T.C.
RECEP TAYYİP ERDOĞAN ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

MATLAB GUI'DE İNTERAKTİF ISI DENKLEMLERİ VE UYGULAMALARI

GAMZE ŞAHİN

TEZ DANIŞMANI
DR. ÖĞR. ÜYESİ İSHAK CUMHUR
TEZ JÜRİLERİ
PROF. DR. KADİR KUTLU
DOÇ. DR. AHMET GÖKDOĞAN

YÜKSEK LİSANS TEZİ
MATEMATİK ANABİLİM DALI

RİZE-2019

T.C.

RECEP TAYYİP ERDOĞAN ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

MATLAB GUI'DE İNTERAKTİF ISI DENKLEMLERİ VE UYGULAMALARI

Dr. Öğr. Üyesi İshak CUMHUR danışmanlığında, Gamze ŞAHİN tarafından hazırlanan bu çalışma, Enstitü Yönetim Kurulu kararıyla oluşturulan jüri tarafından 24/07/2019 tarihinde Matematik Anabilim Dalı'nda **YÜKSEK LİSANS** tezi olarak kabul edilmiştir.

Jüri Üyeleri	**Unvanı Adı Soyadı**	**İmzası**
Başkan	: Prof. Dr. Kadir KUTLU	
Üye	: Doç. Dr. Ahmet GÖKDOĞAN	
Üye	: Dr. Öğr. Üyesi İshak CUMHUR	

Doç. Dr. Ferhat KALAYCI

FEN BİLİMLERİ ENSTİTÜSÜ MÜDÜRÜ

ÖNSÖZ

Yüksek lisans öğrencisi olarak beni kabul eden ve eğitimim boyunca desteğini, tüm bilgisini ve deneyimlerini benden esirgemeyen, değerli hocam Dr. Öğr. Üyesi İshak CUMHUR'a ve beni bugünlere getiren, emek veren, maddi ve manevi desteklerini esirgemeyen aileme özellikle de çalışmalarımı her daim destekleyen ve katkıda bulunan kardeşim Arş. Gör. Sena ŞAHİN AKTURA'ya en içten sevgi, saygı ve teşekkürlerimi sunarım.

Gamze ŞAHİN

TEZ ETİK BEYANNAMESİ

Tarafımdan hazırlanan " Matlab GUI'de İnteraktif Isı Denklemleri ve Uygulamaları" başlıklı bu tezdeki bütün bilgileri etik davranış ve akademik kurallar çerçevesinde elde ettiğimi, tez yazım kurallarına uygun olarak hazırladığımı ve aksinin ortaya çıkması durumunda her türlü yasal sonucu kabul ettiğimi beyan ederim. 24/07/2019

Gamze ŞAHİN

Uyarı: *Bu tezde kullanılan özgün ve/veya başka kaynaklardan sunulan içeriğin kaynak olarak kullanımı, 5846 sayılı Fikir ve Sanat Eserleri Kanunundaki hükümlere tabidir.*

ÖZET

MATLAB GUI'DE İNTERAKTİF ISI DENKLEMLERİ VE UYGULAMALARI

Gamze ŞAHİN

Recep Tayyip Erdoğan Üniversitesi
Fen Bilimleri Ensititüsü
Matematik Anabilim Dalı
Yüksek Lisans Tezi
Danışmanı: Dr. Öğr. Üyesi İshak CUMHUR

Bu tezde ilk olarak daha sonraki konular için kullanılacak kavramlar verildi. Daha sonra ısı iletim problemi için klasik sonlu fark yöntemleri çalışıldı. Sonlu farklar için yöntem olarak açık-kapalı yöntemler, Crank-Nicolson, ağırlıklı ortalama ve Dufort Frankel yöntemleri üzerinde duruldu. Model problemler değişik sınır şartlarda düşünülmüştür. Nümerik ve analitik çözümler interaktif bir şekilde Matlab GUI'de sunulmuştur.

2019, 70 sayfa

Anahtar Kelimeler: Isı Denklemleri, Açık ve Kapalı Yöntemler, Sonlu Farklar Yöntemi, MatLab, MatLab GUI

ABSTRACT

INTERACTIVE HEAT EQUATIONS AND APPLICATIONS IN MATLAB GUI

Gamze ŞAHİN

Recep Tayyip Erdogan University
Graduate School of Natural and Applied Sciences
Department of Mathematics
Master Thesis
Supervisor: Asst. Prof. Dr. İshak CUMHUR

In this thesis, the basic concepts which will be used in the later chapters are given. Classical finite difference methods(FDMs) are studied for heat conduction problem with various boundary conditions. Explicit, Implicit, Crank-Nicolson, Weighted-average and Dufort Frakel methods are presented in FDMs. Model problems are considered for heat conduction equation with various boundary conditions. The numerical and analytical solutions are presented interactively in Matlab GUI.

2019, 70 pages

Keywords: Heat Equations, Explicit and Implicit Methods, Finite Differences Method, MatLab, MatLab GUI

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SEMBOLLER ve KISALTMALAR DİZİNİ

L	İnce Bir Çubuğun Boyu
T	Zaman Parametresi
$u(x,t)$	t Anında x Konumundaki Sıcaklık
c	Çubuğun Özgül Isısı
ρ	Çubuğun Yoğunluğu
α	Çubuğun Isı Yayılım Katsayısı
$B_1[u]$	Sağ Karışık Şart
$B_2[u]$	Sol Karışık Şart
$P_n(x)$	n. Dereceden Taylor Polinomu
$T_{i,j}$	Lokal Kesme Hatası
Δ	İleri Fark Operatörü
∇	Geri Fark Operatörü
δ	Merkezi Fark Operatörü
Δx	x_i'ler Arasındaki Mesafe (Konum Adım Büyüklüğü)
Δt	Zaman Adım Büyüklüğü
$u(x,t)$	Sürekli Çözüm (Gerçek Çözüm)
$u(x,0)$	Başlangıç Sıcaklık Dağılımı
$u(x_i,t_j)$	Ağ Noktalarında Hesaplanan Gerçek Çözüm Değerleri
$U_{i,j}$	Sonlu Farklarla Elde Edilen Sayısal Çözümler
r	Yakınsama Parametresi
θ	Ağırlıklı Ortalama Yöntemi Parametresi
$\mathcal{O}(\Delta x)$	Δx ile Orantılı Hata Terimi
$\mathcal{O}(\Delta t)$	Δt ile Orantılı Hata Terimi
BTCS	Zamanda Geri Konumda Merkezi Fark
FTCS	Zamanda İleri Konumda Merkezi Fark
KDD	Kısmi Diferansiyel Denklemler
MATLAB	Matrix Laboratory
MatLab GUI	Graphical User Interfaces

1. GENEL BİLGİLER

1.1. Giriş

Isı denklemleri özel tipte bir Kısmi Diferansiyel Denklemler (KDD) olup, bu kısımda öncelikle KDD ile alakalı temel kavramlar verilecektir. KDD uygulamalı matematiğin önemli bir kısmını oluşturmaktadır. Bu tip denklemlerin temel bilimlerde ve mühendisliğin her dalında pek çok uygulaması olmaktadır.

Özellikle nonlineer (Lineer olmayan) kısmi diferansiyel denklemler; akışkanlar mekaniği, akustik, nonlineer optik gibi fiziksel problemlerde, biyolojik ve kimyasal problemlerde, doğanın temel yasalarını formüllemede ve çoğu uygulamalarda ortaya çıkmaktadır. Bu denklemlerden bazılarının isimleri ve ortaya çıktığı bazı alanlar şu şekildedir:

- $(u_x)^2 + \left(u_y\right)^2 = n^2$, *Eikonal denklemi,* Nonlineer optik, Sismoloji
- $u_t - (F(u)u_x)_x = 0$, *Nonlineer ısı denklemi*, Isı akışı, Termal enerjinin yayılımı, Sınır tabaka akımı
- $u_t + uu_x = vu_{xx}$, *Burgers denklemi,* Trafik akışı, Şok dalgaları
- $u_t - u_{xx} = au(1-u)$, *Fisher denklemi,* Nüfus artışı, Kütle transfer
- $\begin{cases} v_x - 2u = 0 \\ v_t - 2u_x + u^2 = 0 \end{cases}$, *Burgers sistemi,* Şok dalgaları, Nonlineer akustik
- $\begin{cases} u_t - v_x = 0 \\ v_t - F\big(u(x,t)\big)u_x - G\big(u(x,t)\big) = 0 \end{cases}$, *Telgraf Sistemi,* Elektrik sinyalleri

Buna göre ilk olarak KDD ile alakalı temel tanımlar özet bir şekilde verilecektir. Tanımlamalar yapıldıktan sonra özel olarak KDD ısı denklemi ele alınacaktır. Bu tezde Isı denklemlerini sayısal çözümleri üzerinde durulacaktır. Bunun için Sonlu Fark Yöntemi ele alınacak ve farklı uygulama alanlarıyla beraber sayısal çözümler elde edilecektir. En çok kullanılan sonlu fark yöntemleri açık (explicit), kapalı (implicit) ve Crank-Nicolson yöntemleri olup bu klasik sonlu fark yöntemlerine ek olarak farklı sayısal çözüm yöntemleri verilecektir. Klasik yöntemde denkleme ek olarak verilen

başlangıç şartları Dirichlet, Neumann ve Robin şartları olup, ele alınacak yeni yöntemler tüm bu yan şartlara göre irdelenecektir.

Bu çalışmanın orijinal kısmını oluşturan son bölümde, ısı denklemlerinin çözümlerini incelemek için MatLab GUI (Graphical User Interfaces)'de bir grafik ara yüzü geliştirilecektir ve bu ara yüzde kullanıcı etkileşimli olarak bu çözümler uygulamalı olarak test edilebilecektir. Geliştirilen ara yüzde herhangi bir kullanıcının, program bilgisine gerek duymadan, sadece parametre değerleri girerek ısı denkleminin çözümlerini test etmesi hedeflenmektedir.

1.2. Tanımlar

Tanım 1. İçerisinde bilinmeyen bir fonksiyonu ve türevlerini barındıran denkleme *diferansiyel denklem* denir. Bir diferansiyel denklem tek bir değişken içeriyorsa denkleme *adi diferansiyel denklem*; iki veya daha fazla bağımsız değişken içeren diferansiyel denkleme *kısmi diferansiyel denklem* denir.

u bağımlı; x ve y bağımsız değişkenler olmak üzere bir kısmi diferansiyel denklem genel olarak $F\left(x, y, u, u_x, u_y, u_{xx}, u_{xy}, u_{yy}, \dots\right) = 0$ şeklindedir. Burada $u_x = \frac{\partial u}{\partial x}$, $u_y = \frac{\partial u}{\partial y}$, $u_{xx} = \frac{\partial^2 u}{\partial x^2}$, $u_{xy} = \frac{\partial^2 u}{\partial x \partial y}$, $u_{yy} = \frac{\partial^2 u}{\partial y^2}$ gösterimleri kullanılmaktadır. n bağımsız ve bir bağımlı değişkene sahip kısmi diferansiyel denklemlerin genel şekli, $x = (x_1, x_2, x_3, \dots, x_n)$, $u = u(x)$ olmak üzere

$$F\left(x_1, x_2, \dots, x_n, u, u_{x_1}, u_{x_2}, \dots, u_{x_n}, u_{x_1 x_1}, u_{x_1 x_2}, \dots\right) = 0 \tag{1}$$

şeklindedir. Burada $x_1, x_2, \dots, x_n$ bağımsız değişkenleri, u ise bağımlı değişkeni göstermekte ve $u_{x_i} = \frac{\partial u}{\partial x_i}$, $u_{x_i y_j} = \frac{\partial^2 u}{\partial x_i \partial y_j}$, $i, j = 1,2, \dots, n$ dir.

Birbirine bağlı birden fazla olayların modellenmesinde tek bir denklem yerine olayların sayısı kadar bağımlı değişken içeren bir diferansiyel denklem sistemi ortaya çıkmaktadır.

$\begin{cases} u\frac{\partial v}{\partial x} + v\frac{\partial u}{\partial y} = x + y \\ u\frac{\partial u}{\partial x} + v\frac{\partial v}{\partial y} = x - y \end{cases}$ ve $\begin{cases} \frac{\partial u}{\partial x} - \frac{\partial v}{\partial y} = 0 \\ \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} = 0 \end{cases}$ sistemleri; u ile v bağımlı, x ile y bağımsız değişkenlerine sahip kısmi diferansiyel denklem sistemleridir.

Tanım 2. Bir KDD'de görülen en yüksek mertebeden kısmi türevin mertebesine denklemin *mertebesi* denir.

Örnek 1. $u_{tt} + au_t = c^2 u_{xx}$, $(a, c\ sabit)$ sönümlü dalga denklemi ikinci mertebeden bir kısmi diferansiyel denklemdir.

Tanım 3. Bir kısmi diferansiyel denklemde bağımlı değişken ve bunların denklemdeki bütün kısmi türevleri birinci dereceden ve denklemi, bağımlı değişken ile onun türevleri parantezinde yazdığımızda katsayılar yalnızca bağımsız değişkenlerin fonksiyonu oluyorsa bu denkleme *lineer*dir, aksi takdirde *lineer olmayan denklem* denir. Ayrıca KDD de bağımlı değişken, üstel, logaritmik veya trigonometrik olarak bulunuyorsa bu tür denklemler de lineer değildir.

Örnek 2. $u_t - k\left(u_{xx} + u_{yy}\right) = 0$, $(k\ sabit)$ iki boyutlu ısı denklemi ikinci mertebeden lineer bir denklemdir. Burada x, y, t bağımsız, u bağımlı değişkendir.

Örnek 3. Aşağıdaki KDD in hiçbiri lineer değildir.

- $uu_{xy} + u_x u_y = 0$
- $u_{xy}u_{yy} + u_{yy} - 4xu_x - xyu = 0$
- $u_{xx} - 3u_{yy} = \sin u$
- $u_x\left(u_{yy}\right)^2 + 2xuu_{xy} - 5xyu_y = 0$

1.3. İkinci Mertebe KDD'lerin Sınıflandırılması

İki bağımsız değişkenli ikinci mertebeden lineer KDD genel olarak, A, B, C, D, E, F, G x ve y bağımsız değişkenlerinin bir fonksiyonu olmak üzere

$$A\frac{\partial^2 u}{\partial x^2} + B\frac{\partial^2 u}{\partial x \partial y} + C\frac{\partial^2 u}{\partial y^2} + D\frac{\partial u}{\partial x} + E\frac{\partial u}{\partial y} + Fu + G = 0 \quad (2)$$

şeklindedir. Denklem (2)

$$Au_{xx} + Bu_{xy} + Cu_{yy} + f(x, y, u_x, u_y, u) = 0 \quad (3)$$

şeklinde de yazılabilir. Şayet KDD sabit katsayılı ise, A, B, C, D, E, F, G katsayıları sabitler olmak üzere

$$Au_{xx} + Bu_{xy} + Cu_{yy} + Du_x + Eu_y + Fu + G = 0 \quad (4)$$

yazılabilir. (4) denkleminin bir sınıfı, yüksek mertebeli terimlerin katsayıları ile belirlenmektedir. Buna göre $B^2 - 4AC$ diskriminantının işaretine göre; $B^2 - 4AC > 0$ ise (4) denklemi hiperbolik, $B^2 - 4AC = 0$ ise denklem parabolik ve $B^2 - 4AC < 0$ ise (4) denklemi eliptik olarak adlandırılır.

Şayet Denklem (4) değişken katsayılı bir KDD ise bu durumda sınıfını tanımlamak istediğimiz bölgedeki her (x, y) noktasında diskriminanta göre değerlendirme yapılmalıdır. Örneğin eliptik denklem için, ilgilenilen bölgedeki her (x, y) değeri için $B^2(x, y) - 4A(x, y)C(x, y) < 0$ sağlanması gerekmektedir. Böylece değişken katsayılı KDD için

- (x, y) noktasında $B^2(x, y) - 4A(x, y)C(x, y) > 0$ ise (4) denklemi (x, y) noktasında hiporbolik
- (x, y) noktasında $B^2(x, y) - 4A(x, y)C(x, y) = 0$ ise (4) denklemi (x, y) noktasında parabolik
- (x, y) noktasında $B^2(x, y) - 4A(x, y)C(x, y) < 0$ ise (4) denklemi (x, y) noktasında eliptik

özellikleri geçerlidir.

Örnek 4. Bazı özel tipte KDD şu şekilde verilmektedir.

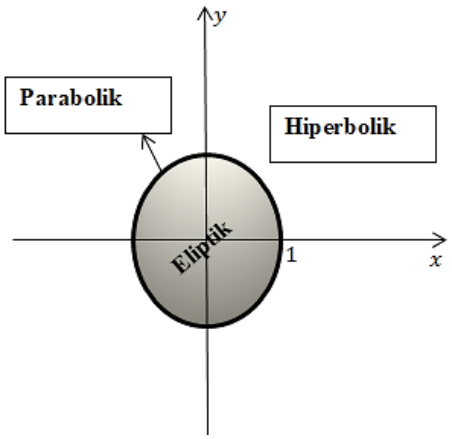
a) $u_{xx} + u_{yy} = 0$ (*Laplace denklemi*). Verilen denklemde $A = 1,\ B = 0,\ C = 1$ ve $B^2 - 4AC = -4 < 0$ olduğundan denklem eliptik tiplidir.

b) $u_t = u_{xx}$ (*Isı denklemi*). Burada $A = -1,\ B = 0,\ C = 0$ ve $B^2 - 4AC = 0$ olduğundan denklem parabolik tiplidir.

c) $u_{tt} - u_{xx} = 0$ (*Dalga denklemi*). Bu denklemde $A = -1,\ B = 0,\ C = 1$ ve $B^2 - 4AC = 4 > 0$ olduğundan denklem hiperbolik tiplidir.

d) $u_{xx} + xu_{yy} = 0,\quad x \neq 0$ (*Tricomi denklemi*). Burada $B^2 - 4AC = -4x$ olduğundan, $x < 0$ için denklem hiperbolik ve $x > 0$ için denklem eliptik tiplidir. Bu örnek gösteriyor ki değişken katsayılı denklemlerin tipi, verilen bölgenin farklı yerlerinde değişebilmektedir.

Örnek 5. $(1 - x^2)u_{xx} - 2xyu_{xy} + (1 - y^2)u_{yy} + xu_x + 3x^2yu_y - 2u = 0$ kısmi diferansiyel denkleminde $B^2 - 4AC = (-xy)^2 - (1 - x^2)(1 - y^2) = -1 + x^2 + y^2$ olduğundan KDD, $x^2 + y^2 > 1$ için hiperbolik, $x^2 + y^2 = 1$ için parabolik ve son durumda $x^2 + y^2 < 1$ için eliptik denklemdir (Şekil 1).



Şekil 1. Farklı bölgelerde KDD için sınıflandırmanın yapılması

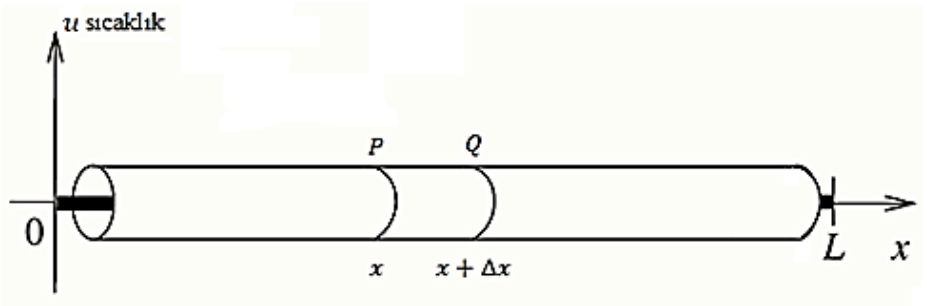
1.4. Isı Denklemi ve Isı Denkleminin Elde Edilişi

Bu kısımda, bu tez boyunca ele alınacak parabolik KDD olan bir boyutlu ısı denklemi ve sınır şartlarından bahsedilecektir. Bu denklemler; ısı iletimi, kimyasal

konsantrasyon gibi difüzyon işlemlerini açıkladığı için daha sık olarak difüzyon denklemi olarak da adlandırılabilmektedir.

Enerjinin korunumu ilkesi ve sıcak bölgeden soğuk bölgeye ısı akışı gerçeğinden yola çıkarak ısı denklemi türetilebilmektedir.

L uzunluğunda ısıyı iletebilen bir çubuk veya tel düşünelim (Şekil 2). Çubuğun $x = 0$ ve $x = L$ uçları haricinde dış yüzeyinin yalıtılmış olduğunu varsayalım. $u(x,t)$; t anında x konumundaki sıcaklığı göstersin. $u(x,t)$ nin verilen bir zamanda çubuğun her kesitinde sabit olduğu düşünülmektedir.



Şekil 2. ***L*** uzunluğunda ince bir çubuk örneği

Enerjinin korunumu ilkesinden, PQ dilimi (x ve $x + \Delta x$ arasında) içerisindeki net ısı değişimi, çubuğun uçları boyunca net ısı akısı ile PQ içerisinde üretilen ısının toplamına eşit olduğu ifade edilebilir. c çubuğun özgül ısısı, ρ çubuğun yoğunluğu, A çubuğun kesit alanı, k çubuğun termal iletkenlik katsayısı ve $f(x,t)$ harici ısı kaynağı olmak üzere bu terimler şu şekilde hesaplanmaktadır:

t anında PQ dilimi içerisindeki toplam ısı miktarı	$= \int_x^{x+\Delta x} c\rho A u(\tau,t)d\tau$
PQ içerisindeki net ısı değişimi	$= \frac{d}{dt}\int_x^{x+\Delta x} c\rho A u(\tau,t)d\tau = c\rho A \int_x^{x+\Delta x} u_t(\tau,t)d\tau.$
Uçlar boyunca net ısı akısı	$= kA[u_x(x+\Delta x,t) - u_x(x,t)].$
PQ içerisinde üretilen harici ısıdan kaynaklı ısı	$= A\int_x^{x+\Delta x} f(\tau,t)d\tau.$

Enerjinin korunumu ilkesinden,

$$\frac{d}{dt}\int_x^{x+\Delta x} c\rho A u(\tau,t)d\tau = c\rho A\int_x^{x+\Delta x} u_t(\tau,t)d\tau \\ = kA[u_x(x+\Delta x,t)-u_x(x,t)] + A\int_x^{x+\Delta x} f(\tau,t)d\tau \quad (5)$$

yazılabilir. İntegraller için ortalama değer teoremi ($\int_a^b f(x)dx = f(\xi)(b-a)$) uygulanırsa, $\xi_1, \xi_2 \epsilon (x, x+\Delta x)$ olmak üzere,

$$c\rho A u_t(\xi_1,t)\Delta x = kA[u_x(x+\Delta x,t)-u_x(x,t)] + Af(\xi_2,t)\Delta x \quad (6)$$

ve böylece,

$$u_t(\xi_1,t) = \frac{k}{c\rho}A\left[\frac{u_x(x+\Delta x,t)-u_x(x,t)}{\Delta x}\right] + \frac{1}{c\rho}f(\xi_2,t) \quad (7)$$

elde edilir. $\Delta x \to 0$ alarak limite geçilirse, $\alpha^2 = k/(c\rho)$ çubuğun ısı yayılım katsayısı ve $F(x,t) = \frac{1}{c\rho}f(x,t)$ ısı kaynağı yoğunluğu olmak üzere,

$$u_t(x,t) = \alpha^2 u_{xx}(x,t) + F(x,t) \quad (8)$$

sonucuna varırız. (8) denklemi ısı denkleminin tek boyutta formülize edilmiş şeklidir.

1.5. Başlangıç ve Sınır Şartları

İnce çubuktaki ısı iletiminde $u(x,t)$ sıcaklık fonksiyonu (8) denklemini sağlamalıdır. Fiziksel açıdan diferansiyel denklem, tek başına çubukta herhangi bir anda sıcaklık dağılımını belirlemeyeceği için çubuğun uç noktalarında sağlaması gereken şartlarla, çubuğun başlangıç sıcaklığı hakkında ek bilgilere ihtiyaç duyulmaktadır. Yani başlangıç anında $(t = 0)$, $u(x,t)$ fonksiyonun ve çubuğun her iki ucuyla, çubuğun bulunduğu ortamdaki ısı enerjisinin nasıl değiştiği belirlenmesi gerekmektedir.

Eğer hiç ısı kaynağı yoksa, $t = 0$ da çubuğun sıcaklık dağılımı $f(x)$ fonksiyonu ile tanımlanıyor ve çubuğun her iki ucundan sabit sıfır sıcaklığı veriliyorsa bu durumda $t > 0$ anlarında çubuktaki $u(x,t)$ sıcaklık dağılımı; $u_t = \alpha^2 u_{xx}(x,t),\ 0 \le x \le L,$ $t > 0$ diferansiyel denkleminin

$$u(x,0) = f(x),\ 0 \le x \le L \tag{9}$$

$$u(0,t) = u(L,t) = 0,\ t > 0 \tag{10}$$

şartlarıyla çözülmesiyle bulunur. Bu problem, ısı denklemi için başlangıç-sınır değer problemi olarak adlandırılır ve (9)-(10) şartlarına ise sırasıyla başlangıç şartı ve sınır şartları denir. Isı iletim probleminde ortaya çıkan sınır şartlar üç tiptir.

Çubuktaki ısı akış probleminde $(0 \le x \le L)$, uçlarda $u(0,t)$ ve $u(L,t)$ sıcaklıklarının belirtilmesi, *Dirichlet tip*li sınır şartına bir örnektir. Dirichlet sınır şartları bir sınır boyunca çözümün değerini verir.

Eğer çubuğun her iki ucu ısı geçişi olmayacak şekilde izole edilirse (8) denklemin sınır şartı

$$u_x(0,t) = u_x(L,t) = 0,\ t > 0 \tag{11}$$

şeklini alır. Bu *Neumann tip*li sınır şartına bir örnektir. Neumann şartı, sınır boyunca çözümün normal türevinin değerini verir.

Uçlardaki şartlar hem Dirichlet tip hem de Neumann tipin bir karışımı şeklinde veriliyorsa sınır şart *Robin tip*li sınır şartı ya da *karışık şart* olarak adlandırılır. Örnek olarak, h bir sabit ve $g(t)$ sınırdaki t'ye bağlı sıcaklığı ifade etmek üzere

$$u_x = -h\big(u - g(t)\big),\ t > 0 \tag{12}$$

verilebilir. Robin şartı, uç boyunca içe doğru olan akının, u sıcaklığı ile belli bir andaki g sıcaklığının arasındaki farkla orantılı olduğunu söyler. uçtaki u sıcaklığı zamana göre verilen sıcaklıktan büyükse ısı akışı dışa doğru, küçükse ısı akışı içe doğrudur. Robin şartı, Dirichlet ve Neumann şartlarının lineer bir kombinasyonundan oluşmaktadır.

Sınırdaki ısı değişimlerinin genel formda tanımlanması şu şekildedir:

$$\begin{aligned} B_1[u] &\equiv a_{11}u(0,t) + a_{12}u_x(0,t) = \alpha(t), \\ B_2[u] &\equiv a_{21}u(L,t) + a_{22}u_x(L,t) = \beta(t),\ t > 0. \end{aligned} \tag{13}$$

Verilen ısı denklemi ilk etapta, genel sınır şartlarından ziyade birkaç özel durumda ele alınacaktır.

1.6. Boyutsuzlaştırma

Isı denkleminde bazı fiziksel sabitlerin bilinmesi gerekmektedir. Örnek olarak k termal iletkenlik katsayısı gösterilebilir. Isı denkleminin çözümünün daha anlamlı ve daha basit olması için bazı fiziksel sabitlerin mümkün olduğu kadar gruplandırılması gerekmektedir. Gruplandırma işlemleri genellikle boyutsuzlaştırma olarak bilinmektedir.

L uzunluğunda yalıtılmış bir çubuk için ısı denklemi $u_t(x,t) = \alpha^2 u_{xx}(x,t)$ olarak elde edilmiş ve başlangıç sıcaklık dağılımı $u(x,0) = f(x)$ ile verilmiştir. $\kappa = \alpha^2$ olmak üzere κ'nın birimi alan/zaman şeklinde ifade edilir. t ve x bağımsız değişkenleri sırasıyla zaman ve uzunluk birimlerine sahiptir. Yeni konum ve zaman değişkenleri ve diğer parametreler

$$\tilde{x} = \frac{x}{L},\ \tilde{t} = \frac{\kappa t}{L^2},\ \tilde{u} = \frac{u}{u_*},\ \tilde{f} = \frac{f}{u_*} \tag{14}$$

ile verilir. Bu durumda x'in birimi ortadan kalkar ve $\tilde{x}$ yalnızca bir sayı olur. Bu durum boyutsuzlaştırma olarak bilinir. Benzer şekilde $\tilde{t}$ boyutsuz bir parametredir. Türevler için zincir kuralı kullanıldığında

$$u_t = \tilde{u}_{\tilde{t}} \frac{u_* \kappa}{L^2}, u_x = \tilde{u}_{\tilde{x}} \frac{u_*}{L}, u_{xx} = \tilde{u}_{\tilde{x}\tilde{x}} \frac{u_*}{L^2}. \quad (15)$$

ifadeleri ısı denkleminde yerine yazılırsa

$$\tilde{u}_{\tilde{t}} \frac{u_* \kappa}{L^2} = \kappa \tilde{u}_{\tilde{x}\tilde{x}} \frac{u_*}{L^2}, \tilde{u}_{\tilde{t}} = \tilde{u}_{\tilde{x}\tilde{x}} \quad (16)$$

elde edilir. Başlangıç sıcaklık dağılımı $\tilde{u}(\tilde{x}, 0) = \tilde{f}(\tilde{x})$ şeklinde boyutsuz hale getirildikten sonra fiziksel sabitlerle artık ilgilenilmemektedir. Sabitler artık sadece sayı olarak girilmektedir. Boyutsuzlaştırmadan sonra çubuğun uzunluğunun da 1 olduğu düşünülmektedir. Buna göre harflerin üzerlerindeki şapkalar atılırsa,

$$\begin{aligned} &u_t = u_{xx}, 0 < x < 1 \\ &u(x, 0) = f(x), 0 < x < 1 \end{aligned} \quad (17)$$

boyutsuz problem elde edilir. buna göre x ve t, boyutsuz konum ve zaman ölçekleridir. Benzer şekilde boyutsuzlaştırma işlemi, ısı denkleminin sınır şartlarına da uygulanabilmektedir.

1.7. Taylor Serisi ve Sonlu Fark Yaklaşımlarına Giriş

Bu bölümde sonlu fark yaklaşımları kullanılarak ısı denklemlerinin nümerik çözümleri hakkında bilgi verilecektir. Zamanda ileri konumda merkezi fark (FTCS), zamanda geri konumda merkezi fark (BTCS) ve Crank-Nicolson yöntemleri geliştirilerek tek boyutta ısı denklemine uygulanacaktır. Bu tezin amacı bu teknikler verildikten sonra daha değişik yöntemleri geliştirerek farklı başlangıç şartlarla beraber ele almaktır. Yöntemler verildikten sonra MatLab GUI de yöntemlerin hepsini ele aldığımız bir kullanıcı arayüzü geliştirilecektir. Bu arayüzde kullanıcı (öğrenciler) hiçbir program bilgisine gerek kalmadan sadece belirli parametreler girerek ısı denklemlerinin çözümlerini irdeleyebileceklerdir. Bu kullanıcıya farklı parametrelerle çözümleri karşılaştırma olanağı sunmaktadır. Geliştirilen bu arayüzle kullanıcı FTCS, BTCS, Crank-Nicolson yöntemi ve diğer yöntemlerin farklılıklarını test edebilecektir. Ayrıca yöntemlerin kararlı olduğu limit değerleri arayüzde farklı deneylerle test

edebilecektir. Buna göre kararlılık, tutarlılık, yakınsaklık vs. gibi kavramların açıklanması gerekmektedir.

Tanım 1.1. f fonksiyonu a noktasını içeren bir aralıkta her mertebeden türevlenebilir olmak üzere,

$$f(x) = \sum_{k=0}^{\infty} \frac{f^{(k)}(a)}{k!}(x-a)^k$$

serisine a noktasında f fonksiyonun Taylor serisi denir. Taylor serisinde sonlu sayıda terim alarak oluşan,

$$P_n(x) = \sum_{k=0}^{n} \frac{f^{(k)}(a)}{k!}(x-a)^k$$

polinomuna n. dereceden Taylor polinomu denir. Bu polinom açık bir şekilde,

$$P_n(x) = f(x_0) + f'(x_0)(x-x_0) + f''(x_0)\frac{(x-x_0)^2}{2!} + \cdots + f^{(n)}(x_0)\frac{(x-x_0)^n}{n!} + \cdots$$

şeklinde yazılabilir. $f(x)$ fonksiyonu için kalan terim veya hata terimi, $\xi(x)$, x ile x_0 arasında olmak üzere,

$$f(x) - P_n(x) = R_n(x) = \frac{f^{(n+1)}(\xi(x))}{(n+1)!}(x-x_0)^{n+1}$$

şeklinde verilmektedir.

Tanım 1.2. (Lokal Kesme Hatası) (i,j) bileşenine karşılık gelen düğüm noktasında bir kısmi diferansiyel denklemin çözümünün yaklaşık sonuçlarını veren fark denklemi, U fark denkleminin tam çözümü olmak üzere, $F_{i,j}(U) = 0$ ile gösterilirse $(i,j) - inci$ düğüm noktasında sonlu fark yaklaşımının lokal kesme hatası, u kısmi diferansiyel denklemin tam çözümü olmak üzere, $T_{i,j} = F_{i,j}(U)$ olarak tanımlanır.

Lokal kesme hatası, sonlu fark yaklaşımının kısmi diferansiyel denkleme ne kadar iyi yaklaştığını açıklayan bir kavramdır. Taylor seri açılımının kullanılmasıyla lokal kesme hatası, h ve k değerlerinin kuvvetleri ve (ih, jk) noktasında kısmi diferansiyel denklemin tam çözümünün (u) kısmi türevleri cinsinden kolayca açıklanabilmektedir.

Tanım 1.3. (Tutarlılık) $h, k \to 0$ olduğunda lokal kesme hatasının limit değeri sıfıra yaklaşıyorsa fark denklemi tutarlıdır denir. Yani bu durum $\lim_{h,k\to 0} T_{i,j} = 0$ ile ifade edilmektedir.

Tanım 1.4. (Kararlılık) Kısmi diferansiyel denkleme karşılık gelen sonlu fark denkleminin çözümünün kısmi diferansiyel denklemin çözümüne yakınsadığı duruma kararlıdır denir.

Tanım 1.5. (Yakınsaklık) U fark denkleminin ve u ise kısmi diferansiyel denklemin tam çözümleri olmak üzere $\lim_{h,k\to 0} U_{i,j} = u_{i,j}$ ise sonlu fark denkleminin çözümü kısmi diferansiyel denklemin çözümüne yakınsar denir.

Bu tanımlara göre bir fark denkleminin hangi şartlar altında kararlı olduğunun araştırılmasına kararlılık analizi denir.

Teorem 1.1. (Lax Denklik Teoremi) Sonlu fark yönteminin yakınsak olması için gerek ve yeter şart yöntemin tutarlı ve kararlı olmasıdır.

Sonlu fark yöntemi KDD in sayısal çözümün elde etmede kullanılan birkaç yöntemden biridir. Sayısal çözümü elde ederken sürekli KDD in yerine ayrık bir yaklaşım yazılmaktadır. Bu tezde ayrık kelimesi yerine fiziksel bir bölgede sonlu sayıda noktanın sayısal çözümlerinden bahsedilmektedir. Bu noktaların sayısı dışarıdan keyfi olarak seçilebilir. Noktaların sayısındaki artış, çözümün kararlılığını arttırdığı gibi sayısal çözümün doğruluğuna da katkısı olmaktadır.

Ayrıklaştırma cebirsel denklemlerin bir kümesini ortaya çıkarır ve bilinmeyen değerler bu cebirsel denklemlerin çözümü ile elde edilirler. Şekil (3). nümerik çözümlerin temsili bir gösterimidir.

$u(x,t)$ için sürekli KDD	$\xrightarrow{Sonlu\ farklar}$	Ayrık fark denklemi	$\xrightarrow{Çözüm\ yöntemi}$	$U_{i,j}$ yaklaşık çözümleri

Şekil 3. Sürekli ve ayrık problemler arasındaki ilişki

Ağ, ayrık çözümlerin hesaplandığı yerlerin kümesidir. Bu noktalar düğüm olarak adlandırılır Ağ için Δx, konuma göre komşu noktalar arasındaki mesafe ve Δt, komşu zaman adımları arasındaki mesafedir. Bu tezde Δx ve Δt'in ağ boyunca düzgün olduğu düşünülecektir.

Verilen bir diferansiyel denkleme sonlu fark yöntemi uygulandığında, denklem içerisindeki tüm türevler fark formülleriyle yer değiştirir. Isı denkleminde zamana göre ve konuma göre türevler vardır. Fark formülleriyle ağ noktalarının farklı kombinasyonları kullanıldığında sonlu farklar için, farklı yöntemler ortaya çıkar. Limit durumunda Δx ve Δt sıfıra yakın seçilirse, elde edilen yaklaşık çözümün gerçek çözüme yakın sonuçlar verdiği bir yöntem ortaya çıkar. Bununla beraber Δx ve Δt'nin kötü seçimleri çözümü hesaplamada hataya yol açabilir.

1.8. Ağ Noktaları

Sonlu fark yöntemi, sonlu sayıda x ve t değerleri için $U(x,t)$ yaklaşık çözümlerini elde eder. Ayrık x değerleri $0 \leq x \leq L$ aralığında düzgün konumlandırılmıştır. Buna göre N, konumu belirleyen ve sınırları da içeren düğümlerin sayısı olmak üzere

$$x_i = (i-1)\Delta x,\ i = 1,2,\dots,N$$

yazılabilir. L ve N değerleri verildiğinde x_i ler arasındaki mesafe $\Delta x = \frac{L}{N-1}$ ile hesaplanır. Benzer şekilde kesikli t değerleri $0 \leq t \leq t_{max}$ aralığında düzgün bir

şekilde yerleştirilir. M zamana göre adımların sayısı ve Δt zaman adım büyüklüğü olmak üzere $\Delta t = \frac{t_{max}}{M-1}$ olmak üzere

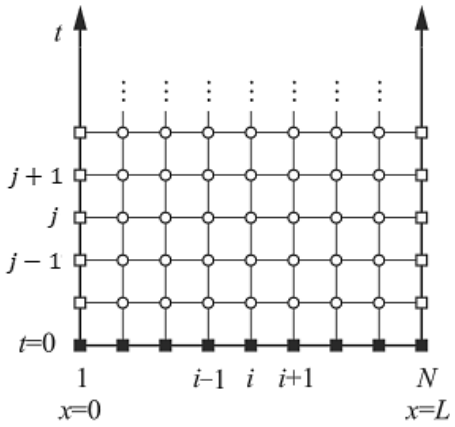
$t_j = (j-1)\Delta t, j = 1,2,\dots,M$

şeklinde hesaplanmalıdır. Isı denkleminin yaklaşık çözümünü elde etmek ve sonuçları analiz etmek için kullanılan gösterimler Tablo (1)' de verilmiştir.

Tablo 1. Kullanılan gösterimler

Semboller	Açıklamalar
$u(x,t)$	Sürekli çözümler(Gerçek çözüm)
$u(x_i,t_j)$	Ağ noktalarında hesaplanan sürekli çözüm
$U_{i,j}$	Sonlu fark denklemlerinin çözümünden elde edilen yaklaşık sayısal çözümler

Sonlu farklar yönteminin uygulanacağı çözüm bölgesi Şekil (4)'de gösterilmiştir. Şekil (4)'de görüldüğü gibi verilen başlangıç ve sınır şartları altında ağ iç düğüm noktalarında sayısal çözüm elde edilecektir.



Şekil 4. Tek boyutlu ısı denkleminin çözümü için kullanılan ağ

1.9. Sonlu Fark Yaklaşımları

Sonlu fark yöntemi, $\frac{\partial u}{\partial x} \approx \frac{U_{i+1}-U_i}{\Delta x}$ ifadesinde olduğu gibi ayrık yaklaşımların kullanılmasına dayanır. Sağ taraftaki nicelikler ağ noktalarını tanımlar. Diferansiyel

denklemdeki tüm sürekli türevler yerine bu şekildeki sonlu fark formülleri yazılarak yaklaşımlar elde edilmektedir. Sonlu fark modelden hesaplanan $U_{i,j}$ çözümleri, sürekli problemi ayrık probleme dönüştürerek elde edilir.

Sonlu fark yöntemi ilk olarak x bağımsız değişkenine bağlı $U = U(x)$ şeklinde yazılan U bağımlı değişkeni ile geliştirilir. Formüllerde hem konum değişkeni olan x'e hem de zaman değişkeni olan t'ye göre yaklaşık türevler kullanılır.

1.9.1. Birinci Mertebe İleri Fark

$u(x)$ fonksiyonun x_i noktası civarında Taylor seri açılımı, $\delta(x)$, x_i'ye göre x'deki bir değişim olmak üzere,

$$u(x_i + \delta x) = u(x_i) + \delta x \frac{\partial u}{\partial x}(x_i) + \frac{\delta x^2}{2}\frac{\partial^2 u}{\partial x^2}(x_i) + \frac{\delta x^3}{3!}\frac{\partial^3 u}{\partial x^3}(x_i) + \cdots \quad (18)$$

şeklinde düşünülmektedir. Denklem (18)'de $\delta x = \Delta x$ alınırsa, yani u'nun değeri x_{i+1} konumunda düşünülürse,

$$u(x_i + \Delta x) = u(x_i) + \Delta x \frac{\partial u}{\partial x}(x_i) + \frac{\Delta x^2}{2}\frac{\partial^2 u}{\partial x^2}(x_i) + \frac{\Delta x^3}{3!}\frac{\partial^3 u}{\partial x^3}(x_i) + \cdots \quad (19)$$

elde edilir. Denklem (19), $\frac{\partial u}{\partial x}(x_i)$ için çözülürse,

$$\frac{\partial u}{\partial x}(x_i) = \frac{u(x_i+\Delta x)-u(x_i)}{\Delta x} - \frac{\Delta x}{2}\frac{\partial^2 u}{\partial x^2}(x_i) - \frac{\Delta x^2}{3!}\frac{\partial^3 u}{\partial x^3}(x_i) + \cdots \quad (20)$$

elde edilir. Tam çözümle yaklaşık çözüm, $U_i \approx u(x_i)$ ve $U_{i+1} \approx u(x_{i+1})$ şeklinde yer değiştirirse,

$$\frac{\partial u}{\partial x}(x_i) = \frac{U_{i+1}-U_i}{\Delta x} - \frac{\Delta x}{2}\frac{\partial^2 u}{\partial x^2}(x_i) - \frac{\Delta x^2}{3!}\frac{\partial^3 u}{\partial x^3}(x_i) + \cdots \quad (21)$$

elde edilir. $x_i \leq \xi \leq x_{i+1}$ olmak üzere ortalama değer teoremine göre,

$$\frac{\Delta x^2}{2}\frac{\partial^2 u}{\partial x^2}(x_i) + \frac{\Delta x^3}{3!}\frac{\partial^3 u}{\partial x^3}(x_i) + \cdots = \frac{\Delta x^2}{2}\frac{\partial^2 u}{\partial x^2}(\xi)$$

elde edilir. Böylece,

$$\frac{\partial u}{\partial x}(x_i) \approx \frac{U_{i+1}-U_i}{\Delta x} + \frac{\Delta x^2}{2}\frac{\partial^2 u}{\partial x^2}(\xi) \text{ veya } \frac{\partial u}{\partial x}(x_i) - \frac{U_{i+1}-U_i}{\Delta x} \approx \frac{\Delta x^2}{2}\frac{\partial^2 u}{\partial x^2}(\xi) \quad (22)$$

yazılabilir.

Denklem (22)'nin sağ tarafındaki terim sonlu fark yaklaşımının kesme hatası olarak adlandırılır. Genellikle ξ bilinmeyendir. Ayrıca, $u(x,t)$ de bilinmeyen olduğundan, $\partial^2 u/\partial x^2$ hesaplanamaz. Kesme hatasının büyüklüğü bilinmemesine rağmen, "büyük $\mathcal{O}$" gösterimi kesme hatasının düğümler arası mesafeye bağlı olduğunu ifade etmek için kullanılır. Denklem (22)'nin sağ tarafı Δx parametresini içermektedir ve dışarıdan keyfi olarak belirlenmektedir. Hatayı belirleyen tek parametre bu olduğu için kesme hatası basit bir şekilde

$$\frac{\Delta x^2}{2}\frac{\partial^2 u}{\partial x^2}(\xi) = \mathcal{O}(\Delta x^2)$$

şeklinde yazılır. Buradaki eşitlik tam eşitlik manasında değildir ve büyüklük anlamında doğru olduğunu göstermektedir. Ayrıca sol taraf bilinmeyen bir sabit ile Δx^2'nin bir çarpımdır. Son denklemin sol tarafı bize tam olarak büyüklüğü vermemesine rağmen, Δx değeri azaltıldığı zaman bu terimin sıfıra yaklaştığı ile alakalı pratik bilgi verir.

$\mathcal{O}$ gösterimini kullanırsak Denklem (21),

$$\frac{\partial u}{\partial x}(x_i) \approx \frac{U_{i+1}-U_i}{\Delta x} + \mathcal{O}(\Delta x) \quad (23)$$

şeklinde yazılabilir. Denklem (23), x_i ve x_{i+1} düğümlerini içerdiğinden $(\partial u/\partial x)_{x_i}$ için ileri fark formülü olarak adlandırılır. İleri fark yaklaşımının kesme hatası $\mathcal{O}(\Delta x)$'dir. Ağ

boyutunu Δx belirlediğinden dolayı kesme hatasının büyüklüğü bu şekilde kontrol altına alınmaktadır. O halde bizim kontrol altına alamadığımız kesme hatası $|\partial u/\partial x\,|_{\xi}$'dir.

1.9.2. Birinci Mertebe Geri Fark

Alternatif birinci mertebe sonlu fark formülü, Denklem (18)'de Taylor serisinde $\delta x = -\Delta x$ alarak elde edilebilir. Tüm bilinmeyenlerin yerine ayrık ağ parametreleri kullanılırsa Denklem (24) elde edilir.

$$U_{i-1} = U_i - \Delta x \frac{\partial u}{\partial x}(x_i) + \frac{(\Delta x)^2}{2}\frac{\partial^2 u}{\partial x^2}(x_i) - \frac{(\Delta x)^3}{3!}\frac{\partial^3 u}{\partial x^3}(x_i) + \cdots \tag{24}$$

Denklem (24)'de sağ taraftaki terimlerin işaretlerinin değiştiğine dikkat edilmelidir. $(\partial u/\partial x)_{x_i}$ için son ifade çözülürse,

$$\frac{\partial u}{\partial x}(x_i) = \frac{U_i - U_{i-1}}{\Delta x} + \frac{\Delta x}{2}\frac{\partial^2 u}{\partial x^2}(x_i) - \frac{(\Delta x)^2}{3!}\frac{\partial^3 u}{\partial x^3}(x_i) + \cdots \tag{25}$$

veya $\mathcal{O}$ gösterimi kullanılırsa,

$$\frac{\partial u}{\partial x}(x_i) \approx \frac{U_i - U_{i-1}}{\Delta x} + \mathcal{O}(\Delta x) \tag{26}$$

elde edilir. x_i ve x_{i-1} noktalarında U'nun değerlerini içerdiği için Denklem (26)'e geri fark formülü denir. Kesme hatasının büyüklüğü ileri fark yaklaşımında olduğu gibidir. Yani geri fark yaklaşımının kesme hatası $\mathcal{O}(\Delta x)$'dir.

1.9.3. Birinci Mertebe Merkezi Fark

U_{i+1} ve U_{i-1} için Taylor seri açılımları yazılırsa,

$$U_{i+1} = U_i + \Delta x \frac{\partial u}{\partial x}(x_i) + \frac{(\Delta x)^2}{2}\frac{\partial^2 u}{\partial x^2}(x_i) + \frac{(\Delta x)^3}{3!}\frac{\partial^3 u}{\partial x^3}(x_i) + \cdots \tag{27}$$

$$U_{i-1} = U_i - \Delta x \frac{\partial u}{\partial x}(x_i) + \frac{(\Delta x)^2}{2}\frac{\partial^2 u}{\partial x^2}(x_i) - \frac{(\Delta x)^3}{3!}\frac{\partial^3 u}{\partial x^3}(x_i) + \cdots \tag{28}$$

elde edilir. Denklem (28)'den Denklem (27) çıkarılırsa,

$$U_{i+1} - U_{i-1} = 2\Delta x \frac{\partial u}{\partial x}(x_i) + \frac{2(\Delta x)^3}{3!}\frac{\partial^3 u}{\partial x^3}(x_i) + \cdots$$

bulunur ve $(\partial u/\partial x)_{x_i}$ için son ifade çözülürse,

$$\frac{\partial u}{\partial x}(x_i) = \frac{U_{i+1} - U_{i-1}}{2\Delta x} - \frac{(\Delta x)^2}{3!}\frac{\partial^3 u}{\partial x^3}(x_i) + \cdots$$

veya

$$\frac{\partial u}{\partial x}(x_i) = \frac{U_{i+1} - U_{i-1}}{2\Delta x} + \mathcal{O}(\Delta x^2) \tag{29}$$

elde edilir. Bu yaklaşıma $(\partial u/\partial x)_{x_i}$ için merkezi fark formülü denir. Sürekli probleme iyi bir yaklaşım için küçük Δx seçilir. $\Delta x \ll 1$ olduğunda, merkezi fark için kesme hatası Denklem (23) veya Denklem (26)'daki kesme hatalarına göre daha hızlı sıfıra yaklaşır.

Buradaki tek sorun Denklem (29)'un U_i'yi içermiyor oluşudur. Diferansiyel denkleme yaklaşımda merkezi fark kullanıldığında problemler ortaya çıkabilmektedir.

1.9.4. İkinci Mertebe Merkezi Fark

Yüksek mertebeden türevler için sonlu fark yaklaşımları, Taylor seri açılımında yapılacak olan ek düzenleme ile elde edilebilir. Denklem (27) ve Denklem (28) toplanırsa,

$$U_{i+1} + U_i = 2U_i + (\Delta x)^2 \frac{\partial^2 u}{\partial x^2}(x_i) + \frac{2(\Delta x)^4}{4!}\frac{\partial^4 u}{\partial x^4}(x_i) + \cdots$$

ve bu ifadeden $(\partial^2 u/\partial x^2)_{x_i}$ çözülürse,

$$\frac{\partial^2 u}{\partial x^2}(x_i) = \frac{U_{i+1}-2U_i+U_{i-1}}{\Delta x^2} + \frac{(\Delta x)^2}{12}\frac{\partial^4 u}{\partial x^4}(x_i) + \cdots$$

veya mertebe gösterimi kullanılırsa,

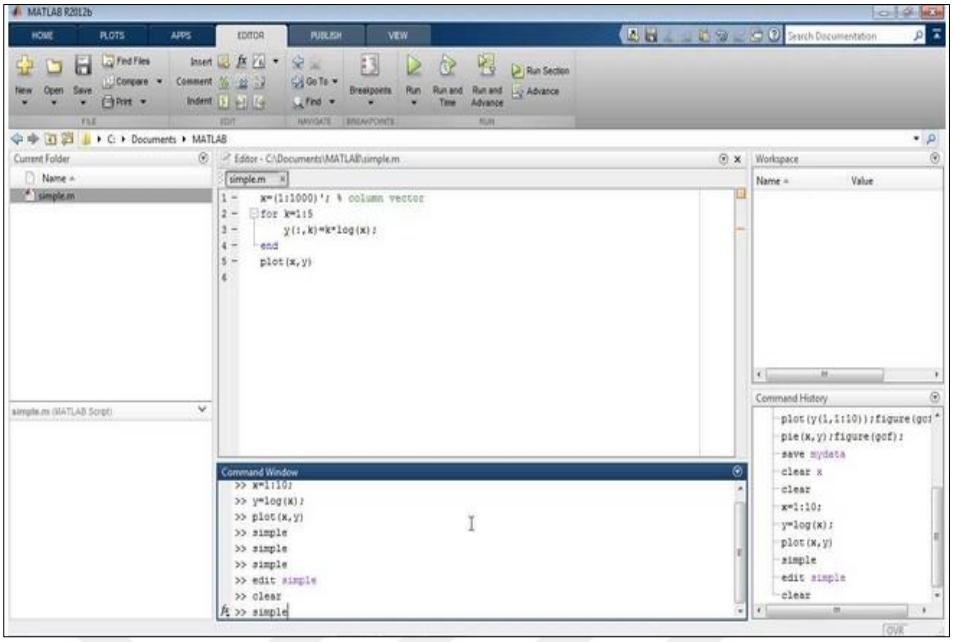
$$\frac{\partial^2 u}{\partial x^2}(x_i) = \frac{U_{i+1}-2U_i+U_{i-1}}{\Delta x^2} + \mathcal{O}(\Delta x^2) \quad (30)$$

elde edilir. Buna ikinci türevler için merkezi fark formülü denir. Denklem (29) birinci türevler için yaklaşımı ve Denklem (30) ikinci türevler için yaklaşımı ifade etmektedir.

1.10. Matlab

MATLAB, genellikle pozitif bilim ve mühendislik hesaplamaları için kullanılan bir bilgisayar programıdır. Amerika Birleşik Devletleri merkezli MathWorks firması tarafından geliştirilen MATLAB, aynı zamanda bir programlama dilidir. İngilizce "Matrix Laboratory" kelimelerinin birleştirilmesi ile oluşmuş olan MATLAB, isminden de anlaşılacağı gibi matris tabanlı bir çalışma sistemine sahiptir.

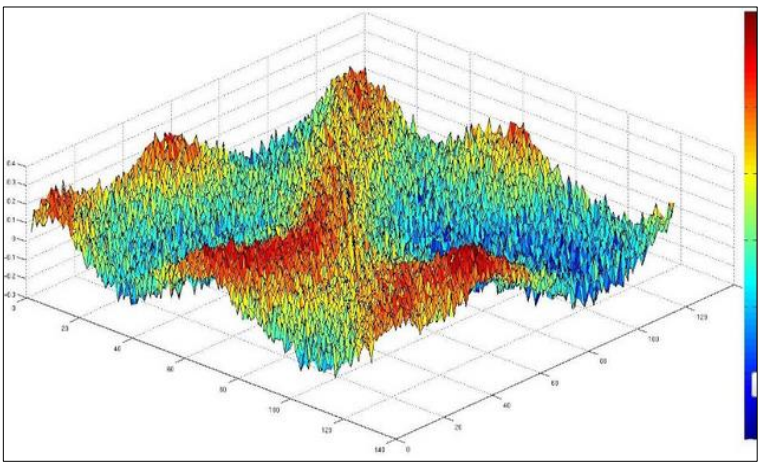
Lineer cebir, istatistik, optimizasyon, nümerik analiz, optimizasyon, Fourier analizi gibi pek çok matematiksel hesaplamanın etkili ve hızlı şekilde yapılmasına olanak sağlayan MATLAB programı aynı zamanda 2D ve 3D grafik çizimi için de kullanılır.



Şekil 5. Matlab arayüzü ve araçları

MATLAB ile kullanıcılar kendi programlarını hazırlayabilirler. Matrisler ve onların etkileşim içinde olduğu fonksiyonlarla programlama yapılmasına izin veren MATLAB ile çok karmaşık matematik hesaplamaları bile birkaç saniye içinde tamamlanır. Temel Programlama fonksiyonları ile benzer fonksiyonların kullanılabildiği MATLAB ile etkili ve pratik programlar hazırlanabilir.

C, Java gibi programlama dillerindeki dizilerin kullanımı ile aynı mantıkla matrislerin kullanıldığı MATLAB programında bir, iki veya daha fazla boyutta matrisler ile çalışmak mümkündür.

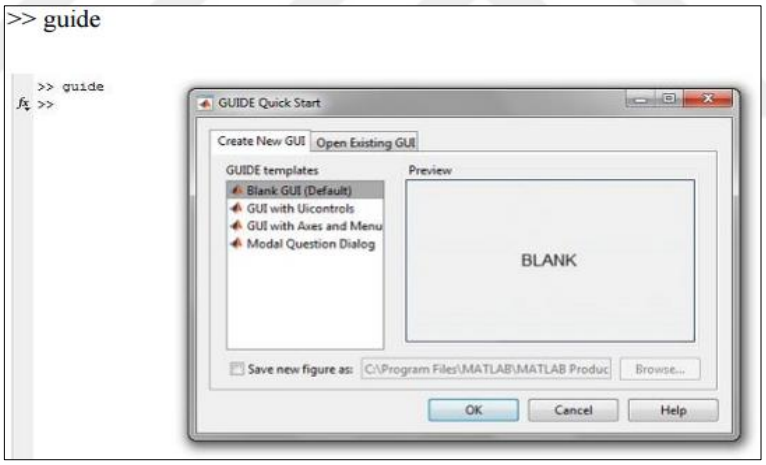


Şekil 6. Matlab 3D görseli

MATLAB ile temel matematik fonksiyonlarının iki ve üç boyutlu grafikleri çizilebilir. Polinomlar, paraboller, sinüs dalgaları başta olmak üzere her tür iki ve üç boyutlu matematiksel grafik MATLAB ile elde edilebilir (URL-2).

1.11. Grafiksel Kullanıcı Arayüzü (Matlab GUI)

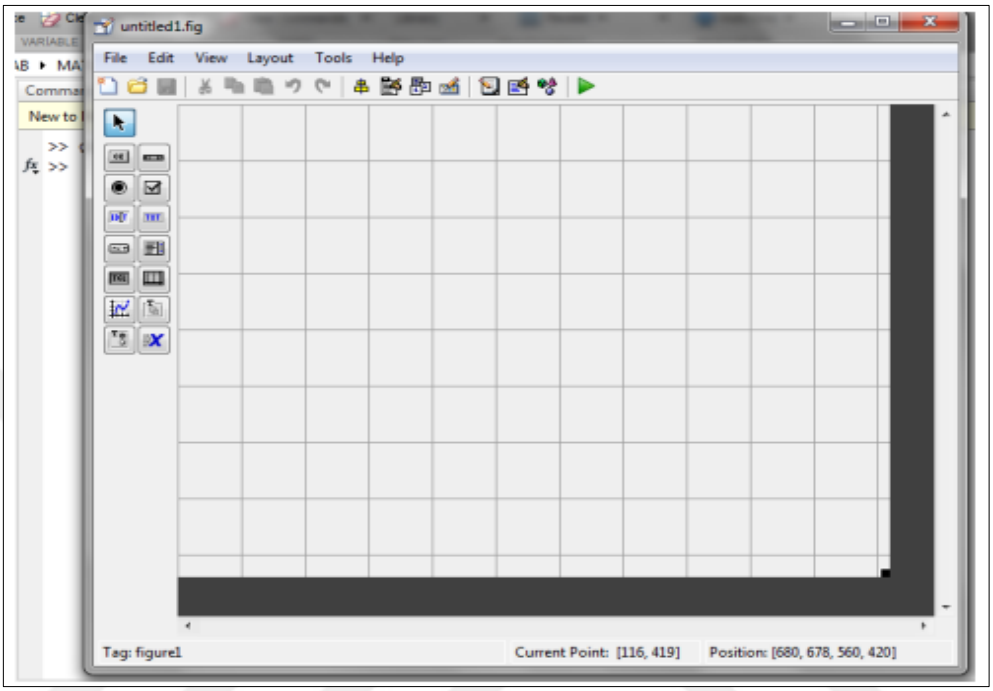
GUIDE MATLAB'ın GUI tasarımcılarına sunduğu içerisinde çeşitli araçlar içeren ve kolaylık sağlayan bir grafiksel GUI geliştirme ortamıdır. GUIDE kullanılarak tıkla ve sürükle-bırak tekniği ile GUI arayüzüne nesneler (örneğin butonlar, text kutuları, liste kutuları, grafikler vs.) kolaylıkla eklenebilir. Ayrıca, eklenen nesnelerin hizalanması, tab sırasının değiştirilmesi, görsel ayarlar üzerinde manipülasyonlar yapılması da bu ortamın tasarımcılara sunduğu imkânlardan bazılarıdır. Bu aracı çalıştırmak için ya MATLAB komut satırından GUIDE komutu verilir. Bu adımdan sonra karşımıza Şekil (7)'deki gibi bir pencere gelir.



Şekil 7. Matlab GUI çalışma ekranı açma işlemi

Bu pencereden yeni bir GUI tasarımı yapılacaksa Blank GUI seçeneği seçilmelidir. Şayet önceden yapılmış bir tasarım açmak istenirse Open Existing GUI sekmesinden sonra istenilen dosya seçilebilir. Burada yeni bir tasarım oluşturulacağı için Blank GUI seçeneği seçilerek OK düğmesi tıklanılır. Böylece Şekil (7)'deki GUIDE LAYOUT Editor (GUIDE Çalışma Alanı) penceresine ulaşır.

Bu adımdan sonra File/Prefences/Guide yolu kullanılarak gelen pencerede “Show names in component palette” seçeneği tıklanarak OK düğmesine basılır ise Şekil (8)’teki gibi bir pencere ekrana gelecektir.



Şekil 8. Matlab GUI çalışma alanı

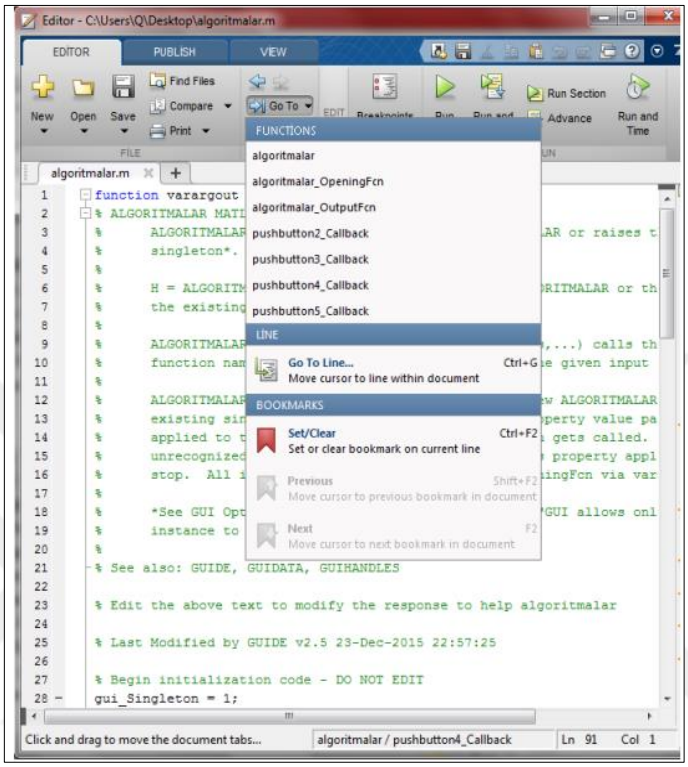
1.11.1. GUI Tasarımını Kaydetme ve Çalıştırma

Bitmiş olan bir GUI arayüzünü çalıştırarak görmek için öncelikle Tools/Run yolundan Run (Çalıştır) komutu verilir. Burada çalışmanın Run edilebilmesi için kaydedilmesi gerektiğini bildiren bir pencere çıkar. Yes butonuna tıklayarak çalışmanın kaydedilmesi sağlanır. Bu adımdan sonra MATLAB GUIDE tasarımın kaydedileceği dosya ismini soran bir pencere getirir. Bu pencereden çalışmamıza bir isim vererek tasarım kaydedilebilir.

1.11.2. GUI Arayüzünün Programlanması

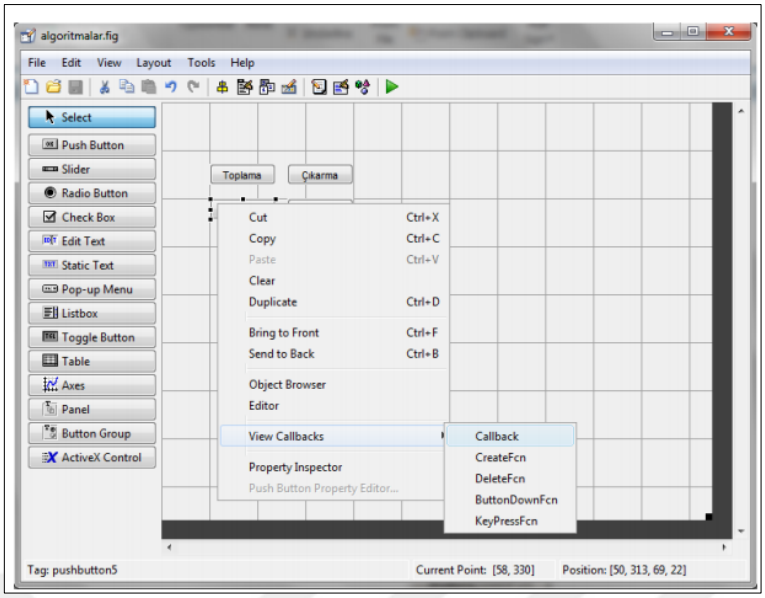
Bir GUI arayüzünün programlanması demek o çalışmanın kaydedildiği isimle aynı zamanla oluşturulan .m uzantılı dosya içerisine kodlama satırlarının eklenmesi demektir. Bu dosyanın içini görebilmek, değişiklik yapabilmek için GUIDE çalışma

ekranı penceresinden View/Editor komutu işletilebilir veya herhangi bir nesnenin üzerine gelerek farenin sağ tuşu ile tıklanarak Editor seçeneği seçilebilir. Karşımıza Şekil (9)'deki gibi bir pencere gelecektir.



Şekil 9. Matlab GUI editör sayfası

Şekil (9)'daki pencerede hazırlanan GUI tasarımına ait kodlar gözükmektedir. Burada pek çok kodun hazır eklenmiş olduğu görülecektir. Bu kodlar otomatik olarak MatLab GUIDE tarafından eklenmiştir. Burada ilgili butonlara ve liste kutularına ya da istenilen bir nesneye ait callback isimli alt program parçalarına ilgili kodlar yazılmalıdır. Bir nesneye ait callback in bulunduğu satıra gitmek için araç çubuğunda yer alan "Go To" butonuna tıklanır ve açılan listeden ilgili nesneye ait callback'in ismi seçilir. Bu durum Şekil (9)'da görülmektedir. Ayrıca, GUIDE çalışma ekranından da direk istenilen bir callback satırına gidilebilir. Bunun için ilgili nesne üzerinde sağ tıklanır ve açılan pencereden View Callbacks/Callback menüsünden ilgili callback tıklanması ya da ilgili nesne seçilip View/View Callbacks/Callback yolu üzerinden gidilebilir (Şekil 10) (URL-3).

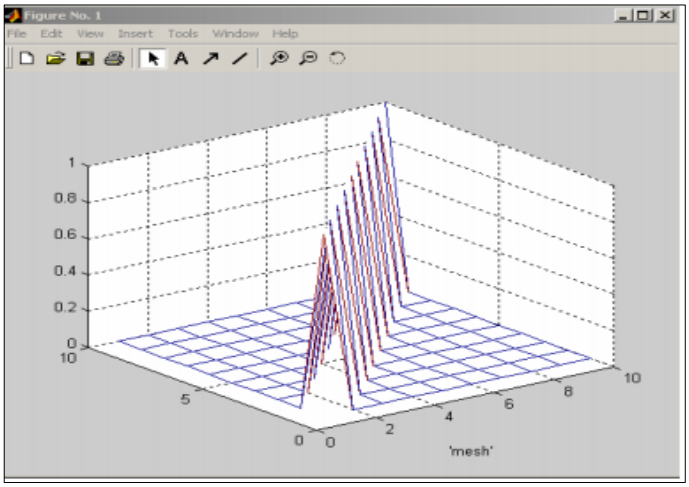


Şekil 10. Matlab GUI callback seçeneğinin işlevi

1.11.3. Grafik ve Eğri Çizimleri

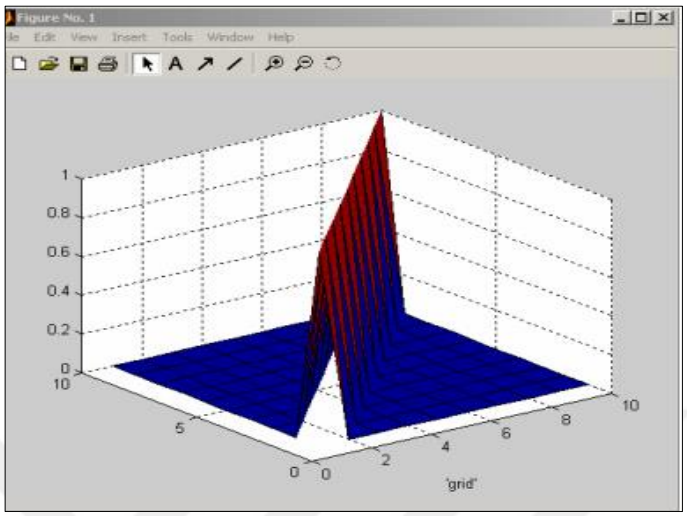
3 boyutlu ağ ve yüzey çizimlerinde kullanılan komutlardan biri mesh(...) komutudur. Bu komut verilen girişi z bileşeni olarak algılar ve dikdörtgen x-y düzlemi üzerinde z ekseni boyunca çizim yapar. surf(...) komutu ise aynı işi yüzey olarak yapar. Aşağıdaki komut satırlarının çizim görüntüleri Şekil (11-12)'de verilmiştir (URL-4).

- mesh(eye(10));
- grid



Şekil 11. 3D Mesh grafiği

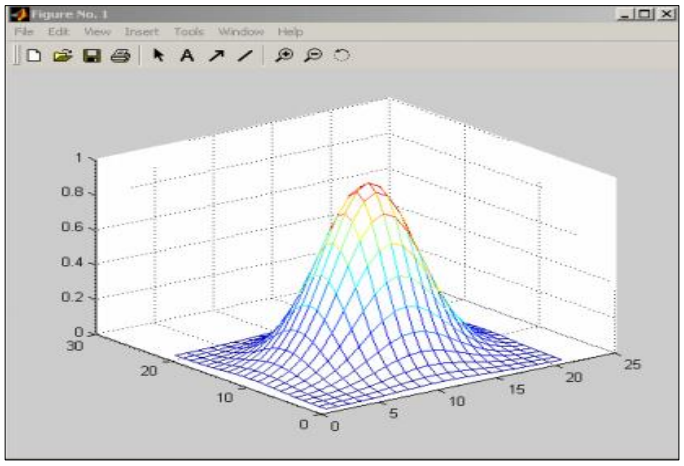
- surf(eye(10));
- grid



Şekil 12. 3D Surf grafiği

$z = e^{-x^2-y^2}$ fonksiyon yüzeyini $[-2,2] \times [-2,2]$ tanım aralığında 3 boyutlu olarak çizdirilebilir (Şekil 13). Bunun için aşağıdaki komutlar girilmelidir.

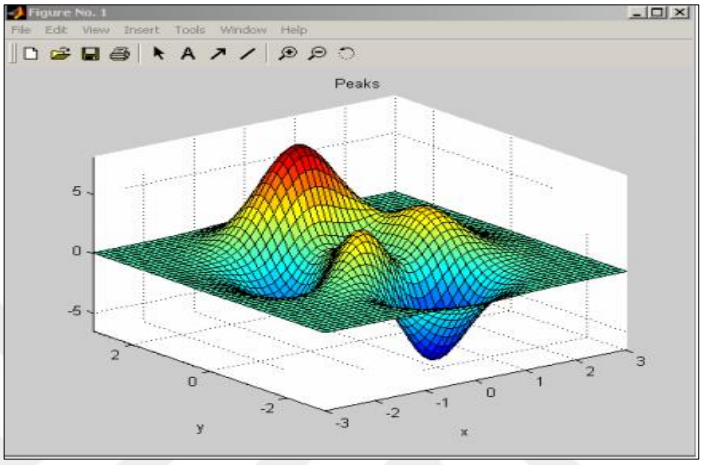
- x=-2:0.2:2;
- y=x;
- [x,y]=meshgrid(x,y);
- z=exp(-x.^2-y.^2);
- mesh(z)



Şekil 13. $z = e^{-x^2-y^2}$ fonksiyonunun Mesh grafiği

Herhangi bir yüzey grafiğinde tepe ve alt tepe (minimum ve maximum) değerlerini göstererek yapılan çizimlerde peaks(...) komutu kullanılır (Şekil 14).

- [X,Y]=meshgrid(-3:0.125:3);
- peaks(X,Y)



Şekil 14. Yüzey minimum ve maksimum gösterimleri

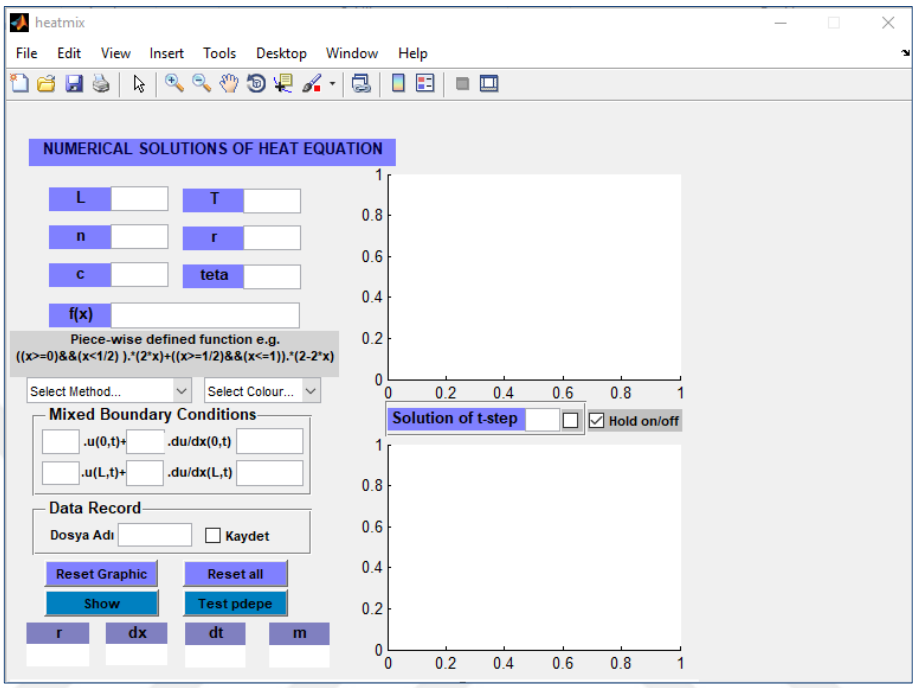
2. YAPILAN ÇALIŞMALAR

Uygulamalı bilim, fizik ve mühendislikteki birçok problem matematiksel olarak kısmi diferansiyel denklemlerle ifade edilip modellenebilir. Bir bağımsız değişkenden daha fazlasını içeren diferansiyel denklem kısmi diferansiyel denklem olarak bilinir. Kısmi diferansiyel denklemleri eliptik, parabolik ve hiperbolik olmak üzere üç sınıfta incelemek mümkündür. Hiperbolik kısmi diferansiyel denklemlere örnek olarak dalga denklemlerini, parabolik kısmi diferansiyel denklemlere örnek olarak ısı denklemlerini ve eliptik kısmi diferansiyel denklemlere örnek olarak da Laplace denklemlerini gösterebiliriz. Bu tip kısmi diferansiyel denklemlerin çözümü analitik ve sayısal olarak iki grupta incelenebilir. Sayısal çözüm için genelde sonlu farklar veya sonlu elemanlar yöntemi kullanılır. Isı denklemlerinin sonlu farklar yöntemi altında sayısal çözümünü irdelemek mümkündür.

Bilgisayarın çıkışı, programlama dillerinin gelişimi ile birlikte incelediğimiz bu ısı denklemlerinin değişik sonlu fark yöntemleri altında sayısal çözümleri için algoritmalar geliştirilmeye ve çeşitli programlama dilleri ile programlar yazılmaya başlanmıştır. Günümüzde birçok matematiksel problemin çözümü için MATLAB, MAPLE, MATHEMATICA gibi paket yazılımlar çıkmış ve kullanılmaya başlanmıştır. Diğer yandan araştırmacıların ve öğrencilerin matematiksel uygulamaları içeren algoritma veya program yapabilmeleri için temel düzeyde bilgisayar bilgisi ve bir programlama dilini öğrenerek algoritma geliştirmesi ya da bu paket yazılımlardan birini öğrenmesi gerekmektedir.

Bu çalışmamızda ısı denklemlerinin sayısal çözümünü, sonlu farklar yöntemi kullanılarak Explicit, Implicit, Crank-Nicolson, Ağırlıklı ortalama vs. gibi yöntemlerle ilgili uygulamalar hazırlan arayüzden kullanıcının hiçbir program bilgisine gerek kalmadan kolaylıkla irdeleyebilmesini hedefledik. Bunun için sistem arayüzünü hazırlamak için MATLAB GUI kullanıcı arayüzünden yararlanılmıştır

Bu tezde boyutsuz bir şekilde verilen $u_t = u_{xx}$, $u(x,0) = f(x)$ şeklindeki ısı denkleminin karışık sınır şartlarında sayısal çözümleri için Matlab GUI'de kullanıcı etkileşimli bir arayüz tasarlanmıştır(Şekil 15).



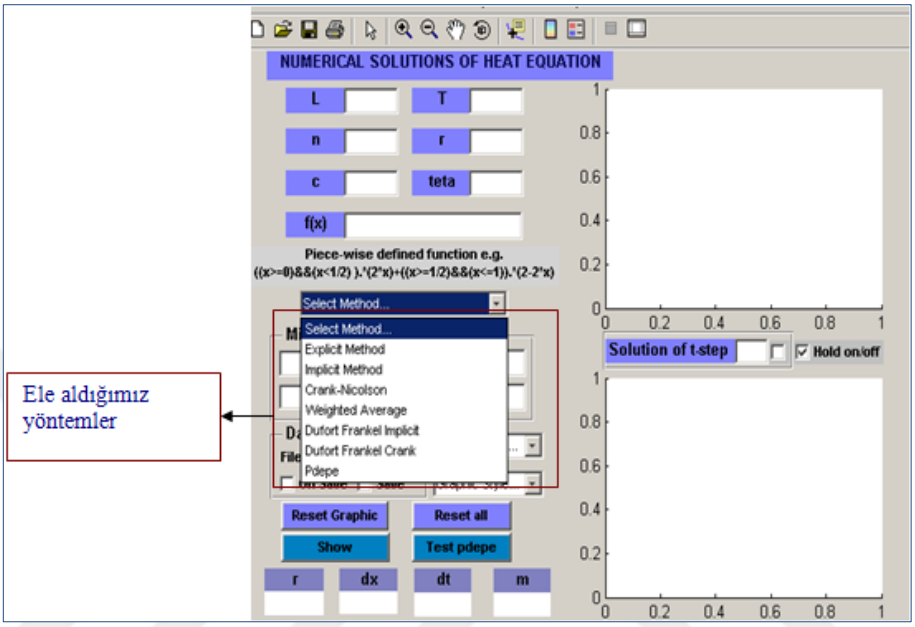
Şekil 15. Isı denklemlerinin sayısal çözümleri için geliştirilmiş arayüz

Tasarlanan arayüzde farklı yöntemler tek bir ekranda ele alınmıştır. En genel sınır şartları olarak verilen

$$B_1[u] \equiv a_{11}u(0,t) + a_{12}u_x(0,t) = \alpha(t),$$
$$B_2[u] \equiv a_{21}u(L,t) + a_{22}u_x(L,t) = \beta(t),\ t > 0.$$

şeklindeki sınır şartlar altında çözümler, veri olarak alınmasının yanında grafikleri ise 2 ve 3 boyutta olmak üzere ayrı ayrı oluşturulmuştur. Tasarlanan arayüzün bölümlerini Şekil (17)'deki numaralara göre açıklayabiliriz. 1 numaraları kısımlara ısı denkleminin parametreleri girilmelidir. Burada L tek boyutlu çubuğun uzunluğu, T zaman parametresi, n değeri $x-$yönünde adım uzunluğu, r yakınsaklık parametresi, c ısı iletim katsayısı, θ ağırlıklı ortalama yöntemi için girilecek parametre, $f(x)$ ise ısı denkleminin başlangıç sıcaklık dağılımı için girilecek fonksiyondur. Ayrıca parçalı sürekli fonksiyon için ise fonksiyon örneği hemen alt kısmında mevcuttur. 2 numaralı

kısımda ısı denkleminin sayısal çözümü için yöntemler seçilebilir. Bu yöntemler açık ve kapalı, Cranck-Nicolson, ağırlıklı ortalama ve Dufort Frankel yöntemleridir. Uygun bir yöntem ve yöntemi kararlı yapacak bir r değeri belirleyerek sayısal çözüm incelenebilir. İncelenen yöntemler için açılır menü Şekil (16)'da olduğu gibidir.

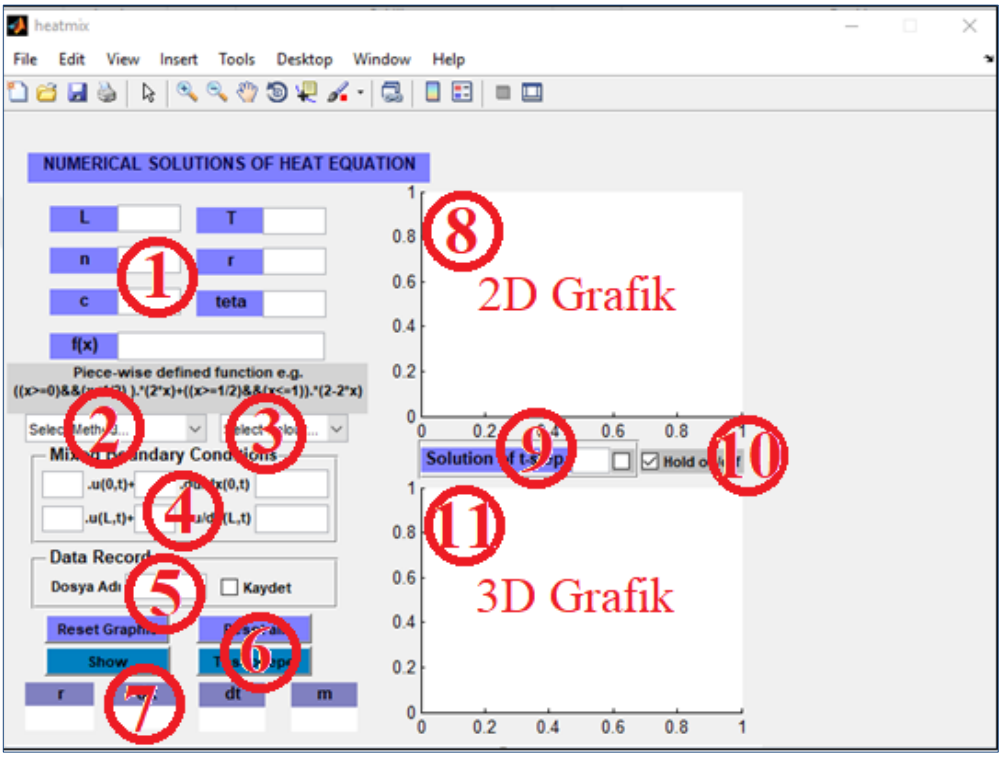


Şekil 16. Sonlu farklar için ele alınan yöntemler

3 Numaralı kısımda 2-boyutlu grafiğin rengi değiştirilebilir. Bu sayede farlı parametrelerle çözüm yapıldığında yöntemlerin karşılaştırması sağlanır. 4 numaralı kısımda ısı denklemi için karışık sınır şartları girilebilir. Kutucuklara yazılacak uygun değerlerle Dirichlet, Neumann veya Robin şartları ile çözümün sayısal incelenmesi sağlanır. 5 numaralı kısımda sayısal verilerin excel dosyasına aktarılması sağlanır. Bu tezde sayısal veriler hem grafik olarak hem de veri olarak elde edilmektedir. Elde edilen 2D ve 3D grafikler GUI arayüzünde ve veriler ise hazırlanan GUI arayüzünün bulunduğu grafikten alınabilir.

6 numaralı kısımda grafik sıfırlama, tüm verileri sıfırlama ve girilen parametrelere sonucu gösterme düğmeleri mevcuttur. Ayrıca MATLAB'in kendi komutu olan pdepe ile çözümün doğruluğu test edilebilmektedir. Bu bize denge konumundaki çözüm ile sayısal çözüm tutarlığı hakkında bilgi verir. 7 numaralı kısımda girilen r parametresi, $x-$yönündeki adım uzunluğu, zamansal adım uzunluğu ve zamansal yönde seçilen

çözümler olan m değerini eş zamanlı ekrana yansımaktadır. 8 numaralı kısımda iki boyutlu grafik seçilen adım uzunluğuna göre oluşmaktadır. 9 numaralı kısımda her adımda çözmek yerine sadece seçilen belli adım değerlerinde çözüm oluşturulur. 10 numaralı kısımda ekran üzerindeki grafik üzerine başka bir grafik eklenip eklenmeyeceğine karar verilir. 11 numaraları kısımda verilerin 3 boyutlu grafikleri elde edilir ve ekrana oluşturulur.



Şekil 17. GUI arayüzünün ayrıntılı açıklaması

Hazırlanan GUI arayüzünün adı Heatmix olup MATLAB komut satırında heatmix komutu grafik arayüzü ekranı açılır. İstenen parametre değerleri girildiğinde grafik ve veriler söylenen doğrultuda oluşturulabilir. Kullanıcı hiçbir program bilgisine gerek duymadan sadece parametre değerleri girerek çözümlerin irdelemesini bu sayede çok rahat bir şekilde yapabilmektedir.

Bir sonraki bölümde farklı yöntemler için hazırlanan arayüzden elde edilecek sonuçlar incelenecektir. Her bir yöntem farklı bir başlıkta incelenecektir. Yöntemler için grafikler ve sayısal sonuçlar irdelenecektir.

3. BULGULAR

3.1. Isı Denklemi için Sayısal Yöntemler

Önceki bölümlerde geliştirilen sonlu fark yaklaşımları ısı denkleminde ayrık yaklaşım şeklinde uygulanacaktır. Hem zaman hem de konum türevlerinin yerini sonlu fark yaklaşımları almaktadır. Bunun için U'nun zaman ve konum yerlerinin belirlenmesi gerekir. Bunun için ayrık çözümlerin zaman adımlarını isimlendirmek için j indisini kullanmaya gerek duyulmaktadır. Konum türevlerinin hesaplandığı yerde zaman adımlarının uygun seçilişi sonlu fark modellemede büyük önem taşımaktadır.

3.2. Zamanda İleri Konumda Merkezi Fark (FTCS)

Denklem (8)'de zaman türevi için ileri fark yaklaşımı

$$\frac{\partial u}{\partial t}(x_i, t_{j+1}) = \frac{U_{i,j+1} - U_{i,j}}{\Delta t} + \mathcal{O}(\Delta t) \tag{31}$$

şeklinde verilir. Denklemin sağ tarafındaki terimler sadece $x = x_i$'deki U değerini içermektedir. $(\partial^2 u/\partial x^2)_{x_i}$'e merkezi fark yaklaşımı uygulanır ve j. zaman adımındaki bütün terimler buradan hesaplanabilmektedir.

$$\frac{\partial^2 u}{\partial x^2}(x_i) = \frac{U_{i-1,j} - 2U_{i,j} + U_{i+1,j}}{\Delta x^2} + \mathcal{O}(\Delta x^2) \tag{32}$$

Isı denkleminde Denklem (31) ve Denklem (32) yazılırsa hata terimleriyle beraber aşağıdaki ifade elde edilmektedir.

$$\frac{U_{i,j+1} - U_{i,j}}{\Delta t} = \alpha \frac{U_{i-1,j} - 2U_{i,j} + U_{i+1,j}}{\Delta x^2} + \mathcal{O}(\Delta t) \; + \mathcal{O}(\Delta x^2) \tag{33}$$

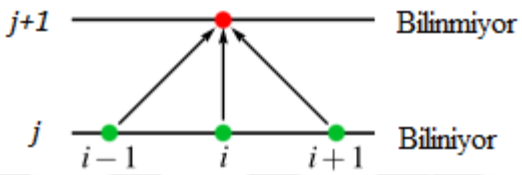
Zamana ait ve konumsal hatalar farklı mertebelere sahiptirler. Ayrıca U'nun diğer değerleri için $U_{i,j+1}$ açık bir şekilde elde edilebilmektedir. Kesme hatası terimleri atılırsa ve $U_{i,j+1}$ için denklem çözülürse,

$$U_{i,j+1} = U_{i,j} + \frac{\alpha\Delta t}{\Delta x^2}(U_{i+1,j} - 2U_{i,j} + U_{i-1,j}) \quad (34)$$

elde edilir. (34) denklemi zamanda ileri, konumda merkezi fark veya ısı denklemine FTCS yaklaşımı olarak adlandırılır. Denklemde yeniden düzenleme yapılırsa hesaplama verimliliğinde etkili bir iyileştirme sağlanabilir. Buna göre $r = \alpha\Delta t/\Delta x^2$ olmak üzere

$$U_{i,j+1} = rU_{i+1,j} + (1 - 2r)U_{i,j} + rU_{i-1,j} \quad (35)$$

şeklinde düzenlenebilir. Bu yöntem için ağ yapısı Şekil (18)'de olduğu gibidir.



Şekil 18. FTCS taslağı için ağ noktaları gösterimi

$U_{i,j+1}$ değerleri birbirlerinden bağımsız olarak hesaplanabildiği için FTCS taslağını uygulamak kolaydır. Tüm çözüm iki döngü içerir. Bunlar bütün zaman adımlarını kapsayan bir dış döngü ve bütün iç ağ noktaları kapsayan iç döngü şeklindedir. Tablo (2)'de kod parçası, FTCS taslağını uygulamanın ne kadar kolay olduğunu göstermektedir.

Denklem (9,10,11,12)'deki başlangıç ve sınır koşullarına göre Denklem (8)'in çözümlerin hepsi sınırlı, sönümlü fonksiyonlardır. Bir başka deyişle, çözümün büyüklüğü başlangıç koşulundan itibaren bir sabite doğru gerileyecektir. FTCS, Δt değeri çok büyükse, kararsız ve gittikçe büyüyen çözümler verebilir. FTCS taslağıyla kararlı çözümler aşağıdaki şart geçerli olduğunda elde edilebilir:

$$r = \frac{\alpha\Delta t}{\Delta x^2} < \frac{1}{2}. \quad (36)$$

Tablo 2. Isı denkleminin çözümü için FTCS taslağı kod parçacığı

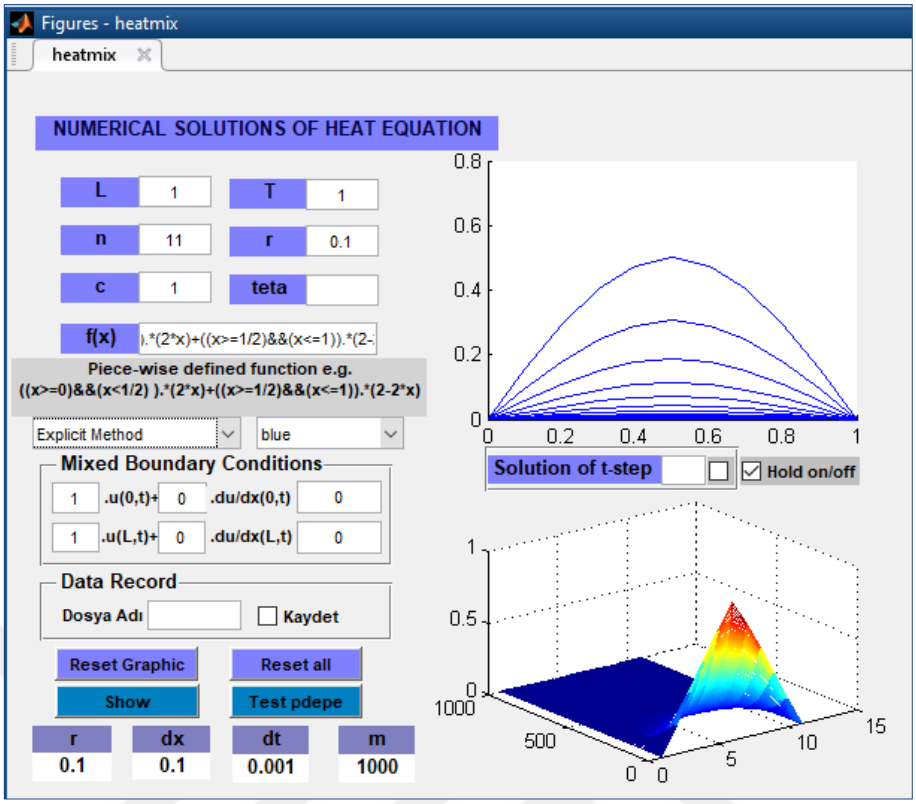
```
% --- Define constants and initial condition
L = ...             %  length of domain in x direction
tmax = ...          %  end time
nx = ...            %  number of nodes in x direction
nt = ...            %  number of time steps
dx = L/(nx-1);
dt = tmax/(nt-1);
r = alpha*dt/dx^2;    r2 = 1 - 2*r;

% --- Loop over time steps
t = 0
u = ...             %   initial condition
for m=1:nt
  uold = u;         %  prepare for next step
  t = t + dt;
  for i=2:nx-1
    u(i) = r*uold(i-1) + r2*uold(i) + r*uold(i+1);
  end
end
```

Denklem (35), bir matris çarpımı olarak $U^{(j+1)} = AU^{(j)}$ şeklinde de ifade edilebilmektedir. Burada A tridiagonal matristir.

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ r & (1-2r) & r & 0 & 0 & 0 \\ 0 & r & (1-2r) & r & 0 & 0 \\ 0 & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & r & (1-2r) & r \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$U^{(j+1)}$ değeri, $j+1$ zaman adımında U değerlerinin vektörüdür ve $U^{(j)}$ ise j zaman adımındaki U değerlerinin vektörüdür. A'nın ilk ve son satırları, matris-vektör çarğımı hesaplandığında U'nun sınır değerleri değişmeyecek şekilde ayarlanırlar.

Örnek olarak $u_t = u_{xx}$, $0 < x < 1$, $t > 0$ ısı probleminin sayısal çözümünü sonlu farklarla $u(x,0) = \begin{cases} 2x, & 0 \le x \le \frac{1}{2} \\ 2(1-x), & \frac{1}{2} \le x \le 1 \end{cases}$ başlangıç koşuluna ve $u(0,t) = u(1,t) = 0$ sınır koşuluna göre GUI arayüzünde inceleyelim. Buna göre çözümler Şekil (19)'da verilmiştir.



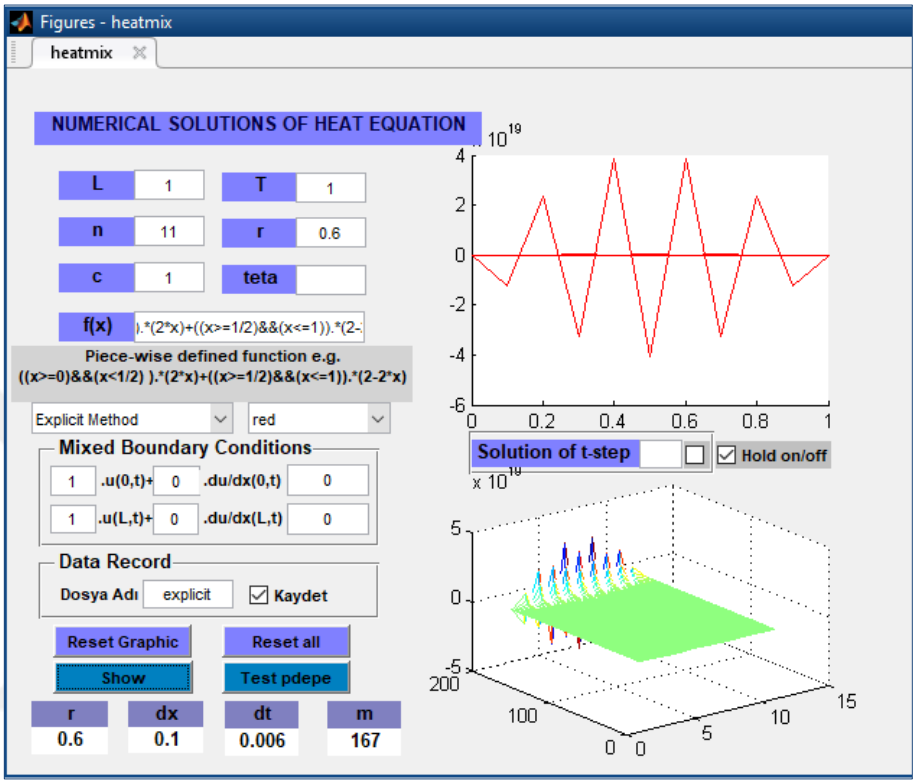
Şekil 19. Isı denkleminin $\boldsymbol{u(x,0)} = \begin{cases} \boldsymbol{2x}, & \boldsymbol{0 \le x \le \frac{1}{2}} \\ \boldsymbol{2(1-x)}, & \boldsymbol{\frac{1}{2} \le x \le 1} \end{cases}$ başlangıç koşuluna ve $\boldsymbol{u(0,t) = u(1,t) = 0}$ koşuluna göre Explicit yöntem ile sayısal çözümleri

Şayet Explicit isminde bir dosya açıp kaydet kutucuğu işaretlenirse aşağıdaki sayısal veriler arka plandan kolayca alınabilir. Buna göre sayısal sonuçlar Tablo (3)'de olduğu gibidir.

Tablo 3. Isı denklemi Explicit yöntem için sayısal veriler tablo değerleri

		0.0000	0.2000	0.4000	0.6000	0.8000	0.9600	0.8000	0.6000	0.4000	0.2000	0.0000
		0.0000	0.2000	0.4000	0.6000	0.7960	0.9280	0.7960	0.6000	0.4000	0.2000	0.0000
		0.0000	0.2000	0.4000	0.5996	0.7896	0.9016	0.7896	0.5996	0.4000	0.2000	0.0000
		0.0000	0.2000	0.4000	0.5986	0.7818	0.8792	0.7818	0.5986	0.4000	0.2000	0.0000
		0.0000	0.2000	0.3998	0.5971	0.7732	0.8597	0.7732	0.5971	0.3998	0.2000	0.0000
		0.0000	0.2000	0.3996	0.5950	0.7643	0.8424	0.7643	0.5950	0.3996	0.2000	0.0000
		0.0000	0.1999	0.3992	0.5924	0.7551	0.8268	0.7551	0.5924	0.3992	0.1999	0.0000
		0.0000	0.1999	0.3986	0.5893	0.7460	0.8125	0.7460	0.5893	0.3986	0.1999	0.0000
		0.0000	0.1998	0.3978	0.5859	0.7370	0.7992	0.7370	0.5859	0.3978	0.1998	0.0000
		0.0000	0.1996	0.3968	0.5822	0.7281	0.7867	0.7281	0.5822	0.3968	0.1996	0.0000
		0.0000	0.1993	0.3956	0.5783	0.7194	0.7750	0.7194	0.5783	0.3956	0.1993	0.0000
		0.0000	0.1990	0.3942	0.5741	0.7108	0.7639	0.7108	0.5741	0.3942	0.1990	0.0000
		0.0000	0.1986	0.3927	0.5698	0.7025	0.7533	0.7025	0.5698	0.3927	0.1986	0.0000
		0.0000	0.1982	0.3910	0.5653	0.6943	0.7431	0.6943	0.5653	0.3910	0.1982	0.0000
		0.0000	0.1977	0.3892	0.5608	0.6863	0.7333	0.6863	0.5608	0.3892	0.1977	0.0000
		0.0000	0.1970	0.3872	0.5562	0.6784	0.7239	0.6784	0.5562	0.3872	0.1970	0.0000
		0.0000	0.1963	0.3851	0.5515	0.6708	0.7148	0.6708	0.5515	0.3851	0.1963	0.0000
		0.0000	0.1956	0.3828	0.5468	0.6632	0.7060	0.6632	0.5468	0.3828	0.1956	0.0000
		0.0000	0.1948	0.3805	0.5420	0.6559	0.6975	0.6559	0.5420	0.3805	0.1948	0.0000
		0.0000	0.1939	0.3781	0.5373	0.6487	0.6891	0.6487	0.5373	0.3781	0.1939	0.0000
		0.0000	0.1929	0.3756	0.5325	0.6416	0.6810	0.6416	0.5325	0.3756	0.1929	0.0000
		0.0000	0.1919	0.3730	0.5277	0.6346	0.6731	0.6346	0.5277	0.3730	0.1919	0.0000
		0.0000	0.1908	0.3704	0.5229	0.6278	0.6654	0.6278	0.5229	0.3704	0.1908	0.0000
		0.0000	0.1897	0.3677	0.5182	0.6211	0.6579	0.6211	0.5182	0.3677	0.1897	0.0000

Bu yöntem $r \leq 0.5$ için kararlıdır. Aksi durumda yöntem belli bir adımdan sonra ıraksar. Bunu sistem arayüzünden herhangi bir örnekle test edebiliriz. Şekil (20)'den ıraksamanın olduğu açık bir şekilde görülmektedir.



Şekil 20. Explicit yönteminde $\boldsymbol{r \leq \frac{1}{2}}$ şartının sağlanmama durumunda ıraksaması

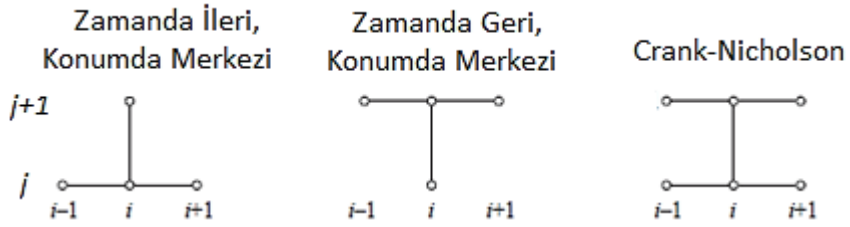
Yukarıda incelediğimiz yöntem Explicit yöntemidir. Fakat bu yöntemin sadece $r \leq 0.5$ için kararlı olması bizi başka yöntemleri araştırmaya teşvik etmektedir. Örnek olarak diğer incelediğimiz yöntemlerden biri olan İmplicit yöntemi tüm r değerleri için kararlıdır. Bu çalışmada ele aldığımız yöntemler, sonlu fark denklemleri ve kararlılık şartları ile birlikte aşağıdaki gibi verilebilir.

3.3. Zamanda Geri Konumda Merkezi Fark (BTCS)

Daha önce denkleminin sol tarafındaki zaman türevi için ileri yönde fark formülü kullanılmıştır. Bu kısımda geri yönde farkı seçilecektir. Zaman türevi için geri yönde fark formülü

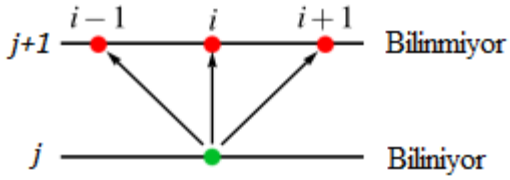
$$\frac{\partial u}{\partial t}(x_i, t_{j+1}) = \frac{U_{i,j} - U_{i,j-1}}{\Delta t} + \mathcal{O}(\Delta t) \quad (37)$$

şeklindedir.



Şekil 21. Isı denklemine sonlu fark yaklaştırmaları için hesaplamalı moleküller

Denklem (26) ve Denklem (30)'u Denklem (8)'de yazıp, kesme hatası terimlerini toplayarak aşağıda denklem elde edilmektedir. Bu yöntem için ağ yapısı Şekil (22)'de olduğu gibidir.



Şekil 22. BTCS taslağı için ağ noktaları gösterimi

$$\frac{U_{i,j} - U_{i,j-1}}{\Delta t} = \alpha \frac{U_{i-1,j} - 2U_{i,j} + U_{i+1,j}}{\Delta x^2} + \mathcal{O}(\Delta t) \; + \mathcal{O}(\Delta x^2). \quad (38)$$

Bu yaklaşımdaki kesme hataları, Denklem (33)'deki kesme hatalarıyla aynı büyüklük mertebesine sahiptirler. Fakat Denklem 38, $U_{i,j}$ yi hesaplamak için basit cebirsel ifade oluşturmaya uygun değildir. $U_{i+1,j}$ için bir benzer denklem geliştirildiğinde $U_{i+1,j}$ değerinin $U_{i+2,j}$ ile $U_{i,j}$'ye bağlı olduğu görülmektedir. Bu nedenle, Denklem (38), $i = 2, 3, \ldots, N-1$ konum ağ noktalarının iç noktalarında U'nun değerleri için bir denklem sistemidir.

Denklem sistemini daha açık bir şekilde görmek için, kesme hatası terimlerini Denklem (38) çıkarıp elde edilen denklemi yeniden düzenleyerek aşağıda belirtilen denklem elde edilmektedir.

$$-\frac{\alpha}{\Delta x^2}U_{i-1,j}+\left(\frac{1}{\Delta t}+\frac{2\alpha}{\Delta x^2}\right)U_{i,j}-\frac{\alpha}{\Delta x^2}U_{i+1,j}=\frac{1}{\Delta t}U_{i,j-1}. \quad (39)$$

Denklem sistemi aşağıda gösterildiği gibi matris formunda temsil edilebilir.

$$\begin{bmatrix} b_1 & c_1 & 0 & 0 & 0 & 0 \\ a_2 & b_2 & c_2 & 0 & 0 & 0 \\ 0 & a_3 & b_3 & c_3 & 0 & 0 \\ 0 & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & a_{N-1} & b_{N-1} & c_{N-1} \\ 0 & 0 & 0 & 0 & a_N & b_N \end{bmatrix}\begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ \vdots \\ U_{N-1} \\ U_N \end{bmatrix}=\begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ \vdots \\ d_{N-1} \\ d_N \end{bmatrix}. \quad (40)$$

Burada iç noktaların katsayıları,

$$a_i=-\alpha/\Delta x^2, \qquad i=2,3,\dots,N-1$$
$$b_i=\left(\frac{1}{\Delta t}\right)+\left(\frac{2\alpha}{\Delta x^2}\right),$$
$$c_i=-\alpha/\Delta x^2,$$
$$d_i=\left(\frac{1}{\Delta t}\right)U_{i,j-1}$$

şeklindedir. Denklem (10)'da Dirichlet sınır koşulları için

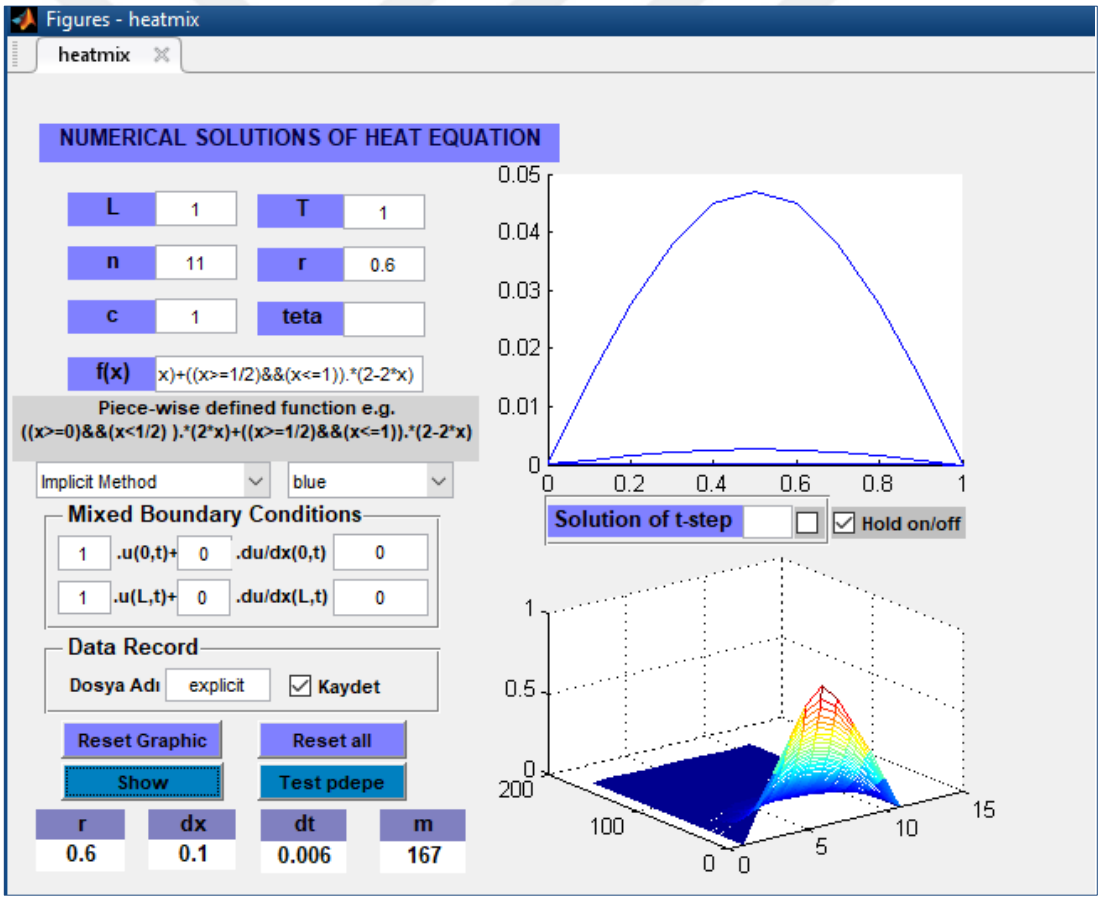
$$b_1=1, \quad c_1=0, \quad d_1=U_0$$
$$a_N=0, \quad b_N=1, \quad d_n=U_L$$

yazılabilir.

BTCS taslağının uygulamasında her zaman adımında bir denklem sisteminin çözülmesi gerekmektedir. Kodun geliştirilmesi zorluğunun yanısıra, BTCS şemasında her zaman adımı için hesaplama adımları, FTCS taslağının her zaman adımı için hesaplama adımlarından daha fazladır. BTCS şemasının, FTCS şemasına göre çok büyük bir avantajı ise koşulsuz olarak (ısı denklemine ilişkin çözümler için) kararlı olmasıdır. BTCS taslağı FTCS taslağı kadar doğru sonuç vermektedir. Bu nedenle, biraz fazladan bir çabayla, BTCS taslağı Δt ve Δx seçimlerine göre sağlam bir hesaplama

modeli verir. Buna rağmen bu avantajı her zaman başarılı değildir ve FTCS taslağı yine de bazı problemler için kullanışlıdır.

Explicit yöntemde yakınsamayan problem için bu yöntemde yakınsamanın olduğu görülebilir. Aynı problem olan $u_t = u_{xx},\ 0 < x < 1,\ t > 0$ ısı probleminin sayısal çözümünü sonlu farklarla $u(x,0) = \begin{cases} 2x, & 0 \le x \le \frac{1}{2} \\ 2(1-x), & \frac{1}{2} \le x \le 1 \end{cases}$ başlangıç koşuluna ve $u(0,t) = u(1,t) = 0$ sınır koşuluna göre sayısal çözümleri bu yöntemde artık tüm r değerleri için yakınsak olduğu görülebilir. İlk yöntemde ıraksak olan problem $r = 0.6$ için artık yakınsamaktadır ve sonuçlar Şekil (23)'de görüldüğü gibidir.



Şekil 23. Isı denkleminin $\boldsymbol{u(x,0) = \begin{cases} 2x, & 0 \le x \le \frac{1}{2} \\ 2(1-x), & \frac{1}{2} \le x \le 1 \end{cases}}$ başlangıç koşuluna ve $\boldsymbol{u(0,t) = u(1,t) = 0}$ koşuluna göre Implicit yöntem ile sayısal çözümleri

Örnek için sonuçlar Tablo (4)'de verilmiştir.

Tablo 4. Isı denklemi Implicit yöntem için sayısal veriler tablo değerleri

		0.0000	0.1991	0.3966	0.5885	0.7614	0.8698	0.7614	0.5885	0.3966	0.1991	0.0000
		0.0000	0.1966	0.3890	0.5687	0.7155	0.7856	0.7155	0.5687	0.3890	0.1966	0.0000
		0.0000	0.1924	0.3778	0.5445	0.6709	0.7231	0.6709	0.5445	0.3778	0.1924	0.0000
		0.0000	0.1867	0.3641	0.5185	0.6297	0.6721	0.6297	0.5185	0.3641	0.1867	0.0000
		0.0000	0.1800	0.3488	0.4922	0.5918	0.6283	0.5918	0.4922	0.3488	0.1800	0.0000
		0.0000	0.1726	0.3328	0.4664	0.5570	0.5894	0.5570	0.4664	0.3328	0.1726	0.0000
		0.0000	0.1648	0.3166	0.4415	0.5247	0.5541	0.5247	0.4415	0.3166	0.1648	0.0000
		0.0000	0.1569	0.3006	0.4175	0.4946	0.5217	0.4946	0.4175	0.3006	0.1569	0.0000
		0.0000	0.1490	0.2849	0.3948	0.4666	0.4916	0.4666	0.3948	0.2849	0.1490	0.0000
		0.0000	0.1413	0.2698	0.3731	0.4403	0.4636	0.4403	0.3731	0.2698	0.1413	0.0000
		0.0000	0.1339	0.2553	0.3526	0.4156	0.4374	0.4156	0.3526	0.2553	0.1339	0.0000
		0.0000	0.1267	0.2414	0.3331	0.3923	0.4128	0.3923	0.3331	0.2414	0.1267	0.0000
		0.0000	0.1198	0.2283	0.3147	0.3704	0.3897	0.3704	0.3147	0.2283	0.1198	0.0000
		0.0000	0.1133	0.2157	0.2973	0.3498	0.3679	0.3498	0.2973	0.2157	0.1133	0.0000
		0.0000	0.1071	0.2039	0.2808	0.3303	0.3474	0.3303	0.2808	0.2039	0.1071	0.0000
		0.0000	0.1012	0.1926	0.2653	0.3120	0.3281	0.3120	0.2653	0.1926	0.1012	0.0000
		0.0000	0.0956	0.1820	0.2506	0.2946	0.3098	0.2946	0.2506	0.1820	0.0956	0.0000
		0.0000	0.0903	0.1719	0.2367	0.2783	0.2926	0.2783	0.2367	0.1719	0.0903	0.0000
		0.0000	0.0854	0.1624	0.2235	0.2628	0.2764	0.2628	0.2235	0.1624	0.0854	0.0000
		0.0000	0.0806	0.1534	0.2111	0.2482	0.2610	0.2482	0.2111	0.1534	0.0806	0.0000
		0.0000	0.0762	0.1449	0.1994	0.2345	0.2465	0.2345	0.1994	0.1449	0.0762	0.0000
		0.0000	0.0719	0.1369	0.1884	0.2215	0.2329	0.2215	0.1884	0.1369	0.0719	0.0000
		0.0000	0.0680	0.1293	0.1779	0.2092	0.2199	0.2092	0.1779	0.1293	0.0680	0.0000
		0.0000	0.0642	0.1221	0.1681	0.1976	0.2077	0.1976	0.1681	0.1221	0.0642	0.0000
		0.0000	0.0606	0.1153	0.1587	0.1866	0.1962	0.1866	0.1587	0.1153	0.0606	0.0000

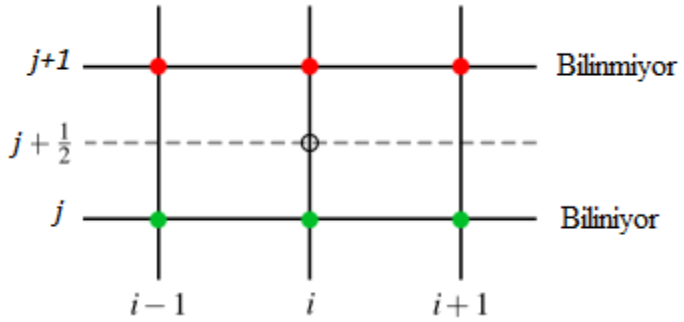
3.4. Crank-Nicolson Yöntemi

FTCS ve BTCS taslakları, $\mathcal{O}(\Delta t)$ değerinde zamansal kesme hatasına sahiptirler. Doğru zaman çözümler önemli olduğunda Crank-Nicolson taslağının önemli avantajları vardır. Crank-Nicolson taslağı BTCS taslağını uygulamasından çok daha zor değildir ve $\mathcal{O}(\Delta t^2)$ değerinde bir zamansal kesme hatasına sahiptir. Crank-Nicolson şeması, BTCS gibi kapalıdır ve aynı zamanda koşulsuz kararlıdır.

Isı denkleminin sol tarafı, BTCS taslağında da kullanılan geri yönde farkla ifade edilir. Isı denkleminin sağ tarafı ise mevcut ve önceki zaman adımında hesaplanan merkezi fark taslağının ortalamasıyla ifade edilir. Bundan dolayı Denklem (8) sonlu fark formülleriyle beraber

$$\frac{U_{i,j}-U_{i,j-1}}{\Delta t}=\frac{\alpha}{2}\left[\frac{U_{i-1,j}-2U_{i,j}+U_{i+1,j}}{\Delta x^2}+\frac{U_{i-1,j-1}-2U_{i,j-1}+U_{i+1,j-1}}{\Delta x^2}\right] \tag{41}$$

şeklinde yazılır. Bu yöntem için ağ yapısı Şekil (24)'de olduğu gibidir.



Şekil 24. Crank-Nicolson taslağı için ağ noktaları gösterimi

j ve $j-1$ zaman adımındaki U değerlerinin sağ tarafta yer aldıklarına dikkat edilmelidir. Denklem (41) j. anda U değerlerini hesaplamak için kullanılır ve böylece $j-1$. andaki U'nun bütün değerlerinin bilindiği varsayılır. Denklem (41)'in, j anındaki U değerleri solda ve $j-1$ anındaki U değerleri sağda olacak şekilde yeniden düzenlenmesiyle aşağıdaki denklem elde edilir.

$$-\frac{\alpha}{2\Delta x^2}U_{i-1,j}+\left(\frac{1}{\Delta t}+\frac{\alpha}{\Delta x^2}\right)U_{i,j}-\frac{\alpha}{2\Delta x^2}U_{i+1,j}=\frac{1}{\Delta t}U_{i,j-1}+\frac{\alpha}{2\Delta x^2}U_{i-1,j-1}+\left(\frac{1}{\Delta t}+\frac{\alpha}{\Delta x^2}\right)U_{i-1,j-1}-\frac{\alpha}{2\Delta x^2}U_{i+1,j-1} \quad (42)$$

Crank-Nicolson şeması kapalıdır ve dolayısıyla, U için bir denklemler sisteminin her zaman adımında çözülmesi gerekmektedir. Denklem sistemi, (40) Denkleminin formuyla aynıdır, sadece farklı katsayılara sahiptir:

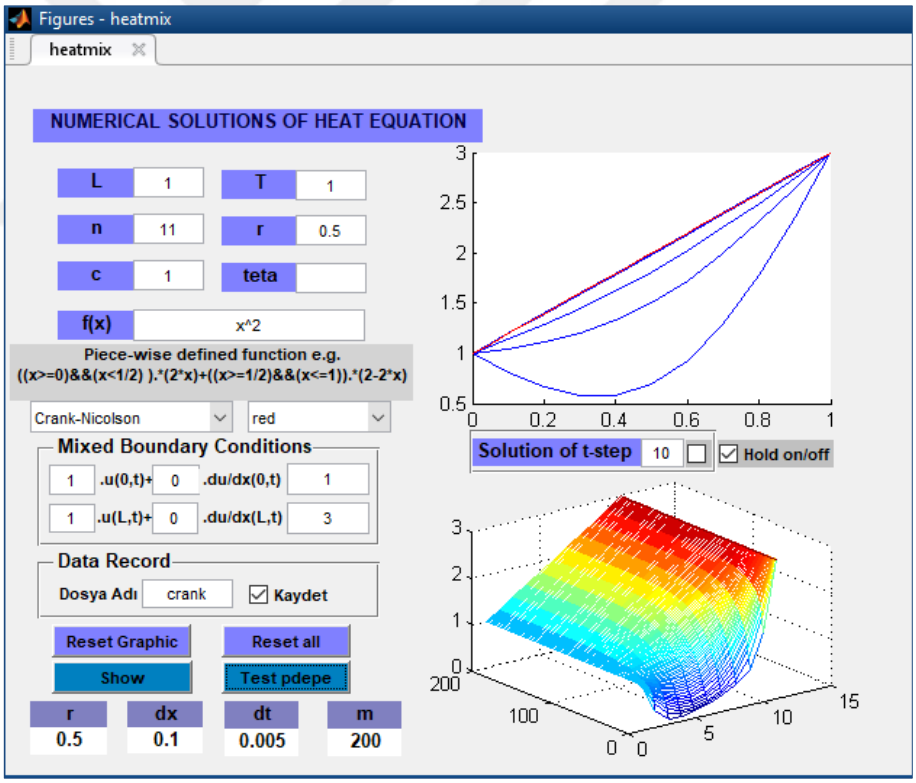
$$a_i=-\alpha/(2\Delta x^2), \qquad i=2,3,\dots,N-1$$
$$b_i=\left(\frac{1}{\Delta t}\right)+\left(\frac{\alpha}{\Delta x^2}\right),$$
$$c_i=-\alpha/(2\Delta x^2),$$
$$d_i=\left(\frac{1}{\Delta t}\right)U_{i,j-1}+a_iU_{i-1,j-1}+(a_i+c_i)U_{i,j-1}+c_iU_{i+1,j-1}$$

Crank-Nicolson taslağı için a_i, b_i ve c_i değerleri BTCS taslağındaki hesaplanan değerlerden farklıdır. Bu küçük farklılık dışında, BTCS ile Crank-Nicolson taslağı arasındaki yegâne önemli uygulama farkı, d_i'de ekstra U_{j-1} terimlerinin yer alıyor olmasıdır.

Algoritmik olarak, BTCS ve Crank-Nicolson taslağı birbirine çok benzer. Crank-Nicolson şeması $\mathcal{O}(\Delta t^2)+\mathcal{O}(\Delta x^2)$ şeklinde bir kesme hatasına sahiptir. Bir başka

deyişle, zamansal kesme hatası, BTCS şemasının zamansal kesme hatasından önemli ölçüde daha küçüktür.

Örnek olarak, $u_t = u_{xx},\ 0 < x < 1,\ t > 0$ ısı probleminin sayısal çözümünü sonlu farklarla $u(x,0) = x^2$ başlangıç koşuluna ve $u(0,t) = 1,\ u(1,t) = 3$ sınır koşuluna incelenebilir. Bu durumda veriler için GUI arayüzü Şekil (25)'de ve sayısal veriler Tablo (5)'de olduğu gibidir. Kırmızı çizgi ile gösterilen çözüm Matlab'ın kendi çözüm aracı olan pdepe komutu ile oluşturulmuştur. Bu çözüm denge durumundaki çözümü ifade etmektedir. Sonlu farklar ile elde edilen çözümün tutarlı bir çözüm olup olmadığı buradan kontrol edilebilmektedir. Her bir örnek için pdepe'yi kullanarak çözümlerin sağlaması kolayca yapılabilir. Bu tip seçenek için farklı renk kullanarak sayısal çözümle karıştırılmamalıdır.



Şekil 25. Isı denkleminin $\boldsymbol{u(x,0) = x^2}$ başlangıç koşuluna ve $\boldsymbol{u(0,t) = 1, u(1,t) = 3}$ koşuluna göre Crank-Nicolson yöntemi ile sayısal çözümleri

Tablo 5. Isı denklemi Crank-Nicolson yöntem için sayısal veriler tablo değerleri

1	1.0000	0.3614	0.1086	0.1101	0.1718	0.2609	0.3735	0.5202	0.7675	1.5046	3.0000
2	1.0000	0.5094	0.2251	0.1527	0.1906	0.2763	0.4000	0.5955	0.9916	1.7947	3.0000
3	1.0	0.5939	0.3194	0.2101	0.2203	0.3015	0.4455	0.7000	1.1721	1.9589	3.0000
4	1.0000	0.6501	0.3937	0.2690	0.2601	0.3392	0.5063	0.8061	1.3127	2.0671	3.0000
5	1.0000	0.6913	0.4539	0.3254	0.3069	0.3878	0.5752	0.9052	1.4248	2.1453	3.0000
6	1.0000	0.7235	0.5045	0.3791	0.3583	0.4439	0.6472	0.9957	1.5169	2.2054	3.0000
7	1.0000	0.7500	0.5487	0.4303	0.4123	0.5042	0.7194	1.0782	1.5945	2.2537	3.0000
8	1.0000	0.7728	0.5883	0.4796	0.4675	0.5663	0.7902	1.1537	1.6614	2.2939	3.0000
9	1.0000	0.7932	0.6249	0.5271	0.5228	0.6286	0.8586	1.2230	1.7203	2.3282	3.0000
10	1.0000	0.8117	0.6591	0.5731	0.5774	0.6901	0.9243	1.2870	1.7728	2.3583	3.0000
11	1.0000	0.8290	0.6916	0.6175	0.6309	0.7500	0.9870	1.3464	1.8204	2.3850	3.0000
12	1.0000	0.8454	0.7226	0.6605	0.6830	0.8079	1.0467	1.4018	1.8638	2.4090	3.0000
13	1.0000	0.8610	0.7524	0.7021	0.7334	0.8637	1.1034	1.4536	1.9038	2.4309	3.0000
14	1.0000	0.8759	0.7810	0.7421	0.7820	0.9172	1.1572	1.5021	1.9409	2.4511	3.0000
15	1.0000	0.8902	0.8085	0.7807	0.8287	0.9684	1.2083	1.5477	1.9754	2.4697	3.0000
16	1.0000	0.9040	0.8350	0.8179	0.8737	1.0174	1.2568	1.5906	2.0076	2.4871	3.0000
17	1.0000	0.9172	0.8604	0.8536	0.9167	1.0641	1.3028	1.6310	2.0379	2.5033	3.0000
18	1.0000	0.9300	0.8849	0.8879	0.9579	1.1087	1.3464	1.6692	2.0663	2.5185	3.0000
19	1.0000	0.9422	0.9084	0.9207	0.9974	1.1512	1.3879	1.7054	2.0931	2.5327	3.0000
20	1.0000	0.9540	0.9310	0.9522	1.0351	1.1917	1.4273	1.7395	2.1183	2.5461	3.0000
21	1.0000	0.9653	0.9526	0.9824	1.0711	1.2302	1.4646	1.7719	2.1422	2.5588	3.0000
22	1.0000	0.9761	0.9734	1.0112	1.1055	1.2670	1.5002	1.8026	2.1647	2.5707	3.0000
23	1.0000	0.9865	0.9932	1.0388	1.1383	1.3020	1.5339	1.8317	2.1861	2.5821	3.0000
24	1.0000	0.9964	1.0122	1.0652	1.1697	1.3353	1.5660	1.8593	2.2063	2.5928	3.0000
25	1.0000	1.0059	1.0304	1.0904	1.1996	1.3671	1.5965	1.8855	2.2255	2.6029	3.0000

3.5. Ağırlıklı Ortalama (Weighted-Average) Yöntemi

Konumda merkezi fark olmak koşuluyla ileri ve geri zamana ait farkların ağırlıklı ortalaması

$$\frac{U_{i,j+1}-U_{i,j}}{\Delta t} = \alpha \frac{\theta\delta^2 U_{i,j+1}+(1-\theta)\delta^2 U_{i,j}}{\Delta x^2} \tag{43}$$

şeklinde tanımlıdır. δ merkezi fark operatörü ile,

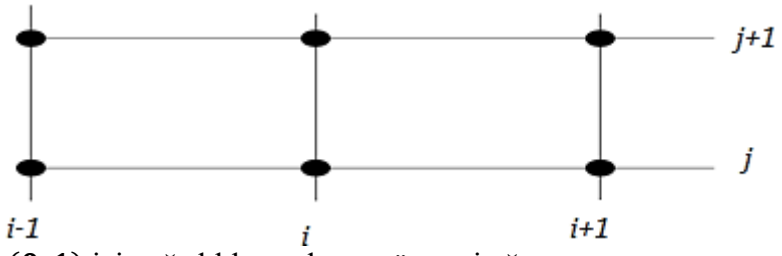
$$\delta U_{i,j} = U_{i,j+\frac{1}{2}} - U_{i,j-\frac{1}{2}} \tag{44}$$

ve

$$\delta^2 U_{i,j} = U_{i,j+1} - 2U_{i,j} + U_{i,j-1} \tag{45}$$

fark ifadeleri daha sade yazılabilmektedir.

θ parametresi, sıfır ve bir arasında seçilen bir ağırlık çarpanıdır $0 < \theta < 1$ aralığında ağırlıklı ortalama şablonu Şekil (26)'da verilmiştir.



Şekil 26. $\boldsymbol{\theta} \in (\mathbf{0}, \mathbf{1})$ için ağırlıklı ortalama yöntemi ağ yapısı

Ağırlıklı Ortalama (Weighted-Average) taslağı içerisinde farklı parametrelerle diğer yöntemlerde ortaya çıkmaktadır.

Eğer $\theta = 0$ ise (43) nolu denklem açık (Explicit) yöntem,

Eğer $\theta = 1$ ise (43) nolu denklem kapalı (Implicit) yöntem,

Eğer $\theta = \frac{1}{2}$ ise (43) nolu denklem Crank Nicolson yöntemi,

olarak adlandırılır.

Bu yöntemde $\theta \neq 0$ değeri için başlangıç-sınır değer problemini çözmek istiyoruz. Her zamanki gibi $U_{i,j}$, $i = 1, 2, \dots, N$, bilinen çözümün olduğunu varsayalım ve $U_{i,j+1}$, $i = 1, \dots, N$ için çözümü belirlemek istiyoruz. Sınır koşullarını kullanarak $f_j \equiv f(j\Delta t)$ olmak üzere

$$U_{0,j+1} = f_{j+1} \quad , U_{i,j+1} = g_{j+1}, \tag{46}$$

elde edilir.

Denklem (43)'de farkları hesaplamak için Denklem (45) ile beraber tüm iç düğüm noktalarındaki sonuçları $i = 1,2, \dots, N-1$ için

$$-r\theta U_{i-1,j+1} + (1 + 2r\theta)U_{i,j+1} - r\theta U_{i+1,j+1} = r(1-\theta)U_{i-1,j} + [1 - 2r(1-\theta)]U_{i,j} + r(1-\theta)U_{i+1,j} \tag{47}$$

elde edilir. Eğer $U_{0,j+1}$ ve $U_{i,j+1}$ bilinen değerler olarak dikkate alınırsa (47) nolu denklem $U_{i,j+1}$, $i = 1,2, \dots, N-1$ bilinmeyenler için $N-1$ denklemden oluşan lineer bir sistemdir. (47) denklemi vektörel formda

$$\mathrm{U}^{j} = [U_{1,j}, U_{2,j}, \dots, U_{N-1,j}]^{T} \tag{48}$$

şeklinde yazılabilir. Vektörel formda n bilinmeyeni kullanarak (47) denklemi yeniden yazılırsa $C = \begin{bmatrix} -2 & 1 & & \\ 1 & -2 & 1 & \\ & \ddots & \ddots & \ddots \\ & & 1 & -2 \end{bmatrix}$ ve $f^{j} = \begin{bmatrix} \theta f_{j+1} + (1-\theta) f_j \\ 0 \\ \vdots \\ 0 \\ \theta g_{j+1} + (1-\theta) g_j \end{bmatrix}$ olmak üzere

$$[\mathrm{I} - r\theta C]\mathrm{U}^{j+1} = [\mathrm{I} + r(1-\theta)C]\mathrm{U}^{j} + rf^{j} \tag{49}$$

elde edilir.

Denklem (49) gibi tridiagonal sistemler sayısal analizde sıkça ortaya çıkarlar ve onları çözecek hızlı bir algoritmaya ihtiyaç duyulur. Bazı özel durumlarda Gauss eliminasyon işe yarar. Fakat daha geniş bir çözüm yolu üzerinde durulmalıdır. F ve X, n −boyutlu vektörler ve $A = \begin{bmatrix} a_1 & c_1 & & & \\ b_2 & a_2 & c_2 & & \\ & & \ddots & & \\ & & b_{N-1} & a_{N-1} & c_{N-1} \\ & & & b_N & a_N \end{bmatrix}$ tridiagonal matrisi için

$$\mathrm{AX} = \mathrm{F} \tag{50}$$

sistemini ele alalım. Buna göre

$$a_i = 1 + 2r\theta, \qquad i = 1,2, \dots, N, \tag{51}$$

$$b_i = -r\theta, \qquad i = 2,3, \dots, N, \tag{52}$$

$$c_i = -r\theta, \qquad i = 1,2, \dots, N-1, \tag{53}$$

ve buradan $i = 1,2, \dots, N$ için

$$F_i = r(1-\theta)U_{i-1,j} + [1-2r(1-\theta)]U_{i,j} + r(1-\theta)U_{i+1,j} + r\delta_{i,1}[\theta f^{j+1} + (1-\theta)f^j] + r\delta_{i,N}[\theta g^{j+1} + (1-\theta)g^j] \quad (54)$$

elde edilir. $\delta_{j,k}$ değeri Kronecker deltası olup $i = k$ olduğunda birim değer, aksi halde sıfır değerini alır. Pivotlamanın gerekli olmadığını farz edersek A matrisi

$$A = LU \quad (55)$$

şeklinde düşünülebilir. Burada L ve U bidiagonal forma sahip alt ve üst matrislerdir. Buna göre matrisler $L = \begin{bmatrix} 1 & & & & \\ l_2 & 1 & & & \\ & l_3 & 1 & & \\ & & & \ddots & \\ & & & l_N & 1 \end{bmatrix}$ ve $U = \begin{bmatrix} u_1 & v_1 & & & \\ & u_2 & v_2 & & \\ & & \ddots & & \\ & & & u_{N-1} & v_{N-1} \\ & & & & u_N \end{bmatrix}$ şeklinde yazılabilirler.

$l_i, i = 2,3,\dots,N$, $u_i, i = 1,2,\dots,N$ ve $v_i, i = 1,2,\dots,N-1$ katsayıları belirlendiğinde Denklem (50) ileri ve geriye doğru yerine koyma yoluyla kolayca çözülebilir. Böylece Denklem (50) ve Denklem (55) denklemlerini kullanarak

$$LUX = F \quad (56)$$

$$UX = Y \quad (57)$$

$$LY = F \quad (58)$$

sistemleri elde edilir.

Denklem (58)'den Y bilinmeyeni ileriye doğru yerine koyma ile elde edilir. Y değeri hesaplandığında Denklem (57)'den X bilinmeyeni geriye doğru yerine koyma ile çözülebilir.

L ve U katsayıları direk hesaplama ile elde edilebilir. Buna göre katsayıları

$$u_1 = a_1 \quad (59)$$

$$l_i = b_i/u_{i-1}\,, u_i = a_i - l_i c_{i-1}\; i = 2,3,\dots,N \tag{60}$$

$$v_i = c_i,\; i = 2,3,\dots,N \tag{61}$$

şeklinde elde etmek mümkündür. $l_i, i = 2,3,\dots,N$, $u_i, i = 1,2,\dots,N$, ve $v_i, i = 1,2,\dots,N-1$ olarak belirlendiğinde Denklem (57)-(58)'dan

$$Y_1 = F_1,\; Y_i = F_i\text{-}l_i Y_{i-1},\; i = 2,3,\dots,N \tag{62}$$

$$X_N = y_N/u_N\,,\; X_i = (Y_i - v_i X_{i+1})/u_i\,, i = N-1, N-2,\dots,1 \tag{63}$$

bulunur. Bu prosedür tridiagonalleştirme algoritması olarak bilinir. Etkin uygulama için a_j, b_j, c_j ve F_j ile u_j, l_j, v_j ve x_j değerleri yazılmalıdır.

Ağırlıklı ortalama taslağı Denklem (47) için kararlılığı von Neumann yöntemi ile açıklanabilir. Başlangıç koşullarının ve çözümün x'e göre periyodik olduğunu düşünerek çözüm ayrık Fourier serisi olarak yazılabilir.

$$U_{i,j} = \sum_{k=0}^{N} A_{i,k}\, w_{k,j},\; w_j = e^{2\pi i j/(N+1)}. \tag{64}$$

Denklem (64), Denklem (47)'da yerine yazılırsa,

$$\sum_{k=0}^{N}\{ A_{i+1,k}\left[-r\theta e^{-2\pi ik/(N+1)} + 1 + 2r\theta - r\theta e^{2\pi ik/(N+1)}\right] - A_{i,k}[r(1-\theta)e^{-2\pi ik/(N+1)} + 1 - 2r(1-\theta) + r(1-\theta)e^{2\pi ik/(N+1)}]\}\, w_{k,j} = 0$$

elde edilir. Ortagonal özelliği ve Euler özdeşliğiyle $M_k = 1 - \frac{2r(1-cos2k\pi/(N+1))}{1+2r\theta(1-cos2k\pi/(N+1))}$ olmak üzere $A_{i+1,k} = M_k A_{i,k}$ yazılabilir. Yarım açı formülünü kullanarak,

$$M_k = 1 - \frac{4r sin^2 k\pi/(N+1)}{1+4r\theta sin^2 k\pi/(N+1)} \tag{65}$$

bulunur. Alışıldığı gibi, $A_{i,k}$ için birinci mertebeden fark bağıntılarını $A_{i,k} = (M_k)^i A_{0,k}$ alarak çözebiliriz. Çözümün azalan bir çözüm olduğu düşüncesine dayanarak tüm k değerleri için $|M_k| \leq 1$ yazılabilir. M_k, reel ve r ve θ negatif olmadığından

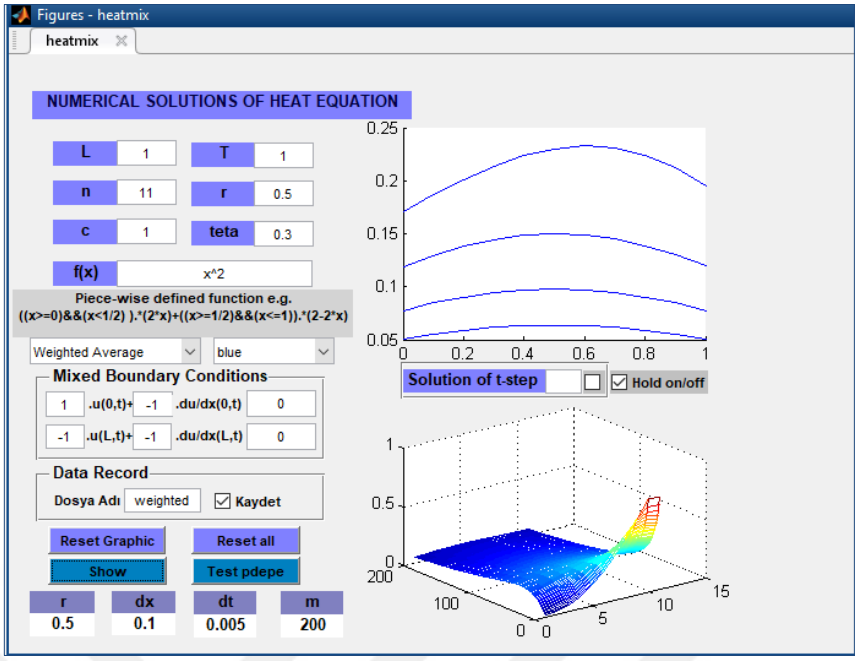
$$-1 \leq 1 - \frac{4rsin^2 k\pi/(N+1)}{1+4r\theta sin^2 k\pi/(N+1)} \leq 1 \tag{66}$$

Sağdaki eşitsizlik her zaman sağlanır. Böylece $\frac{4rsin^2 k\pi/(N+1)}{1+4r\theta sin^2 k\pi/(N+1)} \leq 2$ veya $2r(1-2\theta)sin^2 \frac{k\pi}{(N+1)} \leq 1$ yazılabilir. Tüm k değerleri için bu sağalandığından

$$2r(1-2\theta) \leq 1 \tag{67}$$

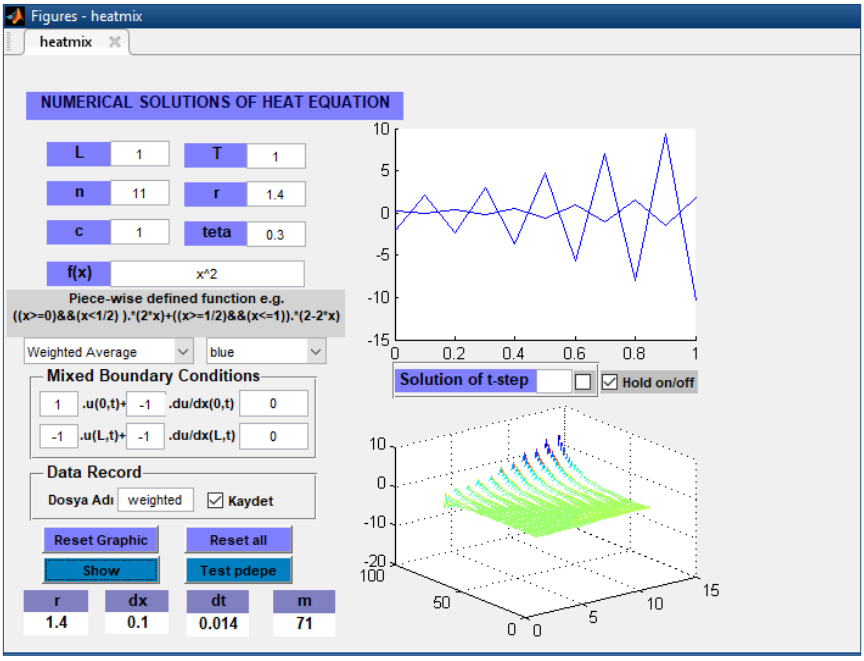
sınırlaması görülür. Eğer $1/2 \leq \theta \leq 1$ ise $1-2\theta$ terimi pozitif olmayacak ve Denklem (67) tüm $r > 0$ değerleri için sağlanıyor olacaktır. Böylece Denklem (47) tüm seçilen Δt ve Δx için kararlıdır. Bu durumda koşulsuz kararlıdır denir. Eğer $0 \leq \theta \leq 1/2$ ise bu durumda $|M_k| \leq 1$ eşitsizliği $r \leq \frac{1}{2(1-2\theta)}$ için sağlanır.

Örnek olarak, $u_t = u_{xx},\ 0 < x < 1,\ t > 0$ ısı probleminin sayısal çözümünü sonlu farklarla $u(x,0) = x^2$ başlangıç koşuluna ve $u_x(0,t) = u(0,t),\ u_x(1,t) = -u(1,t)$ karışık sınır koşuluna incelenebilir. $\theta = 0.3$ olarak seçilmiştir. Buna göre GUI arayüzü ve grafikler Şekil (27)'de olduğu gibidir.



Şekil 27. Isı denkleminin $\boldsymbol{u(x,0) = x^2}$ başlangıç koşuluna ve $\boldsymbol{u_x(0,t) = u(0,t)}$, $\boldsymbol{u_x(1,t) = -u(1,t)}$ koşuluna göre ağırlıklı ortalama yöntemi ile sayısal çözümleri

Yakınsama koşulundan $\theta = 0.3$ değeri için $r \leq \frac{1}{2(1-2\theta)} = 0.7143$ şartı sağlanması gerekir. Bunun dışında seçilen $r = 1.4$ değeri için çözümün ıraksadığı Şekil (28)'de görülmektedir.



Şekil 28. Ağırlıklı ortalama yöntemi için yakınsama şartının sağlanmaması

3.6. Dufort Frankel Yöntemi

Daha kararlı bir algoritma üretmek için önceki yöntemler değiştirilebilir. Bunun için denklemin sağ tarafındaki $U_{i,j}$ yerine önceki ve şimdiki zamanın ortalama değerleri olan $n-1$ ve $n+1$. adım değerleri konularak denklemin yeni hali elde edilir. Bu yeni formülasyon Dufort Frankel yöntemi olarak adlandırılır. Sonlu fark formülleri

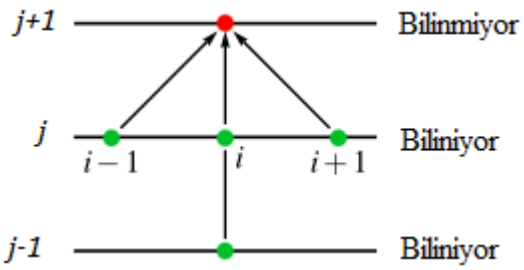
$\frac{U_{i,j+1}-U_{i,j-1}}{2\Delta t} = \frac{\alpha}{\Delta x^2}\left[U_{i+1,j} - 2\left(\frac{U_{i,j-1}+U_{i,j+1}}{2}\right) + U_{i-1,j}\right]$ veya,

$U_{i,j+1} = U_{i,j-1} + \frac{2\alpha\Delta t}{\Delta x^2}(U_{i+1,j} - U_{i,j-1} - U_{i,j+1} + U_{i-1,j})$ veya,

$$(1+2d)U_{i,j+1} = (1-2d)U_{i,j-1} + 2d(U_{i-1,j} + U_{i+1,j}) \tag{68}$$

ile verilmektedir. Bu yöntem için ağ yapısı Şekil(29)'da olduğu gibidir.

Bu yöntem açık bir yöntemdir ve von Neumann kararlılık analizine göre koşulsuz yakınsaktır. Dufort Frankel yöntemi iki seviyeli bir yöntemdir. Yani şablon geçerli seviye n'den farklı iki zaman düzeyinde u değerleri içerir. Sonuç olarak, hesaplamayı başlatmak için n ve $n-1$'deki u değerleri gereklidir. Bu nedenle, tek adımlı bir yöntem ek veri kümelerini elde etmek için kullanılmalıdır. Yöntemin doğruluk mertebesi sırası ile $\mathcal{O}(\Delta t^2, \Delta x^2, (\Delta t/\Delta x)^2)$ dir. Yöntem kayıtsız yakınsak olsa bile doğru çözüm, $\Delta t \ll \Delta x$ şartı sağlandığında elde edilebilecektir.



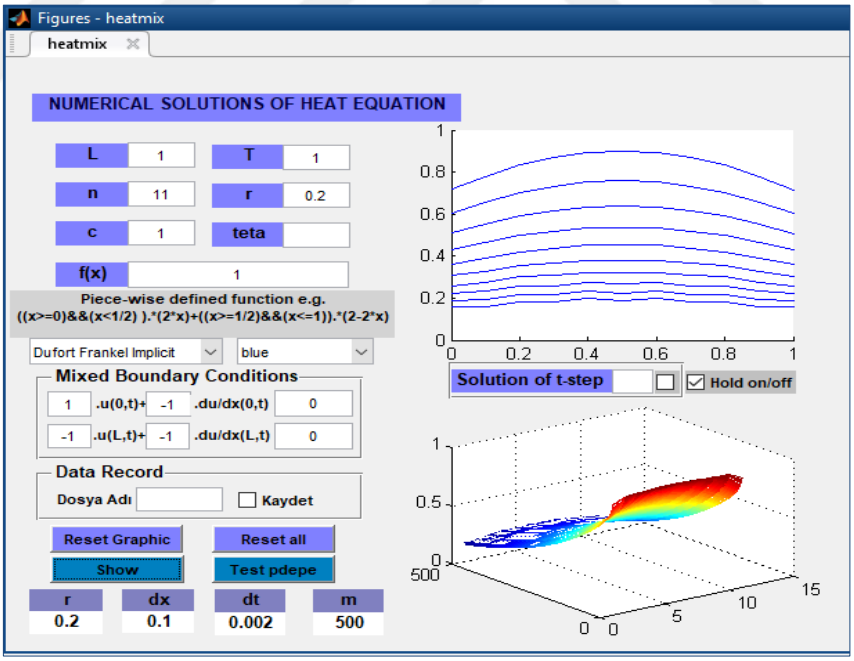
Şekil 29. Dufort Frankel taslağı için ağ noktaları gösterimi

Örnek olarak, $u_t = u_{xx}$, $0 < x < 1$, $t > 0$ ısı probleminin sayısal çözümünü sonlu farklarla $u(x,0) = 1$ başlangıç koşuluna ve $u_x(0,t) = u(0,t)$, $u_x(1,t) =$

$-u(1,t)$ karışık sınır koşuluna incelenebilir. Buna göre sayısal çözümler Tablo (6)'de ve GUI arayüzü Şekil (30)'da olduğu gibidir.

Tablo 6. Isı denklemi Dufort Frankel yöntem için sayısal veriler tablo değerleri

1	0.9421	0.9916	0.9988	0.9998	1.0000	1.0000	1.0000	0.9998	0.9988	0.9916	0.9421
2	0.9289	0.9813	0.9973	0.9996	0.9999	1.0000	0.9999	0.9996	0.9973	0.9813	0.9289
3	0.9114	0.9753	0.9940	0.9991	0.9999	1.0000	0.9999	0.9991	0.9940	0.9753	0.9114
4	0.9033	0.9650	0.9915	0.9981	0.9997	0.9999	0.9997	0.9981	0.9915	0.9650	0.9033
5	0.8904	0.9594	0.9869	0.9971	0.9994	0.9998	0.9994	0.9971	0.9869	0.9594	0.8904
6	0.8845	0.9499	0.9839	0.9952	0.9990	0.9996	0.9990	0.9952	0.9839	0.9499	0.8845
7	0.8739	0.9450	0.9787	0.9939	0.9983	0.9994	0.9983	0.9939	0.9787	0.9450	0.8739
8	0.8691	0.9364	0.9757	0.9914	0.9976	0.9988	0.9976	0.9914	0.9757	0.9364	0.8691
9	0.8599	0.9321	0.9702	0.9898	0.9965	0.9984	0.9965	0.9898	0.9702	0.9321	0.8599
10	0.8560	0.9242	0.9672	0.9868	0.9956	0.9975	0.9956	0.9868	0.9672	0.9242	0.8560
11	0.8478	0.9204	0.9618	0.9850	0.9940	0.9968	0.9940	0.9850	0.9618	0.9204	0.8478
12	0.8443	0.9131	0.9589	0.9817	0.9929	0.9955	0.9929	0.9817	0.9589	0.9131	0.8443
13	0.8369	0.9097	0.9536	0.9798	0.9909	0.9946	0.9909	0.9798	0.9536	0.9097	0.8369
14	0.8338	0.9029	0.9508	0.9763	0.9896	0.9929	0.9896	0.9763	0.9508	0.9029	0.8338
15	0.8270	0.8998	0.9456	0.9743	0.9873	0.9918	0.9873	0.9743	0.9456	0.8998	0.8270
16	0.8242	0.8934	0.9429	0.9707	0.9859	0.9897	0.9859	0.9707	0.9429	0.8934	0.8242
17	0.8178	0.8905	0.9379	0.9687	0.9832	0.9884	0.9832	0.9687	0.9379	0.8905	0.8178
18	0.8154	0.8845	0.9353	0.9649	0.9817	0.9860	0.9817	0.9649	0.9353	0.8845	0.8154
19	0.8093	0.8818	0.9303	0.9628	0.9788	0.9846	0.9788	0.9628	0.9303	0.8818	0.8093
20	0.8071	0.8761	0.9279	0.9590	0.9771	0.9819	0.9771	0.9590	0.9279	0.8761	0.8071
21	0.8014	0.8737	0.9230	0.9569	0.9740	0.9803	0.9740	0.9569	0.9230	0.8737	0.8014
22	0.7993	0.8682	0.9207	0.9530	0.9723	0.9774	0.9723	0.9530	0.9207	0.8682	0.7993
23	0.7939	0.8659	0.9159	0.9510	0.9690	0.9757	0.9690	0.9510	0.9159	0.8659	0.7939
24	0.7920	0.8606	0.9137	0.9470	0.9672	0.9726	0.9672	0.9470	0.9137	0.8606	0.7920
25	0.7867	0.8584	0.9090	0.9449	0.9637	0.9708	0.9637	0.9449	0.9090	0.8584	0.7867



Şekil 30. Isı denkleminin $\boldsymbol{u(x,0)=1}$ başlangıç koşuluna ve $\boldsymbol{u_x(0,t)=u(0,t)}$, $\boldsymbol{u_x(1,t)=-u(1,t)}$ koşuluna göre Dufort Frankel yöntemi ile sayısal çözümleri

4. TARTIŞMA ve SONUÇLAR

Isı denklemleri farklı başlangıç ve sınır değerlerle sonlu farklar kullanarak sayısal olarak çözülmüştür. Elde edilen bu çözümler GUI'de geliştirilen arayüzlerle etkileşimli bir şekilde incelemeye imkan verecek şekilde hazırlanmıştır. Hiçbir program bilgisine gerek duymadan sadece parametre değerleri girerek çözümlerin kolay bir şekilde incelenmesi sağlanmıştır. Elde edilen çözümlerin analitik çözümlerle uyumlu olduğu gözlemlenmiştir.

5. ÖNERİLER

Farklı boyutta verilen ısı denklemleri için daha farklı arayüzler geliştirilebilir. İncelenen bu yöntemler haricinde faklı yöntemler için Matlab GUI'de arayüzler tasarlanabilir. Tasarlanan arayüzlerin eğitimde özellikle lisans öğrencileri için fayda sağlayacağı umulmaktadır. Aslında teorik olarak işlenen konunun bu şekildeki uygulamaları ile öğrenilen bilgiler pekiştirilebilmektedir.

KAYNAKLAR

Aktaş, C., 1995. Diferansiyel Denklemlerin Nümerik Çözümleri.Yüksek Lisans Tezi. Balıkesir Üniversitesi, Fen Bilimleri Enstitüsü, Balıkesir, Türkiye, 67 s.

Ames, W.F., 1992. Numerical Methods for Partial Differential Equations. Academic Press, Inc., Boston, third edition.

Arifoğlu, U., 2005. Matlab 7.04 Simulink ve Mühendislik Uygulamaları, Alfa Yayınları, İstanbul, Türkiye, 750 s.

Bakioğlu, M., 2004. Sayısal Analiz, İstanbul: Birsen Yayınevi, pp 425-429.

Burden, R.L. and Faires, J.D., 1997. Numerical Analysis. Brooks/Cole Publishin Co., New York, sixth edition.

Causon, D.M. and Mingham, C.G., 2010. Introductory Finite Difference Methods for PDEs, ISBN 978-87-7681-642-1.

Cooper, J., 1998. Introductin to Partial Differential Equations with Matlab. Birkhauser, Boston.

Demiralp, A., 2013. Parabolik Denklemler İçin Sonlu Fark Yaklaşımları, Yüksek Lisans Tezi. İnönü Üniversitesi, Fen Bilimleri Enstitüsü, Malatya, Türkiye, 98 s.

Flaherty, J.E., 1999. Difference Methods for Parabolic Partial Differential Equation. Available: http://www.cs.rpi.edu/~flaherje/pdf/pde4.pdf

Fletcher, C.A.J., 1988. Computational Techniquess for Fluid Dynamics. Springer-Verlag, Berlin.

Golub, G. and Ortega, J.M., 1993. Scientific Computing: an Introduction with Parallel Computing. Academic Press, Inc., Boston.

Hoffman, J.D. and Frankel, S., 2001. Numerical Methods for Engineers and Scientists. New York: CRC Press, 2nd ed., ISBN: 0-8247-0443-6, pp 39-40.

Isaacson, E. and Keller, H.B., 1994. Analysis of Numerical Methods. Dover, New York.

İnan B., 2009. Isı Denkleminin Üstel Sonlu Fark Yöntemi ile Çözümü, Yüksek Lisans Tezi, İnönü Üniversitesi, Fen Bilimleri Enstitüsü, Malatya, Türkiye, 83 s.

John, H.M. and Kurtis, D.F., 1999. Numerical methods using Matlab, Upper Saddle River, NJ 07458, Prentice-Hall İnternational Editions.

Kahya, D., 2007. Isı Denklemi için Yaklaşım Teorisi Yöntemleri, Yüksek Lisans Tezi. Ankara Üniversitesi, Fen Bilimleri Enstitüsü, Türkiye, 46 s.

Kreyszig, E., 1993. Advanced Engineering Mathematics. Wiley, New York, seventh edition.

Laurene, V.F., 1999. Applied Numerical Analysis using Matlab, Prentice Hall, New Jersey.

Mathews, J.H. and Fink, K.D., 1999. Numerical Methods using Matlab, Prentice Hall, Uppet Saddle River, ISBN 0-13-270042-5.

Morton, K.W. and Mayers, D.F., 1994. Numerical Solution of Partial Differential Equations: An Introduction. Cambridge University Press, Cambridge, England.

Recktenwald, G.W., 2004. Finite-Difference Approximations to the Heat Equation. Ders Notu.

Saka, B., 1998. Parabolik Denklemlerin Nümerik Çözümleri, Yüksek Lisans Tezi. Eskişehir Osmangazi Üniversitesi, Fen Bilimleri Enstitüsü, Eskişehir, Türkiye, 111 s.

Salih, A., 2013. Finite Difference Method for Parabolic PDE. Available: https://www.iist.ac.in/sites/default/files/people/parabolic.pdf

Seshaiyer, P., 2010. Finite-Difference Method for the 1D Heat Equation. Available: http://math.gmu.edu/~pseshaiy/F12/m679/m679_F12_notes_hw1.pdf

Sewell, G., 1988. The Numerical Solution of Ordinary and Partial Differential Equations, Academic Press Inc., London, ISBN 0-12-637475-9.

Shingavera, I.K., 2011. Solving Nonlinear Partial Differential Equations with Maple and Mathematica, New York: SpringerWien, ISBN 978-3-7091-0516-0

Smith, G.D., 1985. Numerical Solution of Partial Differential Equations Finite Difference Methods, Oxford, Clarendon Press.

Topçu, A., 2014. Nümerik Analiz Notları. Available: http://mmf2.ogu.edu.tr/atopcu/index_dosyalar/BDNADersNotlari.htm

Uysal, M., 2004. Matlab ile Matematiksel Uygulamalar ve Mühendislik Uygulamaları, Beta Basım Yayın, İstanbul, Türkiye, 682 s.

URL-1, 2019. http://www.infocobuild.com/education/audio-video-courses/mechanical-engineering/ComputationalFluidDynamics-IIT-Roorkee/lecture-19.html (2 Mart 2019).

URL-2, 2018. https://cadsay.com/matlab-nedir-nerelerde-kullanilir (8 Ekim 2018).

URL-3, 2019. http://www.yildiz.edu.tr/~ayten/Algoritma_Prog_Deney4.pdf (2 Mayıs 2019).

URL-4, 2019. https://web.itu.edu.tr/kents/matlab.pdf (23 Nisan 2019).

URL–5, 2012. http://Math.gmu.edu/~pseshaiy/F12/m679/m679_F12_notes_hw1.pdf (7 Mayıs 2014)

URL–6, 2014. http:// mat.iitm.ac.in/home/sryedida/public_html (12 Mayıs 2014)

Yang, Y.W., Cao, W., Chang, S.T. and Morris, J., 2005. Applied Numerical Methods Using MATLAB. John Wiley&Sons, Inc., New :Jersey, ISBN 0-471-69833-4.

Yükselen, M.A., 2008. Kısmi Diferansiyel Denklemlerin Sayısal Çözümü. Available: https://web.itu.edu.tr/yukselen/HM504/05-Kısmidiferansiyeldenklemler.pdf

EKLER

EK-1. Açık Yöntem MatLab Programı

```
function [U,r,dx,dt,k,x]=explicit(L,T,c1,n,r,a1,b1,a2,b2,f,alpha,beta)
%du/dt=d^2/dx^2 ; u(x,0)=f(x) ; a1*u(0,t)+b1*du/dx(0,t)=alpha(t)(central-difference) ;
%a2*u(L,t)+b2*du/dx(L,t)=beta(t)(central-difference)

dx=L/(n-1); % xi değerleri arasındaki uzaklık
dx2=dx*dx;
dt=(r*dx2)/c1;

for i=1:n
   x(i)=(i-1)*dx;  %x(i)leri tanımladık
end
k=round(T/dt); % t boyutunda kaç iterasyon yapılacağını belirledik.

for i=1:n
   u(i)=feval(f,x(i)); %u(x,0)=f(x) başlangıç şartının sayısal ifadesi
end
t=0;
      e = ones(n,1);
      a = spdiags([0*e e 0*e],-1:1, n, n);
      b = spdiags([r*e (1-2*r)*e r*e],-1:1, n, n);
   if (b1==0)&&(b2~=0)
      a(1,1)=a1;
      a(1,2)=0;
      a(n,n)=1/2;
      b(1,1)=0;
      b(1,2)=0;
      b(n,n)=(1/2)-r-r*(a2/b2)*dx;
   elseif(b1~=0)&&(b2==0)
      a(1,1)=1/2;
      a(n,n-1)=0;
      a(n,n)=a2;
      b(1,1)=1/2-r+r*(a1/b1)*dx;
      b(n,n-1)=0;
      b(n,n)=0;
   elseif(b1~=0)&&(b2~=0)
      a(1,1)=1/2;
      a(n,n)=1/2;
      b(1,1)=(1/2)-r+r*(a1/b1)*dx;
      b(n,n)=(1/2)-r-r*(a2/b2)*dx;
   elseif(b1==0)&&(b2==0)
      e = ones(n-2,1);
      a = spdiags([0*e 2*e 0*e],-1:1, n-2, n-2);
      b = spdiags([2*r*e (2-4*r)*e 2*r*e],-1:1, n-2, n-2);
   end
```

EK-1 (Devam). Açık Yöntem MatLab Programı

```
for j=1:k
   t=t+dt;
   if (b1==0)&&(b2~=0)
      alfbet=zeros(1,n);
      alfbet(1)=alpha(t+dt);
      alfbet(n)=r*dx*(beta(t)/b2);
      by=b*u(1:n)'+alfbet';
      uy=a\by;
      u(1:n)=uy(1:n)';
   elseif(b1~=0)&&(b2==0)
      alfbet=zeros(1,n);
      alfbet(1)=-r*dx*(alpha(t)/b1);
      alfbet(n)=beta(t+dt);
      by=b*u(1:n)'+alfbet';
      uy=a\by;
      u(1:n)=uy(1:n)';
   elseif(b1~=0)&&(b2~=0)
      alfbet=zeros(1,n);
      alfbet(1)=-r*dx*(alpha(t)/b1);
      alfbet(n)=r*dx*(beta(t)/b2);
      by=b*u(1:n)'+alfbet';
      uy=a\by;
      u(1:n)=uy(1:n)';
   elseif(b1==0)&&(b2==0)
      bs=b*u(2:n-1)';
      u(2)=((2-4*r)*u(2)+2*r*u(3))+2*r/a1*alpha(t);
      u(n-1)=(2*r*u(n-2)+(2-4*r)*u(n-1))+2*r/a2*beta(t);
      us=u(2:n-1)';
      us(2:n-3)=bs(2:n-3);
      uy=a\us;
      u(2:n-1)=uy(1:n-2)';
      u(1)=alpha(t)/a1;  %u(0,t)=alpha sınır şartı
      u(n)=beta(t)/a2;   %u(l,t)=beta sınır şartı
   end

   U(j,1:n)=u(1:n);

end
```

EK-2. Kapalı Yöntem MatLab Programı

```
function [U,r,dx,dt,k,x]=implicit(L,T,c1,n,r,a1,b1,a2,b2,f,alpha,beta)
%du/dt=d^2u/dx^2 ; u(x,0)=f(x) ; a1*u(0,t)+b1*du/dx(0,t)=alpha(t) ;
a2*u(L,t)+b1*du/dx(L,t)=beta(t)

dx=L/(n-1); % xi değerleri arasındaki uzaklık
dx2=dx*dx;
dt=(r*dx2)/c1;

for i=1:n
   x(i)=(i-1)*dx;  %x(i)leri tanımladık
end

k=round(T/dt); % t boyutunda kaç iterasyon yapılacağını belirledik.

for i=1:n
   u(i)=f(x(i)); %u(x,0)=f(x) başlangıç şartının sayısal ifadesi
end
t=0;
      e = ones(n,1);
      a = spdiags([-r*e (1+2*r)*e -r*e],-1:1, n, n);

   if (b1==0)&&(b2~=0)
      a(1,1)=a1;
      a(1,2)=0;
      a(n,n-1)=-2*r;
      a(n,n)=1+2*r+2*r*(a2/b2)*dx;
   elseif(b1~=0)&&(b2==0)
      a(1,1)=1+2*r-2*r*(a1/b1)*dx;
      a(1,2)=-2*r;
      a(n,n-1)=0;
      a(n,n)=a2;
   elseif(b1~=0)&&(b2~=0)
      a(1,1)=1+2*r-2*r*(a1/b1)*dx;
      a(1,2)=-2*r;
      a(n,n-1)=-2*r;
      a(n,n)=1+2*r+2*r*(a2/b2)*dx;
   elseif(b1==0)&&(b2==0)
      e = ones(n-2,1);
      a = spdiags([-r*e (1+2*r)*e -r*e],-1:1, n-2, n-2);
   end
```

EK-2 (Devam). Kapalı Yöntem MatLab Programı

```
for j=1:k
   t=t+dt;
   if (b1==0)&&(b2~=0)
      alfbet=zeros(1,n);
      alfbet(1)=0;
      alfbet(n)=2*r*dx*beta(t+dt)/b2;
      u(1)=alpha(t+dt);
      by=u(1:n)'+alfbet';
      uy=a\by;
      u(1:n)=uy(1:n)';
   elseif(b1~=0)&&(b2==0)
      alfbet=zeros(1,n);
      alfbet(1)=-2*r*dx*alpha(t+dt)/b1;
      alfbet(n)=0;
      u(n)=beta(t+dt);
      by=u(1:n)'+alfbet';
      uy=a\by;
      u(1:n)=uy(1:n)';
   elseif(b1~=0)&&(b2~=0)
      alfbet=zeros(1,n);
      alfbet(1)=-2*r*dx*alpha(t+dt)/b1;
      alfbet(n)=2*r*dx*beta(t+dt)/b2;
      by=u(1:n)'+alfbet';
      uy=a\by;
      u(1:n)=uy(1:n)';
   elseif(b1==0)&&(b2==0)
      u(2)=u(2)+r/a1*alpha(t);
      u(n-1)=u(n-1)+r/a2*beta(t);
      us=u(2:n-1)';
      uy=a\us;
      u(2:n-1)=uy(1:n-2)';
      u(1)=alpha(t)/a1;   %u(0,t)=alpha sınır şartı
      u(n)=beta(t)/a2;    %u(l,t)=beta sınır şartı
   end

   U(j,1:n)=u(1:n);

end
```

EK-3. Crank-Nicolson Yöntemi Matlab Programı

```
function [U,r,dx,dt,k,x]=crank(L,T,c1,n,r,a1,b1,a2,b2,f,alpha,beta)
%du/dt=d^2u/dx^2 ; u(x,0)=f(x) ; a1*u(0,t)+b1*du/dx(0,t)=alpha(t) ;
a2*u(L,t)+b2*du/dx(L,t)=beta(t)

dx=L/(n-1); % xi değerleri arasındaki uzaklık
dx2=dx*dx;
dt=(r*dx2)/c1;

for i=1:n
   x(i)=(i-1)*dx;  %x(i)leri tanımladık
end

k=round(T/dt); % t boyutunda kaç iterasyon yapılacağını belirledik.

for i=1:n
   u(i)=f(x(i)); %u(x,0)=f(x) başlangıç şartının sayısal ifadesi
end
t=0;
      e = ones(n,1);
      a = spdiags([-r*e (2+2*r)*e -r*e],-1:1, n, n);
      b = spdiags([r*e (2-2*r)*e r*e],-1:1, n, n);

   if (b1==0)&&(b2~=0)
      a(1,1)=a1;
      a(1,2)=0;
      a(n,n)=1+r+r*(a2/b2)*dx;
      b(1,1)=0;
      b(1,2)=0;
      b(n,n)=1-r-r*(a2/b2)*dx;
   elseif(b1~=0)&&(b2==0)
      a(1,1)=1+r-r*(a1/b1)*dx;
      a(n,n-1)=0;
      a(n,n)=a2;
      b(1,1)=1-r+r*(a1/b1)*dx;
      b(n,n-1)=0;
      b(n,n)=0;
   elseif(b1~=0)&&(b2~=0)
      a(1,1)=1+r-r*(a1/b1)*dx;
      a(n,n)=1+r+r*(a2/b2)*dx;
      b(1,1)=1-r+r*(a1/b1)*dx;
      b(n,n)=1-r-r*(a2/b2)*dx;
   elseif(b1==0)&&(b2==0)
      e = ones(n-2,1);
      a = spdiags([-r*e (2+2*r)*e -r*e],-1:1, n-2, n-2);
      b = spdiags([r*e (2-2*r)*e r*e],-1:1, n-2, n-2);
   end
```

EK-3 (Devam). Crank-Nicolson Yöntemi Matlab Programı

```
for j=1:k
   t=t+dt;
   if (b1==0)&&(b2~=0)
      alfbet=zeros(1,n);
      alfbet(1)=alpha(t+dt);
      alfbet(n)=r*dx*((beta(t)+beta(t+dt))/b2);
      by=b*u(1:n)'+alfbet';
      uy=a\by;
      u(1:n)=uy(1:n)';
   elseif(b1~=0)&&(b2==0)
      alfbet=zeros(1,n);
      alfbet(1)=-r*dx*((alpha(t)+alpha(t+dt))/b1);
      alfbet(n)=beta(t+dt);
      by=b*u(1:n)'+alfbet';
      uy=a\by;
      u(1:n)=uy(1:n)';
   elseif(b1~=0)&&(b2~=0)
      alfbet=zeros(1,n);
      alfbet(1)=-r*dx*((alpha(t)+alpha(t+dt))/b1);
      alfbet(n)=r*dx*((beta(t)+beta(t+dt))/b2);
      by=b*u(1:n)'+alfbet';
      uy=a\by;
      u(1:n)=uy(1:n)';
   elseif(b1==0)&&(b2==0)
      bs=b*u(2:n-1)';
      u(2)=(2*(1-r)*u(2)+r*u(3))+r/a1*(alpha(t)+alpha(t+dt));
      u(n-1)=(r*u(n-2)+2*(1-r)*u(n-1))+r/a2*(beta(t)+beta(t+dt));
      us=u(2:n-1)';
      us(2:n-3)=bs(2:n-3);
      uy=a\us;
      u(2:n-1)=uy(1:n-2)';
      u(1)=alpha(t)/a1;  %u(0,t)=alpha sınır şartı
      u(n)=beta(t)/a2;   %u(l,t)=beta sınır şartı
   end

   U(j,1:n)=u(1:n);

end
```

EK-4. Ağırlıklı Ortalama Yöntemi MatLab Programı

```
function [U,r,dx,dt,k,x]=weightedaverage(L,T,c1,n,r,teta,a1,b1,a2,b2,f,alpha,beta)
%du/dt=d^2u/dx^2 ; u(x,0)=f(x) ; a1*u(0,t)+b1*du/dx(0,t)=alpha(t) ;
a2*u(L,t)+b1*du/dx(L,t)=beta(t)

dx=L/(n-1); % xi değerleri arasındaki uzaklık
dx2=dx*dx;
dt=(r*dx2)/c1;

for i=1:n
   x(i)=(i-1)*dx;  %x(i)leri tanımladık
end

k=round(T/dt); % t boyutunda kaç iterasyon yapılacağını belirledik.

for i=1:n
   u(i)=f(x(i)); %u(x,0)=f(x) başlangıç şartının sayısal ifadesi
end
t=0;
      e = ones(n,1);
      a = spdiags([-r*teta*e (1+2*r*teta)*e -r*teta*e],-1:1, n, n);
      b = spdiags([r*(1-teta)*e (1-2*r*(1-teta))*e r*(1-teta)*e],-1:1, n, n);

   if (b1==0)&&(b2~=0)
      a(1,1)=a1;
      a(1,2)=0;
      a(n,n)=(1/2)+r*teta+r*teta*(a2/b2)*dx;
      b(1,1)=0;
      b(1,2)=0;
      b(n,n)=(1/2)-r*(1-teta)-r*(1-teta)*(a2/b2)*dx;
   elseif(b1~=0)&&(b2==0)
      a(1,1)=(1/2)+r*teta-r*teta*(a1/b1)*dx;
      a(n,n-1)=0;
      a(n,n)=a2;
      b(1,1)=(1/2)-r*(1-teta)+r*(1-teta)*(a1/b1)*dx;
      b(n,n-1)=0;
      b(n,n)=0;
   elseif(b1~=0)&&(b2~=0)
      a(1,1)=(1/2)+r*teta-r*teta*(a1/b1)*dx;
      a(n,n)=(1/2)+r*teta+r*teta*(a2/b2)*dx;
      b(1,1)=(1/2)-r*(1-teta)+r*(1-teta)*(a1/b1)*dx;
      b(n,n)=(1/2)-r*(1-teta)-r*(1-teta)*(a2/b2)*dx;
   elseif(b1==0)&&(b2==0)
      e = ones(n-2,1);
      a = spdiags([-2*r*teta*e (2+4*r*teta)*e -2*r*teta*e],-1:1, n-2, n-2);
      b = spdiags([2*r*(1-teta)*e (2-4*r*(1-teta))*e 2*r*(1-teta)*e],-1:1, n-2, n-2);
   end
```

EK-4 (Devam). Ağırlıklı Ortalama Yöntemi MatLab Programı

```
for j=1:k
   t=t+dt;
   if (b1==0)&&(b2~=0)
      alfbet=zeros(1,n);
      alfbet(1)=alpha(t+dt);
      alfbet(n)=r*dx*(((1-teta)*beta(t)+teta*beta(t+dt))/b2);
      by=b*u(1:n)'+alfbet';
      uy=a\by;
      u(1:n)=uy(1:n)';
   elseif(b1~=0)&&(b2==0)
      alfbet=zeros(1,n);
      alfbet(1)=-r*dx*(((1-teta)*alpha(t)+teta*alpha(t+dt))/b1);
      alfbet(n)=beta(t+dt);
      by=b*u(1:n)'+alfbet';
      uy=a\by;
      u(1:n)=uy(1:n)';
   elseif(b1~=0)&&(b2~=0)
      alfbet=zeros(1,n);
      alfbet(1)=-r*dx*(((1-teta)*alpha(t)+teta*alpha(t+dt))/b1);
      alfbet(n)=r*dx*(((1-teta)*beta(t)+teta*beta(t+dt))/b2);
      by=b*u(1:n)'+alfbet';
      uy=a\by;
      u(1:n)=uy(1:n)';
   elseif(b1==0)&&(b2==0)
      bs=b*u(2:n-1)';
      u(2)=((2-4*r*(1-teta))*u(2)+2*r*(1-teta)*u(3))+2*r/a1*((1-
teta)*alpha(t)+teta*alpha(t+dt));
      u(n-1)=(2*r*(1-teta)*u(n-2)+(2-4*r*(1-teta))*u(n-1))+2*r/a2*((1-
teta)*beta(t)+teta*beta(t+dt));
      us=u(2:n-1)';
      us(2:n-3)=bs(2:n-3);
      uy=a\us;
      u(2:n-1)=uy(1:n-2)';
      u(1)=alpha(t)/a1;  %u(0,t)=alpha sınır şartı
      u(n)=beta(t)/a2;   %u(l,t)=beta sınır şartı
   end

   U(j,1:n)=u(1:n);

end
```

EK-5. Dufort Frankel İmplicit Yöntemi MatLab Programı

```
function [U,r,dx,dt,k,x]=Dufort2impkar(L,T,c1,n,r,a1,b1,a2,b2,f,alpha,beta)

dx=L/(n-1); % xi değerleri arasındaki uzaklık
dx2=dx*dx;
dt=(r*dx2)/c1;

for i=1:n
   x(i)=(i-1)*dx;  %x(i)leri tanımladık
end

k=round(T/dt); % t boyutunda kaç iterasyon yapılacağını belirledik.

for i=1:n
   u(i)=f(x(i)); %u(x,0)=f(x) başlangıç şartının sayısal ifadesi
end
t=0;
   e = ones(n,1);
   a = spdiags([-r*e (1+2*r)*e -r*e],-1:1, n, n);

   if (b1==0)&&(b2~=0)
      a(1,1)=a1;
      a(1,2)=0;
      a(n,n-1)=-2*r;
      a(n,n)=1+2*r+2*r*(a2/b2)*dx;
   elseif(b1~=0)&&(b2==0)
      a(1,1)=1+2*r-2*r*(a1/b1)*dx;
      a(1,2)=-2*r;
      a(n,n-1)=0;
      a(n,n)=a2;
   elseif(b1~=0)&&(b2~=0)
      a(1,1)=1+2*r-2*r*(a1/b1)*dx;
      a(1,2)=-2*r;
      a(n,n-1)=-2*r;
      a(n,n)=1+2*r+2*r*(a2/b2)*dx;
   elseif(b1==0)&&(b2==0)
      e = ones(n-2,1);
      a = spdiags([-r*e (1+2*r)*e -r*e],-1:1, n-2, n-2);
   end
for j=1:1
   t=t+dt;
   if (b1==0)&&(b2~=0)
      alfbet=zeros(1,n);
      alfbet(1)=0;
      alfbet(n)=2*r*dx*beta(t+dt)/b2;
      u(1)=alpha(t+dt);
      by=u(1:n)'+alfbet';
      uy=a\by;
      u(1:n)=uy(1:n)';
```

EK-5 (Devam). Dufort Frankel İmplicit Yöntemi MatLab Programı

```
    elseif(b1~=0)&&(b2==0)
        alfbet=zeros(1,n);
        alfbet(1)=-2*r*dx*alpha(t+dt)/b1;
        alfbet(n)=0;
        u(n)=beta(t+dt);
        by=u(1:n)'+alfbet';
        uy=a\by;
        u(1:n)=uy(1:n)';
    elseif(b1~=0)&&(b2~=0)
        alfbet=zeros(1,n);
        alfbet(1)=-2*r*dx*alpha(t+dt)/b1;
        alfbet(n)=2*r*dx*beta(t+dt)/b2;
        by=u(1:n)'+alfbet';
        uy=a\by;
        u(1:n)=uy(1:n)';
    elseif(b1==0)&&(b2==0)
        u(2)=u(2)+r/a1*alpha(t);
        u(n-1)=u(n-1)+r/a2*beta(t);
        us=u(2:n-1)';
        uy=a\us;
        u(2:n-1)=uy(1:n-2)';
        u(1)=alpha(t)/a1;   %u(0,t)=alpha sınır şartı
        u(n)=beta(t)/a2;    %u(l,t)=beta sınır şartı
    end
end
    for i=1:n
        uf1(i)=f(x(i)); %u(x,0)=f(x) başlangıç şartının sayısal ifadesi
    end
    uf2(1:n)=u(1:n);
t=0;
for j=1:k
    t=t+dt;
    %dufort denklemi
    if (b1==0)&&(b2~=0)
        uf3(1)=alpha(t)/a1;
        uf3(n)=((1-2*r)/(1+2*r))*uf1(n)+((4*r)/(1+2*r))*(uf2(n-1)-
(a2/b2)*dx*uf2(n)+(dx/b2)*beta(t));
        uf3(2:n-1) =((1-2*r)/(1+2*r))*uf1(2:n-1)+((2*r)/(1+2*r))*(uf2(1:n-2)+uf2(3:n));
        uf1(1:n)=uf2(1:n);
        uf2(1:n)=uf3(1:n);
    elseif(b1~=0)&&(b2==0)
        uf3(1)=((1-2*r)/(1+2*r))*uf1(1)+((4*r)/(1+2*r))*(uf2(2)+(a1/b1)*dx*uf2(1)-
(dx/b1)*alpha(t));
        uf3(n)=beta(t)/a2;
        uf3(2:n-1) =((1-2*r)/(1+2*r))*uf1(2:n-1)+((2*r)/(1+2*r))*(uf2(1:n-2)+uf2(3:n));
        uf1(1:n)=uf2(1:n);
        uf2(1:n)=uf3(1:n);
```

EK-5 (Devam). Dufort Frankel İmplicit Yöntemi MatLab Programı

```
   elseif(b1~=0)&&(b2~=0)
      uf3(1)=((1-2*r)/(1+2*r))*uf1(1)+((4*r)/(1+2*r))*(uf2(2)+(a1/b1)*dx*uf2(1)-
(dx/b1)*alpha(t));
      uf3(n)=((1-2*r)/(1+2*r))*uf1(n)+((4*r)/(1+2*r))*(uf2(n-1)-
(a2/b2)*dx*uf2(n)+(dx/b2)*beta(t));
      uf3(2:n-1) =((1-2*r)/(1+2*r))*uf1(2:n-1)+((2*r)/(1+2*r))*(uf2(1:n-2)+uf2(3:n));
      uf1(1:n)=uf2(1:n);
      uf2(1:n)=uf3(1:n);
   elseif(b1==0)&&(b2==0)
      uf3(1)=alpha(t)/a1;
      uf3(n)=beta(t)/a2;
      uf3(2:n-1) =((1-2*r)/(1+2*r))*uf1(2:n-1)+((2*r)/(1+2*r))*(uf2(1:n-2)+uf2(3:n));
      uf1(1:n)=uf2(1:n);
      uf2(1:n)=uf3(1:n);
   end

   U(j,1:n)=uf3(1:n);

end
```

EK-6. Dufort Frankel Crank-Nicolson Yöntemi MatLab Programı

```
function [U,r,dx,dt,k,x]=Dufort2crankkar(L,T,c1,n,r,a1,b1,a2,b2,f,alpha,beta)

dx=L/(n-1); % xi değerleri arasındaki uzaklık
dx2=dx*dx;
dt=(r*dx2)/c1;

for i=1:n
   x(i)=(i-1)*dx;  %x(i)leri tanımladık
end

k=round(T/dt); % t boyutunda kaç iterasyon yapılacağını belirledik.

for i=1:n
   u(i)=f(x(i)); %u(x,0)=f(x) başlangıç şartının sayısal ifadesi
end
t=0;
   e = ones(n,1);
   a = spdiags([-r*e (2+2*r)*e -r*e],-1:1, n, n);
   b = spdiags([r*e (2-2*r)*e r*e],-1:1, n, n);

  if (b1==0)&&(b2~=0)
      a(1,1)=a1;
      a(1,2)=0;
      a(n,n)=1+r+r*(a2/b2)*dx;
      b(1,1)=0;
      b(1,2)=0;
      b(n,n)=1-r-r*(a2/b2)*dx;
   elseif(b1~=0)&&(b2==0)
      a(1,1)=1+r-r*(a1/b1)*dx;
      a(n,n-1)=0;
      a(n,n)=a2;
      b(1,1)=1-r+r*(a1/b1)*dx;
      b(n,n-1)=0;
      b(n,n)=0;
   elseif(b1~=0)&&(b2~=0)
      a(1,1)=1+r-r*(a1/b1)*dx;
      a(n,n)=1+r+r*(a2/b2)*dx;
      b(1,1)=1-r+r*(a1/b1)*dx;
      b(n,n)=1-r-r*(a2/b2)*dx;
   elseif(b1==0)&&(b2==0)
      e = ones(n-2,1);
      a = spdiags([-r*e (2+2*r)*e -r*e],-1:1, n-2, n-2);
      b = spdiags([r*e (2-2*r)*e r*e],-1:1, n-2, n-2);
   end
```

EK-6 (Devam). Dufort Frankel Crank-Nicolson Yöntemi MatLab Programı

```
   %dufort denklemi
   if (b1==0)&&(b2~=0)
      uf3(1)=alpha(t)/a1;
      uf3(n)=((1-2*r)/(1+2*r))*uf1(n)+((4*r)/(1+2*r))*(uf2(n-1)-
(a2/b2)*dx*uf2(n)+(dx/b2)*beta(t));
      uf3(2:n-1) =((1-2*r)/(1+2*r))*uf1(2:n-1)+((2*r)/(1+2*r))*(uf2(1:n-2)+uf2(3:n));
      uf1(1:n)=uf2(1:n);
      uf2(1:n)=uf3(1:n);
   elseif(b1~=0)&&(b2==0)
      uf3(1)=((1-2*r)/(1+2*r))*uf1(1)+((4*r)/(1+2*r))*(uf2(2)+(a1/b1)*dx*uf2(1)-
(dx/b1)*alpha(t));
      uf3(n)=beta(t)/a2;
      uf3(2:n-1) =((1-2*r)/(1+2*r))*uf1(2:n-1)+((2*r)/(1+2*r))*(uf2(1:n-2)+uf2(3:n));
      uf1(1:n)=uf2(1:n);
      uf2(1:n)=uf3(1:n);
   elseif(b1~=0)&&(b2~=0)
      uf3(1)=((1-2*r)/(1+2*r))*uf1(1)+((4*r)/(1+2*r))*(uf2(2)+(a1/b1)*dx*uf2(1)-
(dx/b1)*alpha(t));
      uf3(n)=((1-2*r)/(1+2*r))*uf1(n)+((4*r)/(1+2*r))*(uf2(n-1)-
(a2/b2)*dx*uf2(n)+(dx/b2)*beta(t));
      uf3(2:n-1) =((1-2*r)/(1+2*r))*uf1(2:n-1)+((2*r)/(1+2*r))*(uf2(1:n-2)+uf2(3:n));
      uf1(1:n)=uf2(1:n);
      uf2(1:n)=uf3(1:n);
   elseif(b1==0)&&(b2==0)
      uf3(1)=alpha(t)/a1;
      uf3(n)=beta(t)/a2;
      uf3(2:n-1) =((1-2*r)/(1+2*r))*uf1(2:n-1)+((2*r)/(1+2*r))*(uf2(1:n-2)+uf2(3:n));
      uf1(1:n)=uf2(1:n);
      uf2(1:n)=uf3(1:n);
   end

   U(j,1:n)=uf3(1:n);

end
```

EK-6 (Devam). Dufort Frankel Crank-Nicolson Yöntemi MatLab Programı

```
   %dufort denklemi
   if (b1==0)&&(b2~=0)
      uf3(1)=alpha(t)/a1;
      uf3(n)=((1-2*r)/(1+2*r))*uf1(n)+((4*r)/(1+2*r))*(uf2(n-1)-
(a2/b2)*dx*uf2(n)+(dx/b2)*beta(t));
      uf3(2:n-1) =((1-2*r)/(1+2*r))*uf1(2:n-1)+((2*r)/(1+2*r))*(uf2(1:n-2)+uf2(3:n));
      uf1(1:n)=uf2(1:n);
      uf2(1:n)=uf3(1:n);
   elseif(b1~=0)&&(b2==0)
      uf3(1)=((1-2*r)/(1+2*r))*uf1(1)+((4*r)/(1+2*r))*(uf2(2)+(a1/b1)*dx*uf2(1)-
(dx/b1)*alpha(t));
      uf3(n)=beta(t)/a2;
      uf3(2:n-1) =((1-2*r)/(1+2*r))*uf1(2:n-1)+((2*r)/(1+2*r))*(uf2(1:n-2)+uf2(3:n));
      uf1(1:n)=uf2(1:n);
      uf2(1:n)=uf3(1:n);
   elseif(b1~=0)&&(b2~=0)
      uf3(1)=((1-2*r)/(1+2*r))*uf1(1)+((4*r)/(1+2*r))*(uf2(2)+(a1/b1)*dx*uf2(1)-
(dx/b1)*alpha(t));
      uf3(n)=((1-2*r)/(1+2*r))*uf1(n)+((4*r)/(1+2*r))*(uf2(n-1)-
(a2/b2)*dx*uf2(n)+(dx/b2)*beta(t));
      uf3(2:n-1) =((1-2*r)/(1+2*r))*uf1(2:n-1)+((2*r)/(1+2*r))*(uf2(1:n-2)+uf2(3:n));
      uf1(1:n)=uf2(1:n);
      uf2(1:n)=uf3(1:n);
   elseif(b1==0)&&(b2==0)
      uf3(1)=alpha(t)/a1;
      uf3(n)=beta(t)/a2;
      uf3(2:n-1) =((1-2*r)/(1+2*r))*uf1(2:n-1)+((2*r)/(1+2*r))*(uf2(1:n-2)+uf2(3:n));
      uf1(1:n)=uf2(1:n);
      uf2(1:n)=uf3(1:n);
   end

   U(j,1:n)=uf3(1:n);

end
```

ÖZGEÇMİŞ

Gamze ŞAHİN, 12.05.1984 tarihinde Malatya'da doğdu. İlköğretimini 1995 yılında Mersin İbrahim Karaoğlanoğlu İlkokulu'nda ve ortaöğretimini 2002 yılında Malatya Anadolu Lisesi'nde tamamladı. 20.08.2007 tarihinde Kahramanmaraş Sütçü İmam Üniversitesi Fen Edebiyat Fakültesi Matematik Bölümü'nden mezun oldu. 23.06.2010 tarihinde Fırat Üniversitesi/Fen Bilimleri Enstitüsü/Ortaöğretim Fen ve Matematik Alanlar Eğitimi'nde Tezsiz Yüksek Lisans yaptı. 2013 yılında Recep Tayyip Erdoğan Üniversitesi Fen Bilimleri Enstitüsü Matematik Anabilim Dalı'nda başladığı yüksek lisans öğrenimini halen devam ettirmektedir. 12.09.2014 yılı itibariyle T.C. Hazine ve Maliye Bakanlığı Vergi Denetim Kurulu'nda göreve başlamış olup Boğaziçi Küçük ve Orta Ölçekli Mükellefler Grup Başkanlığı'nda Vergi Müfettiş Yardımcısı olarak görev yapmaktadır.