ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE
ENGINEERING AND TECHNOLOGY

HEAT TRANSFER MODES ENHANCEMENT
BY USING MAGNETIC NANOFLUIDS

Ph.D. THESIS

Elgiz BAŞKAYA

Department of Aeronautics and Astronautics Engineering

Aeronautics and Astronautics Engineering Doctorate Programme

MARCH 2018

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE

ENGINEERING AND TECHNOLOGY

HEAT TRANSFER MODES ENHANCEMENT
BY USING MAGNETIC NANOFLUIDS

Ph.D. THESIS

Elgiz BAŞKAYA
(511102101)

Department of Aeronautics and Astronautics Engineering

Aeronautics and Astronautics Engineering Doctorate Programme

Thesis Advisor: Prof. Dr. İbrahim ÖZKOL

MARCH 2018

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

MANYETİK NANOAKIŞKANLARIN KULLANILMASI İLE
ISI TRANSFER MODLARININ İYİLEŞTİRİLMESİ

DOKTORA TEZİ

Elgiz BAŞKAYA
(511102101)

Uçak ve Uzay Mühendisliği Anabilim Dalı

Uçak ve Uzay Mühendisliği Doktora Programı

Tez Danışmanı: Prof. Dr. İbrahim ÖZKOL

MART 2018

Elgiz BAŞKAYA, a Ph.D. student of ITU Graduate School of Science Engineering and Technology 511102101 successfully defended the thesis entitled “HEAT TRANSFER MODES ENHANCEMENT BY USING MAGNETIC NANOFLUIDS”, which he/she prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Prof. Dr. İbrahim ÖZKOL** Istanbul Technical University	....
Jury Members :	**Prof. Dr. Metin Orhan KAYA** Istanbul Technical University	....
	Asc. Prof. Güven KÖMÜRGÖZ KIRIŞ Istanbul Technical University	....
	Prof. Dr. Erol UZAL Istanbul University	....
	Prof. Dr., Colonel Abdurrahman HACIOĞLU Istanbul Gelişim University	....

Date of Submission : 2 February 2018
Date of Defense : 9 March 2018

To my mother Nergiz, father Adıgüzel and brother Uygar,

FOREWORD

This long lived Ph.D. has initiated its roots in formation flying of satellites, had a short affair with aircraft trajectory estimation in presence of shared wind information end hopefully dies in the hands of heat transfer enhancement via utilizing magnetic nanofluids. Along the way, a lot of people has contributed some way or another, but the first one to mention is my advisor for the final station, Prof. Dr. İbrahim ÖZKOL. First of all, I am grateful to have an advisor to back me up, with a fatherly manner, understanding and cheerful. His guidance led me to this point, I can not thank him enough.

And then comes Asc. Prof. Güven KÖMÜRGÖZ KIRIŞ, although not officially my co-advisor, she was much more than that. Her support on this work is may be more than anyone, not even mentioning her friendly attitude..

My family, my mom Nergiz and father Adiguzel, I am grateful to you most, for the most precious gift you gave me, my life. Mom, you are the reason of my freedom, my imagination and my free will. I always admired you being competent since I first saw you at work, passionately talking on your profession, eyes closed, hard to believe that it was my mother.

And dad, you are the reason for my artistic spirit, which tries to burst out of me sometimes. You both always supported me on music, always supported my choices, I was lucky to be grown as a free spirit, I think this is where I am grateful the most.

My brother, you are really special to me. It is enough to make me happy just knowing you are there.

My grandmom, nananem. The cleverest women I have ever known, lucky to be raised by you. I wont forget that you being the person that I get along the best, never even an argument, like magic. My flees to captains bridge has never been punished on the way along the sea to summer house, my never finishing questions never left unanswered, my passion for sea shells and flower seeds plugging the washing machines have been ignored.

My uncle Cengiz, where my logic genes have been inherited. Again, the person I have never argued even once in my lifetime, no delays in communication, or may be sometimes when something else has been processed. The best person I have ever known in my life, an angel. Too clever, too talented. And aunt Leyla, woke me up everyday, patiently, thank you many times.

Rapunzel, my companion, you have lived your most of your life during this thesis. Although you wont be able to read it, you wont be the first one.

Ertan, you have helped me numerous times I stopped counting, my crisis, my anxieties, my vulnerabilities. Those days I was about to quit, you made me back, not let me to leave Rapunzel, told me that I can do it, and I did it.

Ellie Arroway, my hero for AeroAstro, most probably the biggest reason that I am here, although not very logic. one down.. a couple million to go..

Utku, ne 3 u la, my perfect lab mate. Hope to work AI together here or there, one day..

The drone team in ENAC, with whom my last three years passed. Best team ever. Felt like home. Thank you all..

March 2018 Elgiz BAŞKAYA

TABLE OF CONTENTS

Page

ABBREVIATIONS

DQM	**:** Differential Quadrature Method
GDQM	**:** Generalized Differential Quadrature Method
NR	**:** Newton Raphson

SYMBOLS

B_0	magnetic field intensity
c_p	specific heat $[J/K]$
H	channel diameter $[m]$
N	number of grids
g	gravitational acceleration $\left[m/s^2\right]$
k	thermal conductivity $[W/mK]$
α	magnetic field orientation angle$[rad]$
ϕ	inclination angle$[rad]$
α_f	thermal diffusivity $\left[m^2/s\right]$
ψ	solid volume fraction
μ	dynamic viscosiy $\left[m^2/s\right]$
υ	kinematics viscosity $\left[m^2/s\right]$
σ	electrical conductivity
Ha	Hartmann number
Ra	Rayleigh number
Ec	Eckert number
Pr	Prantl number
Br	Brinkman number
u	x component of velocity vector$[m/s]$
U	dimensionless x component of velocity vector
T	temperature $[K]$
θ	dimensionless temperature
x,y	cartesian coordinates $[m]$
X,Y	dimensionless cartesian coordinates
f	fluid
p	nanoparticle
nf	nanofluid

LIST OF TABLES

LIST OF FIGURES

HEAT TRANSFER MODES ENHANCEMENT BY USING MAGNETIC NANOFLUIDS

SUMMARY

To predict and control the nature has always been a human desire. Formerly answering the matters of life and death, probably in time has turned into a survival instinct and sticked to us. Now, not necessarily for urgencies, we have a tendency to rule the nature for our needs. With such a motive, this work discusses the addition of nano-sized particles into conventional fluids to acquire tunable properties and flow. Ferrofluids, a mixture of nonmagnetic fluids with magnetic nano-sized particles, offer advantages such as adjustable thermal conductivity and ability to be controlled via an external magnetic field. Although referred to as not a matter of life and death, ferrofluids have various medical applications which might serve as one soon. An example for promising medical application of nanofluids is cancer treatments, where external magnetic field is utilized to guide special magnetic nanoparticles directly to the malign tumors giving less harm to the surrounding tissues. Their not totally understood superiority for heat conductance also offers effective solutions for a variety of different cooling applications.

This work focuses on investigation of flow characteristics of this new engineered fluids in channel flows. Study here involves numerical calculations rather then experimentation. Thus, efficient tools to solve the system equations plays a vital role. Nanoparticle specific properties of the flow has been simulated by the available emprical models in the literature.

Starting with an introduction of the relatively new topic of nanofluids and a literature survey, an overview of the topic has been aimed. Then, to solve the partial differential equations of motion for the flow, a not widely used but powerful method has been suggested for the solution of the problem. This method is called the Differential Quadrature Method (DQM) and in this work, it proved itself to be a highly efficient and powerful solver when compared to conventional techniques especially for very few grids. To be able to show its effectiveness, a permeable channel problem has been solved via DQM, with no nanoparticles involved for the convenience of literature for comparison. To demonstrate DQM's superiority, a variable viscosity channel flow under constant magnetic field is investigated. Nonlinear, coupled differential equations, representing steady viscous incompressible flow of an electrically conducting fluid between two porous plates under magnetic field, are discretized utilizing Generalized Differential Quadrature Method (GDQM), which is a semi numerical-analytical solution technique and gives serial form solutions, and solved via Newton- Raphson (NR) method. It is shown that, this combination leads to more accurate results compared to available ones in the literature. Furthermore, entropy generation mechanisms due to magnetic field, fluid friction and heat transfer are discussed in detail. An equipartition phenomenon, which corresponds to the degree of freedom of the system, is investigated and it is found that when Ha number

takes greater values, equipartition phenomenon is observed. The study highlights that GDQM is a very strong tool to converge to accurate solutions with a few grid points, enabling more computationally efficient solutions for channel flow problems under magnetic field, which has numerous applications in engineering and industry.

Further on, DQM is utilized for nanofluid flow solutions. Dispersion of superparamagnetic nanoparticles in a nonmagnetic carrier fluid, ferrofluids, offer the advantages of tunable thermophysical properties and eliminate the need for moving parts to induce flow. Following the selection of the efficient numerical tool for solutions, nanofluid flow characteristics of an inclined channel flow exposed to constant magnetic field and pressure gradient is investigated. This part of the study is based on the work done in the author's paper [7]. The nanofluid considered is water based Cu nanoparticles with a volume fraction of 0.06. The viscous dissipation is taken into account in the energy equation and the governing differential equations are nondimensionalized. The coupled one dimensional differential equations are solved via Generalized Differential Quadrature Method (GDQM) discretization followed by Newton-Raphson method. Furthermore, the effect of magnetic field, inclination angle of the channel and volume fraction on nanoparticles in the nanofluid on velocity and temperature profiles are examined and represented by figures to give a thorough understanding of the system behavior. It is observed that an increase in magnetic field density suppresses the flow field significantly, and this effects gets stronger as the volume fraction of nanoparticles in nanofluids increases. The velocity and temperature increases as the inclination of the channel rises but this dependency diminishes with larger magnetic field intensity or volume fraction of nanofluids. The velocity decreases with an increase in volume fraction of nanofluid. When magnetic field gets stronger, the volume fraction dependence of the velocity also increases and gets more dependent to temperature distribution. The temperature distribution is not very effected by volume fraction and this dependence gets even less with increased magnetic field magnitude.

Last part of the thesis investigates ferrofluid flow characteristics in an inclined channel under inclined magnetic field and constant pressure gradient. The ferrofluid considered in this work is comprised of Cu particles as the nanoparticles and water as the base fluid. The governing differential equations including viscous dissipation are non-dimensionalised and discretized with Generalized Differential Quadrature Method. The resulting algebraic set of equations are solved via Newton-Raphson Method. This part of the study is based on the work done in the author's paper [8].The work done here contributes to the literature by searching the effects of magnetic field angle and channel inclination separately on the entropy generation of the ferrofluid filled inclined channel system in order to achieve best design parameter values so called entropy generation minimization is implemented. Furthermore, the effect of magnetic field, inclination angle of the channel and volume fraction of nanoparticles on velocity and temperature profiles are examined and represented by figures to give a thorough understanding of the system behavior.

The effects of magnetic field orientation angle and channel inclination angle are separately investigated, when the channel filled with ferrofluid, on the entropy generation. And for such a channel system, related parameters investigated to produce minimum entropy generation case. The viscous dissipations and buoyancy

effects are included in the governing equations. Derived governing equations are non-dimensionalized by using physically appropriate parameters.

Equations for flow and thermal fields are discretized using GDQM, a new computationally efficient tool giving fairly accurate results for even very few grid points. The discretized system of equations are solved simultaneously utilizing Runge–Kutta scheme. An increase in magnetic field density suppresses the flow field significantly, and these effects get stronger as the volume fraction of nanoparticles in nanofluids increases. The velocity and temperature increases as the inclination of the channel rises but this tendency diminishes with larger magnetic field intensity or volume fraction of nanofluids. The velocity decreases with an increase in volume fraction of nanoparticles. When magnetic field gets stronger, the volume fraction dependence of the velocity also increases and gets more dependent to temperature distribution. Influence of a change in particle volume fraction on temperature distribution is minor and diminished with increased magnetic field magnitude. The entropy generation decreases with increasing magnetic field angle for smaller values of channel inclination. For higher values of channel inclination, with an increase in magnetic field angle, the entropy generation first decreases and then increases. The minimum entropy generation is observed around when the magnetic field angle is perpendicular to the channel. Thus, to further optimize the system by managing velocity and temperature distributions, an multi-objective optimization method should be utilized to serve the users needs.

MANYETİK NANOAKIŞKANLARIN KULLANILMASI İLE ISI TRANSFER MODLARININ İYİLEŞTİRİLMESİ

ÖZET

Doğayı tahmin ve elden geldiğince kontrol etme istegi insanoğlunun her zaman gündeminde olmuştur. Ilk zamanlar ölüm kalım meselesinin bir gereği olarak ortaya çıkan bu güdü zaman içerisinde içselleşmiş ve bizimle bütünleşmiştir. Artık, ölüm kalım meselesi olmayan durumlarda dahi doğayı kendi faydamız doğrultusunda manipüle etmeye eğilimli hale gelmiş durumdayız. Işte benzer bir güdüyle, bu tez de nano-parçacıkları geleneksel akışkanların içerisine dahil ederek akışın belli özelliklerini istegimiz dogrultusunda değiştirme ve akışı yönlendirmek üzere manipüle etmek amacıyla özelliklerini ve davranışlarını anlamak isteğinden yola çıkar.

Ferroakışkanlar, manyetik özelliği olmayan akışkanların içerisine nano boyutta parçacıkların karıştırılmasıyla elde edilir. Bu yeni akışkanın ısıl iletim gibi bazı özellikleri baz akışkanın özelliklerine nazaran çok farklı ve avantajlı olabilir. özellikle ısıl iletkenlik özelliğinde sağladıkları belirgin üstünlük onları popüler bir araştırma konusu haline getirmiştir. Bunların yanı sıra, dışarıdan uygulanan manyetik alan ile akışkanın hız ve sıcaklık gibi parametrelerini etkin bir şekilde değiştirme olanaği sağlarlar. Basta her ne kadar ölüm kalım meselesi gibi görünmese de, nanoakışkanlar kullanılarak yapılan tıp alanındaki son çalışmalar, bu girişimlerin ölüm kalım meselesi çözecek aşamaya getirme yolundadır. Umut vadeden bir tıp uygulaması örneği, tedavi edici manyetik parçacıklarla hazırlanmış nanoakışkanların, dışarıdan uygulanan bir manyetik alan yardımıyla kanserli hücrelere yönlendirilmesinin sağlanmasıyla sorunsuz hücrelerin minimum zarar almasını sağlamak olarak gösterilebilir. Isı iletimindeki iyileştirme becerileri, her ne kadar altında yatan sebepleri tam olarak henuz çözulememiş de olsa, farklı soğutma uygulamalarında kullanılabilir. Bu çalışma kanal içine enjekte edilmiş nanoparçacıların akışa etkisi üzerini nümerik hesaplamalar ile araştırmaktadır. Bu sebeple, denklem sistemlerinin verimli bir şekilde çözülmesi bu çalısmanın önemli bir parşasını oluşturmaktadır. Nanoakışların kendine has özelliklerinin simulasyonunda deneylere dayalı empirik modeller literatürden araştırılarak kullanılmıştır.

Bu tez, görece yeni bir araştırma alanı olan nanoakışkan konseptine bir giriş ve literatür taramasının özetlenmesi ile başlamaktadır. Daha sonra, akışı tasvir eden diferansiyel denklemler tanıtılmış ve boyutsuzlaştırılmıştır. Denklemlerin ccözülmesinde kullanılacak Differential Quadrature Methodunun tanıtılması ile devam eden çalışmada, simülasyonlarla methodun yüksek derecede verimli olduğu gösterilmiştir. özellikle daha az grid kullanılması gereken durumlarda, method, diğer nümerik çözüm yöntemlerine göre üstünlüğünü kanıtlamaktadır. Bunun gösterilebilmesi için, analitik çözümü literatürde var olan geçirgen duvarlı kanal problemi nanoakışkanlar olmaksızın bu yöntem ile çözülmüş ve başka bir method ile kiyaslanmıştır. Değişken viskoziteli sıvı işeren ve dışarıdan etkiyen bir manyetik alan etkisine maruz kalan bu kanal problemini tasvir eden lineer olmayan diferansiyel

denklemler GDQM (Generalized Differential Quadrature method) kullanılarak ayrıklaştırılmış, ve Newton-Raphson methoduyla elde edilmiş olan denklem setleri çözülmüştür. Tezde gösterilmiştir ki, GDQM methodu daha az grid sayısıyla bile diğer methodlara kıyasla üstün cevaplar vermektedir. Daha sonrasında, entropi üretim mekanizmaları olan manyetik alan, akış sürtünmesi ve ısı iletimi tartışılmış ve Ha sayısı yüksek sayılarında equapartitioning fenomeni gözlenmistir. Çalışmanın bu bölümü, GDQM'nin, mühendislik ve endüstride bir çok uygulaması olan manyetik alan etkisi altında kanal probleminin cözümünde, işlem yükü acısından verimli sonuclar verdiğini vurgulamayı amaçlamaktadır.

Çözüm yönteminin secilmesiyle birlikte, kanal problemi geliştirilerek icerisine nanoparcaciklar entegre edilmiştir. Buradaki ana fikir, manyetik parcacikların akışa dahil edilmesiyle değiştirilebilen termo-fiziksel özelliklere sahip akışlanlar, ferroakışkanlar, elde etmekdir. Sabit bir manyetik alan etkisindeki bir eğik kanala enjekte edilen akışın karakteristikleri, GDQM methodu ile cözülerek araştırılmıştır. Bu çalışmada çalışıkan nanoakışlan, Cu nanoparçacıklarının parçacık oranı 0.06 olacak şekilde suya eklenmesiyle elde edilmektedir. Akışın modellenmesinde enerji denkleminde viskos disipasyon goz önüne alınmış ve denklemler boyutlu şekilde sınır koşullarıyla verilmiştir. Daha sonrasında denklemler boyutsuzlaştırılmış, ve adımlar açık bir csekilde okuyucuya sunulmuştur. GDQM ile ayrıklaştırılan denklemler sonucunda elde edilen nonlineer denklem seti, NR metoduyla çözülmüştür. Bunların yanında, manyetik alanın büyüklüğünün, eğik kanalin eğikliğinin, ve parçacık oranının nanoakış hız ve sıcaklık profiline etkisi araştırılmış ve grafiklerle gösterilmiştir. Manyetik alanin büyüklüğundeki artışın akış hızını muhim bir şekilde yavaşlattığı ve nanoakışkan içerisindeki parçacık oranının artmasıyla bu etkinin arttığı gözenmiştir. Kanal eğimindeki artış, akışın hız ve sıcaklığını arttırmış, ancak arttırma miktarı manyetik alanın büyüklüğünün ve nanoparçacık oranının artmasıyla azalmıştır. Nanoparcacık oranının artması akışın hızını düşürmektedir. Manyetik alanın büyüklüğü arttıkça, hızın parçacık oranına bağlılığı da artmaktadır. Sıcaklık dağılımı nanoparcacık oranına fazla bağımlı değildir ve yüksek manyetik alan etkisinde bu bağlantının daha da azaltığı gözlenmiştir.

Tezin son bölümünde, eğik kanal problemine, açısı ve büyüklüğü değişen bir manyetik alan etkidiğinde ferroakışkanın davranışları incelenmektedir. Problemi tasvir eden boyutlu nonlineer diferansiyel denklemler verilmiştir. Boyutsuzlaştırma detayları, bir önceki probleme benzerliği sebebiyle detaylarıyla açıklanmaya gerek görülmemiş, boyutsuz denklemler direkt olarak sunulmuştur. Boyutsuz denklemler ve sınır koşulları GDQM ile ayrıklaştırılmış ve elde edilen nonlineer denklem seti NR methodu ile çözülmüştür. Buradaki çalışmanın literatüre katkısı, manyetik alan açısı ve kanal eğiminin ayrı ayrı değiştirilerek entropi üretimine etkilerinin incelenmesidir. Entropi üretiminin incenmesi optimal sistemlerin tasarlanması icin kullanilan bir yöntemdir. Entropi üretimi minimizasyonu, ferroakışkan içeren eğik kanala uygulanmış ve değişken manyetik alan etkisinin etkileri incelenmiştir. Daha önceki problemlerde olduğu gibi, manyetik alanın büyüklüğünün, bu probleme özgün olarak manyetik alan acısının ve nanoparcacık oranının hız ve sıcaklık profillerine etkisi detaylı olarak incelenmiştir. Manyetik alan açısı ve kanal eğim açısının entropi üretimi üzerine etkileri ayrı ayrı araştırılmıştır. Incelemeler sonunda görülen odur ki, manyetik alanın büyüklüğündeki artış akış hızı yavaşlatmakta ve bu etki nanoparcacıkların oranının artmasıyla artmaktadır. Hız ve sıcaklık kanal boyunca kanalın eğikliğinin yükselmesiyle artmakta ancak bu artış daha yüksek manyetik alan büyüklüklerinde

ve nanoparcacık oranlarında azalmaktadır. Manyetik alanın büyüklüğü arttıkca, hızın parçacık oranına bağlılığı da artmaktadır. Parçacık oranınındaki değişimin sıcaklık alanına etkisi yok sayılacak kadar azdır ve artan manyetik alan büyüklerinde daha da azalmaktadır. Kanal eğiminin düşük değerleri için, manyetik alan büyüklüğünün artmasıyla entropi üretimi azalmaktadır. Kanal eğiminin yüksek acılarında, manyetik alan büyüklüğünün artması entropi üretimini önce düşürmekte ancak belli bir değerden sonra arttırmaktadır. Minimum entropi üretimi manyetik alanin kanala dik olduğu durumlarda gözlenmiştir.

1. INTRODUCTION

1.1 Introduction to Nanofluids

The fast pace of microprocessor technology, the invention of higher performance smaller processors, have risen a new challenge to cope with: thermal management. Cooling plays a vital role for many systems, not only electronic designs due to the lack of space in todays tiny components but also from efficiency perspective as in automative systems such as fuel efficiency thanks to the smaller heat exchangers by increasing heat transfer enabling lighter automobile designs.

Heat flow is governed by three main variables; heat transfer coefficient, heat transfer area and temperature difference. Increasing any of them or some of them simultaneously results in an increase in the heat transfer. Increasing the heat transfer difference is mostly possible by reducing the temperature of the coolant since increasing the temperature of the system is controversial to the fact that the opposite, reducing temperature of the system, is the end goal. Most of the times a temperature decrease in the coolant is not very efficient to go through, yet used in many applications. Another approach is to maximize the heat transfer area as seen in heat exchangers but is mostly constrained by todays attitude toward miniaturization such as in microprocessors and microelectromechanical systems (MEMS) where the heat transfer area is strictly constrained or aerospace systems where it might increase the weight which is substantially undesired for this specific domain [9]. Increasing heat transfer coefficient to enhance heat transfer is another viable option, which can be achieved either via efficient heat transfer methods or use of materials with superior properties, such as forced convection or adding some particles to the conventional heat transfer fluids to change their inherently poor characteristics.

The idea to add particles originates form the fact that metals in solid form have orders-of-magnitude higher thermal conductivities than those of fluids Fig. 1.2. To give an idea, thermal conductivity of copper at room temperature is about 700 times

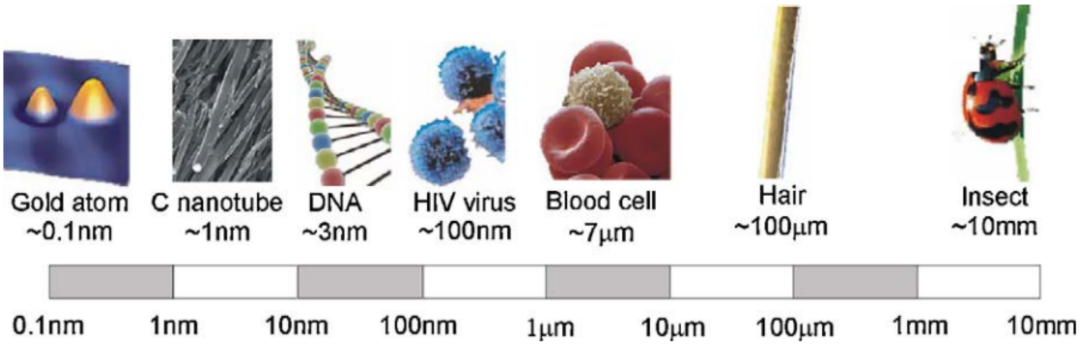

Figure 1.1 : Nano length scale examples [1].

greater than that of water and about 3000 times greater than that of engine oil. Also, the thermal conductivity of metallic liquid score over than that of non-metallic liquids.

This supports the fact that solid metallic particle suspension would have higher thermal conductivities compared to conventional heat transfer fluids. This drove the researches to experiment on this idea especially on micro and larger sized particles since smaller ones were not yet readily available due to not yet delivered technologies. Originally, this idea has ancient roots, going back to Maxwell who worked on the topic more than 100 years ago. But then, solid particles were settling out of suspension fast since they were large in size and dense. Thus the lack of stability prevented their practical use in the real world problems.

The evolved technology made it possible to design and produce particles of sizes much smaller as in Fig. 1.1. Nanotechnology became a popular topic since then, with its applications on a wide variety of topics ranging from medicine to cosmetics. The nano world is really fascinating, the materials of that size possess unique chemical and physical properties.

Nano-sized particle suspensions are called nanofluids. Use of nanofluids has appeared as an innovative way to enhance heat transfer via suspensions of metallic nano-sized particles in conventional heat transfer fluids.

The technology advanced to a level where production of materials with an average crystallite size below 50 nm is possible. The term nanofluids, fluids with suspended nanoparticles, coined by Choi in 1995 in Argonne National Laboratory. Those next-generation heat transfer fluids offer exciting advantages over the conventional

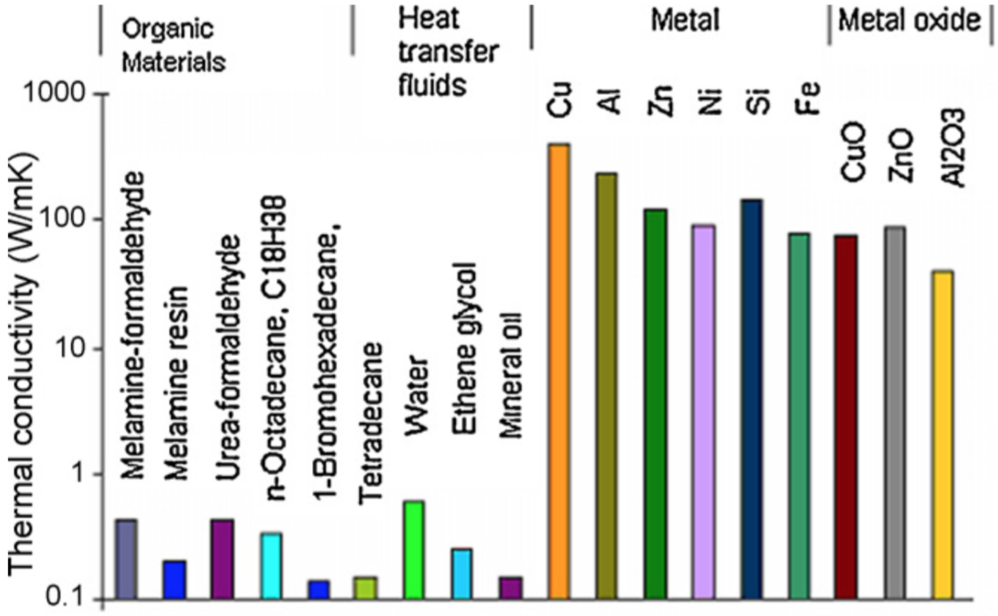

Figure 1.2 : Thermal conductivities of common liquids, polymers and solids [2].

heat transfer fluids and also over the suspensions of micro-sized metallic particles. Increased surface area of the nanofluids results in enhanced heat transfer capabilities and the stability of the suspension is also improved. Aside those advantages about heat transfer, they are found to be better in terms of abrasion related properties. So now it seems to be obvious that clever use of those engineered fluids results in better designs such as smaller and light-weight heat exchanger systems. But literature seems to be not agreed upon the abilities of those fluids since there is a lack of agreement between the results of experiments and deserted theoretical understanding behind the mechanisms which govern nanofluids.

The change in the size of the particles may result in different forces to dominate on the system and scaling theory is a tool to understand the most influential ones. As the size of the particles approach to nano scale, the effect of particle Brownian motion on the heat and mass transfer increases. The interaction force between the particles also becomes more evident compared to macro-sized particles where those are mostly neglected. So to deliver good designs, it is vital to identify the dominant mechanisms corresponding to the size scale of the system.

Some of the advantages of nanofluids can be listed but not limited to [10]:

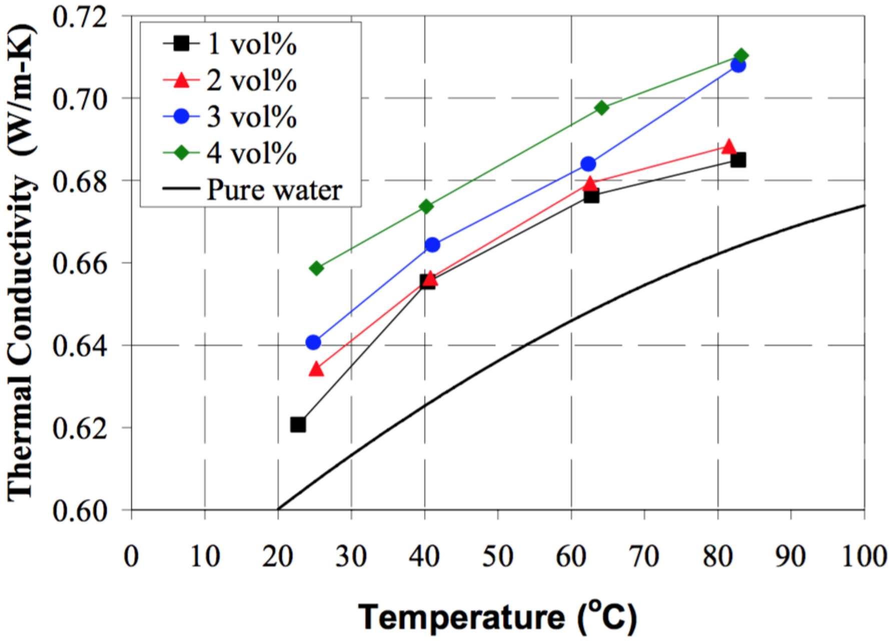

Figure 1.3 : Al_2O_3 nanofluid temperature dependency on thermal conductivity [3].

- Increased specific surface area which results in increased heat transfer surface in-between particles and fluids
- Increased stability in dispersion thanks to dominant Brownian motion
- Reduced pumping power in comparison to pure liquids to achieve same heat transfer intensification task
- Reduced particle clogging
- Adjustable properties via changing particle concentrations as in Fig. 1.4 and sometimes magnetic field and/or temperature as in Fig. 1.3

1.2 Literature Review

Currently, the cooling needs of cutting edge technologies pose a challenge to existing cooling fluids since they are actually poor conductors of heat. The chance of designing an enhanced, more conductive cooling fluid led some researchers to discover the strange world of the nano size. Nanofluids, engineered colloidal suspensions of nanoparticles (typically less than 100 nm) in a base fluid, usually conventional cooling fluids, seem to be a new key to hurdle with the thermal bottleneck for

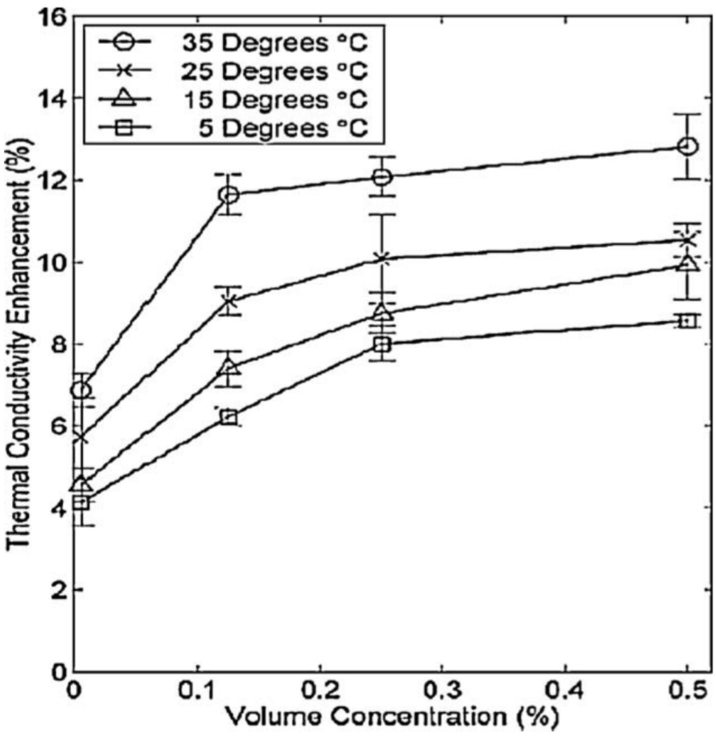

Figure 1.4 : Nanofluid thermal conductivity enhancement dependency on particle volume concentration in different temperatures [4].

various applications. Thus the research guided to these tiny particles as efficient tools to cope up with the thermal needs of not only small in size applications such as microelectromechanical systems (MEMS), but also giant processes such as nuclear reactors. The underlying mechanisms of the novel properties concerning nanofluids are still a mystery from the scientific point of view. That is the reason for many researchers being amazed by the capability achieved by just adding very tiny particles, typically made of metals, oxides, carbides or carbon nanotubes into conventional cooling fluids. New models proposed, many experiments initiated to understand the new phenomenon.

The idea to utilize particles to enhance convective heat transfer and thermal conductivity goes back to Maxwell [11], when he used micro particles within a base fluid. Back then, the method did not work well due to constraints mostly inherent to the size of the particles and the experiments resulted unsatisfactory due to clogging, erosion, rapid sedimentation and high-pressure drop. Advancement of technology enabled to explore the world of tinier particles. Coined the name "nanofluids" by Choi [10], this innovative engineering fluid gained popularity with same author's work showing an evident increase in thermal conductivity is achieved by using nanofluids instead of a conventional heat transfer fluid. Research community immediately directed to this newer topic since heat transfer is a game changer in most of the engineering designs and various work dedicated to show superiority of their thermal conductivity [12–16]. Investigations on not only the heat transfer properties but also tribology shows nanoparticle addition to lubricating oils improves the load-carrying, and friction reduction features [17, 18]. Eastman et al. [19] showed that the thermal conductivity increased up to 40% compared to its basefluid by using copper nanoparticles in ethylene glycol with a solid volume ratio of 0.3%. Among the studies conducted, heat transfer deterioration has also been observed [20–22]. Various geometries have been investigated such as trapezoidal cavities [23], porous cavities [24], and channels [7].

Nanofluid flow has been also deeply studied as a function of different parameters such as volume fraction and nanoparticle size for a variety of models offered [25]. But the applications are not yet constrained by the engineering community, the interest of the medical domain reached at a level not to be underestimated. Drug targeting is an example of promising medical applications of nanofluids for cancer treatment [26]. Injected magnetic nanoparticles with chemotherapeutic agents could be directed to the tumour in order to cause less damage to the surrounding tissues with the utilization of an external magnetic field [27–31]. Enlarging sets of applications of these magnetic nanofluids caused them to be given an alies as ferrofluids. The ability to control the flow via an external magnetic field also raises the questions to have an optimal flow by decreasing the entropy generation. Such an idea drives to more research on entropy generation in the presence of nanofuids [32].

Magnetohydrodynamics (MHD), study of electrically conducting fluids dynamics, has called attention due to its numerous applications on engineering and industry. The idea in MHD applications is to induce currents in an electrically conducting fluid to create a force, utilizing magnetic field. The phenomenon involves the coupled dynamics of electrically conducting fluid's velocity field and present magnetic field. In mathematical sense, both the Navier-Stokes equations of fluid dynamics and Maxwell equations of electromagnetism should be solved simultaneously. This research area finds itself a great role in not only giant scenes appears in astrophysics but also teeny tiny world of atoms.

The idea of an interaction between an electrically conducting fluid and magnetic field first appear in Michael Faraday's mind, when he tried to detect tides in the London's salty river Thames due to electromagnetic induction in 1837 [33]. He tried to measure the potential difference between two river banks but failed because of the inadequacy of the equipment available at that time [34]. Some researchers argue that this was not an MHD experiment since dynamical effect of the field on motion is not included [35]. In 1851, Dr. William Hyde Wollaston was able to measure the voltage induced by the tide in the English Channel [36]. Hannes Alfven was the first one to use of the word 'magneto-hydrodynamics' in his work as "As the term 'electromagnetic-hydrodynamic waves' is somewhat complicated, it mat be convenient to call the phenomenon 'magneto-hydrodynamic' waves." [37]. He is called as "the father of the modern discipline of classical physics known as hydromagnetism or magnetohydrodynamics" [38], for which he was awarded the Nobel Prize for Physics in 1970 for "the fundamental work and discoveries in magnetohyrodynamics with fruitful applications in different parts of plasma physics" [39].

An understanding about the effect of magnetic field on flow fields is essential in many applications such as MHD power generator and boundary layer flow control [40]. MHD became an active area of research with the invention of MHD pump by Northrup in 1907 [41, 42]. Especially when associated with heat transfer applications, MHD applies to wide range of research areas such as cooling of nuclear reactors and solar technology [43]. Electronics may be another area to count for the effects of MHD, since magnetic field influences the electrically conducting fluid in microelectronic heat devices during the production and service stages [44–46]. Considering these effects may enable more effective designs [47]. Drug targeting is another promising application of MHD for cancer treatment [26]. The injected magnetic nanoparticles with chemotherapeutic agents could be directed via magnetic field to the tumour, causing less damage to the surrounding tissues [27–31]. Besides, MHD propulsion is an active area of research. YAMATO 1, a ship built by Mitshubishi group, has an electromagnetohydrodynamic propulsion system [48]. MHD is even enabling science fiction come true. Subrata Roy, associate professor in Mechanical and Aerospace Engineering of University of Florida, submitted a patent in various countries, for the concept of a wingless hovering micro air vehicle (WHOMAV) [49], which reminds an UFO. Several studies conducted for various geometries and each modes of heat transfer [50]. The effect of magnetic field on natural convection flow and heat transfer in cavities are discussed in literature [51–55]. MHD flow over rotating bodies is also another interesting field due to its application areas such as estimating the flight path of spin stabilized missiles, microelectronic devices, liquid cooling of nuclear reactors [56–59].

The flow through circular cross section pipes held special attraction due to its common use in daily life. Independently, Hagen [60] and Poiseuille [61] achieved to obtain exact solution for a steady viscous hydrodynamic laminar axial flow in a pipe [62, 63]. MHD channel flow is another common research area due to its various applications on MHD generators, geothermal reservoirs, cooling of nuclear reactors, petroleum reservoirs, accelerators, pumps, flow meter, astrophysics, metallurgy, crystal growth, magnetic filtration and separation, jet printers, microfluidic devices [64–66]. In 1937, Hartmann and Lazarus conducted experiments to investigate the influence of homogenous magnetic field on the flow of mercury in circular or rectangular cross section pipes [67]. Lehnert investigated the behaviour of electrically conducting fluid under magnetic field theoretically [68]. Flow formation in Couette motion in magnetohydrodynmaics is presented by M. Katagiri et. all [69]. Then, after approximately a decade, unsteady Couette flow in hydromagnetics is studied by M. Balaram et all [70]. Makinde et all. presented the combined effect of transverse magnetic field and radiative heat transfer on unsteady flow of a conducting optically thin fluid through a channel filled with saturated porous medium and non-uniform wall temperature [71]. Seth et all. investigated unsteady MHD Couette flow of a viscous incompressible electrically conducting fluid between two parallel porous plates in the presence of a transverse magnetic field [72].

Designing systems optimally, drive the search for a measure of destruction of system's available work. Entropy generation can be imagined as a measure for the irreversibility associated to the processes. The idea of entropy generation goes back to 1824, when Carnot recognized the importance of avoiding irreversible processes since entropy is produced as a result [73]. Clasius was the one to introduce the term 'entropy', and also gave a mathematical expression for entropy production [74, 75]. Bejan presented the idea of entropy generation minimization in order to identify the factors responsible for the loss of available work of the system [5, 76–80]. Since then, numerous studies have been conducted on entropy generation minimization to utilize energy efficiently under various flow configurations [81–84]. Analytical entropy generation analysis for modelling and optimization of magnetohydrodynamic induction devices is investigated by Salas et al [85]. Mahmud et al. [86] studied thermodynamics analysis of mixed convection in a channel with transverse hydromagnetic effect. Chauhan and Kumar conducted a study on the heat transfer and entropy generation during compressible fluid flow in a channel partially filled with porous medium [87]. Entropy generation in a porous channel with hydromagnetic effects is investigated by Tasnim et al. [88]. Eegunjobi and Makinde [89] analysed the combined effect of buoyancy force and Navier slip on entropy generation in a vertical porous channel. Jery et al. presented the effect of an external oriented magnetic field on entropy generation in natural convection [90]. The incompressible viscous laminar flow through a channel filled with porous media is studied by Dwivedi et all [91]. Numerical investigation of buoyancy effects on hydromagnetic unsteady flow through a porous channel with suction/injection is conducted by Makinde and Chinyoka [92]. The literature on entropy generation minimization in nanofluid flows is exremely limited [93–97].

More recently, the field of entropy generation minimization welcomes a newer approach, equipartition, introduced by Tondeur and Kvaalen [98]. The argument is that, in a process, total entropy produced is minimal when the local rate of entropy production is uniformly distributed along space and/or time variables. This idea opened a new area of interest for various researchers passioned for the topic of optimal processes [99, 100].

1.3 Purpose of Thesis

Exhibiting both magnetic and fluid properties, magnetic nanofluids (ferrofluids) constitute a special class, with the flexibility to be controlled by an external magnetic field. Magnetic force could be utilized not only to control the flow of the ferrofluid but also its properties. Preparation of magnetic nanofluids involves the dispersion of nanoscale superparamagnetic particles into a nonmagnetic carrier liquid such as water, ethylene glycol, hydrocarbon oil etc. Nowadays, enhancement of heat transfer in natural convection flow by utilizing nanofluids in the presence of magnetic field is a popular science problem. Currently, the cooling needs of cutting edge technologies pose a challenge to existing cooling fluids since they are actually poor conductors of heat. The chance of designing an enhanced, more conductive cooling fluid led some researchers to discover the strange world of the nano size. Nanofluids, engineered colloidal suspensions of nanoparticles (typically less than 100 nm) in a base fluid, usually conventional cooling fluids, seem to be a new key to hurdle with the thermal bottleneck for various applications. Thus the research guided to these tiny particles as efficient tools to cope up with the thermal needs of not only small in size applications such as MEMS, but also giant processes such as nuclear reactors. The underlying mechanisms of the novel properties concerning nanofluids are still a mystery from the scientific point of view. That is the reason for many researchers being amazed by the capability achieved by just adding very tiny particles, typically made of metals, oxides, carbides or carbon nanotubes into conventional cooling fluids. To align experimental results with the models available for almost 1.5 century, the researchers fail. New models proposed, many experiments initiated to understand the new phenomenon. To make use of this concept cleverly, the underlying principles must be examined thoroughly.

This work focuses on investigation of flow characteristics of this new engineered fluids in channel flows. Study here involves numerical calculations rather then experimentation. Thus, efficient tools to solve the system equations plays a vital role. Nanoparticle specific properties of the flow has been simulated by the available emprical models in the literature.

First, steady viscous incompressible flow of an electrically conducting fluid having variable viscosity bounded by two infinite horizontal permeable parallel plates is investigated. The system equations are solved numerically utilizing GDQM & NR methods for different parameters of the system. The velocity, temperature, local entropy generation, total entropy generation, and equipartition are investigated for wide range of parameters such as Ha number, V*, a function of viscosity variation parameter and then results discussed via numerous figures. Then, the performance of GDQM method using different gridding techniques and for different grid numbers are examined in relation to the existing work using Runge-Kutta (RK) method [65].

In this work, magnetic nanoparticles' effects on flow are studied with regard to changing magnetic field applied from outside the flow. Exhibiting both magnetic and fluid properties, magnetic nanofluids (ferrofluids) constitute a special class, with the flexibility to be controlled by an external magnetic field. By merging the science on nanofluids such as available models describing nanofluid properties and optimal system design with grounds on entropy generation minimization, effects magnetic field change on inclined channel flow has been studied. The magnetic field angle and channel inclination is handled separately, considering the cases for fixed inlined channels. To solve the equation set, a semi numerical tool, Generalized Differential Quadrature Method (GDQM), is used for discretization for its advantages such as computationally efficiency. GDQM is a numerical technique for solving differential equations by approximating the derivative of a function at any location by a linear summation of all the functional values along a mesh line. Then the resulting sets of algebraic equations are solved with Newton-Raphson (NR) method. The effect of magnetic field, and nanofluid variable properties on heat transfer enhancement and rheology are examined and represented by various figures to give a thorough understanding of the system efficiency behavior. This work contributes to the literature by showing the effects of magnetic field angle and channel inclination separately on the entropy generation of the ferrofluid filled inclined channel system to achieve an optimal design.

2. MATHEMATICAL MODEL

2.1 Channel Flow

In this study, steady viscous incompressible flow of an electrically conducting fluid possessing variable viscosity bounded by two infinite horizontal permeable parallel plates with a distance h in between is considered Fig. 2.1. A constant magnetic field is uniformly applied perpendicular to the plates along the y – axis. The plates are held at constant temperature. The uniform injection and suction through the lower and upper plates are kept at a constant velocity V along the y direction. The assumptions carried through the study can be summarized as;

- The problem is one-dimensional since the plates are assumed to be infinitely long
- The induced magnetic field is neglected when compared with the applied magnetic field, due to inherently small magnetic Reynolds number for magnetic liquids and partially ionized fluids [101]
- Electric field and Hall effects, the ion-slip and thermoelectric effects, and the electron pressure gradient are all neglected [72]
- The flow is assumed to be fully developed and the edge effects are neglected.

These assumptions enable to write the momentum equation as in (2.1) for the system given in Fig. 2.1

$$V\frac{du}{dy} = -\frac{1}{\rho}\frac{\partial p}{\partial x} + \frac{1}{\rho}\frac{d}{dy}\left(\overline{\mu}\left(T\right)\frac{du}{dy}\right) + \frac{B_0}{\rho}J_z \tag{2.1}$$

where u is the fluid velocity in x direction, ρ is the fluid density, p is the fluid pressure, V is the uniform suction/injection velocity at the channel walls and J_z is the z dimension of current density vector, $\overrightarrow{J}$ given by the Ohm's Law given in (2.2)

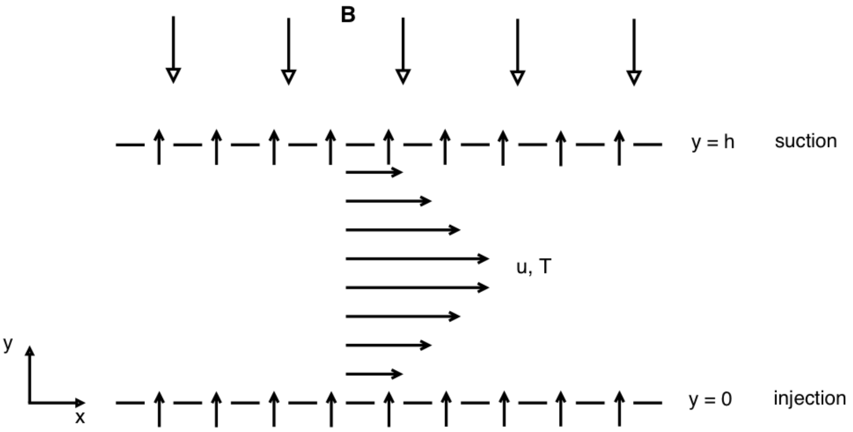

Figure 2.1 : System Schematic.

$$\overrightarrow{J} = \sigma \left[\overrightarrow{E} + \overrightarrow{q} \times \overrightarrow{B} \right] \tag{2.2}$$

In (2.2), $\overrightarrow{E}$, $\overrightarrow{q}$, σ represent the electric field vector, the velocity vector and the electrical conductivity of the fluid respectively.

The energy equation can be written as in (2.3)

$$V \frac{dT}{dy} = \frac{k}{\rho c_p} \frac{d^2 T}{dy^2} + \frac{\overline{\mu}(T)}{\rho c_p} \left(\frac{du}{dy} \right)^2 + \frac{{J_z}^2}{\sigma \rho c_p} \tag{2.3}$$

where $\overline{\mu}$ is the viscosity coefficient, k is the thermal conductivity coefficient, c_p is the specific heat at constant pressure, T is the fluid temperature. Considering the motion is steady $\Delta \times \overrightarrow{E} = 0$, results in

$$\frac{dE_x}{dy} = 0 \qquad\qquad \frac{dE_x}{dy} = 0 \tag{2.4}$$

Since E_x and E_z are constants, assuming $E_x = E_z = 0$ yields

$$J_x = 0 \qquad\qquad J_z = -\sigma u B_0 \tag{2.5}$$

Evaluating these relations in (2.1)-(2.3), gives the governing equations as in (2.6)-(2.7) [65, 102, 103]

$$V \frac{du}{dy} = -\frac{1}{\rho} \frac{\partial p}{\partial x} + \frac{1}{\rho} \frac{d}{dy} \left(\overline{\mu}(T) \frac{du}{dy} \right) - \frac{\sigma {B_0}^2 u}{\rho} \tag{2.6}$$

$$V \frac{dT}{dy} = \alpha \frac{d^2 T}{dy^2} + \frac{\overline{\mu}(T)}{\rho c_p} \left(\frac{du}{dy} \right)^2 + \frac{\sigma {B_0}^2 u^2}{\rho c_p} \tag{2.7}$$

where viscosity is given in (2.8)

$$\bar{\mu}(T) = \mu_0 e^{-m(T-T_1)} \tag{2.8}$$

m is viscosity variation parameter, μ_0 is the fluid dynamic viscosity at the ambient temperature. The boundary conditions are given in (2.9)-(2.10)

$$u(0) = 0 \qquad -k\frac{dT}{dy}(0) = \gamma_0 (T_1 - T(0)) \tag{2.9}$$

$$u(h) = 0 \qquad -k\frac{dT}{dy}(h) = \gamma_1 (T(h) - T_2) \tag{2.10}$$

where γ_0 and γ_1 are the heat transfer coefficients at the lower and upper plates respectively. T_1 is the temperature of the hot fluid at the lower permeable plate and T_2 is the ambient temperature above the upper plate.

Introducing the non-dimensional variables

$$Y = \frac{y}{h} \qquad X = \frac{x}{h} \qquad U = \frac{u}{U_\infty} \qquad \theta = \frac{T - T_2}{T_1 - T_2} \tag{2.11}$$

$$\mu = \frac{\overline{\mu}}{\mu_0} \qquad P = \frac{ph}{\mu_0 U_\infty} \qquad G = -\frac{\partial P}{\partial X} \qquad V^* = \frac{V}{U_\infty} \tag{2.12}$$

Although the non-dimensional variables given above is written in advance, it is not a one step approach and the non-dimensionalization is held flawlessly after defining them. It is an iterative process and for some people is even art, since knowledge of the system in question is required. These are the resulted non-dimensionalized variables which enable the non-dimensionalization as explained below. To explain the non-dimensionalization (2.6) and (2.7) will be handled one by one. First calculate the derivatives for the dimensionalized temperature T to substitute with the non-dimensionalized temperature θ in (2.7).

$$d\theta = \frac{dT}{T_1 - T_2} \tag{2.13}$$

$$dY = \frac{dy}{h} \tag{2.14}$$

And dividing and rearranging yields,

$$\frac{dT}{dy} = \frac{(T_1 - T_2)}{h}\frac{d\theta}{dY} \tag{2.15}$$

And the second derivative

$$\begin{aligned}\frac{d^2T}{dy^2} &= \frac{d}{dy}\left(\frac{dT}{dy}\right) = \frac{1}{h}\frac{d}{dY}\left(\frac{T_1-T_2}{h}\cdot\frac{d\theta}{dY}\right)\\ &= \frac{T_1-T_2}{h^2}\frac{d^2\theta}{dY^2}\end{aligned} \tag{2.16}$$

Likewise for the derivative of u;

$$\frac{du}{dy} = \frac{U_\infty}{h}\frac{dU}{dY} \tag{2.17}$$

(2.15), (2.16) and (2.17) substituted in (2.7) yields

$$V\frac{T_1-T_2}{h}\frac{d\theta}{dY} = \alpha\frac{T_1-T_2}{h^2}\frac{d^2\theta}{dY^2} + \frac{\overline{\mu}(T)}{\rho c_p}\left(\frac{U_\infty}{h}\frac{dU}{dY}\right)^2 + \frac{\sigma B_0^2 U^2 U_\infty^2}{\rho c_p} \tag{2.18}$$

Multiplying the equation (2.18) with $\frac{h^2}{\alpha(T_1-T_2)}$ and using the viscosity definition given in (2.8),

$$\begin{aligned}\frac{h^2}{\alpha(T_1-T_2)}\frac{V(T_1-T_2)}{h}\frac{d\theta}{dY} &= \frac{d^2\theta}{dY^2} + \frac{h^2}{\alpha(T_1-T_2)}\frac{\mu_0 e^{-m(T-T_1)}}{\rho c_p}\frac{U_\infty^2}{h^2}\left(\frac{dU}{dY}\right)^2\\ &+ \frac{h^2}{\alpha(T_1-T_2)}\frac{\sigma B_0^2 U_\infty^2}{\rho c_p}U^2\end{aligned} \tag{2.19}$$

Introducing Eckart number as in (2.20);

$$Ec = \frac{U_\infty^2}{c_p(T_1-T_2)} \tag{2.20}$$

and rearranging terms and multiply and divide the last term by μ_0 leads to (2.21).

$$\frac{d^2\theta}{dY^2} - \frac{hV}{\alpha}\cdot\frac{d\theta}{dY} + \frac{\mu_0 e^{-m(T-T_1)}}{\alpha\rho}\cdot Ec\cdot\left(\frac{dU}{dY}\right)^2 + \frac{B_0^2 h^2\sigma}{\alpha\rho}\cdot\frac{\mu_0}{\mu_0}\cdot Ec\cdot U^2 = 0 \tag{2.21}$$

Next step is to define Ha number as in Eq. (2.22).

$$Ha = B_0 h\sqrt{\frac{\sigma}{\mu_0}} \tag{2.22}$$

Substituting Ha number in (2.21) and manipulating the second term by multiplying and dividing by $U_\infty \cdot V \cdot \alpha \cdot \upsilon$

$$\begin{aligned}\frac{d^2\theta}{dY^2} - \frac{hV}{\alpha} \cdot \frac{U_\infty}{U_\infty} \cdot \frac{V}{V} \cdot \frac{\alpha}{\upsilon} \cdot \frac{\upsilon}{\alpha} \cdot \frac{d\theta}{dY} + \frac{\mu_0 e^{-m(T-T_1)}}{\alpha\rho} \cdot Ec \cdot \left(\frac{dU}{dY}\right)^2 \\ + \frac{Ha^2\mu_0}{\alpha\rho} \cdot Ec \cdot U^2 = 0\end{aligned} \tag{2.23}$$

Introducing Re and Pr number

$$Re = \frac{U_\infty h}{\upsilon} \tag{2.24}$$

$$Pr = \frac{\upsilon}{\alpha} \tag{2.25}$$

and evaluate in (2.23) gives (2.26).

$$\frac{d^2\theta}{dY^2} - Re \cdot Pr \cdot \frac{V}{U_\infty} \cdot \frac{d\theta}{dY} + \frac{\mu_0 e^{-m(T-T_1)}}{\alpha\rho} \cdot Ec \cdot \left(\frac{dU}{dY}\right)^2 + Ha^2 \cdot Pr \cdot Ec \cdot U^2 = 0 \tag{2.26}$$

For further manipulations, power of the exponential terms has been multiplied and divided by $\varepsilon \cdot \theta$

$$\begin{aligned}\frac{d^2\theta}{dY^2} - Re \cdot Pr \cdot \frac{V}{U_\infty} \cdot \frac{d\theta}{dY} + \frac{\mu_0 e^{-m(T-T_2)\cdot\frac{\varepsilon}{m(T_1-T_2)}\cdot\theta\cdot\frac{(T_1-T_2)}{(T-T_2)}}}{\alpha\rho} \cdot Ec \cdot \left(\frac{dU}{dY}\right)^2 \\ + Ha^2 \cdot Pr \cdot Ec \cdot U^2 = 0\end{aligned} \tag{2.27}$$

where

$$\varepsilon = m \cdot (T_1 - T_2) \tag{2.28}$$

Rearranging results in the final form of non-dimensionalized equation as in (2.29)

$$\frac{d^2\theta}{dY^2} - Re \cdot Pr \cdot \frac{V}{U_\infty} \cdot \frac{d\theta}{dY} + Pr \cdot Ec \cdot e^{-\varepsilon\theta} \cdot \left(\frac{dU}{dY}\right)^2 + Ha^2 \cdot Ec \cdot Pr \cdot U^2 = 0 \tag{2.29}$$

Since one of the two equations is now non-dimensionalised, Eq.(2.6) will be non-dimensionalised next. For that purpose, Eq. (2.8) for viscosity, derivative of velocity from Eq. (2.17), and derivative of pressure is calculated as in Eq. (2.30)

$$dP = \frac{dp \cdot h}{\mu_0 U_\infty} \tag{2.30}$$

and substituted in Eq.(2.6) gives

$$V\frac{U_\infty}{h}\frac{dU}{dy} = -\frac{1}{\rho}\cdot\frac{\mu_0 U_\infty}{h^2}\cdot\frac{dP}{dX} + \frac{\mu_0}{\rho}\cdot\frac{d}{dy}\left(e^{-m(T-T_2)}\cdot\frac{du}{dy}\right) - \frac{\sigma B_0^2 U_\infty}{\rho}U \tag{2.31}$$

taking the derivative of the second term in the right side of the equation gives

$$\begin{aligned} V\frac{U_\infty}{h}\frac{dU}{dY} =& -\frac{1}{\rho}\cdot\frac{\mu_0 U_\infty}{h^2}\cdot\frac{dP}{dX} + \frac{\mu_0}{\rho}\left[-mTe^{-m(T-T_2)}\frac{dT}{dy}\frac{du}{dy} + e^{-m(T-T_2)}\frac{d^2u}{dy^2}\right] \\ & -\frac{\sigma B_0^2 U_\infty}{\rho}U \end{aligned} \tag{2.32}$$

Now substituting Eq. (2.15) and Eq. (2.17) in Eq. (2.32)

$$\begin{aligned} V\frac{U_\infty}{h}\frac{dU}{dY} =& -\frac{1}{\rho}\cdot\frac{\mu_0 U_\infty}{h^2}\cdot\frac{dP}{dX} \\ & + \frac{\mu_0}{\rho}\left[-me^{-m(T-T_2)}\frac{(T_1-T_2)}{h}\frac{U_\infty}{h}\frac{d\theta}{dY}\frac{dU}{dY} + e^{-m(T-T_2)}\frac{U_\infty}{h^2}\frac{d^2U}{dY^2}\right] \\ & -\frac{\sigma B_0^2 U_\infty}{\rho}U \end{aligned} \tag{2.33}$$

Dividing the equation with $\frac{\mu_0}{\rho}\frac{U_\infty}{h^2}e^{-m(T-T_2)}$ and substituting for Ha number given in Eq. (2.22)

$$\begin{aligned} \frac{d^2U}{dY^2} - e^{m(T-T_2)}\frac{V}{U_\infty}U_\infty\frac{\rho}{\mu_0}h\frac{dU}{dY} - e^{m(T-T_2)}\frac{dP}{dX} \\ -m(T_1-T_2)\frac{d\theta}{dY}\frac{dU}{dY} - Ha^2 e^{m(T-T_2)}U = 0 \end{aligned} \tag{2.34}$$

Substituting Re number from Eq. (2.24) and nondimensionalized temperature in the exponential term

$$\begin{aligned} &\frac{d^2U}{dY^2} - e^{m(T-T_2)\varepsilon \frac{1}{m(T_1-T_2)} \theta \frac{(T_1-T_2)}{(T-T_2)}} \frac{V}{U_\infty} Re \frac{dU}{dY} - e^{m(T-T_2)} \frac{dP}{dX} \\ &\quad - \varepsilon \frac{d\theta}{dY} \frac{dU}{dY} - e^{\varepsilon\theta} Ha^2 U = 0 \end{aligned} \tag{2.35}$$

Finally rearranging gives the nondimensionalized form of Eq.(2.6)

$$\frac{d^2U}{dY^2} - e^{\varepsilon\theta}\left[\frac{V}{U_\infty} Re \frac{dU}{dY} + \frac{dP}{dX} + Ha^2 U\right] - \varepsilon \frac{d\theta}{dY}\frac{dU}{dY} = 0 \tag{2.36}$$

So non-dimensional governing equations in summary can be written as

$$\frac{d^2U}{dY^2} - \varepsilon \frac{d\theta}{dY}\frac{\partial U}{\partial Y} - e^{\varepsilon\theta}\left(V^* \mathrm{Re} \frac{dU}{dY} + Ha^2 U - G\right) = 0 \tag{2.37}$$

$$\frac{d^2\theta}{dY^2} - V^* \mathrm{Re}\,\mathrm{Pr} \frac{d\theta}{dY} + Bre^{-\varepsilon\theta}\left(\frac{dU}{dY}\right)^2 + BrHa^2U^2 = 0 \tag{2.38}$$

where

$$\mathrm{Re} = \frac{U_\infty h}{\upsilon} \qquad \mathrm{Pr} = \frac{\upsilon}{\alpha} \qquad Ec = \frac{U_\infty^2}{c_p\left(T_1 - T_2\right)} \tag{2.39}$$

$$Br = Ec \cdot \mathrm{Pr} \qquad Ha = B_0 h \sqrt{\frac{\sigma}{\mu_0}} \qquad \varepsilon = m\left(T_1 - T_2\right) \tag{2.40}$$

$$Bi_0 = \frac{\gamma_0 h}{k} \qquad Bi_1 = \frac{\gamma_1 h}{k} \qquad V^* = \frac{V}{U_\infty} \tag{2.41}$$

The non-dimensionalized governing equations given in (2.37)-(2.38) are subject to boundary conditions below

$$U(0) = 0 \qquad \frac{d\theta}{dY}(0) = Bi_0\left(\theta(0) - 1\right) \tag{2.42}$$

$$U(1) = 0 \qquad \frac{d\theta}{dY}(1) = -Bi_1 \theta(1) \tag{2.43}$$

The governing equations given in (2.37)-(2.38) and the boundary conditions including partial derivatives in (2.42)-(2.43) are discretized utilizing GDQM. Then the resulting system of algebraic equations is solved by Newton-Raphson Algorithm.

2.2 Entropy Generation

Designing optimal engineering systems is a key goal in energy efficiency [104, 105]. To predict the performance of the processes, second law of thermodynamics has been applied for years. However, Bejan introduced the idea of entropy generation minimization to optimize heat exchange system. The convection process in a channel is inherently irreversible which causes entropy generation. Local volumetric entropy generation rate formula for a viscous incompressible conducting fluid in the presence of magnetic field is given by [106] as (2.44), indicating each terms irreversibility reason

$$E_G = \underbrace{\frac{k}{T_2^2}\left(\frac{dT}{dy}\right)^2}_{heat\_transfer} + \underbrace{\frac{\mu}{T_2}\left(\frac{du}{dy}\right)^2}_{viscous\_dissipation} + \underbrace{\frac{\sigma B_0^2}{T_2}u^2}_{magnetic\_field} \tag{2.44}$$

The last term in the right side of (2.44) is the entropy generation due to magnetic field. Evaluating non-dimensional parameters given in (2.11)-(2.12), (2.39) - (2.41), local entropy generation rate is defined as in (2.45).

$$N_S = \frac{T_2^2 h^2 E_G}{k(T_1 - T_2)^2} = \left(\frac{d\theta}{dY}\right)^2 + \frac{Br}{\Omega}\left(e^{-\varepsilon\theta}\left(\frac{dU}{dY}\right)^2 + HaU^2\right) \tag{2.45}$$

Here Ω and Br stand for the temperature difference parameter and Brinkmann number respectively and can be given as $\Omega = \frac{(T_1 - T_2)}{T_2}$, $Br = Ec\,\mathrm{Pr}$. To show the dominance of irreversibility due to heat transfer with respect to the combined effect of fluid friction and magnetic fields, Bejan number is introduced as follows

$$Be = \frac{N_{heat}}{N_S} = \frac{\left(\frac{d\theta}{dY}\right)^2}{\left(\frac{d\theta}{dY}\right)^2 + \frac{Br}{\Omega}\left(e^{-\varepsilon\theta}\left(\frac{dU}{dY}\right)^2 + HaU^2\right)} \tag{2.46}$$

Be number ranges between 0 and 1 with a meaning of dominance of irreversibility due to heat transfer dominates for the values closer to 1, and the combination of fluid friction and magnetic field for the values closer to 0.

2.3 Nanofluid Channel Flow

In this study, steady viscous incompressible flow of a Cu - Water nanofluid bounded by two infinite horizontal parallel plates with a distance *H* in between is considered.

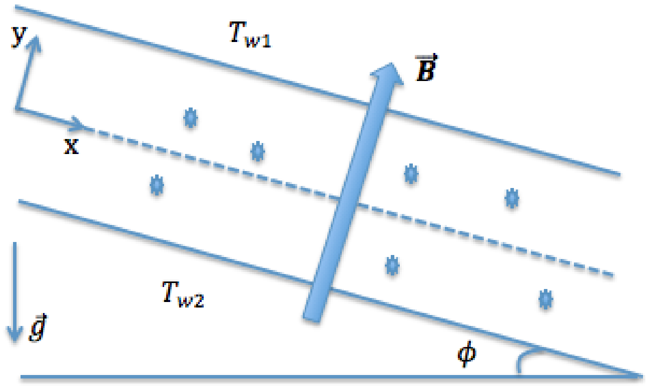

Figure 2.2 : Schematic configuration of the problem.

Cartesian coordinate axes are selected as reference, x xis parallel to plates and located at the same distance to the plates, y - axis along the direction perpendicular to the plates as can be seen in Fig. 2.2. A constant magnetic field is applied perpendicular to the plates uniformly along y xis. The plates are sustained to have a constant temperature of T_{w_2} and T_{w_1}, at the lower and upper plates respectively. The assumptions carried through the study can be summarized as;

- The plates are infinitely long, enabling to consider the problem one dimensional.
- The induced magnetic field is neglected when compared with the applied magnetic field , due to inherently small magnetic Reynolds number for magnetic liquids and partially ionized fluids [107].
- Electric field and Hall effects, the ion-slip and thermoelectric effects, and the electron pressure gradient are neglected [34, 108].
- The flow is assumed to be fully developed and the edge effects are neglected. The schematic configuration of the considered system is given in Figure 2.2. The inclined channel filled with nanofluids is exposed to magnetic field B_0. Under these assumptions the nonlinear coupled differential equations governing the system is given in (2.47) - (2.48).

$$\mu_{nf}\frac{d^2u}{dy^2} + (\rho\beta)_{nf} g \sin\phi\,(T - T_{w2}) - \sigma_{nf} B_0^2 u = \frac{dp}{dx} \tag{2.47}$$

$$\frac{d^2T}{dy^2} + \frac{\mu_{nf}}{k_{nf}}\left(\frac{du}{dy}\right)^2 + \frac{\sigma_{nf}}{k_{nf}} B_0^2 u^2 = 0 \tag{2.48}$$

The system of equations in (2.47) - (2.48) is subject to boundary conditions given in (2.49) - (2.50)

$$u(H/2) = 0 \qquad u(-H/2) = 0 \tag{2.49}$$

$$T(H/2) = T_{w1} \qquad T(-H/2) = T_{w2} \tag{2.50}$$

Addition of nanoparticles changes the thermophysical properties of the fluid flow. There are numerous work to model the surprising nature of the nanofluids but there is no consensus yet. Available models depend on different parameters, e.g the nanoparticle material, the volume fraction of nanofluids, specific heat, and others. In this study, the thermal conductivity model for the nanofluids given in Eq. (2.53) is presented by Choi [25], Brinkmann model [109] given in Eq. (2.54) for nanofluid viscosity. The temperature dependency of these parameters given in Eqs. (2.51)-(2.54) are ignored as usual in the literature investigating nanofluid flows, but should be considered in the following research.

$$\alpha_f = \frac{k_f}{(\rho c_p)_f} \tag{2.51}$$

$$\rho_{nf} = (1-\psi)\rho_f + \psi\rho_p \tag{2.52}$$

$$\frac{k_{nf}}{k_f} = \frac{k_p + 2k_f - 2\psi\left(k_f - k_p\right)}{k_p + 2k_f + 2\psi\left(k_f - k_p\right)} \tag{2.53}$$

$$\frac{\mu_{nf}}{\mu_f} = \frac{1}{(1-\psi)^{2.5}} \tag{2.54}$$

To derive the dimensionless forms of the governing equations, the following parameters in (2.55)-(2.57) are defined.

$$Y = \frac{y}{h} \qquad X = \frac{x}{h} \tag{2.55}$$

$$U = \frac{uH}{\alpha_f} \qquad \theta = \frac{T - T_{w1}}{T_{w2} - T_{w1}} \tag{2.56}$$

$$P = \frac{H^3}{\alpha_f^2 \rho_{nf}}\left(-\frac{dp}{dx}\right) \tag{2.57}$$

Dimensionalized velocity as a function of non-dimensionalized velocity as given in (2.56) is derived once and twice yields to (2.58) and (2.59) respectively.

$$\frac{du}{dy} = \frac{\alpha_f}{H^2} \cdot \frac{dU}{dY} \tag{2.58}$$

$$\frac{d^2u}{dy^2} = \frac{\alpha_f}{H^3} \cdot \frac{d^2U}{dY^2} \tag{2.59}$$

Substituting the non-dimensionalized equals of first and second derivatives in the system equations (2.47) - (2.48) yields

$$\frac{\alpha_f}{H^3} \cdot P + \mu_{nf} \cdot \frac{\alpha_f}{H^3} \cdot \frac{d^2U}{dY^2} + (\rho\beta)_{nf} g \sin\phi (T_{w2} - T_{w1})\theta - \sigma_{nf}\beta_0^2 \frac{\alpha_f}{H} U = 0 \tag{2.60}$$

Multiplying with $\frac{H^3}{\alpha_f^2 \rho_{nf}}$ results, equation becomes

$$P + \frac{\mu_{nf}}{\alpha_f \rho_{nf}} \cdot \frac{d^2U}{dY^2} + (\rho\beta)_{nf} g \sin\phi (T_{w2} - T_{w1})\theta \frac{H^3}{\alpha_f^2 \rho_{nf}} - \frac{\sigma_{nf} H^2}{\alpha_f \rho_{nf}} \beta_0^2 U = 0 \tag{2.61}$$

Defining the non-dimensional variables Ra and Ha

$$Ra = \frac{g\beta_f H^3 (T_{w2} - T_{w1})}{\upsilon_f \alpha_f} \tag{2.62}$$

$$Ha = B_0 H \sqrt{\frac{\sigma_{nf}}{\rho_{nf} \upsilon_f}} \tag{2.63}$$

and multiplying and dividing some of the terms of (2.61) with μ_f and υ_f to substitute with Ra and Ha in the equation

$$P + \frac{\mu_{nf}}{\alpha_f \rho_{nf}} \cdot \frac{d^2U}{dY^2} + \frac{(\rho\beta)_{nf} g \sin\phi (T_{w2} - T_{w1})\theta}{\alpha_f \rho_{nf} \alpha_f} \cdot \frac{\beta_f}{\beta_f} \cdot \frac{\nu_f}{\nu_f} - \frac{\sigma_{nf} H^2 \beta_0^2}{\rho_{nf} \alpha_f} \cdot U \cdot \frac{\nu_f}{\nu_f} = 0 \tag{2.64}$$

Replacing the defined Ra and Ha in

$$P + \frac{\mu_{nf}}{\alpha_f \rho_{nf}} \cdot \frac{d^2U}{dY^2} + \frac{(\rho\beta)_{nf} g \sin\phi Ra Pr \theta}{\rho_{nf} \beta_f} - Pr \cdot Ha^2 \cdot U = 0 \tag{2.65}$$

Thus finally (2.65) is derived from the original dimensionalized equation (2.47). So now, non-dimensionalization of (2.48) will be shown.

First, the second derivative of the temperature as a function of non-dimensionalized temperature is calculated using (2.56),

$$\frac{d^2T}{dy^2}=\frac{(T_{w2}-T_{w1})}{H^2}\cdot\frac{d^2\theta}{dY^2} \tag{2.66}$$

Substituting the states with their non-dimensionalized forms in (2.48) gives

$$\frac{(T_{w2}-T_{w1})}{H^2}\cdot\frac{d^2\theta}{dy^2}+\frac{\mu_{nf}}{k_{nf}}\left[\frac{\alpha_f}{H}\cdot dU\cdot\frac{1}{HdY}\right]^2+\frac{\sigma_{nf}}{k_{nf}}\cdot\beta_0^2\cdot\frac{\alpha_f^2}{H^2}\cdot U^2=0 \tag{2.67}$$

Rearranging the terms,

$$\frac{(T_{w2}-T_{w1})}{H^2}\cdot\frac{d^2\theta}{dy^2}+\frac{\mu_{nf}}{k_{nf}}\cdot\frac{\alpha_f^2}{H^4}\cdot\left(\frac{dU}{dY}\right)^2+\frac{\sigma_{nf}}{k_{nf}}\cdot\beta_0^2\cdot\frac{\alpha_f^2}{H^2}\cdot U^2=0 \tag{2.68}$$

Multiplying with $\frac{H^2}{(T_{w2}-T_{w1})}$ results, equation becomes

$$\frac{d^2\theta}{dy^2}+\frac{\mu_{nf}\alpha_f^2}{k_{nf}H^2(T_{w2}-T_{w1})}\cdot\left(\frac{dU}{dY}\right)^2+\frac{\sigma_{nf}}{k_{nf}}\cdot\beta_0^2\cdot\frac{\alpha_f^2}{(T_{w2}-T_{w1})}\cdot U^2=0 \tag{2.69}$$

Defining the non-dimensional variables Ec, Pr and Br

$$Ec=\frac{\alpha_f^2}{H^2c_{p_f}(T_{w2}-T_{w1})} \tag{2.70}$$

$$Pr=\frac{\mu_f c_{p_f}}{k_f} \tag{2.71}$$

$$Br=Ec\cdot Pr=\frac{\alpha_f^2\mu_f}{H^2(T_{w2}-T_{w1})k_f} \tag{2.72}$$

and multiplying and dividing some of the terms of (2.69) with μ_f, k_f, μ_{nf}, H^2, ρ_{nf}, υ_{nf} to substitute with Ec and Pr, Br and Ha in the equation

$$\begin{split}\frac{d^2\theta}{dy^2}+\frac{\alpha_f^2}{H^2(T_{w2}-T_{w1})}\cdot\frac{\mu_f}{k_f}\cdot\frac{k_f}{\mu_f}\cdot\frac{\mu_{nf}}{k_{nf}}\cdot\left(\frac{dU}{dY}\right)^2\\+\frac{\sigma_{nf}}{k_{nf}}\cdot\frac{\alpha_f^2}{(T_{w2}-T_{w1})}\cdot\beta_0^2\cdot U^2\frac{H^2}{H^2}\cdot\frac{\rho_{nf}}{\rho_{nf}}\cdot\frac{\upsilon_{nf}}{\upsilon_{nf}}=0\end{split} \tag{2.73}$$

Replacing the defined Br and Ha in

$$\frac{d^2\theta}{dy^2}+Br\cdot\frac{k_f}{k_{nf}}\cdot\frac{\mu_{nf}}{\mu_f}\cdot\left(\frac{dU}{dY}\right)^2+\frac{\alpha_f^2\rho_{nf}\upsilon_f}{k_{nf}H^2(T_{w2}-T_{w1})}\cdot Ha^2\cdot U^2\cdot\frac{\mu_f}{\mu_f}\cdot\frac{k_f}{k_f}=0 \tag{2.74}$$

and rearranging gives

$$\frac{d^2\theta}{dy^2}+\frac{k_f}{k_{nf}}\cdot\frac{\mu_{nf}}{\mu_f}\cdot Br\cdot\left(\frac{dU}{dY}\right)^2+\frac{\rho_{nf}}{\rho_f}\cdot\frac{k_f}{k_{nf}}\cdot Br\cdot Ha^2\cdot U^2=0 \tag{2.75}$$

To sum up, considering the dimensionless parameters the non-dimensionalized differential equations describing the flow is given in Eq. (2.76)-(2.77)

$$\frac{\mu_{nf}}{\rho_{nf}\alpha_f}\frac{d^2U}{dY^2}+\frac{(\rho\beta)_{nf}}{\rho_{nf}\beta_f}\sin\phi Ra\Pr\theta-\Pr Ha^2U+P=0 \tag{2.76}$$

$$\frac{d^2\theta}{dY^2}+\frac{k_f\mu_{nf}}{k_{nf}\mu_f}Br\left(\frac{dU}{dY}\right)^2+\frac{\rho_{nf}k_f}{\rho_f k_{nf}}BrHa^2U^2=0 \tag{2.77}$$

and dimensionless parameters describing characteristics of the fluid flow utilized in dimensionalization are summarized in Eq. (2.78)-(2.80)

$$Ha=B_0H\sqrt{\frac{\sigma_{nf}}{\rho_{nf}\upsilon_f}} \qquad Ra=\frac{g\beta_fH^3\left(T_{w2}-T_{w1}\right)}{\upsilon_f\alpha_f} \tag{2.78}$$

$$Ec=\frac{\alpha_f^2}{H^2C_{P_f}\left(T_{w2}-T_{w1}\right)} \qquad \Pr=\frac{\mu_fC_{P_f}}{k_f} \tag{2.79}$$

$$Br=Ec\cdot\Pr=\frac{\alpha_f^2\mu_f}{H^2\left(T_{w2}-T_{w1}\right)k_f} \tag{2.80}$$

The nondimensionalized system of equations given in Eq. (2.78)-(2.80) is subject to nondimensionalized boundary conditions given in Eq. (2.81)-(2.82)

$$U(-1)=0 \qquad U(1)=0 \tag{2.81}$$

$$\theta(-1)=0 \qquad \theta(1)=1 \tag{2.82}$$

The viscous dissipation effects and the dependence of temperature to magnetic field are taken into account for this one dimensional problem. The equations are first discretized utilizing GDQM. Then the resulting system of algeabric equations are solved by Newton Raphson Algorithm.

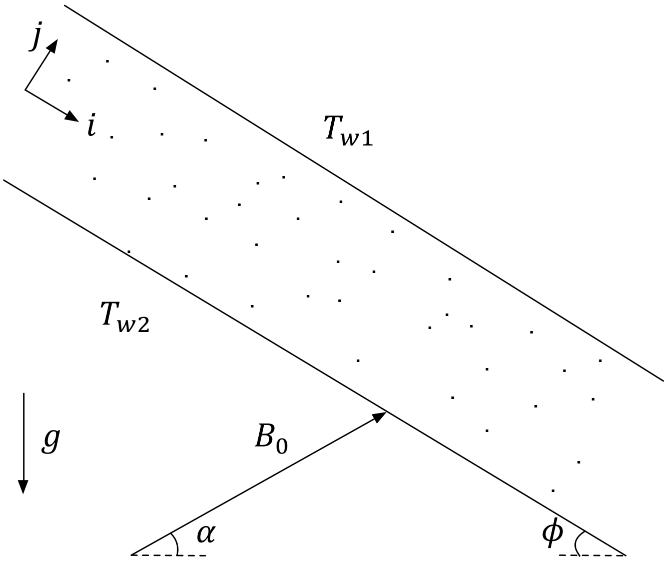

Figure 2.3 : Schematic configuration of the studied problem.

2.4 Nanofluid Channel Flow Exposed to Inclined Magnetic Field

In this study, steady viscous incompressible flow of a Cu-Water nanofluid bounded by two infinite horizontal parallel plates with a distance H in between is considered. Cartesian coordinate axes are selected as reference, x-axis parallel to plates and located at the same distance to the plates, y-axis along the direction perpendicular to the plates as can be seen in Figure 2.3. A constant magnetic field is applied perpendicular to the plates uniformly along y-axis. The plates are sustained to have a constant temperature of T_{w_2} and T_{w_1}, at the lower and upper plates respectively. The assumptions carried through the study can be summarized as:

- The plates are infinitely long, enabling to consider the problem one dimensional.
- The induced magnetic field is neglected when compared with the applied magnetic field, due to inherently small magnetic Reynolds number for magnetic liquids and partially ionized fluids [107].
- Electric field and Hall effects, the ion-slip and thermoelectric effects, and the electron pressure gradient are neglected [**?**, 108].
- The flow is assumed to be fully developed and the edge effects are neglected.

Literature for an inclined channel problem is quite rich yet still requires to be combined to point the specific problem tackled. As such, in [110] discusses the influence of inclined magnetic field on peristaltic flow of a Williamson fluid model in an inclined channel without nanofluid. In [111] encounters the effects of and external oriented magnetic field on entropy in a cavity configuration without nanoparticles. Reference [112] studies the MHD effects on heat transfer and entropy generation of nanofluid flow in an open cavity in the presence of an inclined magnetic field. References [113, 114] also referred for their application of inclined channel flow in the presence of nanofluid with fixed external magnetic field. The missing effects of one another has been merged to find out the equations of motion which simplifies to the equations in the literature for under special conditions, such as constant external magnetic field.

The schematic configuration of the considered system is given in Figure 2.3. The inclined channel filled with nanofluids is exposed to magnetic field B_0. Under these assumptions the nonlinear coupled differential equations governing the system can be written as Equations (2.83) and (2.84).

$$\mu_{nf}\frac{d^2u}{dy^2}+(\rho\beta)_{nf}g\sin\phi\,(T-T_{w2})-\sigma_{nf}B_0^2\sin^2(\alpha+\phi)\,u=\frac{dp}{dx} \tag{2.83}$$

$$\frac{d^2T}{dy^2}+\frac{\mu_{nf}}{k_{nf}}\left(\frac{du}{dy}\right)^2+\frac{\sigma_{nf}}{k_{nf}}B_0^2\sin^2(\alpha+\phi)\,u^2=0 \tag{2.84}$$

The system of equations in Equations (2.83) and (2.84) is subject to boundary conditions Equations (2.85) and (2.86).

$$u(H/2)=0 \qquad u(-H/2)=0 \tag{2.85}$$

$$T(H/2)=T_{w1} \qquad T(-H/2)=T_{w2} \tag{2.86}$$

Addition of nanoparticles changes the thermophysical properties of the fluid flow. There are numerous work in open literature to model the surprising nature of the nanofluids but there is no consensus yet. Available models depend on different parameters, e.g., the nanoparticle material, the volume fraction of nanofluids, specific heat, and others. In this study, thermal diffusivity of the nanofluid α_{nf} and effective density of the nanofluid ρ_{nf} is expressed as.

$$\alpha_{nf}=\frac{k_{nf}}{(\rho c_p)_{nf}} \tag{2.87}$$

$$\rho_{nf}=(1-\psi)\,\rho_f+\psi\rho_p \tag{2.88}$$

Effective dynamic viscosity of the nanofluid is given by the Brinkmann model [109] as Equation (2.89).

$$\frac{\mu_{nf}}{\mu_f} = \frac{1}{(1-\psi)^{2.5}} \tag{2.89}$$

The effective thermal conductivity k_{nf} and the electrical conductivity σ_{nf} of the nanofluid are modeled by Maxwell [11] and given as.

$$\frac{k_{nf}}{k_f} = \frac{k_p + 2k_f - 2\psi\left(k_f - k_p\right)}{k_p + 2k_f + \psi\left(k_f - k_p\right)} \tag{2.90}$$

$$\frac{\sigma_{nf}}{\sigma_f} = 1 + \frac{3\left(\frac{\sigma_p}{\sigma_f} - 1\right)\psi}{\left(\frac{\sigma_p}{\sigma_f} + 2\right) - \left(\frac{\sigma_p}{\sigma_f} - 1\right)\psi} \tag{2.91}$$

To derive the dimensionless forms of the governing equations, the following parameters in Equations (2.92)–(2.94) are defined.

$$Y = \frac{y}{h} \qquad X = \frac{x}{h} \tag{2.92}$$

$$U = \frac{uH}{\alpha_f} \qquad \theta = \frac{T - T_{w1}}{T_{w2} - T_{w1}} \tag{2.93}$$

$$P = \frac{H^3}{\alpha_f^2 \rho_{nf}}\left(-\frac{dp}{dx}\right) \tag{2.94}$$

Some other dimensionless parameters describing characteristics of the fluid flow utilized in dimensionalization can be listed as Equations (2.95)–(2.97).

$$Ha = B_0 H\sqrt{\frac{\sigma_{nf}}{\rho_{nf}\upsilon_f}} \qquad Ra = \frac{g\beta_f H^3\left(T_{w2} - T_{w1}\right)}{\upsilon_f \alpha_f} \tag{2.95}$$

$$Ec = \frac{\alpha_f^2}{H^2 C_{P_f}\left(T_{w2} - T_{w1}\right)} \qquad \mathrm{Pr} = \frac{\mu_f C_{P_f}}{k_f} \tag{2.96}$$

$$Br = Ec \cdot \mathrm{Pr} = \frac{\alpha_f^2 \mu_f}{H^2\left(T_{w2} - T_{w1}\right)k_f} \tag{2.97}$$

Considering the dimensionless parameters, the non-dimensionalized differential equations describing the flow can be summarized with Equations (2.98) and (2.99).

$$\frac{\mu_{nf}}{\rho_{nf}\alpha_f}\frac{d^2U}{dY^2} + \frac{(\rho\beta)_{nf}}{\rho_{nf}\beta_f}\sin\phi Ra \mathrm{Pr}\,\theta - \mathrm{Pr} Ha^2 \sin^2(\alpha + \phi)U + P = 0 \tag{2.98}$$

$$\frac{d^2\theta}{dY^2} + \frac{k_f\mu_{nf}}{k_{nf}\mu_f}Br\left(\frac{dU}{dY}\right)^2 + \frac{\rho_{nf}k_f}{\rho_f k_{nf}}BrHa^2\sin^2(\alpha + \phi)U^2 = 0 \tag{2.99}$$

The nondimensionalized system of Equations (2.98) and (2.99) are subject to non-dimensionalized boundary conditions Equations (2.100) and (2.101).

$$U(-1) = 0 \qquad U(1) = 0 \tag{2.100}$$

$$\theta(-1) = 0 \qquad \theta(1) = 1 \tag{2.101}$$

The viscous dissipation effects and magnetic field effects on temperature are taken into account for this one dimensional problem. The equations are first discretized utilizing GDQM. Then the resulting system of algebraic equations are solved by Newton-Raphson Algorithm.

3. GENERALIZED DIFFERENTIAL QUADRATURE METHOD

3.1 Motivation and Historical Advance

Solving differential equations has always been a challenge either they are ordinary or partial. When there is no analytical solution, numerical techniques are referred to and thanks to the exponential increase in the capacity of processing power in processors, it is possible to have solutions to a variety of complicated problems with an satisfactory accuracy. But still, for some specific applications, such as embedded systems, computational power might be limited which requires some intelligent techniques for discretizing the equations to have fast and accurate solutions. Most of the times, there exists a trade - off between accuracy and computation time, so the balance would also be a function of the requirements of the system in the focus of interest. Solution of differential equations might be necessary for optimization and optimal control problems, making it a crucial step to complete starting from the design step to eventually controlling it during its mission execution.

For the problems whose governing partial differential equations are difficult or not possible to be solved in closed-form, approximate solutions are turned to. Basically, approximate solutions are the relations between the derivatives in the partial differential equation and the functional values at certain discrete points(grid points or mesh points). This relation between those two is attained by numerical discretization and coins the terms for the solutions as numerical solutions.

Numerical discretization techniques are not new or lacking, a wide variety of options are available in the literature. Each possess some advantage over the others. To name a few are the finite difference method, finite element method under low order methods whereas spectral and pseudospectral methods as global methods. Finite difference method relies on Taylor series expansion or polynomial approximation while finite element method is based on variational principle or the weighted residuals principle. Spectral methods use a developed version of method of weighted residuals which heavily rely on base and weighting functions. The similarity that both finite element and spectral methods use of set of base functions differentiate in their choices of base functions.

Most of the engineering problems can be solved via low order methods such as finite element, finite difference, and finite volume, for a large number of grid points. The necessity to use a larger set of grip points to have satisfactorily accurate solutions might be a burden for applications where solutions for the whole physical domain is not really necessary, i.e the user only needs the solutions for a specific part of the domain.

Accurate solutions with less number of grids is of concern also since it requires less computational need which is required for a variety of disciplines. With such a goal, Differential Quadrature Method(DQM) has been offered by Richard Bellmann and his colleagues in 1971. They were in search to have satisfactorily precise solutions with a small number of grids.

Differential quadrature relies on the calculation of a partial derivative of a function with respect to a coordinate direction by a linear weighted sum of all evaluated function values at all the grid points along the same direction. It is inspired from integral quadrature.

Integral quadrature is the evaluation of integral $\int_a^b f(x)dx$ over a finite internal $[a,b]$. When it is not possible to find an explicit solution for F such that $\frac{dF}{dx} = f$, the knowledge of f at grid points can be used to approximate the integral. This integral also can be represented as the area under the curve represented by $f(x)$ as seen in Figure 3.1. So the idea is to find the approximate value of the integral, the area under the curve can be calculated approximately such as

$$\int_a^b f(x)dx = w_1 f_1 + w_2 f_2 + ... + w_n f_n = \sum_{k=1}^{n} w_k f_k \tag{3.1}$$

where $w_1, w_2, ..., w_n$ are the weighting coefficients, $f_1, f_2, f, ..., f_n$ are the functional values at grid points $a = x_1, x_2, ..., x_n = b$ and Equation (3.1) is the equation of integral quadrature. The choice of these discrete grid points vary depending on the problem.

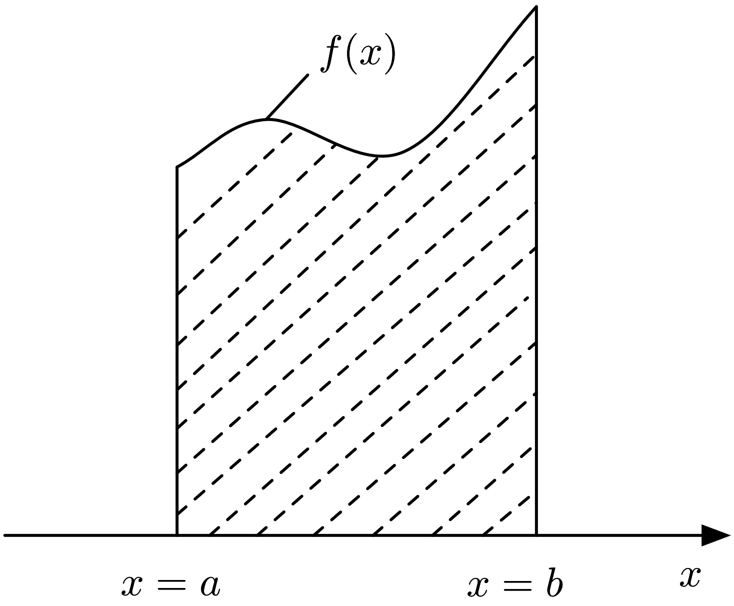

Figure 3.1 : Integral of $f(x)$ over an interval.

This idea of calculating the integral of a function inspired the differential quadrature. Bellmann suggested that the 1st order derivative of a function $f(x)$ with respect to x evaluated at a grid point x_i can be approximated as a linear sum of all the functional values evaluated at all grid points along this dimension. For one-dimensional problem as given in Figure 3.2 the 1st order derivative of a function $f(x)$ with respect to x evaluated at a grid point x_i can be written as

$$f_x(x_i) = \left.\frac{df}{dx}\right|_{x_i} = \sum_{j=1}^{N} a_{ij} \cdot f(x_j), \quad for\, i = 1, 2, ..., N \tag{3.2}$$

where a_{ij} are the weighting coefficients, N is the number of grids. The key procedure of DQM is to select the weighting coefficients.

Differential Quadrature, inspired by integral quadrature, is a powerful technique to approximate the derivatives of a function at a point as a weighted linear sum of all the functional values at discrete points along the derivation direction. To generalize, in DQM, the n^{th} order derivative of a single variable function $u(y)$ can be approximated as given in (3.3).

$$u^{(n)}(y_i) = \sum_{j=1}^{N} c_{ij}^{(n)} u(y_j) \quad i = 1, 2, \ldots, N \tag{3.3}$$

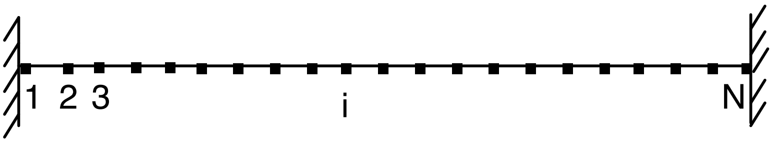

Figure 3.2 : One-dimensional problem gridding example.

where $c_{ij}^{(n)}$ are the weighting coefficients for the n^{th} order derivative, and N is the number of grids. The challenging step in utilization of DQM is the calculation of weighting coefficients. Bellmann offered two methods to calculate the first order coefficients [115]. To show how, let's specialize in the first order derivative case rather than the general n^{th} order case. For that write again the first order derivative approximation, but this time indicating that this is the first derivative as

$$f_x^{(1)}(x_i) = \sum_{j=1}^{N} a_{ij} \cdot f(x_j), \quad for\, i = 1,2,...,N \tag{3.4}$$

Both of Bellmann's methods to calculate the weighting coefficients are based on test functions which defer from one to the other. In the first method, the N test functions are selected as

$$g_k(x) = x^k, \quad for\, k = 0,2,...,N-1 \tag{3.5}$$

since i and j is varied from 1 to N as seen from Equ. (3.4), the number of weighting coefficients a_{ij} is NxN. When these N test functions given in Equ. (3.5) are applied to N grid points, NxN algebraic equations

$$\begin{aligned}
&\sum_{j=1}^{N} a_{ij} = 0 \\
&\sum_{j=1}^{N} a_{ij} \cdot x_j = 0 \\
&\sum_{j=1}^{N} a_{ij} \cdot x_j^k = k \cdot x_i^{k-1}, \quad k = 2,3,...,N-1 \\
&for \quad i = 1,2,...,N
\end{aligned} \tag{3.6}$$

are obtained to solve for NxN weighting coefficients a_{ij}. The system of equations form a Vandermonde matrix and has a unique solution. But when N gets larger the matrix becomes ill-conditioned and inversion is challenging. In its applications, the number of grids is restricted to be less than 13. Although the coordinates of the grid points are chosen arbitrarily, the restriction on the number of grids widely narrowed its applications.

In the second method, the test functions used are

$$g_k(x) = \frac{L_N(x)}{(x-x_k)L_N^{(1)}(x_k)}, \quad for\, k = 1,2,...,N \tag{3.7}$$

where $L_N(x)$ is the Legendre polynomial of degree N and $L_N^1(x)$ is its first derivative.When x_k is chosen to be the roots of the Legendre polynomial and Equ. (3.7) is written at all grid points, an algebraic relations is attained to calculate the weighting coefficients.

$$\begin{aligned} a_{ij} &= \frac{L_N^{(1)}(x_i)}{(x_i-x_j)L_N^{(1)}(x_j)}, \quad for j \neq i \\ a_{ii} &= \frac{1-2x_i}{2x_i(x_i-1)} \end{aligned} \tag{3.8}$$

Although calculation of weighting coefficients is a simple algebraic formulation, the selection of grid point coordinates are restricted to lie at the Legendre polynomial.

These drawbacks limited the use of method until a major breakthrough is attained by Shu and Richards [116] where all the methods available are generalized under the analysis of a high order polynomial approximation and analysis of a linear vector space. This methodology to calculate the weighting coefficients gives the following formulas given in (3.9).

$$a_{ij} = \begin{cases} \frac{M^{(1)}(x_i)}{(x_i-x_j)M^{(1)}(x_j)} & i \neq j \\ -\sum\limits_{j=1,j\neq i}^{N} a_{ij} & i = j \end{cases} \tag{3.9}$$

where $M^{(1)}$ is given by (3.10).

$$M^{(1)}(x_i) = \prod_{k=1,k\neq i}^{N} (x_i - x_k) \tag{3.10}$$

This method lets the user to calculate the second and higher order weighting coefficients by a recurrence relationship with arbitrary choice of grid points [117]. To solve the weighting coefficients of higher order derivatives, following recurrence relations given in (3.11) is used.

$$w_{ij}^{(m)} = \begin{cases} m\left(w_{ii}^{(m-1)} a_{ij} - \frac{w_{ij}^{(m-1)}}{(x_i - x_j)}\right) & i \neq j \\ -\sum\limits_{j=1,j\neq i}^{N} w_{ij}^{(m)} & i = j \end{cases} \tag{3.11}$$

For the choice of grid points locations, there is a vast number of methods. The two different types of grid distribution selected for numerical study of the problems in this work are equal gridding and Chebyshev-Gauss-Lobatto gridding. Equal gridding ensures that the successive grids have an equal distance in-between. In Chebyshev-Gauss-Lobatto gridding, the grid points are denser at the places closer to the boundaries. The relation giving the coordinates of the grids in Chebyshev-Gauss-Lobatto gridding is given in (3.27).

$$y_i = \cos\left(\frac{i-1}{N-1}\pi\right) \tag{3.12}$$

For a better understanding of the DQM, let's apply it to the equation

$$\left[\frac{1-y^2}{2}\right]\frac{\partial T}{\partial x} - \frac{\partial^2 T}{\partial y^2} = 0 \tag{3.13}$$

with boundary conditions

$$T(0,y) = 0 \tag{3.14}$$

$$T(x,1) = 1 \tag{3.15}$$

$$\frac{\partial T}{\partial y}(x,0) = 0 \tag{3.16}$$

Using the definition of DQM n^{th} order derivative approximation given in (3.3)

$$\left[\frac{1-y_j^2}{2}\right]\sum_{k=1}^{N} w_{ik}^{(1)}T_{kj}-\sum_{k=1}^{M}\bar{w}_{jk}^{(2)}T_{ik}=0 \tag{3.17}$$

and the discretized boundary conditions

$$T_{1,j}=0 \tag{3.18}$$

$$T_{i,M}=1 \tag{3.19}$$

$$\frac{\partial T_{i1}}{\partial y}=0 \tag{3.20}$$

where $i=1...N$ and $j=1...M$. In order to write the set of equations in a matrix format, some of the equations are written explicitly by evaluating some values of i and j such as for $i=1,\ j=1$

$$\begin{aligned}\left[\frac{1-y_1^2}{2}\right]&\left[w_{11}^{(1)}T_{11}+w_{12}^{(1)}T_{21}+\cdots+w_{1N}^{(1)}T_{N1}\right]\\&-\left[\bar{w}_{11}^{(2)}T_{11}+\bar{w}_{12}^{(2)}T_{12}+\cdots+\bar{w}_{1M}^{(2)}T_{1M}\right]=0\end{aligned} \tag{3.21}$$

and $i=1,\ j=2$

$$\begin{aligned}\left[\frac{1-y_2^2}{2}\right]&\left[w_{11}^{(1)}T_{12}+w_{12}^{(1)}T_{22}+\cdots+w_{1N}^{(1)}T_{N2}\right]\\&-\left[\bar{w}_{21}^{(2)}T_{11}+\bar{w}_{22}^{(2)}T_{12}+\cdots+\bar{w}_{2M}^{(2)}T_{1M}\right]=0\end{aligned} \tag{3.22}$$

for $i=1,\ j=M$

$$\begin{aligned}\left[\frac{1-y_M^2}{2}\right]&\left[w_{11}^{(1)}T_{1M}+w_{12}^{(1)}T_{2M}+\cdots+w_{1N}^{(1)}T_{NM}\right]\\&-\left[\bar{w}_{M1}^{(2)}T_{11}+\bar{w}_{M2}^{(2)}T_{12}+\cdots+\bar{w}_{MM}^{(2)}T_{1M}\right]=0\end{aligned} \tag{3.23}$$

and also $i=2,\ j=1$

$$\begin{aligned}\left[\frac{1-y_1^2}{2}\right]&\left[w_{21}^{(1)}T_{11}+w_{22}^{(1)}T_{21}+\cdots+w_{2N}^{(1)}T_{N1}\right]\\&-\left[\bar{w}_{11}^{(2)}T_{21}+\bar{w}_{12}^{(2)}T_{22}+\cdots+\bar{w}_{1M}^{(2)}T_{2M}\right]=0\end{aligned} \tag{3.24}$$

and also $i = N,\ j = 1$

$$\left[\frac{1-y_1^2}{2}\right]\left[w_{N1}^{(1)}T_{11} + w_{N2}^{(1)}T_{21} + \cdots + w_{NN}^{(1)}T_{N1}\right] \\ -\left[\bar{w}_{11}^{(2)}T_{N1} + \bar{w}_{12}^{(2)}T_{N2} + \cdots + \bar{w}_{1M}^{(2)}T_{NM}\right] = 0 \tag{3.25}$$

finally and also $i = N,\ j = M$

$$\left[\frac{1-y_M^2}{2}\right]\left[w_{N1}^{(1)}T_{1M} + w_{N2}^{(1)}T_{2M} + \cdots + w_{NN}^{(1)}T_{NM}\right] \\ -\left[\bar{w}_{M1}^{(2)}T_{N1} + \bar{w}_{M2}^{(2)}T_{N2} + \cdots + \bar{w}_{MM}^{(2)}T_{NM}\right] = 0 \tag{3.26}$$

These sets of equations are now gathered in matrix form as shown in Figure 3.3

$$\boldsymbol{WT} = 0 \tag{3.27}$$

so that it can be solved via inverting the matrix. Then the boundary conditions are applied to the system of equations by updating the corresponding rows in the matrix representation which shown in Figure 3.4. This is a linear set of equations which means that the system can be represented in the form (3.28).

$$\boldsymbol{Ax} = \boldsymbol{b} \tag{3.28}$$

To solve for $\boldsymbol{x}$, $\boldsymbol{A}$ is inverted and multiplied by $\boldsymbol{b}$

$$\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b} \tag{3.29}$$

If the system can not be represented in such a form, than it can not be solved via inverting the matrix. For nonlinear problems, after discretization via DQM, some other method, such as Newton-Raphson should be referred for the solution of system of equations.

$$
\left[\begin{array}{cccc|cccc|c|cccc}
\left[\frac{1-y_1^2}{2}\right]w_{11}^{(1)}-\bar{w}_{11}^{(2)} & -\bar{w}_{12}^{(2)} & \cdots & -\bar{w}_{1M}^{(2)} & \left[\frac{1-y_1^2}{2}\right]w_{12}^{(1)} & & & & \cdots & \left[\frac{1-y_1^2}{2}\right]w_{1N}^{(1)} & & & \\
-\bar{w}_{21}^{(2)} & \left[\frac{1-y_2^2}{2}\right]w_{11}^{(1)}-\bar{w}_{22}^{(2)} & \cdots & -\bar{w}_{2M}^{(2)} & & \left[\frac{1-y_2^2}{2}\right]w_{12}^{(1)} & & & \cdots & & \left[\frac{1-y_2^2}{2}\right]w_{1N}^{(1)} & & \\
\vdots & \vdots & \ddots & \vdots & & & \ddots & & \cdots & & & \ddots & \\
\vdots & \vdots & & \vdots & & & & & \cdots & & & & \\
-\bar{w}_{M1}^{(2)} & -\bar{w}_{M2}^{(2)} & \cdots & \left[\frac{1-y_M^2}{2}\right]w_{11}^{(1)}-\bar{w}_{MM}^{(2)} & & & & \left[\frac{1-y_M^2}{2}\right]w_{12}^{(1)} & \cdots & & & & \left[\frac{1-y_M^2}{2}\right]w_{1N}^{(1)} \\
\hline
\left[\frac{1-y_1^2}{2}\right]w_{21}^{(1)} & & & & \left[\frac{1-y_1^2}{2}\right]w_{22}^{(1)}-\bar{w}_{11}^{(2)} & -\bar{w}_{12}^{(2)} & \cdots & -\bar{w}_{1M}^{(2)} & \cdots & \left[\frac{1-y_1^2}{2}\right]w_{2N}^{(1)} & & & \\
& \left[\frac{1-y_2^2}{2}\right]w_{21}^{(1)} & & & -\bar{w}_{21}^{(2)} & \left[\frac{1-y_2^2}{2}\right]w_{22}^{(1)}-\bar{w}_{22}^{(2)} & \cdots & -\bar{w}_{2M}^{(2)} & \cdots & & \left[\frac{1-y_2^2}{2}\right]w_{2N}^{(1)} & & \\
& & \ddots & & \vdots & \vdots & \ddots & \vdots & \cdots & & & \ddots & \\
& & & & \vdots & \vdots & & \vdots & \cdots & & & & \\
& & & \left[\frac{1-y_M^2}{2}\right]w_{21}^{(1)} & -\bar{w}_{M1}^{(2)} & -\bar{w}_{M2}^{(2)} & \cdots & \left[\frac{1-y_M^2}{2}\right]w_{22}^{(1)}-\bar{w}_{MM}^{(2)} & \cdots & & & & \left[\frac{1-y_M^2}{2}\right]w_{2N}^{(1)} \\
\hline
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \vdots \\
\hline
\left[\frac{1-y_1^2}{2}\right]w_{N1}^{(1)} & & & & \left[\frac{1-y_1^2}{2}\right]w_{N2}^{(1)} & & & & \cdots & \left[\frac{1-y_1^2}{2}\right]w_{NN}^{(1)}-\bar{w}_{11}^{(2)} & -\bar{w}_{12}^{(2)} & \cdots & -\bar{w}_{1M}^{(2)} \\
& \ddots & & & & \ddots & & & \cdots & -\bar{w}_{21}^{(2)} & -\bar{w}_{22}^{(2)} & \ddots & -\bar{w}_{2M}^{(2)} \\
& & & & & & \ddots & & \cdots & \vdots & \vdots & & \vdots \\
& & & & & & & & \cdots & \vdots & \vdots & & \vdots \\
& & & \left[\frac{1-y_M^2}{2}\right]w_{N1}^{(1)} & & & & \left[\frac{1-y_M^2}{2}\right]w_{N2}^{(1)} & \cdots & -\bar{w}_{M1}^{(2)} & -\bar{w}_{M2}^{(2)} & \cdots & \left[\frac{1-y_M^2}{2}\right]w_{NN}^{(1)}-\bar{w}_{MM}^{(2)}
\end{array}\right]
=
\left[\begin{array}{c}
T_{11} \\ T_{12} \\ \vdots \\ T_{1(M-1)} \\ T_{1M} \\ \hline T_{21} \\ T_{22} \\ \vdots \\ T_{2(M-1)} \\ T_{2M} \\ \hline \vdots \\ \hline T_{N1} \\ T_{N2} \\ \vdots \\ T_{N(M-1)} \\ T_{NM}
\end{array}\right]
$$

Figure 3.3 : Matrix representation of the system of equations.

$$
\begin{bmatrix}
1 & & \cdots & & 0 & \cdots & & 0 & \cdots & 0 & \cdots & & 0 \\
 & 1 & \cdots & & 0 & \cdots & & 0 & \cdots & 0 & & & \\
\vdots & \vdots & \ddots & \vdots & \vdots & & \ddots & & \cdots & \vdots & & \ddots & \\
\vdots & \vdots & & \vdots & & & & & \cdots & & & & \\
 & & \cdots & 1 & & & & 0 & \cdots & 0 & & & 0 \\
\hline
 & & & & \bar{w}_{11}^{(1)} & \bar{w}_{12}^{(1)} & \cdots & \bar{w}_{1M}^{(1)} & \cdots & & & & \\
 & & & & & & \cdots & & \cdots & & & & \\
 & & \ddots & & \vdots & \vdots & \ddots & \vdots & \cdots & & & \ddots & \\
 & & & & \vdots & \vdots & & \vdots & \cdots & & & & \\
0 & 0 & \cdots & 0 & 0 & & \cdots & 1 & \cdots & & & & \\
\hline
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \vdots \\
\hline
 & & & & & & & & \cdots & & & \cdots & \\
 & \ddots & & & & \ddots & & & \cdots & & & \ddots & \\
 & & & & & & \ddots & & \cdots & \vdots & \vdots & & \vdots \\
 & & & & & & & & \cdots & \vdots & \vdots & & \vdots \\
0 & 0 & \cdots & 0 & 0 & \cdots & & \cdots & \cdots & & & \cdots & 1
\end{bmatrix}
\begin{bmatrix}
T_{11} \\ T_{12} \\ \vdots \\ \\ T_{1M} \\ \hline T_{21} \\ T_{22} \\ \vdots \\ T_{2(M-1)} \\ T_{2M} \\ \hline \vdots \\ \hline T_{N1} \\ T_{N2} \\ \vdots \\ T_{N(M-1)} \\ T_{NM}
\end{bmatrix}
=
\begin{bmatrix}
0 \\ 0 \\ \vdots \\ \\ 0 \\ \hline 0 \\ \\ \vdots \\ 0 \\ 1 \\ \hline \vdots \\ \hline 0 \\ \vdots \\ \\ \\ 1
\end{bmatrix}
$$

Figure 3.4 : Rows to update in the system of equations due to boundary conditions.

Newton-Raphson Method

Newton-Raphson method is used for finding successively better approximations to the roots of a real-valued function. This method can be implemented for higher dimensions, at which there are several functions of several variables. Newton-Raphson method for higher dimension is given in Eqn.(3.30).

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{J}^{-1}\mathbf{y}_k \tag{3.30}$$

where $\mathbf{x} = [x_1;\ x_2;\ \ldots;\ x_n]$ and $\mathbf{y} = [y_1;\ y_2;\ \ldots;\ y_n]$. $\mathbf{J}$ is the Jacobian matrix and it is given (3.31).

$$\mathbf{J} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} \tag{3.31}$$

4. ENTROPY GENERATION MINIMIZATION

4.1 Motivation and Historical Advance

Entropy generation minimization (EGM) is an interdisciplinary topic merging the ideas in thermodynamics, heat and mass transfer and fluid mechanics as shown by Bejan in Figure 4.1. The aim of the method is to apply thermodynamic optimization to real systems where irreversibilities due to heat transfer, fluid flow and mass transfer exist. The theoretical nature of these fields is tricky to apply on real world problems, thus EGM method is proposed by Bejan to cope with this problem. So EGM with Bejan's own words: "the minimization of thermodynamic irreversibility in real-world applications by accounting for the finite-size constraints of actual devices and the finite-time constraints of actual processes" [78].

1970s witness the emergence of EGM method with the fundamental results via Bejan [78]. Total entropy generation is described as a summation of several levels as shown in Figure 4.2. The indigenous idea behind EGM is to minimize the calculated entropy generation rate. The foundations of the method has laid in graduate study years of Bejan while he realized the promising structure of the method such that it can be applied to a wide variety of application areas. The method experiences a steep growth since then, especially along the years 1980s and 1990s in three main aspects: 1. change in energy policies forcing a growth in engineering, 2. the lifting of the Iron Curtain, 3. new advancements in finite time thermodynamics. The method can be applied to a wide variety of applications ranging from power plant power maximization to power consumption minimization in refrigerators or heat pumps.

EGM method first applied to whole systems assuming that not all the mechanisms are irreversible, ignores some of them and acknowledged the ones with the biggest contribution to the irreversibilities. This led to some complications since it was dependent on the intuition of the person selecting the biggest contribution. This has changed after the method's introduction to heat transfer, and from then on the

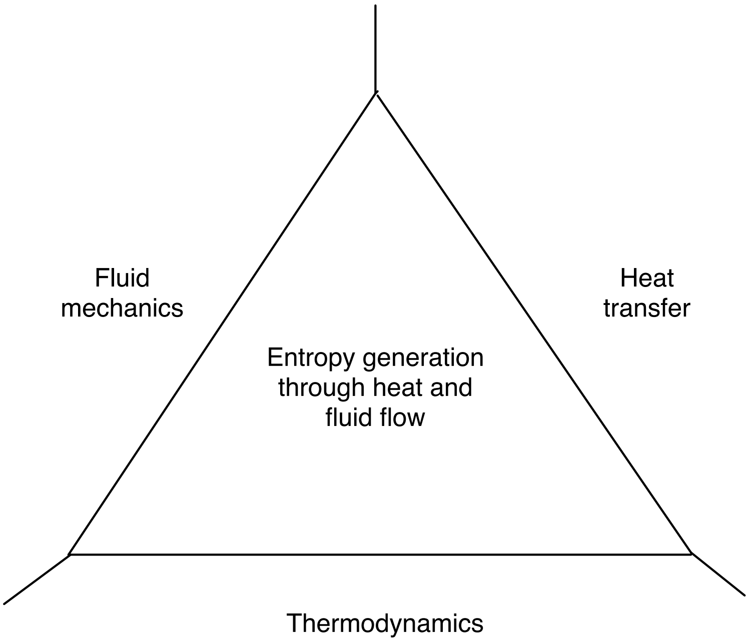

Figure 4.1 : Interdisciplinary Entropy Generation Minimization method is at the intersection of three main fields: fluid mechanics, heat transfer, and thermodynamics [5].

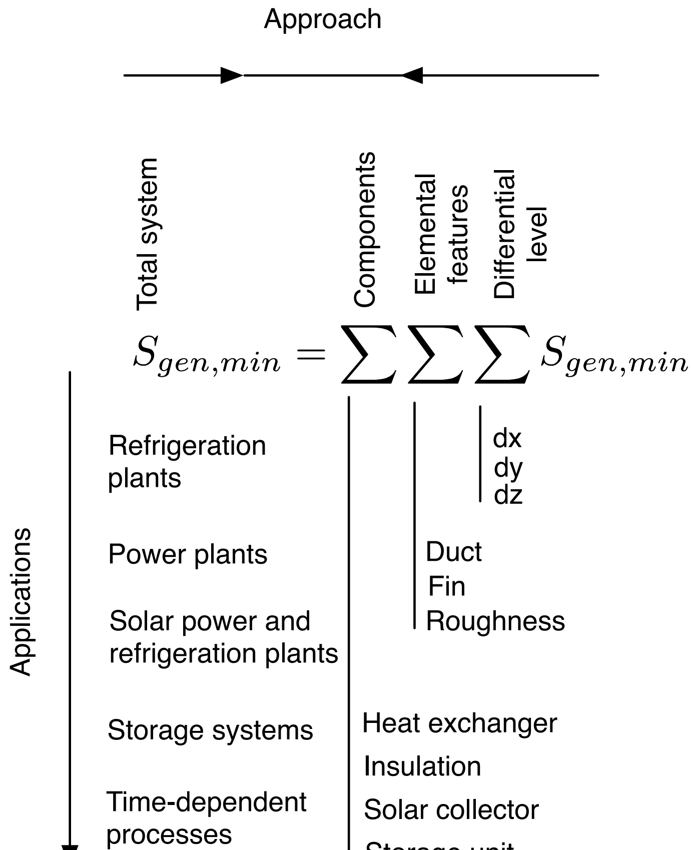

Figure 4.2 : EGM two-dimensional structure [5].

method recognized that the systems are made of actual hardware *components*, each of these components can be made up of smaller *elemental features* whose processes cause irreversibilities at *differential levels* as indicated in Figure 4.2. Figure 4.2 also shows the historical change in approach to the optimized total system, starting from directly focusing on total system as EGM's first applications before its application to heat transfer, or from right to left, indicating first approaching the components of the system. The final approach of the method was to minimize the entropy generation starting from simple level and heading towards more complicated levels.

Heat transfer is always accompanied by entropy generation meaning a destruction in the systems available work, which should be minimized for optimized engineering systems.

In order to understand the general perspective of heat transfer design practice, let's start by giving the net heat transfer rate between to surfaces having temperatures T_1 and T_2

$$Q = \bar{h}A(T_1 - T_2) \tag{4.1}$$

When second law of thermodynamics applied to temperature gap existing in between the systems of different temperatures, entropy generation can be written as

$$S_{gen} = \frac{Q}{T_2} - \frac{Q}{T_1} = \frac{Q(T_1 - T_2)}{T_1 T_2} \tag{4.2}$$

which also shows that the entropy generation is positive as long as there exists the temperature difference.

Heat transfer design objectives can be categorized under two major groups:

- Heat transfer augmentation problem: thermal conductance $\bar{h}A$ is increased to improve thermal contact (lower temperature difference $T_1 - T_2$) since most of the times in this sort of problems Q is given. What can also be seen from Equ. (4.7) is that since $T_1 - T_2$ decreased, entropy generation rate is also decreased.

- Thermal insulation problem: effective thermal conductance $\bar{h}A$ is decreased to decrease the net heat transfer rate Q in a setting where temperatures T_1 and T_2 thus

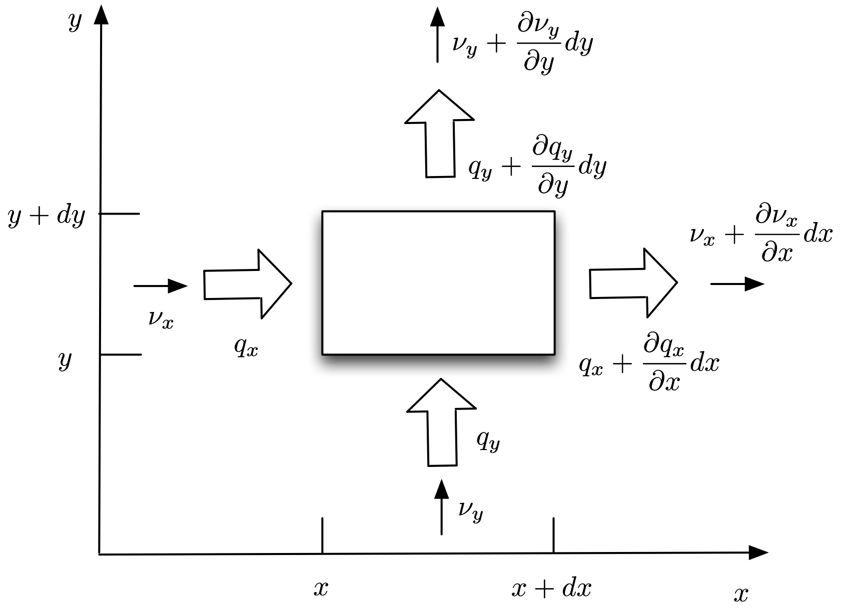
y
$\nu_y + \frac{\partial \nu_y}{\partial y} dy$
$q_y + \frac{\partial q_y}{\partial y} dy$
$y + dy$
$\nu_x + \frac{\partial \nu_x}{\partial x} dx$
ν_x
y
q_x
$q_x + \frac{\partial q_x}{\partial x} dx$
q_y
ν_y
x
$x + dx$
x

Figure 4.3 : EGM two-dimensional structure [5].

their differences are fixed. Again from Equ. (4.7) entropy generation is decreased thanks to decreasing net heat transfer rate Q.

From here on the discussion is narrowed to the topic of entropy generation minimization in convective heat transfer where the irreversibilities are sourced from mainly two effects: heat transfer across a finite (nonzero) temperature difference and fluid friction.

The control surface formed by the *dxdy* rectangle in a fluid engaged in convective heat transfer, as shown in Figure 4.3, is studied as an open thermodynamic system subjected to mass fluxes, energy transfer and entropy transfer interactions. Thermodynamic state of the fluid inside this rectangle is assumed to be uniform, meaning that it is position independent, thanks to the element's small size. But the thermodynamic state of this element may change with time. The fluid is in local thermodynamic equilibrium.

With the assumptions and explanations above, the entropy generation rate per unit volume $\dot{S}'''_{gen}[W/m^3K]$, can be calculated by writing the second law of thermodynamics for the element *dxdy* as

$$
\begin{aligned}
\dot{S}'''_{gen}dxdy = {} & \frac{q_x+\frac{\partial q_x}{\partial x}dx}{T+\frac{\partial T}{\partial x}dx}dy+\frac{q_y+\frac{\partial q_y}{\partial y}dy}{T+\frac{\partial T}{\partial y}dy}dx-\frac{q_x}{T}dy-\frac{q_y}{T}dx \\
& +\left(s+\frac{\partial s}{\partial x}dx\right)\left(v_x+\frac{\partial v_x}{\partial x}dx\right)\left(\rho+\frac{\partial r}{\partial x}dx\right)dy \\
& +\left(s+\frac{\partial s}{\partial y}dy\right)\left(v_y+\frac{\partial v_y}{\partial y}dy\right)\left(\rho+\frac{\partial \rho}{\partial y}dy\right)dx \\
& -sv_x\rho dy-sv_y\rho dx+\frac{\partial(\rho s)}{\partial t}dxdy
\end{aligned}
\tag{4.3}
$$

where the first four terms (first line of the equation) account for the entropy transfer associated with heat transfer, next four represent entropy convected in and out of the system and the last stands for entropy accumulation time rate in the control volume. Local rate of entropy generation can be given via dividing the equation to *dxdy*

$$
\begin{aligned}
\dot{S}'''_{gen} &= \frac{1}{T}\left(\frac{\partial q_x}{\partial x}+\frac{\partial q_y}{\partial y}\right)-\frac{1}{T^2}\left(q_x\frac{\partial T}{\partial x}+q_y\frac{\partial T}{\partial y}\right)+\rho\left(\frac{\partial s}{\partial t}+v_x\frac{\partial s}{\partial x}+v_y\frac{\partial s}{\partial y}\right) \\
&= s\left[\frac{\partial \rho}{\partial t}+v_x\frac{\partial \rho}{\partial x}+v_y\frac{\partial \rho}{\partial y}+\rho\left(\frac{\partial v_x}{\partial x}+\frac{\partial v_y}{\partial y}\right)\right]
\end{aligned}
\tag{4.4}
$$

Due to the mass conservation principle,

$$
\frac{D\rho}{Dt}+\rho\Delta\cdot\boldsymbol{v} \tag{4.5}
$$

where

$$
\frac{D}{Dt}=\frac{\partial}{\partial t}+v_x\frac{\partial}{\partial x}+v_y\frac{\partial}{\partial y} \tag{4.6}
$$

last term in Equ. (4.4) is becomes zero.

$$
\frac{D\rho}{Dt}+\rho\Delta\cdot\boldsymbol{v} \tag{4.7}
$$

Thus volumetric rate of entropy generation can be written as

$$
\dot{S}'''_{gen}=\frac{1}{T}\Delta\cdot\boldsymbol{q}-\frac{1}{T^2}\boldsymbol{q}\cdot\Delta T+\rho\frac{Ds}{Dt} \tag{4.8}
$$

Now, lets put this equation to await a while and write the first law of thermodynamics for a point in the convective medium with assuming the fluid is Newtonian

$$\rho\frac{Du}{Dt} = -\Delta\cdot\boldsymbol{q} - P(\Delta\cdot\boldsymbol{v} + \mu\Phi) \tag{4.9}$$

where μ is the viscosity and Φ is the viscous dissipation function with units $[s^{-2}]$. This equation shows that rate of change in internal energy per unit volume equals to the addition of the net heat transfer rate by conduction, the work transfer rate due to compression, and the work transfer rate per unit volume associated with viscous dissipation.

Giving the canonical relation

$$du = Tds - Pd(1/\rho) \tag{4.10}$$

and using the substantial derivative notation given in Equ. (4.6) and substitute in Equ. (4.9) results in Equ. (4.11)

$$\rho\frac{Ds}{Dt} = \frac{\rho}{T}\frac{Du}{Dt} - \frac{P}{\rho T}\frac{D\rho}{Dt} \tag{4.11}$$

and finally substituting Equ. (4.11) and Equ. (4.9) in Equ. (4.8) results in

$$\dot{S}'''_{gen} = -\frac{1}{T^2}\boldsymbol{q}\cdot\Delta T + \frac{\mu}{T}\Phi \tag{4.12}$$

Now, assuming isotropic medium Fourier law of heat conduction is given as

$$\boldsymbol{q} = -k\Delta T \tag{4.13}$$

and if substituted in Equ. (4.12) results

$$\dot{S}'''_{gen} = -\frac{k}{T^2}(\Delta T)^2 + \frac{\mu}{T}\Phi \tag{4.14}$$

Thus finally the equation reduces to Equ. (4.15) for a two-dimensional cartesian system

$$\dot{S}'''_{gen} = \frac{k}{T^2}\left[\left(\frac{\partial T}{\partial x}\right)^2 + \left(\frac{\partial T}{\partial y}\right)^2\right] + \frac{\mu}{T}\left\{2\left[\left(\frac{\partial v_x}{\partial x}\right)^2 + \left(\frac{\partial v_y}{\partial y}\right)^2\right] + \left(\frac{\partial v_x}{\partial y} + \frac{\partial v_y}{\partial x}\right)^2\right\} \tag{4.15}$$

Local volumetric entropy generation rate formula for one dimensional viscous incompressible conducting fluid flow in the presence of magnetic field is given by [106] as (4.16), indicating each terms irreversibility source

$$E_G = \underbrace{\frac{k}{T_2^2}\left(\frac{dT}{dy}\right)^2}_{heat\_transfer} + \underbrace{\frac{\mu}{T_2}\left(\frac{du}{dy}\right)^2}_{viscous\_dissipation} + \underbrace{\frac{\sigma B_0^2}{T_2}u^2}_{magnetic\_field} \tag{4.16}$$

The last term in the right side of (4.16) is the entropy generation due to magnetic field. Evaluating non-dimensional parameters given in (2.11)-(2.12), (2.39) - (2.41), local entropy generation rate is defined as in (4.17).

$$N_S = \frac{T_2^2 h^2 E_G}{k(T_1 - T_2)^2} = \left(\frac{d\theta}{dY}\right)^2 + \frac{Br}{\Omega}\left(e^{-\varepsilon\theta}\left(\frac{dU}{dY}\right)^2 + HaU^2\right) \tag{4.17}$$

Here Ω and Br stand for the temperature difference parameter and Brinkmann number respectively and can be given as $\Omega = \frac{(T_1 - T_2)}{T_2}$, $Br = Ec\,\mathrm{Pr}$. To show the dominance of irreversibility due to heat transfer with respect to the combined effect of fluid friction and magnetic fields, Bejan number is introduced as follows

$$Be = \frac{N_{heat}}{N_S} = \frac{\left(\frac{d\theta}{dY}\right)^2}{\left(\frac{d\theta}{dY}\right)^2 + \frac{Br}{\Omega}\left(e^{-\varepsilon\theta}\left(\frac{dU}{dY}\right)^2 + HaU^2\right)} \tag{4.18}$$

Be number ranges between 0 and 1 with a meaning of dominance of irreversibility due to heat transfer dominates for the values closer to 1, and the combination of fluid friction and magnetic field for the values closer to 0.

5. SIMULATIONS AND RESULTS

5.1 Channel Flow with Suction-Injection

In this section, simulations are carried out for different parameters of the system via GDQM&NR methods. The velocity and temperature distributions, local entropy generation, total entropy generation and equipartioning are investigated with respect to key system parameters, such as Ha number, variable viscosity parameter, suction/injection parameter. Then, the performance of the GDQM method utilizing different gridding techniques with different grid numbers are investigated in relation to the existing work by using RK method [39]. This study has been submitted to a journal. All analyses are held in MATLAB® simulation environment.

Fig. 5.1 shows the non-dimensional velocity with respect to normalized channel width. Considered parameters for the simulation are taken as $G = 1$, $Br = 0.071$, $Bi_0 = 0.1$, $Bi_1 = 0.1$, $\varepsilon = 0.1$.

The flow velocity field is shown for different Ha and V* numbers. Due to no slip boundary conditions, the velocity is zero at channel walls and it has a parabolic shape along the channel. The velocity is decreased with an increase in Ha number, which represents the increase in magnetic field intensity since Lorentz force is occurring in the reverse direction. This result matches with the literature [65], and the exact solution for the constant viscosity case. The increase in V*, corresponding physically injection at the lower plate and suction at the upper plate, results in a distorted parabolic velocity profile. Along the channel, the maximum value of the velocity moves to the upper plate where suction dominates to flow field. The velocity profiles take comparatively small values for higher values of V*.

Fig. 5.2 represents the 3D velocity values with respect to the non-dimensional channel width, Y and suction injection parameter, V*, under four different Ha number cases. It can be noticed that for smaller values of Ha number, a small change in V* has a dramatic impact on velocity.

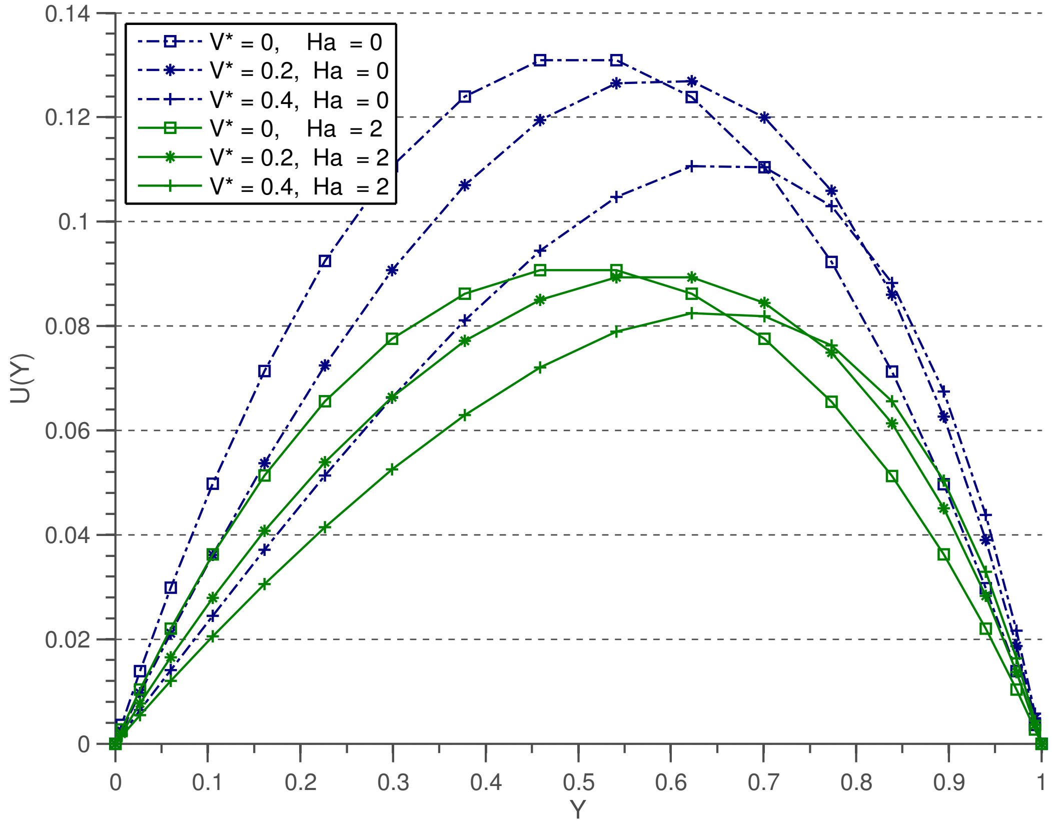

Figure 5.1 : Velocity profile of the system for different Ha and V*.

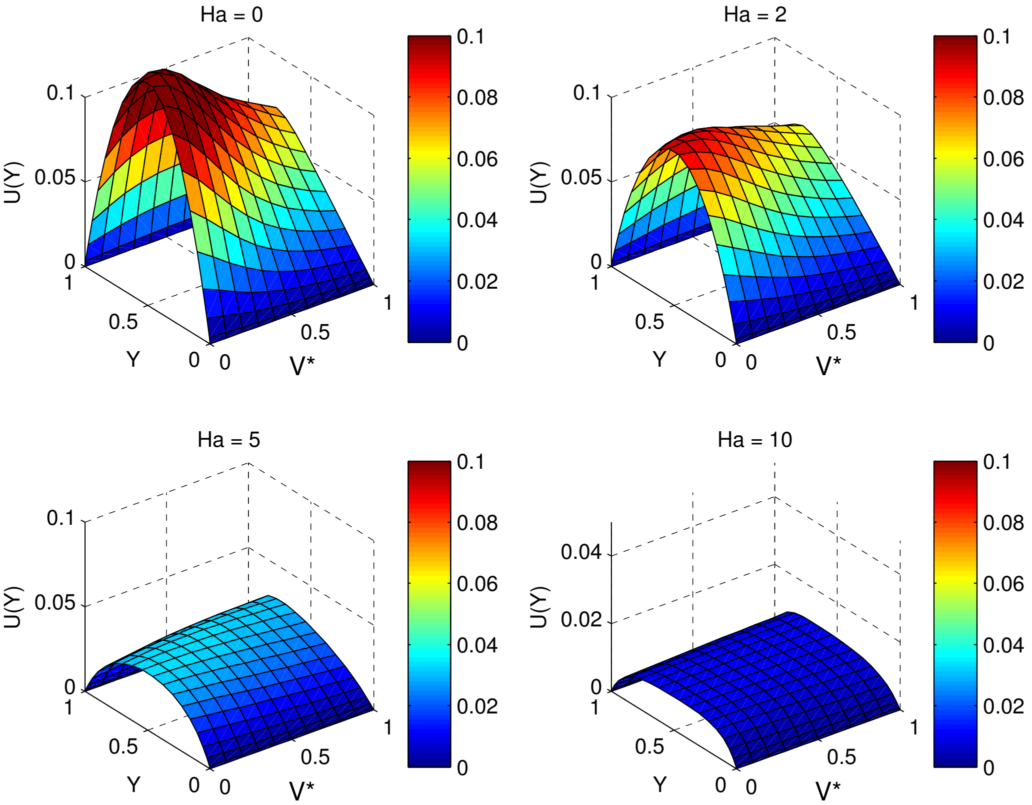

Figure 5.2 : 3D Velocity profile change with V* for Ha = 0,2,5,10.

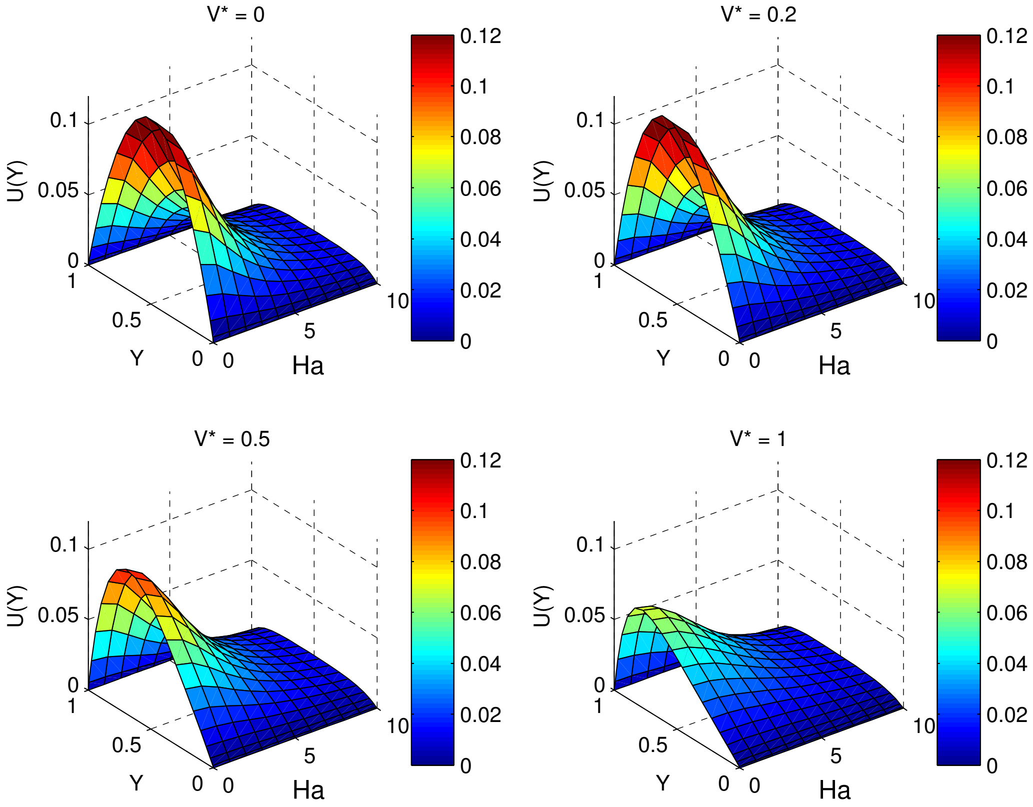

Figure 5.3 : 3D Velocity profile change with Ha for V* = 0.1, 0.2, 0.5, 1.

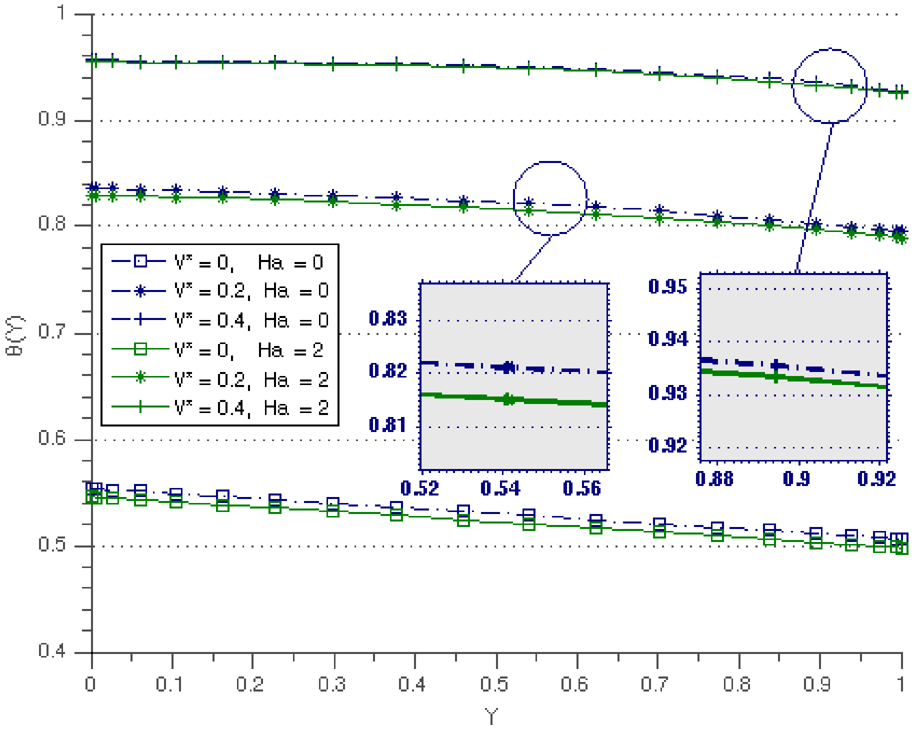

Figure 5.4 : Temperature distribution of the system for different Ha and V*.

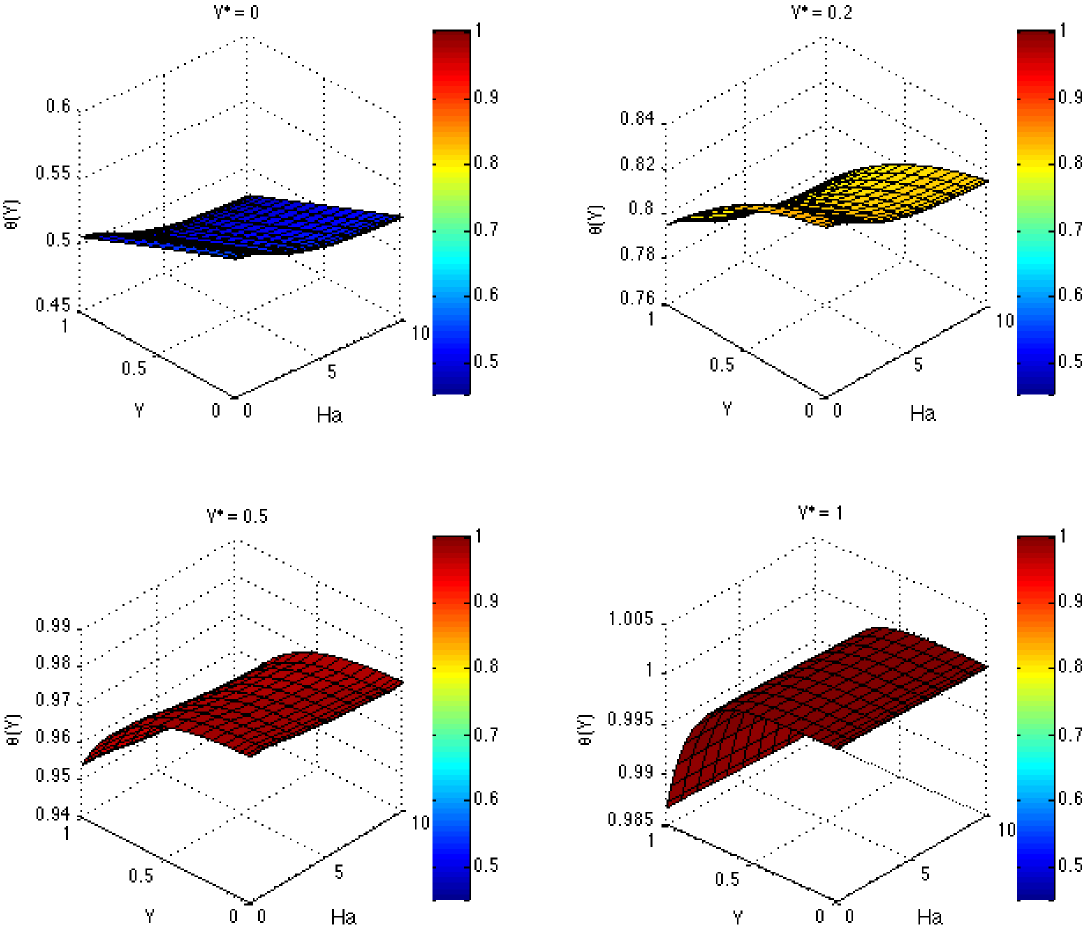

Figure 5.5 : 3D Temperature change with Ha for V* = 0.1, 0.2, 0.5, 1.

3D velocity profile with respect to Y, the non-dimensional channel width, and Ha number for different values of V* are represented in Fig. 5.3.

Ha number's effect on velocity is more dominant for other parameters values of Ha, V*, Re and numbers as seen in Fig. 5.1- 5.3.

Temperature at the lower plate is the highest due to convective heating and decreases gradually to its lowest value at the upper plate due to convective heat loss as forced by the thermal boundary conditions. In Fig. 5.4, the dependence of temperature distribution on Ha and V* is displayed. Temperature is decreased with an increase in Ha number. On the contrary, increase in V* increases the temperature of the fluid. The change of V* has a stronger influence than Ha number on temperature change for the selected values of Ha, V*, Re and ε numbers.

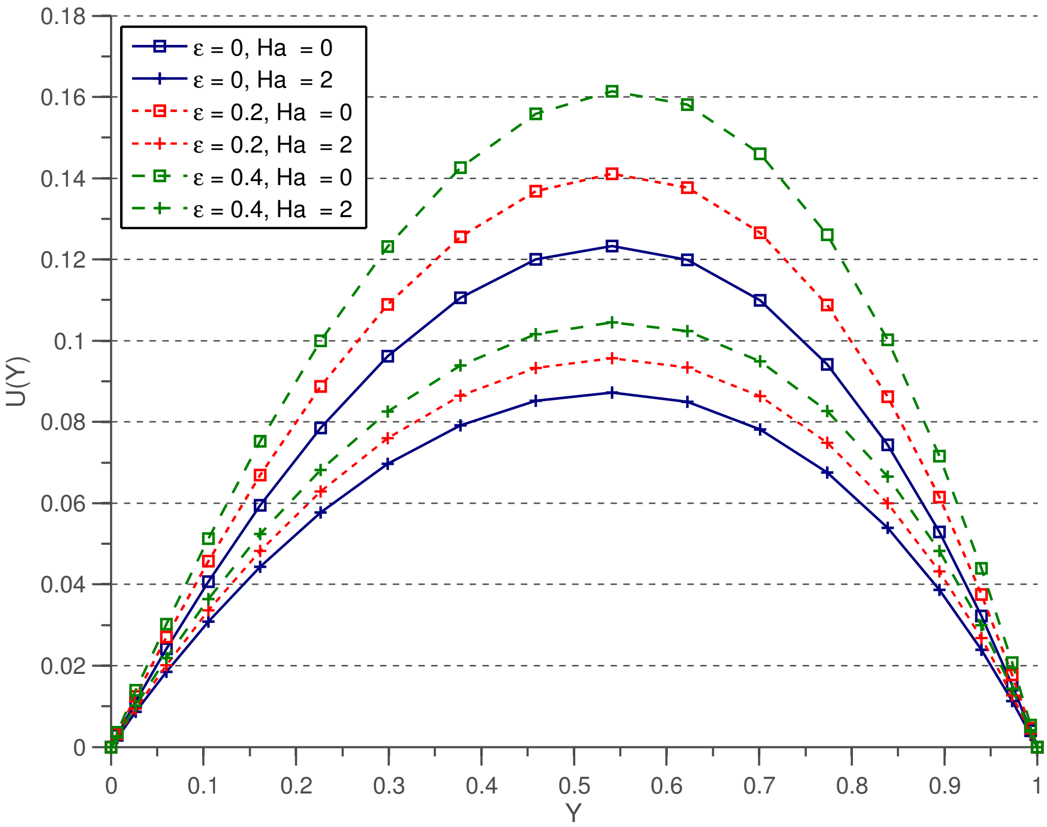

Figure 5.6 : Velocity profile of the system for different Ha and ε.

Fig. 5.5 represents the 3D view of temperature change across the channel width, Y, with respect to Ha number for different V* values. Nearly linear trend in the temperature change along channel width, evolves to nonlinear shape for increasing V* as seen in Fig. 5.5. On the contrary, temperature becomes more stable with respect to Ha number as V* increases.

Fig. 5.6 shows the velocity profiles for different values of variable viscosity parameter, ε , subjected to a variation of magnetic field intensity values. is defined in (**??**) and $\varepsilon = 0$ corresponds to the constant viscosity case. An increase in ε results in an increase in fluid velocity as seen in Fig. 5.6. The suppression effect of Ha number on the velocity field can be easily noticed for the higher values of ε .

Fig. 5.7 shows that the temperature of the fluid increases with an increase in variable viscosity parameter ε . The suppression effect of Ha number on the temperature can be noticed for higher values of ε.

To design optimal engineering systems, the performance of the process of engineering system should be quantified. One way is to measure entropy generation and minimize it

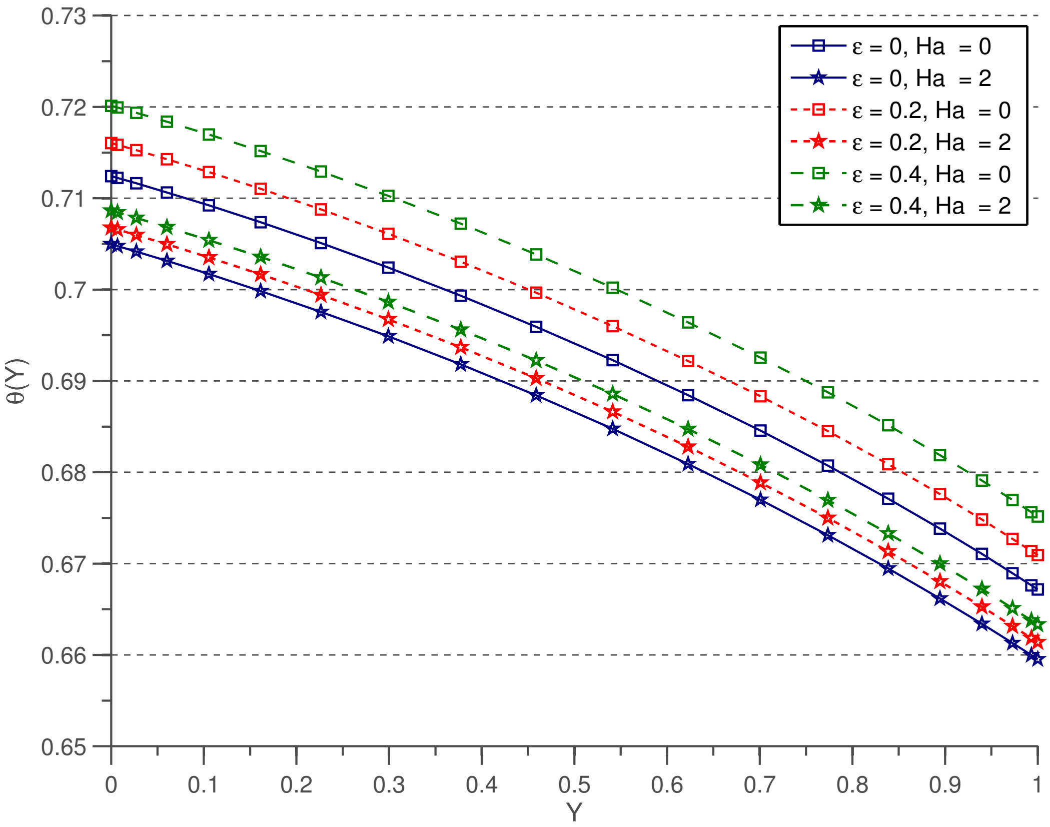

Figure 5.7 : Temperature distribution of the system for different Ha and ε.

for energy efficiency. For that purpose, dominating factors in entropy generation must be determined and eliminated. The convection process in such a power generating fluidic system is inherently irreversible. Using the local non-dimensional entropy generation rate formula for a viscous incompressible conducting fluid in the presence of magnetic field is given in (4.17), local entropy change is given in Fig. 5.8 for a variety of Ha numbers and V*.

From Fig. 5.8, it can be seen that for V* = 0, local entropy generation decreases for higher values of Ha in the channel near the wall regions. In the middle of the channel, local entropy generation is not much dependent on the magnetic field intensity change but it slightly takes higher values for increasing Ha numbers.

These locations of the nearly constant entropy generation values for different Ha numbers and the behavior of the local entropy generation curves are distorted by changing V*. Constant entropy generation locations get nearer to the wall with suction as in Fig. 5.8 for V* = 0.2. A further increase entropy generation to have higher

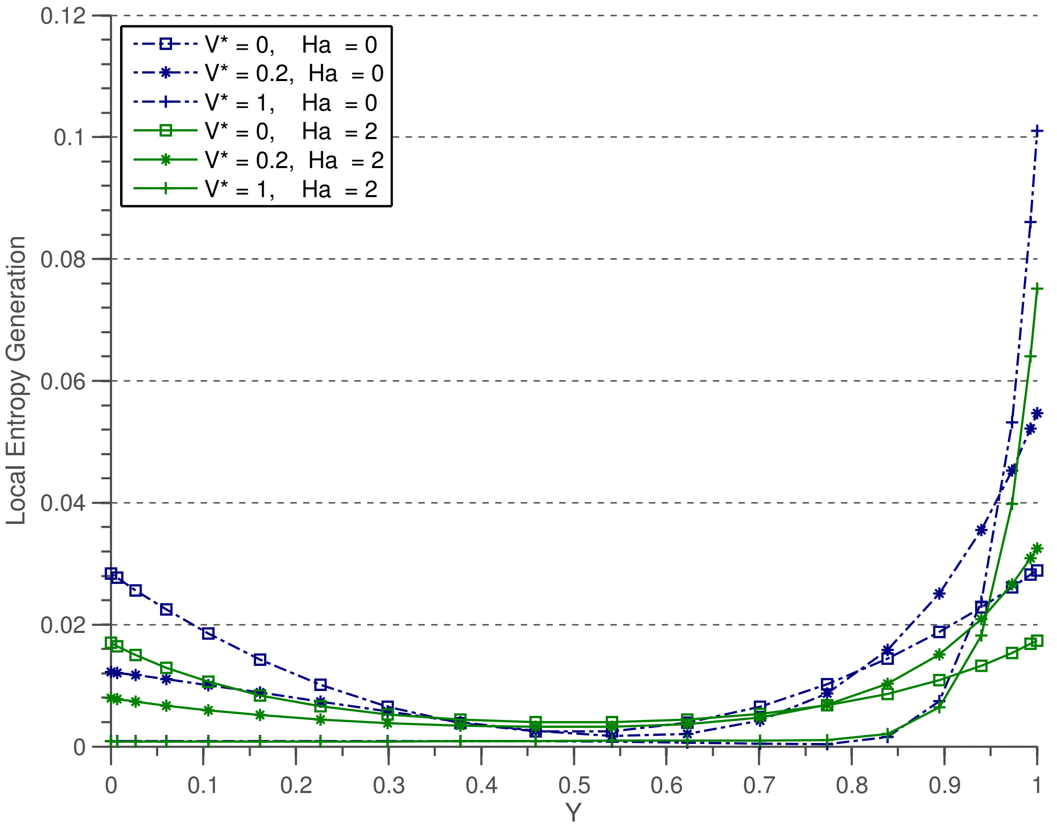

Figure 5.8 : Local Entropy generation for different Ha and V*.

values near the wall and lower values, nearly insensitive to Ha change, near the wall with injection as seen in Fig. 5.8.

Fig. 5.9 presents the total entropy generation which consists of viscous dissipation, magnetic field, and heat transfer terms. Entropy generation changes with Ha and V*. To obtain an efficient system, an optimization based on these variables is needed.

The total entropy generation and three terms contribution for a variety of Ha and variable viscosity parameter, ε, are given in Fig. 5.10. When Ha number takes greater values than 5, the contribution of V* to entropy generation is minimum whatever the value V* takes.

For a maximum efficient system, the entropy generation in the system must take its minimum value. When the system takes this minimum value, a physical parameter, in this case is Ha number, controlled equipartioning phenomenon is occurred, which means that all entropy generation mechanism, fluid friction irreversibility, heat transfer irreversibility and magnetic field irreversibility, generates system entropy generation rate. The optimization of the entropy production with parameters leads to equapartition

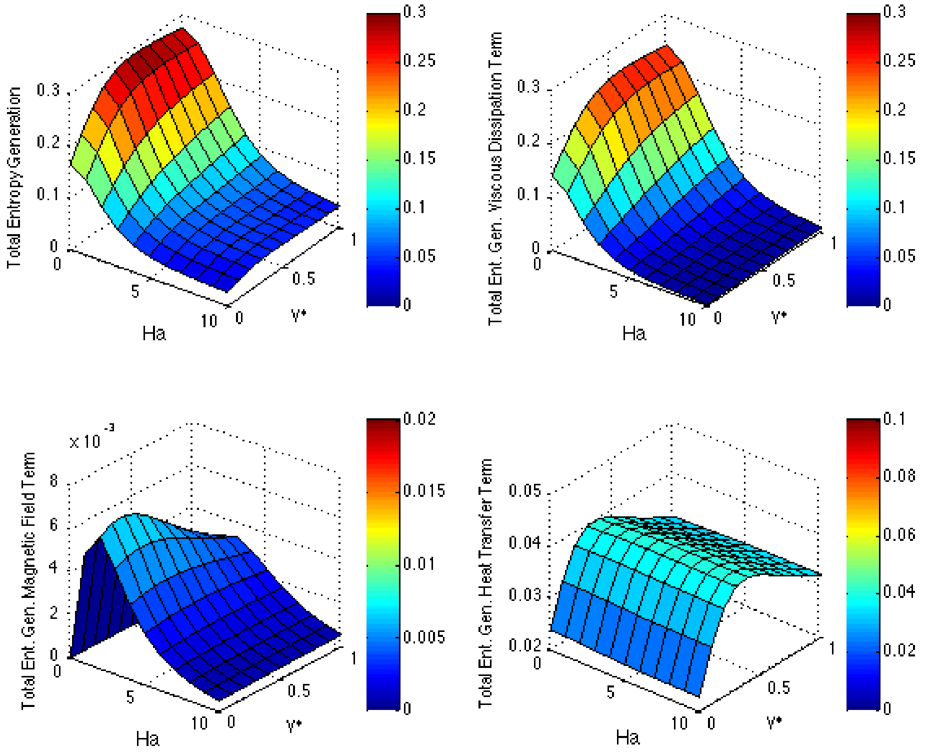

Figure 5.9 : Total Entropy generation and its components for different V* and Ha.

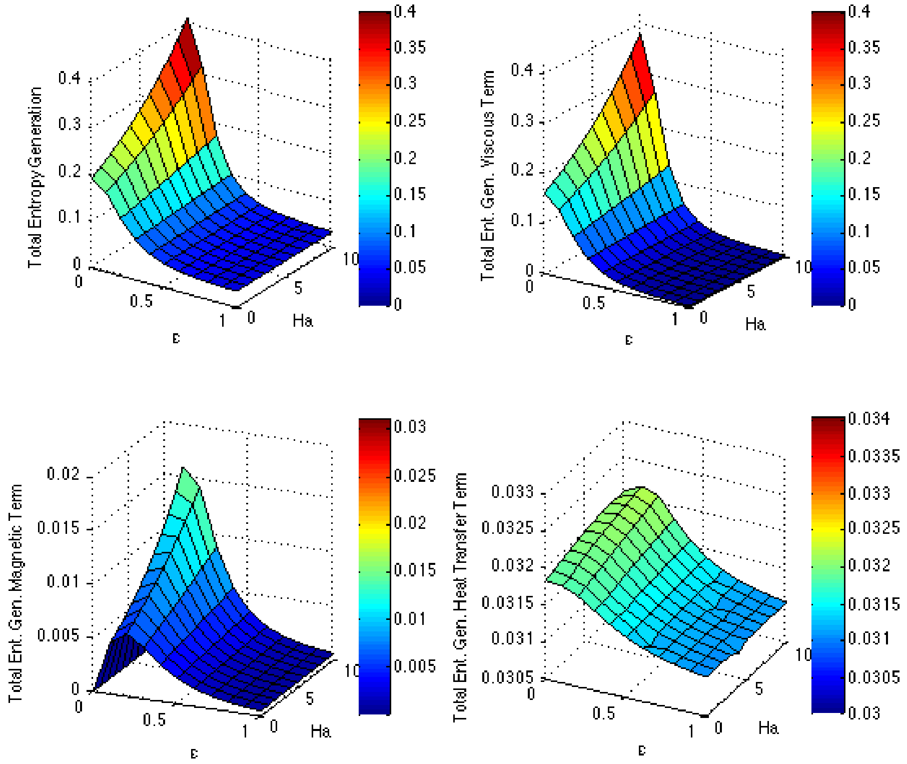

Figure 5.10 : Total Entropy generation and its components for different V* and Ha.

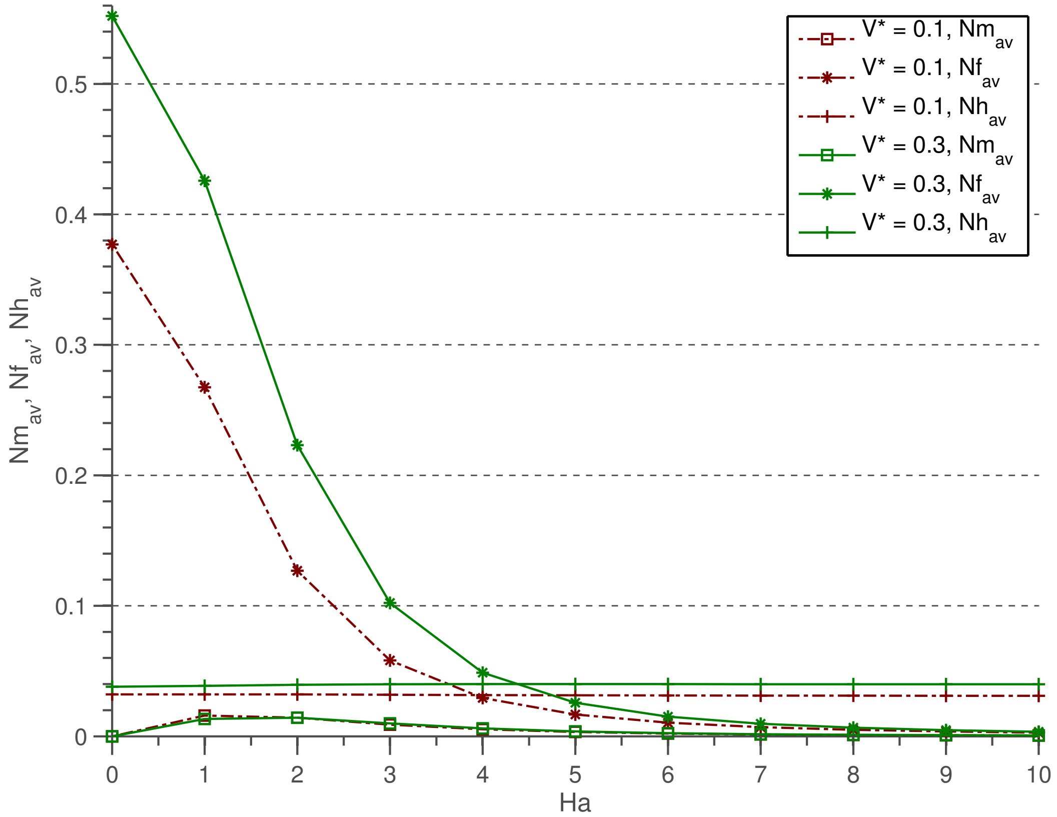

Figure 5.11 : Equipartitioning for different Ha and V*.

of entropy contributions in the system. Equipartioning phenomenon between entropy generation mechanisms due to fluid friction irreversibility, heat transfer irreversibility and magnetic field irreversibility can be seen in Fig. 5.11 [56, 118].

GDQM is used for the discretization of the governing equations and Newton-Raphson method is utilized for the solution of this discretized equation system. Obtained results are given in comparison with [65] in which the 4^{th} order RK method was used. Table 5.1 shows that, GDQM results in more accurate solutions than 4^{th} order RK when compared with the analytical solutions for the constant viscosity case, $\varepsilon = 0$. Error percentages of GDQM and 4^{th} order RK method available in [65] are given in Table 5.1 at each 11 equal grid points.

Table 5.2 shows that, numerical error can further be reduced with the appropriate selection of gridding type. Two different types of grid distribution selected for this numerical study are equal gridding and Chebyshev-Gauss-Lobatto gridding. It can be seen that 2 - norm and inf - norm of error are reduced with Chebyshev-Gauss-Lobatto gridding for this problem.

Table 5.1 : Comparison of the velocities solved via Analytical Solution, Generalized Differential Quadrature Method (GDQM) combined with Newton-Raphson (NR) and Runge Kutta (RK).

Y	Exact Sol	GDQM & NR	GDQM & NR Error	RK	RK Error
0	0	0	N/A	0	N/A
0.1	0.03582281906	0.03582281909	6.48581E-10	0.035822	N/A
0.2	0.0652643652	0.0652643653	2.80919E-10	0.065264	N/A
0.3	0.08796341184	0.08796341185	1.62283E-10	0.087963	N/A
0.4	0.1034497707	0.1034497708	9.43549E-11	0.103449	N/A
0.5	0.111127884370	0.111127884374	4.42553E-11	0.111127	0.001367804
0.6	0.1102573709647	0.1102573709643	-3.51006E-12	0.110257	N/A
0.7	0.09993000073	0.09993000072	-6.2924E-11	0.09993	N/A
0.8	0.07904248292	0.07904248291	-1.608E-10	0.079042	N/A
0.9	0.04626433809	0.04626433807	-4.55426E-10	0.046264	N/A
1	0	0.03582281909	N/A	0	N/A

Table 5.2 : Error norm comparison of solutions via GDQM&NR and RK. 11 grids are evaluated.

	2-Norm Relative Error	Inf-Norm Relative Error
GDQM&NR (CGL gridding)	1.861398109731811E-12	2.164441621907274E-12
GDQM&NR (equal gridding)	1.660858566579642E-10	2.090674898860163E-10
RK (equal gridding) [65]	5.878346513550312E-04	0.001367804

Table 5.3 : Error norm comparison for RK and changing grid number/type in GDQM.

	2-Norm Relative Error	Inf-Norm Relative Error
GDQM&NR (CGL 7 Grids)	9.940789071349217e-07	1.066009063843525e-06
GDQM&NR (CGL 3 Grids)	1.509365451917393e-04	1.509365451917393e-04
GDQM&NR (Equal 7 Grids)	8.976353133050371e-06	1.103889549073898e-05
RK (Equal 11 Grids) [65]	5.878346513550312E-04	0.001367804

One of the main advantages of GDQM method is its strength to converge to accurate solutions with a few grid points. Table 5.3 shows the utilization of a few grid points results in accurate solutions. From Table 5.3, it can be seen that with only 3 grids in GDQM, the error in the solution is less than RK method with 11 grids. Table 3 also indicates that the accuracy is further increased with a higher number of grids and CGL gridding.

To conclude, steady viscous incompressible flow of an electrically conducting fluid between two porous plates under magnetic field is studied. A combination of GDQM & NR methods is applied to obtain velocity and temperature fields in the channel having constant injection and suction walls. The velocity, temperature fields, local entropy generation rate and total entropy generation are investigated depending on physical parameters such as Ha number, Suction/Injection parameter, V*, function of viscosity variation, ε.

Then, the performance of GDQM method is examined for equal gridding, CGL gridding techniques and different number of grids. The results compared with existing work using Runge-Kutta (RK) method. GDQM is a strong semi numerical solution tool for discretization of partial derivatives. Combining GDQM and Newton-Raphson Method gives more accurate results than solutions in the open literature via RK Method. It is also noticed that with a few grid points in GDQM solution technique, the results obtained are accurate enough. The inherent success of GDQM to converge to accurate solutions using a few number of grid points is also observed in the simulations and pointed out in tables. The prime importance of this study, though many similar studies are exist in open literature, is the application of GDQM method to magnetic field effected channel-flow type problem.

Investigation of entropy generation and equipartition constitutes another main argument of this study. Quantification of the destruction of system's available work is a key issue in the design of efficient system. The contribution of each component of entropy generation mechanisms, magnetic field irreversibility, heat transfer irreversibility, fluid friction irreversibility is discussed. Their variation with flow parameters such as Ha number and V* number are investigated. An equipartion phenomenon, equal distribution of entropy generation contributions, is observed between magnetic field irreversibility, heat transfer irreversibility and fluid friction

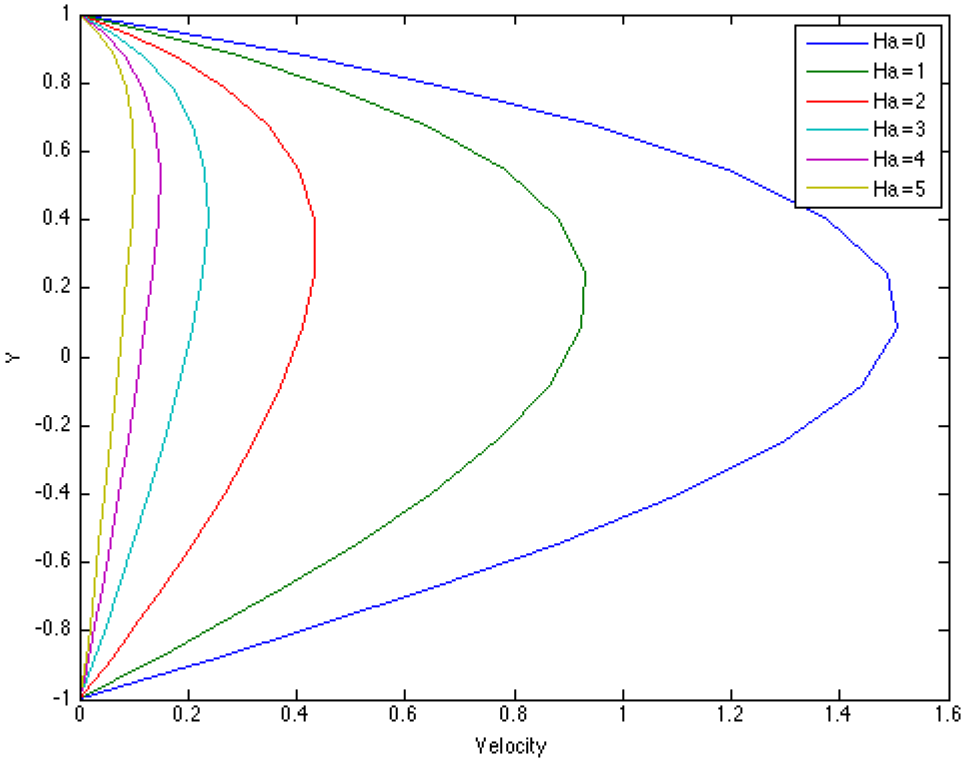

Figure 5.12 : Velocity profile dependence to magnetic field.

irreversibility for higher values of Ha numbers. The variation of total entropy generation with flow parameters such as Ha number and V* number, is not linear, thus minimizing entropy generation should be handled via an optimization algorithm.

5.2 Inclined Nanofluid Channel Flow

In this section, the effect of parameters on thermal distribution and velocity profile are analyzed utilizing MATLAB. The particle volume fraction ψ is changed from 0 - 0.06, where $\psi = 0$ corresponding to the pure base fluid case without nanoparticles. Magnetic field effect on the flow is also investigated through changing the dimensionless Ha number. The inclination angle dependence is briefly pointed. This part of the study is based on the work done in the author's paper [7].

Figure 5.12 shows the effect of Hartmann number on the velocity profile. Considered parameters for the simulation given in Fig 5.12 and Fig 5.13 are $P = 1$; $Br = 0.1$; $Ra = 10$; $Pr = 6.7$; $\psi = 0.06$; $\phi = pi/6$. As the Ha number increase the flow field is suppressed significantly due to the strong retarding effect of Lorentz force. This is

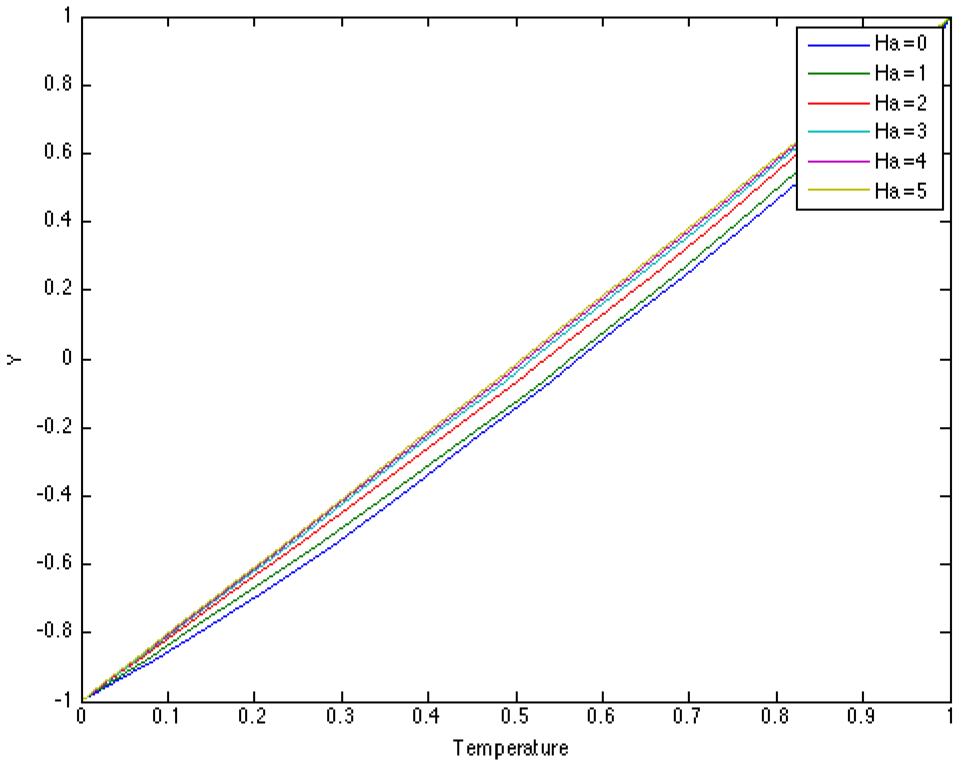

Figure 5.13 : Temperature distribution dependence to magnetic field.

also the case for fluids without nanoparticles but results show that for a concentration of $\psi = 0.06$, the magnetic field braking effect increases when compared to the case of no particles, i.e $\psi = 0$. Figure 5.13 also shows the temperature tendency vary almost linearly and decrease with increasing magnetic field intensity.

Figure 5.14 shows the velocity profile dependence on the inclination angle. Considered parameters for the simulation given in Fig 5.14 and Fig 5.15 are $Ha = 1$; $P = 1$; $Br = 0.1$; $Ra = 10$; $Pr = 6.7$; $\psi = 0.06$.The simulation converged for the inclination angle range $\phi = 0$ - $\phi = 60$ degrees but for larger angle values it diverged. The velocity dependence to inclination decreases with the addition of nanoparticles. The inclination dependence decreases as magnetic field increases when there is no particles, i.e $\psi = 0$. In nanofluid flow, the inclination dependence possess a similar tendency but not as strong as in clear water case. Figure 5.15 also shows that the temperature distribution is strongly dependent to inclination angle, temperature increases with increasing inclination angle.

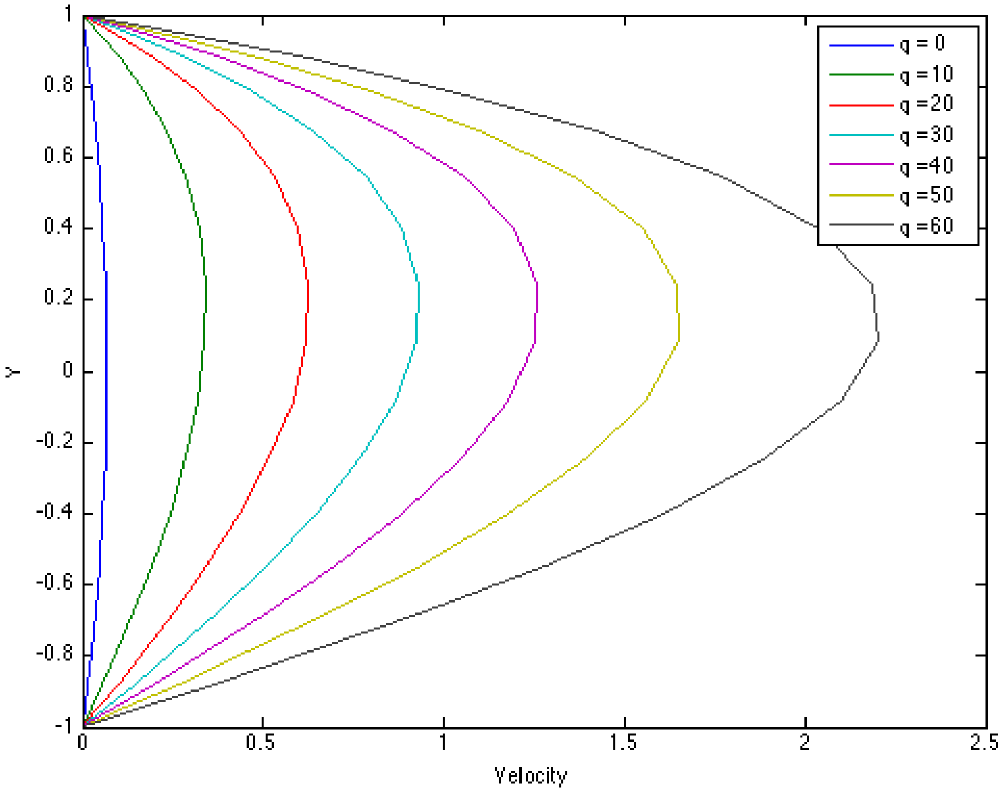

Figure 5.14 : Velocity profile dependence to inclination angle.

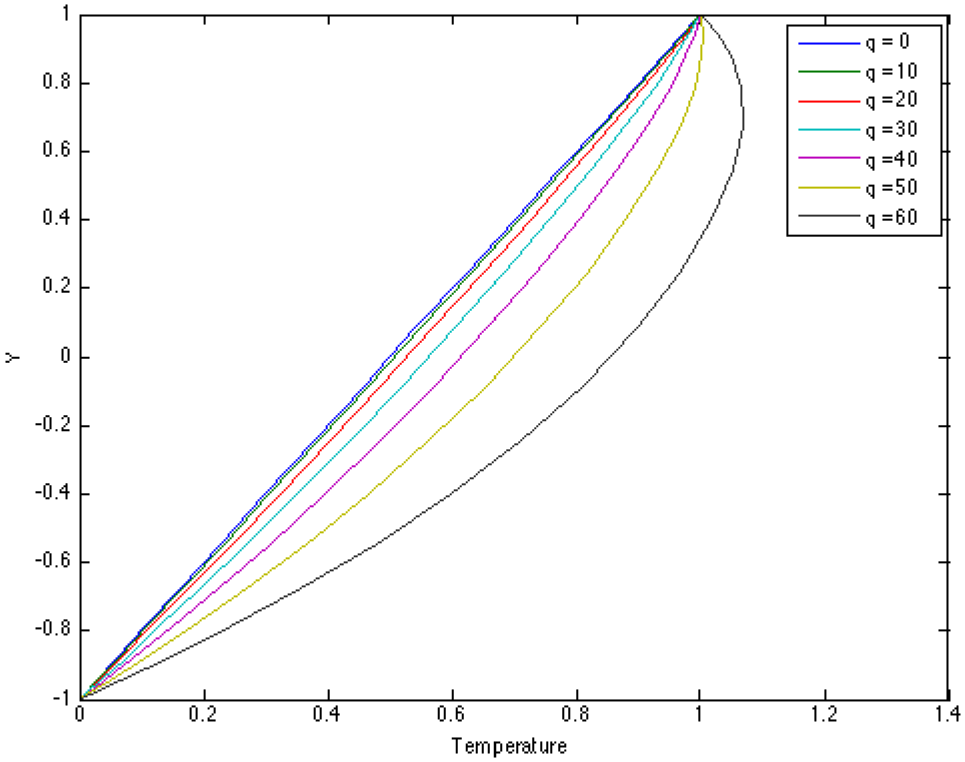

Figure 5.15 : Temperature distribution dependence to inclination angle.

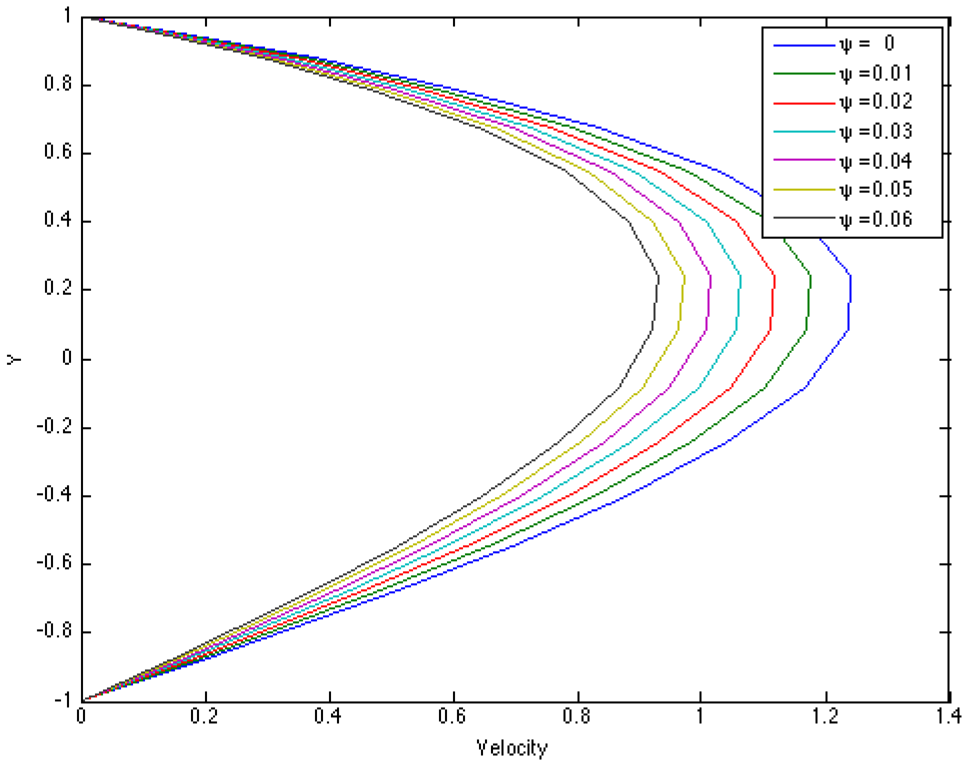

Figure 5.16 : Velocity profile dependence to nanoparticle volume fraction.

Figure 5.16 shows the velocity profile dependence on the volume fraction of nanofluids. Considered parameters for the simulation given in Fig 5.16 and Fig 5.17 are $Ha = 1$; $P = 1$; $Br = 0.1$; $Ra = 10$; $Pr = 6.7$; $\phi = pi/6$. The velocity decreases with an increase in volume fraction of nanofluid. When magnetic field gets stronger, the volume fraction dependence of the velocity also increases and gets more dependent to temperature distribution. The temperature distribution, given in Fig. 5.17, is not very effected by volume fraction and this dependence gets even less with increased magnetic field magnitude.

5.3 Inclined Nanofluid Channel Flow Exposed to Oriented Magnetic Field

In this section, simulations are carried out for a variety of parameters and solved via GDQM & NR methods. The velocity and temperature distributions, local entropy generation and total entropy generation are investigated with respect to key system parameters, such as magnetic field angle, inclination of the channel, Ha number and nanoparticle volume fraction. The codes are implemented in MATLAB environment. This part of the study is based on the work done in the author's paper [8].

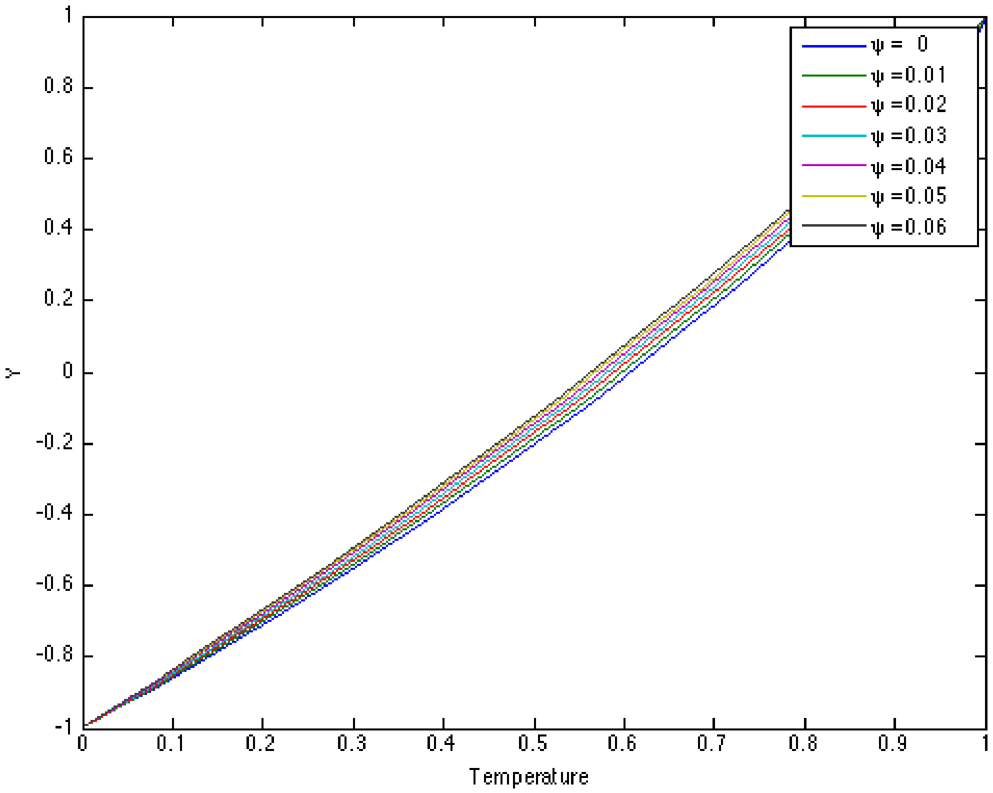

Figure 5.17 : Temperature distribution dependence to nanoparticle volume fraction.

Since the numeric solution is designed in-house, the need to verify its validity, a simpler geometry is refered where numerical results are easily available in the literature and simplifies to an analytical solution under some conditions. This study adopts a suction-injuction channel flow without nanofluids. The results show that DQM tool implemented in-house gives even more accurate results then Runge-Kutta solutions given in [119] for the specific case where an analytical solution is also available for comparison [120]. Furthermore, another strong feature of DQM is observed, ability to give relatively accurate solutions for few number of grids. Further verification has been proceed with a comparison with the literature for trends of change of velocity and temperature profiles while parameters of the flow is changed such as Ha number.

This problem is designed such that the user of the system would have more parameters to configure the flow. One of the main controllable parameter is angle of the magnetic field. Throughout this study, the direction of the magnetic field is not assumed to be perpendicular to the channel walls while the inclination of the channel is changing. This configuration might have been helpful to have an insight for the applications

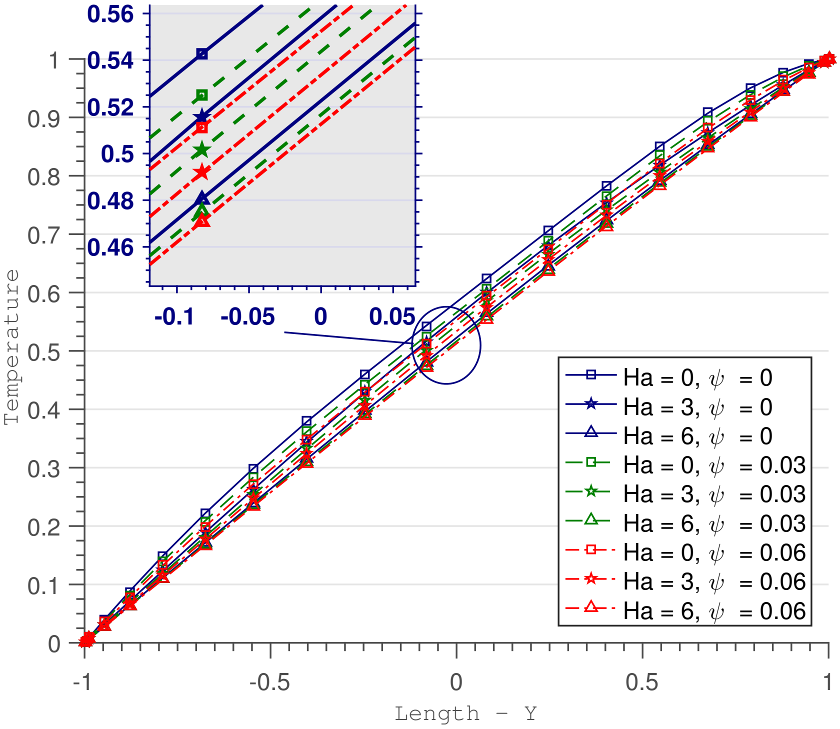

Figure 5.18 : Nondimensionalized temperature change with channel width for a variety of magnetic field intensity and solid volume fraction.

where inclination of the channel could not be changed but the magnetic field angle could be changed.

First of all, effect of magnetic field magnitude (discarding the direction) on the flow is investigated by changing the dimensionless Ha number but keeping the magnetic field direction constant. Considered parameters for the simulation given in Figures 5.18 and 5.19 are $P = 1$; $Br = 0.1$; $Ra = 10$; $Pr = 6.7$; $\alpha = 10$; $\phi = 30$. Figures 5.18 and 5.19 also shows the effect of nanoparticle volume fraction on velocity and temperature profiles. By letting more particles to the system or discarding some, the flow can be supervised. In the simulations, the particle volume fraction is changed between ψ = 0–0.06 from where $\psi = 0$ corresponding to the pure base fluid case without nanoparticles. The velocity decreases with an increase in volume fraction of nanofluid.

Figure 5.19 shows the effect of Hartmann number on the velocity profile. It is clear that, increasing the value of Ha have a tendency to slow down the fluid motion because of the presence of the transverse magnetic field. Transverse magnetic field creates a resistive force similar to the drag force, which acts in the opposite direction of the fluid motion. The velocity of the fluid decreases with resistive force and it is clear that the applied magnetic field has a retarding effect on the flow field. This is also the case for

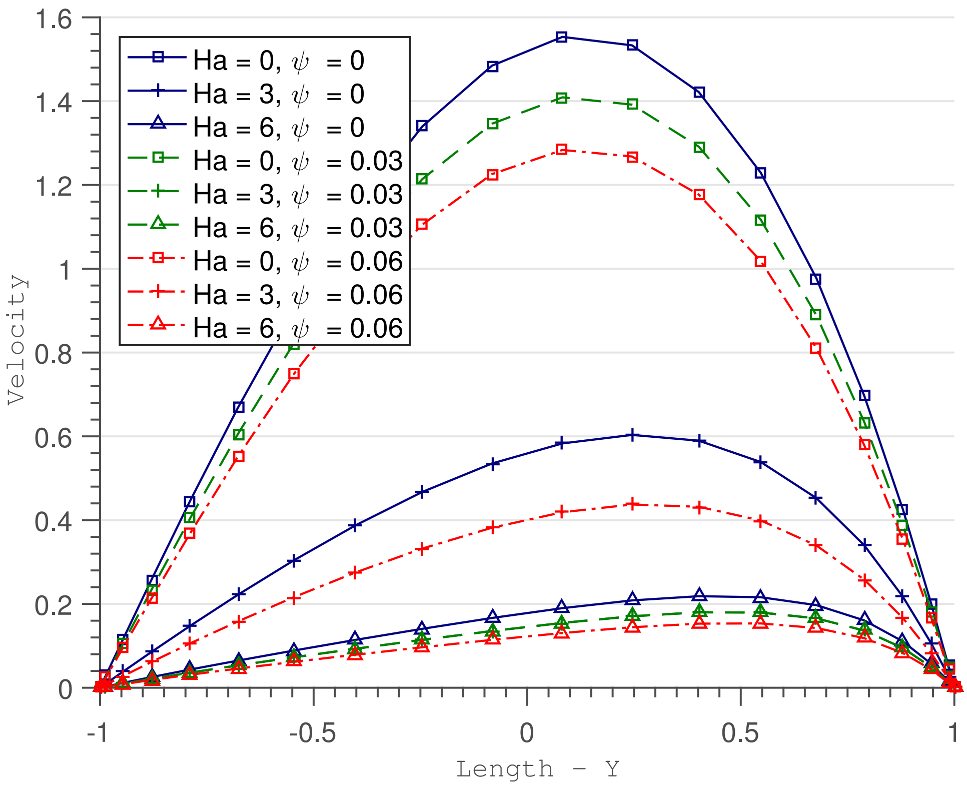

Figure 5.19 : Nondimensionalized velocity change with channel width for a variety of magnetic field intensity and solid volume fraction.

fluids without nanoparticles but results show that for a concentration of $\psi = 0.06$, the magnetic field braking effect increases when compared to the case of no particles, i.e., $\psi = 0$.

Figure 5.18 shows the temperature variation is almost linear with respect to non-dimensionalized channel width and decrease with increasing magnetic field intensity. The fluid temperature decreases increasing the value of Ha within the channel due to increasing magnetic field intensity. This behavior is attributed to decrease the fluid velocity due to the magnetic field.

As mentioned above, the main parameter to be utilized in order to achieve more dominance over the behavior of the flow is the magnetic field angle (α). To further differentiate the effect of magnetic field and gravitation over the flow, the magnetic field angle is offered to be independent from the angle of the channel. This could be a good modeling approach for the problems where the channel has a fixed inclination while the magnetic field can be directed independently. An example case could be an medical application where the veins should not be moved while the magnetic field applied from outside could be changed in favor of the application.

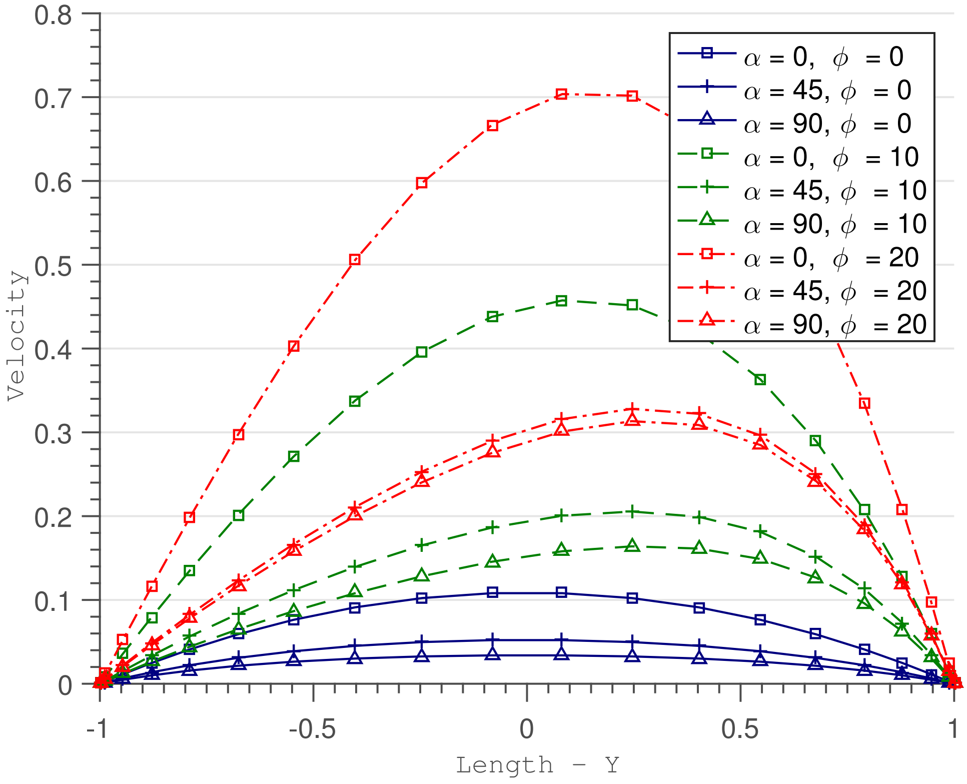

Figure 5.20 : Nondimensionalized velocity with respect to magnetic field angle α and channel inclination ϕ.

Figure 5.20 shows the behavior of the velocity along the channel for a variety of magnetic field and channel inclination values for $\psi = 0.06$ and $Ha = 2$. The rise in the inclination causing an increase in the flow velocity is observed, obviously matching the common sense. The velocity profiles are unsymmetrical about the centerline of the channel due to inclination angle as expected. It also shows that when the inclination of the magnetic field increased, the flow velocity decreases. A key point to realize is that the flow velocity can be decreased to a level less then even a higher inclination of the channel velocity profile by changing the magnetic field angle less then 45. This can be seen clearly from the curves representing $\alpha = 0$, $\phi = 10$ and $\alpha = 45$, $\phi = 20$ in Figure 5.20. This gives the user of the channel a good dominance to control the flow as is intended. A second point in Figure 5.20 is that for higher values of magnetic field angle, the less change will be observed in the velocity change corresponding to a change in magnetic field angle. This can be observed by checking the same color curves, and realize that the difference of flow velocity suppression is higher in the change from $\alpha = 0$ to $\alpha = 45$ compared to the change in $\alpha = 45$ to $\alpha = 90$. The flow velocity supression in higher magnetic field inclinations also decreases for

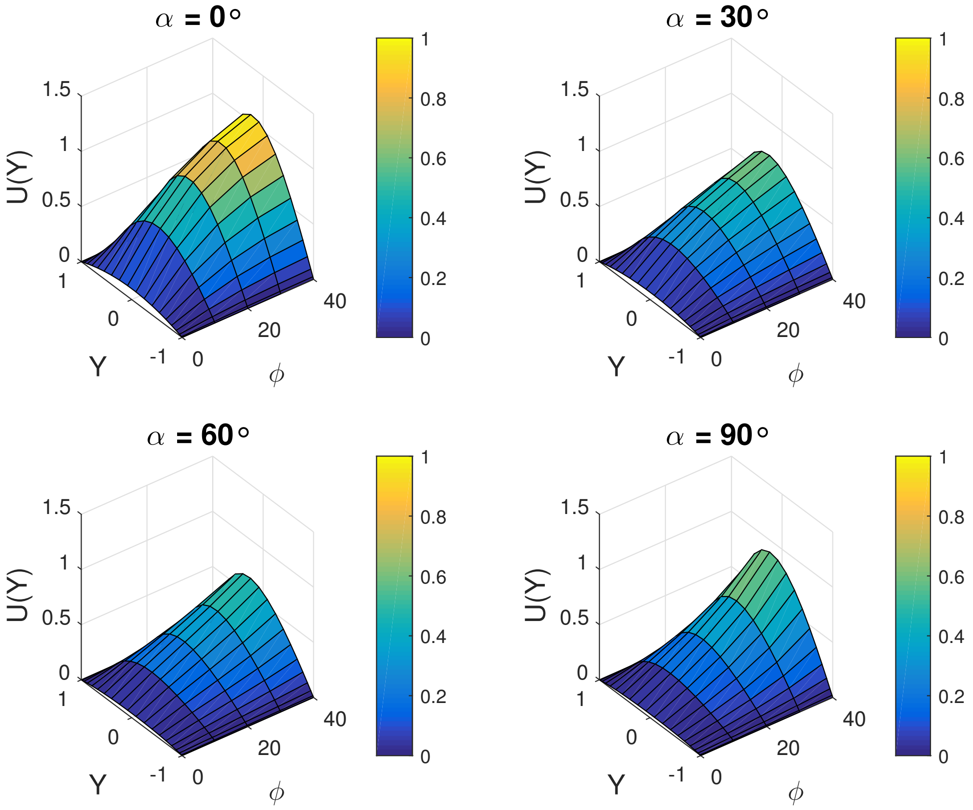

Figure 5.21 : 3D flow behaviour surface for nondimensionalized velocity with respect to channel inclination ϕ for four different magnetic field angle α.

higher channel inclination. This can be seen by observing the decrease in velocity for $\alpha = 45$ to $\alpha = 90$ for $\phi = 20$ is less than the decrease in velocity for $\alpha = 45$ to $\alpha = 90$ for $\phi = 10$. Another result is that these behaviour does not change for different nano-particle volume fractions ψ but not showed here due to page limitations.

Figure 5.21 is a broader look at the trend of the effect of α and ϕ on flow velocity, by showing 3D figures representing the change of flow velocity with respect to inclination of the channel ϕ for four different magnetic field angle α. The results discussed until now can be seen in this figure as well as a trend in change in shape of the surface for different values of magnetic field angles α.

Engineering system design is a complicated process and most of the times requires comprise from one property or another. A wise choice to tune in between favorable properties, there are various optimization methods. One of them is to calculate and minimize the entropy generation. The influences of the different parameters on entropy generation within the channel are presented in Figures 5.22–5.25. It is seen from Figures that the entropy generation depends on magnetic field angle, channel

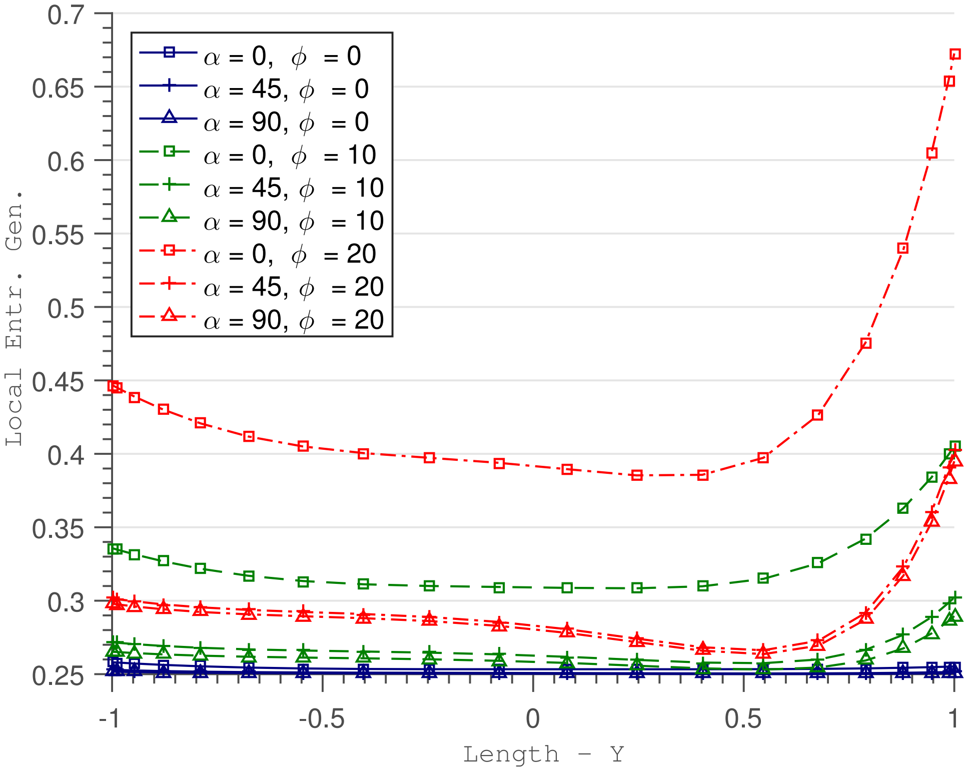

Figure 5.22 : Local entropy generation as a function of channel width for specific magnetic field angle α and channel inclination ϕ.

inclination. Using the local non-dimensional entropy generation rate formula for nanofluid in the presence of magnetic field is given in Equation (**??**), local entropy generation for a variety of α and ϕ for $Ha = 2$, $\psi = 0.06$ and given in Figure 5.22. It shows that the entropy generation trend changes heavily for different channel inclination angle ϕ. A change in magnetic field angle α may also effect the entropy generation abruptly. When the magnetic field angle is increased, a decrease in entropy generation is observed in Figure 5.22. But this effect gets nearly negligible for higher values of magnetic field angle. This can be checked by investigating the distance in between the local entropy generation curves for $\alpha = 0$–45 and $\alpha = 45$–90. Furthermore, addition of nanoparticles decrease the local entropy generation but does not change the effect of α and ϕ on the local entropy generation. Here only the trend for $\psi = 0.06$ is given but in the course of the study, the effect of α and ϕ on the local entropy generation is investigated for $\psi = 0, 0.03, 0.06$ but not all given due to the page constraints.

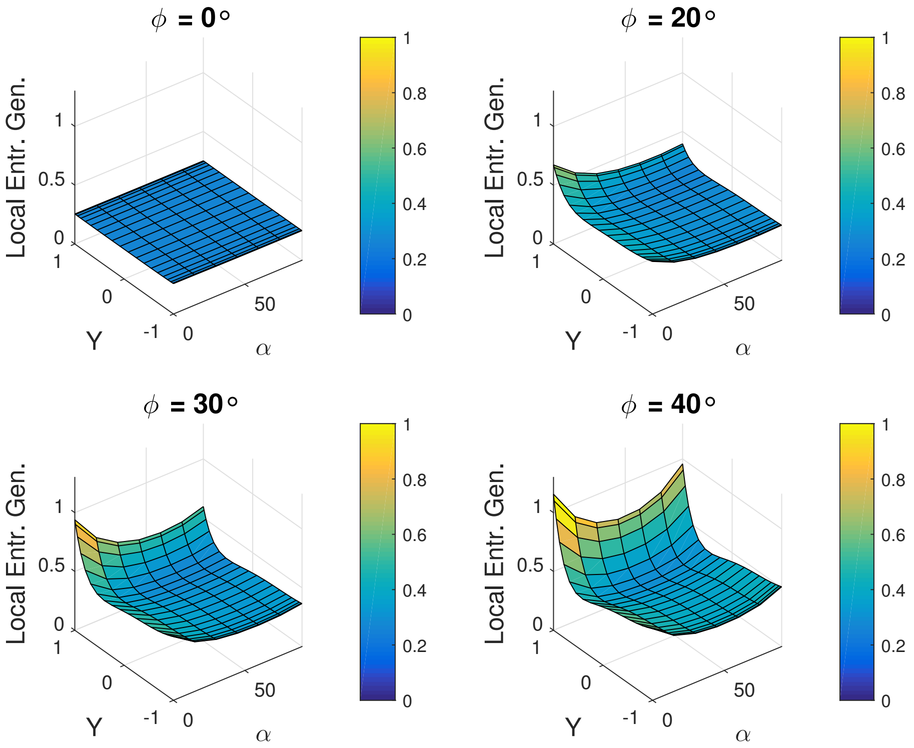

Figure 5.23 : Local entropy generation as a function of magnetic field angle and channel width for four different channel inclinations $\phi = 0, 20, 30, 40$.

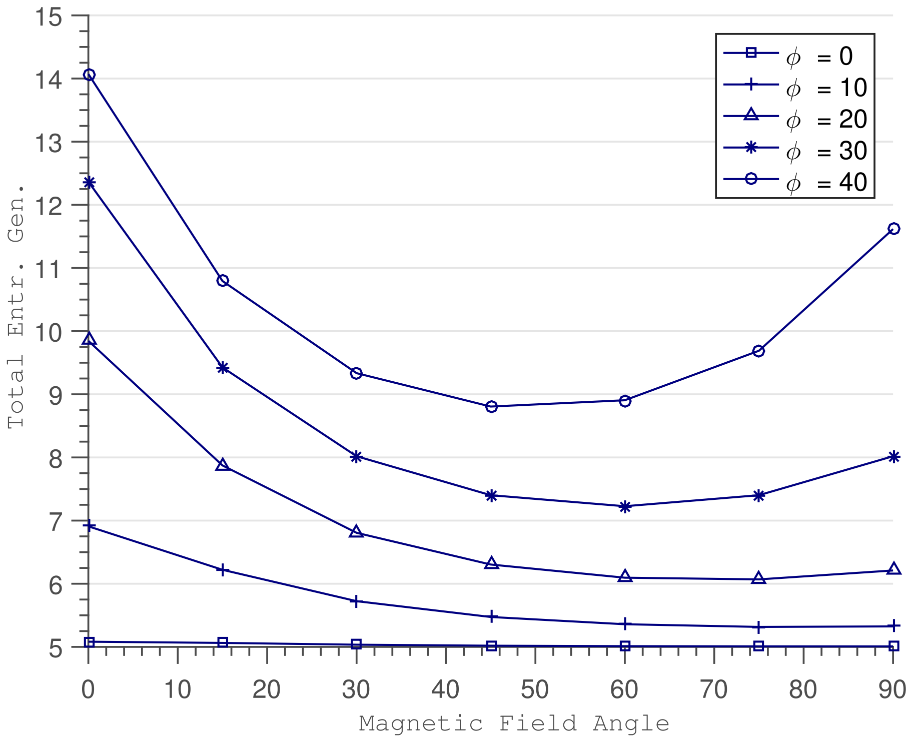

Figure 5.24 : Total entropy generation as a function of magnetic field angle α for a variety of channel inclination ϕ.

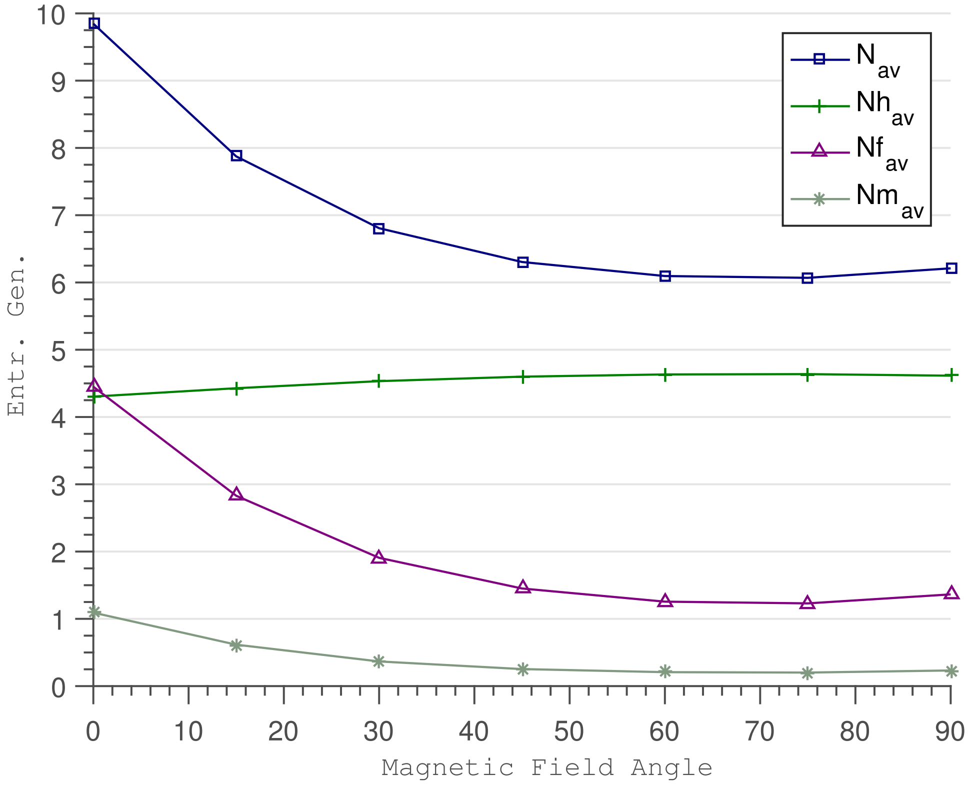

Figure 5.25 : Total entropy generation due to its terms; heat transfer term, viscous dissipation term, magnetic field term as a function of magnetic field angle α for channel inclination $\phi = 20$.

The influences of the different parameters on entropy generation within the channel are presented in Figures 5.22–5.25. It is seen from Figs that the entropy generation number NS depends on magnetic field angle, channel inclination.

To gain further insight about the behavior of local entropy generation as a function of magnetic field angle and channel inclination, four 3D graphes are presented as Figure 5.23. In this simulation $Ha = 2$ and $\psi = 0.06$. These four surfaces in Figure 5.23 present the change of local entropy generation with magnetic field angle α for four different channel inclinations $\phi = 0, 20, 30, 40$.

It can be seen from the upper left surface corresponding to $\phi = 0$, the change in magnetic field angle α does not help much to decrease the local entropy generation. But for channel inclination $\phi = 20$, local entropy generation is decreased for higher values of α. For higher values of ϕ, $\phi = 30, 40$, an increase in α first results in a decrease in local entropy generation. But after some specific value, the local entropy generation starts to increase for increasing α. This can be seen also from Figure 5.24, where local entropy generation is averaged along the width of channel and no longer dependent on the location, namely total entropy generation. Those result means that magnetic field angle can be used to decrease local entropy generation but depending on the inclination of the channel, this might be the contrary. This means that the user can utilize these behavior to optimize the system depending on the perquisites of the problem. Another result observed is that this behavior is not changed for nanoparticle volume fraction ψ. But for larger ψ, a general decreasing effect is seen in the simulations for all values of α and ϕ.

A further investigation given in Figure 5.25 on the total entropy generation concerns the visualization of entropy generation terms separately. As seen from Equation (4.17), the entropy generations is comprised of entropy generation due to heat transfer irreversibility, entropy generation due to fluid friction irreversibility and entropy generation due to magnetic field. Figure 5.25 shows that the decrease in total entropy generation via an increase in magnetic field angle belongs to viscous dissipation and magnetic field terms of entropy generation. The entropy generation term due to heat transfer irreversibility remains nearly unchanged. To achieve a further decrease in entropy generation, this terms might be handled with a different controllable parameter.

6. CONCLUSION

In this thesis, nanofluid filled channel flows have been studied. Nanofluids offer a variety of promising properties although the underlying principles enabling these advantages not thoroughly understood. Emprical models in the literature are used to model the nanofluid properties such as thermal conductivity. The properties of the flow mainly discussed in terms velocity and temperature profiles due to their effects on thermal management. And then the entropy generation is discussed for designing more efficient systems via minimizing it. So the effects of a variety of parameters on entropy generation is examined.

This work discusses three different problems, where the two are quite similar. The first problem investigates the GDQM methods performance in solving channel problems. For that purpose, the steady viscous incompressible flow of an electrically conducting fluid between two porous plates under magnetic field is studied. A combination of GDQM & NR methods is applied to obtain velocity and temperature fields in the channel having constant injection and suction walls. The velocity, temperature fields, local entropy generation rate and total entropy generation are investigated depending on physical parameters such as Ha number, Suction/Injection parameter, V*, function of viscosity variation, ε.

Then, the performance of GDQM method is examined for equal gridding, CGL gridding techniques and different number of grids. The results compared with existing work using Runge-Kutta (RK) method. GDQM is a strong semi numerical solution tool for discretization of partial derivatives. Combining GDQM and Newton-Raphson Method gives more accurate results than solutions in the open literature via RK Method. It is also noticed that with a few grid points in GDQM solution technique, the results obtained are accurate enough. The inherent success of GDQM to converge to accurate solutions using a few number of grid points is also observed in the simulations and pointed out in tables.

In second part of the work, nanofluids are included in the study. The effect of magnetic field, inclination angle of the channel, and volume fraction on the velocity profile and the temperature distribution are investigated in an inclined channel problem. The governing equations are given including viscous dissipation and bouyancy terms. Utilizing appropriate dimensionless parameters and defining some others, the equations are nondimensionalized. Equations for flow and thermal fields are discretized using GDQM, giving fairly accurate results for even very few grid points. The discritized system of equations are solved simultanously utilizing Runge- Kutta scheme.

It is observed that an increase in magnetic field density suppresses the flow field significantly, and this effects gets stronger as the volume fraction of nanoparticles in nanofluids increases. The velocity and temperature increases as the inclination of the channel rises but this dependency diminishes with larger magnetic field intensity or volume fraction of nanofluids. The velocity decreases with an increase in volume fraction of nanofluid. When magnetic field gets stronger, the volume fraction dependence of the velocity also increases and gets more dependent to temperature distribution. The temperature distribution is not very effected by volume fraction and this dependence gets even less with increased magnetic field magnitude.

Investigation of entropy generation and equipartition constitutes another main argument of this part of the study. Quantification of the destruction of system's available work is a key issue in the design of efficient system. The contribution of each component of entropy generation mechanisms, magnetic field irreversibility, heat transfer irreversibility, fluid friction irreversibility is discussed. Their variation with flow parameters such as Ha number and V* number are investigated. An equipartion phenomenon, equal distribution of entropy generation contributions, is observed between magnetic field irreversibility, heat transfer irreversibility and fluid friction irreversibility for higher values of Ha numbers. The variation of total entropy generation with flow parameters such as Ha number and V* number, is not linear, thus minimizing entropy generation should be handled via an optimization algorithm.

In the last part of the work, the effects of magnetic field orientation angle and channel inclination angle are separately investigated, when the channel is filled with ferrofluid. And for such a channel system, related parameters investigated to produce minimum entropy generation such as magnetic field magnitude, channel inclination angle and nanofluid volume fraction. The viscous dissipations and buoyancy effects are included in the governing equations. Derived governing equations are non-dimensionalised by using physically appropriate parameters.

Equations for flow and thermal fields are discretized using GDQM, a new computationally efficient tool giving fairly accurate results for even very few grid points. The discretized system of equations are solved simultaneously utilizing Runge–Kutta scheme. An increase in magnetic field density suppresses the flow field significantly, and these effects get stronger as the volume fraction of nanoparticles in nanofluids increases. The velocity and temperature increases as the inclination of the channel rises but this tendency diminishes with larger magnetic field intensity or volume fraction of nanofluids. The velocity decreases with an increase in volume fraction of nanoparticles. When magnetic field gets stronger, the volume fraction dependence of the velocity also increases and gets more dependent to temperature distribution. Influence of a change in particle volume fraction on temperature distribution is minor and diminished with increased magnetic field magnitude. The entropy generation decreases with increasing magnetic field angle for smaller values of channel inclination. For higher values of channel inclination, with an increase in magnetic field angle, the entropy generation first decreases and then increases. The minimum entropy generation is observed around when the magnetic field angle is perpendicular to the channel. Thus, to further optimize the system by managing velocity and temperature distributions, an multi-objective optimization method should be utilized to serve the users needs.

7. FUTURE WORK

Extracting the essence of nature's working principles has always attracted curious minds. Biomimetics or biomimicry is becoming a more appreciated source of inspiration for system designs in a variety of topics, thanks to the rapid development of technology enabling to reveal the mysterious mechanisms evolved in centuries by nature. Utilization of these uncovered working principles on science and engineering problems offer elegant solutions for challenges.

The idea of impedance pump is inherited from the developmental biology of the embryonic vertebrate heart. The researchers have been amazed realizing that blood pumping in the embryonic vertebrate heart starts before the the discernable chambers and valves have been developed. Although this phenomenon has been explained as peristaltic motion in early stages, advances in confocal laser scanning microscopy and four-dimensional visualization, utilized to investigate the cell movements of embryonic zebrafish hearth tube, showed contradictory foundings [121]. Forouhar and his colleagues offered that this can be explained with the elastic wave propagation in the heart tube causing a suction induces that pumping behavior.

It is comprised of flexible tubing of different impedances filled with fluid. The advantage is that a flow can be induced without the need for valves but by applying a compression at asymmetric positions from the ends. The frequency of the pinching has a major importance on the flow characteristics and the direction of the flow can even be reversed by changing the frequency and duty cycle.

The study of valveless pumps goes back to 1954 when Gerhart Libeau got suspicious about the high efficiency of blood circulations may not be explained only by the work done by the heart [122–124]. Although he also achieved to show through experiments that the valveless pumping (See Fig. 7.1) effect in elastic tubes exists, an attempt to model the phenomenon delayed more than 10 years until von Bredow [125] derived a set of linear equations matcing Liebau phenomenon. Consideringring an elastic tube connecting two tanks, Rath and Teipel [126] developed a nonlinear mathematical

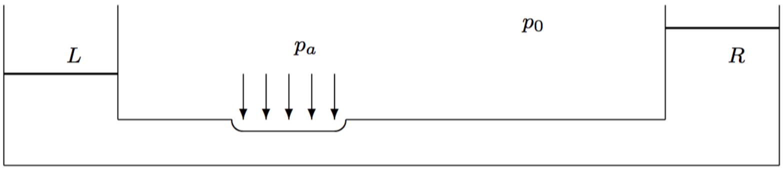

Figure 7.1 : A Liebau pump [6]

model for this one dimensional flow problem and solved it numerically. Takagi and Takahashi [127] were the first to show experimentally and numerically that the valveless pumping is possible for rigid pipes, although previous numerical studies also indicated that result [6].

Addition of nanofluids to this pumping mechanism might lead to interesting results. Using an external magnetic field to induce motion instead of pinching might be considered. A challenge to cope with is the limited number of studies investigating numerical models for this pumping mechanism.

REFERENCES

[1] **Serrano, E.**, **Rus, G. and Garcia-Martinez, J.** (2009). Nanotechnology for sustainable energy, *Renewable and Sustainable Energy Reviews*, *13*(9), 2373–2384.

[2] **Wen, D.**, **Lin, G.**, **Vafaei, S. and Zhang, K.** (2009). Review of nanofluids for heat transfer applications, *Particuology*, *7*(2), 141–150.

[3] **Shen, B.** (2008). *Minimum quantity lubrication grinding using nanofluids*, University of Michigan (Ph.D. Thesis).

[4] **Choi, S.** (2006). Nanofluids for improved efficiency in coolings systems, *Heavy vehicle systems review*, Argonne National Laboratory.

[5] **Bejan, A.** (1982). *Entropy Generation Through Heat and Fluid Flow*, Wiley.

[6] **Borzi, A. and Propst, G.** (2003). Numerical investigation of the Liebau phenomenon, *Zeitschrift fur angewandte Mathematik und Physik ZAMP*, *54*, 1050–1072.

[7] **Baskaya, E.**, **Fidanoglu, M.**, **Komurgoz, G. and Ozkol, I.** (2014). Investigation of MHD Natural Convection Flow Exposed to Constant Magnetic Field via Generalized Differential Quadrature Method, *ASME 2014 12th Biennial Conference on Engineering Systems Design and Analysis*, American Society of Mechanical Engineers.

[8] **Baskaya, E.**, **Komurgoz, G. and Ozkol, I.** (2017). Investigation of Oriented Magnetic Field Effects on Entropy Generation in an Inclined Channel Filled with Ferrofluids, *Entropy*, *19*(7), 377.

[9] **Beck, M.P.** (2008). *Thermal conductivity of metal oxide nanofluids*, Georgia Institute of Technology (Ph.D. Thesis).

[10] **Choi, S. and Eastman, J.** (1995). Enhancing thermal conductivity of fluids with nanoparticles, *ASME-Publications-Fed*, *231*, 99–106.

[11] **Maxwell, J.C.** (1904). *A Treatise on Electricity and Magnetism, vol. II. Clarendon*, Oxford.

[12] **Eastman, J.**, **Choi, U.**, **Li, S.**, **Thompson, L. and Lee, S.** (1996). Enhanced thermal conductivity through the development of nanofluids, *MRS Online Proceedings Library Archive*, *457*.

[13] **Liu, M.S.**, **Lin, M.C.**, **Huang, I.T. and Wang, C.C.** (2006). Enhancement of thermal conductivity with CuO for nanofluids, *Chemical engineering & technology*, *29*(1), 72–77.

[14] **Hwang, Y.**, **Park, H.**, **Lee, J. and Jung, W.** (2006). Thermal conductivity and lubrication characteristics of nanofluids, *Current Applied Physics*, *6*, e67–e71.

[15] **Yu, W.**, **Xie, H.**, **Chen, L. and Li, Y.** (2009). Investigation of thermal conductivity and viscosity of ethylene glycol based ZnO nanofluid, *Thermochimica Acta*, *491*(1), 92–96.

[16] **Mintsa, H.A.**, **Roy, G.**, **Nguyen, C.T. and Doucet, D.** (2009). New temperature dependent thermal conductivity data for water-based nanofluids, *International Journal of Thermal Sciences*, *48*(2), 363–371.

[17] **Verma, P.**, **Chaturvedi, P.**, **Rawat, J.S.**, **Kumar, M.**, **Pal, S.**, **Bal, M.**, **Rawal, D.**, **Vyas, H.P.**, **Ghosal, P. and Bhatnagar, P.** (2007). Elimination of current non-uniformity in carbon nanotube field emitters, *Journal of Materials Science: Materials in Electronics*, *18*(6), 677–680.

[18] **Xu, J.**, **Zhang, J.**, **Du, Y.**, **Zhang, X. and Li, Y.** (1996). Ultrasonic velocity and attenuation in nano-structured Zn materials, *Materials Letters*, *29*(1-3), 131–134.

[19] **Eastman, J.A.**, **Choi, S.**, **Li, S.**, **Yu, W. and Thompson, L.** (2001). Anomalously increased effective thermal conductivities of ethylene glycol-based nanofluids containing copper nanoparticles, *Applied physics letters*, *78*(6), 718–720.

[20] **Abu-Nada, E.** (2011). Rayleigh-Bénard convection in nanofluids: effect of temperature dependent properties, *International Journal of Thermal Sciences*, *50*(9), 1720–1730.

[21] **Abu-Nada, E.**, **Masoud, Z.**, **Oztop, H.F. and Campo, A.** (2010). Effect of nanofluid variable properties on natural convection in enclosures, *International Journal of Thermal Sciences*, *49*(3), 479–491.

[22] **Sebdani, S.M.**, **Mahmoodi, M. and Hashemi, S.M.** (2012). Effect of nanofluid variable properties on mixed convection in a square cavity, *International Journal of Thermal Sciences*, *52*, 112–126.

[23] **Bondareva, N.**, **Sheremet, M.A. and Pop, I.** (2015). Magnetic field effect on the unsteady natural convection in a right-angle trapezoidal cavity filled with a nanofluid: Buongiorno's mathematical model, *International Journal of Numerical Methods for Heat & Fluid Flow*, *25*(8), 1924–1946.

[24] **Sheremet, M.A. and Pop, I.** (2015). Natural convection in a wavy porous cavity with sinusoidal temperature distributions on both side walls filled with a nanofluid: Buongiorno's mathematical model, *Journal of Heat Transfer*, *137*(7), 072601.

[25] **Jang, S.P. and Choi, S.** (2006). Effects of Various Parameters on Nanofluid Thermal Conductivity, *Journal of Heat Transfer*, *129*(5), 617–623.

[26] **Nacev, A.** (2013). *Magnetic Drug Targeting: Developing the Basics*.

[27] **Arruebo, M.**, **Fernandez-Pacheco, R.**, **Ibarra, M.R. and Santamaria, J.** (2007). Magnetic nanoparticles for drug delivery, *Nano Today*, *2*(3).

[28] **Panhurst, Q.A.**, **Connolly, J.**, **Jones, S.K. and Obson, J.** (2003). Applications of magnetic nanoparticles in biomedicine, *J. Phys. Appl. Phys.*, *36*(13), R167.

[29] **Panhurst, Q.A.**, **Thanh, N.K.**, **Jones, S.K. and Obson, J.** (2009). Progress in applications of magnetic nanoparticles in biomedicine, *J. Phys. Appl. Phys.*, *42*, 22400116.

[30] **Lubbe, A.S.**, **Bergemann, C.**, **Riess, H.**, **Schriever, F.**, **Reichardt, P.**, **Possinger, K.**, **Matthias, M.**, **Dorken, B.**, **Herrmann, F.**, **Gurtler, R.**, **Hohenberger, P.**, **Haas, N.**, **Sohr, R.**, **Sander**, **Lemke, A.J.**, **Ohlendorf, D.**, **Huhnt, W.**, **and Huhn, D.** (1996). Clinical experiences with magnetic drug targeting: A phase I study with 4-epidoxorubicin in 14 patients with advanced solid tumors, *Cancer Res.*, *56*(20), 46864693.

[31] **Obson, J.** (2006). Magnetic micro- and nano-particle-based targeting for drug and gene delivery, *Nanomed.*, *1*(1), 31–37.

[32] **Sheremet, M.A.**, **Oztop, H.F.**, **Pop, I. and Abu-Hamdeh, N.** (2015). Analysis of entropy generation in natural convection of nanofluid inside a square cavity having hot solid block: Tiwari and Das model, *Entropy*, *18*(1), 9.

[33] **Faraday, M.** (1832). Experimental Researches in Electricity, *Trans. R. Soc. Lond.*, *122*, 163–194.

[34] **G. S. Seth, M. S. Ansari, R.N.** (2011). Unsteady Hydromagnetic Couette Flow within a porous Channel, *Tamkang Journal of Science and Engineering*, *14*(1), 7–14.

[35] **S. Molokov, R. Moreau, H.M.** (2007). *Magnetohydrodynamics Historical Evolution and Trends*, Springer.

[36] **McKetta, J.** (1999). *Encyclopedia of Chemical Processing and Design*, *66*.

[37] **Hannes, A.** (1943). On the Existance of Electromagnetic-Hydrodynamic Waves, *Arkiv fr matematik, astronomi och fysik XXIXb*, *7*.

[38] **Hannes, A.** (1988). Cosmology in the plasma universe, *Laser and Particle Beams*, *6*, 389–398.

[39] **Foundation, T.N.**, (1970), nobelPrizes, `http://www.nobelprize.org/nobel_prizes/physics/laureates/1970/`, [Online; accessed 2015].

[40] **Arikoglu, A.**, **Ozkol, I. and Komurgoz, G.** (2008). Effect of slip on entropy generation in a single rotating disk in MHD flow, *Applied Energy*, *85*, 1225–1236.

[41] **Northrup, E.** (1907). Some newly observed manifestations of forces in the interior of an electrical conductor, *Phys Rev*, *24*(6), 474.

[42] **M.S. Tillack, N.M.** (1998). *Magnetohydrodynamics Standart handbook for electrical engineer.*, McGraw Hill.

[43] **A.M. Mahmoudi, I. Pop, M.S.** (2013). MHD natural convection and entropy generation in a trapezoidal enclosure using Cu-water nanofluid, *Computers and Fluids*, *72*, 46–62.

[44] **C. Ahn, M.A.** (1995). Fluid micropumps based on rotary magnetic actuators, *Proc. MEMS*, *408*(95), 12.

[45] **G. Mensing, T. Pearce, D.B.** (2002). An in-plane active magnetic mixer for microfluidic applications, *IEEE EMBSMMMB Conference*, (95), 531–534.

[46] **el Hak, M.G.** (1999). The fluid mechanics of microdevices, *J Fluid Eng ASME the Freeman Scholar Lecture*, *121*(1), 5–33.

[47] **Mahmud, S.**, **Tasnim, S. and Mamun, M.A.H.** (2003). Thermodynamic analysis of mixed convection in a channel with transverse hydromagnetic effect, *Inf J Therm Sci*, *42*(731), 40.

[48] **Takezawa, S.**, **Tamama, H.**, **Sugawawa, K.**, **Sakai, H.**, **Matsuyama, C.**, **Morita, H.**, **Suzuki, H. and Ueyama, Y.** (1995). Operation of the Thruster for Superconducting Electromagnetohydrodynamic Propulsion Ship YAMATO 1, *Bulletin of the M.E.S.J.*, *23*(1), 46–55.

[49] **Roy, S.**, (2013), Wingless Hovering of Micro Air Vehicle (WHOMAV). United States Patent No. 8, 382,029, [issued on February 26, 2013].

[50] **G. W. Sutton, A.S.** (1965). *Engineering Magnetohydrodynamics*, McGrawHill.

[51] **Mahmoudi, A.H.**, **Pop, I.**, **Shahi, M. and Talebi, F.** (2013). MHD natural convection and entropy generation in a trapezoidal enclosure using Cu-water nanofluid, *Computers and Fluids*, *72*, 46–62.

[52] **Kandaswamy, P.**, **Sundari, S. and Nithyadevi, N.** (2008). Magnetoconvection in an enclosure with partially active vertical walls, *Int J Heat Mass Transfer*, *51*(1946), 54.

[53] **M. Pirmohammadi, M.G.** (2009). Effect of magnetic field on convection heat transfer inside a tilted square enclosure, *Int Commun Heat Mass Transfer*, *36*(776), 80.

[54] **Saleh, H.**, **Roslan, R. and Hashim, I.** (2011). Natural convection in a porous trapezoidal enclosure with an inclined magnetic field, *Computers and Fluids*, *47*(155), 64.

[55] **Grosan, T.**, **Revnic, C.**, **Pop, I. and Ingham, D.B.** (2009). Magnetic field and internal heat generation effects on the free convection in a rectangular cavity filled with a porous medium, *Int J Heat Mass Transfer*, *52*(1525), 33.

[56] **Arikoglu, A.**, **Ozkol, I. and Komurgoz, G.** (2010). Effect of slip on entropy generation in a single rotating disk in MHD flow, *Applied Energy*, *85*, 1225–1236.

[57] **M.S. Tillack, N.M.** (1998). *Magnetohydrodynamics : Standard handbook for electrical engineer*, McGraw Hill.

[58] **Alvarez, A.**, **Suarez, P.**, **Caceres, D.**, **Cordero, E.**, **Ceballos, J. and Perez, B.** (2005). Disk shaped superconducting rotor under a rotating magnetic field: speed dependence, *IEEE Trans. Appl. Supercond.*, *15*(2), 2174 –2177.

[59] **Salas, H.**, **Cuevas, S. and Haro, M.** (1999). Entropy generation analysis of magnetohydrodynamic induction devices, *J. Phys. D: Appl. Phys.*, *32*, 2605–2608.

[60] **Hagen, G.** (1839). er die Bewegung des Wassers in engen zylindrischen Rohren, *Pogg. Ann.*, *46*.

[61] **Poiseuille, J.L.M.** (1840). Recherches expmentales sur le mouvement des liquides dans les tubes de tretits diames, *Comptes Rendus Acad. Sc*, *11*, 961.

[62] **Brown, G.O.** (2004). The History of the Darcy Weisbach Equation for Pipe Flow Resistance, *Environment and Water Resources History*.

[63] **B. Singh, A.B.** (2014). Wavelet optimized finite difference mesh for MHD flow in a circular duct, *Computers Mathematics with Applications*, *67*, 1582–1594.

[64] **Moreau, R.** (1990). *Magnetohydrodynamics*, Kluwer Academic Publishers.

[65] **Eegunjobi, A.S. and Makinde, O.D.** (2013). Entropy Generation Analysis in a Variable Viscosity MHD Channel Flow with Permeable Walls and Convective Heating, *Mathematical Problems in Engineering*.

[66] **S. Das, R.N.J.** (2013). Effects of Hall Currents on Entropy Generation in a Porous Channel with Suction/Injection, *International Journal of Energy and Technology*, *5*, 25.

[67] **Hartmann, J. and Lazarus, F.** (1937). *Hg - DYNAMICS II: Experimental Investigations on the flow of mercury in a homogenous magnetic field*, Levin & Munksgaard.

[68] **Lehnert, B.** (1952). On the behaviour of an electrically conductive liquid in a magnetic field, *Arkiv for Fysik*, *5*(5), 69–90.

[69] **Katagiri, M.** (1962). Flow formation in Couette motion in magnetohydrodynamics, *J. Phys. Soc. Jpn.*, *17*, 393.

[70] **M. Balaram, T.G.** (1973). Unsteady Couette Flow in hydromagnetics, *J.Appl. Mech. Trans ASME*, *40*, 620.

[71] **O. D. Makinde, P.Y.M.** (2005). Heat transfer to MHD oscillatory flow in a channel filled with porous medium, *Romanian J. Physics*, *50*(9), 931–938.

[72] **Seth, G.S.**, **Ansari, M. and Nandkeolyar, R.** (2011). Unsteady hydromagnetic couette flow with a porous chanller, *Tamkang J. Science and Engineering*, *14*(1), 7–14.

[73] **Carnot, N.** (1824). Reflexions sur la puissance motrice du feu, *Bachelier, Paris (critical edition by Fox R, Vrin, Paris, 1978).*

[74] **Clausius, R.** (1854). On a modified form of the second principal theorem of mechanical theory, *Annals of Physics*, *169*(12), 481–506.

[75] **Clausius, R.** (1865). On various forms of the principal equations of mechanical theory, *Annals of Physics*, *201*(7), 353–400.

[76] **Bejan, A.** (1980). Second law analysis in heat transfer, *Energy Int.*, *5*, 721–732.

[77] **Bejan, A.** (1982). Second-law analysis in heat transfer and thermal design, *Adv. Heat Transfer*, *15*, 1–58.

[78] **Bejan, A.** (1996). *Entropy Generation Minimization*, CRC Press: New York.

[79] **Bejan, A.** (1979). A study of entropy generation in fundamental convective heat transfer, *J. Heat Transfer*, *101*, 718–725.

[80] **Bejan, A.**, **Tsatsaronis, G. and Moran, M.** (1996). *Thermal Design and Optimization*, Wiley: New York.

[81] **Roy, M.**, **Basak, T.**, **Roy, S. and et all** (2015). Analysis of Entropy Generation for Mixed Convection in a Square Cavity for Various Thermal Boundary Conditions, *Numerical Heat Transfer Part A: Applications*, *68*(1), 44–74.

[82] **S Bhardwaj, A.D.** (2015). Effect of Undulations on the Natural Convection Heat Transfer and Entropy Generations Inside a Porous Right-Angled Triangular Enclosure, *Numerical Heat Transfer Part A: Applications*, *67*(9), 972–991.

[83] **Yang, Y.T.**, **Wang, Y.H.**, **Yi-Hsien, H. and Huang, B.Y.** (2015). Numerical Optimization for Nanofluid Flow in Microchannels using Entropy Generation Minimization, *Numerical Heat Transfer Part A: Applications*, *67*(5), 571–588.

[84] **Komurgoz, G.**, **Arikoglu, A.**, **Turker, E. and et al** (2010). Second-Law Analysis for an Inclined Channel Containing Porous-Clear Fluid Layers by Using the Differential Transform Method, *Numerical Heat Transfer Part A: Applications*, *57*(8), 603–623.

[85] **H. Salas, e.a.** (1999). Entropy Generation Analysis of Magnetohydrodynamic Induction Devices, *J. Phys D: Appl Phys.*, *32*, 2605–2608.

[86] **S. Mahmud, e.a.** (2003). Thermodynamic Analysis of Mixed Convection in a Channel with Transverse Hydromagnetic Effect, *Int. J. Thermal Sciences*, *42*(8), 731–740.

[87] **D.S. Chauhan, V.K.** (2011). Heat transfer and entropy generation during compressible fluid flow in a channel partially filled with porous medium, *Int. J. Energ. Tech.*, *3*, 1–10.

[88] **Tasnim, S.**, **Mahmud, S. and Mamum, M.** (2002). Entropy generation in a porous channel with hydromagetic effect, *Int. J. Exergy*, *3*, 300–308.

[89] **A.S. Eegunjobi, O.M.** (2012). Combined effect of buoyancy force and Navier slip on entropy generation in a vertical porous channel, *Entropy*, *14*, 10281044.

[90] **A. E. Jery, e.a.** (2010). Effect of an External Oriented Magnetic Field on Entropy Generation in Natural Convection, *Entropy*, *12*(6), 1391–1417.

[91] **Dwivedi, R.**, **Singh, S.P.**, **and Singh, B.B.** (2010). Analysis of incompressible viscous laminar flow through a channel filled with porous media, *Int. J. Stability and Fluid Mechanics*, *1*(1), 127–134.

[92] **Makinde, O.D. and Chinyoka, T.** (2013). Numerical investigation of buoyancy effects on hydromagnetic unsteady flow through a porous channel with suction/injection, *Mechanical Science and Technology*, *27*(5), 1557–1568.

[93] **Mahmoudi, A.H.**, **Shahi, M. and Talebi, F.** (2012). Entropy generation due to natural convection in a partially open cavity with a thin heat source subjected to a nanofluid, *Numerical Heat Transfer, Part A: Applications*, *61*(4), 283–305.

[94] **Shahi, M.**, **Mahmoudi, A.H. and Raouf, A.H.** (2011). Entropy generation due to natural convection cooling of a nanofluid, *International Communications in Heat and Mass Transfer*, *38*(7), 972–983.

[95] **Singh, P.K.**, **Anoop, K.**, **Sundararajan, T. and Das, S.K.** (2010). Entropy generation due to flow and heat transfer in nanofluids, *International Journal of Heat and Mass Transfer*, *53*(21), 4757–4767.

[96] **Moghaddami, M.**, **Mohammadzade, A. and Esfehani, S.A.V.** (2011). Second law analysis of nanofluid flow, *Energy Conversion and Management*, *52*(2), 1397–1405.

[97] **Esmaeilpour, M. and Abdollahzadeh, M.** (2012). Free convection and entropy generation of nanofluid inside an enclosure with different patterns of vertical wavy walls, *International Journal of Thermal Sciences*, *52*, 127–136.

[98] **Tondeur, D. and Kvaalen, E.** (1987). Equipartition of Entropy Production. An Optimality Criterion for Transfer and Separation Processes, *Ind. Eng. Chem. Res.*, *26*, 50–56.

[99] **Thiel, G.P.**, **McGovern, R.K.**, **Zubair, S.M. and V, J.H.L.** (2014). Thermodynamic equipartition for increased second law efficiency, *Applied Energy*, *118*, 292–299.

[100] **Wang, Z. and Li, Y.** (2016). A combined method for surface selection and layer pattern optimization of a multistream plate-fin heat exchanger, *Applied Energy*, *165*, 815–827.

[101] **Cramer, K.R. and Pai, S.I.** (1973). *Magnetofluiddynamics for Engineers and Applied Physicists*, McGraw Hill.

[102] **Das, S. and Jana, R.N.** (2013). Entropy Generation in MHD Porous Channel Flow Under Constant Pressure Gradient, *Applied Mathematics and Physics*, *1*(2), 78–89.

[103] **Agarwal, J.P.** (1962). On Generalized Incompressible Coquette Flow in Hydromechanics, *Applied Scientific Research B*, *9*(4), 255–266.

[104] **Ziapour, B.M. and Dehnavi, R.** (2015). Heat transfer in a large triangular-roof enclosure based on the second law analysis, *Heat and Mass Transfer*, *51*, 931–940.

[105] **Aghanajafi, C.**, **Bakhtiarpoor, M.A.**, **Taghipour, M. and Mohamadi, F.** (2015). Entropy generation analysis for microscale forced convection with radiation in thermal entrance region, *Heat and Mass Transfer*, *51*, 307–312.

[106] **Woods, L.C.** (1975). *Thermodynamics of Fluid Systems*, Oxford University Press.

[107] **Cramer, K. and Pai, S.** (1973). *Magnetofluiddynamics for Engineers and Applied Plysicists*, McGraw Hill Book Company, New York.

[108] **Eegunjobi, A.S. and Makinde, O.D.** (2013). Entropy Generation Analysis in a Variable Viscosity MHD Channel Flow with Permeable Walls and Convective Heating, *Mathematical Problems in Engineering*, *2013*.

[109] **Brinkmann, H.C.** (1952). The Viscosity of Concentrated Suspensions and Solutions, *J. Chem. Phys*, *20*, 571–581.

[110] **Hayat, T.**, **Bibi, S.**, **Rafiq, M.**, **Alsaedi, A. and Abbasi, F.** (2016). Effect of an inclined magnetic field on peristaltic flow of Williamson fluid in an inclined channel with convective conditions, *Journal of Magnetism and Magnetic Materials*, *401*, 733–745.

[111] **El Jery, A.**, **Hidouri, N.**, **Magherbi, M. and Brahim, A.B.** (2010). Effect of an external oriented magnetic field on entropy generation in natural convection, *Entropy*, *12*(6), 1391–1417.

[112] **Mehrez, Z.**, **El Cafsi, A.**, **Belghith, A. and Le Quéré, P.** (2015). MHD effects on heat transfer and entropy generation of nanofluid flow in an open cavity, *Journal of magnetism and Magnetic Materials*, *374*, 214–224.

[113] **Cimpean, D.S. and Pop, I.** (2012). Fully developed mixed convection flow of a nanofluid through an inclined channel filled with a porous medium, *International journal of heat and mass transfer*, *55*(4), 907–914.

[114] **You, X.C.**, **Xu, H. and Pop, I.** (2014). Analysis of Fully Developed Opposing Mixed Convection Flow in an Inclined Channel Filled by a Nanofluid, *Journal of Heat Transfer*, *136*(12), 124502.

[115] **Bellman, R.E. and Casti, J.** (1971). Differential Quadrature and Long Term Integration, *J. Math. Anal. Appl.*, *34*, 235–238.

[116] **Shu, C. and Richards, B.E.** (1990). High Resolution of Natural Convection in a Square Cavity by Generalized Differential Quadrature, *Proceedings of 3rd Conference on Advances in Numerical Methods in Engineering: Theory and Applications*, volume 2, Swansea, UK, pp.978–985.

[117] **Shu, C.** (2000). *Differential Quadrature and Its Applications in Engineering*, Springer.

[118] **Bedeaux, D.**, **Standaert, F.**, **Hemmes, K. and Kjelstrup, S.** (1999). Optimization of processes by equipartition, *J. Non-Equilibrium Thermodynamics*, *24*, 242–259.

[119] **Eegunjobi, A. and Makinde, O.** (2013). Entropy generation analysis in a variable viscosity MHD channel flow with permeable walls and convective heating, *Mathematical Problems in Engineering*, *2013*.

[120] **Başkaya, E.**, **Kömürgöz, G. and Özkol, I.** (2014). Analysis of Variable Viscosity Channel Flow under Constant Magnetic Field via Generalized Differential Quadrature Method, *Advanced Materials Research*, volume1016, Trans Tech Publ, pp.564–568.

[121] **Forouhar, A.S.**, **Liebling, M.**, **Hickerson, A.**, **Nasiraei-Moghaddam, A.**, **Tsai, H.J.**, **Hove, J.R.**, **Fraser, S.E.**, **Dickinson, M.E. and Gharib, M.** (2006). The embryonic vertebrate heart tube is a dynamic suction pump, *Science*, *312*(5774), 751–753.

[122] **Liebau, G.** (1954). Uber ein ventilloses Pumpprinzip, *Naturwissenschaften*, *41*(14), 327.

[123] **Liebau, G.** (1954). Arterielle Pulsation und venose Repulsation, *Z Gesamte Exp Med*, *123*(1), 71–90.

[124] **Liebau, G.** (1955). Herzpulsation und Blutbewegung, *Z Gesamte Exp Med*, *125*(5), 482–498.

[125] **von Bredow, H.J.** (1968). Untersuchung eines ventillosen Pumpprinzips, *Fortschr. Ber. VDI-Zeitschr.*, *7*(9).

[126] **Rath, H.J. and Teipel, I.** (1978). Der Fordereffekt in ventillosen, elastischen Leitungen, *Zeitschrift fur angewandte Mathematik und Physik ZAMP*, *29*, 123–133.

[127] **Takagi, S. and Takahashi, K.** (1985). Study of a piston pump without valves, *Bulletin of JSME*, *28*, 831–836.

APPENDICES

APPENDIX A.1 : Codes

APPENDIX A

1.1 Codes for suction-injection problem

```
% Written by Elgiz Baskaya
% elgizbaskaya@gmail.com 13.01.2014

% This code solves the one dimensional channel problem
% in the presence of a constant magnetic field and under
% constant pressure gradient. The Nonlinear coupled
% differential equations are discretized using GDQM method.
% Then the nonlinear algeabraic equation set is solved
% utilizing NewtonRaphson Method

% SAMPLE INPUT THROUGH SIO
% Enter the grid numbers for each axis and time within brackets
% [xGridNum yGridNum tGridNum]
% If the corresponding axis does not exist write 0

% To have the same results in table and compare with the work
% of Makinde.
% Input = [0 11 0]

% Write down the limits of the axes and time
% [x0 xN y0 yN t0 tN] for the starting and ending points within
% brackets (define problem domain)
% If the corresponding axis does not exist write 0 0

% To have the same results in table and compare with the work
% of Makinde.
% Input = [0 0 0 1 0 0]

%% PROBLEM DEFINITION
clc;
clear all;
close all;
format long;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Constants given in the problem

% Pressure Gradient
G = 1;

epsil = 0;
Bi0 = 0.1;
Bi1 = 0.1;
```

```
% to include Joule dissipation lambda = 1; to exclude
% Joule dissipation lambda = 0;
lamba = 1;

Ha = 1;
Ec = 0.1;
Pr = 0.7;
Br = Ec * Pr;
Re = 1;
% Br / Gama = 0.1
Gama = 0.71;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% GRIDDING

% Option 1. Equal Gridding

% Call function equalGridding to calculate gridded axes
% values. This problem is one directional and the state
% variables only change in y direction. Thus a valid input
% to the function when called would be like [0 11 0]
% meaning that only y axis will be gridded to 5 elements
% and [0 0 -1 1 0 0] meaning the lower and upper values
% of y will be -1 and 1 respectively
griddedAxes = equalGridding();
yValues = griddedAxes.y';

% Option 2. Gauss Lobato Gridding
% Chyebisev Gauss Lobato gridding
%
% N = 7;
% yValues = CGS_grid(0,1,N);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% DIFFERENTIAL QUADRATURE METHOD
% number of unknowns n
% Number of  different functions sets
fn = 2; % Since we only try to find velocity and temperature
m = length(yValues);
n = fn * m;

% Vector of unknown variables
% x_1 ... x_m representing U_21 ... U_2M
% x_m+1 ... x_2m representing theta_21 ... theta_2M

% Unknown variables to be calculated written symbolically
% here
```

```
xSym = sym('x',[n 1]);

% soyle yapilirsa class(x1) = sym oluyor; yukardaki gibi
% olmuyor.
% syms U T;
% U=sym(zeros(N,1));
% Th=sym(zeros(N,1));
%
% for n=1:1:N
%     eval(strcat('syms',' u',num2str(n)));
%     U(n)=eval(strcat('u',num2str(n)));
%     eval(strcat('syms',' th',num2str(n)));
%     Th(n)=eval(strcat('th',num2str(n)));
% end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Initialize

% Weight coefficients for first and second order derivatives
weightCoef2nd = zeros(m,m);
weightCoef1st = zeros(m,m);
F = sym(zeros(2 * m, 1));

% Function in symbolic variables to take the Jacobian
for j = 1 : m
    for k = 1 : m
       weightCoef2nd(j,k) = ...
       calculateSecondOrderCoef(yValues,j,k);
       weightCoef1st(j,k) = ...
       calculateFirstOrderCoef(yValues,j,k);
    end

    % Momentum Equations in discreatized form
    F(j) = weightCoef2nd(j, :) * xSym(1 : m) - ...
    epsil * (weightCoef1st(j, :) * ...
    xSym(m + 1 : 2 * m)) * ...
    (weightCoef1st(j, :) * xSym(1 : m)) - ...
    exp(epsil * xSym(m + j)) * (Re * weightCoef1st(j, :) *...
    xSym(1 : m) +  Ha * xSym(j) - G);

    % Boundary Conditions for temperature
    if j == 1
        F(m + j) = weightCoef1st(1, :) * ...
        xSym(m + 1 : 2 * m) - Bi0 * (xSym(m + j) - 1);
    elseif j == m
        F(m + j) = weightCoef1st(m, :) * ...
        xSym(m + 1 : 2 * m) + Bi1 * xSym(m + j);
    else
   % Energy Equations in discreatized form
    F(m + j) = weightCoef2nd(j, :) * xSym(m + 1 : 2 * m) - ...
    Re * Pr * weightCoef1st(j, :) * xSym(m + 1 : 2 * m) + ...
    Br * exp(- epsil * xSym(m + j)) * ...
    (weightCoef1st(j, :) * xSym(1 : m))^2 + ...
    Br * Ha * (xSym(j))^2;
    end
end

% Boundary conditions for velocity u(0) = u(L) = 0
F = subs(F, [xSym(1) xSym(m)], [0 0]);

% Functions corresponding to boundary conditions
F(1,:) = [];
F(m - 1 ,:) = [];

xSym(1) = [];
xSym(m - 1) = [];

% Take the Jacobian of the function
jacob = jacobian(F, xSym(:));

%% NEWTON RAPHSON METHOD
% NEwton Raphson method is utilized in this section to
% solve the nonlinear algeabraic equation set
% developed above

tol = 1e-3;

% Initial guess for the state vector
%Guess = [ 0.9 1.2 1.3 1.1 1.1 1.3 1.2 0.9 0.9 ...
% 0.8 0.7 0.6 0.5 0.4 0.3 0.2]’;
%x = Guess;
%Guess = sin(pi * yValues(2:m-1)/2 +pi/2)’;
x = [ones(m - 2,1); ones(m,1);];

%set the error 2*tol to make sure the loop runs at least once
error = 2*tol;

% preallocating
velocityCalc = zeros(m - 2, 1);
temperatureCalc = zeros(m, 1);
velocity = zeros(m - 2 , 1);
temperature = zeros(m, 1);

while error > tol
% %calculate the function values at the current iteration
% F = feval(MyFunc,x);
% %calculate the jacobian matrix
% J = feval(Jacobian,x);

F_eval = double(subs(F,xSym,x));
J_eval = double(subs(jacob, xSym,x));

%calculate the update (solve the linear system)
dx = - J_eval \ F_eval;
```

```
%update the x value
x = x + dx;

% divide the unknown vector to unknown states
velocityCalc = x(1 : m - 2);
temperatureCalc = x(m - 1 : 2 * m - 2);
% include boundaries conditions
velocity = [0; velocityCalc ;0];
temperature = temperatureCalc;

%calculate the error
F_eval = double(subs(F,xSym,x));
error = max(abs(F_eval));
end %while loop

display(velocity)
display(temperature)

%% ANALYTICAL SOLUTION
% Ref:
alfa = (Re + sqrt(Re^2 + 4* Ha)) / 2;
beta = (Re - sqrt(Re^2 + 4 * Ha)) / 2;
velocityAnal = G / Ha * (((exp(alfa * yValues) * ...
(exp(beta) - 1) - exp(beta * yValues) * ...
(exp(alfa) - 1)) / (exp(alfa) - exp(beta))) + 1);

display(velocityAnal)

%% Error calculation
% http://netlib.org/scalapack/slug/node135.html
% http://www.netlib.org/lapack/lug/node75.html

% Relative ERROR :
% Two Norm :
errorTwoNorm = norm(velocity - velocityAnal, 2) / ...
norm(velocityAnal,2)

% Infinity Norm :
errorInfNorm = norm(velocity - velocityAnal, Inf) / ...
norm(velocityAnal,Inf)
% Absolute ERROR :
% Two Norm :

% Infinity Norm:


% Entropy Generation
S = (weightCoef1st * temperature).^2 + ...
Br / Gama * ((exp(-epsil * temperature) .* ...
(weightCoef1st * velocity).^2) + Ha * velocity.^2);
```

```
function [coef_ii_firstOrder] = coeFirstOrder_ii(griddedAxis,i)
 % case i == j  or coef_ii

 coef_ii_firstOrder = 0;
    for j = 1: 1: length(griddedAxis)
        if i ~= j
            coef_ii_firstOrder = coef_ii_firstOrder - ...
            coeFirstOrder_ij(griddedAxis,i,j);
        end
    end

end
```

```
function [coef_ij_firstOrder]=coeFirstOrder_ij(griddedAxis,i,j)

% M1 first order derivative
% M2 second order derivative

% M1_atAPoint = MfirstOrder(griddedAxis.x,gridIndex);

    coef_ij_firstOrder = MfirstOrder(griddedAxis, i) / ...
    (griddedAxis(i) - griddedAxis(j)) / ...
    MfirstOrder(griddedAxis,j );

end
```

```
function [coef_ii_secondOrder]=coefSecondOrder_ii(griddedAxis,i)
 % case i == j  or coef_ii

 coef_ii_secondOrder = 0;
    for j = 1: 1: length(griddedAxis)
        if i ~= j
            coef_ii_secondOrder = coef_ii_secondOrder - ...
            coefSecondOrder_ij(griddedAxis,i,j);
        end
    end

end
```

```
function[coef_ij_secondOrder]=coefSecondOrder_ij(griddedAxis,i,j)

    coef_ij_secondOrder = 2 * ...
    coeFirstOrder_ij(griddedAxis, i , j) * ...
    (coeFirstOrder_ii(griddedAxis,i) - ...
    (1 / (griddedAxis(i) - griddedAxis(j))));
```

```
end
```

```
function [griddedAxis] = equalGridding()

%% Info

% OUTPUT of the function
% griddedAxis   : a struct where
% griddedAxis.x : gridded x-axis values
% griddedAxis.y : gridded y-axis values
% griddedAxis.t : gridded time values

%% Gridding
    gridNumber = input(’Enter the grid numbers for each ...
    axis and time within brackets ...
    [xGridNum yGridNum tGridNum]\n ...
    If the corresponding axis does not exist write 0 \n’);
    boundaries = input(’Write down the limits of the axes...
    and time [x0 xN y0 yN t0 tN] for the starting and ...
    ending points within brackets (define problem ...
    domain)\nIf the corresponding axis does not exist ...
    write 0 0 \n’);
    griddedAxis.x = zeros(1,gridNumber(1));
    griddedAxis.y = zeros(1,gridNumber(2));
    griddedAxis.t = zeros(1,gridNumber(3));

    for i = 1 : 1 : gridNumber(1)
        griddedAxis.x(i) = boundaries(1) + ...
        (boundaries(2) - boundaries(1)) / ...
        (gridNumber(1) - 1 ) * (i - 1);
    end

    for j = 1 : 1 :  gridNumber(2)
        griddedAxis.y(j) = boundaries(3) + ...
        (boundaries(4) - boundaries(3)) / ...
        (gridNumber(2) - 1 ) * (j - 1);
    end

    for k = 1 : 1 : gridNumber(3)
        griddedAxis.t(k) = boundaries(5) + ...
        (boundaries(6) - boundaries(5)) / ...
        (gridNumber(3) - 1 ) * (k - 1);
    end

end
```

```
function [ x ] = CGS_grid( x1, xn, N )
%CGS_GR?D Chebyshev-Gauss-Lobatto Grid
% x1: Start point of grid at x direction
% xn: End point of grid at x direction
% y1: Start point of grid at y direction
% ym: End point of grid at y direction
% N: number of grids at x direction
% M: number of grids at y direction

x=zeros(N,1);
for k=1:1:N
    x(k)=0.5*(1-cos((k-1)/(N-1)*pi))*(xn-x1)+x1;
end


end
```

```
function[coeFirstOrder]=calculateFirstOrderCoef(griddedAxis...
, i, j)

    if i == j
        coeFirstOrder = coeFirstOrder_ii(griddedAxis, i);
    else
        coeFirstOrder = coeFirstOrder_ij(griddedAxis, i,j);
    end
end
```

```
function[coefSecondOrder]=...
calculateSecondOrderCoef(griddedAxis, i, j)

    if i == j
        coefSecondOrder = coefSecondOrder_ii(griddedAxis, i);
    else
        coefSecondOrder = coefSecondOrder_ij(griddedAxis, i,j);
    end
end
```

```
function [M1_atApoint] = MfirstOrder(griddedAxis,k)
M1_atApoint = 1;
    for m = 1: 1: length(griddedAxis)
        if k ~= m
        M1_atApoint = ( griddedAxis(k) - griddedAxis(m) ) *...
        M1_atApoint;
        end
    end
end
```

```
% Investigation of  the problem on dimensionless parameters
% (Gr/Re, Ha, Br) for the given number of grids (N)

N = 20;
yValues = CGS_grid(0,1,N);

% griddedAxes = equalGridding();
% yValues = griddedAxes.y';

% Dependence on Hartmann Number

% Pressure Gradient
G = 1;

epsil = 0.1;
Bi0 = 0.1;
Bi1 = 0.1;

% Ha = 1;
Ec = 0.1;
Pr = 0.71;
Br = Ec * Pr;
Re = 0.1;
% Br / Gama = 0.1
Gama = 0.71;

velocityAnal = zeros(length(yValues),4);
% vel(:,k) = velocity;
% temp(:,k) = temperature;

k = 1;

for Ha = 1: 1 : 4

    [velocity,temperature,S] = ...
    oneDimPorousChannelwithMagFPartialBound(G, ...
    epsil, Bi0, Bi1, Ha, Pr, Re, Br, Gama);

    % Analytical Solution
    alfa = (Re + sqrt(Re^2 + 4* Ha)) / 2;
    beta = (Re - sqrt(Re^2 + 4 * Ha)) / 2;
    velocityAnal.data(:,k) = G / Ha * ...
    (((exp(alfa * yValues) * (exp(beta) - 1) - ...
    exp(beta * yValues) * (exp(alfa) - 1)) / ...
    (exp(alfa) - exp(beta))) + 1);

    vel.data(:,k) = velocity;
    temp.data(:,k) = temperature;

    k = k + 1;

end
```

```
figVel = figure;
hold on;

vel1 = plot(yValues,vel.data(:,1));
set(vel1                            , ...
  'Marker'          , 's'         , ...
  'Color'           , [0 0 .5]    ,...
  'LineWidth'       , 1.         );
vel2 = plot(yValues,vel.data(:,2));
set(vel2                            , ...
  'Marker'          , '+'         , ...
  'Color'           , [0 0 .5]  ,...
  'LineWidth'       , 1.         );
vel3 = plot(yValues,vel.data(:,3));
set(vel3                            , ...
  'Marker'          , '^'         , ...
  'Color'           , [0 0 .5]  ,...
  'LineWidth'       , 1.         );

vel4 = plot(yValues,vel.data(:,4));
set(vel4                            , ...
  'Marker'          , '*'         , ...
  'Color'           , [0 0 .5]  ,...
  'LineWidth'       , 1.         );


hTitle  = title ('Velocity Profile for different ...
values of Ha ');
hYLabel = ylabel('Velocity'                    );
hXLabel = xlabel('Length - Y'                     );

hLegend = legend( ...
  [vel1, vel2, vel3, vel4], ...
  'Ha = 1' , ...
  'Ha = 2'     , ...
  'Ha = 3'      , ...
  'Ha = 4' , 2);

set( gca                       , ...
    'FontName'    , 'Helvetica' );
set([hTitle, hXLabel, hYLabel], ...
    'FontName'    , 'AvantGarde');
set([hLegend, gca]             , ...
    'FontSize'    , 12          );
set([hXLabel, hYLabel]  , ...
    'FontSize'    , 12         );
set( hTitle                    , ...
    'FontSize'    , 12         , ...
    'FontWeight'  , 'bold'     );
```

```
set(gca, ...
  'Box'         , 'off'     , ...
  'TickDir'     , 'out'     , ...
  'TickLength'  , [.02 .02] , ...
  'XMinorTick'  , 'on'      , ...
  'YMinorTick'  , 'on'      , ...
  'YGrid'       , 'on'      , ...
  'XColor'      , [.3 .3 .3], ...
  'YColor'      , [.3 .3 .3], ...
  'LineWidth'   , 1         );

% hText    = text(10, 800, ...
%   sprintf('\\it{C = %0.1g \\pm %0.1g (CI)}', ...
%   c, cint(2)-c));
set(gcf, 'PaperPositionMode', 'auto');
print -depsc2 velDist.eps

fixPSlinestyle('velDist.eps');


% Graphes for temperature change

figTemp = figure;
hold on;

temp1 = plot(yValues,temp.data(:,1));
set(temp1                            , ...
  'Marker'          , 's'           , ...
  'Color'           , [0 .5 0]      ,...
  'LineWidth'       , 1.           );
temp2 = plot(yValues,temp.data(:,2));
set(temp2                            , ...
  'Marker'          , '+'           , ...
  'Color'           , [0 .5 0]   ,...
  'LineWidth'       , 1.           );
temp3 = plot(yValues,temp.data(:,3));
set(temp3                            , ...
  'Marker'          , '^'           , ...
  'Color'           , [0 .5 0]   ,...
  'LineWidth'       , 1.           );

temp4 = plot(yValues,temp.data(:,4));
set(temp4                            , ...
  'Marker'          , '*'           , ...
  'Color'           , [0 .5 0]    ,...
  'LineWidth'       , 1.           );


hTitle  = title ('Temperature distribution for different ...
values of Ha ');
hYLabel = ylabel('Temperature'                      );
hXLabel = xlabel('Length - Y'                       );
```

```
hLegend = legend( ...
  [temp1, temp2, temp3, temp4], ...
  'Ha = 1' , ...
  'Ha = 2'      , ...
  'Ha = 3'       , ...
  'Ha = 4' , 1);

set( gca                       , ...
    'FontName'   , 'Helvetica' );
set([hTitle, hXLabel, hYLabel], ...
    'FontName'   , 'AvantGarde');
set([hLegend, gca]             , ...
    'FontSize'   , 12           );
set([hXLabel, hYLabel]  , ...
    'FontSize'   , 12          );
set( hTitle                    , ...
    'FontSize'   , 12          , ...
    'FontWeight' , 'bold'      );

set(gca, ...
  'Box'         , 'off'     , ...
  'TickDir'     , 'out'     , ...
  'TickLength'  , [.02 .02] , ...
  'XMinorTick'  , 'on'      , ...
  'YMinorTick'  , 'on'      , ...
  'YGrid'       , 'on'      , ...
  'XColor'      , [.3 .3 .3], ...
  'YColor'      , [.3 .3 .3], ...
  'LineWidth'   , 1         );

% hText   = text(10, 800, ...
%   sprintf('\\it{C = %0.1g \\pm %0.1g (CI)}', ...
%   c, cint(2)-c));
set(gcf, 'PaperPositionMode', 'auto');
print -depsc2 temp.eps

fixPSlinestyle('temp.eps');
```

```
%%%%%%%%%% This code is written by Elgiz Baskaya %%%%%%%%%%%%%%
%%%% 21.11.2014 for questions mail to elgizbaskaya@gmail.com %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Investigation of  the problem on dimensionless parameters
% (Gr/Re, Ha, Br) for the given number of grids (N)
tic;
% close all;
clear all;
clc;
close all;
```

```
N = 20;
yValues = CGS_grid(0,1,N);

% griddedAxes = equalGridding();
% yValues = griddedAxes.y';

% Dependence on Hartmann Number

% Pressure Gradient
G = 1;

% epsil = 0.1;
Bi0 = 0.1;
Bi1 = 0.1;

% Ha = 1;
Ec = 0.1;
Pr = 0.71;
Br = Ec * Pr;
Re = 100;
% Br / Gama = 0.1
Gama = 0.71;
epsil = 0.1;

velocityAnal = zeros(length(yValues),4);
NG = zeros(length(yValues), 11, 11);
Nh = zeros(length(yValues), 11, 11);
Nf = zeros(length(yValues), 11, 11);
Nm = zeros(length(yValues), 11, 11);
NGav = zeros(11,11);
Nhav = zeros(11,11);
Nfav = zeros(11,11);
Nmav = zeros(11,11);

% vel(:,k) = velocity;
% temp(:,k) = temperature;

i = 1;
k = 1;
for  nonDimV = 0 : 0.1 : 1

    for Ha = 0 : 1 : 10

        [velocity,temperature,S,NheatTrans,NfluidFric,...
        NmagField] = oneDimPorousChannelwithMagFPartialBound...
        (G, epsil, Bi0, Bi1, Ha, Pr, Re, Br, Gama, nonDimV);

        % Analytical Solution valid only for epsil = 0 which
        % means constant fluid viscosity
        alfa = (Re + sqrt(Re^2 + 4* Ha)) / 2;
        beta = (Re - sqrt(Re^2 + 4 * Ha)) / 2;
        velocityAnal.data(:,i,k) = G / Ha * ...
        (((exp(alfa * yValues) * (exp(beta) - 1) - ...
        exp(beta * yValues) * (exp(alfa) - 1)) / ...
        (exp(alfa) - exp(beta))) + 1);

        vel.data(:,i,k) = velocity;
        temp.data(:,i,k) = temperature;

        NG(:,i,k) = S;         % Local entropy generation rate
                               % NG = Nh + Nf + Nm
        Nh(:,i,k) = NheatTrans; % Irreversibility due to heat
                                              % transfer
        Nf(:,i,k) = NfluidFric; % Entropy genereation due to
                                % viscous dissipation
        Nm(:,i,k) = NmagField;  % Entropy generation due to the
                                % effect of magnetic field
                                % (Joule heating or Ohmic heating)


        NGav(i,k) = S’ * yValues;   % Dimensionless total entropy
                                   % generation rate
        Nhav(i,k) = NheatTrans’ * yValues; % Dimensionless total
                                       % entropy generation
                                       % rate due to
                                       % heat transfer
        Nfav(i,k) = NfluidFric’ * yValues; % Dimensionless total
                                   % entropy generation
                                       % rate due to
                                       % heat transfer
        Nmav(i,k) = NmagField’ * yValues; % Dimensionless total
                                     % entropy generation
                                      % rate due
                                      % to heat transfer


        k = k + 1;

    end
    k = 1;
    i = i + 1;
end

 Be = Nh ./ NG; % Bejan number



% % FIGURES
%
% % This part of code should be modified for changing
% % numbers of velocity and temperature profiles.
% % Now it simulates for 3 different Ha and 3 different
% % Re configurations resulting 9 different conditions.
% % ( Examp : Re 0, Ha 1; Re 0 , Ha 2; ...)
%
% figVel = figure;
% hold on;
%
% vel1 = plot(yValues, vel.data(:,1,1));
% set(vel1                            , ...
%   ’Marker’          , ’s’         , ...
%   ’Color’           , [0 0 .5]    ,...
%   ’LineWidth’       , 1.2         );
% vel2 = plot(yValues, vel.data(:,1,6));
% set(vel2                            , ...
%   ’Marker’          , ’+’         , ...
%   ’Color’           , [0 0 .5]  ,...
%   ’LineWidth’       , 1.2         );
% vel3 = plot(yValues, vel.data(:,1,11));
% set(vel3                            , ...
%   ’Marker’          , ’^’         , ...
%   ’Color’           , [0 0 .5]  ,...
%   ’LineWidth’       , 1.2         );
% vel4 = plot(yValues, vel.data(:,6,1));
% set(vel4                            , ...
%   ’LineStyle’       ,’:’         , ...
%   ’Marker’          , ’s’         , ...
%   ’Color’           , [1 0 0]   ,...
%   ’LineWidth’       , 1.2         );
% vel5 = plot(yValues, vel.data(:,6,6));
% set(vel5                            , ...
%   ’LineStyle’       ,’:’         , ...
%   ’Marker’          , ’+’         , ...
%   ’Color’           , [1 0 0]   ,...
%   ’LineWidth’       , 1.1         );
% vel6 = plot(yValues, vel.data(:,6,11));
% set(vel6                            , ...
%   ’LineStyle’       ,’:’         , ...
%   ’Marker’          , ’^’         , ...
%   ’Color’           , [1 0 0]  ,...
%   ’LineWidth’       , 1.2         );
% vel7 = plot(yValues,vel.data(:,11,1));
% set(vel7                            , ...
%   ’LineStyle’       ,’--’         , ...
%   ’Marker’          , ’s’         , ...
%   ’Color’           , [0 0.5 0]   ,...
%   ’LineWidth’       , 1.2         );
% vel8 = plot(yValues, vel.data(:,11,6));
% set(vel8                            , ...
%   ’LineStyle’       ,’--’         , ...
%   ’Marker’          , ’+’         , ...
%   ’Color’           , [0 .5 0]   ,...
%   ’LineWidth’       , 1.2         );
% vel9 = plot(yValues, vel.data(:,11,11));
% set(vel9                            , ...
%   'LineStyle'       ,'--'         , ...
%   'Marker'          , '^'         , ...
%   'Color'           , [0 0.5 0]  ,...
%   'LineWidth'       , 1.2         );
%
% %hTitle  = title ('Velocity Profile for different ...
% values of Ha and Re numbers');
% hXLabel = xlabel('Y'                     );
% hYLabel = ylabel('w(Y)'                      );
%
% hLegend = legend( ...
%   [vel1, vel2, vel3, vel4, vel5, vel6, vel7, vel8, vel9], ...
%   '\epsilon = 0, Ha  = 0 ' , ...
%   '\epsilon = 0, Ha  = 5'       , ...
%   '\epsilon = 0, Ha  = 10'    , ...
%   '\epsilon = 0.05, Ha  = 0' , ...
%   '\epsilon = 0.05, Ha  = 5'      , ...
%   '\epsilon = 0.05, Ha  = 10'                , ...
%   '\epsilon = 0.1, Ha  = 0'                , ...
%   '\epsilon = 0.1, Ha  = 5'    , ...
%   '\epsilon = 0.1, Ha  = 10', 2);
%
% set( gca                       , ...
%     'FontName'   , 'Helvetica' );
% % set([hTitle, hXLabel, hYLabel], ...
% %     'FontName'   , 'AvantGarde');
% set([hLegend, gca]             , ...
%     'FontSize'   , 11           );
% set([hXLabel, hYLabel]  , ...
%     'FontSize'   , 12          );
% % set( hTitle                    , ...
% %     'FontSize'   , 12          , ...
% %     'FontWeight' , 'bold'      );
%
% set(gca, ...
%   'Box'         , 'off'     , ...
%   'TickDir'     , 'out'     , ...
%   'TickLength'  , [.02 .02] , ...
%   'XMinorTick'  , 'on'      , ...
%   'YMinorTick'  , 'on'      , ...
%   'YGrid'       , 'on'      , ...
%   'XColor'      , [.3 .3 .3], ...
%   'YColor'      , [.3 .3 .3], ...
%   'LineWidth'   , 1         );
%
% % hText   = text(10, 800, ...
% %   sprintf('\\it{C = %0.1g \\pm %0.1g (CI)}', ...
% %   c, cint(2)-c));
% set(gcf, 'PaperPositionMode', 'auto');
% print -depsc2 '-r300' velDistEpsilonSmallerRange.eps
```

```
%
% fixPSlinestyle('velDistEpsilonSmallerRange.eps', ...
% 'velDistEpsilonSmallerRange2.eps');
%
%
% % Graphes for temperature change
%
% figTemp = figure;
% hold on;
%
% temp1 = plot(yValues, temp.data(:,1,1));
% set(temp1                            , ...
%   'Marker'          , 's'         , ...
%   'Color'           , [0 0 .5]    ,...
%   'LineWidth'       , 1         );
% temp2 = plot(yValues, temp.data(:,1,6));
% set(temp2                            , ...
%   'Marker'          , 'p'         , ...
%   'Color'           , [0 0 .5]  ,...
%   'LineWidth'       , 1         );
% temp3 = plot(yValues, temp.data(:,1,11));
% set(temp3                            , ...
%   'Marker'          , '^'         , ...
%   'Color'           , [0 0 .5]  ,...
%   'LineWidth'       , 1         );
% temp4 = plot(yValues, temp.data(:,6,1));
% set(temp4                            , ...
%   'LineStyle'       ,':'         , ...
%   'Marker'          , 's'         , ...
%   'Color'           , [1 0 0]   ,...
%   'LineWidth'       , 1.1        );
% temp5 = plot(yValues, temp.data(:,6,6));
% set(temp5                            , ...
%   'LineStyle'       ,':'         , ...
%   'Marker'          , 'p'         , ...
%   'Color'           , [1 0 0]   ,...
%   'LineWidth'       , 1         );
% temp6 = plot(yValues, temp.data(:,6,11));
% set(temp6                            , ...
%   'LineStyle'       ,':'         , ...
%   'Marker'          , '^'         , ...
%   'Color'           , [1 0 0]  ,...
%   'LineWidth'       , 1.1         );
%
% temp7 = plot(yValues, temp.data(:,11,1));
% set(temp7                            , ...
%   'LineStyle'       ,'--'         , ...
%   'Marker'          , 's'         , ...
%   'Color'           , [0 0.5 0]   ,...
%   'LineWidth'       , 1        );
% temp8 = plot(yValues, temp.data(:,11,6));
% set(temp8                            , ...
%   'LineStyle'       ,'--'         , ...
%   'Marker'          , 'p'         , ...
%   'Color'           , [0 .5 0]   ,...
%   'LineWidth'       , 1         );
% temp9 = plot(yValues, temp.data(:,11,11));
% set(temp9                            , ...
%   'LineStyle'       ,'--'         , ...
%   'Marker'          , '^'         , ...
%   'Color'           , [0 0.5 0]  ,...
%   'LineWidth'       , 1         );
%
% hLegend = legend( ...
%   [temp1, temp2, temp3, temp4, temp5, temp6, temp7, ...
%    temp8, temp9], ...
%   '\epsilon = 0, Ha  = 0 ' , ...
%   '\epsilon = 0, Ha  = 5'        , ...
%   '\epsilon = 0, Ha  = 10'    , ...
%   '\epsilon = 0.05, Ha  = 0' , ...
%   '\epsilon = 0.05, Ha  = 5'      , ...
%   '\epsilon = 0.05, Ha  = 10'                , ...
%   '\epsilon = 0.1, Ha  = 0'                , ...
%   '\epsilon = 0.1, Ha  = 5'    , ...
%   '\epsilon = 0.1, Ha  = 10', 1);
%
%
%
%
% %hTitle  = title ('Temperature Distribution for ...
% % different values of Ha and Re numbers');
% hXLabel = xlabel('Y'                     );
% hYLabel = ylabel('T(Y)'                      );
%
% % set(hLegend, 'Position', [0.65, 0.3, 0.25, 0.25]);
% set( gca                       , ...
%     'FontName'   , 'Helvetica' );
% % set([hTitle, hXLabel, hYLabel], ...
% %     'FontName'   , 'AvantGarde');
% set([hLegend, gca]             , ...
%     'FontSize'   , 11           );
% set([hXLabel, hYLabel]  , ...
%     'FontSize'   , 12          );
% % set( hTitle                    , ...
% %     'FontSize'   , 12          , ...
% %     'FontWeight' , 'bold'      );
%
% set(gca, ...
%   'Box'         , 'off'     , ...
%   'TickDir'     , 'out'     , ...
%   'TickLength'  , [.02 .02] , ...
%   'XMinorTick'  , 'on'      , ...
%   'YMinorTick'  , 'on'      , ...
%   'YGrid'       , 'on'      , ...
%   'XColor'      , [.3 .3 .3], ...
%   'YColor'      , [.3 .3 .3], ...
%   'LineWidth'   , 1         );
%
% magnifyOnFigure(...
%         figTemp,...
%         'units', 'pixels',...
%         'magnifierShape', 'ellipse',...
%         'initialPositionSecondaryAxes', ...
%          [210 300 140 80],...
%         'initialPositionMagnifier', ...
%          [290 230 40 40],...
%         'mode', 'interactive',...
%         'displayLinkStyle', 'straight',...
%         'edgeWidth', 1,...
%         'edgeColor', [0 0 0.5],...
%         'secondaryAxesFaceColor', ...
%          [0.91 0.91 0.91]...
%             );
%
% magnifyOnFigure(...
%         figTemp,...
%         'units', 'pixels',...
%         'magnifierShape', 'ellipse',...
%         'initialPositionSecondaryAxes', ...
%          [115 100 140 80],...
%         'initialPositionMagnifier',...
%          [290 160 40 40],...
%         'mode', 'interactive',...
%         'displayLinkStyle', 'straight',...
%         'edgeWidth', 1,...
%         'edgeColor', [0 0 0.5],...
%         'secondaryAxesFaceColor', ...
%          [0.91 0.91 0.91]...
%             );
%
% set(gcf, 'PaperPositionMode', 'auto');
% print -depsc2 tempDistEpsilonSmallerRange.eps
%
% fixPSlinestyle('tempDistEpsilonSmallerRange.eps', ...
% 'tempDistEpsilonSmallerRange2.eps');

% % hText   = text(10, 800, ...
% %   sprintf('\\it{C = %0.1g \\pm %0.1g (CI)}', ...
% %   c, cint(2)-c));
%
%
% %'LineStyle'       ,'-.'         , ...
%
% %    figure(1);
% %    plot(velocity,yValues)
% %    legend(cellstr(num2str((0:0.5:1)')))
% %    legend(cellstr(strcat(’\psi  =...
% %     ’,num2str((0:0.01:0.06)’))))
% %    legend(’ha=%d’,Ha)
% %    Ha = % d \n’,Ha
% %    xlabel(’Velocity’)
% %    ylabel(’Y’)
% %    hold all
% %    figure(2);
% %    plot(temperature,yValues)
% %    legend(cellstr(strcat(’\psi  =...
% %     ’,num2str((0:0.01:0.06)’))))
% %    xlabel(’Temperature’)
% %    ylabel(’Y’)
% %    hold all
%
% % % Entropy Generation
% %
% % plot(yValues, S(:,3,1))
% %
% %
% % figEntroGen = figure;
% % hold on;
% %
% % entroGen = plot(yValues, S);
% % set(entroGen                            , ...
% %   ’Marker’          , ’s’         , ...
% %   ’Color’           , [0 0 .5]    ,...
% %   ’LineWidth’       , 1.2         );

save channelWorkspace_35;

toc;
```

```
% Written by Elgiz Baskaya
% elgizbaskaya@gmail.com 13.01.2014

% This code solves the one dimensional channel problem
% in the presence of a constant magnetic field and under
% constant pressure gadient. The Nonlinear coupled
% differential equations are discretized using GDQM method.
% Then the nonlinear algeabraic equation set is solved
%utilizing NewtonRaphson Method

function [velocity,temperature,S,NheatTrans,NfluidFric,...
NmagField]=oneDimPorousChannelwithMagFPartialBound(G, ...
epsil, Bi0, Bi1, Ha, Pr, Re, Br, Gama, nonDimV)

% Problem Definition

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% Gridding

% Option 1. Equal Gridding

% Call function equalGridding to calculate gridded axes values.
% This problem is one directional and the state variables only
% change in y direction. Thus a valid input to the function
% when called would be like [0 10 0] meaning that only y
% axis will be gridded to 5 elements and [0 0 -1 1 0 0]
% meaning the lower and upper values of y will be -1 and 1
% respectively
% griddedAxes = equalGridding();
% yValues = griddedAxes.y';

% Option 2. Gauss Lobato Gridding
% Chyebisev Gauss Lobato gridding
%
N = 20;
yValues = CGS_grid(0,1,N);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% number of unknowns n
% Number of  different functions sets
fn = 2; % Since we only try to find velocity and temperature
m = length(yValues);
n = fn * m;

% Vector of unknown variables
% x_1 ... x_m representing U_21 ... U_2M
% x_m+1 ... x_2m representing theta_21 ... theta_2M

% Unknown variables to be calculated written symbolically
% here
xSym = sym('x',[n 1]);

% soyle yapilirsa class(x1) = sym oluyor; yukardaki gibi
% olmuyor.
% syms U T;
% U=sym(zeros(N,1));
% Th=sym(zeros(N,1));
%
% for n=1:1:N
%     eval(strcat('syms',' u',num2str(n)));
%     U(n)=eval(strcat('u',num2str(n)));
%     eval(strcat('syms',' th',num2str(n)));
%     Th(n)=eval(strcat('th',num2str(n)));
% end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Initialize
```

```
% Weight coefficients for first and second order derivatives
weightCoef2nd = zeros(m,m);
weightCoef1st = zeros(m,m);
F = sym(zeros(2 * m, 1));

% Function in symbolic variables to take the Jacobian
for j = 1 : m
    for k = 1 : m
       weightCoef2nd(j,k) = ...
       calculateSecondOrderCoef(yValues,j,k);
       weightCoef1st(j,k) = ...
       calculateFirstOrderCoef(yValues,j,k);
    end

    % Momentum Equations in discreatized form
    F(j) = weightCoef2nd(j, :) * xSym(1 : m) - ...
    epsil * (weightCoef1st(j, :) * ...
    xSym(m + 1 : 2 * m)) * (weightCoef1st(j, :) * ...
    xSym(1 : m)) - exp(epsil * xSym(m + j)) * ...
    (nonDimV * Re * weightCoef1st(j, :) * ...
    xSym(1 : m) + Ha^2 * xSym(j) - G);

    % Boundary Conditions for temperature
    if j == 1
        F(m + j) = weightCoef1st(1, :) * ...
        xSym(m + 1 : 2 * m) - Bi0 * ...
        (xSym(m + j) - 1); % Equation (8)
    elseif j == m
        F(m + j) = weightCoef1st(m, :) * ...
        xSym(m + 1 : 2 * m) + Bi1 * ...
        xSym(m + j);       % Equation (8)
    else
   % Energy Equations in discretized form
    F(m + j) = weightCoef2nd(j, :) * ...
    xSym(m + 1 : 2 * m) - nonDimV * Re *...
    Pr * weightCoef1st(j, :) * xSym(m + 1 : 2 * m) + ...
    Br * exp(- epsil * xSym(m + j)) * ...
    (weightCoef1st(j, :) * xSym(1 : m))^2 + ...
    Br * Ha^2 * (xSym(j))^2;
    end
end

% Boundary conditions for velocity u(0) = u(L) = 0
F = subs(F, [xSym(1) xSym(m)], [0 0]);

% Functions corresponding to boundary conditions
F(1,:) = [];
F(m - 1 ,:) = [];

xSym(1) = [];
xSym(m - 1) = [];
```

```
% Take the Jacobian of the function
jacob = jacobian(F, xSym(:));

% Newton Raphson method is utilized in this section
% to solve the nonlinear algeabraic equation set
% developed above

tol = 1e-3;

% Initial guess for the state vector
%Guess = [ 0.9 1.2 1.3 1.1 1.1 1.3 1.2 0.9 0.9 0.8 ...
% 0.7 0.6 0.5 0.4 0.3 0.2]’;
%x = Guess;
%Guess = sin(pi * yValues(2:m-1)/2 +pi/2)’;
x = [ones(m - 2,1); ones(m,1);];

%set the error 2*tol to make sure the loop runs at
% least once
error = 2*tol;

% preallocating
velocity = zeros(m - 2 , 1);
temperature = zeros(m, 1);

while error > tol
% %calculate the function values at the current iteration
% F = feval(MyFunc,x);
% %calculate the jacobian matrix
% J = feval(Jacobian,x);

F_eval = double(subs(F,xSym,x));
J_eval = double(subs(jacob, xSym,x));

%calculate the update (solve the linear system)
dx = - J_eval \ F_eval;

%update the x value
x = x + dx;

% divide the unknown vector to unknown states
velocityCalc = x(1 : m - 2);
temperatureCalc = x(m - 1 : 2 * m - 2);
% include boundaries conditions
velocity = [0; velocityCalc ;0];
temperature = temperatureCalc;

%calculate the error
F_eval = double(subs(F,xSym,x));
error = max(abs(F_eval));
end %while loop
```

```
% Entropy Generation

NheatTrans = (weightCoef1st * temperature).^2;
NfluidFric = Br / Gama * (exp(-epsil * temperature) .* ...
(weightCoef1st * velocity).^2);
NmagField = Br / Gama * (Ha * velocity.^2);

% dimensionless local entropy generation rate
 S = NheatTrans + NfluidFric + NmagField;
```

```
% Written by elgiz elgizbaskaya@gmail.com
% on 21.10.2014


% Entropy generation graphes
% Change of total entropy generation with respect to Re
% number for different Ha numbers.

function [] = plotGeneration()

load channel_HaandRe_from0to60;

figure;
%mrk=; %These are the markers
    set(0,'defaultaxescolororder')
    set(0,'defaultaxeslinestyleorder',...
    {'-o','-p','*-','-v','-+','-^','-.',':x','-d',...
    '--<','-->','-p',':h'})
%     set(0,'defaultLineLineWidth','remove')
%

    plot(0:2:60, Nhav(:,1),'r-o');
%     hold on
%     plot(0:2:60, Nhav(16,:),'g-p');
    hold on
    plot(0:2:60, Nhav(:,11),'*--');
    hold on
    plot(0:2:60, Nfav(:,1),'r-x');
%     hold on
%     plot(0:2:60, Nfav(16,:),'g-p');
    hold on
    plot(0:2:60, Nfav(:,11),'+--');
    hold on
    plot(0:2:60, Nmav(:,1),'r-p');
%     hold on
%     plot(0:2:60, Nmav(16,:),'g-p');
    hold on
    plot(0:2:60, Nmav(:,11),'-->');

hLegend = legend( ...
  'Nh_a_v for Ha = 0'         ,...
  'Nh_a_v for Ha = 20'    , ...
  'Nf_a_v for Ha = 0'    ,...
  'Nf_a_v for Ha = 20'        ,...
  'Nm_a_v for Ha = 0'    , ...
  'Nm_a_v for Ha = 20'        ,...
  'Nf_a_v for Ha = 2'   ,...
  'Nh_a_v for Ha = 2'        ,...
  'Nf_a_v for Ha = 2'    , 1);



% hTitle  = title ('Velocity Profile for  ...
% different values of Ha and Re numbers');
hXLabel = xlabel('Re'                );
hYLabel = ylabel('Nh_a_v, Nf_a_v, Nm_a_v '                );
set( gca                       , ...
    'FontName'   , 'Helvetica' );
% set([hTitle, hXLabel, hYLabel], ...
%     'FontName'   , 'AvantGarde');
set([hLegend, gca]             , ...
    'FontSize'   , 11          );
set([hXLabel, hYLabel]  , ...
    'FontSize'   , 12         );
% set( hTitle                    , ...
%     'FontSize'   , 12          , ...
%     'FontWeight' , 'bold'      );

set(gca, ...
  'Box'          , 'off'     , ...
  'TickDir'      , 'out'     , ...
  'TickLength'   , [.02 .02] , ...
  'XMinorTick'   , 'on'      , ...
  'YMinorTick'   , 'on'      , ...
  'YGrid'        , 'on'      , ...
  'XColor'       , [.3 .3 .3], ...
  'YColor'       , [.3 .3 .3], ...
  'LineWidth'    , 1         );

print -depsc2 '-r300' equipartitioningwrtRe_Ha0_20.eps
```

1.2 Codes for Nanofluid Inclined Channel Flow

```
% Written by Elgiz Baskaya
% elgizbaskaya@gmail.com 11.02.2014

% This code solves the one dimensional inclined channel filled
% with nanofluids in the presence of a constant magnetic field.
% The Nonlinear coupled differential equations are discretized
% using GDQM method. Then the nonlinear algeabraic equation
% set is solved utilizing NewtonRaphson Method

%% Problem Definition

clc;
clear all;
close all;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Constants given in the problem

% F  = index indicates fluids      -> For this study the base
%                                    fluid is water
% P  = index indicates particles   -> For this study the
%                                  nanoparticles are  copper
% NF = index indicates nanofluids

P = 1;
Br = 0.1;
Ra = 1e1;    % 1e7 is a critical value after which the
% solution diverges
Ha = 0;
Pr = 6.5;    % Around 7 for water
% REF : "MHD natural convection and entropy generation in
% a trapeziodal enclosure using Cu-water nanofluid"
%  Amir Housang Mahmoudi et all.
fi = pi / 6; % should be rad
volFrac = 0.06; % volume fraction of solid nanoparticles

muF = 0.855e-3;  % [N * s / m^2] - Dynamic Viscosity of
% base fluid (@ T = 300K ) REF : "Effects of Various
% Parameters on Nanofluid Thermal Conduction"
% J Choi Stephen
kF  = 0.613;     % [W / m / K]   - Thermal Conductivity of
% base fluid (@ T = 300K ) REF : "Effects of Various
% Parameters on Nanofluid Thermal Conduction"
% J Choi Stephen
kP  = 401;       % [W / m / K]   - Thermal Conductivity of
% nanoparticle (@ T = 300K )
% REF : "Effects of Various Parameters on Nanofluid
% Thermal Conduction" J Choi Stephen
roF = 997;       % [m^3 / kg]    - Density of base fluid
% (@ T = 300K ) REF : "Effects of Various Parameters
% on Nanofluid Thermal Conduction" J Choi Stephen
roP = 8933;      % [m^3 / kg]    - Density of nanoparticles
% (@ T = 300K ) REF : "Effects of Various Parameters
% on Nanofluid Thermal Conduction" J Choi Stephen
CpF = 4170;      % [J / kg / K]  - Specific Heat of base fluid
% (@ T = 300K ) REF : "Effects of Various Parameters
% on Nanofluid Thermal Conduction" J Choi Stephen
betaF = 2.1e-4;  % [1 / K]    - Coefficient of volume
% expansion REF : "MHD natural convection and
% entropy generation in a trapeziodal enclosure using
% Cu-water nanofluid" Amir Housang Mahmoudi et all.
betaP = 1.67e-5; % [1 / K]    - Coefficient of volume
% expansion REF : "MHD natural convection and entropy
% generation in a trapeziodal enclosure using
% Cu-water nanofluid" Amir Housang Mahmoudi et all.

muNF = muF / ((1 - volFrac)^(2.5));      % [N * s / m^2]
% Effective Dynamic Viscosity of Nanofluid -
% BRINKMAN MODEL - REF : "The Viscosity of
% Concentrated Suspensions and solutions,
% J. Chem. Phys.1952; 20: 571-81
alfaF = kF / roF / CpF;    % [m^2 / s] - Thermal diffusivity
%of the base fluid
roNF = (1 - volFrac) * roF + volFrac * roP;   % [m^3 / kg]
% Effective Density of Nanofluid
robetaNF =  (1 - volFrac) * roF * betaF + volFrac *...
 roP * betaP;
kNF = kF * (kP + 2 * kF - 2 * volFrac * (kF - kP)) / ...
(kP + 2 * kF + 2 * volFrac * (kF - kP));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Gridding

% Option 1. Equal Gridding

% Call function equalGridding to calculate gridded axes
% values. This problem is one directional and the state
% variables only change in y direction. Thus a valid input
% to the function when called would be like [0 10 0]
% meaning that only y axis will be gridded to 5 elements
% and [0 0 -1 1 0 0] meaing the lower and upper values
% of y will be -1 and 1 respectively
% griddedAxes = equalGridding();
% yValues = griddedAxes.y;

% Option 2. Gauss Lobato Gridding
% Chyebisev Gauss Lobato gridding

N = 20;
yValues = CGS_grid(-1,1,N);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% number of unknowns n
% Number of  different functions sets
fn = 2; % Since we only try to find U2 and theta_2
m = length(yValues);
n = fn * m;
```

```
% Vector of unknown variables
% x_1 ... x_m representing U_21 ... U_2M
% x_m+1 ... x_2m representing theta_21 ... theta_2M

% Unknown variables to be calculated written symbolically
% here
xSym = sym(’x’,[n 1]);

% soyle yapilirsa class(x1) = sym oluyor; yukardaki gibi
% olmuyor.
% syms U T;
% U=sym(zeros(N,1));
% Th=sym(zeros(N,1));
%
% for n=1:1:N
%     eval(strcat(’syms’,’ u’,num2str(n)));
%     U(n)=eval(strcat(’u’,num2str(n)));
%     eval(strcat(’syms’,’ th’,num2str(n)));
%     Th(n)=eval(strcat(’th’,num2str(n)));
% end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Initialize

% Weight coefficients for first and second order derivatives
weightCoef2nd = zeros(m,m);
weightCoef1st = zeros(m,m);
F = sym(zeros(2 * m, 1));

% Function in symbolic variables to take the Jacobian
for j = 1 : m
    for k = 1 : m
       weightCoef2nd(j,k) = ...
       calculateSecondOrderCoef(yValues,j,k);
       weightCoef1st(j,k) = calculateFirstOrderCoef...
        (yValues,j,k);
    end
    F (j) = muNF / alfaF / roNF * weightCoef2nd(j,:) * ....
    xSym(1:m) + (robetaNF / (roNF * betaF) )* sin(fi) * ...
    Ra * Pr * xSym(m + j) - Pr * Ha^2 * xSym(j) + P;
    F (m + j) = weightCoef2nd(j,:) * xSym(m + 1:2*m) + ...
    kF / kNF * muNF / muF * Br * (weightCoef1st(j,:) * ...
    xSym(1:m))^2 + roNF / roF * kF / kNF * Br * ...
    Ha^2 * (xSym(j))^2;
end

% Boundary conditions
% Boundary conditions are given as u(0) = u(L) = 0
%                                  t(0) = 0 t(L) = 1

F = subs(F ,[xSym(1) xSym(m) xSym(m+1) xSym(2*m)], ...
[0 0 0 1 ]);
```

```
% Functions corresponding to boundary conditions
F(1,:) = [];
F(m - 1 ,:) = [];
F(m + 1 -2, :) = [];
F(2 * m -3,:) = [];

% F(:,1) = [];
% F( :,m - 1) = [];
% F(:,m + 1 -2 ) = [];
% F(:,2 * m -3) = [];

xSym(1) = [];
xSym(m - 1) = [];
xSym(m + 1 - 2) = [];
xSym(2 * m - 3) = [];

% Take the Jacobian of the function
jacob = jacobian(F, xSym(:));

%% Newton Raphson method is utilized in this section ...
% to solve the nonlinear algeabraic equation set developed
% above

tol = 1e-3;

% Initial guess for the state vector
%Guess = [ 0.9 1.2 1.3 1.1 1.1 1.3 1.2 0.9 0.9 0.8 ...
% 0.7 0.6 0.5 0.4 0.3 0.2]’;
%x = Guess;
Guess = ones(m - 2,1);
%Guess = sin(pi * yValues(2:m-1)/2 +pi/2)’;
x = [Guess; Guess];

%set the error 2*tol to make sure the loop runs at least once
error = 2*tol;

% preallocating
velocityCalc = zeros((length(x)/2),1);
temperatureCalc = zeros((length(x)/2),1);
velocity = zeros((length(x)/2) + 2,1);
temperature = zeros((length(x)/2) + 2,1);

while error > tol
% %calculate the function values at the current iteration
% F = feval(MyFunc,x);
% %calculate the jacobian matrix
% J = feval(Jacobian,x);

F_eval = subs(F,xSym,x);
J_eval = subs(jacob, xSym,x);
```

```
%calculate the update (solve the linear system)
dx = - J_eval \ F_eval;

%update the x value
x = x + dx;

% divide the unknown vector to unknown states
velocityCalc = x(1:(size(x)/2));
temperatureCalc = x(size(x)/2+1 :(size(x)));
% include boundaries conditions
velocity = [0; velocityCalc ;0];
temperature = [0 ;temperatureCalc; 1];


%calculate the error
F_eval = subs(F,xSym,x);
error = max(abs(F_eval));
end %while loop
```

```
% Written by Elgiz Baskaya
% elgizbaskaya@gmail.com 11.02.2014

% This code solves the one dimensional inclined channel filled
% with nanofluids in the presence of a constant magnetic field.
% The Nonlinear coupled differential equations are discretized
% using GDQM method. Then the nonlinear algeabraic equation
% set is solved utilizing NewtonRaphson Method

function [velocity, temperature,S,NheatTrans,NfluidFric,...
NmagField] = oneDimIncChannelwithMagF(P,Ha,Br,Ra,...
fi,alfa, volFrac,Gama)

% Problem Definition
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Constants given in the problem

% F  = index indicates fluids      -> For this study the base
% fluid is water
% P  = index indicates particles   -> For this study the
% nanoparticles are copper
% NF = index indicates nanofluids

% P = 1;
% Ha = 1;
% Br = 0.1;
% Ra = 1e5;    % 1e7 is a critical value after which the ...
% solution diverges
% Pr = 6.5;    % Around 7 for water - REF : "MHD natural ...
% convection and entropy generation in a trapeziodal ...
% enclosure using Cu-water ...
% nanofluid" Amir Housang Mahmoudi et all. Table 1
% fi = pi / 6; % should be rad
% volFrac = 0.06; % volume fraction of solid nanoparticles


muF = 0.855e-3;  % [N * s / m^2] - Dynamic Viscosity of base
% fluid (@ T = 300K ) REF : "Effects of Various Parameters
% on Nanofluid Thermal Conduction" J Choi Stephen
kF  = 0.613;     % [W / m / K]   - Thermal Conductivity of
% base fluid (@ T = 300K ) REF : "Effects of Various
% Parameters on Nanofluid Thermal Conduction"
% J Choi Stephen
kP  = 401;       % [W / m / K]   - Thermal Conductivity of
% nanoparticle (@ T = 300K ) REF : "Effects of Various
% Parameters on Nanofluid Thermal Conduction"
% J Choi Stephen
roF = 997;       % [m^3 / kg]    - Density of base fluid
% (@ T = 300K ) REF : "Effects of Various Parameters on
% Nanofluid Thermal Conduction" J Choi Stephen
roP = 8933;      % [m^3 / kg]    - Density of nanoparticles
% (@ T = 300K ) REF : "Effects of Various Parameters on
% Nanofluid Thermal Conduction" J Choi Stephen
CpF = 4170;      % [J / kg / K]  - Specific Heat of base fluid
% (@ T = 300K ) REF : "Effects of Various Parameters on
% Nanofluid Thermal Conduction" J Choi Stephen
betaF = 2.1e-4;  % [1 / K]       - Coefficient of volume
% expansion
% REF : "MHD natural convection and entropy generation
% in a trapeziodal enclosure using Cu-water nanofluid"
% Amir Housang Mahmoudi et all.
betaP = 1.67e-5; % [1 / K]       - Coefficient of volume
% expansion
% REF : "MHD natural convection and entropy generation in
% a trapeziodal enclosure using Cu-water nanofluid"
% Amir Housang Mahmoudi et all.
% sgmaF = 0.05
% sgmaP = 5.97e7

muNF = muF / (1 - volFrac)^(2.5);    % [N * s / m^2]
% Effective Dynamic Viscosity of Nanofluid
% BRINKMAN MODEL REF : "The Viscosity of
% Concentrated Suspensions and solutions,
% J. Chem. Phys.1952; 20: 571-81
alfaF = kF / roF / CpF;              % [m^2 / s]
% Thermal diffusivity of the base fluid
roNF = (1 - volFrac) * roF + volFrac * roP;  % [m^3 / kg]
% Effective Density of Nanofluid
%sgmaNF = (1 - volFrac) * sgmaF + volFrac * sgmaP
robetaNF =  (1 - volFrac) * roF * betaF + ...
volFrac * roP * betaP;
kNF = kF * (kP + 2 * kF - 2 * volFrac * (kF - kP)) / ...
(kP + 2 * kF + 2 * volFrac * (kF - kP));
```

```
Pr = muF * CpF / kF;

% Needed : Bo, H, sgmaNF, (TH - TC),
% Ha = Bo * H * sqrt(sgmaNF / roNF / (muF/roF))
% Ec = alfaF ^ 2 / H ^ 2 / CpF / (TH - TC)
% Br = Ec * Pr
% Ra = g * betaF * H ^ 3 * (TH - TC) / (muF / roF) / alfaF
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Gridding

% Option 1. Equal Gridding

% Call function equalGridding to calculate gridded axes
% values. This problem is one directional and the state
% variables only change in y direction. Thus a valid input
% to the function when called would be like [0 10 0]
% meaning that only y axis will be gridded to 5 elements
% and [0 0 -1 1 0 0] meaing the lower and upper values
% of y will be -1 and 1 respectively
% griddedAxes = equalGridding();
% yValues = griddedAxes.y;

% Option 2. Gauss Lobato Gridding
% Chyebisev Gauss Lobato gridding

N = 11;
yValues = CGS_grid(-1,1,N);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% number of unknowns n
% Number of  different functions sets
fn = 2; % Since we only try to find U2 and theta_2
m = length(yValues);
n = fn * m;

% Vector of unknown variables
% x_1 ... x_m representing U_21 ... U_2M
% x_m+1 ... x_2m representing theta_21 ... theta_2M

% Unknown variables to be calculated written symbolically
% here
xSym = sym('x',[n 1]);

% soyle yapilirsa class(x1) = sym oluyor; yukardaki gibi
% olmuyor.
% syms U T;
% U=sym(zeros(N,1));
% Th=sym(zeros(N,1));

% for n=1:1:N
%     eval(strcat('syms',' u',num2str(n)));
%     U(n)=eval(strcat('u',num2str(n)));
%     eval(strcat('syms',' th',num2str(n)));
%     Th(n)=eval(strcat('th',num2str(n)));
% end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Initialize

% Weight coefficients for first and second order derivatives
weightCoef2nd = zeros(m,m);
weightCoef1st = zeros(m,m);
F = sym(zeros(2 * m, 1));

% Function in symbolic variables to take the Jacobian
for j = 1 : m
    for k = 1 : m
       weightCoef2nd(j,k) = ...
       calculateSecondOrderCoef(yValues,j,k);
       weightCoef1st(j,k) = ...
       calculateFirstOrderCoef(yValues,j,k);
    end
    F (j) = muNF / alfaF / roNF * weightCoef2nd(j,:) * ...
    xSym(1:m) + robetaNF / roNF / betaF * sin(fi) * ...
    Ra * Pr * xSym(m + j) - Pr * Ha^2 *...
    (sin(alfa + fi))^2 * xSym(j) + P;
    F (m + j) = weightCoef2nd(j,:) * xSym(m + 1:2*m) + ...
    kF / kNF * muNF / muF * Br * (weightCoef1st(j,:) * ...
    xSym(1:m))^2 + roNF / roF * kF / kNF * Br * Ha^2 * ...
    (sin(alfa + fi))^2 * (xSym(j))^2;
end

% Boundary conditions
% Boundary conditions are given as u(0) = u(L) = 0
%                                   t(0) = 0 t(L) = 1

F = subs(F ,[xSym(1) xSym(m) xSym(m+1) xSym(2*m)], ...
[0 0 0 1 ]);

% Functions corresponding to boundary conditions
F(1,:) = [];
F(m - 1 ,:) = [];
F(m + 1 -2, :) = [];
F(2 * m -3,:) = [];

% F(:,1) = [];
% F( :,m - 1) = [];
% F(:,m + 1 -2 ) = [];
% F(:,2 * m -3) = [];

xSym(1) = [];
xSym(m - 1) = [];
xSym(m + 1 - 2) = [];
xSym(2 * m - 3) = [];

% Take the Jacobian of the function
jacob = jacobian(F, xSym(:));

% Newton Raphson method is utilized in this section to
% solve the nonlinear algeabraic equation set developed
% above

tol = 1e-3;

% Initial guess for the state vector
%Guess = [ 0.9 1.2 1.3 1.1 1.1 1.3 1.2 0.9 0.9 0.8 0.7 ...
% 0.6 0.5 0.4 0.3 0.2]';
%x = Guess;
Guess = ones(m - 2,1);
%Guess = sin(pi * yValues(2:m-1)/2 +pi/2)';
x = [Guess; Guess];

%set the error 2*tol to make sure the loop runs at least once
error = 2*tol;

% preallocating
velocity = zeros((length(x)/2) + 2,1);
temperature = zeros((length(x)/2) + 2,1);

while error > tol
% %calculate the function values at the current iteration
% F = feval(MyFunc,x);
% %calculate the jacobian matrix
% J = feval(Jacobian,x);

F_eval = double(subs(F,xSym,x));
J_eval = double(subs(jacob, xSym,x));

%calculate the update (solve the linear system)
dx = - J_eval \ F_eval;

%update the x value
x = x + dx;

% divide the unknown vector to unknown states
velocityCalc = x(1:(size(x)/2));
temperatureCalc = x(size(x)/2+1 :(size(x)));
% include boundaries conditions
velocity = [0; velocityCalc ;0];
temperature = [0 ;temperatureCalc; 1];
```

```
%calculate the error
F_eval = double(subs(F,xSym,x));
error = max(abs(F_eval));
end %while loop

% Entropy Generation

NheatTrans = (weightCoef1st * temperature).^2;
NfluidFric = Br / Gama * ((weightCoef1st * velocity).^2);
NmagField = Br / Gama * (Ha * velocity.^2);

% dimensionless local entropy generation rate
 S = NheatTrans + NfluidFric + NmagField;
```

```
%%%%%%%%%% This code is written by Elgiz Baskaya %%%%%%%%%%%%%%
%%%% 21.02.2014 for questions mail to elgizbaskaya@gmail.com %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Investigation of  the problem on dimensionless parameters
% (P,Ha,Br,Ra,Pr,fi,volFrac) and inclination angle of the
% channel (fi [rad]) for the given number of grids (N)
close all;
clear;

tic;
N = 11;
yValues = CGS_grid(-1,1,N);

% Dependence on Hartmann Number

%Ha = 0;
P = 1;
Br = 0.1;
Ra = 20;    % 1e7 is a critical value after which the solution
% diverges
% fi = pi / 18; % should be rad
% alfa = pi / 2 - pi / 12;
Ha = 2;
volFrac = 0.03;
Gama = 0.71;
%volFrac = 0; % volume fraction of solid nanoparticles

%      % To check the Ha dependence of the flow is changing
%      % for different
%      % nanoparticle volume fraction
% velEx = zeros(length(yValues),1);
% i = 1;
% velChangeWithHa = zeros(1:length(yValues),1);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%     % To check the inclination angle dependence of the flow
%     % is changing for different nanoparticle volume fraction
% velEx = zeros(length(yValues),1);
% i = 1;
% velChangeWithInc = zeros(1:length(yValues),1);

% Initializations for preallocation speed
% velChangeWithInc = zeros(length(yValues),1);
% velChangeWithHa = zeros(length(yValues),1);
velChangeWithfi = zeros(length(yValues),1);
% tempChangeWithInc = zeros(length(yValues),1);
% tempChangeWithHa = zeros(length(yValues),1);
tempChangeWithfi = zeros(length(yValues),1);

i = 1;
k = 1;
for fi = 0: pi/18: pi/3

    for alfa = 0 : pi/18 : pi/2

        [velocity,temperature,S,NheatTrans,NfluidFric,NmagField] = ...
        oneDimIncChannelwithMagF(P,Ha,Br,Ra,fi,alfa,volFrac,Gama);
        vel.fi.name = 'velocity change with respect to channel ...
        inclination and magnetic field direction';
        vel.fi.alfa.data(:,i,k) = velocity;
        temp.fi.name = 'temperature change with respect to ...
        channel inclination and magnetic field direction';
        temp.fi.alfa.data(:,i,k) = temperature;

        NG(:,i,k) = S;                % Local entropy generation
        % rate  NG = Nh + Nf + Nm
        Nh(:,i,k) = NheatTrans;       % Irreversibility due to
        % heat transfer
        Nf(:,i,k) = NfluidFric;       % Entropy generation due
        % to viscous dissipation
        Nm(:,i,k) = NmagField;        % Entropy generation
        % due to the effect of magnetic field
        % (Joule heating or Ohmic heating)

        NGav(i,k) = S' * yValues;     % Dimensionless total
        % entropy generation rate
        Nhav(i,k) = NheatTrans' * yValues;    % Dimensionless
        % total entropy generation rate due to heat transfer
        Nfav(i,k) = NfluidFric' * yValues;    % Dimensionless
        % total entropy generation rate due to heat transfer
        Nmav(i,k) = NmagField' * yValues;    % Dimensionless
        % total entropy generation rate due to heat transfer

        k = k + 1
    end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

k = 1;
i = i + 1
%     % To check the Ha dependence of the flow is changing
%     %for different nanoparticle volume fraction
%     velChangeWithHa(1:length(yValues),i) = ...
%     (velEx - velocity) ./ velEx;
%     velEx = velocity;
%     i = i+1;

%     % To check the inclination angle dependence of the
%     % flow is changing for different nanoparticle volume
%     % fraction
%     velChangeWithInc(1:length(yValues),i) = ...
%     (velEx - velocity) ./ velEx;
%      velEx = velocity;
%     i = i+1;
end
%
% % Graphes for velocity change
%
% figVel = figure;
% hold on;
%
% vel1 = plot(vel.fi.alfa.data(:,1,1));
% set(vel1                            , ...
%   'Marker'          , 's'         , ...
%   'Color'           , [0 0 .5]    ,...
%   'LineWidth'       , 1.1         );
% vel2 = plot(vel.fi.alfa.data(:,1,2));
% set(vel2                            , ...
%   'Marker'          , '+'         , ...
%   'Color'           , [0 0 .5]  ,...
%   'LineWidth'       , 1.1         );
% vel3 = plot(vel.fi.alfa.data(:,1,3));
% set(vel3                            , ...
%   'Marker'          , '^'         , ...
%   'Color'           , [0 0 .5]  ,...
%   'LineWidth'       , 1.1         );
%
% vel4 = plot(vel.fi.alfa.data(:,2,1));
% set(vel4                            , ...
%   'LineStyle'       ,'--'         , ...
%   'Marker'          , 's'         , ...
%   'Color'           , [0 .5 0]   ,...
%   'LineWidth'       , 1.1         );
% vel5 = plot(vel.fi.alfa.data(:,2,2));
% set(vel5                            , ...
%   'LineStyle'       ,'--'         , ...
%   'Marker'          , '+'         , ...
%   'Color'           , [0 .5 0]   ,...
%   'LineWidth'       , 1.1         );
% vel6 = plot(vel.fi.alfa.data(:,2,3));
% set(vel6                            , ...
%   'LineStyle'       ,'--'         , ...
%   'Marker'          , '^'         , ...
%   'Color'           , [0 .5 0]   ,...
%   'LineWidth'       , 1.1         );
% vel7 = plot(vel.fi.alfa.data(:,3,1));
% set(vel7                            , ...
%   'LineStyle'       ,'-.'         , ...
%   'Marker'          , 's'         , ...
%   'Color'           , [1 0 0]    ,...
%   'LineWidth'       , 1.1         );
% vel8 = plot(vel.fi.alfa.data(:,3,2));
% set(vel8                            , ...
%   'LineStyle'       ,'-.'         , ...
%   'Marker'          , '+'         , ...
%   'Color'           , [1 0 0] ,...
%   'LineWidth'       , 1.1         );
% vel9 = plot(vel.fi.alfa.data(:,3,3));
% set(vel9                            , ...
%   'LineStyle'       ,'-.'         , ...
%   'Marker'          , '^'         , ...
%   'Color'           ,[1 0 0]  ,...
%   'LineWidth'       , 1.1         );
%
% %hTitle  = title ('Velocity Profile for different channel ...
% % inclination and magnetic field direction angles');
% hXLabel = xlabel('Velocity'                     );
% hYLabel = ylabel('Length - Y'                      );
%
% hLegend = legend( ...
%   [vel1, vel2, vel3, vel4, vel5, vel6, vel7, vel8, vel9], ...
%   'alfa = 0, \phi  = 0 ' , ...
%   'alfa = 3, \phi  = 0'      , ...
%   'alfa = 6, \phi  = 0'       , ...
%   'alfa = 0, \phi  = 0.03'    , ...
%   'alfa = 3, \phi  = 0.03'                 , ...
%   'alfa = 6, \phi  = 0.03' , ....
%   'alfa = 0, \phi  = 0.06'    , ...
%   'alfa = 3, \phi  = 0.06'                 , ...
%   'alfa = 6, \phi  = 0.06' , 2);
%
% set( gca                       , ...
%     'FontName'   , 'Helvetica' );
% set([hXLabel, hYLabel], ...
%     'FontName'   , 'AvantGarde');
% set([hLegend, gca]             , ...
%     'FontSize'   , 14          );
% set([hXLabel, hYLabel]  , ...
%     ’FontSize’   , 14          );
% % set( hTitle                    , ...
% %     ’FontSize’   , 18          , ...
% %     ’FontWeight’ , ’bold’      );
%
% set(gca, ...
%   ’Box’          , ’off’     , ...
%   ’TickDir’      , ’out’     , ...
%   ’TickLength’   , [.02 .02] , ...
%   ’XMinorTick’   , ’on’      , ...
%   ’YMinorTick’   , ’on’      , ...
%   ’YGrid’        , ’on’      , ...
%   ’XColor’       , [.3 .3 .3], ...
%   ’YColor’       , [.3 .3 .3], ...
%   ’LineWidth’    , 1         );
%
% % hText   = text(10, 800, ...
% %   sprintf(’\\it{C = %0.1g \\pm %0.1g (CI)}’, ...
% %   c, cint(2)-c));
% set(gcf, ’PaperPositionMode’, ’auto’);
% print -depsc2 velDist.eps
%
% fixPSlinestyle(’velDist.eps’, ’velDist2.eps’);
%
%
% % Graphes for temperature change
%
% figTemp = figure;
% hold on;
%
% temp1 = plot(temp.fi.alfa.data(:,1,1));
% set(temp1                            , ...
%   ’Marker’          , ’s’         , ...
%   ’Color’           , [0 0 .5]    ,...
%   ’LineWidth’       , 1         );
% temp2 = plot(temp.fi.alfa.data(:,1,2));
% set(temp2                            , ...
%   ’Marker’          , ’p’         , ...
%   ’Color’           , [0 0 .5]  ,...
%   ’LineWidth’       , 1         );
% temp3 = plot(temp.fi.alfa.data(:,1,3));
% set(temp3                            , ...
%   ’Marker’          , ’^’         , ...
%   ’Color’           , [0 0 .5]  ,...
%   ’LineWidth’       , 1         );
%
% temp4 = plot(temp.fi.alfa.data(:,2,1));
% set(temp4                            , ...
%   ’LineStyle’       ,’--’         , ...
%   ’Marker’          , ’s’         , ...
%   ’Color’           , [0 .5 0]   ,...
%   ’LineWidth’       , 1        );
% temp5 = plot(temp.fi.alfa.data(:,2,2));
% set(temp5                            , ...
%   'LineStyle'       ,'--'          , ...
%   'Marker'          , 'p'          , ...
%   'Color'           , [0 .5 0]   ,...
%   'LineWidth'       , 1         );
% temp6 = plot(temp.fi.alfa.data(:,2,3));
% set(temp6                            , ...
%   'LineStyle'       ,'--'          , ...
%   'Marker'          , '^'          , ...
%   'Color'           , [0 .5 0]  ,...
%   'LineWidth'       , 1         );
%
% temp7 = plot(temp.fi.alfa.data(:,3,1));
% set(temp7                            , ...
%   'LineStyle'       ,'-.'          , ...
%   'Marker'          , 's'          , ...
%   'Color'           , [1 0 0]   ,...
%   'LineWidth'       , 1        );
% temp8 = plot(temp.fi.alfa.data(:,3,2));
% set(temp8                            , ...
%   'LineStyle'       ,'-.'          , ...
%   'Marker'          , 'p'          , ...
%   'Color'           , [1 0 0]   ,...
%   'LineWidth'       , 1         );
% temp9 = plot(temp.fi.alfa.data(:,3,3));
% set(temp9                            , ...
%   'LineStyle'       ,'-.'          , ...
%   'Marker'          , '^'          , ...
%   'Color'           , [1 0 0]  ,...
%   'LineWidth'       , 1         );
%
% hLegend = legend( ...
%   [temp1, temp2, temp3, temp4, temp5, temp6, temp7, ...
% temp8,temp9], ...
%   'alfa = 0, \phi  = 0 ' , ...
%   'alfa = 3, \phi  = 0'      , ...
%   'alfa = 6, \phi  = 0'       , ...
%   'alfa = 0, \phi  = 0.03'    , ...
%   'alfa = 3, \phi  = 0.03'                , ...
%   'alfa = 6, \phi  = 0.03' , ...
%   'alfa = 0, \phi  = 0.06'    , ...
%   'alfa = 3, \phi  = 0.06'                , ...
%   'alfa = 6, \phi  = 0.06' , 4);
%
% % hTitle  = title ('Temperature Distribution for different ...
% % values of alfa and Volume Concentration');
% hXLabel = xlabel('Temperature'                     );
% hYLabel = ylabel('Length - Y'                      );
%
% set( gca                       , ...
%     'FontName'   , 'Helvetica' );
% set([hXLabel, hYLabel], ...
%     'FontName'   , 'AvantGarde');
% set([hLegend, gca]             , ...
%     'FontSize'   , 14           );
% set([hXLabel, hYLabel]  , ...
%     'FontSize'   , 14          );
% % set( hTitle                    , ...
% %     'FontSize'   , 14          , ...
% %     'FontWeight' , 'bold'      );
%
% set(gca, ...
%   'Box'          , 'off'     , ...
%   'TickDir'      , 'out'     , ...
%   'TickLength'   , [.02 .02] , ...
%   'XMinorTick'   , 'on'      , ...
%   'YMinorTick'   , 'on'      , ...
%   'YGrid'        , 'on'      , ...
%   'XColor'       , [.3 .3 .3], ...
%   'YColor'       , [.3 .3 .3], ...
%   'LineWidth'    , 1         );

% magnifyOnFigure(...
%         figTemp,...
%         'units', 'pixels',...
%         'magnifierShape', 'ellipse',...
%         'initialPositionSecondaryAxes', [120 220 155 180],...
%         'initialPositionMagnifier',     [280 200 45 45],...
%         'mode', 'interactive',...
%         'displayLinkStyle', 'straight',...
%         'edgeWidth', 1,...
%         'edgeColor', [0 0 0.5],...
%         'secondaryAxesFaceColor', [0.91 0.91 0.91]...
%             );
%
% set(gcf, 'PaperPositionMode', 'auto');
% print -depsc2 tempDist.eps
%
% fixPSlinestyle('tempDist.eps', 'tempDist2.eps');

% hText   = text(10, 800, ...
%   sprintf('\\it{C = %0.1g \\pm %0.1g (CI)}', ...
%   c, cint(2)-c));


%'LineStyle'       ,'-.'         , ...

%    figure(1);
%    plot(velocity,yValues)
%    legend(cellstr(num2str((0:0.5:1)')))
%    legend(cellstr(strcat('\psi  = ',num2str((0:0.01:0.06)'))))
%    legend('ha=%d',Ha)
%    Ha = % d \n',Ha
%    xlabel('Velocity')
%    ylabel('Y')
%    hold all
%    figure(2);
%    plot(temperature,yValues)
%    legend(cellstr(strcat('\psi  = ',num2str((0:0.01:0.06)'))))
%    xlabel('Temperature')
%    ylabel('Y')
%    hold all
toc;
```

```
 % graphes

% % Graphes for velocity change
%
figVel = figure;
hold on;
%
% alfa = 0:10:90;
% Y = 0 : 0.05 : 0.95

% vel1 = plot(alfa,Nmav(2,:));
% set(vel1                            , ...
%   'LineStyle'       ,'--'          , ...
%   'Marker'          , 's'          , ...
%   'Color'           , [0 0 .5]     ,...
%   'LineWidth'       , 1.1          );
% vel2 = plot(alfa,Nhav(2,:));
% set(vel2                            , ...
%   'LineStyle'       ,'-.'          , ...
%   'Marker'          , '+'          , ...
%   'Color'           , [0 0.5 .5]   ,...
%   'LineWidth'       , 1.1          );
% vel3 = plot(alfa,Nfav(2,:));
% set(vel3                            , ...
%   'Marker'          , '^'          , ...
%   'Color'           , [0.5 0 .5]   ,...
%   'LineWidth'       , 1.1          );

vel4 = plot(NG(:,2,8));
set(vel4                            , ...
  'Marker'          , 's'          , ...
  'Color'           , [0 .5 0]    ,...
  'LineWidth'       , 1.1          );
vel5 = plot(NG(:,3,8));
set(vel5                            , ...
  'Marker'          , '+'          , ...
  'Color'           , [0 .5 0]    ,...
  'LineWidth'       , 1.1          );
vel6 = plot(NG(:,4,8));
```

```
set(vel6                            , ...
  'Marker'            , '^'           , ...
  'Color'             , [0 .5 0]    ,...
  'LineWidth'         , 1.1           );
vel7 = plot(NG(:,5,8));
set(vel7                            , ...
  'Marker'            , '*'           , ...
  'Color'             , [0 0.5 0]     ,...
  'LineWidth'         , 1.1           );
vel8 = plot(NG(:,6,8));
set(vel8                            , ...
  'Marker'            , 'o'           , ...
  'Color'             , [0 0.5 0] ,...
  'LineWidth'         , 1.1           );
vel9 = plot(NG(:,7,8));
set(vel9                            , ...
  'Marker'            , 'p'           , ...
  'Color'             ,[0 0.5 0]  ,...
  'LineWidth'         , 1.1           );


%   'LineStyle'       ,':'          , ...

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% hTitle  = title ('Velocity Profile for different channel ...
% inclination and magnetic field direction angles');
% hXLabel = xlabel('\alpha'          );
% hYLabel = ylabel(' Total Entropy Generation  '       );

% hLegend = legend( ...
%   [vel1, vel2, vel3], ...
%   'N_m_a_g_n_e_t_i_c ' , ...
%   'N_h_e_a_t_t_r_a_n_s_f_e_r'      , ...
%   'N_f_l_u_i_d' , 1);


% hTitle  = title ('Velocity Profile for different channel
% inclination and magnetic field direction angles');
hXLabel = xlabel('Y'             );
hYLabel = ylabel(' Local Entropy Generation '         );

hLegend = legend( ...
  [vel4, vel5, vel6, vel7, vel8, vel9], ...
  '\phi = 10\circ, \alpha = 70\circ' , ...
  '\phi = 20\circ, \alpha = 70\circ' , ...
  '\phi = 30\circ, \alpha = 70\circ' , ...
  '\phi = 40\circ, \alpha = 70\circ', ...
  '\phi = 50\circ, \alpha = 70\circ', ...
  '\phi = 60\circ, \alpha = 70\circ', 2);

set( gca                       , ...
    'FontName'   , 'Helvetica' );
```

```
set([hXLabel, hYLabel], ...
    'FontName'   , 'AvantGarde');
set([hLegend, gca]             , ...
    'FontSize'   , 14           );
set([hXLabel, hYLabel]  , ...
    'FontSize'   , 14          );
% xlim([0 90])
% ylim([-1 1])
% set( hTitle                    , ...
%     'FontSize'   , 18          , ...
%     'FontWeight' , 'bold'      );

set(gca, ...
  'Box'         , 'off'     , ...
  'TickDir'     , 'out'     , ...
  'TickLength'  , [.02 .02] , ...
  'XMinorTick'  , 'on'      , ...
  'YMinorTick'  , 'on'      , ...
  'YGrid'       , 'on'      , ...
  'XColor'      , [.3 .3 .3], ...
  'YColor'      , [.3 .3 .3], ...
  'LineWidth'   , 1         );

% hText   = text(10, 800, ...
%   sprintf('\\it{C = %0.1g \\pm %0.1g (CI)}', ...
%   c, cint(2)-c));
set(gcf, 'PaperPositionMode', 'auto');
print -depsc2 localEntropyHa3_4.eps

fixPSlinestyle('localEntropyHa3_4.eps', ....
'localEntropyHa3_4.eps');
%
%
% % Graphes for temperature change
%
% figTemp = figure;
% hold on;
%
% temp1 = plot(temp.fi.alfa.data(:,1,1));
% set(temp1                            , ...
%   'Marker'          , 's'          , ...
%   'Color'           , [0 0 .5]     ,...
%   'LineWidth'       , 1          );
% temp2 = plot(temp.fi.alfa.data(:,1,2));
% set(temp2                            , ...
%   'Marker'          , 'p'          , ...
%   'Color'           , [0 0 .5]   ,...
%   'LineWidth'       , 1          );
% temp3 = plot(temp.fi.alfa.data(:,1,3));
% set(temp3                            , ...
%   'Marker'          , '^'          , ...
%   'Color'           , [0 0 .5]   ,...
%   'LineWidth'       , 1         );
%
% temp4 = plot(temp.fi.alfa.data(:,2,1));
% set(temp4                            , ...
%   'LineStyle'       ,'--'         , ...
%   'Marker'          , 's'         , ...
%   'Color'           , [0 .5 0]   ,...
%   'LineWidth'       , 1        );
% temp5 = plot(temp.fi.alfa.data(:,2,2));
% set(temp5                            , ...
%   'LineStyle'       ,'--'         , ...
%   'Marker'          , 'p'         , ...
%   'Color'           , [0 .5 0]   ,...
%   'LineWidth'       , 1         );
% temp6 = plot(temp.fi.alfa.data(:,2,3));
% set(temp6                            , ...
%   'LineStyle'       ,'--'         , ...
%   'Marker'          , '^'         , ...
%   'Color'           , [0 .5 0]  ,...
%   'LineWidth'       , 1         );
%
% temp7 = plot(temp.fi.alfa.data(:,3,1));
% set(temp7                            , ...
%   'LineStyle'       ,'-.'         , ...
%   'Marker'          , 's'         , ...
%   'Color'           , [1 0 0]   ,...
%   'LineWidth'       , 1        );
% temp8 = plot(temp.fi.alfa.data(:,3,2));
% set(temp8                            , ...
%   'LineStyle'       ,'-.'         , ...
%   'Marker'          , 'p'         , ...
%   'Color'           , [1 0 0]   ,...
%   'LineWidth'       , 1         );
% temp9 = plot(temp.fi.alfa.data(:,3,3));
% set(temp9                            , ...
%   'LineStyle'       ,'-.'         , ...
%   'Marker'          , '^'         , ...
%   'Color'           , [1 0 0]  ,...
%   'LineWidth'       , 1         );
%
%
% hLegend = legend( ...
%   [temp1, temp2, temp3, temp4, temp5, temp6, temp7,...
%  temp8,temp9], ...
%   'alfa = 0, \phi  = 0 ' , ...
%   'alfa = 3, \phi  = 0'       , ...
%   'alfa = 6, \phi  = 0'        , ...
%   'alfa = 0, \phi  = 0.03'     , ...
%   'alfa = 3, \phi  = 0.03'                , ...
%   'alfa = 6, \phi  = 0.03' , ...
%   'alfa = 0, \phi  = 0.06'     , ...
%   'alfa = 3, \phi  = 0.06'                , ...
%   ’alfa = 6, \phi  = 0.06’ , 4);
%
% % hTitle  = title (’Temperature Distribution for different
% values of alfa and Volume Concentration’);
% hXLabel = xlabel(’Temperature’                     );
% hYLabel = ylabel(’Length - Y’                      );
%
% set( gca                       , ...
%     ’FontName’   , ’Helvetica’ );
% set([hXLabel, hYLabel], ...
%     ’FontName’   , ’AvantGarde’);
% set([hLegend, gca]             , ...
%     ’FontSize’   , 14           );
% set([hXLabel, hYLabel]  , ...
%     ’FontSize’   , 14          );
% % set( hTitle                    , ...
% %     ’FontSize’   , 14          , ...
% %     ’FontWeight’ , ’bold’      );
%
% set(gca, ...
%   ’Box’          , ’off’     , ...
%   ’TickDir’      , ’out’     , ...
%   ’TickLength’   , [.02 .02] , ...
%   ’XMinorTick’   , ’on’      , ...
%   ’YMinorTick’   , ’on’      , ...
%   ’YGrid’        , ’on’      , ...
%   ’XColor’       , [.3 .3 .3], ...
%   ’YColor’       , [.3 .3 .3], ...
%   ’LineWidth’    , 1         );
%
% magnifyOnFigure(...
%         figTemp,...
%         ’units’, ’pixels’,...
%         ’magnifierShape’, ’ellipse’,...
%         ’initialPositionSecondaryAxes’, [120 220 155 180],...
%         ’initialPositionMagnifier’,     [280 200 45 45],...
%         ’mode’, ’interactive’,...
%         ’displayLinkStyle’, ’straight’,...
%         ’edgeWidth’, 1,...
%         ’edgeColor’, [0 0 0.5],...
%         ’secondaryAxesFaceColor’, [0.91 0.91 0.91]...
%             );
%
% set(gcf, ’PaperPositionMode’, ’auto’);
% print -depsc2 tempDist.eps
%
% fixPSlinestyle(’tempDist.eps’, ’tempDist2.eps’);

% hText    = text(10, 800, ...
%   sprintf(’\\it{C = %0.1g \\pm %0.1g (CI)}’, ...
%   c, cint(2)-c));
```

```
%’LineStyle’       ,’-.’         , ...

%    figure(1);
%    plot(velocity,yValues)
%    legend(cellstr(num2str((0:0.5:1)’)))
%    legend(cellstr(strcat(’\psi  = ’,num2str((0:0.01:0.06)’))))
%    legend(’ha=%d’,Ha)
%    Ha = % d \n’,Ha
%    xlabel(’Velocity’)
%    ylabel(’Y’)
%    hold all
%    figure(2);
%    plot(temperature,yValues)
%    legend(cellstr(strcat(’\psi  = ’,num2str((0:0.01:0.06)’))))
%    xlabel(’Temperature’)
%    ylabel(’Y’)
%    hold all
```

```
[xx,yy] = meshgrid(0:10:60,0:1:19);

% zz1(:,:) = temp.data(:,1,:);
% zz2(:,:) = temp.data(:,3,:);
% zz3(:,:) = temp.data(:,6,:);
% zz4(:,:) = temp.data(:,11,:);

subplot(2,2,1)
surf(xx,yy,vel.fi.alfa.data(:,:,1))
% axis([0 inf 0 inf 0 0.2])
title(’ \alpha = 0\circ ’, ’FontSize’, 14)
xlabel(’\Phi’,’FontSize’,13)
ylabel(’Y’, ’FontSize’,13)
zlabel(’U(Y)’, ’FontSize’,13)
caxis([0 1.])
colorbar;
xlim([0 60])
zlim([0 1.7])

subplot(2,2,2)
surf(xx,yy,vel.fi.alfa.data(:,:,4))
% axis([0 inf 0 inf 0 0.2])
% title(’V* = 0.2’)
title(’ \alpha = 30\circ ’, ’FontSize’, 14)
xlabel(’\Phi’,’FontSize’,13)
ylabel(’Y’, ’FontSize’,13)
zlabel(’U(Y)’, ’FontSize’,13)
% zlabel(’Total Ent. Gen. Viscous Dissipation Term’)
caxis([0 1.])
xlim([0 60])
```

```
zlim([0 1.7])
colorbar;
% xlim([0 1])

subplot(2,2,3)
surf(xx,yy,vel.fi.alfa.data(:,:,7))
% axis([0 inf 0 inf 0 0.2])
% title(’V* = 0.5’)
title(’ \alpha = 60\circ ’, ’FontSize’, 14)
xlabel(’\Phi’,’FontSize’,13)
ylabel(’Y’, ’FontSize’,13)
zlabel(’U(Y)’, ’FontSize’,13)
% zlabel(’Total Ent. Gen. Magnetic Field Term’)
caxis([0 1.])
xlim([0 60])
zlim([0 1.7])
colorbar;
% xlim([0 1])

subplot(2,2,4)
surf(xx,yy,vel.fi.alfa.data(:,:,10))
% axis([0 inf 0 inf 0 0.2])
% title(’V* = 1’)
title(’ \alpha = 90\circ ’, ’FontSize’, 14)
xlabel(’\Phi’,’FontSize’,13)
ylabel(’Y’, ’FontSize’,13)
zlabel(’U(Y)’, ’FontSize’,13)
% zlabel(’Total Ent. Gen. Heat Transfer Term’)
caxis([0 1.])
xlim([0 60])
zlim([0 1.7])
colorbar;
% xlim([0 1])

set(gcf, ’PaperPositionMode’, ’auto’);

print -depsc2 ’-r300’ vel_3D.eps

fixPSlinestyle(’vel_3D.eps’, ’vel_3Dmodif.eps’);
```

CURRICULUM VITAE

Name Surname: Elgiz Başkaya

Place and Date of Birth: Istanbul, 31.01.1984

E-Mail: benelgiz@gmail.com

EDUCATION:

- **B.Sc.:** 2007, Istanbul Technical University, Faculty of Aeronautics and Astronautics, Aerospace Engineering
- **M.Sc.:** 2010, Istanbul Technical University, Faculty of Aeronautics and Astronautics, Interdisciplinary Aeronautical and Astronautical Sciences

PROFESSIONAL EXPERIENCE AND REWARDS:

- 2015 - Present Researcher - The ENGIE Ineo - Grope ADP - SAFRAN RPAS Chair
- 2009 - 2015 Research & Teaching Assistant - Faculty of Aeronautics & Astronautics
- 2014 - 2015 Researcher - TUBITAK 113E595
- 2010 - 2013 Researcher - Control and Avionics Laboratory
- 2008 - 2010 Researcher - Space Systems Laboratory

PUBLICATIONS, PRESENTATIONS AND PATENTS ON THE THESIS:

- **Baskaya, E., Komurgoz, G., Ozkol, I.** Investigation of Oriented Magnetic Field Effects on Entropy Generation in an Inclined Channel Filled with Ferrofluids. *Entropy*, 2017, 19.7: 377.
- **Baskaya, E., Komurgoz, G., Ozkol, I.** Entropy Generation and Equipartition Phenomenon Investigation in a Variable Viscosity Channel Flow under Constant Magnetic Field via Generalized Differential Quadrature Method (GDQM) , *Heat Transfer Research,* (submitted)
- **Baskaya, E., Komurgoz, G., Ozkol, I.** Analysis of Variable Viscosity Channel Flow under Constant Magnetic Field via Generalized Differential Quadrature Method, *Advanced Materials Research,* vol. 1016, pp.564-568, 2014 (from conference paper below)

- **Baskaya, E., Komurgoz, G., Ozkol, I.** Analysis of a Variable Viscosity Channel Flow under Constant Magnetic Field via Generalized Differential Quadrature Method *ICMAE 5th International Conference on Mechanical and Aerospace Engineering*, July 18 - 19, 2014

- **Baskaya, E., Fidanoglu, M., et at.** Investigation of MHD Natural Convection Flow Exposed to a Variable Magnetic Field via Differential Quadrature Method *ASME 12th Biennial Conference on Engineering Systems Design and Analysis*, June 24 - 27, 2014

- **Fidanoglu, M., Baskaya, E., et at.** Application of Differential Quadrature Method and Evolutionary Algorithm to MHD Fully Developed Flow of a Couple-Stress Fluid in a Vertical Channel With Viscous Dissipation and Oscillating Wall Temperature *ASME 12th Biennial Conference on Engineering Systems Design and Analysis*, June 24 - 27, 2014

OTHER PUBLICATIONS, PRESENTATIONS AND PATENTS:

- **Baskaya, E., Bronz, M., Delahaye, D.** "Fault Detection & Diagnosis for Small UAVs via Machine Learning " 36th *IEEE/AIAA Digital Avionics Systems Conference (DASC),* September 17 - 21, 2017

- **Baskaya, E., Bronz, M., Delahaye, D.** "Flight Simulation of a MAKO UAV for Use in Data-Driven Fault Diagnosis " 9th *International Micro Air Vehicles,* September 18 - 21, 2017

- **Baskaya, E., Manfredi, G., Bronz, M., Delahaye, D.** "Flexible open architecture for UASs integration into the airspace: Paparazzi autopilot system" 35th *IEEE/AIAA Digital Avionics Systems Conference (DASC),* September 25 - 29, 2016

- **Baskaya, E., Daybelge, U., Sofyali, A., Topal, E., Yarim, C.** "Developments in Astrodynamics in the Light of Chaos " *Journal of Istanbul Kultur University,* vol.4, pp.191-212, 2006

- **Baskaya, E., Inalhan, G., et al.** "Design and Development of a Reliable ADCS and Indigenous Bus Architecture for Nanosatellites : ITUpSAT II" 63rd *International Astronautical Congress*, October 1- 5, 2012

- **Baskaya, E., Eren, U., Inalhan G.** "Development of High - Precision Attitude Determination and Control System : ITUpSAT II project " *National Aeronautical and Astronautical Conference*, September 12 - 14, 2012

- **Baskaya, E., Koyuncu, E., Inalhan, G.** "Design of a Multi - Purpose Nanosatellite Bus : ITUpSAT II project " *National Aeronautical and Astronautical Conference*, September 12 - 14, 2012

- **Baskaya, E., Eren, U., et al.** "A Precise ADCS Design for ITUpSAT II" *International Conference on Student Small Satellites*, April 25 - 27, 2012

- **Eren, U., Baskaya, E., et al.** "Design of a Flexible Bus System for ITUpSAT II" *International Conference on Student Small Satellites*, April 25 - 27, 2012

- **Koyuncu, E., Baskaya, E., et al.** “ITUpSAT II : High - Precision Nanosatellite ADCS Development Project” 5th *International Conference on Recent Advances in Space Technologies*, June 9 - 11, 2011

- **Inalhan, G., Koyuncu, E., Baskaya, E., et al.** “Design and Development of ITUpSAT II : On Orbit Demonstration of a High - Precision ADCS for Nanosatellites ” 8th *International ESA Conference on Guidance & Navigation Control Systems*, June 5 - 10, 2011

- **Baskaya, E., Hajiyev, C., Inalhan, G.,** “Estimation of Small Satellite Attitude Dynamics via EKF using Magnetometers - ITUpSAT II Project” *National Aeronautical and Astronautical Conference*, September 16 - 18, 2010

- **Baskaya, E.** “ITUpSAT II ADCS : Getting Ready for Launch” 8th *Annual CubeSat Developers' Workshop*, April 20 - 22, 2011

- **Baskaya, E., Inalhan, G.** “ITUpSAT II - Nanosatellite Platform for In-Space R&D” 7th *Annual CubeSat Developers' Workshop*, April 21 - 23, 2010

- **Eren, U., Baskaya, E., et al.** “Design of a Flexible Nanosatellite Bus for Science Missions ” *AIAA SPACE 2012 Conference & Exposition*, September 11- 13, 2012

- **Baskaya, E., Eren, U., et al.** “ITUpSAT II - Design and Development ” *Innovation Week Turkey*, December 6- 8, 2012