen deplasmanlar üzerinde yaptığı işe eşittir. Yani,

$$1/2\ (p_1 d_1 + p_2 d_2 + ...)_{XYZ} = 1/2\ (p_1 d_1 + p_2 d_2 + ...)_{x_m y_m z_m} \qquad (4.15)$$

Matris notasyonu kullanır ve 1/2 leri kısaltırsak

$$\{p\}^T_{XYZ}\ \{d\}_{XYZ} = \{p\}^T_{x_m y_m z_m}\ \{d\}_{x_m y_m z_m} \qquad (4.17)$$

yazılır. Burada $\{p\}^T_{XYZ}$ = XYZ global eksen takımına göre verilmiş kuvvet vektörünün transpozudur. Rijitlik denklemini bir kere global eksenlerde

$$\{ p \}_{XYZ} = [k]_{XYZ} \{ d \}_{XYZ} \qquad (4.18)$$

bir kere de çubuk lokal eksenlerinde

$$\{ p \}_{XmYmZm} = [k]_{XmYmZm} \{ d \}_{XmYmZm} \qquad (4.19)$$

şeklinde yazar ve her iki denklemde eşitliğin her iki tarafının transpozunu alır ve denklem 4.17 de yük vektörlerinin yerine koyarsak ;

$$\{d\}^T_{XYZ}[k]_{XYZ}\{d\}_{XYZ} = \{d\}^T_{x_m y_m z_m}[k]_{x_m y_m z_m}\{d\}_{x_m y_m z_m} \qquad (4.20)$$

ifadesi elde edilir. $\{d\}_{x_m y_m z_m}$ nin $\{d\}_{XYZ}$ cinsinden değerini denklem 4.20 nin sağ tarafında yerine koyarsak

$$\{d\}^T_{XYZ}[k]_{XYZ}\{d\}_{XYZ} = \{d\}^T_{XYZ}[T]^T[k]_{x_m y_m z_m}[T]\{d\}_{XYZ} \qquad (4.21)$$

eşitliği elde edilir ve eşitliğin her iki tarafında aynı olan terimlerin sadeleştirilmesi ile aşağıdaki üçlü transformasyon formülasyonu elde edilir.

$$[k]_{XYZ} = [T]^T[k]_{x_m y_m z_m}[T] \qquad (4.22)$$

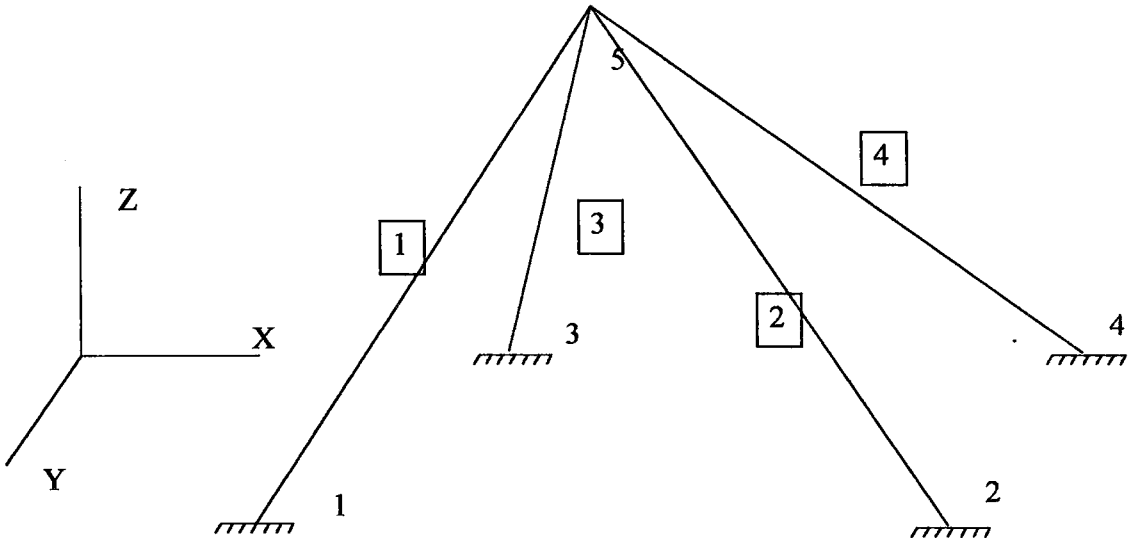
Bu formülasyon yardımı ile çubuk lokal eksenleri doğrultusunda verilmiş bir rijitlik matrisini, global sistem eksenlerine çevirmek mümkündür. Kodlama tablosu yönteminden bahsedilirken de açıklanacağı gibi, sistem rijitlik matrisi süperpozisyon yöntemi ile hesaplanırken, doğrudan doğruya global sistem eksenlerine çevrilmiş rijitlik matrisleri kullanılacaktır.

4.6. Kod Numaraları Yöntemi ve Sistem Deplasmanlarının Hesabı

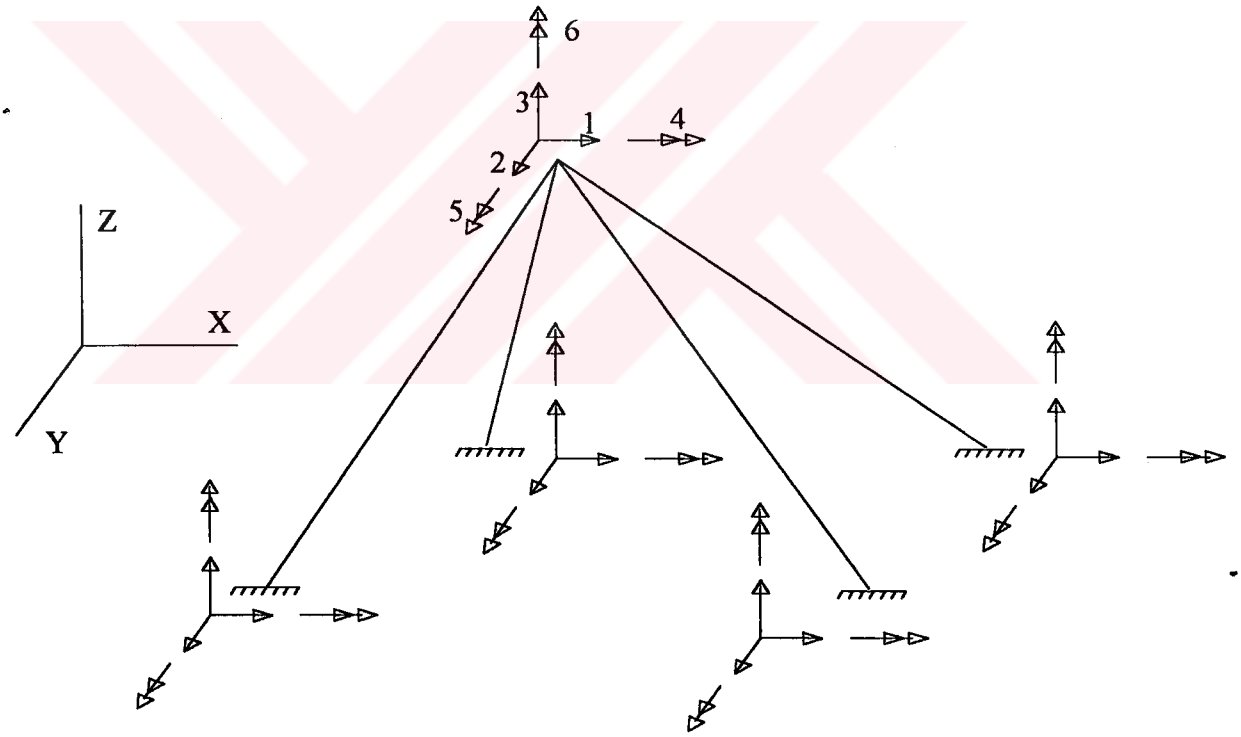
Kod numaraları yöntemi taşıyıcı sistemin rijitlik matrisini doğrudan elde etmek için kullanılır. Bu yöntem herhangi bir düğüm noktasında birleşen çubukların rijitliklerinin toplanması işlemini son derece kolaylaştırır ve üstelik programlamaya da son derece elverişlidir.

Bir çubuğun baş ve son düğüm noktalarının numaraları, o çubuğun taşıyıcı sistem içinde işgal ettiği yeri belirtir. Uzay çerçeve çubuğunun her düğüm noktasında altı adet serbestlik derecesi yani bilinmeyen deplasman vardır. Düğüm noktalarında varlığı kabul edilen bu deplasmanlar problemin bilinmeyenleridir ve birden başlayarak düğüm noktaları numaralarını takiben sıra ile numaralanırlar. Böyle bir numaralamanın varlığı sadece kabul edilir yoksa, kağıt üzerinde bu numaraları göstermenin bir gereği yoktur. Çünkü sistem matrisinin terimleri ve çözümde elde edilen bilinmeyenlerin sırası deplasman numaralarını aynen takip eder. Her düğüm noktasında önce ötelenmelerin sonra dönmelerin numaralandığı ve eksenlerde alfabetik sıraya uyulduğu kabul edilir.

Deplasmanları sıra ile numaralanmış bir taşıyıcı sistemde, herhangi bir çubuğun baş ve sonuna isabet eden düğüm noktalarındaki deplasman numaralarının, yan yana yazılması ile elde edilen rakama, o çubuğun kod numarası denir. Çubuk elemanlarda, kod numarasındaki deplasman numaralarının adedi, sistemin tipik bir düğüm noktasındaki serbestlik derecesinin iki katıdır. Çünkü bir çubukta iki uç vardır ve her deplasman numarasının bir hane işgal ettiği kabul edilirse, bir uzay çerçeve çubuğunun kod numarasında 12 hane vardır. Her bir hanede, o haneye tekabül eden deplasman numarası yazılıdır. Eğer bir düğüm noktasında mesnet varsa veya herhangi bir doğrultuda deplasman sıfır ise, o haneye sıfır yazılır. Örnek olarak bir uzay çerçeve sisteminin deplasmanlarının numaralanması ve kod numaraları şekil 4.7 'de gösterilmiştir. Bu şekle karşılık gelen kodlama tablosu da hazırlanmıştır. 1, 2, 3 ve 4 numaralı düğümler ankastre olduğu için hareket oluşmamaktadır. Dolayısı ile kodlama tablosunda karşılık gelen değerleri sıfırdır.



Şekil 4.7 (a) Eleman ve düğüm numaraları



Şekil 4.7 (b) Kod numaraları ve hareket serbestileri

Çubuk No	Düğüm No (I -> J)	Kod No 1,2,3,4,5,6,7,8,9,10,11,12
1	1-5	0,0,0,0,0,0,1,2,3,4,5,6
2	2-5	0,0,0,0,0,0,1,2,3,4,5,6
3	3-5	0,0,0,0,0,0,1,2,3,4,5,6
4	4-5	0,0,0,0,0,0,1,2,3,4,5,6

Taşıyıcı sistemin herhangi bir düğüm noktasında birleşen bütün çubukların o düğüm noktasında, herhangi bir deplasman doğrultusuna karşılık gelen rijitlik terimlerinin toplamı, taşıyıcı sistemin o deplasman doğrultusundaki rijitlik terimini verir. Ancak böyle bir toplama yapabilmek için, global sistem eksenlerine transforme edilmiş çubuk matrislerini kullanmak gerekir.

Şekil 4.7 'de gösterilen basit uzay çerçeve örneğini ele alalım. Görüldüğü gibi bu sistemde, hepsi beşinci düğümde olmak üzere toplam altı adet hareket serbestisi vardır. Diğer düğümler ankastre olduğu için hareket serbestisi oluşmamaktadır. Yani karşılık gelen kod numaraları sıfırdır. Bu problemin çözümünde yapılması gereken ilk adım, sistemi oluşturan bütün elemanlar için şekil 4.4 'de formülasyonu verilen, lokal eksenlere ait eleman rijitlik matrislerini hesaplamaktır. Bundan sonraki adım denklem 4.12 'de verilen transformasyon matrisini kullanarak, bütün elemanlar için üçlü çarpım transformasyonu ile global sistem eksenlerine ait eleman rijitlik matrislerini hesaplamaktır. Örnek olarak seçilen yapı sisteminde toplam altı adet hareket serbestisi bulunduğundan, sistemi temsil eden rijitlik matrisi 6 x 6 boyutunda olacaktır. Yapılması gereken işlem, bütün elemanlar için tek tek hesaplanan global eksenlerdeki rijitlik matrislerini ve bu elemanlara ait kod numaralarını kullanarak sistem rijitlik matrisini hesaplamaktır. Bu işlemi aşağıdaki gibi özetlemek mümkündür.

Kod numarasındaki hane numaralarının yan yana getirilmesinden ortaya çıkan rakam çifti, global eksenlerdeki çubuk rijitlik matrisinden alınacak terimin satır ve sütün numarasını, bu hanelere karşılık gelen deplasman numaralarının yan yana getirilmesinden ortaya çıkan rakam çifti ise, çubuk rijitlik matrisinden alınan terimin, sistem rijitlik matrisinde yerleştirileceği yerin satır ve sütun numarasını verir.

Sonuçta elde edilecek sistem rijitlik matrisi aşağıdaki şekilde oluşacaktır. k_{ij} terimlerinde i ilgili elemana ait matristeki satır numarasını, j sütun numarasını, üst indis ise hangi elemana ait olduğunu göstermektedir. Simetrik matris olduğundan üst üçgen ayrıca gösterilmemiştir.

	1	2	3	4	5	6
1	$(k_{77})^1+(k_{77})^2+(k_{77})^3+(k_{77})^4$					
2	$(k_{87})^1+(k_{87})^2+(k_{87})^3+(k_{87})^4$	$(k_{88})^1+(k_{88})^2+(k_{88})^3+(k_{88})^4$				
3	$(k_{97})^1+(k_{97})^2+(k_{97})^3+(k_{97})^4$	$(k_{98})^1+(k_{98})^2+(k_{98})^3+(k_{98})^4$	$(k_{99})^1+(k_{99})^2+(k_{99})^3+(k_{99})^4$			
4	$(k_{10\,7})^1+(k_{10\,7})^2+(k_{10\,7})^3+(k_{10\,7})^4$	$(k_{10\,8})^1+(k_{10\,8})^2+(k_{10\,8})^3+(k_{10\,8})^4$	$(k_{10\,9})^1+(k_{10\,9})^2+(k_{10\,9})^3+(k_{10\,9})^4$	$(k_{10\,10})^1+(k_{10\,10})^2+(k_{10\,10})^3+(k_{10\,10})^4$		
5	$(k_{11\,7})^1+(k_{11\,7})^2+(k_{11\,7})^3+(k_{11\,7})^4$	$(k_{11\,8})^1+(k_{11\,8})^2+(k_{11\,8})^3+(k_{11\,8})^4$	$(k_{11\,9})^1+(k_{11\,9})^2+(k_{11\,9})^3+(k_{11\,9})^4$	$(k_{11\,10})^1+(k_{11\,10})^2+(k_{11\,10})^3+(k_{11\,10})^4$	$(k_{11\,11})^1+(k_{11\,11})^2+(k_{11\,11})^3+(k_{11\,11})^4$	
6	$(k_{12\,7})^1+(k_{12\,7})^2+(k_{12\,7})^3+(k_{12\,7})^4$	$(k_{12\,8})^1+(k_{12\,8})^2+(k_{12\,8})^3+(k_{12\,8})^4$	$(k_{12\,9})^1+(k_{12\,9})^2+(k_{12\,9})^3+(k_{12\,9})^4$	$(k_{12\,10})^1+(k_{12\,10})^2+(k_{12\,10})^3+(k_{12\,10})^4$	$(k_{12\,11})^1+(k_{12\,11})^2+(k_{12\,11})^3+(k_{12\,11})^4$	$(k_{12\,12})^1+(k_{12\,12})^2+(k_{12\,12})^3+(k_{12\,12})^4$

Bilinmeyen sistem deplasmanlarına ulaşmak için hesaplanması gereken bir diğer vektör de sistem yük vektörüdür. Bir taşıyıcı sistemin üzerine tesir eden dış yükler tesir ettikleri yer bakımından iki gurupta toplanabilirler. Birinci gurup yükler doğrudan doğruya sistemin düğüm noktalarına etki ederler ve sistem yük vektörüne direk olarak yerleştirilirler. İkinci gurup yükler ise elemanı oluşturan düğüm noktaları arasına, yani açıklıklara (lokal veya global eksenler doğrultusunda olabilirler) etki ederler. Bu ikinci tip yüklerin düğümler üzerindeki etkilerinin hesaplanması için, taşıyıcı sistemin bütün çubukları uçlarından ankastre kabul edilir ve çubuklar üzerindeki yüklerin ankastre uçlarda meydana getirdikleri ankastre reaksiyonları, bilinen formülasyonlarla hesaplanır. Hesaplanan bu reaksiyonların ters işaretlileri düğüm noktalarına doğrudan etki eden birinci gurup yükler ile toplanır. Transformayon kuralı burada da aynen geçerlidir. Eleman lokal eksenlerinde yapılan yüklemeler ve hesaplanan reaksiyonlar varsa, bu tesirler önceden hesaplanan transformasyon matrisleri yardımı ile global sistem eksenlerine transforme edilmelidir. Sonuçta (4.23) 'deki taşıyıcı sisteme ait rijitlik denklemi elde edilmiş olur. Denklemde n, sistemin toplam hareket serbestisi sayısını göstermektedir.

$$\{Q\}_{nx1} = [K]_{nxn} \{D\}_{nx1} \qquad (4.23)$$

Sistem deplasmanları ve sistem yük vektörü bilindiği için, { D } sistem deplasman vektörü, bilinen lineer denklem sistemi çözüm yöntemlerinden herhangi biri ile hesaplanabilir. Sonuçta kodlama tablosundaki numaralara karşılık gelen hareket serbestileri yönünde, etkiyen yüklemeler altında sistemin yapmış olduğu deplasmanlar elde edilmiş olur.

4.7. Çubuk Uç Kuvvetlerinin Hesabı

Yapı sisteminin çözümünde varılan son aşama ise çubukların uçlarında oluşan uç kuvvetlerinin hesaplanmasıdır. Bölüm 4.2'de çubuk uç kuvvetlerinin hesabından bahsedilmiş ve denklem 4.6'da da uzay çerçeve elemana ait rijitlik denklemi verilmiştir. Kodlama tablosu hazırlanırken hangi elemanın uçlarında hangi numaralı hareket serbestilerinin oluşacağı ile ilgili bilgiler, halihazırda mevcuttur. Bir önceki, sistem rijitlik denkleminin çözümü aşamasında, bu numaraları belli olan deplasmanlar hesaplandığı ve ilgili ankastrelik uç reaksiyonları da bilindiği için, yapılması gereken işlem uçlardaki hareket serbestisi değerlerine uygun olarak, uzay çerçeve elemanının deplasman ve ankastrelik uç kuvvetleri vektörlerini kodlamak, global eksen takımına transforme edilmiş eleman rijitlik matrisini de kullanarak denklem 4.7 ile çubuk uç kuvvetlerini hesaplamaktır. Fakat unutulmamalıdır ki hesaplanan uç kuvvetleri global sistem eksenleri doğrultusundadır. Eğer eleman lokal eksenlerinde oluşan reaksiyonlar bulunmak isteniyorsa, daha önceden hesaplanan çubuk transformasyon matrisleri yardımı ile bu reaksiyonlar lokal eleman koordinatlarına döndürülebilirler. Böylelikle rijitlik matrisi yöntemi ile yapı sisteminin statik analizi tamamlanmış olur.

5. HAZIRLANAN YAZILIMIN YAPISI VE ÇALIŞTIRILMASI

5.1 Kullanılan Programlama Dilinin Yapısı

Yapılan çalışmada amaç, sonlu elemanlar yöntemini esas alarak çözümleme yapan rijitlik matrisi metoduna kullanıcının tam anlamı ile hakim olmasını sağlamak, yöntemin anlaşılabilirliğine katkıda bulunmak, eleman ve sistem denklemlerine istenildiği anda ulaşımı sağlayarak, kontrollü bir biçimde yapı sisteminin statik analizini bilgisayar ile yaptırabilmektir.

Bu amaç doğrultusunda hazırlanmış pek çok yazılım halihazırda mevcuttur. Fakat bu yazılımların çoğu ticari birer paket program niteliğinde olduğu için çözümleme aşamalarından ziyade sadece sonuçları hazırlayıp sunmaktadır. Kullanıcının ara işlemlerden, eleman ve sisteme ait denklemlerin işleyişinden haberi olmamaktadır. Bazı yazılımlarda DOS ortamının sınırlı imkanlarını kullanarak yazıldıkları ve görsel bir takım işlevlerden yoksun oldukları için, yukarıda sözü edilen amaçlara tam anlamı ile ulaşmak mümkün olamamaktadır. Bu yüzden dünya üzerinde çok yeni olan ve görsel programlama olarak da adlandırılan teknik ile yazılım hazırlanmıştır. Yazılım dili olarak A.B.D ' de 1995 sonunda piyasaya sürülen BORLAND DELPHİ (Visual and Object Orianted Pascal For Windows) dili kullanılmıştır. Kullanılan programın windows ortamı için hazırlanmış olması sayesinde, windows ortamının imkanlarından faydalanmak mümkün olabilmiştir.

Delphi yapısı itibari ile kullanıcıya kendi fonksiyonlarını tanımlayıp, formüle edebilme imkanını vermektedir. Bunun anlamı şudur: kullanıcı sık sık kullandığı bir takım işlemler, hesaplamalar için kendi fonksiyonlarını yazıp, ana program içinde bu fonksiyon veya prosedürleri çağırarak işlemi tamamlayabilmektedir. Kullanıcı yazmış olduğu bu fonksiyonları ayrı bir dosya şeklinde (Delphi dilinde UNIT) muhafaza etmek suretiyle, başka yazılımlara da çağırarak, daha sonrası içinde faydalanabilme imkanına sahiptir. Yapılan çalışmada bu amaç doğrultusunda matris-matris çarpımları, matris-vektör çarpımları, lineer denklem sistemi çözümleri, elektronik tablolama işlemleri vb. için bu şekilde fonksiyonlar tanımlanmış ve ana program içinde kullanılacağı zaman sadece isminin çağrılması yoluyla gereken işlemler yaptırılabilmiştir.

Delphi dili ayrıca Excel ve Lotus`tan da bildiğimiz elektronik tabloların oluşturulabilmesine imkan vererek, hazırlanan matrislerin ekrana taşınması konusunda

kullanıcıya kolaylıklar sunmaktadır. Bunu yanında kullanıcı, dilin sağlamış olduğu esneklik sayesinde, kendi tuşlarını ve menülerini tanımlayabilme imkanına sahiptir. Bütün bu kolaylıklar ve elektronik tabloların sağlamış olduğu imkanlar sayesinde matrislere ve vektörlere ulaşmak ve istenildiği anda ekrana getirebilmek mümkün olmuştur.

5.2 Hazırlanan Yazılımın Yapısı ve Kullanımı

Yazılım, ana program ve üç adet unitten oluşmaktadır. Ana program yazılımın genel akışını içermektedir. Ana programın kullandığı unitler ise Systypes.PAS, Grdap3da.Pas ve Matap3da.PAS dosyaları olarak saklanmıştır. Elektronik tablolar (Delphi dilindeki adı ile GRID) ile ilgili işlemler ve uygulamalarla ilgili prosedür ve fonksiyonlar Grdap3da.PAS dosyasında bulunmaktadır. Sıklıkla kullanılan matris işlemleri için hazırlanan prosedür ve fonksiyonlar Matap3da.PAS dosyası içinde iken sisteme ve elemanlara ait matrislerin, değişkenlerin ve tiplerin tanımlandığı unit ise Systypes.PAS dosyasıdır.

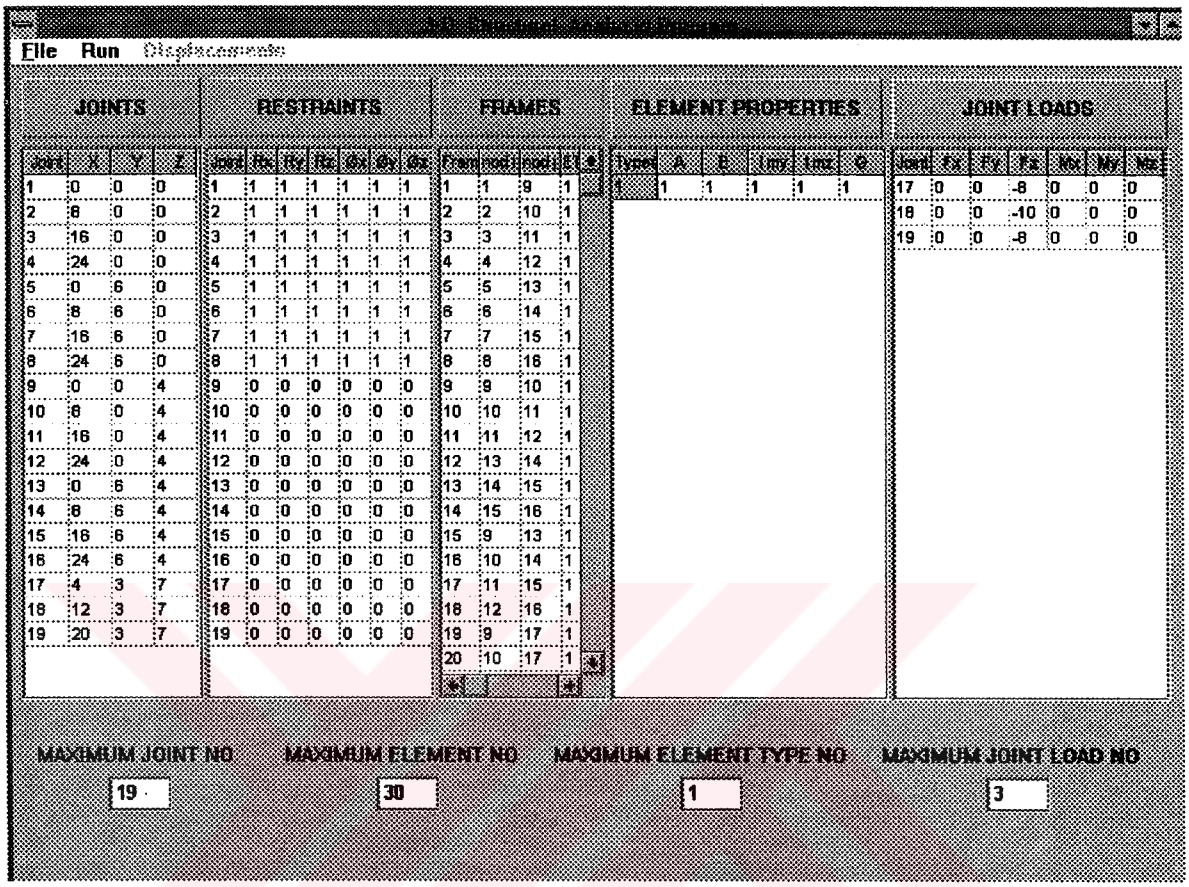
Program çağrıldığı zaman ekrana şekil 5.1 'deki görünümü ile gelmektedir. Daha sonra da görüleceği gibi hazırlanan yazılım, SAP90 programı ile tam uyumluluğun sağlanabilmesi ve daha önceden SAP90 programını kullananların yazılıma kolayca adapte olabilmesini sağlamak amacı ile veri bloklarına SAP90 programının bloklarının (JOINTS, FRAME, RESTRAINTS vb.) isimleri verilmiştir.

Programa öncelikle sistemi oluşturan toplam düğüm sayısının (Maximum Joint No), sistemdeki toplam eleman sayısının (Maximum Element No), sistemi oluşturan birbirinden farklı kaç tip eleman bulunduğunun (Maximum Element Type No) ve sistem düğümlerine etki eden kaç düğüm yükü bulunduğunun (Max JointLoad No) tanımlanması gerekmektedir. Bu tanımlamalar yapıldığı anda, yukarıda bu bölümler ile ilgili elektronik tablolarda, tanımlanan sayılar kadar satır açılmaktadır.

JOINTS (Düğümler) bölümünde, ilk kolon düğüm numarasına, sonraki kolonlar da sırası ile bu düğümlerin X, Y, Z koordinatlarına ayrılmıştır. Sistemdeki bütün düğüm noktalarının koordinat sistemindeki konumları bu bölümde girilir.

RESTRAINTS (Hareket Serbestileri) bloğunda ise tanımlanan düğümlerin tutulmuş yada tutulmamış olduğu ile ilgili tanımlamalar yapılmaktadır. Her düğüm noktasında global X, Y, Z eksenleri doğrultusunda, üç adet ötelenme ve üç adet dönme söz konusudur.

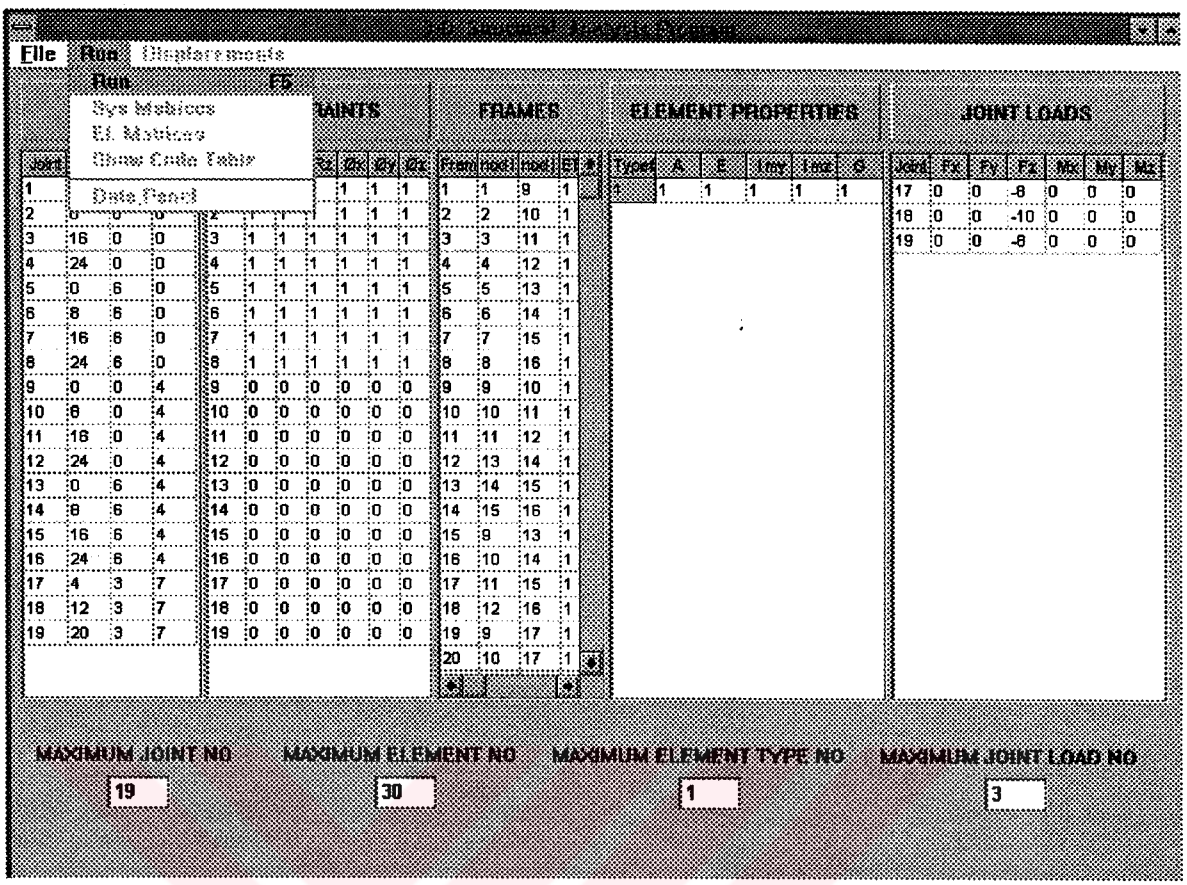
SAP90 programında olduğu gibi " 1 " tutulmuş (hareketten mahrum bırakılmış), " 0 " ise tutulmamış (hareket edebilir) hareket serbestilerini belirtmektedir.



Şekil 5.1 Programın çağırılması

ELEMENT PROPERTIES (Eleman Özellikleri) bloğunda elemanın alan, elastisite modülü, atalet momenti (malzeme eksenleri etrafında) ve kayma modülü değerlerinin girilmesi gerekir. Bu değerler daha sonra her eleman için eleman matrisleri hesaplanırken, program tarafından okunarak işleme tabi tutulur.

JOINT LOADS (düğüm yükleri) bloğunda da adından da anlaşılacağı üzere sistemi oluşturan düğüm noktalarına, global sistem eksenleri doğrultusunda yapılan yüklemeler girilmektedir. Bu yüklemeler sistem eksenleri doğrultusunda etkiyebilecek olan kuvvetler ve momentler olmak üzere toplam altı tanedir.



Şekil 5.2 Çözümlemeden önce menülerin görünümü

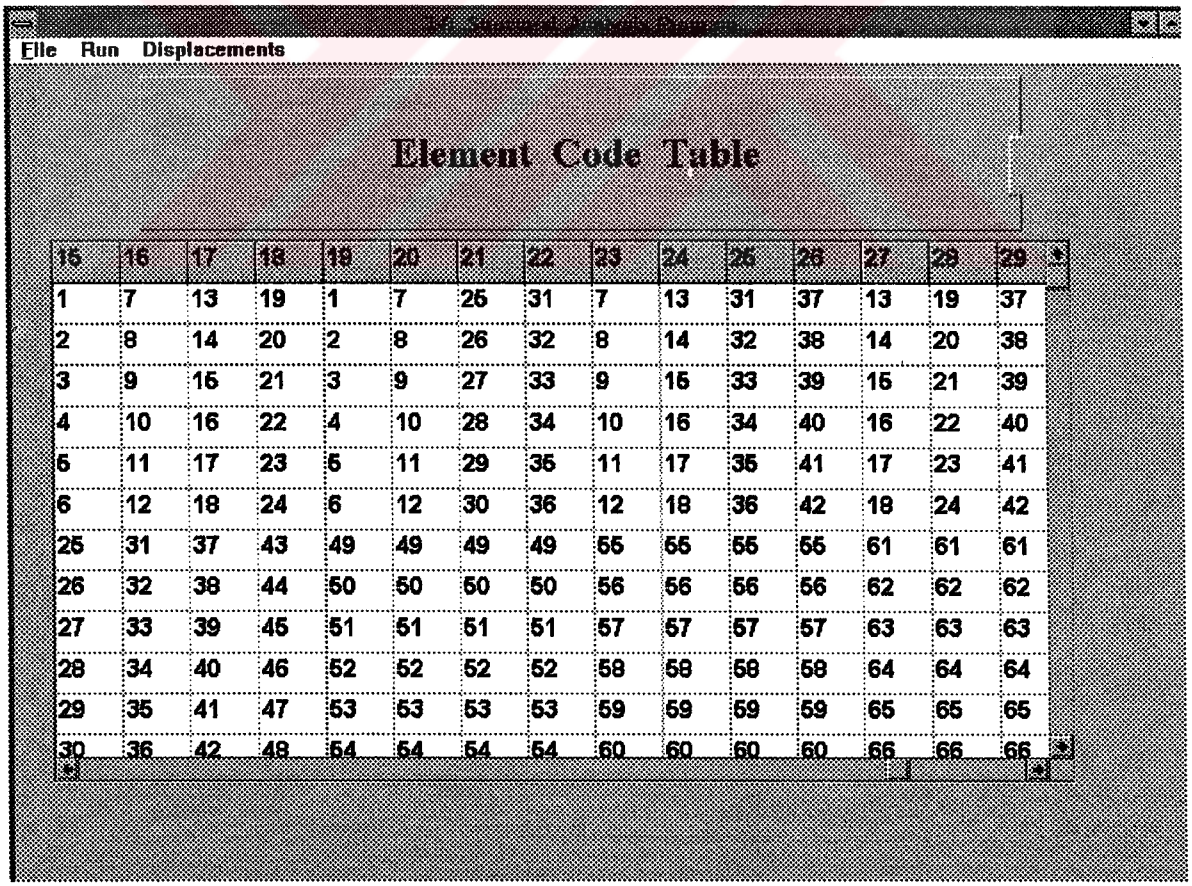
Gereken veriler girildikten sonra yazılımın çalıştırılması için şekil 5.2 'de menü çubuğu üzerinde görülen "Run" opsiyonunun altındaki "Run (F5)" tuşuna basılması gerekmektedir. Şekil 5.2 'de de görüldüğü gibi "RUN" tuşunun altındaki diğer tuşlar ve "Displacements" tuşu bu aşamada henüz aktif değildir, çünkü program şu ana dek bir hesaplama yapmamıştır.

5.2.1 Sistem deplasmanlarının hesaplatılması

"Run" tuşuna basılması ile birlikte program ilk olarak sistem bilinmeyenlerinin sayısını hesaplar. Program restraints bloğunu, kullanıcının düğümleri girmiş olduğu sıraya uyarak yukarıdan aşağı doğru tarar. Her sıfır değeri bir hareket serbestisine karşılık geldiği için, sıfır değerlerini her okuduğunda bilinmeyen sayısını bir arttırarak toplam bilinmeyen sayısını elde eder. Böylelikle sistem rijitlik matrisinin boyutları (Nmax x Nmax) belirlendiği

gibi, aynı zamanda hangi düğüm numarasının karşısında hangi numaralı bilinmeyenlerin bulunduğunu gösteren DOFGRID elektronik tablosu da yazılım tarafından hazırlanır. Bundan sonra yapılması gereken işlem, düğümlere atanan bu bilinmeyen numaralarının aynı zamanda ilgili elemanlara da atanmasıdır. FRAME veri bloğunda hangi elemanın hangi düğümler arasında bulunduğunu gösteren bilgilerden faydalanılarak, her elemanın başlangıç ve son düğümlerine ait deplasman veya bilinmeyen numaralarını gösteren kodlama tablosu program tarafından hazırlanır. Çözümleme işlemi tamamlandıktan sonra, "Run" menüsünün altındaki " Show Code Table " opsiyonunun seçilmesi ile de görülebileceği üzere kodlama tablosu şekil 5.3 'deki biçimde oluşturulur ve program tarafından NCODEGRID adı ile elektronik tablo olarak saklanır.

Şekil 5.3 'de de görüldüğü gibi en üst satır eleman numaralarına ayrılmıştır. Eleman numarasının altındaki kolon boyunca 12 adet değer sıralanmaktadır. Bunlar başlangıç ve son düğümlerine ait 3 adet ötelenme, 3 adet dönme olmak üzere, elemanın uçlarında oluşan bilinmeyen deplasmanların numaralarıdır.



15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
1	7	13	19	1	7	25	31	7	13	31	37	13	19	37
2	8	14	20	2	8	26	32	8	14	32	38	14	20	38
3	9	15	21	3	9	27	33	9	15	33	39	15	21	39
4	10	16	22	4	10	28	34	10	16	34	40	16	22	40
5	11	17	23	5	11	29	35	11	17	35	41	17	23	41
6	12	18	24	6	12	30	36	12	18	36	42	18	24	42
25	31	37	43	49	49	49	49	55	55	55	55	61	61	61
26	32	38	44	50	50	50	50	56	56	56	56	62	62	62
27	33	39	45	51	51	51	51	57	57	57	57	63	63	63
28	34	40	46	52	52	52	52	58	58	58	58	64	64	64
29	35	41	47	53	53	53	53	59	59	59	59	65	65	65
30	36	42	48	54	54	54	54	60	60	60	60	66	66	66

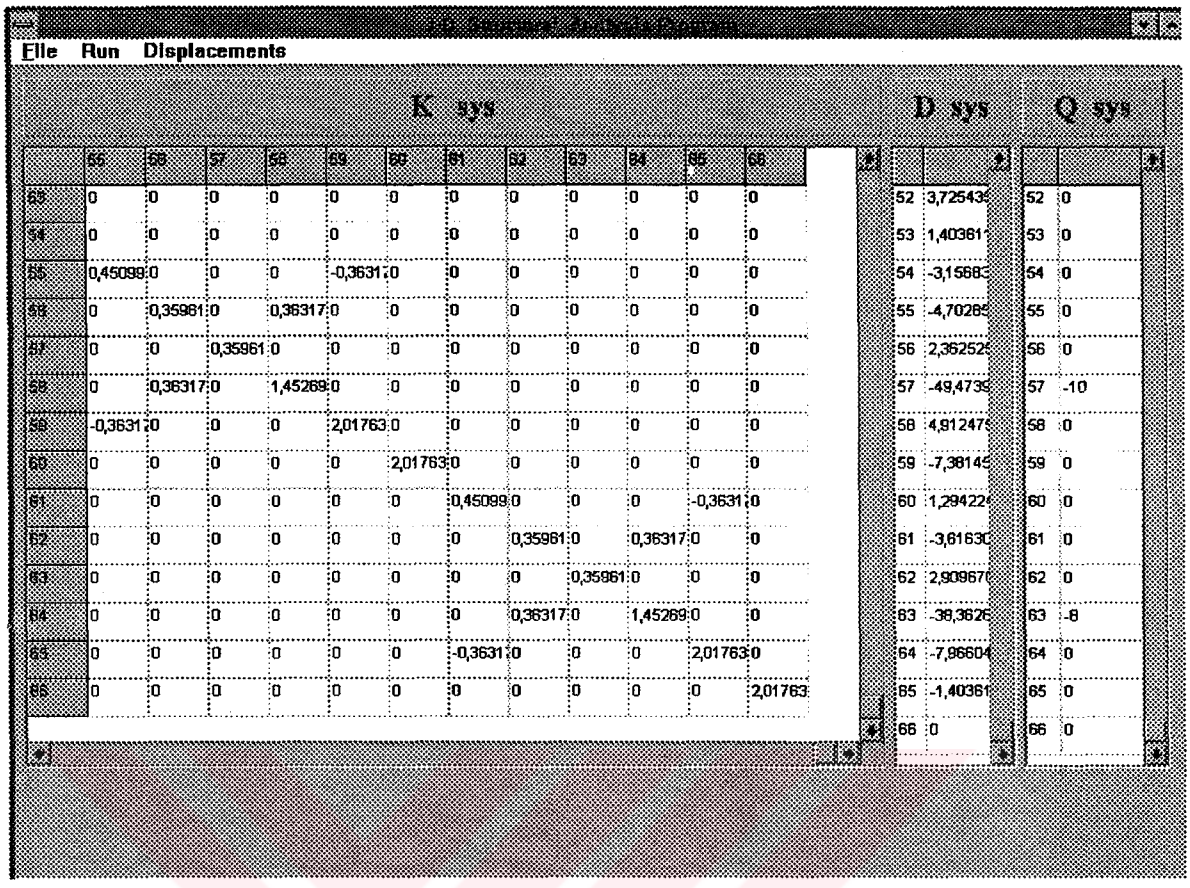
Şekil 5.3 Kodlama tablosunun görünümü

Kodlama tablosunun hazırlanmasından sonra program, veri olarak girilen eleman özelliklerini ve düğüm koordinatlarını kullanarak, (procedüre CreateElementListGrid) her eleman için alan, elastisite modülü, atalet momentleri, kayma modülü, boy ve doğrultu cosinüsleri bilgilerinin saklandığı "ElementListGrid" elektronik tablosunu hazırlar. Böylelikle her elemana ait, ilerki hesaplamalarda kullanılabilecek eleman özellikleri listelenmiş olur.

Program, hazırlamış olduğu "ElementListGrid" elektronik tablosunu kullanarak öncelikle her eleman için eleman koordinatlarındaki rijitlik matrislerini hesaplar. Hesaplanan rijitlik matrislerini, elemanın sistem içindeki yerine bağlı olarak sistem eksenlerine taşıyabilmek için, her elemana ait transformasyon matrislerini hesaplar. Transformasyon matrislerinin transpozelerinin hesabı ve üçlü çarpım yolu ile, elemanlara ait sistem koordinatlarındaki rijitlik matrisleri program tarafından bulunur. Bundan sonraki adım kodlama tablosunu ("NcodeGrid" elektronik tablosunu) kullanarak, eleman uçlarında oluşan bilinmeyen numaralarına göre, sisteme ait genel rijitlik matrisinin hesaplanmasıdır. Böylelikle boyutları, yapı sisteminde oluşan toplam hareket serbestisi sayısı tarafından belirlenen, bütün yapıyı temsil eden sistem rijitlik matrisi elde edilmiş olur. Kullanıcı çözümleme işlemini tamamladıktan sonra sistem rijitlik matrisini, "Run" menüsü altındaki "Sys Matrices" opsiyonunu seçerek görebilir. Ekrana gelen görüntü şekil 5.4 'de gösterilmiştir.

Bölüm 4'de teorisinden de bahsedildiği gibi, hesaplatılması gereken, kodlama tablosunda karşılık gelen numaralar ile ifade edilen deplasmanlardır. Dolayısı ile yapılması gereken işlem lineer denklem sistemi çözümüdür. Varılan aşamada sistemi temsil eden rijitlik matrisi bilinmektedir. Lineer denklem sistemi çözümüne geçebilmek için, sistem yük vektörünün hesaplatılması gerekmektedir.

Hazırlanan yazılım sadece düğüm noktalarına yük uygulama imkanı verdiği için, ara yüklemelerin sebep olduğu ankastrelik uç kuvvetleri vektörü, {f} bu program tarafından hazırlanmamaktadır. Dolayısı ile çözümleme işleminin sonunda, eleman matrisleri çağrılırken {f} vektörü ekranda görünmemektedir.

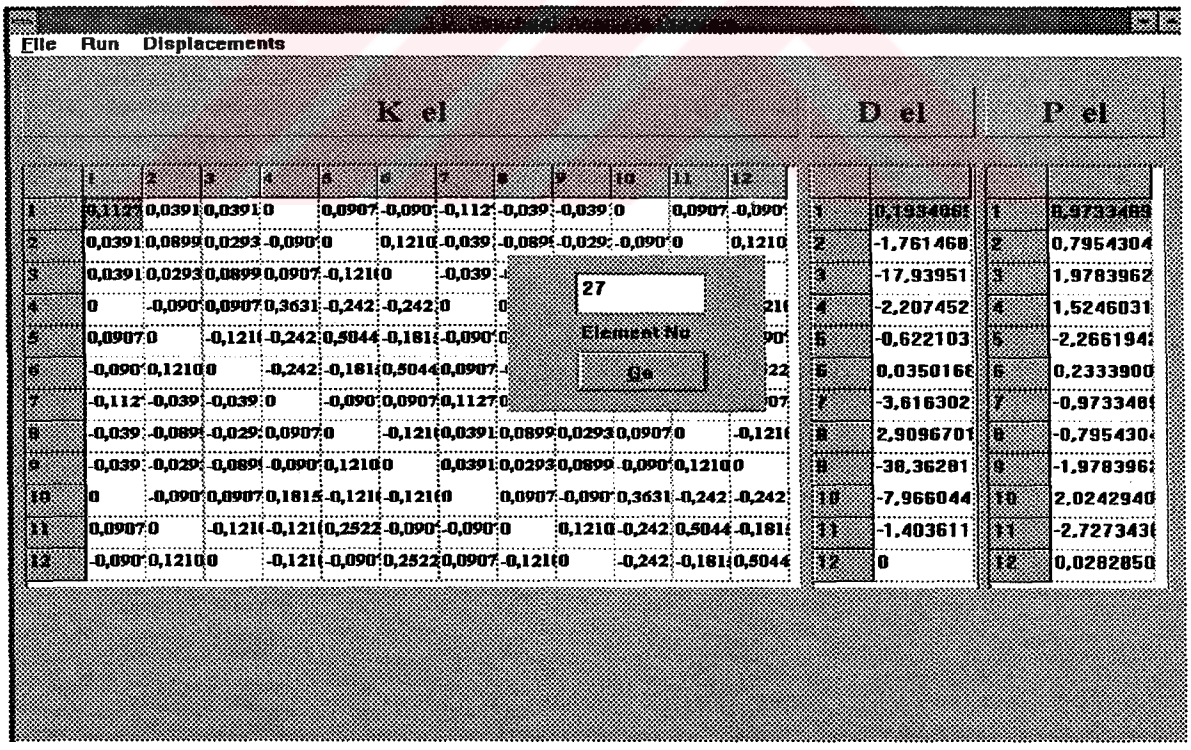


Şekil 5.4 Sistem rijitlik matrisinin çağırılması

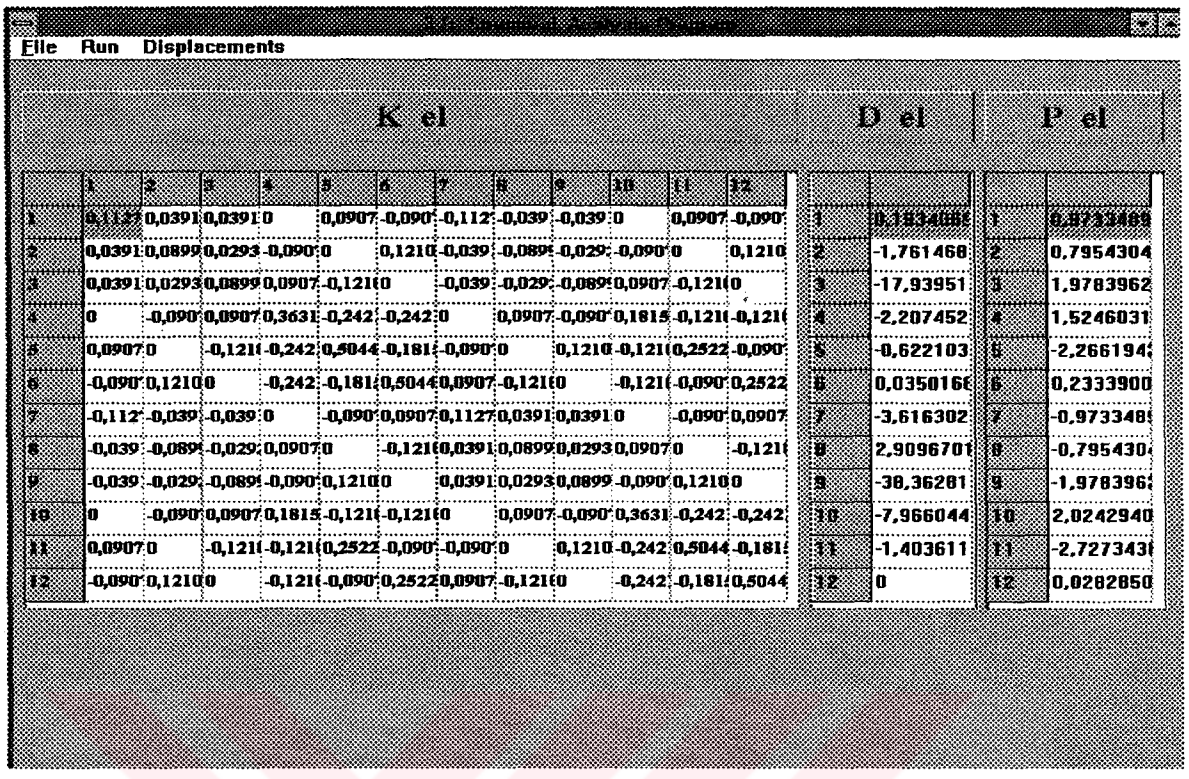
JOINT LOADS bloğunda tanımlanan düğüm yükleri, düğüm noktalarında oluşan hareket serbestisi numaralarına uygun olarak, bir vektör biçiminde tanımlanmalıdır. Veri olarak girilen düğüm yükleri, eğer uygulandıkları yönde bir hareket serbestisi var ise, o hareket serbestisi numarasına karşılık gelecek şekilde yerleştirilir ve böylelikle sistem yük vektörü program tarafından elde edilir (Procedure GetSysJointLoadGrid). Şekil 5.4 'de sistem yük vektörü " Q sys " olarak görülmektedir. Hazırlanan yazılım lineer denklem sistemi çözümlemesinde Gauss-Eliminasyon yöntemini kullanmaktadır. Uygun satır-sütun işlemleri kullanılarak üst üçgen matrisi haline getirilen sistem rijitlik matrisi program tarafından çözülür ve sistem deplasmanları elde edilir. Hesaplatılan bu sistem deplasmanları da şekil 5.4 'de " D sys " kolonu ile ifade edilmektedir. Kullanıcı bu aşamada hangi hareket serbestisi numarasına, hangi deplasmanın karşılık geldiğini bu vektör üzerinde görebilir.

5.2.2 Çubuk Uç Kuvvetlerinin Hesaplatılması

Sitemi genel olarak bu şekilde ele almanın yanında, sistemi oluşturan elemanların tek tek analizi de yapı sisteminin tam çözümünün bir parçasıdır. Bu yüzden " Run " menüsünün altında yer alan " El. Matrices " opsiyonun seçilmesi ile elemanlara ait matrislere ulaşmak mümkün kılınmıştır. Bu opsiyonun seçilmesi durumunda, kullanıcının istediği eleman numarasını girebilmesi için ekrana aşağıdaki görüntü gelecektir (Şekil 5.5). Kullanıcı hangi elemanın matrislerini görmek istiyor ise o elemanın numarasını burada girip " Go "tuşuna basarak istediği bilgilere ulaşabilir. Seçilen elemana ait matrislerin görünümü şekil 5.6 'daki gibidir. " K el " ile ifade edilen matris, global sistem koordinatlarında elemana ait rijitlik matrisidir. Eleman uçlarında oluşan 12 adet hareket serbestisini de içerdiği için boyutları 12x12 dir. " D el " ile ifade edilen vektör ise lineer denklem sistemi çözümü ile elde edilen deplasmanların (bilinmeyenlerin) ilgili (seçilen) elemanın uçlarındaki hareket serbestisi numaralarına göre kodlanması ile elde edilir. " P el " matrisi ise çubukların uçlarında oluşan uç reaksiyonlarını göstermektedir.



Şekil 5.5 Seçilen eleman numarasının programa girilmesi



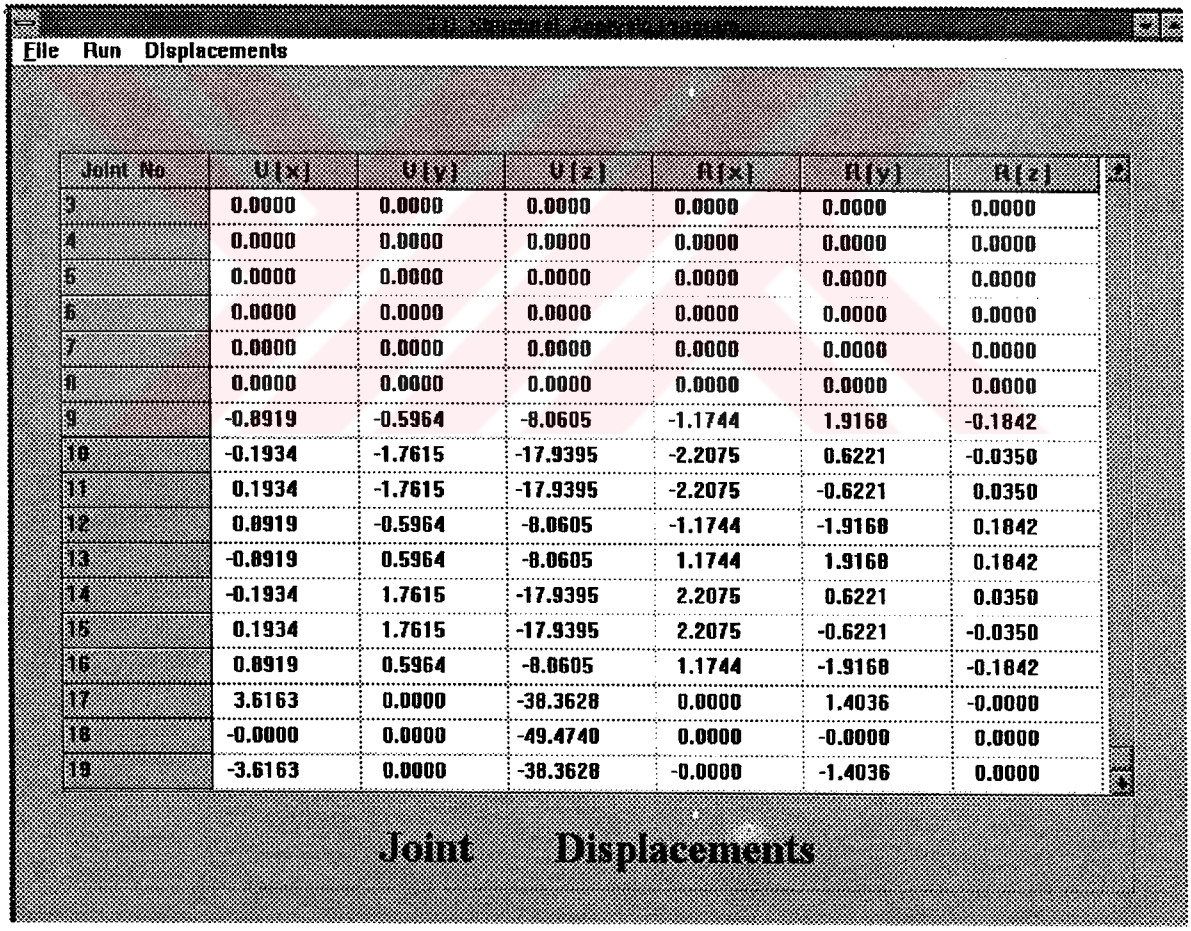
Şekil 5.6 Eleman matrislerinin görünümü

Fakat burada dikkat edilmesi gereken nokta şudur: hesaplanan çubuk uç kuvvetleri eleman eksenlerinde değil, sistem eksenlerindedir, yani X, Y, Z eksenleri etrafında, çubuk uçlarında oluşan kuvvetlerdir. Eleman lokal eksenleri etrafında oluşan reaksiyonların hesaplatılabilmesi için, bu reaksiyonların tekrar transforme edilmesi gerekmektedir. Hazırlanan yazılım şu andaki hali ile bu işlemi yapmamaktadır. Transformasyon gerektirmeyen (malzeme eksenleri ile sistem eksenleri çakışan) elemanlar için, elde edilen sonuçların doğruluğu, " P el " matrisi ile SAP90 tarafından hazırlanan *.F3F dosyalarının karşılaştırılması ile yapılabilir.

Eleman numarasının seçilmesi ve " Go " tuşuna basılması ile birlikte program daha önceden hazırlamış olduğu "ElemementListGrid" elektronik tablosundan seçilen elemana ait malzeme özelliklerini okur ve öncelikle transformasyon matrislerini de kullanarak, üçlü çarpım yolu ile global koordinatlardaki eleman matrisini hesaplar. Lineer denklem sistemi çözümü aşamasında hesaplanan deplasmanları, elemanların uçlarında oluşan hareket serbestisi numaralarını NcodeGrid elektronik tablosundan okuyarak, elemana ait 12x1

boyutundaki deplasman vektörüne kodlar. Bundan sonra yapılan işlem bölüm 4'de de teorisi açıklandığı gibi, hesaplanan eleman rijitlik matrisi ile kodlanan eleman deplasmanları vektörünün çarpımıdır. Sonuçta sistem koordinatlarında " P el " vektörü, yani eleman uç reaksiyonları elde edilmiş olur.

Elde edilen sonuçların Sap90 programının sonuçları ile karşılaştırılması aşamasında rahatlık sağlaması ve kullanıcının düğümlerde oluşan bilinmeyenleri (deplasmanları) toplu halde görebilmesi amacı ile menu çubuğu üzerinde "Displacements" opsiyonu hazırlanmıştır. "Displacements" opsiyonunun seçilmesi durumunda ekrana gelen görüntü şekil 5.7 'de gösterilmiştir. "joint No" başlığını taşıyan ilk kolon boyunca, sıra ile düğüm numaraları görülmektedir. İlgili düğüm numarası boyunca uzanan satırda ise sırası ile X, Y, Z global eksenlerinde oluşan ötelenmeler ve dönmeler yer almaktadır.



Şekil 5.7 Düğüm noktalarında oluşan sistem deplasmanlarının görünümü

"Displacements" opsiyonunun seçilmesi ile program lineer denklem sistemi çözümü aşamasında hesapladığı deplasmanları, düğümlerin sahip olduğu hareket serbestisi numaralarını "DofGrid" elektronik tablosundan okuyarak, yine ilgili düğümlere kodlar ve şekil 5.7 'de gösterilen elektronik tabloyu hazırlar.

Kullanıcı verilerini girmiş olduğu sistemi değiştirmek ve sonuçları yeniden değerlendirmek ister ise, "Run" menüsü altında yer alan "Data Panel" opsiyonunu seçip şekil 5.1 'de gösterilen veri girişi paneline geri dönebilir. Değiştirilen veriler ile çözümün başlatılması için "Run" komutunun tekrarlanması yeterlidir. Böylelikle yukarıda anlatılan işlem basamakları aynen tekrarlanır ve yeni sistem için çözümleme işlemi program tarafından tamamlanır. Program listesi de EK-1 de verilmiştir.

6. HAZIRLANAN YAZILIMIN KONTROLÜ VE SONUÇLAR

Yapılan çalışma neticesinde elde edilen yazılım, iki ve üç boyutlu yapı sistemlerinin statik analizine imkan vermektedir. Bu bölümde inşaat mühendisliği alanında sıklıkla karşılaşılan yapı sistemlerinden örnekler hazırlanmış ve bu örnekler hazırlanan yazılım kullanılarak çözdürülmüştür. İki ve üç boyutlu yapı sistemleri, çerçeve ve kafes sistemler olarak ayrılmış ve dört örnek hazırlanmıştır. Seçilen sistemler aşağıda belirtilmiştir.

1. Uzay Kafes Yapı Sistemi
2. Uzay Çerçeve Yapı Sistemi
3. Düzlem Kafes Yapı Sistemi
4. Düzlem Çerçeve Yapı Sistemi

Hazırlanan yazılım çubuk elemanlardan oluşan yapı sistemlerinin çözümüne imkan vermektedir. Elde edilen sonuçların karşılaştırılması SAP90 programı ile yapılmıştır. SAP90 programı çözümleme aşamasının sonunda *.F3F ve *.SOL uzantılı çıktı dosyaları hazırlamaktadır. *.F3F dosyalarında işaret takımına göre uyarlanmış, eleman lokal eksenleri etrafında oluşan çubuk uç kuvvetleri hazırlanmaktadır. *.SOL uzantılı çıktı dosyalarında ise sistemi oluşturan düğüm noktalarında meydana gelen deplasmanlar ve reaksiyonlar bulunmaktadır. Hazırlanan program, "Displacements" menü opsiyonunun seçilmesi ile, SAP90 tarafından hazırlanan *.SOL dosyasındaki gösterime benzer şekilde düğüm deplasmanlarını sunmaktadır. Sonuçların kontrolü, iki program tarafından hesaplanan bu deplasmanların karşılaştırılması yolu ile yapılacaktır. Bölüm 5'te de belirtildiği gibi "El. Matrices" opsiyonunun seçilmesi ile ekrana gelen eleman matrisleri, global sistem koordinatlarında, herhangi bir işaret takımına uygunluk gözetilmeden eleman uçlarında oluşan deplasman ve reaksiyonları göstermektedir. Bu yüzden SAP90 tarafından hazırlanan *.F3F dosyalarından farklıdır. Ancak lokal eksenler ile sistem eksenlerinin çakıştığı elemanlar için, *.F3F dosyasında elde edilen değerler ile bir karşılaştırma yapılabilir. Çözdürülen örneklerde uzunluklar metre cinsinden alınırken, E, A ve I değerleri "1" olarak alınmıştır. Deplasman değerleri buna göre değerlendirilmelidir, (kafes sistemlerde atalet momentleri sıfır alınmıştır)

6.1 Uzay Kafes Sistem Çözümü

Üç boyutlu bir kafes sistem örneği yazılım tarafından çözdürülmüştür. Hazırlanan veri dosyası şekil 6.1 'de gösterilmiştir. Aynı sistemin sap90 tarafından çözümü için hazırlanan data dosyası aşağıdaki gibidir.

```
SYSTEM
N=31 L=1
JOINTS
1   X=4     Y=0   Z=1.5
4   X=16    Y=0   Z=1.5
13  X=4     Y=12  Z=1.5
16  X=16    Y=12  Z=1.5  Q=1,4,13,16,1,4
17  X=6     Y=2   Z=0
19  X=14    Y=2   Z=0
23  X=6     Y=10  Z=0
25  X=14    Y=10  Z=0    Q=17,19,23,25,1,3
26  X=0     Y=4   Z=1.5
27  X=0     Y=8   Z=1.5
28  X=20    Y=4   Z=1.5
29  X=20    Y=8   Z=1.5
30  X=2     Y=6   Z=0
31  X=18    Y=6   Z=0
RESTRAINTS
1,31,1    R=0,0,0,1,1,1
1,4,3     R=1,1,1,1,1,1
13,16,3   R=1,1,1,1,1,1
26,27,1   R=1,1,1,1,1,1
28,29,1   R=1,1,1,1,1,1
FRAME
NM=1
1 A=1   I=0,0   E=1   G=1
1,1,2     G=2,1,1,1
```

4,26,5

5,5,6 G=2,1,1,1

8,8,28

9,27,9

10,9,10 G=2,1,1,1

13,12,29

14,13,14 G=2,1,1,1

17,1,5 G=3,1,1,1

21,26,27

22,5,9 G=3,1,1,1

26,28,29

27,9,13 G=3,1,1,1

31,17,18 G=1,1,1,1

33,30,20

34,20,21 G=1,1,1,1

36,22,31

37,23,24 G=1,1,1,1

39,17,20 G=2,1,1,1

42,20,23 G=2,1,1,1

45,1,17 G=2,1,1,1

48,26,30

49,5,20 G=2,1,1,1

52,8,31

53,30,9

54,20,10 G=2,1,1,1

57,31,29

58,9,23 G=2,1,1,1

61,23,14 G=2,1,1,1

64,17,6 G=2,1,1,1

67,2,17 G=2,1,1,1

70,17,5 G=2,1,1,1

73,5,30

74,6,20 G=2,1,1,1

77,28,31

78,30,27

79,20,9 G=2,1,1,1

82,31,12

83,10,23 G=2,1,1,1

86,23,13 G=2,1,1,1

LOADS

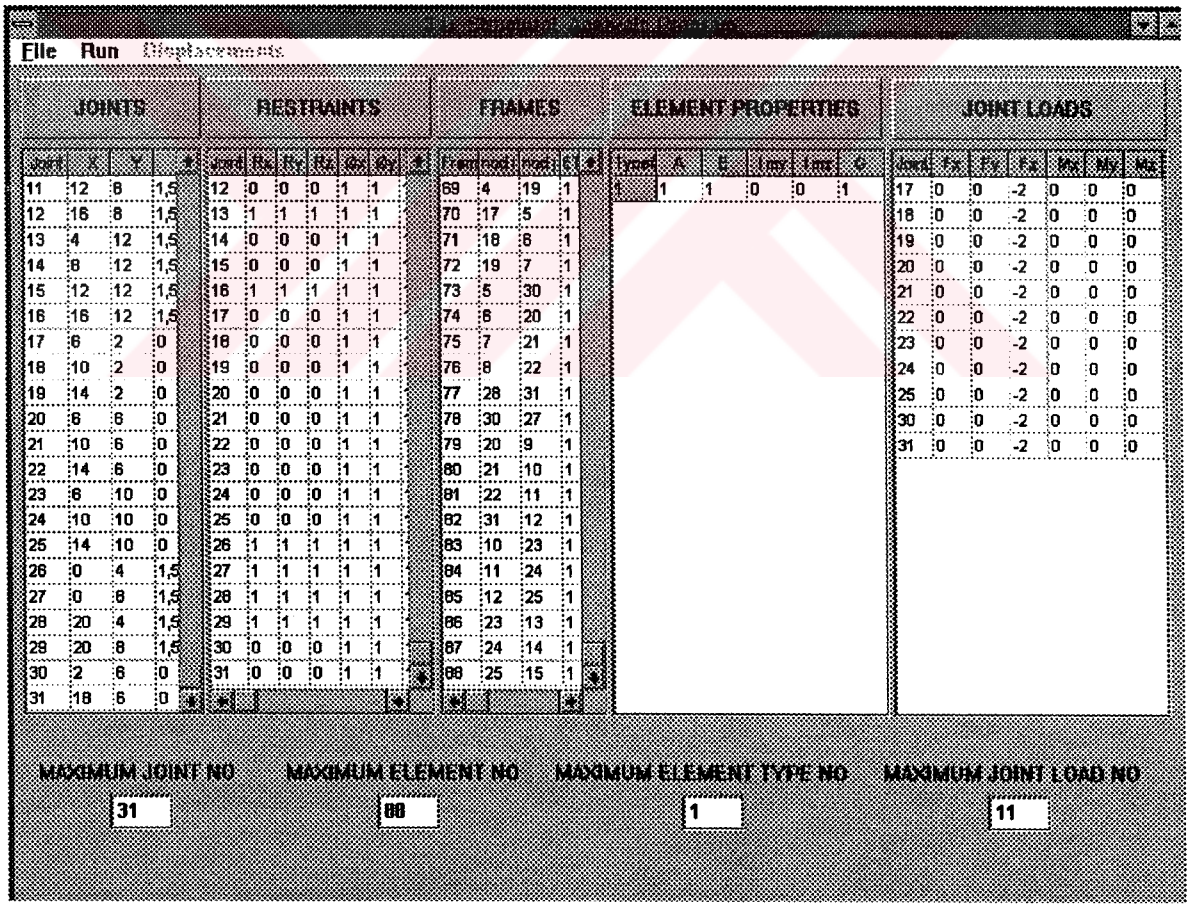
17,19,1 F=0,0,-2

20,22,1 F=0,0,-2

23,25,1 F=0,0,-2

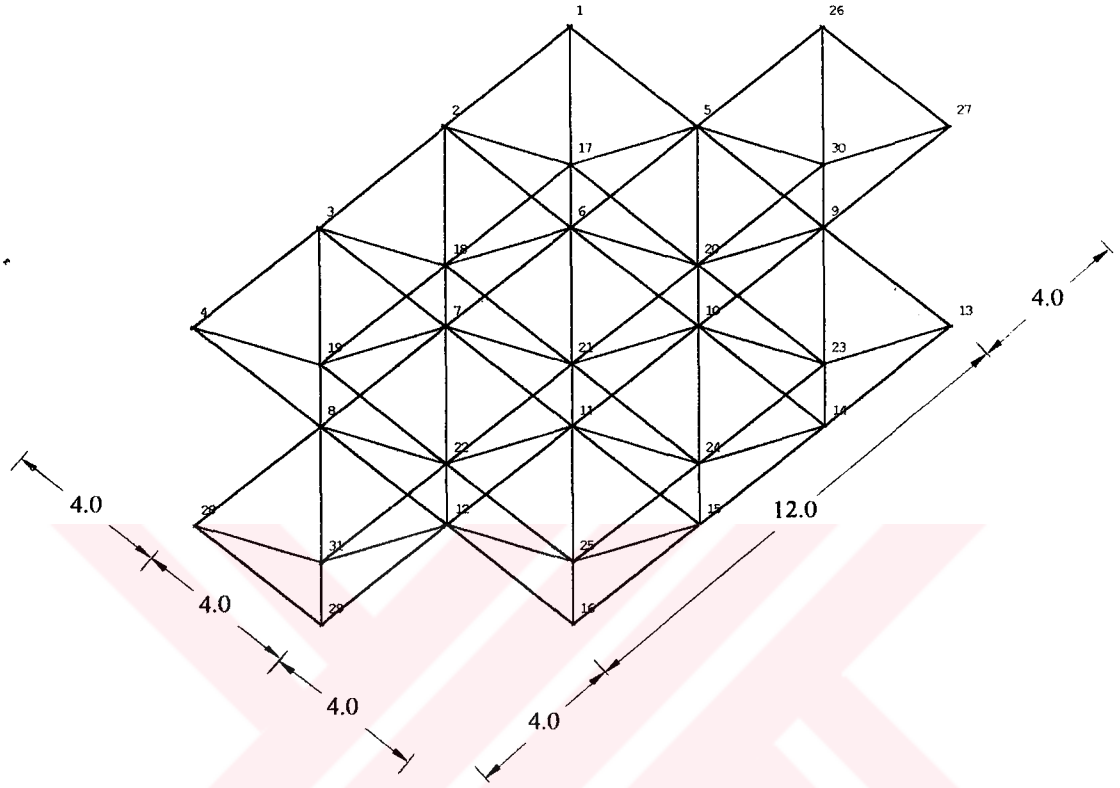
30 F=0,0,-2

31 F=0,0,-2

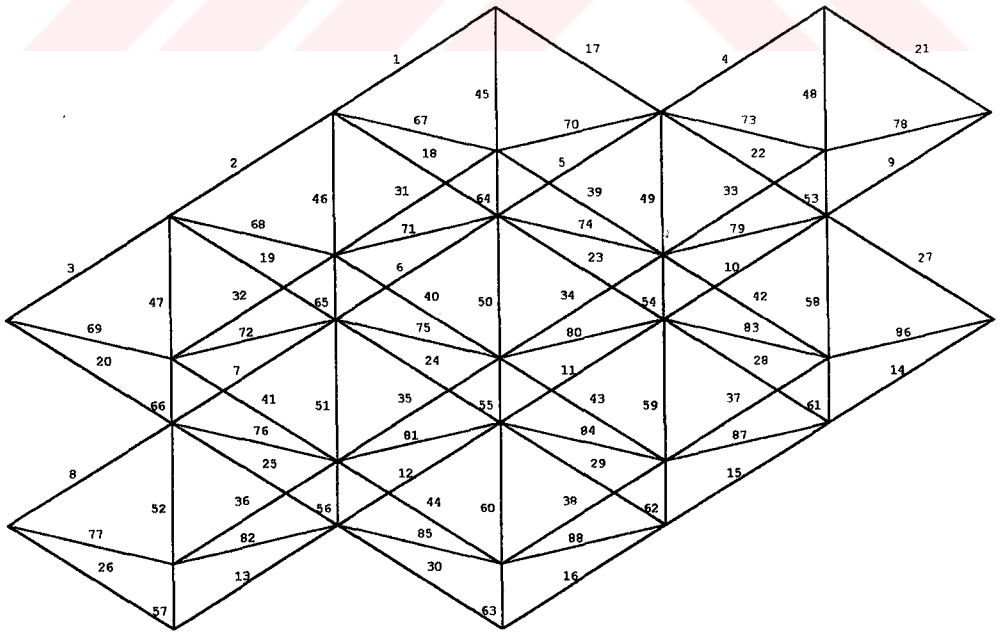


Şekil 6.1 Uzay kafes sistemi veri dosyası

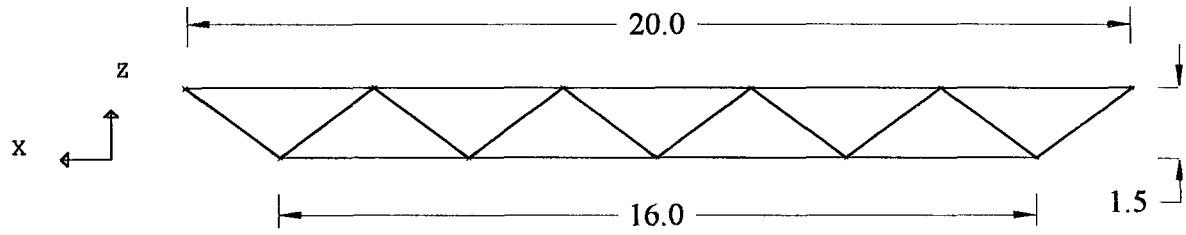
Modellemesi yapılan yapı sisteminin görünüşü aşağıdaki şekillerde ifade edilmiştir. Şekil 6.2 'de sistemin boyutları ve düğüm noktalarının yerleşimleri görülmektedir. Şekil 6.3 'de eleman numaralarının yerleşimi ve şekil 6.5 'te de sisteme etki eden yüklemeler gösterilmiştir.



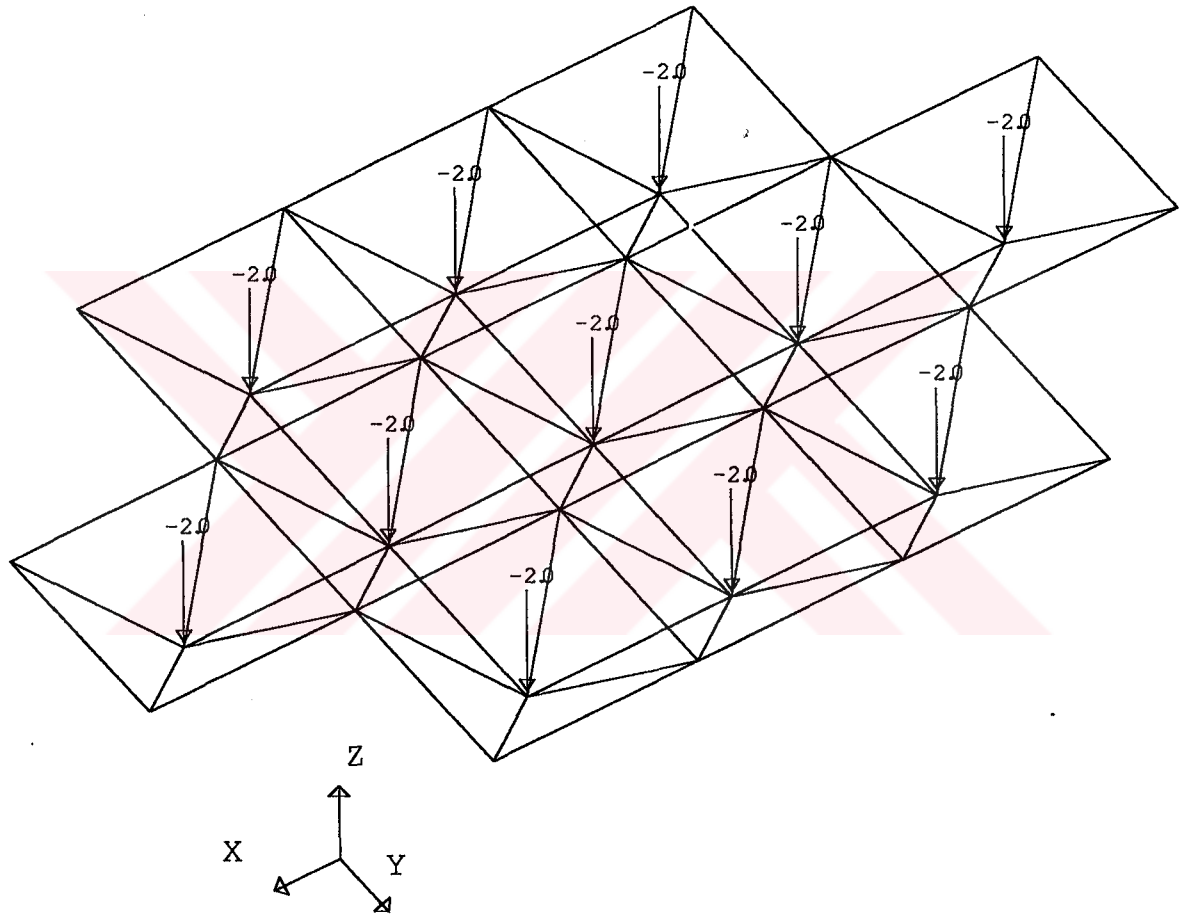
Şekil 6.2 Uzay kafes sisteminin düğüm numaraları ve boyutları



Şekil 6.3 Uzay kafes sisteminde elemanların numaralandırılması



Şekil 6.4 Uzay kafes sistemin yandan görünüşü



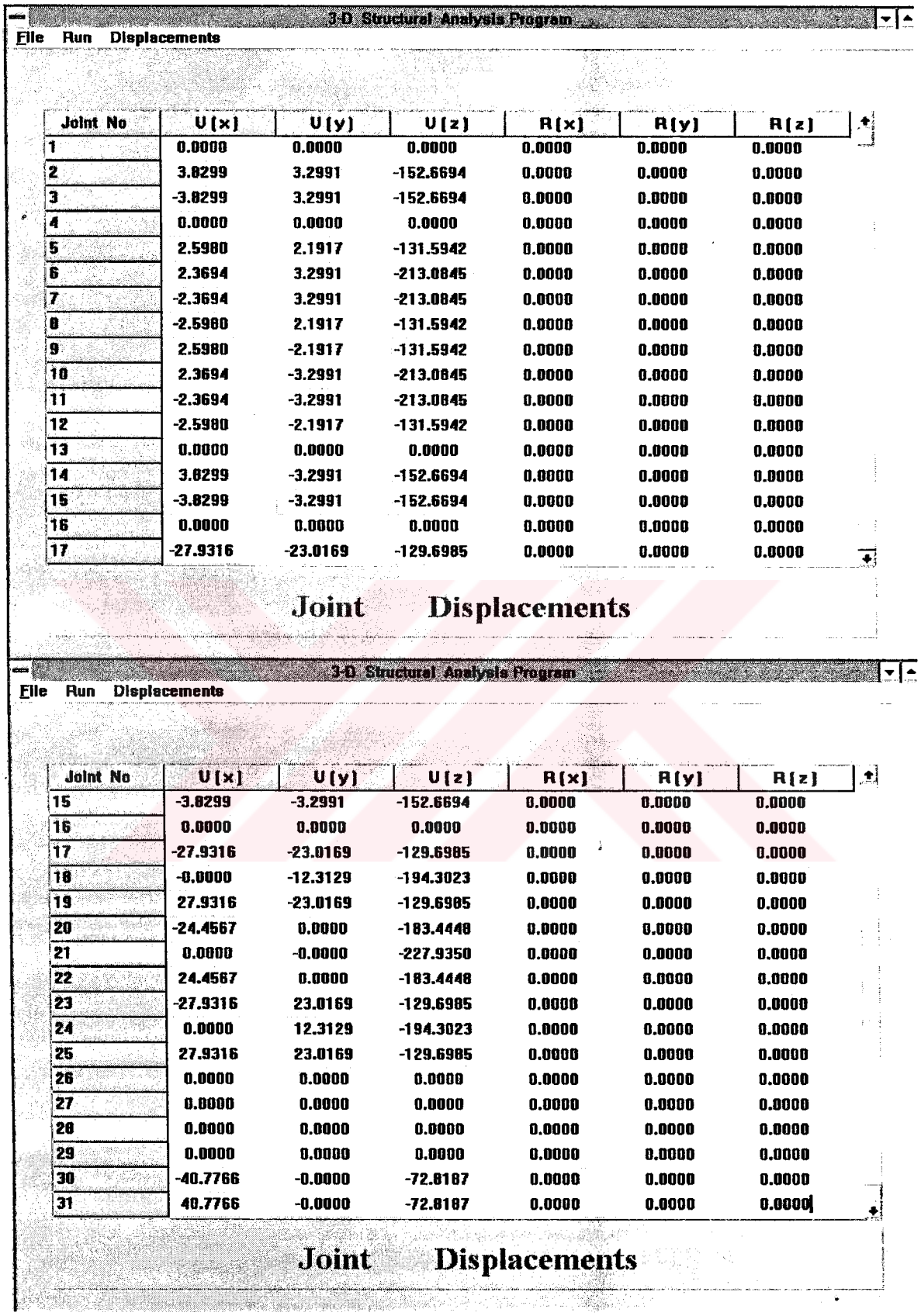
Şekil 6.5 Uzay kafes sistemine etki eden yüklemeler

Çözümleme işlemi sonrasında SAP90 tarafından elde edilen sonuçlar aşağıda gösterilmiştir. Hazırlanan yazılımın vermiş olduğu sonuçlar ise şekil 6.6 'da görülmektedir. Sonuç olarak SAP90 ile aynı deplasman değerleri elde edilmiştir.

J O I N T D I S P L A C E M E N T S

LOAD CONDITION 1 - DISPLACEMENTS "U" AND ROTATIONS "R"

JOINT	U(X)	U(Y)	U(Z)
1	.000000	.000000	.000000
2	3.829921	3.299133	-152.669411
3	-3.829921	3.299133	-152.669411
4	.000000	.000000	.000000
5	2.598022	2.191703	-131.594214
6	2.369419	3.299133	-213.084525
7	-2.369419	3.299133	-213.084525
8	-2.598022	2.191703	-131.594214
9	2.598022	-2.191703	-131.594214
10	2.369419	-3.299133	-213.084525
11	-2.369419	-3.299133	-213.084525
12	-2.598022	-2.191703	-131.594214
13	.000000	.000000	.000000
14	3.829921	-3.299133	-152.669411
15	-3.829921	-3.299133	-152.669411
16	.000000	.000000	.000000
17	-27.931600	-23.016945	-129.698477
18	.000000	-12.312861	-194.302309
19	27.931600	-23.016945	-129.698477
20	-24.456716	.000000	-183.444776
21	.000000	.000000	-227.935042
22	24.456716	.000000	-183.444776
23	-27.931600	23.016945	-129.698477
24	.000000	12.312861	-194.302309
25	27.931600	23.016945	-129.698477
26	.000000	.000000	.000000
27	.000000	.000000	.000000
28	.000000	.000000	.000000
29	.000000	.000000	.000000
30	-40.776631	.000000	-72.818675
31	40.776631	.000000	-72.818675



Şekil 6.6 Uzay kafes sistemi düğüm deplasmanları

6.2 Uzay Çerçeve Sistem Çözümü

Bu örnekte de üç boyutlu bir çerçeve sistem yazılım tarafından çözdürülmüştür ve sonuçlar SAP90 sonuçları ile karşılaştırılmıştır. Hazırlanan data dosyası şekil 6.7 'de görülmektedir. Aynı sistemin sap90 tarafından çözümü için hazırlanan data dosyası aşağıdaki gibidir.

```
SYSTEM
N=19 L=1
JOINTS
1   X=0   Y=0   Z=0
4   X=24  Y=0   Z=0   G=1,4,1
5   X=0   Y=6   Z=0
8   X=24  Y=6   Z=0   G=5,8,1
9   X=0   Y=0   Z=4
12  X=24  Y=0   Z=4   G=9,12,1
13  X=0   Y=6   Z=4
16  X=24  Y=6   Z=4   G=13,16,1
17  X=4   Y=3   Z=7
19  X=20  Y=3   Z=7   G=17,19,1
RESTRAINTS
1,8,1   R=1,1,1,1,1,1
9,19,1  R=0,0,0,0,0,0
FRAME
NM=1
1 A=1  I=1,1  E=1  G=1
1,1,9       M=1  G=3,1,1,1
5,5,13      M=1  G=3,1,1,1
9,9,10      M=1  G=2,1,1,1
12,13,14    M=1  G=2,1,1,1
15,9,13     M=1  G=3,1,1,1
19,9,17     M=1  G=1,1,1,0
```

21,13,17 M=1 G=1,1,1,0

23,10,18 M=1 G=1,1,1,0

25,14,18 M=1 G=1,1,1,0

27,11,19 M=1 G=1,1,1,0

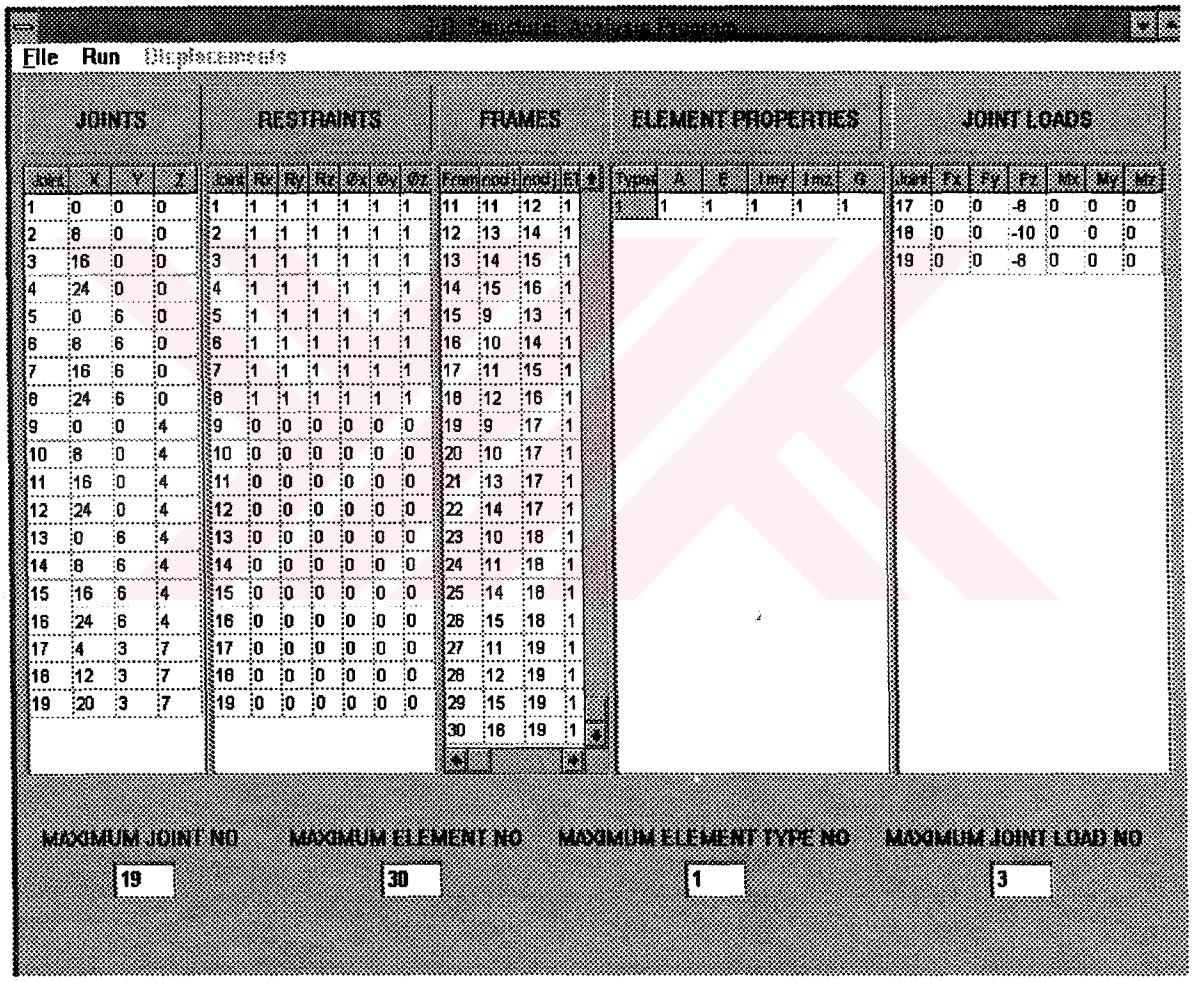
29,15,19 M=1 G=1,1,1,0

LOADS

16 F=0,0,0

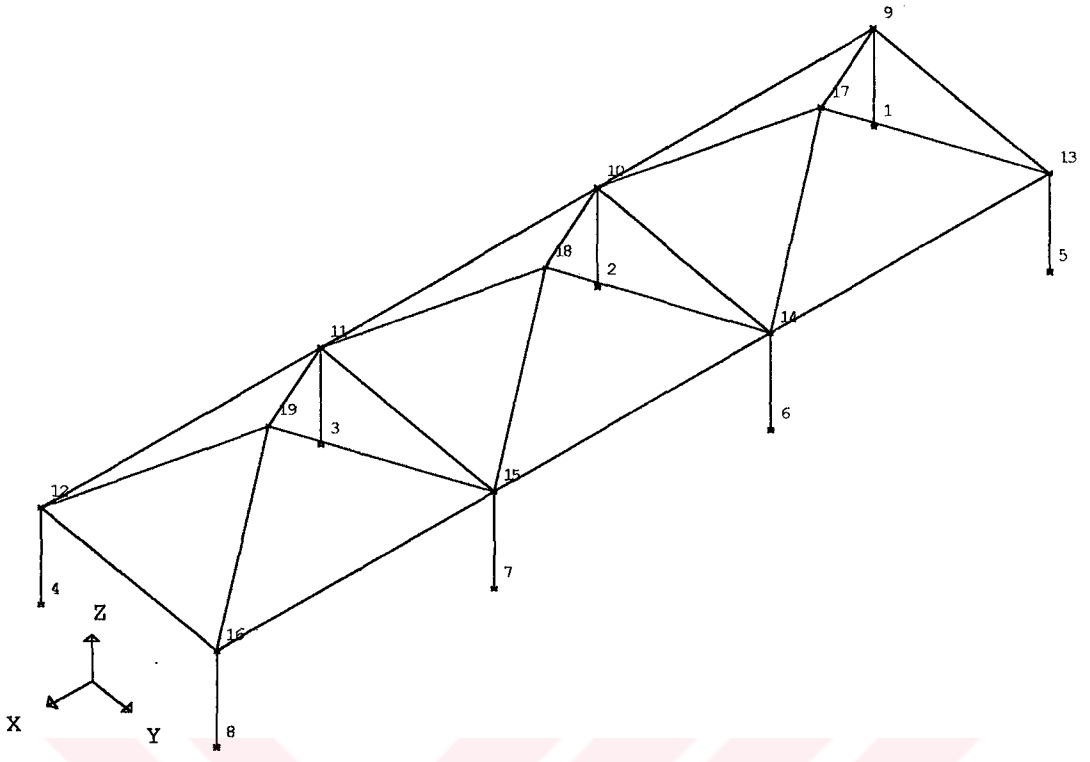
17,19,2 F=0,0,-8

18 F=0,0,-10

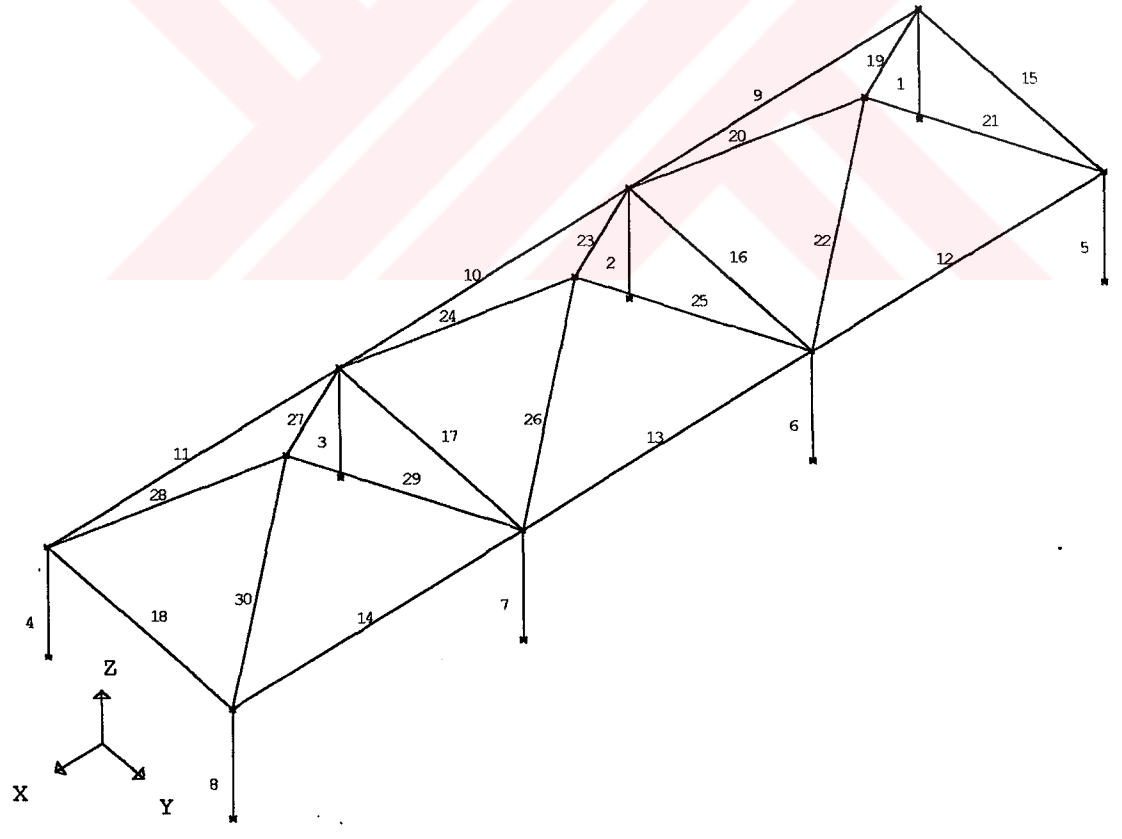


Şekil 6.7 Uzay çerçeve sistemi veri dosyası

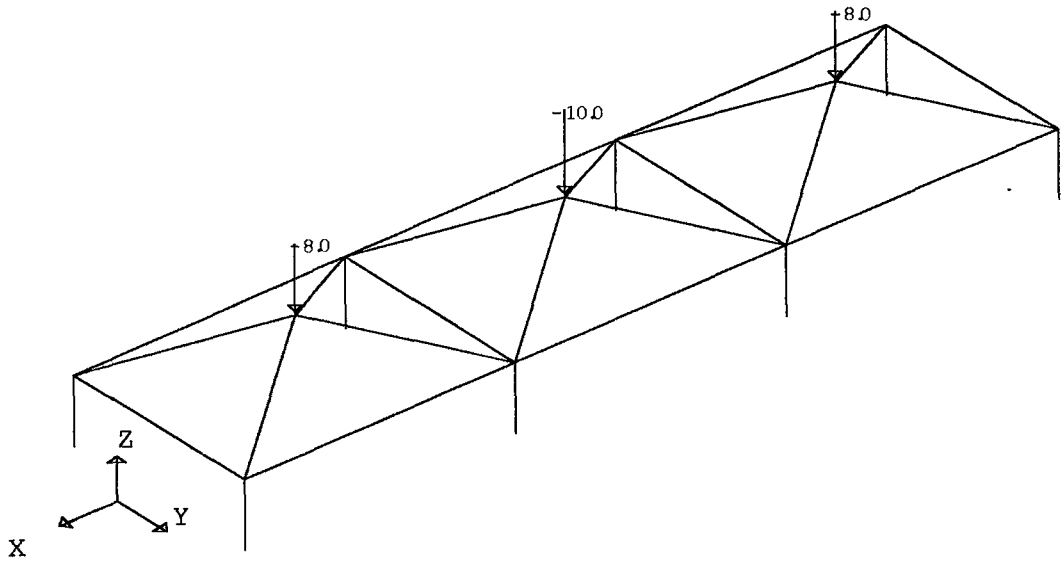
Uzay çerçeve sisteminin biçimi, sistemin boyutları, düğüm noktalarının ve elemanların yerleşimi, sisteme etki eden yüklemeler aşağıdaki şekillerde gösterilmiştir.



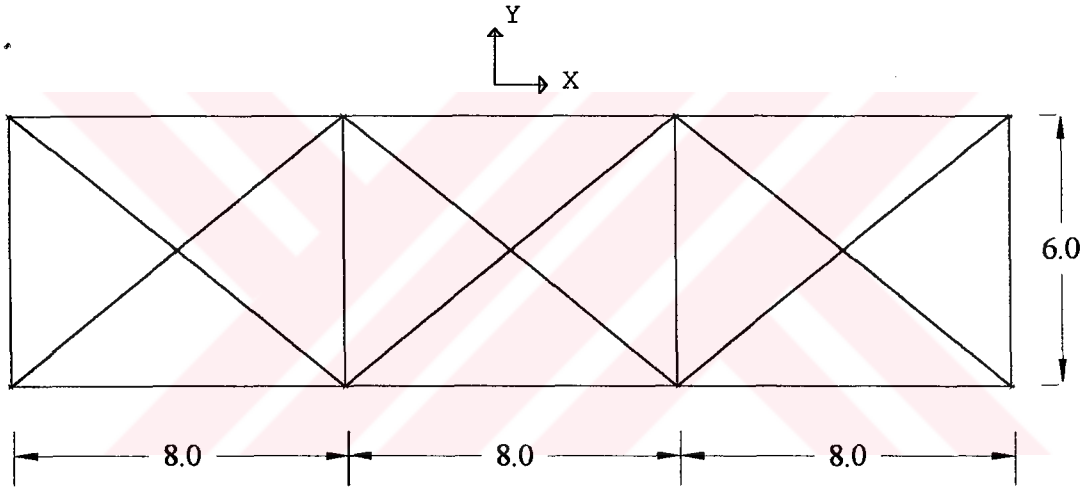
.Şekil 6.8 Uzay çerçeve sisteminin düğüm numaraları



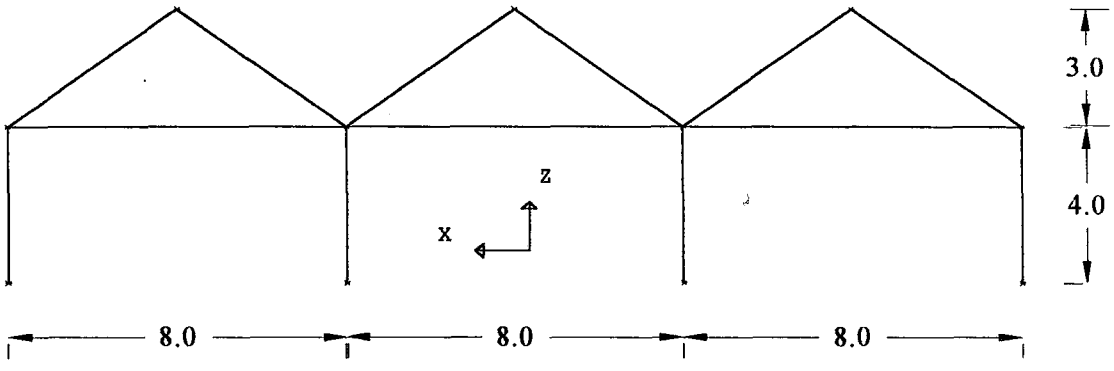
şekil 6.9 Uzay çerçeve sisteminde elemanların numaralandırılması



Şekil 6.10 Uzay çerçeve sistemine etki eden yüklemeler



Şekil 6.11 Uzay çerçeve sisteminin üstten görünüşü.



Şekil 6.12 Uzay çerçeve sisteminin yandan görünüşü

Çözümleme işlemi sonrasında SAP90 tarafından elde edilen sonuçlar aşağıdaki gibidir. Hazırlanan yazılımın vermiş olduğu sonuçlar da şekil 6.13 'de görülmektedir. Sonuç olarak SAP90 ile aynı deplasman değerleri elde edilmiştir.

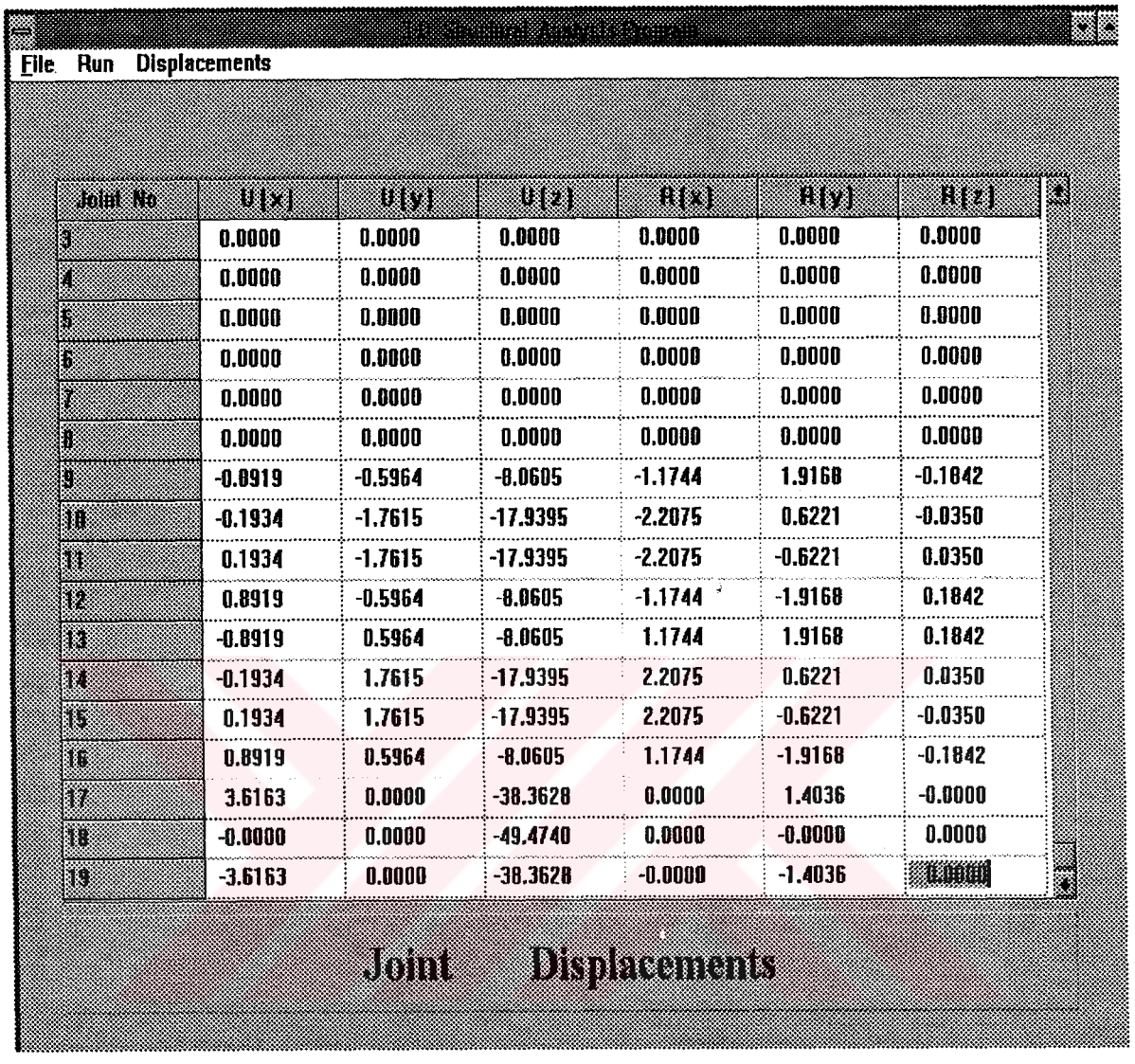
PAGE 1

PROGRAM:SAP90/FILE:spcframe.SOL

JOINT DISPLACEMENTS

LOAD CONDITION 1 - DISPLACEMENTS "U" AND ROTATIONS "R"

JOINT	U(X)	U(Y)	U(Z)	R(X)	R(Y)	R(Z)
1	.000000	.000000	.000000	.000000	.000000	.000000
2	.000000	.000000	.000000	.000000	.000000	.000000
3	.000000	.000000	.000000	.000000	.000000	.000000
4	.000000	.000000	.000000	.000000	.000000	.000000
5	.000000	.000000	.000000	.000000	.000000	.000000
6	.000000	.000000	.000000	.000000	.000000	.000000
7	.000000	.000000	.000000	.000000	.000000	.000000
8	.000000	.000000	.000000	.000000	.000000	.000000
9	-.891934	-.596373	-8.060489	-1.174387	1.916788	-.184215
10	-.193407	-1.761468	-17.939511	-2.207453	.622103	-.035017
11	.193407	-1.761468	-17.939511	-2.207453	-.622103	.035017
12	.891934	-.596373	-8.060489	-1.174387	-1.916788	.184215
13	-.891934	.596373	-8.060489	1.174387	1.916788	.184215
14	-.193407	1.761468	-17.939511	2.207453	0.622103	.035017
15	.193407	1.761468	-17.939511	2.207453	-.622103	-.035017
16	.891934	.596373	-8.060489	1.174387	-1.91678	-.184215
17	3.616303	.000000	-38.362817	.000000	1.403612	.000000
18	.000000	.000000	-49.473964	.000000	.000000	.000000
19	-3.616303	.000000	-38.362817	.000000	-1.403612	.000000



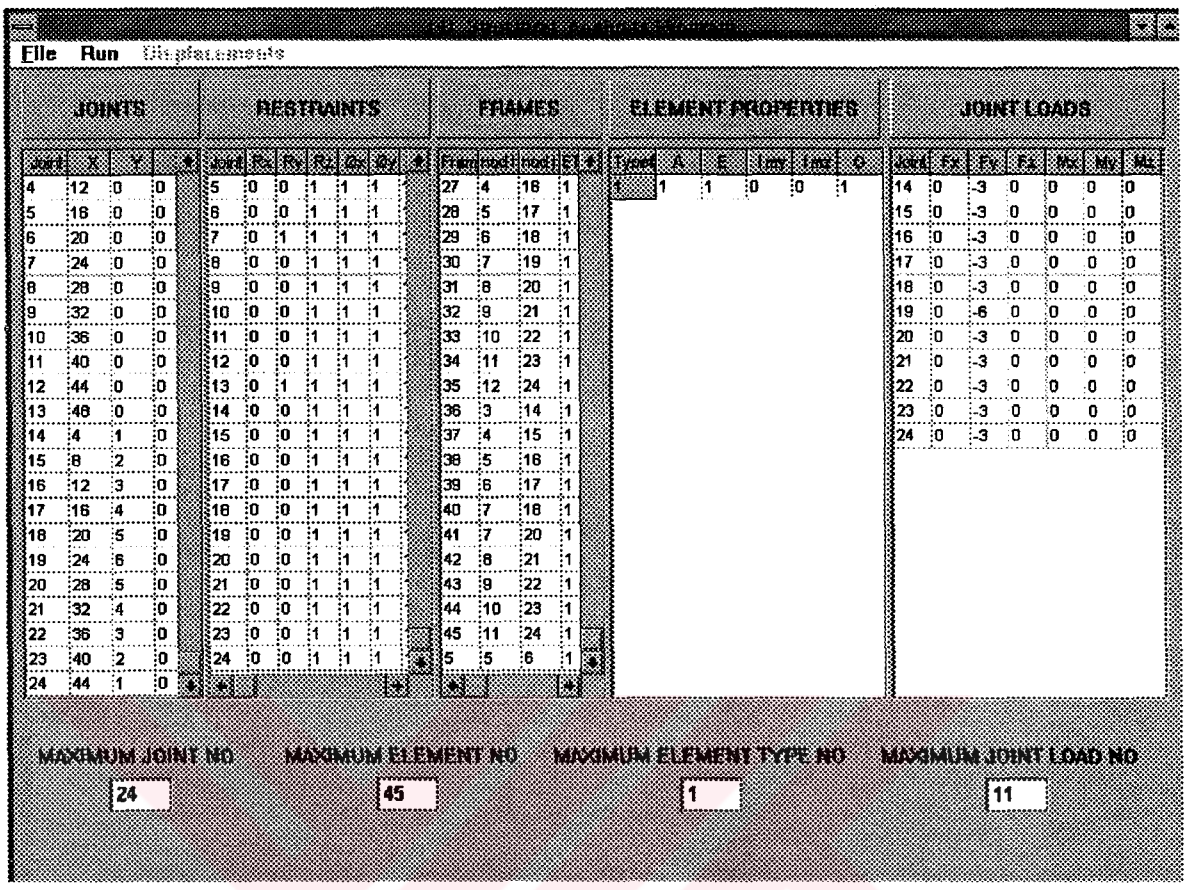
Joint No	U[x]	U[y]	U[z]	R[x]	R[y]	R[z]
3	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
4	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
5	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
6	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
7	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
8	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
9	-0.8919	-0.5964	-8.0605	-1.1744	1.9168	-0.1842
10	-0.1934	-1.7615	-17.9395	-2.2075	0.6221	-0.0350
11	0.1934	-1.7615	-17.9395	-2.2075	-0.6221	0.0350
12	0.8919	-0.5964	-8.0605	-1.1744	-1.9168	0.1842
13	-0.8919	0.5964	-8.0605	1.1744	1.9168	0.1842
14	-0.1934	1.7615	-17.9395	2.2075	0.6221	0.0350
15	0.1934	1.7615	-17.9395	2.2075	-0.6221	-0.0350
16	0.8919	0.5964	-8.0605	1.1744	-1.9168	-0.1842
17	3.6163	0.0000	-38.3628	0.0000	1.4036	-0.0000
18	-0.0000	0.0000	-49.4740	0.0000	-0.0000	0.0000
19	-3.6163	0.0000	-38.3628	-0.0000	-1.4036	0.0000

Şekil 6.13 Uzay çerçeve sistemi düğüm deplasmanları

6.3 Düzlem Kafes Sistem Çözümü

Hazırlanan bu örnekte iki boyutlu bir kafes sistem yazılım tarafından çözdürülmüştür. Hazırlanan veri dosyası şekil 6.14 'de gösterilmiştir. Aynı sistemin sap90 tarafından çözümü için hazırlanan data dosyası aşağıdaki gibidir.

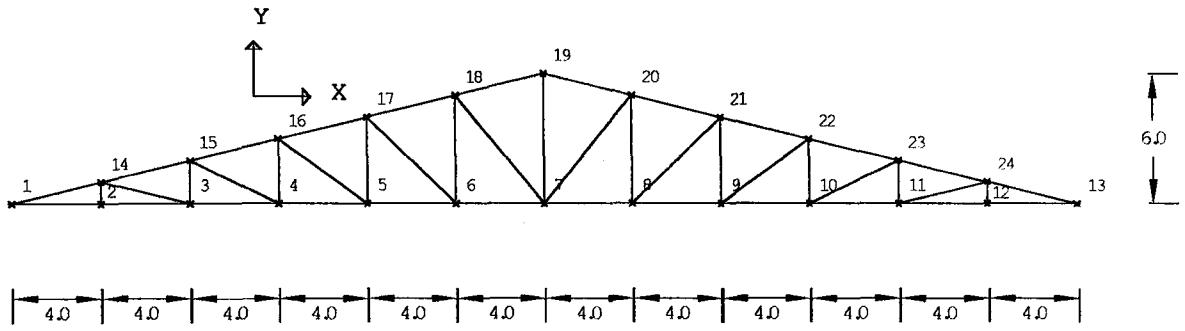
```
SYSTEM
N=24  L=1
JOINTS
1   X=0      Y=0     Z=0
13  X=48     Y=0     Z=0  G=1,13,1
14  X=4      Y=1     Z=0
19  X=24     Y=6     Z=0  G=14,19,1
24  X=44     Y=1     Z=0  G=19,24,1
RESTRAINTS
1,24,1  R=0,0,1,1,1,1
1       R=1,1,1,1,1,1
7,13,6  R=0,1,1,1,1,1
FRAME
NM=1
1  A=1  I=0,0 E=1  G=1
1,1,2       M=1   G=11,1,1,1
13,1,14     M=1
14,14,15    M=1   G=9,1,1,1
24,24,13    M=1
25,2,14     M=1   G=10,1,1,1
36,3,14     M=1   G=4,1,1,1
41,7,20     M=1   G=4,1,1,1
LOADS
14,18,1   F=0,-3,0
20,24,1   F=0,-3,0
19        F=0,-6,0
```



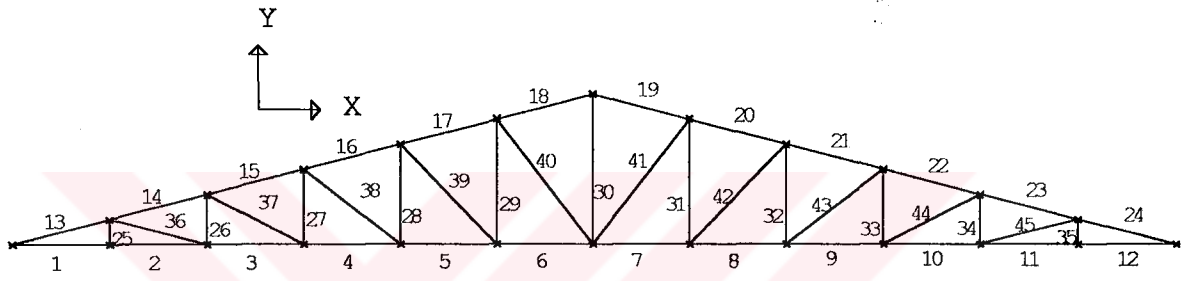
Şekil 6.14 Düzlem kafes sistemi veri dosyası

Düzlem kafes sisteminin biçimi, sistemin boyutları, düğüm noktalarının ve elemanların yerleşimi, sisteme etki eden yüklemeler şekil 6.15, 6.16 ve 6.17 'de görülmektedir.

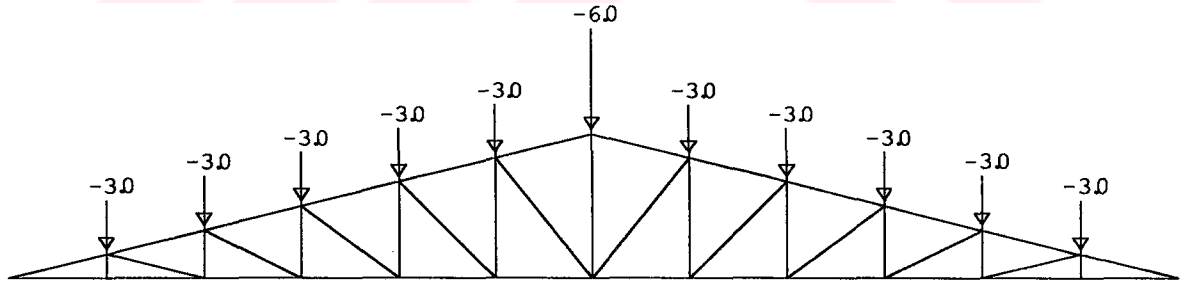
SAP90 programı kullanılarak yapılan çözümden elde edilen sonuçlar şekillerden sonra aşağıda verilmiştir. Yazılımın vermiş olduğu sonuçlar da şekil 6.18 'de görülmektedir. Görüldüğü gibi iki programın verdiği sonuçlar birbirinin aynıdır.



Şekil 6.15 Düzlem kafes sistemin düğüm numaraları ve ölçüler



Şekil 6.16 Düzlem kafes sistemin elemanlarının yerleşimi



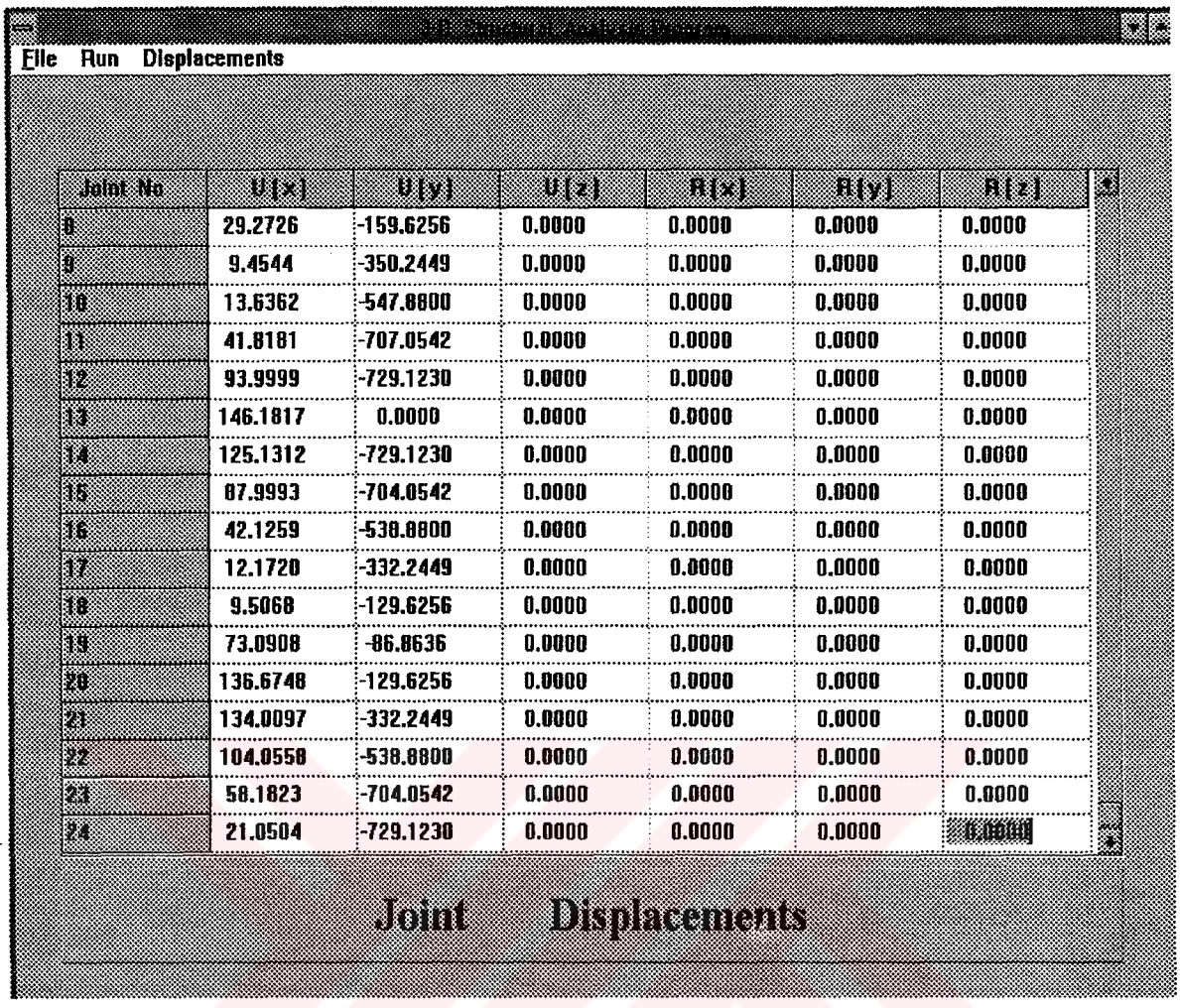
Şekil 6.17 Düzlem kafes sistemine etkiyen yüklemeler

PROGRAM:SAP90/FILE:PLNTRUSS.SOL

J O I N T D I S P L A C E M E N T S

LOAD CONDITION 1 - DISPLACEMENTS "U" AND ROTATIONS "R"

JOINT	U(X)	U(Y)
1	.000000	.000000
2	52.181805	-729.122953
3	104.363610	-707.054234
4	32.545415	-547.880040
5	136.727220	-350.244915
6	116.909026	-159.625619
7	73.090831	.000000
8	29.272636	-159.625619
9	9.454441	-350.244915
10	13.636246	-547.880040
11	41.818051	-707.054234
12	93.999856	-729.122953
13	146.181661	.000000
14	125.131229	-729.122953
15	87.999338	-704.054234
16	42.125877	-538.880040
17	12.171981	-332.244915
18	9.506842	-129.625619
19	73.090831	-86.863646
20	136.674820	-129.625619
21	134.009680	-332.244915
22	104.055784	-538.880040
23	58.182323	-704.054234
24	21.050432	-729.122953



Joint No	U[x]	U[y]	U[z]	R[x]	R[y]	R[z]
8	29.2726	-159.6256	0.0000	0.0000	0.0000	0.0000
9	9.4544	-350.2449	0.0000	0.0000	0.0000	0.0000
10	13.6362	-547.8800	0.0000	0.0000	0.0000	0.0000
11	41.8181	-707.0542	0.0000	0.0000	0.0000	0.0000
12	93.9999	-729.1230	0.0000	0.0000	0.0000	0.0000
13	146.1817	0.0000	0.0000	0.0000	0.0000	0.0000
14	125.1312	-729.1230	0.0000	0.0000	0.0000	0.0000
15	87.9993	-704.0542	0.0000	0.0000	0.0000	0.0000
16	42.1259	-538.8800	0.0000	0.0000	0.0000	0.0000
17	12.1720	-332.2449	0.0000	0.0000	0.0000	0.0000
18	9.5068	-129.6256	0.0000	0.0000	0.0000	0.0000
19	73.0908	-86.8636	0.0000	0.0000	0.0000	0.0000
20	136.6748	-129.6256	0.0000	0.0000	0.0000	0.0000
21	134.0097	-332.2449	0.0000	0.0000	0.0000	0.0000
22	104.0558	-538.8800	0.0000	0.0000	0.0000	0.0000
23	58.1823	-704.0542	0.0000	0.0000	0.0000	0.0000
24	21.0504	-729.1230	0.0000	0.0000	0.0000	0.0000

Şekil 6.18 Düzlem kafes sistemi düğüm deplasmanları

6.4 Düzlem Çerçeve Sistem Çözümü

Son olarak ele alınan bu örnekte iki boyutlu bir çerçeve sistemin statik analizi SAP90 ve hazırlanan yazılım kullanılarak. Data dosyası şekil 6.19 'da gösterilmiştir. Aynı sistemin sap90 tarafından çözümü için hazırlanan data dosyası aşağıdaki gibidir.

```
SYSTEM
N=32  L=1
JOINTS
1   X=0     Y=0   Z=0
8   X=42    Y=0   Z=0   G=1,8,1
9   X=0     Y=3   Z=0
16  X=42    Y=3   Z=0   G=9,16,1
17  X=0     Y=6   Z=0
23  X=36    Y=6   Z=0   G=17,23,1
24  X=6     Y=9   Z=0
27  X=24    Y=9   Z=0   G=24,27,1
28  X=12    Y=12  Z=0
30  X=24    Y=12  Z=0   G=28,30,1
31  X=18    Y=15  Z=0
32  X=24    Y=15  Z=0
RESTRAINTS
1,8,1   R=1,1,1,1,1,1
9,32,1  R=0,0,1,1,1,0
FRAME
NM=1
1  A=1  I=1,1  E=1  G=1
1,1,9       M=1  G=7,1,1,1
9,9,17      M=1  G=6,1,1,1
16,18,24    M=1  G=3,1,1,1
20,25,28    M=1  G=2,1,1,1
23,29,31    M=1  G=1,1,1,1
```

25,9,10 M=1 G=6,1,1,1

32,17,18 M=1 G=5,1,1,1

38,24,25 M=1 G=2,1,1,1

41,28,29 M=1 G=1,1,1,1

43,31,32 M=1

LOADS

32 F=0,-15,0

31 F=12,-15,0

28 F=10,-15,0

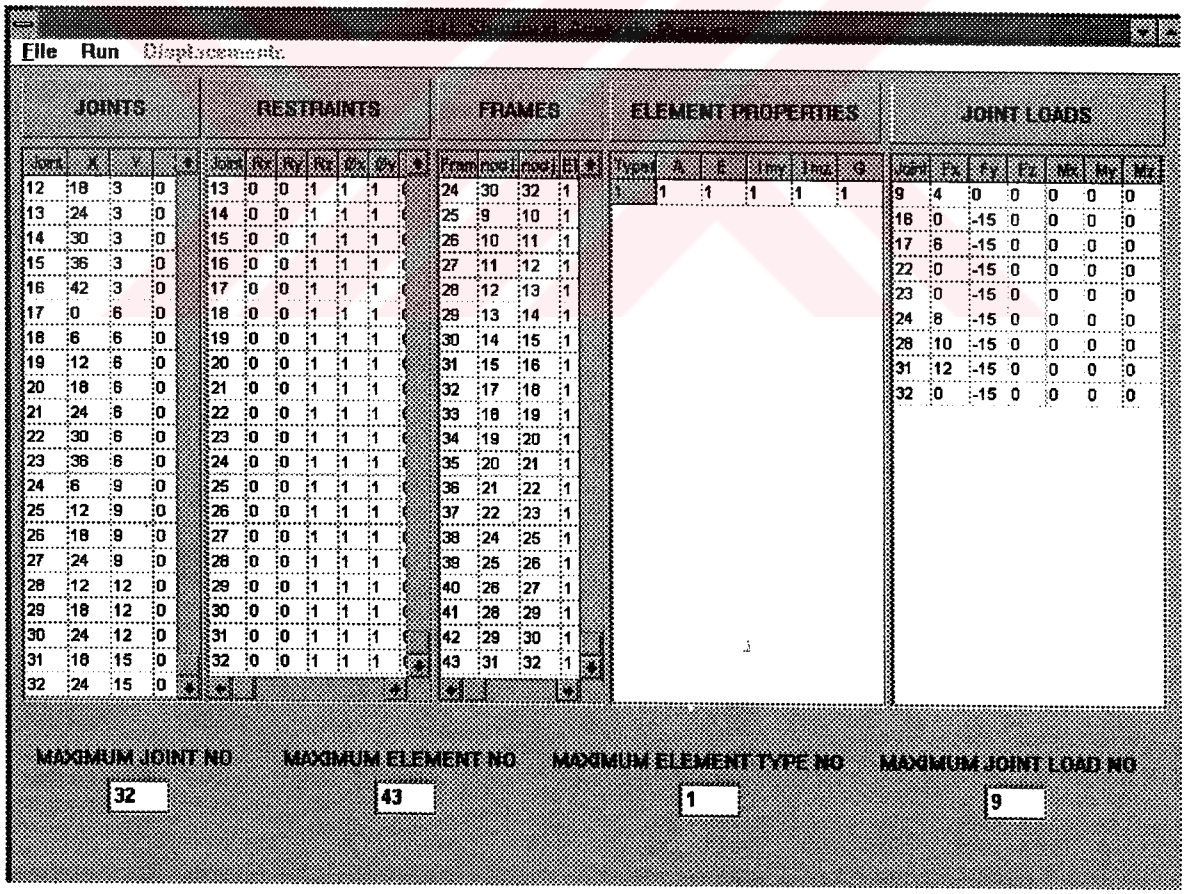
24 F=8,-15,0

17 F=6,-15,0

9 F=4,0,0

22,23,1 F=0,-15,0

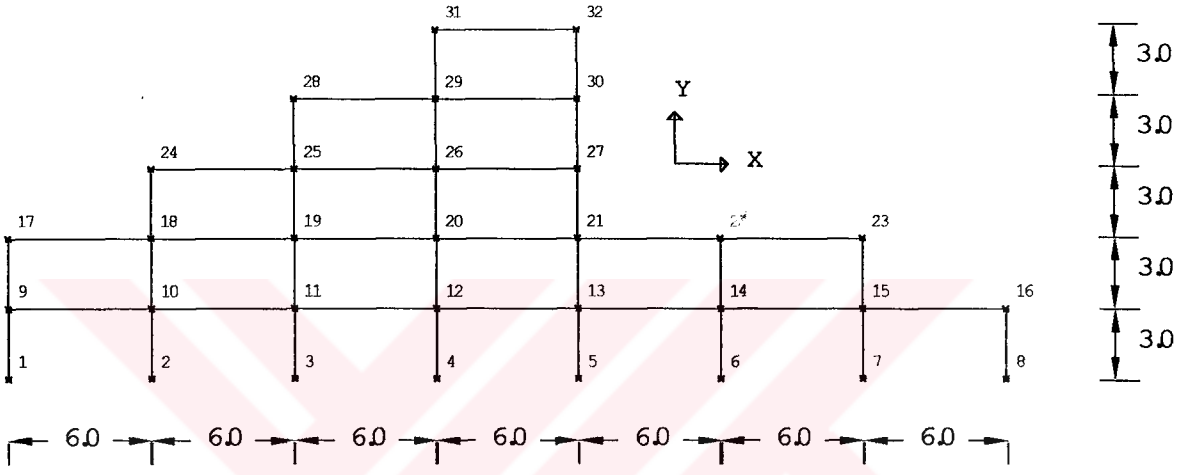
16 F=0,-15,0



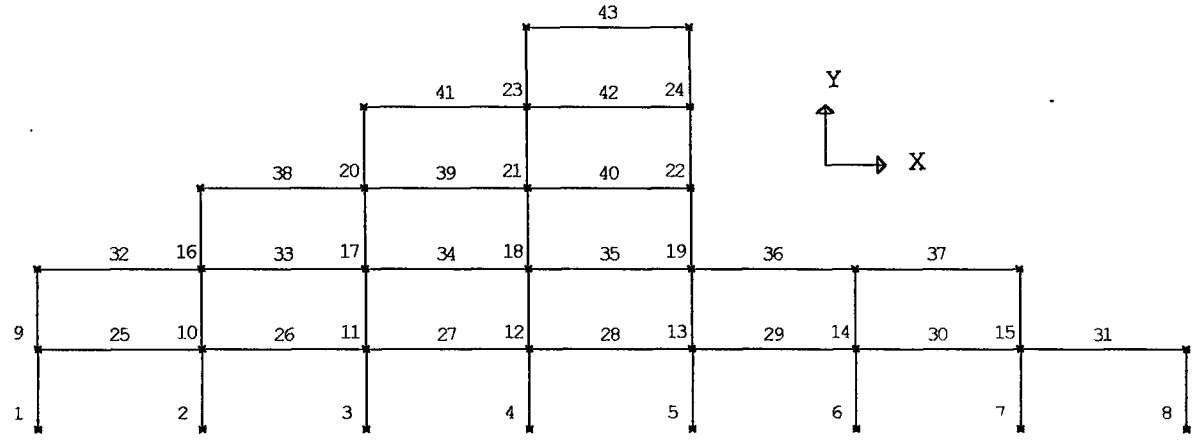
Şekil 6.19 Düzlem çerçeve sistemi veri dosyası

Düzlem çerçeve sisteminin biçimi, sistemin boyutları, düğüm noktalarının ve elemanların yerleşimi, sisteme etki eden yüklemeler şekil 6.20, 6.21 ve 6.22 'de gösterilmiştir.

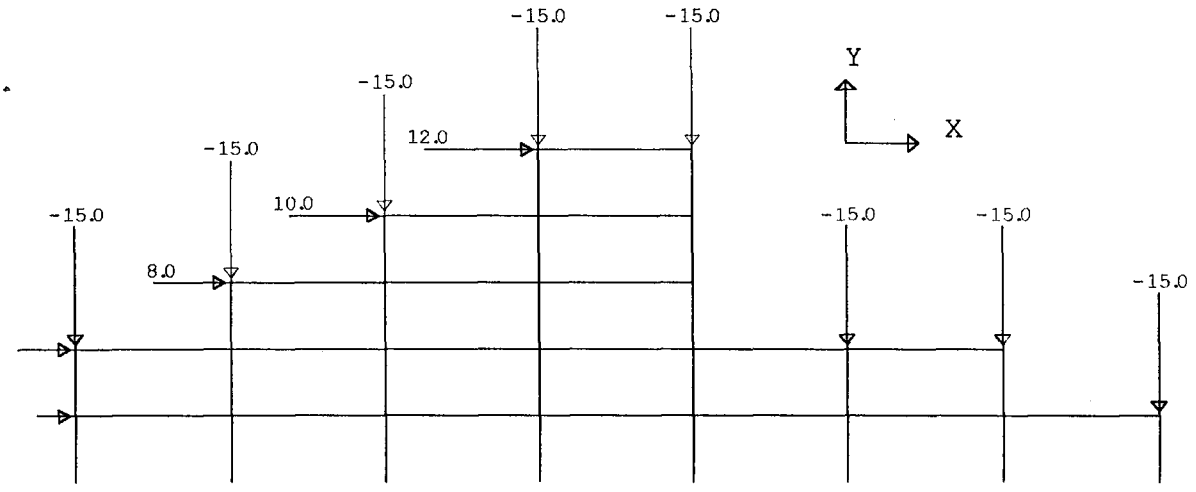
SAP90 çözümünden elde edilen sonuçlar aşağıda, şekillerden sonra verilmiştir. Geliştirilen yazılımın verdiği sonuçlar ise şekil 6.23 'de gösterilmiştir. Bu örnekte de iki programın sonuçlarının aynı olduğu görülmektedir.



Şekil 6.20 Düzlem çerçeve sisteminin düğüm numaraları ve ölçüler



Şekil 6.21 Düzlem çerçeve sistemin elemanlarının yerleşimi



Şekil 6.22 Düzlem çerçeve sistemine etkiyen yüklemeler

PAGE 1
PROGRAM:SAP90/FILE:plnframe.SOL

JOINT DISPLACEMENTS

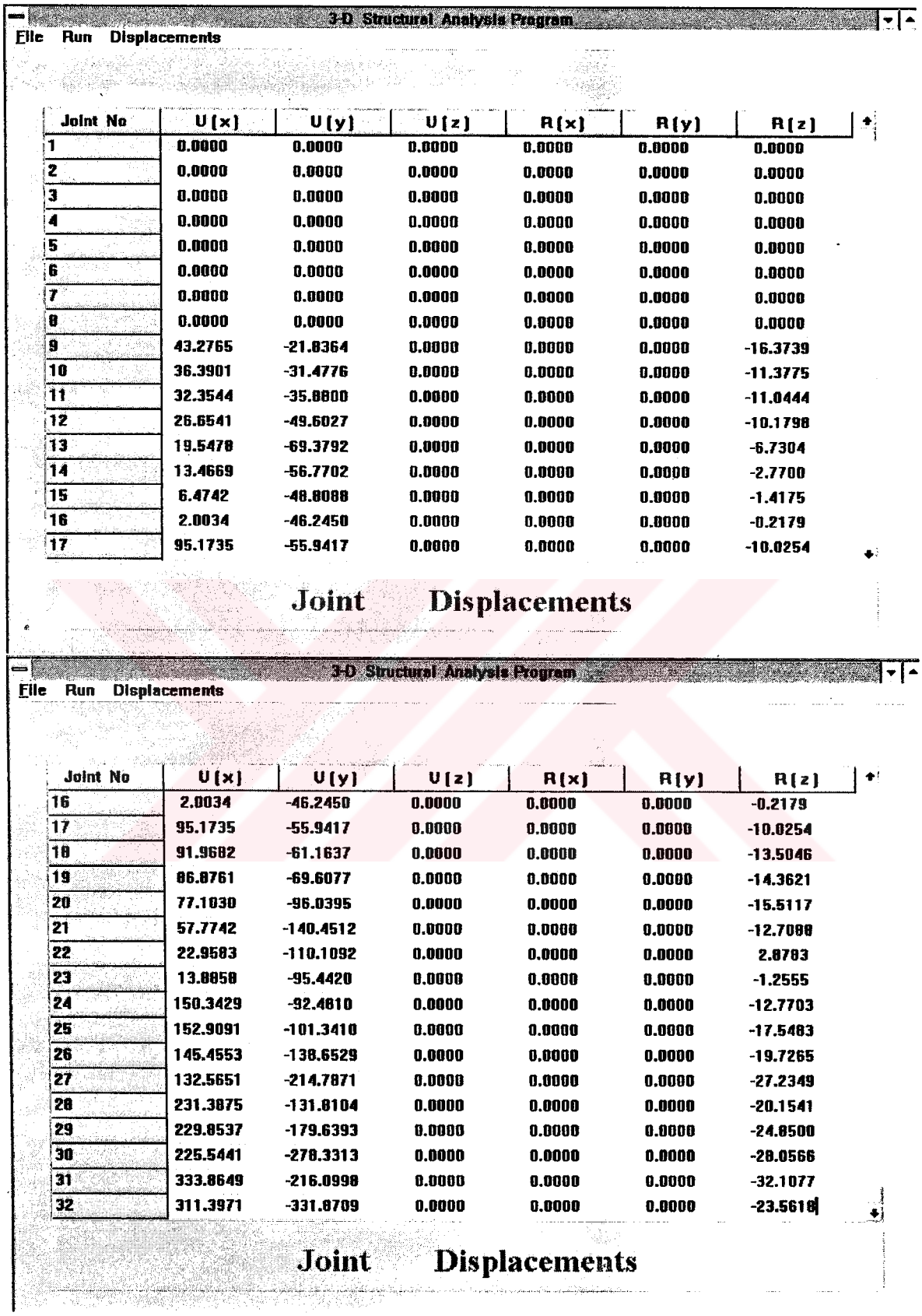
LOAD CONDITION 1 - DISPLACEMENTS "U" AND ROTATIONS "R"

JOINT	U(X)	U(Y)	R(Z)
1	.000000	.000000	.000000
2	.000000	.000000	.000000
3	.000000	.000000	.000000
4	.000000	.000000	.000000
5	.000000	.000000	.000000
6	.000000	.000000	.000000
7	.000000	.000000	.000000
8	.000000	.000000	.000000
9	43.276487	-21.836441	-16.373920
10	36.390117	-31.477631	-11.377544
11	32.354359	-35.879987	-11.044434
12	26.654061	-49.602732	-10.179772
13	19.547771	-69.379241	-6.730398

J O I N T D I S P L A C E M E N T S

LOAD CONDITION 1 - DISPLACEMENTS "U" AND ROTATIONS "R"

JOINT	U(X)	U(Y)	R(Z)
14	13.466871	-56.770155	-2.769995
15	6.474166	-48.808807	-1.417521
16	2.003379	-46.245005	-.217889
17	95.173496	-55.941749	-10.025423
18	91.968150	-61.163658	-13.504598
19	86.876117	-69.607691	-14.362131
20	77.102965	-96.039485	-15.511731
21	57.774203	-140.451193	-12.708788
22	22.958284	-110.109249	2.878257
23	13.885846	-95.441970	-1.255489
24	150.342858	-92.481018	-12.770321
25	152.909071	-101.341000	-17.548285
26	145.455285	-138.652914	-19.726480
27	132.565140	-214.787094	-27.234945
28	231.387501	-131.810401	-20.154115
29	229.853715	-179.639295	-24.850049
30	225.544057	-278.331312	-28.056644
31	333.864873	-216.099755	-32.107664
32	311.397110	-331.870853	-23.561784



Joint No	U[x]	U[y]	U[z]	R[x]	R[y]	R[z]
1	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
2	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
3	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
4	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
5	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
6	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
7	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
8	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
9	43.2765	-21.8364	0.0000	0.0000	0.0000	-16.3739
10	36.3901	-31.4776	0.0000	0.0000	0.0000	-11.3775
11	32.3544	-35.8800	0.0000	0.0000	0.0000	-11.0444
12	26.6541	-49.6027	0.0000	0.0000	0.0000	-10.1798
13	19.5478	-69.3792	0.0000	0.0000	0.0000	-6.7304
14	13.4669	-56.7702	0.0000	0.0000	0.0000	-2.7700
15	6.4742	-48.8088	0.0000	0.0000	0.0000	-1.4175
16	2.0034	-46.2450	0.0000	0.0000	0.0000	-0.2179
17	95.1735	-55.9417	0.0000	0.0000	0.0000	-10.0254

Joint No	U[x]	U[y]	U[z]	R[x]	R[y]	R[z]
16	2.0034	-46.2450	0.0000	0.0000	0.0000	-0.2179
17	95.1735	-55.9417	0.0000	0.0000	0.0000	-10.0254
18	91.9682	-61.1637	0.0000	0.0000	0.0000	-13.5046
19	86.8761	-69.6077	0.0000	0.0000	0.0000	-14.3621
20	77.1030	-96.0395	0.0000	0.0000	0.0000	-15.5117
21	57.7742	-140.4512	0.0000	0.0000	0.0000	-12.7088
22	22.9583	-110.1092	0.0000	0.0000	0.0000	2.8783
23	13.8858	-95.4420	0.0000	0.0000	0.0000	-1.2555
24	150.3429	-92.4810	0.0000	0.0000	0.0000	-12.7703
25	152.9091	-101.3410	0.0000	0.0000	0.0000	-17.5483
26	145.4553	-138.6529	0.0000	0.0000	0.0000	-19.7265
27	132.5651	-214.7871	0.0000	0.0000	0.0000	-27.2349
28	231.3875	-131.8104	0.0000	0.0000	0.0000	-20.1541
29	229.8537	-179.6393	0.0000	0.0000	0.0000	-24.8500
30	225.5441	-278.3313	0.0000	0.0000	0.0000	-28.0566
31	333.8649	-216.0998	0.0000	0.0000	0.0000	-32.1077
32	311.3971	-331.8709	0.0000	0.0000	0.0000	-23.5618

Şekil 6.23 Düzlem çerçeve sistemi düğüm deplasmanları

6.5 Sonuç

Yukarıda çözümü yapılan örneklerde de görüldüğü gibi, elde edilen çözümler SAP90 çözümleri ile tam bir uygunluk göstermektedir. Yapılan çalışmada esas amaç ticari uygulamalar için de kullanılabilecek bir yazılım geliştirmekten ziyade, lisans seviyesinde, özellikle yapı statiği derslerinde teorisi verilen rijitlik matrisi yönteminin anlaşılabilirliğine katkıda bulunmak, çözümleme işleminin adım adım kontrolünü sağlayabilmektir. Aynı zamanda yüksek lisans seviyesinde verilen ileri yapı analizi uygulamalarında da hazırlanan yazılım kolaylıkla kullanılabilecektir.

Öncelikle bu çalışma bir başlangıç niteliğini taşımaktadır. Plak elemanların çözümü, dinamik analiz ve elastik zemin modellenmesi gibi ilave çalışmalar ile yazılımın geliştirilmesine devam edilecektir.

KAYNAKLAR

1. AKTAŞ, Z., ÖNCÜL, H., URAL,S., "Sayısal Çözümleme", ODTÜ,Ankara-1981

2. BANGER, G., "Turbo Pascal 7", Bilim-Teknik Yayınevi, İstanbul, 1993

3. BATHE, K.J., "Finite Element Procedures in Engineering Analysis", Prentice Hall, New Jersey, 1982

4. BREBBIA, C.A., CONNOR, J.J., "Fundamentals of Finite Element Techniques", First Edition, Butterworth Group, London, 1973

5. ÇAKIROĞLU, A.,ÖZDEN, E., ÖZMEN, G.,"Yapı Sistemlerinin Hesabı İçin Matris Metotları ve Elektronik Hesap Makinası Programları", İkinci Baskı, İstanbul Teknik Ünv. İnşaat Fakultesi Matbaası, İstanbul-1992

6. GHALI, A., NEVILLE, A.M., "Structural Analysis", Second Edition, Chapman and Hall, Newyork-1978

7. HAMMINNG, R.W., "Numerical Methods For Scientists and Engineers" McGraw-Hill Book Company, New York- 1962.

8. KRISHNAMOORTY, C.S., "Finite Element Analysis", Second Edition, McGraw-Hill, 1990

9. MARTIN, H.C., CAREY, G.F., "Introduction to Finite Element Analysis", McGraw-Hill, 1973

10. NATH, B., "Fundamental of Finite Elements for Engineers." The Atlone Press of The University of London, London-1974

11. NORRIE, D.H., De VIRIES, G, "An Introduction to Finite Element Analysis." Academic Press Inc. NewYork-1974

12. PACHEO, X., TEIXEIRA, S., "Delphi Developers Guide", First Edition, Sams Publishing, 1995

13. REDDY, C.S., "Basic Structural Analysis", Sixth Reprint, McGraw-Hill, 1989

14. TEZCAN, S., "Çubuk Sistemlerin Elektronik Hesap Makinaları ile Çözümü", Arı Kitabevi Matbaası, İstanbul-1970

15. WANG, C.K., "İntermediate Structural Analysis", McGraw-Hill, 1983

16. WILLIAM, H., "Numericall Recipes", University of Cambiridge, 1989

17. WILSON, E.L., HABİBULLAH, A., "SAP80-90 Structural Analysis programs." Computer & Structures Inc., Berkeley, California-1984

18. ZEINKEWICZ, O.C., "The Finite Element Method", McGraw-Hill Book Company Limited. London- 1979

EK 1

(PROGRAM LİSTESİ)

```
unit Tkong;

interface

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, Grids, StdCtrls, ExtCtrls,
Menus,SysTypes,GrdAp3da,MatAp3da,OpenSave;

type

  Tform1 = class(TForm)
    MainMenu1: TMainMenu;
    OpenDialog1: TOpenDialog;
    SaveDialog1: TSaveDialog;
    MaxElNoBox: TEdit;
    MaxJointNoBox: TEdit;
    MaxElTypeNoBox: TEdit;
    MaxJointLoadNoBox: TEdit;
    MaxSpanLoadNoBox: TEdit;
    File1: TMenuItem;
    Open1: TMenuItem;
    Save1: TMenuItem;
    N1: TMenuItem;
    Exit1: TMenuItem;
    Run1: TMenuItem;
    Run2: TMenuItem;
    SpanLoadsGrid: TStringGrid;
    JointsGrid: TStringGrid;
    restraintsGrid: TStringGrid;
    FrameGrid: TStringGrid;
    DofGrid: TStringGrid;
    GSysFERGrid: TStringGrid;
    GSysDisplacementGrid: TStringGrid;
    ElementListGrid: TStringGrid;
    GSysJntLoadGrid: TStringGrid;
    JointLoadsGrid: TStringGrid;
    GSysStifnesGrid: TStringGrid;
    NcodeGrid: TStringGrid;
    ElementPropertiesGrid: TStringGrid;
    GelStifnesGrid: TStringGrid;
    GelDisplacementGrid: TStringGrid;
    GelFERGrid: TStringGrid;
    GElEndForcesGrid: TStringGrid;
    JOINTSPANEL: TPanel;
    RESTRAINTSPANEL: TPanel;
    FRAMESPANEL: TPanel;
    ElPropPanel: TPanel;
    JntLoadPanel: TPanel;
```

```
  Label1: TLabel;
  Label2: TLabel;
  Label3: TLabel;
  Label4: TLabel;
  SysMatrices1: TMenuItem;
  ElMatrices1: TMenuItem;
  ShowCodeTable1: TMenuItem;
  Panelksys: TPanel;
  Paneldsys: TPanel;
  PanelQsys: TPanel;
  PanelNcode: TPanel;
  PanelElNo: TPanel;
  elnobox: TEdit;
  PanelKel: TPanel;
  PanelDel: TPanel;
  Panel1: TPanel;
  PanelPel: TPanel;
  go: TButton;
  N2: TMenuItem;
  DataPanel: TMenuItem;
  JointResultsGrid: TStringGrid;
  Displacements1: TMenuItem;
  PanelJntDisplc: TPanel;
  procedure FormCreate(Sender: TObject);
  procedure MaxJointNoBoxChange(Sender: TObject);
  procedure MaxElNoBoxChange(Sender: TObject);
  procedure MaxElTypeNoBoxChange(Sender: TObject);
  procedure Exit1Click(Sender: TObject);
  procedure MaxJointLoadNoBoxChange(Sender: TObject);
  procedure MaxSpanLoadNoBoxChange(Sender: TObject);
  procedure Run2Click(Sender: TObject);
  procedure Open1Click(Sender: TObject);
  procedure Save1Click(Sender: TObject);
  procedure SysMatrices1Click(Sender: TObject);
  procedure ShowCodeTable1Click(Sender: TObject);
  procedure ElMatrices1Click(Sender: TObject);
  procedure goClick(Sender: TObject);
  procedure AllVisibleFalse;
  procedure DataPanelClick(Sender: TObject);
  procedure Displacements1Click(Sender: TObject);

 private
   { Private declarations }
 public

   { Public declarations }
 end;
var
 form1: Tform1;
```

```
implementation
Type TmatFile=String[30];
{$R *.DFM}
{*************************************************************************}
procedure Tform1.FormCreate(Sender: TObject);
begin
jointsGrid.cells[0,0]:=' Joint ';
jointsGrid.cells[1,0]:='   X  ';
jointsGrid.cells[2,0]:='   Y  ';
jointsGrid.cells[3,0]:='   Z  ';
RestraintsGrid.cells[0,0]:='Joint';
RestraintsGrid.cells[1,0]:=' Rx   ';
RestraintsGrid.cells[2,0]:=' Ry   ';
RestraintsGrid.cells[3,0]:=' Rz  ';
RestraintsGrid.cells[4,0]:=' Øx  ';
RestraintsGrid.cells[5,0]:=' Øy  ';
RestraintsGrid.cells[6,0]:=' Øz  ';
FrameGrid.Cells[0,0]:='Frame';
FrameGrid.Cells[1,0]:='nod i';
FrameGrid.Cells[2,0]:='nod j';
FrameGrid.Cells[3,0]:='ETyp';
ElementPropertiesGrid.cells[0,0]:='Type#';
ElementPropertiesGrid.cells[1,0]:='   A ';
ElementPropertiesGrid.cells[2,0]:='   E ';
ElementPropertiesGrid.cells[3,0]:='  I my ';
ElementPropertiesGrid.cells[4,0]:='  I mz ';
ElementPropertiesGrid.cells[5,0]:='   G ';
JointLoadsGrid.cells[0,0]:='Joint';
JointLoadsGrid.cells[1,0]:='  Fx ';
JointLoadsGrid.cells[2,0]:='  Fy ';
JointLoadsGrid.cells[3,0]:='  Fz ';
JointLoadsGrid.cells[4,0]:='  Mx ';
JointLoadsGrid.cells[5,0]:='  My ';
JointLoadsGrid.cells[6,0]:='  Mz ';
SpanLoadsGrid.cells[0,0]:='Frame';
SpanLoadsGrid.cells[1,0]:=' Wg-x';
SpanLoadsGrid.cells[2,0]:=' Wg-y';
SpanLoadsGrid.cells[3,0]:=' Wg-z';
ElementListGrid.cells[0,0]:='  El # ';
ElementListGrid.cells[1,0]:='    A   ';
ElementListGrid.cells[2,0]:='    E   ';
ElementListGrid.cells[3,0]:='  I my   ';
ElementListGrid.cells[4,0]:='  I mz   ';
ElementListGrid.cells[5,0]:='   G  ';
ElementListGrid.cells[6,0]:='   L  ';
ElementListGrid.cells[7,0]:='  Cx  ';
ElementListGrid.cells[8,0]:='  Cy  ';
ElementListGrid.cells[9,0]:='  Cz  ';
end;
```

```
{*************************************************************************}
procedure Tform1.MaxJointNoBoxChange(Sender: TObject);
begin
MaxJointNo:=StrToInt(MaxJointNoBox.text);
if maxjointno < 0 then MaxJointNo:=0;
JointsGrid.rowcount:=MaxJointNo+1;
RestraintsGrid.rowcount:=MaxJointNo+1;
end;
{*************************************************************************}
procedure Tform1.MaxElNoBoxChange(Sender: TObject);
begin
MaxElNo:=StrToInt(MaxElNoBox.text);
if MaxElNo < 0 then MaxElNo:=0;
FrameGrid.rowcount:=MaxElNo+1;
NcodeGrid.colcount:=MaxElNo+1;
end;
{*************************************************************************}
procedure Tform1.MaxElTypeNoBoxChange(Sender: TObject);
var i:integer;
begin
MaxElTypeNo:=StrToInt(MaxElTypeNoBox.text);
if MaxElTypeNo < 0 then MaxElTypeNo:=0;
ElementPropertiesGrid.rowcount:=MaxElTypeNo+1;
for i:=1 to MaxElTypeNo do
 begin
 ElementPropertiesGrid.Cells[0,i]:=inttostr(i);
 end;
end;
{*************************************************************************}
procedure Tform1.Exit1Click(Sender: TObject);
begin
close;
end;
{*************************************************************************}
procedure Tform1.MaxJointLoadNoBoxChange(Sender: TObject);
begin
MaxJointLoadNo:=StrToInt(MaxJointLoadNoBox.text);
if MaxJointLoadNo < 0 then MaxJointLoadNo:=0;
JointLoadsGrid.rowcount:=MaxJointLoadNo+1;
end;
{*************************************************************************}
procedure Tform1.MaxSpanLoadNoBoxChange(Sender: TObject);
begin
MaxSpanLoadNo:=StrToInt(MaxSpanLoadNoBox.text);
if MaxSpanLoadNo < 0 then MaxSpanLoadNo:=0;
SpanLoadsGrid.rowcount:=MaxSpanLoadNo+1;
end;
{*************************************************************************}
```

```
procedure Tform1.Run2Click(Sender: TObject);
label continue,continue2;
var i,j,m,k,n,z,t:integer;
   r,sum:real;
   NonZeroElement:integer;
begin
CreateNcodeGrid(RestraintsGrid,FrameGrid,DofGrid,NcodeGrid,nmax);
CreateElementListGrid(FrameGrid,JointsGrid,ElementPropertiesGrid,ElementListGrid);
{----------------------System Stifness Matrix-----------------------------}
   begin
    for i:=1 to nmax do
    for j:=1 to nmax do
    begin
     SysStifnesMatrix[i,j]:=0;
    end;
   for m:=1 to MaxElNo do
    for i:=1 to 12 do
    begin
     k:=StrToInt(NcodeGrid.cells[m-1,i]);
     if k > 0 then
     begin
      for j:=1 to 12 do
      begin
       n:=StrToInt(Ncodegrid.cells[m-1,j]);
       if n > 0 then
       begin
        GElStifnesMatrix[i,j]:=0;
        begin
         Area:=StrToFloat(ElementListGrid.cells[1,m]);
         MofE:=StrToFloat(ElementListGrid.cells[2,m]);
         Imy:=StrToFloat(ElementListGrid.cells[3,m]);
         Imz:=StrToFloat(ElementListGrid.cells[4,m]);
         G:=StrToFloat(ElementListGrid.cells[5,m]);
         L:=StrToFloat(ElementListGrid.cells[6,m]);
         Cx:=StrToFloat(ElementListGrid.cells[7,m]);
         Cy:=StrToFloat(ElementListGrid.cells[8,m]);
         Cz:=StrToFloat(ElementListGrid.cells[9,m]);
         Alfa:=0;
         GetTransformMatrix(Cx,Cy,Cz,Alfa,TransformMatrix);
         GetGElStifnesMatrix(LElStifnesMatrix,GELStifnesMatrix);
         begin
         SysStifnesMatrix[k,n]:=SysStifnesMatrix[k,n]+GElStifnesMatrix[i,j];
         end;
        end;
       end;
      end;
     end;
    end;
   end;
```

```
  begin
  for i:= 1 to nmax do
   begin
    if sysstifnesmatrix[i,i]=0 then sysstifnesmatrix[i,i]:=1e-15;
   end;
  end;
for i:=1 to nmax do for j:=1 to nmax do
  begin
   GSysStifnesGrid.Cells[i,j]:=FloattoStr(SysStifnesMatrix[j,i]);
   GSysStifnesGrid.Cells[0,i]:=intToStr(i);
   GSysStifnesGrid.Cells[i,0]:=intToStr(i);
  end;
GetSysJntLoadGrid(JointLoadsGrid,DofGrid,SysJointLoadVector);
for i:=1 to nmax do
begin
GSysJntLoadGrid.cells[0,i]:=inttoStr(i);
GSysJntLoadGrid.cells[1,i]:=FloatToStr(SysJointLoadVector[i]);
end;
{---------- Solution of Linear Algebraic Equations ----------}
 Begin
  Begin
  for i:=1 to (nmax-1) do
  continue2:
   begin
   if SysStifnesMatrix[i,i]=0 then
    begin
    NonZeroElement:=0;
    for t:=i+1 to nmax do
     begin
     if SysStifnesMatrix[t,i] <> 0 then
      begin
      NonZeroElement:=NonZeroElement+1;
      SysJointLoadVector[0]:=SysJointLoadVector[i];
      SysJointLoadVector[i]:=SysJointLoadVector[t];
      SysJointLoadVector[t]:=SysJointLoadVector[0];
       for z:=1 to n do
       begin
        SysStifnesMatrix[0,z]:=SysStifnesMatrix[i,z];
        SysStifnesMatrix[i,z]:=SysStifnesMatrix[t,z];
        SysStifnesMatrix[t,z]:=SysStifnesMatrix[0,z];
       end;
       goto continue;
      end;
       if NonZeroElement=0 then
        begin
        i:=i+1;
        goto continue2;
        end;
      end;
```

```
      end;
      continue:
     for m:=i+1 to nmax do
      begin
      r:=SysStifnesMatrix[m,i]/SysStifnesMatrix[i,i];
       begin
       for z:=1 to nmax do
         begin
         SysStifnesMatrix[m,z]:=SysStifnesMatrix[m,z]-(r*SysStifnesMatrix[i,z]);
         end;
         SysJointLoadVector[m]:=SysJointLoadVector[m]-r*SysJointLoadVector[i];
         if (SysJointLoadVector[m]>-1e-6) and (SysJointLoadVector[m]<1e-6) then
SysJointLoadVector[m]:=0;
       end;
     end;
   end;
  end;
  Begin
SysDisplacementvector[nmax]:=SysJointLoadVector[nmax]/SysStifnesMatrix[nmax,nmax];
  for j:=(nmax-1) downto 1 do
   begin
   Sum:=0;
   for t:=nmax downto (j+1) do
    begin
    Sum:=Sum+SysStifnesMatrix[j,t]*SysDisplacementvector[t];
    end;
    if SysStifnesMatrix[j,j]=0 then SysStifnesMatrix[j,j]:=1e-15;
    SysDisplacementvector[j]:=(SysJointLoadVector[j]-Sum)/SysStifnesMatrix[j,j];
    if (SysJointLoadVector[j]>-1e-6) and (SysJointLoadVector[j]<1e-6) then
SysJointLoadVector[j]:=0;
   end;
  end;
 end; {GetGSysDisplacements}
SysMatrices1.Enabled:=true;
ElMatrices1.Enabled:=true;
ShowCodeTable1.Enabled:=true;
DataPanel.enabled:=true;
displacements1.enabled:=true;
end;{End of procedure run2}
{*****************************************************************}
procedure Tform1.Open1Click(Sender: TObject);
var
Matfile:File of TMatfile;
i,j,n:integer;
wrtstr:TMatfile;
begin
  if openDialog1.execute then
  begin
  AssignFile(Matfile,OpenDialog1.FileName);
```

```
Reset(MatFile);
Read(MatFile,WrtStr);
MaxJointNo:=StrToInt(WrtStr);
MaxJointNoBox.text:=wrtstr;
Read(MatFile,WrtStr);
MaxElNo:=StrToInt(WrtStr);
MaxElNoBox.text:=wrtstr;
Read(MatFile,WrtStr);
MaxElTypeNo:=StrToInt(WrtStr);
MaxElTypeNoBox.text:=wrtstr;
Read(MatFile,WrtStr);
MaxJointLoadNo:=StrToInt(WrtStr);
MaxJointLoadNoBox.text:=wrtstr;
Read(MatFile,WrtStr);
MaxSpanLoadNo:=StrToInt(WrtStr);
MaxspanLoadNoBox.text:=wrtstr;
for i:=1 to MaxJointNo do
for j:=0 to 3 do
 begin
 read(MatFile,wrtstr);
 jointsGrid.cells[j,i]:=wrtstr;
 end;
for i:=1 to MaxJointNo do
for j:=0 to 6 do
 begin
 read(MatFile,wrtstr);
 restraintsGrid.cells[j,i]:=wrtstr;
 end;
for i:=1 to MaxElNo do
for j:=0 to 3 do
 begin
 read(MatFile,wrtstr);
 FrameGrid.cells[j,i]:=wrtstr;
 end;
for i:=1 to MaxElTypeNo do
for j:=0 to 5 do
 begin
 read(MatFile,wrtstr);
 ElementPropertiesGrid.cells[j,i]:=wrtstr;
 end;
for i:=1 to MaxJointLoadNo do
for j:=0 to 6 do
 begin
 read(MatFile,wrtstr);
 jointLoadsGrid.cells[j,i]:=wrtstr;
 end;
for i:=1 to MaxSpanLoadNo do
for j:=0 to 3 do
 begin
```

```
    read(MatFile,wrtstr);
    SpanLoadsgrid.cells[j,i]:=wrtstr;
    end;
  end;
end;
procedure Tform1.Save1Click(Sender: TObject);
var
Matfile:File of TMatfile;
i,j,n:integer;
wrtstr:TMatfile;
begin
  if SaveDialog1.execute then
  begin
  AssignFile(MatFile,SaveDialog1.FileName);
  ReWrite(MatFile);
  MaxJointNo:=StrToInt(MAxJointNoBox.text);
  wrtstr:=MaxJointNoBox.text;
  write(MatFile,wrtstr);
  MaxElNo:=StrToInt(MaxElNoBox.text);
  wrtstr:=MaxElNoBox.text;
  write(MatFile,wrtstr);
  MaxElTypeNo:=StrToInt(MaxElTypeNoBox.text);
  wrtstr:=MaxElTypeNoBox.text;
  write(MatFile,wrtstr);
  MaxJointLoadNo:=StrToInt(MaxJointLoadNoBox.text);
  wrtstr:=MaxJointLoadNoBox.text;
  write(MatFile,wrtstr);
  MaxSpanLoadNo:=StrToInt(MaxSpanLoadNoBox.text);
  wrtstr:=MaxSpanLoadNoBox.text;
  write(MatFile,wrtstr);
  for i:=1 to MaxJointNo do
  for j:=0 to 3 do
    begin
    wrtstr:=jointsGrid.cells[j,i];
    write(MatFile,wrtstr);
    end;
  for i:=1 to MaxJointNo do
  for j:=0 to 6 do
    begin
    wrtstr:=restraintsGrid.cells[j,i];
    write(MatFile,wrtstr);
    end;
  for i:=1 to MaxElNo do
  for j:=0 to 3 do
    begin
    wrtstr:=FrameGrid.cells[j,i];
    write(MatFile,wrtstr);
    end;
  for i:=1 to MaxElTypeNo do
```

```
 for j:=0 to 5 do
   begin
   wrtstr:=ElementPropertiesGrid.cells[j,i];
   write(MatFile,wrtstr);
   end;
 for i:=1 to MaxJointLoadNo do
 for j:=0 to 6 do
   begin
   wrtstr:=JointLoadsGrid.cells[j,i];
   write(MatFile,wrtstr);
   end;
 for i:=1 to MaxSpanLoadNo do
 for j:=0 to 3 do
   begin
   wrtstr:=spanloadsgrid.cells[j,i];
   write(MatFile,wrtstr);
   end;
  CloseFile(Matfile);
 end;
end;
procedure Tform1.SysMatrices1Click(Sender: TObject);
var i,j:integer;
begin
AllVisibleFalse;
 panelKsys.visible:=true;
 paneldsys.visible:=true;
 panelqsys.visible:=true;
 GSysStifnesGrid.RowCount:=nmax+1;
 GSysStifnesGrid.ColCount:=nmax+1;
 GSysDisplacementGrid.RowCount:=nmax+1;
 GSysJntLoadGrid.RowCount:=nmax+1;
 GSysStifnesGrid.visible:=true;
 GSysDisplacementGrid.visible:=true;
 GSysJntLoadGrid.visible:=true;
 for i:=1 to nmax do for j:=1 to nmax do
   begin
    GSysDisplacementGrid.cells[0,i]:=intToStr(i);
    GSysDisplacementGrid.cells[1,i]:=floattostr(SysDisplacementVector[i]);
   end;
end;
procedure Tform1.ShowCodeTable1Click(Sender: TObject);
begin
 AllVisibleFalse;
 panelncode.visible:=true;
 ncodeGrid.visible:=true;
end;
procedure Tform1.ElMatrices1Click(Sender: TObject);
begin
panelelno.visible:=true;
```

```
elnobox.visible:=true;
Go.visible:=true;
end;
procedure Tform1.goClick(Sender: TObject);
var m,i,j,elno,z,nodei,nodej:integer;
begin
AllVisibleFalse;
GElEndForcesGrid.visible:=true;
GelStifnesGrid.visible:=true;
GelDisplacementgrid.visible:=true;
 panelkel.visible:=true;
 paneldel.visible:=true;
 panelpel.visible:=true;
 begin
 m:=strtoint(elnobox.text);
 Area:=StrToFloat(ElementListGrid.cells[1,m]);
 MofE:=StrToFloat(ElementListGrid.cells[2,m]);
 Imy:=StrToFloat(ElementListGrid.cells[3,m]);
 Imz:=StrToFloat(ElementListGrid.cells[4,m]);
 G:=StrToFloat(ElementListGrid.cells[5,m]);
 L:=StrToFloat(ElementListGrid.cells[6,m]);
 Cx:=StrToFloat(ElementListGrid.cells[7,m]);
 Cy:=StrToFloat(ElementListGrid.cells[8,m]);
 Cz:=StrToFloat(ElementListGrid.cells[9,m]);
 Alfa:=0;
 GetTransformMatrix(Cx,Cy,Cz,Alfa,TransformMatrix);
 GetGElStifnesMatrix(LElStifnesMatrix,GELStifnesMatrix);
 end;
 begin
 for i:=1 to maxelno do
  begin
  if frameGrid.cells[0,i]=inttostr(m) then
   begin
    nodei:=StrToint(framegrid.cells[1,i]);
    nodej:=StrToint(framegrid.cells[2,i]);
    for j:=1 to maxjointno do
     begin
      if strtoint(dofGrid.cells[0,j-1])=nodei then
       begin
        for z:=1 to 6 do
         begin
         if strtoint(dofgrid.cells[z,j-1])=0 then GElDisplacementVector[z]:=0;
         if strtoint(dofGrid.cells[z,j-1])>0 then
         GElDisplacementVector[z]:=SysDisplacementVector[strtoint(DofGrid.cells[z,j-1])];
         end;
        end;
      if strtoint(dofGrid.cells[0,j-1])=nodej then
       begin
        for z:=7 to 12 do
```

```
        begin
        if strtoint(dofgrid.cells[z-6,j-1])=0 then GElDisplacementVector[z-6]:=0;
        if strtoint(dofGrid.cells[z-6,j-1])>0 then
      GElDisplacementVector[z]:=SysDisplacementVector[strtoint(DofGrid.cells[z-6,j-1])];
        end;
       end;
     end;
  end;
 end;
end;
begin
 for i:=1 to 12 do GElReactionVector[i]:=0;
end;
begin
 for i:=1 to 12 do
 for j:=1 to 12 do
 begin
 GElDisplacementGrid.cells[0,i]:=inttostr(i);
 GElDisplacementGrid.cells[1,i]:=floattostr(GElDisplacementvector[i]);
 GElStifnesGrid.cells[0,i]:=inttostr(i);
 GElStifnesGrid.cells[i,0]:=inttostr(i);
 GElStifnesGrid.cells[j,i]:=floattostr(GElStifnesMatrix[i,j]);
 end;
end;
begin
 MatrixDotVector(GElStifnesMatrix,GElDisplacementVector,GElReactionVector);
end;
begin
 for i:=1 to 12 do
 for j:=1 to 12 do
 begin
 GElEndForcesGrid.Cells[0,i]:=inttostr(i);
 GElEndForcesGrid.Cells[1,i]:=Floattostr(GElReactionVector[i]);
 end;
end;
end;
procedure TForm1.AllVisibleFalse;
begin
  PanelJntDisplc.visible:=false;
  JointResultsGrid.Visible:=False;
  JointsGrid.Visible:=False;
  frameGrid.Visible:=False;
  restraintsgrid.Visible:=False;
  ElementPropertiesGrid.Visible:=False;
  JointLoadsGrid.Visible:=False;
  JointsPanel.visible:=false;
  FramesPanel.visible:=false;
  RestraintsPanel.visible:=false;
  ElPropPanel.visible:=false;
```

```
  JntLoadPanel.visible:=false;
  MaxJointNoBox.visible:=false;
  MaxElNoBox.visible:=false;
  MaxElTypeNoBox.visible:=false;
  MaxJointLoadNoBox.visible:=false;
  NcodeGrid.Visible:=False;
  panelKsys.visible:=false;
  paneldsys.visible:=false;
  panelqsys.visible:=false;
  GSysStifnesGrid.visible:=false;
  GSysDisplacementGrid.visible:=false;
  GSysJntLoadGrid.visible:=false;
  panelncode.visible:=false;
  label1.visible:=false;
  label2.visible:=false;
  label3.visible:=false;
  label4.visible:=false;
  panelelno.visible:=false;
  elnobox.visible:=false;
  Go.visible:=false;
  GElEndForcesGrid.visible:=false;
  GelStifnesGrid.visible:=false;
  GelDisplacementgrid.visible:=false;
  panelkel.visible:=false;
  paneldel.visible:=false;
  panelpel.visible:=false;
end;

procedure Tform1.DataPanelClick(Sender: TObject);
begin
AllVisibleFalse;
  Label1.visible:=true;
  Label2.visible:=true;
  Label3.visible:=true;
  Label4.visible:=true;
  JointsGrid.Visible:=true;
  frameGrid.Visible:=true;
  restraintsgrid.Visible:=true;
  JointsGrid.Visible:=true;
  ElementPropertiesGrid.Visible:=true;
  JointLoadsGrid.Visible:=true;
  JointsPanel.visible:=true;
  FramesPanel.visible:=true;
  RestraintsPanel.visible:=true;
  ElPropPanel.visible:=true;
  JntLoadPanel.visible:=true;
  MaxJointNoBox.visible:=true;
  MaxElNoBox.visible:=true;
  MaxElTypeNoBox.visible:=true;
```

```
  MaxJointLoadNoBox.visible:=true;
end;
procedure Tform1.Displacements1Click(Sender: TObject);
var i:integer;
    j:integer;
    JointNo:integer;
    DofNo:integer;
    strg:string;
begin
  AllVisibleFalse;
  JointResultsGrid.RowCount:=MaxJointNo+1;
  JointResultsGrid.Visible:=True;
  panelJntDisplc.visible:=true;
  JointResultsGrid.Cells[0,0]:='  Joint  No ';
  JointResultsGrid.Cells[1,0]:='      U ( x )';
  JointResultsGrid.Cells[2,0]:='      U ( y )';
  JointResultsGrid.Cells[3,0]:='      U ( z )';
  JointResultsGrid.Cells[4,0]:='      R ( x )';
  JointResultsGrid.Cells[5,0]:='      R ( y )' ;
  JointResultsGrid.Cells[6,0]:='      R ( z )'  ;
  for i:=0 to MaxJointNo-1 do
    begin
     JointNo:=StrToInt(DofGrid.Cells[0,i]);
     for j:= 1 to 6 do
     begin
       JointResultsGrid.Cells[0,i+1]:=inttostr(i+1);
       DofNo:=StrToInt(DofGrid.Cells[j,JointNo-1]);
       Str(SysDisplacementVector[DofNo]:9:4,strg);
       JointResultsGrid.Cells[j,JointNo]:=strg;
     end;
    end;
 end;
 end.
```

```
unit Systypes;

interface

uses    SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
        Forms, Dialogs, Grids, StdCtrls, ExtCtrls, Menus;
const ASize=72;

type
   SysMat=array[0..ASize,0..ASize] of real;
   SysVec=array[0..Asize] of real;
   ElMat=array[1..12,1..12] of real;
   ElVec=array[1..12] of real;
   TMatFile=String[30];

var
SysStifnesMatrix:SysMat;
SysJointLoadVector:SysVec;
SysDisplacementVector:SysVec;
GElStifnesMatrix:ElMat;
GElDisplacementVector:ElVec;
GElReactionVector:ElVec;
TransformMatrix:ElMat;
T_TransformMatrix:ElMat;
LElStifnesMatrix:ElMat;
MaxElNo,MaxJointNo,MaxJointLoadNo,MaxSpanLoadNo,MaxElTypeNo,Nmax:integer;
Cx,Cy,Cz,Alfa:real;
Area,MofE,Imx,Imy,Imz,L,G:real;

implementation

end.
```

```
unit Grdap3da;

interface

uses SysTypes,grids,SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
    Forms, Dialogs,StdCtrls, ExtCtrls, Menus,MatAp3DA;

procedure CreateNcodeGrid(RestraintsGrid,FrameGrid:TstringGrid;var
DofGrid,NcodeGrid:tstringGrid
;var nmax:integer);

procedure GetSysJntLoadGrid(JointLoadsGrid,DofGrid:TStringGrid;var
SysJointLoadVector:SysVec);

procedure
CreateElementListGrid(FrameGrid,JointsGrid,ElementPropertiesGrid:TstringGrid;
var ElementListGrid:TstringGrid);

implementation

{*********************************************************************}
procedure
CreateElementListGrid(FrameGrid,JointsGrid,ElementPropertiesGrid:TstringGrid;
var ElementListGrid:TstringGrid);

var TypeNo,ElNo,Nodei,nodej,i,z,j:integer;
   Xi,Xj,Yi,Yj,Zi,Zj:real;
begin
  for i:=1 to MaxElNo do
    begin
    TypeNo:=strtoint(frameGrid.cells[3,i]);
    nodei:=strtoint(frameGrid.cells[1,i]);
    nodej:=strtoint(frameGrid.cells[2,i]);
    Elno:=strtoint(frameGrid.cells[0,i]);
      begin
      for z:=1 to MaxElTypeNo do
        begin
        if TypeNo=StrToint(ElementPropertiesGrid.cells[0,z]) then
          begin
          ElementListGrid.cells[0,Elno]:=inttostr(elno);
          ElementListGrid.cells[1,Elno]:=ElementPropertiesGrid.cells[1,z];
          ElementListGrid.cells[2,Elno]:=ElementPropertiesGrid.cells[2,z];
          ElementListGrid.cells[3,Elno]:=ElementPropertiesGrid.cells[3,z];
          ElementListGrid.cells[4,Elno]:=ElementPropertiesGrid.cells[4,z];
          ElementListGrid.cells[5,Elno]:=ElementPropertiesGrid.cells[5,z];
          end;
        end;
      end;
```

```
begin
   for j:=1 to MaxJointNo do
    begin
    if strtoint(JointsGrid.cells[0,j])=nodei then
     begin
     Xi:=strtofloat(JointsGrid.cells[1,j]);
     Yi:=strtofloat(JointsGrid.cells[2,j]);
     Zi:=strtofloat(JointsGrid.cells[3,j]);
     end;
    if strtoint(JointsGrid.cells[0,j])=nodej then
     begin
     Xj:=strtofloat(JointsGrid.cells[1,j]);
     Yj:=strtofloat(JointsGrid.cells[2,j]);
     Zj:=strtofloat(JointsGrid.cells[3,j]);
     end;
    end;
    L:=Sqrt(sqr(xj-xi)+sqr(yj-yi)+sqr(Zj-Zi));
    Cx:=(Xj-Xi)/Sqrt(sqr(xj-xi)+sqr(yj-yi)+sqr(Zj-Zi));
    Cy:=(Yj-Yi)/Sqrt(sqr(xj-xi)+sqr(yj-yi)+sqr(Zj-Zi));
    Cz:=(Zj-Zi)/Sqrt(sqr(xj-xi)+sqr(yj-yi)+sqr(Zj-Zi));
    ElementListGrid.cells[6,Elno]:=FloatToStr(L);
    ElementListGrid.cells[7,Elno]:=FloatToStr(Cx);
    ElementListGrid.cells[8,Elno]:=FloatToStr(Cy);
    ElementListGrid.cells[9,Elno]:=FloatToStr(Cz);
   end;
  end;
end;
{*********************************************************************}
procedure CreateNcodeGrid(RestraintsGrid,FrameGrid:TstringGrid;var
DofGrid,NcodeGrid:tstringGrid
;var nmax:integer);
var el_no,nodei,nodej,b,i,j:integer;
begin
 begin
  b:=0;
   begin
   for i:=0 to MaxJointNo-1 do
   for j:=1 to 6 do
    begin
    DofGrid.cells[0,i]:=restraintsGrid.cells[0,i+1];
    if strtoint(restraintsGrid.cells[j,i+1])=1 then
     begin
     dofGrid.cells[j,i]:=IntToStr(0);
     end;
    if strtoint(restraintsGrid.cells[j,i+1])=0 then
     begin
     b:=b+1;
     nmax:=b;
     dofGrid.cells[j,i]:=inttostr(b);
```

```
      end;
     end;
    end;
  end;
  begin
  NcodeGrid.colcount:=MaxElNo;
  DofGrid.RowCount:=MaxJointNo;
  for i:=1 to MaxElNo do
   begin
   el_no:=strtoint(FrameGrid.cells[0,i]);
   nodei:=strtoint(FrameGrid.cells[1,i]);
   nodej:=strtoint(FrameGrid.cells[2,i]);
   for j:=1 to MaxJointNo do
    begin
    if dofgrid.cells[0,j-1]=inttostr(nodei) then
     begin
     Ncodegrid.cells[el_no-1,0]:=inttostr(El_no);
     Ncodegrid.cells[el_no-1,1]:=dofgrid.cells[1,j-1];
     Ncodegrid.cells[el_no-1,2]:=dofgrid.cells[2,j-1];
     Ncodegrid.cells[el_no-1,3]:=dofgrid.cells[3,j-1];
     Ncodegrid.cells[el_no-1,4]:=dofgrid.cells[4,j-1];
     Ncodegrid.cells[el_no-1,5]:=dofgrid.cells[5,j-1];
     Ncodegrid.cells[el_no-1,6]:=dofgrid.cells[6,j-1];
     end;
    if Dofgrid.cells[0,j-1]=inttostr(nodej) then
     begin
     Ncodegrid.cells[el_no-1,7]:=dofgrid.cells[1,j-1];
     Ncodegrid.cells[el_no-1,8]:=dofgrid.cells[2,j-1];
     Ncodegrid.cells[el_no-1,9]:=dofgrid.cells[3,j-1];
     Ncodegrid.cells[el_no-1,10]:=dofgrid.cells[4,j-1];
     Ncodegrid.cells[el_no-1,11]:=dofgrid.cells[5,j-1];
     Ncodegrid.cells[el_no-1,12]:=dofgrid.cells[6,j-1];
     end;
    end;
   end;
  end;
 end; {Create Ncode}
 {*************************************************************************}
 procedure GetSysJntLoadGrid(JointLoadsGrid,DofGrid:TStringGrid;var
 SysJointLoadVector:SysVec);
 var
 z,i,j,JointNo:integer;
 begin {GetGSysJntLoadGrid}
  for i:=1 to MaxJointLoadNo do
  begin
  JointNo:=StrToint(JointLoadsGrid.cells[0,i]);
  for j:=1 to MaxJointNo do
   begin
    if StrToint(DofGrid.cells[0,j-1])=JointNo then
```

```
      begin
      SysJointLoadVector[StrToint(DofGrid.cells[1,j-
  1])]:=StrToFloat(JointLoadsGrid.cells[1,i]);
      SysJointLoadVector[StrToint(DofGrid.cells[2,j-
  1])]:=StrToFloat(JointLoadsGrid.cells[2,i]);
      SysJointLoadVector[StrToint(DofGrid.cells[3,j-
  1])]:=StrToFloat(JointLoadsGrid.cells[3,i]);
      SysJointLoadVector[StrToint(DofGrid.cells[4,j-
  1])]:=StrToFloat(JointLoadsGrid.cells[4,i]);
      SysJointLoadVector[StrToint(DofGrid.cells[5,j-
  1])]:=StrToFloat(JointLoadsGrid.cells[5,i]);
      SysJointLoadVector[StrToint(DofGrid.cells[6,j-
  1])]:=StrToFloat(JointLoadsGrid.cells[6,i]);
      end;
    end;
  end;
end;{GetGSysJntLoadGrid}
{*****************************************************************}
end.
```

```
unit Matap3da;

interface

uses  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
      Forms, Dialogs, Grids, StdCtrls, ExtCtrls, Menus,SysTypes;

function
GetSysDisplacements(SysStifnesMatrix:SysMat;SysLoadVector:SysVec;Nmax:integer;
var SysDisplacementvector:Sysvec):real;

procedure MatrixDotVector(A:ElMat;B:ElVec;var C:ElVec);

procedure MatrixDotMatrix(ArowNo,AcolNo,BcolNo:integer;A,B:ElMat;var C:ElMat);

procedure GetGElStifnesMatrix(var LElStifnesMatrix,GELStifnesMatrix:ElMat);

procedure GetTransformMatrix(Cx,Cy,Cz,Alfa:real;var TransformMatrix:ElMat);

implementation
{*************************************************************************}
function
GetSysDisplacements(SysStifnesMatrix:SysMat;SysLoadVector:SysVec;Nmax:integer;
var SysDisplacementvector:Sysvec):real;
var r,sum:real;
   m,i,z,j,t:integer;
 Begin
  Begin
  for i:=1 to (nmax-1) do
   begin
   for m:=i+1 to nmax do
    begin
    r:=SysStifnesMatrix[m,i]/SysStifnesMatrix[i,i];
     begin
     for z:=1 to nmax do
      begin
      SysStifnesMatrix[m,z]:=SysStifnesMatrix[m,z]-(r*SysStifnesMatrix[i,z]);
      end;
      SysLoadVector[m]:=SysLoadVector[m]-r*SysLoadVector[i];
     end;
    end;
   end;
  end;
  Begin
  SysDisplacementvector[nmax]:=SysLoadVector[nmax]/SysStifnesMatrix[nmax,nmax];
  for j:=(nmax-1) downto 1 do
   begin
   Sum:=0;
   for t:=nmax downto (j+1) do
    begin
```

```
      Sum:=Sum+SysStifnesMatrix[j,t]*SysDisplacementvector[t];
      end;
      SysDisplacementvector[j]:=(SysLoadVector[j]-Sum)/SysStifnesMatrix[j,j];
    end;
  end;
 end; {GetGSysDisplacements}
{*************************************************************************}
procedure MatrixDotVector(A:ElMat;B:ElVec;var C:ElVec);
var i,k:integer;
   sum:real;
begin
  for i:=1 to 12 do
  Begin
    SUM:=0;
    for k:=1 to 12 do
    Begin
      SUM:=SUM+A[i,k]*B[k];
    end;
    C[i]:=SUM;
    if (c[i]>-0.0001) and (c[i]<0.0001) then c[i]:=0;
  end;
end; {MatrixDotVector}
{*************************************************************************}
procedure MatrixDotMatrix(ArowNo,AcolNo,BcolNo:integer;A,B:ElMat;var C:ElMat);
var
i,j,k:integer;
sum:real;
Begin
  for i:=1 to ArowNo do
  Begin
   for j:=1 to BcolNo do
   Begin
    SUM:=0;
    for k:=1 to AcolNo do
    Begin
      SUM:=SUM+A[i,k]*B[k,j];
    end;
    C[i,j]:=SUM;
    if (c[i,j]>-0.000001) and (c[i,j]<0.000001) then c[i,j]:=0;
   end;
  end;
 end; {MatrixDotMatrix}
{*************************************************************************}
procedure GetGElStifnesMatrix(var LElStifnesMatrix,GELStifnesMatrix:ElMat);
var i,j:integer;
begin
  begin
  for i:=1 to 12 do for j:=1 to 12 do LElStifnesMatrix[i,j]:=0;
  end;
  begin
```

```
  LElStifnesMatrix[1,1]:=MofE*Area/L;
  LElStifnesMatrix[2,2]:=12*MofE*Imz/(L*L*L);
  LElStifnesMatrix[3,3]:=12*MofE*Imy/(L*L*L);
  LElStifnesMatrix[4,4]:=G*Imx/L;
  LElStifnesMatrix[5,5]:=4*MofE*Imy/L;
  LElStifnesMatrix[6,6]:=4*MofE*Imz/L;
  LElStifnesMatrix[5,3]:=-1*6*MofE*Imy/(L*L);
  LElStifnesMatrix[6,2]:=6*MofE*Imz/(L*L);
  LElStifnesMatrix[7,1]:=-1*LElStifnesMatrix[1,1];
  LElStifnesMatrix[8,2]:=-1*LElStifnesMatrix[2,2];
  LElStifnesMatrix[9,3]:=-1*LElStifnesMatrix[3,3];
  LElStifnesMatrix[10,4]:=-1*LElStifnesMatrix[4,4];
  LElStifnesMatrix[11,5]:=0.5*LElStifnesMatrix[5,5];
  LElStifnesMatrix[12,6]:=0.5*LElStifnesMatrix[6,6];
  LElStifnesMatrix[11,3]:=LElStifnesMatrix[5,3];
  LElStifnesMatrix[12,2]:=LElStifnesMatrix[6,2];
  LElStifnesMatrix[9,5]:=-1*LElStifnesMatrix[5,3];
  LElStifnesMatrix[8,6]:=-1*LElStifnesMatrix[6,2];
  LElStifnesMatrix[7,7]:=LElStifnesMatrix[1,1];
  LElStifnesMatrix[8,8]:=LElStifnesMatrix[2,2];
  LElStifnesMatrix[9,9]:=LElStifnesMatrix[3,3];
  LElStifnesMatrix[10,10]:=LElStifnesMatrix[4,4];
  LElStifnesMatrix[11,11]:=LElStifnesMatrix[5,5];
  LElStifnesMatrix[12,12]:=LElStifnesMatrix[6,6];
  LElStifnesMatrix[11,9]:=-1*LElStifnesMatrix[5,3];
  LElStifnesMatrix[12,8]:=-1*LElStifnesMatrix[6,2];
 for i:=1 to 12 do
 for j:=1 to 12 do
   begin
     LElStifnesMatrix[i,j]:=LElStifnesMatrix[j,i];
   end;
end;
MatrixDotMatrix(12,12,12,LElStifnesMatrix,TransformMatrix,GElStifnesMatrix);
for i:= 1 to 12 do
for j:= 1 to 12 do
  begin
  T_TransformMatrix[i,j]:=TransformMatrix[j,i];
  end;
 MatrixDotMatrix(12,12,12,T_TransformMatrix,GElStifnesMatrix,GElStifnesMatrix);
end ;{Get_GElStifMatrix}
{*************************************************************************}
procedure GetTransformMatrix(Cx,Cy,Cz,Alfa:real;var TransformMatrix:ElMat);
var i,j:integer;
   sum:real;
begin
 begin
   for i:=1 to 12 do for j:=1 to 12 do TransformMatrix[i,j]:=0;
 end;
 TransformMatrix[1,1]:=Cx;
 TransformMatrix[1,2]:=Cy;
```

```
TransformMatrix[1,3]:=Cz;
if (Cx=0) and (Cz=0) then
  begin
   Cx:=1e-9;
   Cz:=1e-9;
  end;
sum:=(Sqrt(Sqr(Cx)+Sqr(Cz)));
TransformMatrix[2,1]:=(-1*Cx*Cy*Cos(Alfa)-Cz*Sin(Alfa))/sum;
TransformMatrix[2,2]:=Sqrt(Sqr(Cx)+Sqr(Cz))*Cos(Alfa);
TransformMatrix[2,3]:=(-1*Cy*Cz*Cos(Alfa)+Cx*Sin(Alfa))/sum;
TransformMatrix[3,1]:=(Cx*Cy*Sin(Alfa)-Cz*Cos(Alfa))/sum;
TransformMatrix[3,2]:=-1*Sqrt(Sqr(Cx)+Sqr(Cz))*Sin(Alfa);
TransformMatrix[3,3]:=(Cy*Cz*Sin(Alfa)+Cx*Cos(Alfa))/sum;
begin
  for i:=4 to 6 do
  for j:=4 to 6 do
  begin
    TransformMatrix[i,j]:=TransformMatrix[i-3,j-3];
  end;
end;
begin
  for i:=7 to 9 do
  for j:=7 to 9 do
  begin
    TransformMatrix[i,j]:=TransformMatrix[i-6,j-6];
  end;
end;
 begin
  for i:=10 to 12 do
  for j:=10 to 12 do
  begin
    TransformMatrix[i,j]:=TransformMatrix[i-9,j-9];
  end;
 end;
end;

end.
```