****

**KOCAELİ ÜNİVERSİTESİ  
BİLGİSAYAR MÜHENDİSLİĞİ**

**MATEMATİK ÇARKI**

Edanur TOSUN, Merve TAFRALI

Bilgisayar Mühendisliği Bölümü

Kocaeli Üniversitesi

eedanurtosun@gmail.com, mervetafralii@gmail.com

**Özet**

Bu projede, amacımız kullanıcıdan alınan matris ile Öz değer, Öz vektör, Schur Teoremi ve Nilpotent Matrisi hesaplayan bir program yazmaktır. Yani Lineer Cebir ve Matematik derslerinde görülen teorik bilgilerin uygulamada kullanılması asıl amacımızdır. Yapılan araştırmalardan farklı olarak bu projede dört farklı işlemi yapmak mümkündür. Dört farklı işlem matematik çarkında birleştirilerek kullanıcıya kolaylık kazandırılmıştır. Projenin verileri dosyaya yazdırılarak elde edilen sonuçlar bir text dosyasında tutulmuştur. Proje C/C++ dilinde code:bloks IDE’si kullanılarak yazılmıştır. Çarkın yapması gereken dört temel işlemin hepsinde başarım oranı %40’a yakın değerler taşımaktadır.

**1. Giriş**

Projede matematik çarkı yaptık, Matematik ve Lineer Cebirde gördüğümüz bilgilerin kodlamaya dökülmesi ve Programlama-1 Programlama-2 derslerinde öğrenilen bilgilerin kullanılmasını amaçladık.

Projede kullanıcıdan başlangıç olarak çarkın kaç kez döneceğini alıp, rastgele üretilen sayıya göre verilen işlemlerden hangisinin yapılacağını seçmekteyiz. Yapılacak işlemler Öz Değer, Öz vektör , Schur Teoremi ve Nilpotent Matrisin bulunmasıdır.

Projenin diğer yapılan çalışmalardan farkı bu dört işlemi aynı anda kullanıcıya sunmasıdır. Diğer çalışmalarda genellikle tek bir işlem yapılmaktadır. Web ortamında online olarak hesap yapan çalışmalar da 3\*3 yada 2\*2 boyutlarında hesaplamalar yapmaktadır.[1] Bizim yaptığımız çalışmada amacımız kullanıcıya boyut kısıtlaması yapmadan öz değer,öz vektör, Schur teoremi ve Nilpotent matrisin bulunmasını sağlamaktır. Yapılan çalışmamızda kütüphane kullanmamız 5\*5 boyutlu matrise kadar uygun olmadığı için tüm matris hesaplamaları başarılı olmamıştır. Sadece 2 boyutlu matrislerde başarım oranı mevcuttur. Çalışmada istenen matematik çarkı ise projenin tüm ihtiyaçlarını hatasız şekilde karşılamaktadır.

Öz değer hesabı için iki boyutta %100 doğru sonuç almamıza karşı öz vektör ve Schur teoremlerinde başarılı işlemler geliştirilmemiştir. Nilpotent matrisin -15 ve +15 arasında rastgele değerler alıp kontrol aşaması ise başarılı bir şekilde yapılmıştır. Projenin sonucunda matematiksel kavramlar anlaşılmış olup kullanıcı odaklı bir program yazılmıştır.

**2. Temel Bilgiler**

Projemizi yazarken Code Blocks programını kullandık ve C programlama dilinde yazdık. Yazdığımız kodların doğruluğunu kontrol edebilmek için internette bulunan uygulamaları kullandık.[2]

Özdeğer :

Öz değer problemi

Ax = λx (1)

lineer sistemi X = 0 dışında çözümlere sahip olacak şekilde λ sayılarının belirlenmesini gerektirir. Bu sayılara öz değerler denir.[3]

Schur Teoremi:

A bir kare matris olsun. A matrisinin elemanlarının karesi ile A matrisinin özdeğerlerinin karesi hesaplanır.Hangisi daha büyükse sınır ona göre yazılır.[4]

Öz Vektör Bulma:

(A - λI) (x) = 0 (2)

denkleminde bulunan (özdeğer) değerlerinin her birinin yerine koyulmasıyla bulunan sonuçlara o matrisin öz vektörleri denir.[5]

Nilpotent Matris:

A bir kare matris olsun. n>2 ,

An = 0 (3)

ise A matrisine nilpotent matris denir.[6]

**3. Diğer Bölümler**

Bu projemizde matematik çarkı yaptık. C programlama dilinde yazmış olduğumuz bu program başlangıç aşamasında ekrana çarkımızın hangi matematiksel işlemler yapacağını yazdırıyor. Öncelikle programımız kullanıcıdan çarkın kaç kez döneceğini hesaplamak için sayı istiyor. Girilen bu sayıya göre çark dönüyor. Kullanıcının girdiği sayıya kadar 0-241 arasında rastgele sayılar üretiyor. Üretilen bu son sayı bizim hangi işlemi yapacağımızı belirliyor.Son sayının 4’e göre modunu aldığımızda sonuç 1 ise özdeğer bulma işlemi, 2 ise schur teoremi, 3 ise öz vektör bulma, 0 ise nilpotent matris işlemini yapıyor.

Programı çalıştırdık çarkın kaç kez dönmesini istediğimizi yazdık, girdiğimiz sayı kadar çark döndü ve rastgele sayılar üretildi.

3.1.

Son sayıya göre mod işleminin sonucunda 1 yani özdeğer bulma işleminin geldiğini varsayalım. Program bizden n\*n matris için boyut istiyor. Girilen boyut kadar matris oluşturuluyor ve matris elemanlarını programa giriyoruz. Program daha sonra bize özdeğerleri buluyor ve ekrana yazdırıyor. Kullanıcı tarafından girilen tüm veriler ozdeger.txt dosyasına yazdırılıyor.

3.2.

Şimdi mod işlemi sonucunda 2 yani schur teoreminin geldiğini varsayalım. Program bizden n\*n matris oluşturabilmek için boyut istiyor. Girilen boyut işlemine göre matrisin elemanlarını yazıp oluşturuyoruz. Kullanıcı tarafından girilen tüm veriler schur.txt dosyasına yazdırılıyor.

3.3.

Mod işlemine göre 3 yani öz vektör bulma işleminin geldiğini varsayalım. Programa istenilen verileri giriyoruz. Matris oluşturmak için bir boyut girdikten sonra matrisi oluşturabilmek için elemanlarını da giriyoruz. Öz vektörün hesaplanabilmesi için önce özdeğerlerin bulunması gerekiyor bu yüzden girilen matrisi öz değer bul fonksiyonuna gönderiyoruz. Döndürülen sonuç ile yapılan matematiksel işlemler sonucunda öz vektörler hesaplanıyor. Kullanıcı tarafından aldığımız tüm veriler ozvektor.xt dosyasına yazdırıyoruz.

3.4.

Son olarak da mod işlemi sonucunda 0 yani nilpotent matris işleminin geldiğini varsayalım. Matrisi oluşturabilmek için bizden boyut istiyor. Boyutu girdikten sonra program, -15 ile +15 arasında rastgele sayılar üretip matris elemanlarına atıyor. Bu matrisin nilpotent mi değil mi olduğunu anlamak için matris çarpma işlemi yapıyoruz ve matrisi kendisiyle çarpıyoruz. Çarpım sonucunda 0 matris oluşursa ekrana “Matrisiniz Nilpotent matristir! “ değilse “Matrisiniz Nilpotent matris değildir!” yazdırıyoruz. Program tüm yaptığı ara işlemleri, kullanıcıdan aldığı verileri nilpotent.txt dosyasına yazdırıyor.[7]

**4. Sonuçlar**

Projemizi çalıştırdığımız zaman matematik çarkımız kullanıcıdan aldığımız sayıya göre dönüyor , rastgele atanan sayıların sonuncusuna göre mod işlemi aldırıyoruz. Alınan mod işlemine göre matematik çarkının gerçekleştirmesi gereken dört işlemden biri belirleniyor. Özdeğer bulma fonksiyonuna geldiği zaman kullanıcıdan öncelikle boyut al fonksiyonumuzla boyutu alıyoruz. Daha sonra matris al fonksiyonumuzla da girilen boyut işlemine göre matris elemanlarını alabiliyoruz. Kullanıcının girdiği bu değerleri, boyutu ve matris elemanlarını fonksiyon şeklinde tanımlayıp aldığımız için yazdığımız diğer fonksiyonlara da gönderebiliyoruz.

Öz değer bulma işleminde programımız sadece 2\*2 matrisler için çözüm bulabiliyor.Yani 3, 4 veya daha fazla boyutlu matrisler için öz değer bulma fonksiyonumuz hesaplama yapamıyor.

Program çalıştırılıp schur teoremi geldiğinde, kullanıcıdan yine boyut ve matrisin eleman bilgilerini girmesini istiyoruz. Girilen bu bilgilere göre teoremin uygulanabilmesi için öncelikle öz değerlerin belirlenmesi gerekiyor. Schur teoreminde aldığımız matrisi öz değer bulma fonksiyonuna gönderdiğimiz zaman, öz değer bul fonksiyonunda gönderilen matris bilgilerine göre sadece 2\*2 matris için özdeğer bulduğundan program cevabı doğru buluyor olsa bile 2\*2 kare matris için iki tane özdeğer olduğundan bu iki parametreyi schur teoremine gönderemiyoruz. bu yüzdende schur teoremi fonksiyonumuza özdeğer bilgileri gelmediği için girilen matris elemanları da göz önünde bulundurularak bir kıyaslama yapılamıyor.

Program çalıştırılıp öz vektör bulma işlemi geldiğinde aynı şekilde matrisimizin boyut bilgisini ve matris elemanlarını aldıktan sonra bu bilgileri öz değer bulan fonksiyonumuza gönderiyoruz. Sadece 2\*2 boyutlu matrisin öz değerlerini hesaplayabildiğimiz için kullanıcı 2’den fazla bir boyut bilgisi girdiğinde program herhangi bir sonuç üretemediği için öz vektör işlemini yapamıyor. Kullanıcı 2\*2 boyutlu bir matris bilgisi girmiş olsun. 2\*2 matris için program öz değer bulabiliyor, ancak bunları öz vektör fonksiyonuna gönderebilsek bile öz vektör bulma için gereken matematiksel denklemleri programlama diline uygun tanımlayamadığımız için öz vektör bulamıyoruz.

Nilpotent işleminde de girilen boyuta göre matrise -15 ile +15 arasında rastgele sayılar atanıyor. Biz matrisin sadece bir kez kendisiyle çarpıyoruz. Nilpotent ise nilpotent , değil ise nilpotent değil yazdırıyoruz. Matris ikinci değil üçüncü kez kendisiyle çarpımı sonucunda da sıfır matris olabilir. Bizim kullandığımız fonksiyon bunu bir kere kontrol ediyor.

5\*5 ve daha fazla boyut için hazır kütüphane kullanılabiliyor fakat biz bulduğumuz kütüphaneyi kendi ortamlarımızda aktif edemedik. Bu yüzden kodumuzda dört işlemden herhangi birini yapabilen hazır kütüphane kullanmadık.

Program çalıştığı sürece kullanıcıdan aldığımız tüm bilgileri de gerekli txt dosyalına yazdırdık.

**5. Kaynakça**

[1] <http://www.wolframalpha.com/widgets/view.jsp?id=9aa01caf50c9307e9dabe159c9068c41>

[2] <https://matrixcalc.org/tr/vectors.html>

[3] <https://abdullahistemi.files.wordpress.com/2017/01/unite09.pdf>

[4] <http://www.ufukcevik.com/dosyalar/schur.teo.pdf>

[5] <https://abdullahistemi.files.wordpress.com/2017/01/unite09.pdf>

[6] <http://kisi.deu.edu.tr//kemal.sehirli/MATRISLER.pdf>

<https://www.youtube.com/watch?v=lFAKDQoDX5k>

<http://kisi.deu.edu.tr//kemal.sehirli/MATRISLER.pdf>

[7] <http://sanalkurs.net/c-de-dosyaya-yazdirma-bolum-1-2121.html>

Deitel,H. M., Deitel P. J. “C ve C++ “, Sistem Yayıncılık, 2010

8 Determinant hesabı :

<http://bilgisayarkavramlari.sadievrenseker.com/2012/05/01/matrisin-determinanti-matrix-determinant/>

9 Dosya işlemleri

2016-2017 Programlama-1, Programlama-2 ders sunumları

10 İkinci dereceden denklem köklerinin bulunması : http://www.matematiktutkusu.com/forum/matematik-formulleri/1671-ikinci-dereceden-bir-bilinmeyenli-denklemlerin-kokleri-delta-formulu.html