Errata

Equation (4.16) on page 77 and the Equation in row 4 of Table 4.2 on page 78 should read:

$$\int_{{}^0V} {}_0\mathbf{C}_{ijrs}\ {}_0\mathbf{e}_{rs}\delta {}_0\mathbf{e}_{ij}\,\mathbf{dV}^0 + \int_{{}^0V} {}_0^t\mathbf{S}_{ij}\,\delta\ {}_0\mathbf{q}_{ij}\,\mathbf{dV}^0 = {}^{t+\Delta t}\mathcal{R} - \int_{{}^0V} {}_0^t\mathbf{S}_{ij}\,\delta\ {}_0\mathbf{e}_{ij}\,\mathbf{dV}^0$$

Equation 4.18 on page 79 should read:

$$\int_{{}^0V} {}_0\mathbf{C}_{ijrs}^{(l-1)}\,\Delta\ {}_0\mathbf{e}_{rs}^{(l)}\,\delta {}_0\mathbf{e}_{ij}\mathbf{dV}^0 + \int_{{}^0V} {}_0^{t+\Delta t}\mathbf{S}_{ij}^{(l-1)}\,\delta\Delta\ {}_0\mathbf{q}_{ij}^{l}\,\mathbf{dV}^0 = {}^{t+\Delta t}\mathcal{R} - \int_{{}^0V} {}_0^{t+\Delta t}\mathbf{S}_{ij}^{(l-1)}\,\delta\ {}_0^{t+\Delta t}\mathbf{e}_{ij}^{(l-1)}\,\mathbf{dV}^0$$

PENINSULA  TECHNIKON

The Development of a Computational Design Tool for use in the design of SMA Actuator Systems

by

Oscar Philander

Under the supervision of

Prof. Dr. G. J. Oliver

and

Prof. J. Gryzagorides

Thesis submitted in fulfillment of the requirements of the Doctor of Technology: Mechanical Engineering (DTECHME) in the Faculty of Engineering at the Peninsula Technikon.

Date of submission: November 2004

Declaration of Originality

The work described in this thesis was carried out in the Faculty of Engineering, Department of Mechanical Engineering at the Peninsula Technikon. Except where otherwise acknowledged, the work presented for this degree is my own and contains no material which has been presented for a degree at this or any other university and, to the best of my knowledge and belief, contains no copy or paraphrase of work published by another person, except where duly acknowledged in the text.

Significant contributions are as follows:

Acknowledgements is given to Prof. Dr. I. Müller and his team at the Technical University of Berlin for the useful discussions pertaining to the thermodynamic and statistical thermodynamic model that was formulated by him and his co-workers. A discussion of this model is presented in Chapter 3.

Mr. R. April and Mr. B. Deez carried out the experimental data collection for the NiTi Shape Memory Alloy wire experiments shown in Chapter 2. For the discussions on the experimental data I would like to acknowledge the assistance of Prof. J. Gryzagorides of The University of Cape Town.

Acknowledgements also go to Prof. Dr. G. J. Oliver for discussions on the Finite Element Method and the implementation of Finite Element Formulation discussed in Chapter 5 into a software program.

Oscar Philander (MTechME, PENTECH)
Cape Town, South Africa
November 2004

Acknowledgements

Firstly I would like to thank God for giving me the gifts that helped me complete this study. The long days and late nights spent working on the theories presented in this thesis brought me closer to You.

I would like to thank Prof. Dr. B. Sun for initiating the Smart Devices and MEMS research group at the Faculty of Engineering, Department of Mechanical Engineering, at the Peninsula Technikon. He provided a high level of sophistication and motivated me to complete this study. He also played a key role in nominating me to study at the prestigious University of Michigan in the United States of America. It was during this student exchange program that I obtained a better insight into the theories of Continuum Mechanics and the Finite Element Method.

The financial assistance given to me by the Peninsula Technikon and the National Research Foundation (NRF) through the course of my post-graduate studies is greatly appreciated. These institutions made it possible for me to study at the Technical University of Berlin under the supervision of the well-known Shape Memory Alloy researcher, Prof. Dr. Dr. Ingo Müller. This is where I gained the knowledge of the physics of phase transformations observed in shape memory alloys.

Thanks to Prof. A. Staak and Mr. K. Jacobs for affording me the opportunity work while completing this research. This provided the financial stability that I needed to complete this study.

To the late Prof. H. Fraansman who always inspired me.

A special word of thanks to my supervisors, Prof. Dr. G. J. Oliver and Prof. J. Grizagorides. This study could never have been completed without your assistance. Your time and effort is really appreciated and I hope that we will continue to work as closely on future projects.

Prof. Oliver's knowledge of Thermodynamic Continuum Mechanics, the Finite Element Method and his extensive knowledge of programming languages coupled with his excellence has always been an inspiration to me and I hope that I will also adopt this kind of philosophy with my students. Prof. G is an experimentalist of high repute and applies his knowledge with great versatility.

To Mr. I Omar who always pointed me in the right direction especially when the light at the end of the tunnel was not so bright.

To my colleagues in the Department of Mechanical Engineering especially Ms. R. Ziegler, Mr. M. N. Riddles and Mr. P. Simelane for your undying support and belief in me.

A special word of thanks to my parents, David and Maureen Philander. Your love and support is paramount to my academic and personal development. This work also represents the fruits of your labor.

To Leslie and Jennifer Samuels who believed in me and always had words of encouragement when I needed inspiration.

Finally, I would like to thank my wife Nicky. You have supported me through the ups and downs of this study and always managed to keep me grounded. You always showed me that there are other things in this world that can be just as interesting as my research. Thank you for the time that you unselfishly gave to me in order to complete this research. Your love has brought us to a new chapter in our lives.

"...Bear in mind that the wonderful things you learn in schools are the work of many generations, produced by enthusiastic effort and infinite labor in every country of the world. All this is put into your hands as your inheritance in order that you may receive it, honor it, add to it, and one day faithfully hand it to your children. Thus do we mortals achieve immortality in the permanent things which we create in common?..."

Albert Einstein

Dedication

To my children:

"...I hope that the text presented in this thesis will inspire you to great heights and that you will one day exceed expectations..."

Abstract

Engineers and Technologists have always been identified as those individuals that put into practice the theories developed by scientists and physicists to enhance the lives of human beings. In the same spirit as those that came before, this thesis describes the development of a computational engineering tool that will aid Engineers and Technologists to design smart or intelligent structures comprising of NiTi shape memory alloy rods for actuation purposes.

The design of smart actuators consisting of NiTi shape memory alloy structural members will be beneficial to industries where light weight, compactness, reliability and failure tolerance is of utmost importance. This is mainly due to the unique material responses exhibited by this smart material. The shape memory effect, one of these material responses consists out of two stages: a low temperature load induced phase transformation causing a macroscopic deformation (either extension, contraction, etc.) also known as quasi-plasticity; and a high temperature phase transformation that erases the low temperature macroscopic deformation and reverts the material to some predefined geometry. When designing actuators consisting of this smart material, the quasi-plastic material response produces the actuation stroke while the high temperature phase transformation produces the actuation force.

The successful engineering design of smart structures and devices particularly suited for applications where they operate in a capacity, as actuators harnessing the shape memory effect are dependent on a few important factors. These include the engineers familiarity with the type of smart material used, the availability of sound experimental data pertaining to the complex material responses exhibited by the smart material, the engineers level of proficiency with existing constitutive models available to simulates these material responses, and the engineers knowledge of simulation tools consisting of a suitable control algorithm for the modeling of not only the device or structure itself but also the actuator involved in the design.

This thesis presents the successful development of a computational engineering design tool specifically suited for the design of actuators consisting of NiTi shape memory alloys harnessing the shape memory effect. The thermo-mechanical model developed for this application is based on a Helmholtz free energy function for this material coupled with phase transformation evolution laws. The design tool is capable of simulating the full spectrum of required data for this type of actuator design. This data includes actuation stroke, actuation force and low and high temperature phase transformations associated with the shape memory effect.

The computational tool combines a novel but powerful numerical thermodynamic and statistical thermodynamic model for simulating the unique phase transformation kinetics observed in shape memory alloys with a finite element method based on the Total Lagrangian formulation for non-linear material responses. This engineering design tool is validated by experimental data obtained from experiments performed to study the uniaxial quasi-plastic behavior of NiTi shape memory alloys subjected to a variety of displacement rates.

The experiments were conducted in the Faculty of Engineering, Department of Mechanical Engineering at the Peninsula Technikon. 1, 2, and 3mm diameter

NiTi shape memory alloy wires with lengths of 100, 150, 200, and 250mm were tested. The specimens were subjected to displacement rates of 2, 3, 4, 5, 7, 10, and 15mm/min. The results obtained from these experiments showed the effect of material geometric properties and different displacement rates on the quasi-plastic material response observed in this material. The results are consistent with that obtained by other researchers in the field and also provides interesting discussion points.

The thesis describes the use of the dynamic one-dimensional thermodynamic and statistical thermodynamic constitutive model proposed by Müller and Achenbach and further refined by Müller and Seelecke in the simulation of shape memory alloy line actuators. This model permits the simulation of the response of a tensile specimen to a thermodynamic input and calculates all phase transformations, phase proportions and deformations as functions of time if the temperature and applied load are prescribed as functions of time.

Certain aspects of this model are used in the formulation of a Shape Memory Alloy Truss (SMAT) finite element capable of experiencing non-linear material responses. A comparison of the SMAT finite element and the numerical model proposed by Müller and his co-workers to experimental data showed that the SMAT finite element provided a better fit to the experimental data than the numerical model. This is ascribed to the non-linear nature of the SMAT finite element. A software program was then developed using the software language C, to simulate the behavior of more complex NiTi shape memory alloy structural members for use in actuator systems.

Contents

List of Figures

Chapter 1

Chapter 2

Chapter 4

Chapter 5

List of Tables

Chapter 1

Introduction

1.1 Problem Statement

Engineers and Technologist are called upon to develop a new generation of "smart" products that may be used in military, aerospace, medical, automotive, and commercial products for the twenty first century. The development of these products has previously been accomplished through the advancement of smart materials and intelligent structures with inherent "brains", "nervous systems", and "skeletons". They were realised through the integration of knowledge bases associated with material science, information technology and biotechnology. Smart materials and structures have been found to be attractive for applications such as active vibration and acoustic transmission control, active shape control, and active damage control [43].

The key methodology behind the design of smart structures and devices particularly suited for applications where they operate in the capacity of actuators, sensors and actuator-sensor environments lies in drastically reducing the weight of these actuator devices by use of shape memory alloy wires as the actuating devices. These smart devices and structures are strong, lightweight and compact compared to conventional linear actuators.

Conventional linear actuators are used in a variety of applications and particularly in space flight application where shape, size and weight of the linear actuator is an important factor typically to produce fine linear adjustments (e.g. optical mounts), small angle tilting motions (e.g. "tilt table" gimbals; thrust vector control), to operate lever-crank mechanisms (e.g. scanners; positioners), to operate mechanisms (e.g. launch locks; movable covers), and generally when ever linear motion is needed.

Linear motion may be required either directly as an output motion, or as input to other mechanisms. The simplest device delivering linear output is the screw-but pair. This device integrates easily with a motor and thus forms an overall compact and symmetrical unit.

The successful engineering design of smart structures and devices particularly suited for applications where they operate in the capacity of actuators, sensors and actuator-sensor environments are dependent on a few important factors. These include the engineers familiarity with the type of smart material used, the availability of sound experimental data pertaining to the complex material responses exhibited by the smart material, the engineers level of proficiency with existing constitutive models available to simulates these material responses, and the engineers knowledge of simulation tools consisting of a suitable control algorithm for the modelling of not only the device or structure itself but also the actuator involved in the design.

1.2 Objectives

This thesis is aimed at developing an engineering tool that can be used for the design of smart or intelligent actuators comprising of shape memory alloy wires. An experimental database pertaining to one aspect of the shape memory effect, the quasi-plastic material response is developed to serve as an engineering design aid but also to serve as a verification tool for constitutive modelling of this material response.

The computational framework of this tool will consist of a calibrated constitutive model (a modified version of the Müller-Achenbach model [32], [39]-[42]) to simulate the complex material responses exhibited by shape memory alloys, which will be incorporated into a finite element formulation with a control algorithm to study multi-dimensional uniaxial shape memory alloy line actuators as structural members in dynamic mechanical systems. It is envisaged that this computational tool will enhance the design of shape memory line actuators by providing necessary information for actuator heating and cooling, actuator force and stroke, and actuator efficiency.

1.3 Background

In a contribution by Shakeri et al. [43] the authors elaborate on clear criteria for identifying a material to be termed "smart" in the following different ways:

1. Materials functioning in a capacity of both sensing and actuation;
2. Materials which have multiple responses to one stimulus in a coordinated fashion;
3. Passively smart materials with self-repairing or stand-by characteristics to withstand sudden changes;
4. Actively smart materials utilizing feedback; and
5. Smart materials reproducing biological functions in load bearing structures.

Shape memory alloys fall within the realm of being termed "smart" in that they belong to a special group of metallic materials that remembers their shape even after severe deformation [46], [54]. When a load is applied to the material at low temperatures, it will deform causing an apparent plastic deformation, which is maintained until an increase in temperature causes the deformation to disappear completely. 4-8% erasable deformations have been observed in alloys like NiTi, CuZnAl, CuAlNi, and AuCd. This material response is termed the Shape Memory Effect [19], [31], [24], [45]-[48].

At high temperatures however, the material exhibits a material response termed the pseudo-elastic behaviour, and plastic deformations are recovered by releasing the applied load [19], [46]. This high temperature material response is also characterised by significant internal damping, non-linear elastic ranges, non-homogeneous deformations and high yield stresses [46], [47].

These unique material responses, i.e., the shape memory effect, and the pseudo-elastic effect are ascribed to shear deformations associated with a thermo-elastic martensitic transformation and its reverse transformation [8], [30], [52].

In a typical transformation during cooling, the high-temperature austenitic phase structure, which has a greater crystallographic symmetry than the low-temperature martensitic phase gives rise to the formation of multiple symmetry-related variants or proportions of twinned martensite when a critical temperature, the martensitic start temperature, M_s, is reached. Up to 24 different variants of martensite may result from this transformation, which is completed when the martensitic finish temperature, M_f, is reached [1]. This transformation however, does not cause the formation of locked in stresses as observed in classical steels, and hence no change in the geometric dimensions of the material occurs. These twinned variants of martensite form complex patterns at a length-scale much smaller than the size of the specimen. The actual length-scale can range from a few nanometers to tenths of millimeters and it depends on a variety of factors including the chemical composition, specimen size, grain size and heating and/or cooling history [6], [8], [23], [38].

When a load is applied to the material, while in the twinned martensitic state (at low temperature), only those variants of martensite sensitive to the applied load will transform through a process of de-twinning the twinned variants of martensite and cause an apparent macroscopic plastic deformation. An increase in temperature will transform this de-twinned martensitic crystal structure back to austenite and the material regains its original dimensions.

This process is termed the shape memory effect. The critical temperature at which the de-twinned martensite transformed to austenite is the austenitic start temperature, A_s, and the temperature at which all the de-twinned martensite fully transforms to austenite is termed the austenitic finish temperature, A_f [1], [5], [6], [23], [38].

If a specimen is heated to a temperature between the M_s and A_f temperatures it may contain variants of twinned martensite, austenite or a mixture of the two. If a load is now applied to the material, it transforms to de-twinned martensite and produces an apparent plastic deformation as in the shape memory effect. Unloading results in either an elastic response or a partial reverse transformation to austenite that follows a hysteretic path, depending on the temperature being either above or below the A_s temperature. The latter material response is also termed the partial pseudo-elastic effect. The remaining apparent plastic deformation can completely be recovered by heating above the A_f temperature. [5], [6], [23], [38].

When a load is applied to the high temperature austenitic phase (above the A_f temperature), it will first perform elastically until a certain transformation or yielding value of stress is reached. The austenite will now transform to de-twinned variants of martensite and again induce an apparent macroscopic plastic deformation. This deformation is maintained until subsequent unloading forces the material to now follow a hysteretic unloading path and all deformation is removed. This process describes the pseudo-elastic behavior of the material. Fig. 1.1. shows the schematic representation of the three material responses [46]-[48].

The reverse transformations of the shape memory effect (upon heating) and the energy dissipation due to the hysteretic partial pseudo-elastic and pseudo-elastic behaviours produce actuation forces that could be harnessed for various actuator applications in mechanical systems. This has been the focus of researchers and inventors for the past forty years [54]. For practical applications, shape memory alloys are usually used in NiTi alloy form [9], [36].

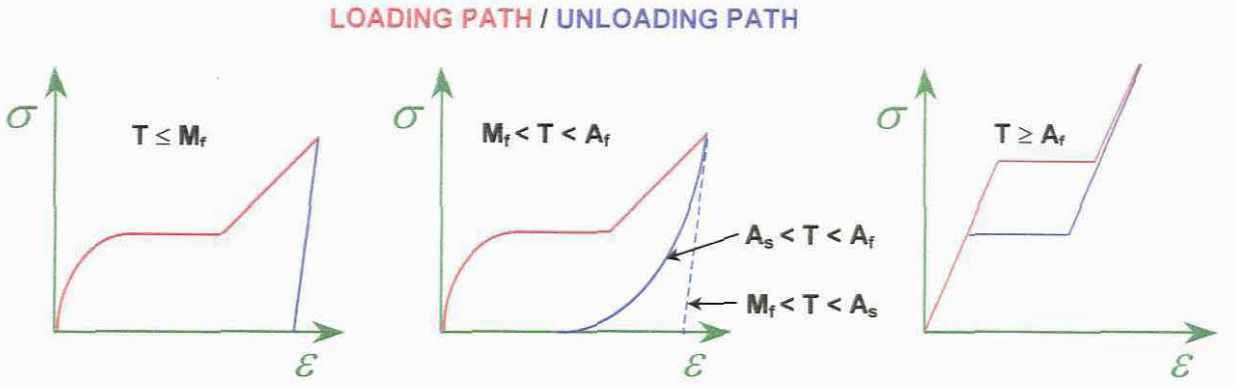


Figure 1.1: Schematic representation of (a) the shape memory effect, (b) the partial pseudo-elastic effect, and (c) the pseudo-elastic effect. (Observation in [9], [38], [45] and [46])

Beuler et al., while commissioned to develop a material for the nose cone of the US Navy missile, SUBROC, discovered this material in 1961. The goal of this work was to find a material that had a high melting point and high impact resistant properties and NiTi alloy best suited these criteria. It was only later through an accident that Beuler stumbled upon the remarkable shape memory characteristics of this material [25].

Since then NiTi alloy wires have be used in many novel applications ranging from shower springs that are activated above a certain temperature to shut off water that is too hot to the active shape control used in the adaptive aerodynamic lifting surface of a variable camber wing both utilizing the shape memory effect. Energy dissipation in pseudo-elastic hysteresis has successfully been used in vibration damping of mechanical systems [2], [9], [44], [54], [55].

The successful design of smart or intelligent structures with either embedded shape memory alloy fibres or structural shape memory alloy components relies on a good constitutive description for this material behaviour [36], [46]. Literature shows us that this is indeed an enormous task in that the complicated coupling between mechanical and thermal properties and many aspects of their interactions as it pertains to the phase transformations

occurring in this material has to be taken into account. This is evidenced by the sensitivity of the transformation temperature and other properties by relatively small changes in the composition of the alloy by various thermo-mechanical treatments and the effect of rate dependencies, which includes thermal and/or mechanical responses on transformation processes [17], [31] [37], [44], [46], [57].

Research related to the constitutive modelling of shape memory alloys have focussed on phenomenological and micro-mechanics approaches. Within these approaches their exist sub-groups such as Sate Space Phenomenological [26], [42], Two- and Three- Component Phenomenological [57], Crystallographic Micro-Mechanics, Free Energy Function Micro-Mechanics [26], [36] and Micro-Plane Models [9]. Constitutive models following the state space phenomenological approach, has been the engineers choice since they are conceptually easy to implement and use measurable quantities as parameters in their construction. Most of these models were constructed to fit one-dimensional experimental data and thus predict the uniaxial mechanical behaviour of these materials quite well [26], [42]. They do however lack the thermodynamic aspects related to the shape memory alloy material response and their suitability for the simulation in actuator applications is very limited [42]. Micro-mechanics models on the other hand follow the crystallographic behaviour of these materials quite well and although they possess a very high level of sophistication, they are far too complex for use in the design of one-dimensional shape memory actuators [42].

Other constitutive models based on the Maxwellian Visco-Elastic approach [13], and the Rubber-Like approach [33], are developed from experimental observations and an in depth knowledge of the mechanics of solid structures. These models however, only show good qualitative agreement with experimental data [13], [33] and work still has to be conducted to bring them to a quantitative level.

This thesis focuses on the use of a constitutive description for shape memory alloys based on the Multi-Well approach [17], [32], [20], [21], [39]-[42]. This approach not only incorporates the complex thermodynamic and mechanical aspects of the shape memory alloy material behaviour, but also lends itself to integration into robust numerical algorithms within a computational setting [17], [32], [39]-[42]. While some of the other constitutive models require the need for numerous material constants (in one case up to eighteen [8]) for their successful implementation to simulate shape memory alloy material response, this model only requires two, i.e., the Elastic Modulus of each metallurgical phase, i.e., low temperature martensite and high temperature austenite [32], [39]-[42].

These factors make this constitutive model a prime candidate for use in the design of complex multi-dimensional structural engineering systems that uses one-dimensional shape memory alloy actuators for active and/or passive vibration control or active shape control applications. We will therefore not try to improve on any of the existing constitutive models, but rather employ the selected model to suit our experimental data through calibration and use it in the design of our smart shape memory alloy line actuators.

A literature review on constitutive models that has been developed over the past forty years will show the validity of the Multi-Well constitutive approach in this regard. Secondly, experimental analyses were performed to study the effects of displacement rates on the martensitic transformation that occurs during the quasi-plastic deformation of the shape memory effect. The experiments were performed at the Department of Mechanical Engineering of the Peninsula Campus of the Cape University of Technology. Thirdly, a full description of the chosen constitutive model will follow with a full numerical implementation that provides qualitative and quantitative agreement with the experimental results. This constitutive model is then used in a Finite Element Formulation to further study complex multi-dimensional structural engineering systems.

1.4 Related Literature

1.4.1 Review of Shape Memory Actuated Devices

Shape memory alloys are the only crystalline solids that exhibit the shape memory effect at low temperatures and the pseudo-elastic effect at high temperatures [8], [30], [52]. These two remarkable material phenomena are ascribed to the shear deformation associated with the thermo-elastic reversible martensitic transformations of the material (Delay et al., Perkins, Schetsky) [52]. Bhattacharya [6] gives an excellent crystallographic and numerical interpretation on the theory of martensitc microstructure, its transformations and its particular manifestations as it pertains to the shape memory effect and the reader is prompted to study this contribution for a clearer understanding of this transformation mechanism. While in its low temperature state, shape memory alloys, exist as different variants of martensite (up to 24) called martensitic twinned variants or the product phase (see Fig. 1.2a. [22]).

The material will remain in this state until acted upon by an external stimulus that will either transform the material's crystallographic structure to austenite (see Fig. 1.2b. [22]) or the parent phase through a temperature induced transformation (heating) that causes no change in the material's macroscopic appearance, or a de-twinned martensite through a stress-induced transformation at low temperature that causes only those twinned martensite variants that are sensitive to the applied stress to reorientate and induce a change in the macroscopic dimensions (an apparent plastic deformation) of the material. The apparent plastic deformation that occurs due to the stress-induced transformation will remain until an increase in temperature transforms this de-twinned martensite back to austenite (see Fig. 1.3.).

Upon cooling from the austenite region the material will once again transform to the twinned variants of martensite with no locked-in or residual stresses. This cycle is termed the shape memory effect (see Fig. 3). The transformation from de-twinned martensite to austenite produces an actuation force that can

be used in various engineering systems. The magnitude of this force is dependant on a number of factors that may include the chemical composition of the material, the dimensions of the material, the degree to which the twinned martensite was transformed, the type of loading applied (tension, compression, torsion, etc.), and the type of heating (electrical or ambient).

At high temperatures, when the material is in its austenite state (see Fig. 1.2b. [22]), and subjected to mechanical loading, it will first perform elastically as many crystalline solids do. The elasticity of the material will remain until a yielding or transformation stress is reached. The austenite will now transform to de-twinned martensite and induce a macroscopic change in the material's dimensions. Macroscopically this change is similar to the hardening phenomenon of classical engineering steels. Upon subsequent unloading from this region however, the material follows a hysteretic unloading path. During this process all the de-twinned martensite transforms back to austenite and the material regains its original dimensions. This process of hysteretic loading and unloading is termed the pseudo-elastic effect with inherent energy dissipation.

The shape memory effect has found applications in the development of smart or active composites (Duerig and Melton, Duerig et al., Escher and Hornbogen, Lagoudas and Tadjbakhsh, Taya et al., Furuya et al., Boyd and Lagoudas, Lagoudas et al., [8], [17], [45], [46]). Here the pre-stressed shape memory alloy (in its de-twinned martensite state) is surrounded or embedded in a non-shape memory alloy matrix material. It was found that this process enhanced the overall yield and hardening characteristics of the composite at elevated temperatures [26]. Liang and Rogers reported on the use of these active metal matrix composites in helicopter rotor blades [36]. Chaudry and Rogers and Hughes and Wen [55] used shape memory alloys in active shape control applications to be used in mechanisms such as the adaptive aerodynamic lifting surface or variable camber wing.

Pruski and Kihl [55] used micro-miniature shape memory alloy actuators in robotic systems where strain measurements from the actuator were used for

position feedback. Giurgiutiu et al. [55] used shape memory alloys as force-displacement actuators to reduce vibration of helicopter rotor blades. Davison et al. and Jardine et al. [36] reported on the usage of shape memory alloys for torque tube actuators in aircraft wings. Melton [54] reported on a shrink to fit shape memory alloy coupling that was used to connect titanium hydraulic tubing in the Grumman F-14 fighter jet. Rediniotis et al. used shape memory alloys as actuation devices for hyper silent naval submersibles [36].

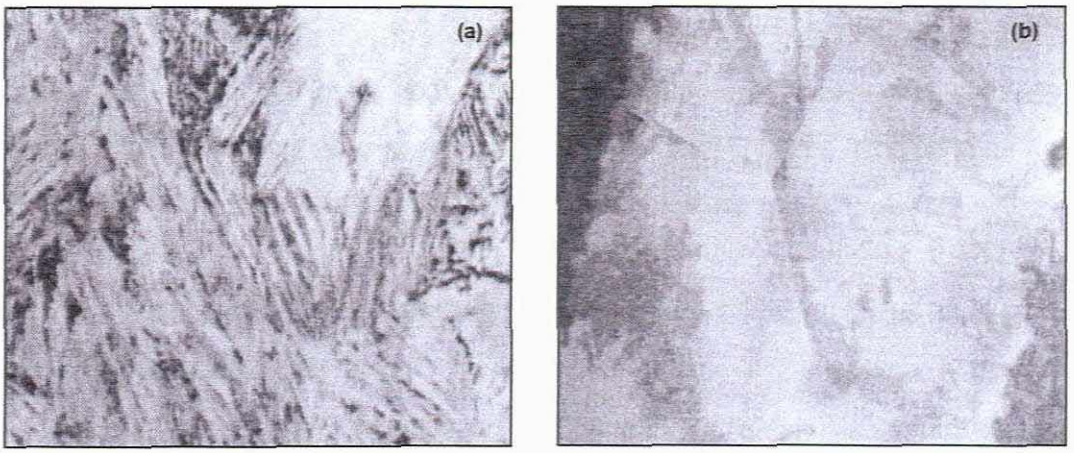


Figure 1.2: Metallurgical phases observed in NiTi alloy shape memory alloys. These images were obtained using transmission electron microscopy (TEM). (a) The low temperature martensite phase showing different twinned variants, and (b) the high temperature austenite phase. [22]

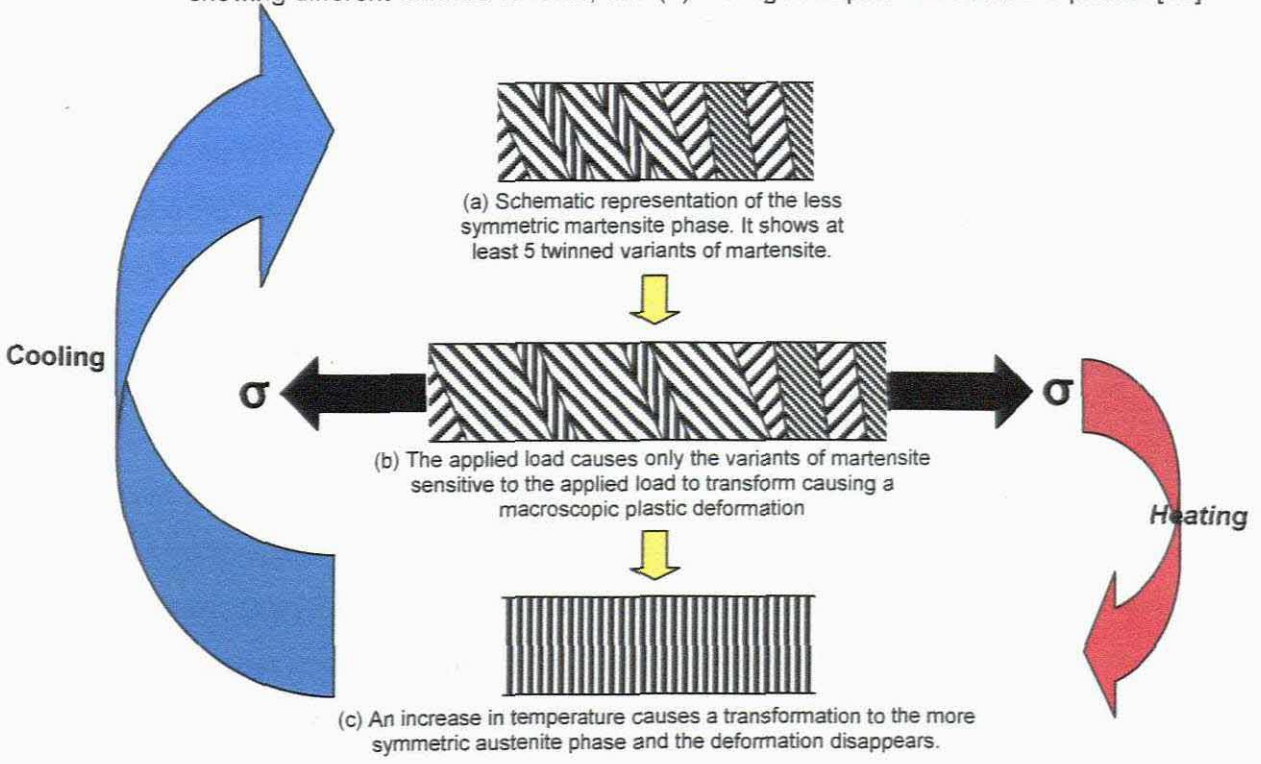


Figure 1.3: Schematic representation of the shape memory effect

Duerig et al. and Pelton et al. [45] have reported innovative applications for the pseudo-elastic effect and its inherent energy dissipation. Sachdeva and Miyazaki [36] found applications for pseudo-elasticity in orthodontic wires. The energy dissipation of shape memory alloys were used for damping applications by Graesser and Cozzarelli, Lagoudas et al., Oberaigner et al., and Boyd and Lagoudas [9].

Other applications included eyeglass frames, frames for brassieres, antennas for portable cellular telephones, blood clot filters, and window latches. The literature is rich with novel applications for shape memory alloys and new devices are developed everyday. Van Hambeeck [54] gives a good overview of more mainstream engineering applications of shape memory alloys.

1.4.2 Review of Experimental Analyses and Findings on Shape Memory Alloys

The above discussion shows that at the heart of the unique material responses that shape memory alloys exhibit lies within its austenite (A) $\Leftrightarrow$ twinned martensite (M^T) transformation upon heating or cooling, the austenite (A) $\Leftrightarrow$ de-twinned martensite (M^{DT}) transformation at elevated temperatures due to an induced stress (either tension, compression, torsion, etc.), and the twinned martensite (M^T) $\Rightarrow$ de-twinned martensite (M^{DT}) transformation at low temperature due to an induced stress (either tension, compression, torsion, etc.). These phase transformations have been the focus of many experimental studies in the past forty years ([10], [14], [22], [28], [31], [38], [46] - [48], [51]).

Otsuka et al., Delaey et al., Perkins, Schetky, and Funakubo [2], [44], [52] were among the first researchers to report that the remarkable behaviour of shape memory alloys is caused by interplay of a high temperature austenite phase and a low temperature martensite phase. Furthermore, Wasilewski et al., Khachin et al. and Miyazaki et al. [46] demonstrated that the transformation temperature and other properties of shape memory alloys could be altered by small changes in the composition of the alloy and by

various thermo-mechanical heat treatments. Daniels, and Pope and Judd used a differential scanning calorimeter to study the phase transformations exhibited by shape memory alloys [46].

For most practical engineering applications, shape memory alloys are usually used in NiTi alloy wire forms [9]. After the shape memory effect was first observed in NiTi alloys a great amount of metallurgical research has gone into it and Wasilewski, Jackson et al., Ling and Kaplow and Miyazaki were among the first researchers to experimentally study NiTi [46]. These experiments looked at the effects of temperature, strain range and mechanical cycling on NiTi alloys.

Due to the complexities of the energy dissipation associated with the hysteretic nature of pseudo-elastic behaviour, most researchers appeared to study this phenomenon experimentally more than the shape memory effect. Wayman and Deurig, Wayman, and Fu are among these researchers while Müller and Xu, and Fu, Müller and Xu were among the first to experimentally study the interior of the hysteresis loop [2], [20-21].

As time progressed and experimental apparatus became more sophisticated Chrysochoos et al., Shaw and Kyriakides, Tobushi et al. and Lim and McDowell found that the pseudo-elastic material response of shape memory alloys is associated with significant temperature variations. Furthermore, the stress-induced austenite (A) $\Leftrightarrow$ de-twinned martensite (M^{DT}) transformation at elevated temperatures occurs in well defined stress – temperature regions, it is dependant on the kind of mechanical- loading – unloading and thermal- heating – cooling cycles, and the combined effects of the mechanical and thermal cycles [2], [46-48].

The stress-induced austenite (A) $\Leftrightarrow$ de-twinned martensite (M^{DT}) transformation at elevated temperatures is an unstable transformation and thus produces inhomogeneous macroscopic deformations as reported by Leo et al., Shaw and Kyriakides, Sittner et al. [2], [46-48], [52]. From a

crystallographic point of view, the de-twinned martensite (M^{DT}) phase nucleates and grows as interfaces within the austenite phase until all the austenite has transformed to de-twinned martensite (M^{DT}). Larche and Cahn, Robin, Heidug and Lerner, Johnson and Alexander were among the first researchers to study the equilibrium conditions of these interfaces and motion of the interfaces between regions of two phases [20], [21]. Salzbrenner and Cohen performed investigations to study the motion of these interfaces by controlling the temperature [20], [21].

By far the most detail and beneficial experimental investigations into the unstable transformation and inhomogeneous macroscopic deformation of shape memory alloys, is the works by Shaw and Kyriakides [46-48]. A brief summary of their findings will thus be given below:

- Experimental observation show that stress-induced martensitic transformation in certain polycrystalline NiTi shape memory alloy can lead to strain localization and propagation phenomena when uniaxially loaded in tension;
- The Number of nucleation events and kinetics of transformation fronts were found to be sensitive to the nature of ambient media and imposed loading rate due to release / absorption of latent heat and the material's inherent temperature sensitivity of transformation stress;
- Nucleation stress is higher than the transformation stress;
- During unstable transformation, deformation is distinctly inhomogeneous;
- Each nucleation spawns two transition fronts and active deformation of the transition fronts is limited to the neighbourhood of these fronts;
- As a result, latent heat is released in discrete local regions rather than distributed over the entire length of the specimen (suggest strong thermo-mechanical coupling);
- Higher nucleation stress and displacement rates results in multiple fronts;
- Coexisting fronts travel at the same speed;
- The front speed is proportional to the rate of the applied end displacement;
- The front speed in inversely proportional to the number of active fronts;

- More proportions of fronts implies lower front speed and reduced local rate of heating;
- Distinct instability and Lüders-like deformation occurs under isothermal conditions;
- Within an insulating media and higher loading rates the material cause self-heating, which leads to higher force-displacement response and multiple transformation fronts; and
- As the number of fronts increase the deformation appears to become homogeneous.

The type of loading cycle (tension, compression, torsion, etc.) to induce the austenite (A) $\Leftrightarrow$ de-twinned martensite (M^{DT}) transformation at elevated temperatures, and the twinned martensite (M^{T}) $\Rightarrow$ de-twinned martensite (M^{DT}) transformation at low temperature also has important implications to the shape memory alloy material response. Adler et al. and Melton showed experimentally that shape memory alloys subjected to compressive loading showed lower recoverable strains, steeper hardening behaviour, and higher transformation stress levels compared to tensile behaviour [36]. Leo et al., Shaw and Kyriakides, Sittner et al., Gall et al., and Zang et al. studied the effects of tension – compression asymmetry on shape memory material response [46-48], [52]. Gall and Sehitoglu performed experiments on one-dimensional shape memory alloy bars loaded in tension and compression. Their results showed that the shape memory alloys phase transformations is influenced by the texture of the specimen [36].

Sittner et al. performed experimental analyses to study the three-dimensional constitutive behaviour of shape memory alloys. They investigated the stabilization of transformation behaviour in stress-induced martensitic transformation in NiTi alloy hollow bar. The loading cycles were for combined Tension and Torsion. The specimen was then loaded either in tension or torsion. The results revealed strain anisotropy, which implies the existence of other strain components [52].

Jacobus et al. studied the effect of stress state on the stress – induced martensitic phase transformation. NiTi alloy bar specimens were subjected to tension, compression, hydrostatic pressure, and various tri-axial compression tests. The results showed that shape memory alloy phase transformation is pressure dependant and the effective stress for phase transformation increase with increasing hydrostatic pressure [36].

Lim and McDowell conducted experiments on shape memory alloy torque tubes using proportional and non-proportional loading conditions. They investigated the stress-temperature coupling in tension, compression, and torsion. Their results showed that more latent heat was available in compression, which was evidenced by larger temperature increases in compression than in tension for the completion of a phase transformation. They also found that the thermo-mechanical behaviour in positive and negative torsion was found to be symmetrical [36].

In conclusion all experimental efforts tries to bridge the gap between the microscopic and macroscopic material responses of shape memory alloys. The experimental findings detailed above shows the strong coupling between thermal and mechanical properties of shape memory alloys. The mechanical material response is dependant on many factors including temperature, strain range, thermo-mechanical history, loading rate, specimen geometry, nature of ambient medium, and the interactions of some of these [see 46-47]. Furthermore, the phase transformations associated with the shape memory effect and the pseudo-elastic effect produces unstable material behaviour and inhomogeneous deformations due to the propagating phase transformation phenomena, which is also temperature and load dependant. Added to this is the sensitivity of the phase transformations on magnitude and type of the applied loading and unloading conditions. All of these material complexities make the constitutive modelling of shape memory alloys an enormous task. In order for successful constitutive models to be constructed knowledge bases associated with Material Science, Physics, Mathematics, and Engineering has to merge.

1.4.3 Review of Constitutive Modelling for Shape Memory Alloys

1.4.3.1 Phenomenological Approach

The earliest form of constitutive models used to study shape memory alloy material response were of a phenomenological nature. The progression of these models took the form of the initial mechanical state space phenomenological models, and then later when thermodynamics were included in the formulations, two- and three-component phenomenological models.

The state space one-dimensional constitutive phenomenological models formulated to study shape memory alloy material response have been ideally suited for the engineering practice since these models make use of measurable quantities as parameters in their construction. Most of these models were constructed to fit one-dimensional experimental data and thus predict the uniaxial behaviour of these materials quite well. These models consist of mechanical laws for the governance of stress – strain behaviour of the material and kinetic laws, which governs the crystallographic behaviour of the material.

The kinetic laws describe the evolution of phase fractions as functions of applied stress and temperature and they usually make use of phase diagrams to show the crystallographic behaviour of the material under stress and/or temperature. Two methodologies are followed for the derivation of the evolution laws. The one employs the use of transformation micro-mechanics (Tanaka, Sato and Tanaka), while the other is obtained by directly matching experimental results (Liang and Rogers, Graesser and Cozzarelli, Barrit) [26]. Brinson and Huang reported that main difference between these models are the kinetic laws since the mechanical elastic strain components are small compared to the transformation strain components and the mechanical part thus plays a less significant role in these models [9].

The phenomenological constitutive model also has two categories where either a two or three component model is described [57]. The two component models consist of crystallographic variants of austenite and a single variant of martensite (Sun and Hwang, Raniecki and Lexcellent), while the three component models describes the existence of austenite and two variants of martensite, i.e., the self accommodating twinned variant of martensite and the non-self accommodating de-twinned variants of martensite (Boyd and Lagoudas, Leclercq and Lexcellent). Both of these categories are derived from the assumption that the direction of the strain rate is the same as that of the devatoric stress. This assumption is used in the construction of the free-energy function and only holds for proportional loading. Furthermore, thermodynamic dissipation potential functions are postulated in the construction of the evolution equations for the phase transformations [57].

Several attempts have been made to extend these models to a three-dimensional setting. The three-dimensional models capture most of the typical features of shape memory alloy material response. There is however problems in evaluating their performance due to a lack of sound experimental data for the multi-axial material response. These models are in the form of plasticity models that consist of an internal variable representing the volume fraction of martensite.

Tanaka and Ngaki proposed a state space uniaxial phenomenological constitutive model that used a fraction of martensite as an internal variable. The phenomenological equation of state is given in the form of an exponential function and uses the Kiostenen – Marburger type of kinetics to describe transformation behaviour of shape memory alloys [42], [52]. The evolution of the phase fraction is dependant on stress and temperature. These researchers were the first to use Edelen's formulation to study the material response of shape memory alloys [8]. This formulation demonstrates the existence of a vector decomposition theorem such that the generalized fluxes appearing in the entropy production inequality can be decomposed into dissipative and non-dissipative components. The dissipative fluxes must now be derived from the dissipation potential [8].

Tanaka et al. constructed a three-dimensional thermo-mechanical framework to study the stress-induced martensitic phase transformation and its reverse transformation which gives rise to the pseudo-elastic and the shape memory effects of shape memory alloys. This constitutive model was generalized form one-dimensional experiments. The transformations are characterised by internal state variables. The thermo-mechanical constitutive equations and phase transformation kinetics are derived as a consequence of the Claussius – Duhem inequality. The Koistenen – Marburger and Wang and Inoue type of transformation kinetics are employed to study phase transformations occurring in shape memory alloys. A theoretical and numerical study on the stress-strain-temperature curves and the shape recovery on heating were examined. This study was however limited to macroscopic behaviour of shape memory alloys and the martensite-martensite and stress-induced transformations were not considered here [52].

Brinson was the first to recognise the decomposition of martensite into twinned and de-twinned components which now made it possible to model the quasi-plasticity of shape memory alloys [42]. He developed a uniaxial phenomenological macroscopic model based on previous work by Tanaka to also study the material response. Here the internal variable consisted of a purely thermal self-accommodating phase fraction (representing twinned martensite) and a thermo-mechanical stress-induced orientated phase fraction (representing de-twinned martensite) [9], [27]. Liang and Rogers studied the one-dimensional behaviour of shape memory alloys with a state space phenomenological model that used a cosine law for the calculation of martensitic volume fractions. This evolution law was obtained by matching experimental data [9], [17], [18], [42]. Liang and Rogers later extended this uniaxial phenomenological model to a three-dimensional setting in the form of a plasticity model with internal state variables representing martensitic volume fractions [9], [36]. Brinson and Lammering also followed this approach [17], [36].

Sun and Hwang presented a two component three-dimensional phenomenological model that was generalized from one-dimensional experimental data [36], [57]. Raniecki and Lexcellent proposed a 3 component phenomenological constitutive model to study transformations that occur during pseudo-elasticity [8], [27], [57]. Boyd and Lagoudas followed the same approach as Sun and Hwang. This model was derived from irreversible thermodynamics with evolution laws being developed from the basis entropy production. The three-dimensional model follows the classical theory of plasticity with yield conditions for the onset of phase transformations. Although thermodynamics is used in this formulation, the authors still used phenomenological observations to construct it [26], [36], [42]. Lagoudas et al. shows that several earlier models based on different approaches are all related to each other under thermodynamic formulations [17].

Leqlercq showed that the effect of reorientation of the self-accommodating product phase was not adequately taken into account, and thus prevents good prediction of the shape memory alloy material behaviour [27]. Tanaka et al. now presented a comprehensive three-dimensional exponential constitutive model that attempts to account for reorientation, kinematic and isotropic hardening to include the dependence on applied stresses. Again generalized from one-dimensional experimental data [36-37].

In their three component phenomenological model, Leclercq and Lexcellent, present a macroscopic description for shape memory alloys to allow for the simulation of the global thermo-mechanical material response. It is based on the framework of thermodynamics of irreversible processes and accounts for two internal variables. One accounts for the self accommodating twinned martensite variant, dependant on temperature only, and the other represents the orientated de-twinned martensite variant dependent on the induced stress. The specific free energy function derived for this model covers the total range of phase transformations. The kinetic equations is formulated according to the Second Law of Thermodynamics and five yield functions are postulated for each of the related processes associated with the material behaviour. The simulated results show good agreement with experimental data [27], [57].

Auricchio and Sacco proposed a macro-level constitutive model based on kinematical approaches and assumes that sections that are plane in the un-deformed configuration remain so in the deformed configuration. Furthermore they assume that shear deformations are negligible. A one-dimensional constitutive equation is used for the stress – strain behaviour and uses an internal scalar variable to represent the martensitic volume fraction. The scalar variable is dependant on evolution equations for the onset of phase transformations. Thermo-mechanical coupling is accounted for by the presence of internal heat sources in the form of phase transformation latent heats and mechanical dissipation. The computational framework was set up for the simulation of shape memory alloy devices consisting of beam elements with generic cross-sections. These elements were subjected to complex loading conditions [2].

Shaw presents a thermo-mechanical boundary value problem and a constitutive model for a NiTi alloy wire subjected to uniaxial loading conditions. The thermodynamic relationships are derived from phenomenological observations consistent with experimental data. An explicit Helmholtz free energy function is developed that includes internal variables representing phase fractions for austenite, and one-dimensional variants of martensite. Hysteretic kinetic relations augment this free energy function, which governs the rate of phase transformation as a chemical driving force. The model accounts for temperature and stress-induced phase transformation and thus makes it possible to study both shape memory and pseudo-elastic behaviours. Furthermore, the model accommodates for possible unstable mechanical behaviour during stress-induced transformations by allowing softening transformation paths including strain gradient effects [45].

Recently, Helm and Haupt proposed a phenomenological material model to represent multi-axial material behaviour of shape memory alloys. The model accounts for the one-way shape memory effect, two-way shape memory effect due to external loading contributions, pseudo-elasticity, pseudo-plasticity, and the transition range between pseudo-elasticity and pseudo-

plasticity. The free energy function is dependent temperature and internal variables representing a martensitic volume fraction, an inelastic strain tensor, and a strain like variable describing internal or residual stress fields. Evolution equations govern the history dependence on these internal variables. Viscous behaviour during phase transformation and reorientation of martensitic variants are dealt with by including a Perzyna type inelastic multiplier. Different deformation mechanisms are dealt with by introducing case distinctions into evolution equations. Thermodynamic consistency is ensured since the constitutive model satisfies the Cluasius – Duhem inequality [18].

1.4.3.2 Micro-Mechanics Approach

The constitutive models following the micro-mechanics approach follow closely the crystallographic phenomena occurring in the internal structure of the shape memory alloy material. These models use the Thermodynamic Laws to describe the transformation phenomena. Two methodologies are also followed here. One uses the crystallographic modelling of a single crystal or grain represented by a volume that consist of inclusions representing possible variants of martensite. The results (stresses and strains) of this modelling are averaged over what is termed the representative volume element (RVE) to obtain the overall polycrystalline response. Researchers following this approach included Patoor et al., Sun and Hwang, Tokuda et al., and Lim and McDowell [2], [8], [9], [36].

Raniecki and Lexcellent, Reisner et al., Bo and Lagoudas, Gillet et al., and Lagoudas and Bo [36], [42] used the other approach which starts off with the construction of a macro free energy potential which is obtained directly from micro-mechanical modelling. The Coleman-Noll procedure is employed to derive constraints on the material constitutive behaviour

These constitutive models employ dissipation potentials in conjunction with the Second Law of Thermodynamics to derive Evolution Laws that describe internal variables. The main difference between these methodologies lies in the form of the transformation strain induced strain hardening term in the free

energy function. Also, different choices of transformation function or criteria, which defines the thermo-elastic domain, are made.

Qidwai and Lagoudas [36] reported that researchers that used the J_2 transformation function (Boyd and Lagoudas [36], Bo and Lagoudas [36], [42]) are incapable of modelling asymmetric behaviour and the dependence of phase transformations on hydrostatic pressure. Volumetric changes associated with phase transformations also cannot be modelled if the associative transformation strain flow rule is assumed. The $J_2 - I_1$ transformation function (Aurricchio et al. [36]) can model the tension – compression asymmetry and phase transformation pressure dependence. Models that use the $J_2 - J_3 - I_1$ transformation function however demonstrate the ability to capture most of the material dependence on loading conditions such as volumetric changes associated with phase transformation, asymmetrical behaviour, and the phase transformation pressure dependence by distorting the transformation surface [36].

1.4.3.3 Micro-Plane Approach

The constitutive model derived from Micro-Plane Approach falls between the two methods described above. Brocca et al. [9] first introduced this theory. It is in essence a phenomenological model that is aimed at reproducing the macroscopic mechanical behaviour of a material. In the micro-plane approach the macroscopic material response is obtained along several planes of different orientations, which is called micro-planes. This is in contrast to the phenomenological approach since constitutive laws are expressed here directly in terms of stress and strain tensors and their invariants. By considering micro-planes to be representative of the material microstructure, the cumbersome task of microstructural modelling is eliminated. This then lends a new method at dealing with material phenomena at a micro-scale. Results obtained on these micro-planes can be used to describe phenomena such as crystallographic slip, shear bands, crack opening, friction, etc.

The model proposed by Brocca et al. [9] for the material behaviour of polycrystalline shape memory alloys can predict three-dimensional response by superposing the effects of inelastic deformations obtained on several planes with different orientation. This then closely mimics the actual physical behaviour of the material. The constitutive law for the material is composed of a simple constitutive model and a robust kinetic expression for each of the orientated micro-planes. This model showed promising results in that uniaxial material response of shape memory alloys were easily reproduced. The model also tackles complex responses such as stress – strain sub-loops, tension – compression asymmetry, and material effects associated with non-proportional loading paths.

1.4.3.4 Multi-Well Approach

This approach has been followed by researchers such as Achenbach and Müller, Achenbach, Müller and Xu, Huo and Müller, Abeyaratne et al., and Abeyaratne and Kim [8], [21], [27]. The extended version of the Müller-Achenbach model [32], [42] describes the thermo-mechanical behaviour of a shape memory alloys wire for actoric purposes. This model is strongly dependent on experimental data, and the resulting mathematical structure is given by a set of differential-algebraic equations [32], [39-42]. The basic element of this model is a small piece of metallic lattice, which can exist in three equilibrium conditions, i.e., austenite, and the martensitic twins. Microscopically, a tensile specimen is arranged as layers of such alternating lattice particles. A train of parabola or a triple-well function gives the postulated form of the potential energy of this lattice. It is characterised by two stable minima each representing one of the martensite twins and a metastable minimum for the austenite. The points at which these wells are joined are called the potential barriers [32], [39-42]. Under certain thermal and mechanical loading conditions, the layers are able to overcome the potential barriers through thermal activation, which gives rise to time rates of change for the different phase variants. These transformation kinetics is based on statistical physics, wherein it is postulated that the rate of transformation between constituents is proportional to the net probability that one phase will

overcome the energetic barrier required to transform to a second phase. This model also uses the analytical description for the coherence energy first introduced Müller (1989) [21], for the hysteretic behaviour of shape memory alloys material response during pseudo-elasticity. Müller and Seelecke [32] used this model and it agrees at least qualitatively with experimental data. Seelecke and Büskens [41] used it to optimally control beam structures using shape memory alloys Seelecke [39] studied the torsional vibration of shape memory alloys with it.

1.4.3.5 Maxwellian Visco-Elastic Approach

This approach is used to find simple and appropriate dissipative mechanisms to be included in the constitutive description of materials so that the nucleation and propagation of phase fractions associated with phase transformations may be analysed. Suliciu [13] formulated a new approach to solve material instabilities caused by phase transformations or strain localization. This approach falls in the framework of a one-dimensional and isothermal context. Other researchers that followed this approach include Făciu, Suliciu, Făciu and Suliciu, Făciu, and Milhăilescu-Suliciu and Suliciu [13]. This constitutive approach is based on rate-type visco-elastic constitutive equations with Maxwellian viscosity. The rate of stress, rate of strain, and the over stress function are considered to be linearly dependent. Equilibrium states are considered to be situated on a stress – strain curve corresponding to a non-monotone elastic material. The model may be compared to a simple body with one internal variable (inelastic strain) and is capable of describing relaxation, creep, and instantaneous processes occurring within a material. Furthermore, the nucleation, growth of one phase in another, the creation and propagation of phase boundaries is automatically accounted for in this model [13].

Făciu and Milhăilescu-Suliciu [13] used this approach and included thermal effects associated with the phase transformations occurring in shape memory alloys to formulate a rate-type semi-linear thermo-visco-elastic constitutive

approach for a non-linear and non-monotone thermo-elastic body. Thermo-mechanical coupling is considered by including a Newton type convection mode for heat transfer between the material and its environment. Numerical simulations with this model were conducted for three end-displacement rates and compared qualitatively with experimental results [13]

1.4.3.6 Rubber-Like Elasticity Approach

This theory found it's footing in the curious aging effects observed in shape memory alloys. Sawada et al. [38] performed experiments on NiTi alloy wire to determine the effect that the shape memory processing temperature has on its material response. The wires were held at a certain temperature (623K ≤ T ≤ 773K) for one hour and cooled in a furnace. This process is referred to as aging of the NiTi alloy wires. The wires were now subjected to a uniaxial tensile test at different temperatures. It was found that the aging process increased the martensite start temperature and decreased the austenite start temperature [38]. This caused the aged martensite to transform to austenite at a lower temperature and thus pseudo-elasticity is now observed at lower temperatures. Ohta [33] summarizes the aging effects as follows: (1) martensitic phases are stabilized by aging, and (2) the aged martensite reveals rubber-like elasticity (pseudo-elasticity). Ölander discovered rubber-like elasticity and a number of experimental studies regarding the phenomena have been conducted. Otha formulated a theory for the rubber-like behaviour exhibited by shape memory alloys. The kinetics of the martensitic phase transformation is represented in terms of a local strain field or primary order parameter. The evolution of the primary is followed by a secondary order parameter that has an extremely slow movement. This set of equations then enables the derivation for the equation of motion for twin boundaries. The stress – strain relation due to the motion of this twin boundary is obtained by taking account of the aging effect. The results obtained here were found to be qualitatively consistent with experimental data [33].

1.4.3.7 Concluding Remarks on Constitutive Modelling of Shape Memory Alloys

Most of the material responses, i.e., shape memory effect, partial pseudo-elasticity, and pseudo-elasticity, observed in shape memory alloys have been captured to some extent by all constitutive models described above. It is clear that all researchers acknowledge that at the heart of the unique material responses exhibited by shape memory alloys lies the thermo-elastic reversible martensitic phase transformation and thus all constitutive models used this as an initial starting point for their formulations. Since Thermodynamics is the choice of science for the derivation of shape memory alloy constitutive models, the extent to which the thermodynamics of phase transformations were studied and implemented in these models displayed their level of accuracy in simulating the material responses. Other aspects related to the inhomogeneous deformation associated with the phase transformations (Shaw [45]) and the dependence of the phase transformations on thermal and mechanical fields, and type of loading conditions, were also accounted for (Auricchio and Sacco [2], Brocca et al. [9]).

State space phenomenological constitutive models (Tanaka and Ngaki, Tanaka, Liang and Rogers [8], [9], [18], [37], [42]) although the choice of engineers lack the thermodynamic aspects related to the shape memory alloy material response. Their suitability for the simulation of actuator applications is also very limited [42]. Two- and Three- Component Phenomenological Models (Sun and Hwang, Raniecki and Lexcellent, [54]), Micro-mechanics Models (Patoor et al., Sun and Hwang, Tokuda et al. [9], [36]), and the Micro-Plane Model (Brocca et al. [9]) possess a high level of sophistication and can simulate shape memory alloy material response to a high degree in that they offer very good agreements with experimental data, both qualitatively and quantitatively. They are however too complex for use in the design one-dimensional shape memory alloy actuators. Most of these models require the calculation of different constants for their successful implementation. The model by Shaw [45] requires eight material constants while the model by Boyd and Lagoudas [8] requires up to eighteen material constants.

Other constitutive models based on the Maxwellian Visco-Elastic approach (Faciu and Mihailescu-Suliciu [13]), and the Rubber-Like approach (Otha [33]), are developed from experimental observations and an in depth knowledge of the constitutive approach that was used. These models however, only show good qualitative agreement with experimental data and work still has to be conducted to bring them to a quantitative level [13], [33].

The Multi-Well or Triple Well Function approach (Müller and Seelecke, [32]) used in the constitutive modelling of shape memory alloys provides a good one-dimensional description of the material response accounting for both thermodynamic and mechanical aspects [32], [42]. All model variables required for the implementation of this constitutive approach are obtained from only two tensile tests. This model requires only two material constants for its implementation, i.e., the Elastic Modulus of each metallurgical phase. Furthermore, it offers the possibility of being implemented into advanced control algorithms [39], [40], [41].

1.5 Scope of the Thesis

Chapter 2 gives details about an experimental investigation that was conducted on NiTi shape memory alloys wires. This section served as a reference when various constitutive responses were consulted for their use in the computational design tool. Chapter 3 presents the constitutive model that was used in the finite element formulation and Chapter 4 gives the formulation of the Shape Memory Alloy Truss (SMAT) finite element. Chapter 5 gives the computation framework of the design tool and includes a discussion on the numerical solution scheme used. This section also shows the use of the design tool by simulating the behaviour of shape memory alloy actuators harnessing the shape memory effect.

Chapter 2

Experimental and Macroscopic Observations of Shape Memory Alloys

2.1 Introduction

In order to fix ideas about our present investigation into the complex constitutive modelling of shape memory alloys it is necessary to present some experimental data and results obtained for these materials. Shape memory alloys exhibit two unique material responses ascribed to solid-solid thermo-elastic martensitic forward and reverse phase transformations, i.e., the shape memory effect and the pseudo-elastic effect. The experimental investigation was performed to study only one aspect of the shape memory effect, the quasi-plastic material response under tensile loading conditions. Shape memory alloy while in its low temperature martensitic state exhibit this behaviour under different loading conditions (tension, compression, torsion, tension-torsion, compression-torsion, etc.). The experimental results obtained are consistent with what other researchers found in the field and even show some interesting findings regarding this unique material response. It is hoped that these results will add to the body of work performed in this area of research but also more importantly provide an insight into the different approaches used in the constitutive modelling of shape memory alloys. All

experiments were conducted at the Strength of Materials Laboratory of the Department of Mechanical Engineering at the Peninsula Technikon.

2.2 Materials Tested

Commercially available NiTi shape memory alloy wires of 1, 2 and 3mm diameter were used in these experiments. The material was obtained from the supplier Johnson Matthey, based in the United Sates of America and Table 2.1 shows their properties. It should be noted that quantitative comparison of these three materials would not be possible since slight changes in the chemical composition may result in drastic inconsistencies with regards to material responses. We will therefore have a qualitative discussion on the material response of three materials and a quantitative discussion on each.

Diameter (mm)	Chemical Composition					Temper	Surface	Active A_f (^{0}C)
	Ni	Ti	C	O	Total All Others			
1	55.32	44.67	≤ 0.05	≤ 0.05	0.20	Straight Annealed	Oxide	60.7
2	55.38	44.62	≤ 0.05	≤ 0.05	≤ 0.20	Straight Annealed	Oxide	60.5
3	55.32	Bal	≤ 0.05	≤ 0.05	≤ 0.30	Straight Annealed	Oxide	73
All Others Are: Al, Co, Cr, Cu, Fe, Mn, Mo, Nb, Si, W								
Materials obtained from Johnson Matthey, 1070 COMMERCIAL ST., SUITE 110, SAN JOSE, CA 95112								

Table 2.1: Shape memory alloy material data

2.3 Experimental Set-ups

Four lengths of 100, 150, 200 and 250mm for each diameter wire were considered for testing. Three specimens of each length were cut from the respective wire reels using an industrial bolt cutter and gave a total of thirty-six specimens to be tested. The as cut specimens were placed in an furnace and heated to a temperature of 100^{0}C. The specimens were held at this temperature for a sufficient amount of time and this ensured that all specimens were transformed to their austenitic state (Table 2.1 shows austenitic finish temperature of each material). All specimens were then removed from the furnace and allowed to cool to room temperature in still air

ensuring that at the beginning of testing they were all in their martensitic states. A Hounsfield Tensile Tester was used for tensile experiments and specimens were clamped in such a way that the effective gauge lengths were 60, 110, 160 and 210mm. Loads were measured with 5kN (for 1mm diameter specimens) and 20kN (for 2 and 3mm diameter wires) load cells. The cross head displacement was measured by a digital encoder built in to the tensile tester. Each specimen were subjected to displacement rates of 2, 3, 4, 5, 7, 10 and 15mm/min which gave three data sheets per length and gives a total eighty-four data sheets per diameter. The specimens were never fractured and after unloading they were placed back into the furnace and heated to restore their original geometric properties.

2.4 Quasi-Plastic Material Responses Exhibited by Shape Memory Alloys

The quasi-plastic material response is inherent to the shape memory effect exhibited by shape memory alloys. When a load is applied to the material at low temperatures, it will deform causing an apparent plastic deformation, i.e. quasi-plasticity. This deformation is maintained until an increase in temperature causes the deformation to disappear completely. 4-8% erasable deformations have been observed in alloys like NiTi, CuZnAl, CuAlNi, and AuCd [19], [31], [24], [45]-[48]. For a general discussion on this material response it is suggested that the reader not focus on the values for force and displacement at this stage but rather look at a qualitative description. A discussion of load vs. extension behaviour for low, intermediate and high displacement rates for the different diameter wires are shown below.

2.5 Results and Discussion

2.5.1 Qualitative Discussion

The most observable feature of thequasi-plastic material response is the three distinct regions it shows when plotted graphically (see Fig 2.1a, b and c). For discussion purposes lets call them respectively the initial elastic region (IE), a nearly horizontal region (NH), and finally another elastic region (FE). All of

these regions show an amount of yielding. Since regions 1 and 3 shows elastic behaviour (which implies elastic properties), unloading from these regions will remove the strains obtained here. These two elastic regions suggest that the specimen changes from its initial state to some final state. Furthermore it is also seen that the slopes of these elastic curves are similar thus implying equal elastic constants. The nearly horizontal regions on these graphs thus determine the location of the transformation behaviour from one elastic state to the other. It also has the largest value of displacement compared to the other two regions. The end of transformation usually has a distinct point. This is evidenced by the graphs shown in Figs. 2.1a, b and c.

Literature tells us that this initial state is in fact composed of different variants of martensite formed during the transformation from austenite (particularly during cooling). One explanation for this horizontal region [60] is that when these variants are subjected to an applied load only those variants sensitive to the applied load will elongate once a certain transformation load is achieved. The sum of these variant elongations thus produces the overall or total deformation of the specimen.

Another explanation considers these variants as alternate layers or martensitic twins packed on top of each other. When a load is applied to these twins, shear loads are produced at the twin boundaries. When the shear load reaches a certain value one of the twins flip and takes the orientation of the other. This flipping or de-twinning of layers thus produces the total deformation of the specimen [42], [45].

A closer look at the transformation region for a low displacement rate (Fig. 2.1a) of the 1mm diameter shape memory alloy wire shows some very interesting load-displacement behaviour. The insert in Fig 2.1a shows an exploded view of the transformation region. It shows very small fluctuations of the load with regions of constant load between them. It is now seen that the first explanation given above disqualifies itself since it does not give an explanation for these fluctuations. If we consider the second explanation given for the horizontal region, we can take these fluctuations as the load required

to produce the flipping or de-twinning of the martensitic twins. This de-twinning occurs almost at constant load and thus suggests a load induced phase transformation.

Fig. 2.1a shows a decrease in the frequency of these fluctuations with an increase in the displacement rate. This implies that at low displacement rates more of the martensitc twins transform and at high displacement rates less transform. This would then suggest that at low displacement rates greater deformation should be achieved. This is however not the case as seen in Fig. 2.1a. High displacement rates cause lesser fluctuations with approximately the same amount of deformation indicating that higher displacement rates induce a more homogeneous deformation in the 1mm diameter NiTi shape memory alloy wire. An explanation for this could be that at higher displacement rates certain variants of martensite undergo deformation for a longer period of time thus giving the appearance of near homogeneous deformation.

Shaw and Kyriakedis [46-48] experimentally studied propagating phase fronts of shape memory alloys. They found that increases in the displacement rate also increased the occurrence of these propagating fronts and thus produced more homogeneous deformation behaviour.

The 2 and 3mm diameter NiTi shape memory alloy wires does not show the fluctuations observed in the transformation region (see Figs 2.1b and c). These specimens actually show a steeper transformation region and thus imply different kind transformation behaviour.

After the transformation load has been reached and transformation commences, the 1mm diameter wire transforms from one state to the other with a seemingly perfectly plastic behaviour while the 2 and 3mm diameter wires can take the form of plasticity with isotropic or kinematic hardening.

The steeper transformation region observed in the 2 and 3mm diameter NiTi shape memory alloy wire could be ascribed to the geometric changes of the

material only. Thinner wires (1mm or less) can thus be considered as uniaxial test specimens showing the actual horizontal transformation behaviour of shape memory alloys.

Thicker wires thus show a transformational behaviour with hardening, which brings into play the Poisson effect as exhibited by classical steels. These kinds of constitutive approaches using the framework of plasticity-based models was followed by researchers like Shaw and Kyriakides and Bo and Lagoudas to study mostly pseudo-elastic material responses of shape memory alloys [42], [48].

Another interesting observation regarding the quasi-plastic material response is that certain displacement rates applied to different lengths of wire induce the same kind of material response (the same transformation loads, amount of deformation during transformation, and the same slopes for the initial and final elastic regions) in all three specimens tested. Examples of this material response is observed in 1mm diameter wires of 100, 200 and 250mm lengths at displacement rates of 2 and 15mm/min (see Fig. 2.4), 2, 4 and 10mm/min and 2 and 10mm/min respectively. Other displacement rates simply produced different values for the material response. This would then imply that quasi-plasticity has preferred displacement rates and this kind of material response is also observed in 2 (see Fig. 2.5) and 3mm diameter wires.

In their experimental investigation into the behaviour of the pseudo-elastic material response exhibited by shape memory alloys, Shaw and Kyriakides [46-48] found the nucleation load of transformation to be higher than that of the transformation load. This behaviour was also observed for the quasi-plastic material response of 2 and 3mm diameter wires but not for the 1mm diameter wire. Fig. 2.6 shows this phenomena for 2 and 3mm diameter wire of lengths 100 and 150mm respectively at a displacement rate of 2mm/min.

This behaviour also resembles the Lüders deformation found in classical Engineering Steels. This behaviour however disappears with an increase in the displacement rate for certain lengths of wire. Fig. 2.6 shows the load-

extension behaviour of three 2mm diameter wires of length 100mm at a displacement rate of 7mm/min. This behaviour occurs only in one specimen and it is interesting to note that when specimens behave in this way the transformation region becomes steeper showing hardening-like behaviour. This observation was made for most of the 3mm diameter wire (see Fig. 2.8) and also for certain 2mm diameter specimens. A summary of the experimental data is graphically given in Appendix A.

2.5.2 Quantitative Discussion

Experiments were performed to find the effect of displacement rates on geometric properties (diameter and length) and mechanical properties (Initial and Second Yield Stress, Total Transformation Strain) of the NiTi shape memory alloy wire specimens. Exponential functions were extrapolated from the averaged data obtained from the experiments to establish the trend of quasi-plastic material responses for the three different wire diameters that was tested.

Fig. 2.9a, b and c show the relationship between the total transformation strain and the different displacement rates that the specimens were subjected to. The trend lines indicate that an increase in the displacement rate had no effect on the total transformation strain for the 1mm diameter wire specimens but showed slight decreases for the 2 and 3mm diameter wires. Fig. 2.10a, b and c show a slight decrease in the value of the initial yield stress of the different wire specimens as the displacement rate is increased.

Increases in the specimen length show a decrease in the value for the initial yield stresses for the 1 and 2mm diameter wire specimens (see Fig2.10 a and b). This would then suggest that longer wires of these diameters yield easier than shorter wires. For a 1mm diameter wire however, the value of the second yield stress increases with the increase in specimen length and thus compensates for the drop in initial yield stress (see Fig 2.10a). This behaviour then maintains the total amount of transformation strain that the 1mm diameter specimens produce.

The 2mm diameter wire specimens exhibit a different behaviour for the second yield stress (see Fig. 2.9b) compared to that of the 1mm diameter wire specimens. Although this trend line shows a decrease in the value of the second yield with increasing specimen length, it is not as declivitous as the initial yield stress trend line. This could then validate the slight decrease in the total transformation strain as seen in Fig. 2.7b for the 2mm diameter wire specimen.

Both trend lines for the initial and second yield stress values for the 3mm diameter wire specimens show an increase with increasing specimen length. The trend line showing the behaviour of the second yield stress is however steeper than that of the trend line for the initial yield stress and almost resembles the behaviour of the 1mm diameter wire. We can then conclude that this is the way that the 3mm wire specimens are trying to maintain the amount of transformation strain that these wires produce. The slope of the initial yield stress trend line would then suggest a decrease in the total transformation strain and this is indeed the case as seen in Fig 2.9c.

Fig. 2.10a, b and c show trend lines for difference in yield stress values (second yield – initial yield) with respect to increasing specimen length. These trend lines show a parabolic increase and we can thus conclude that this is the way in which the material tries to maintain the total transformation strain that the wire specimens exhibit.

2.5.3 Summary of Experimental Findings

The experimental investigation was performed to determine the effect that different displacement rates have on geometric and mechanical properties of 1, 2 and 3mm diameter NiTi shape memory alloys wires specimens of varying lengths. Selected stress-strain graphs are shown in Appendix A. A summary of these findings will now follow.

- Quasi-plastic material response displays three distinct regions when plotted graphically, i.e., an initial elastic region (IE), a nearly horizontal region (NH), and finally another elastic region (FE).

- Elastic regions suggest that the specimen changes from its initial state to some final state through a transformation and the slopes of these elastic regions are similar.

- The nearly horizontal regions on these graphs thus determine the location of the transformation behaviour from one elastic state to the other.

- Transformation region of the 1mm diameter shape memory alloy wire specimens shows very small fluctuations of the load with regions of constant load between them. These load fluctuations can be regarded as the flipping or de-twinning of the martensitic twins.

- Increase in the displacement rate causes a decrease in the frequency of these fluctuations and thus produce a more homogeneous deformation.

- The 2 and 3mm diameter NiTi shape memory alloy wires does not show the fluctuations observed in the transformation region and show a steeper transformation region.

- 1mm diameter wire transforms from one state to the other with a seemingly perfectly plastic behaviour while the 2 and 3mm diameter wires can take the form of plasticity models that exhibit hardening behaviours.

- The steeper transformation region observed in the 2 and 3mm diameter NiTi shape memory alloy wire could be ascribed to the geometric changes of the material only.

- Thinner wires (1mm or less) can thus be considered as uniaxial test specimens showing the actual horizontal transformation behaviour of shape memory alloys.

- Thicker wires thus show a transformational behaviour with hardening-like behaviour.

- Quasi-plasticity has preferred displacement rates.

- For quasi-plasticity, the nucleation load of transformation is higher than that of the transformation load.

- Lüders-like deformation was observed but disappears with an increase in the displacement rate for certain lengths of wire.

- Increase in the displacement rate had no effect on the total transformation strain for the 1mm diameter wire specimens but showed slight decreases for the 2 and 3mm diameter wires.

- Decreases in the value of the initial yield stress of the different wire specimens as the displacement rate is increased.

- The difference in yield stress values (second yield – initial yield) indicates the way in which the material tries to maintain the total transformation strain that the wire specimens produce.

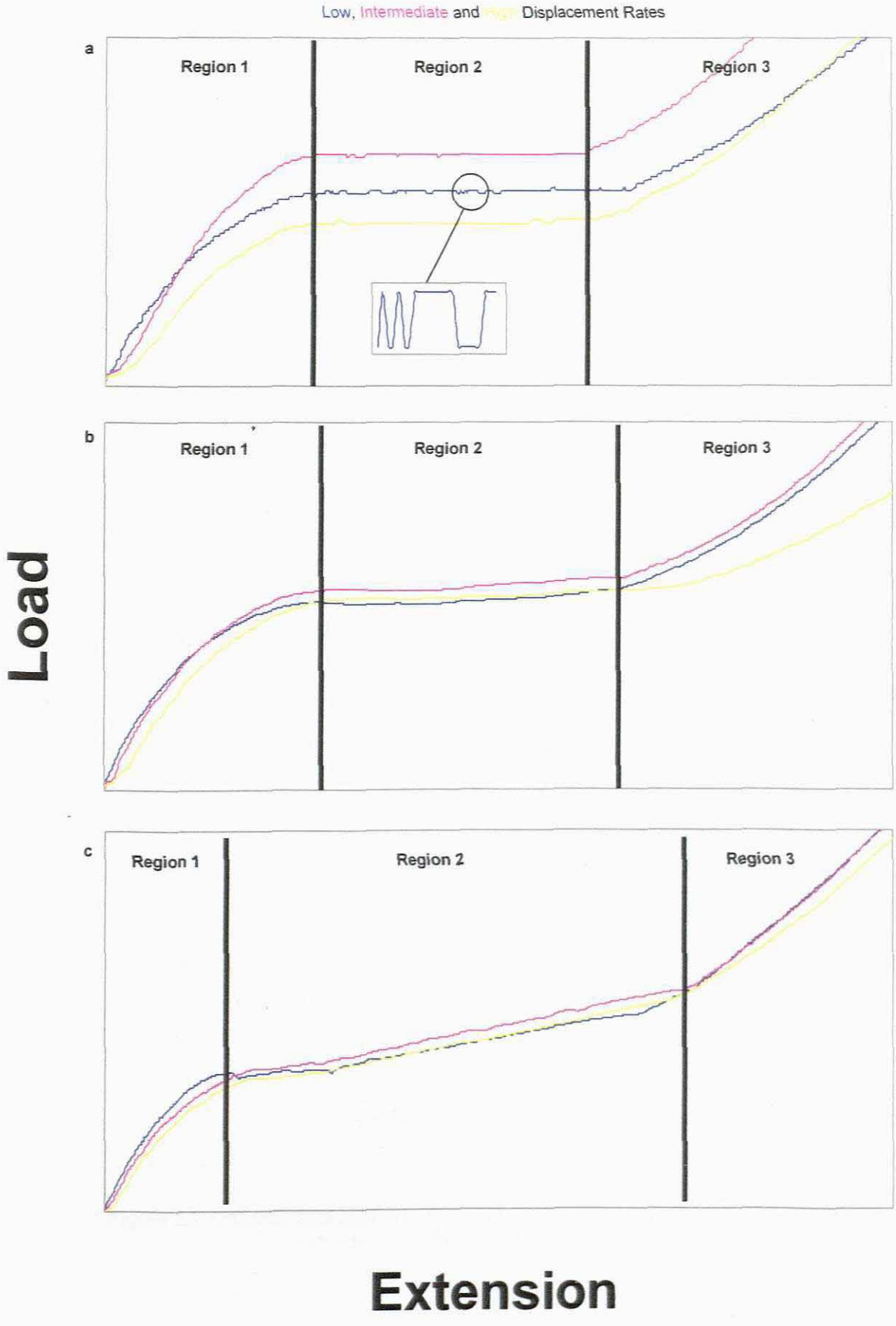


Figure 2.1: Load – Extension behaviour for NiTi Shape memory alloy wires of (a) 1mm, (b) 2mm and (c) 3mm diameters at Low, Intermediate and High Displacement Rates

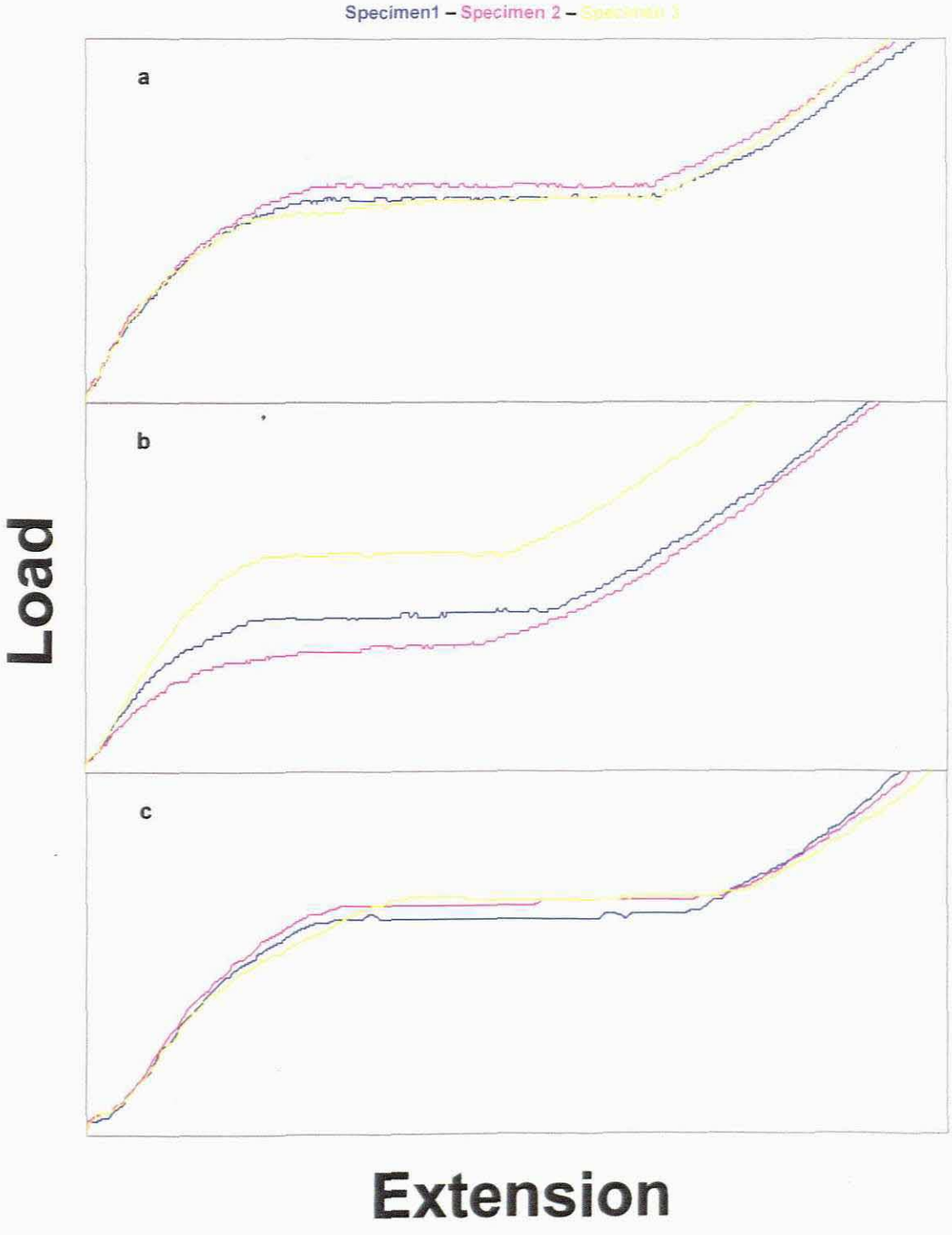


Figure 2.2: Effect of displacement rate on 1mm diameter wire of length 100mm. (a) 2mm/min, (b) 4mm/min and (c) 15mm/min.

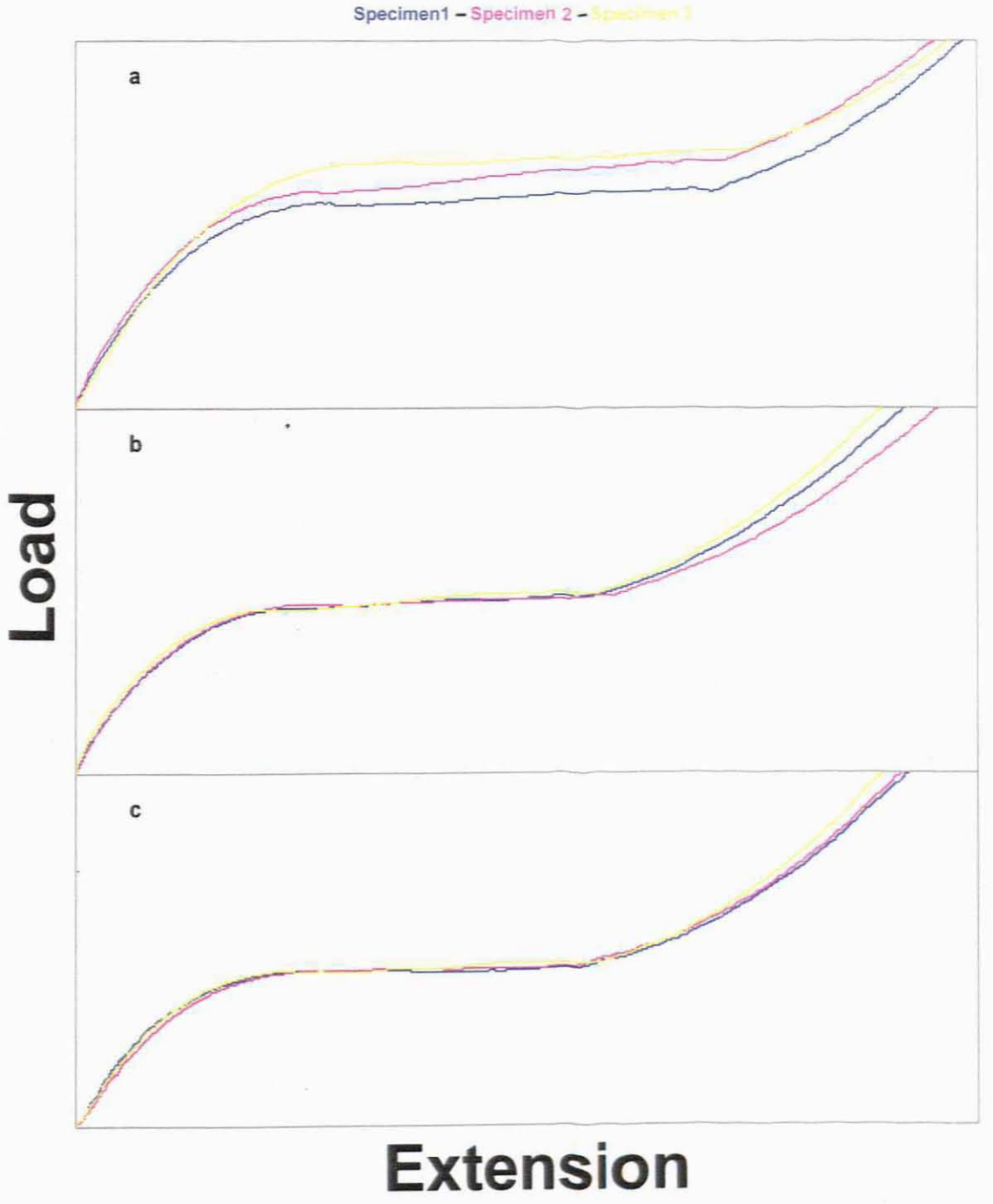


Figure 2.3: Effect of displacement rate on 2mm diameter wire of length 250mm. (a) 2mm/min, (b) 4mm/min and (c) 10mm/min.

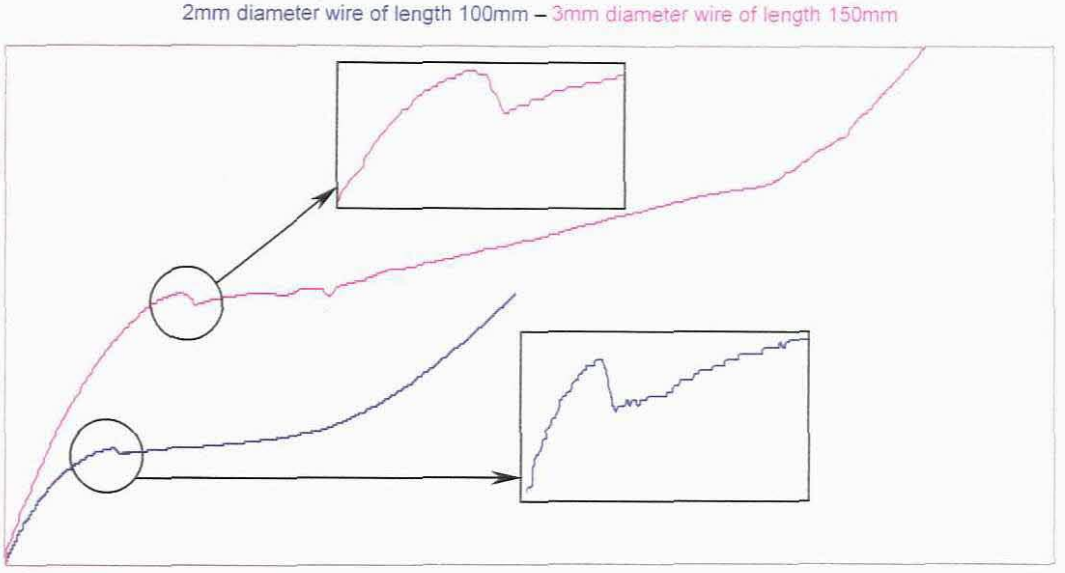


Figure 2.4: Quasi-plastic material response showing relation between Nucleation and Transformation Load behaviour at a displacement rate of 2mm/min.

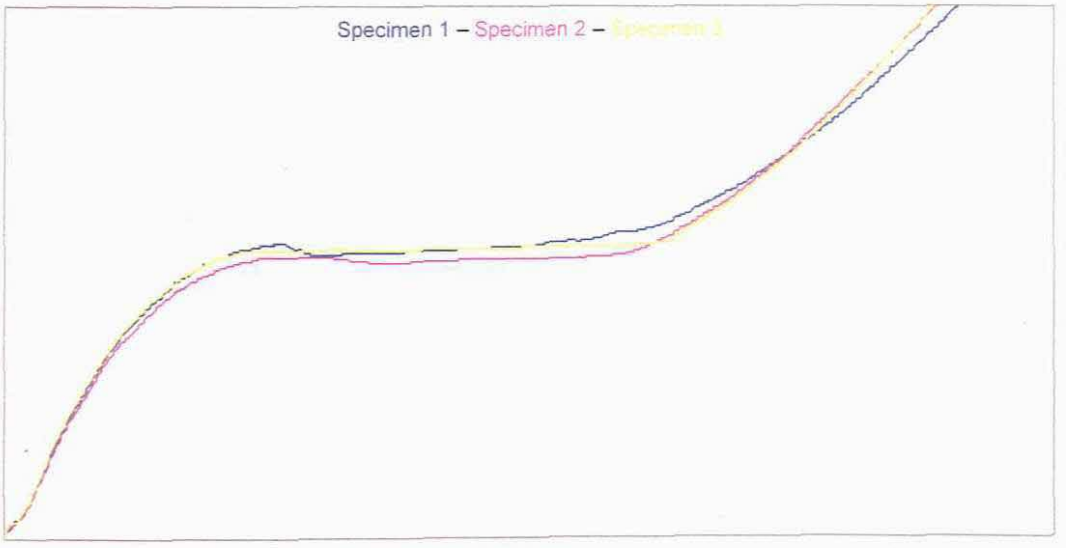


Figure 2.5: Three 2mm diameter specimens of length 100mm each subjected to a displacement rate of 7mm/min.

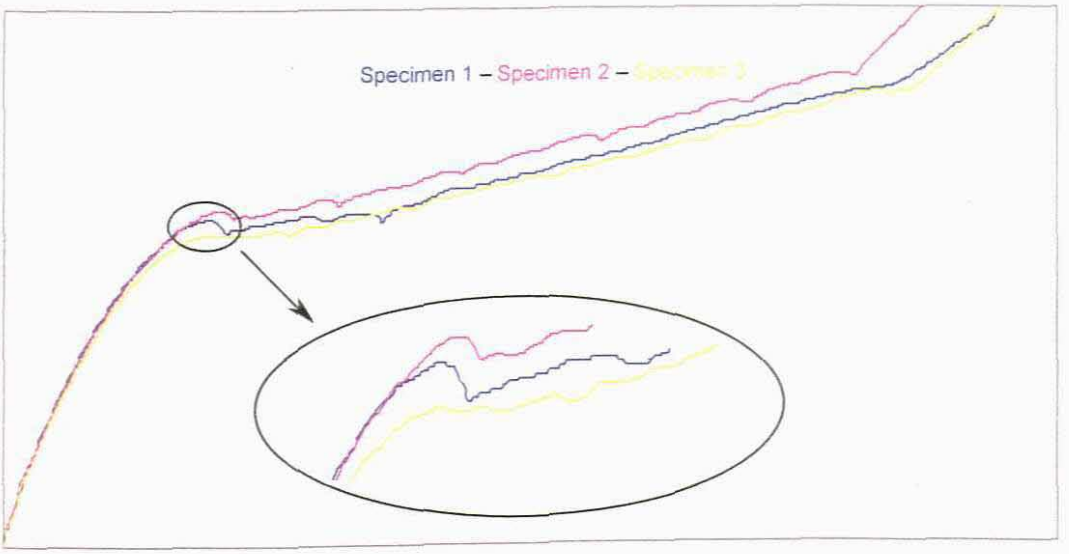


Figure 2.6: Three 3mm diameter specimens of length 150mm each subjected to a displacement rate of 2mm/min.

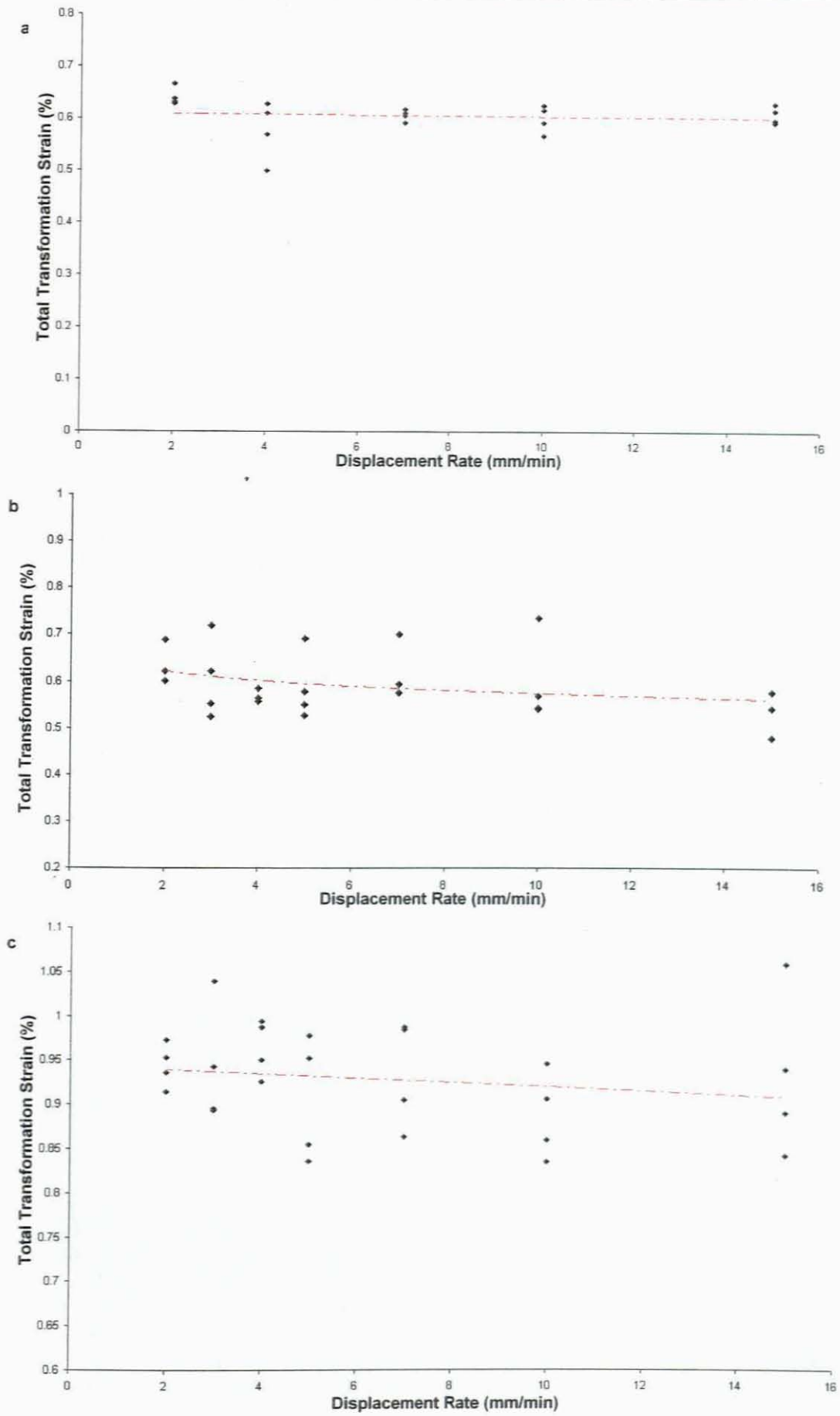


Figure 2.7: Effect of Displacement Rate on the Total Transformation Strain for (a) 1mm, (b) 2mm and (c) 3mm diameter wire.

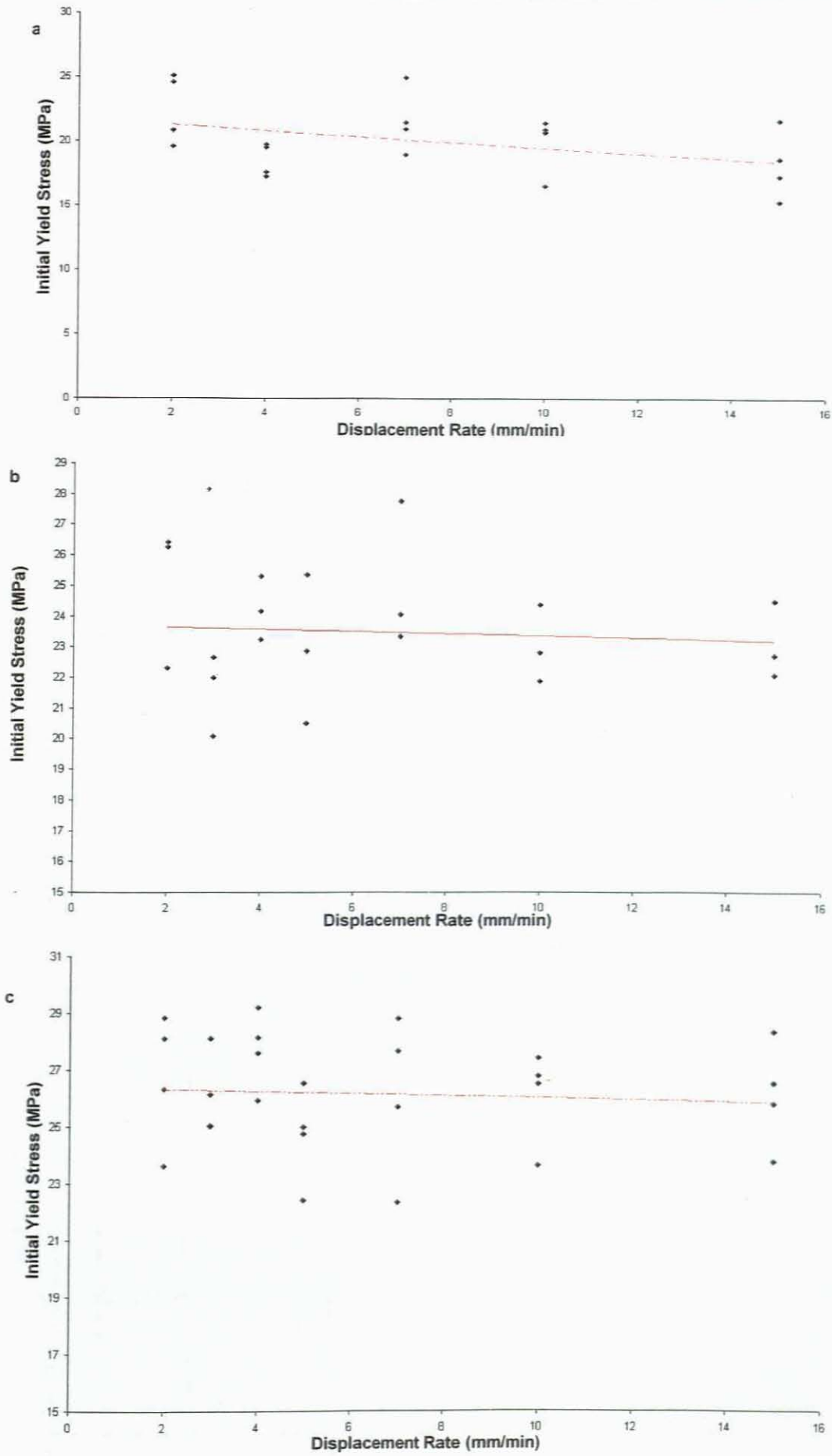


Figure 2.8: Effect of Displacement Rate on Initial Yield Stress for (a) 1mm, (b) 2mm and (c) 3mm diameter NiTi Shape Memory Alloy Wire.

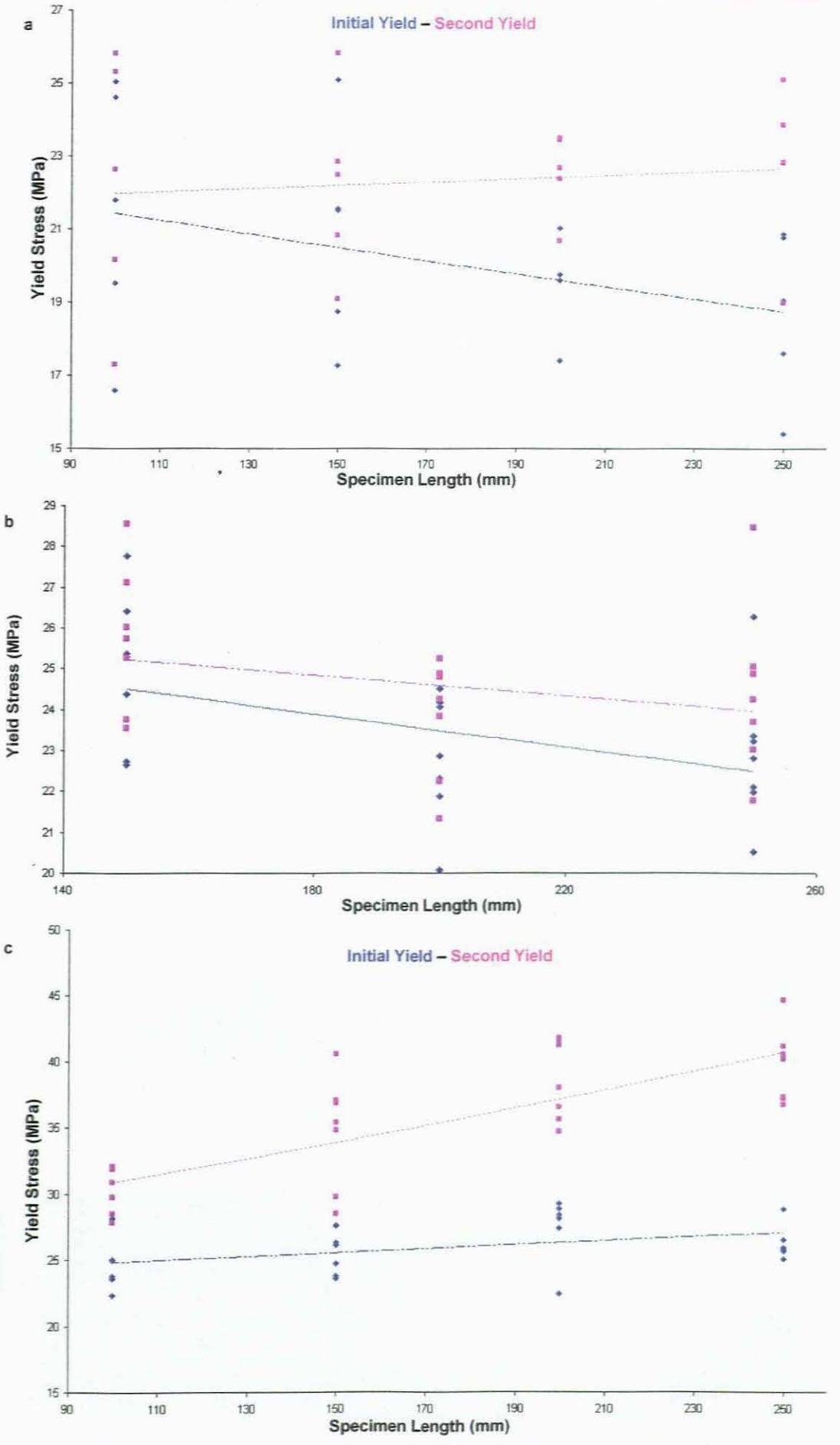


Figure 2.9: Effect of Specimen Length on Initial and Second Yield Stresses for (a) 1mm, (b) 2mm and (c) 3mm diameter NiTi Shape Memory Alloy Wire

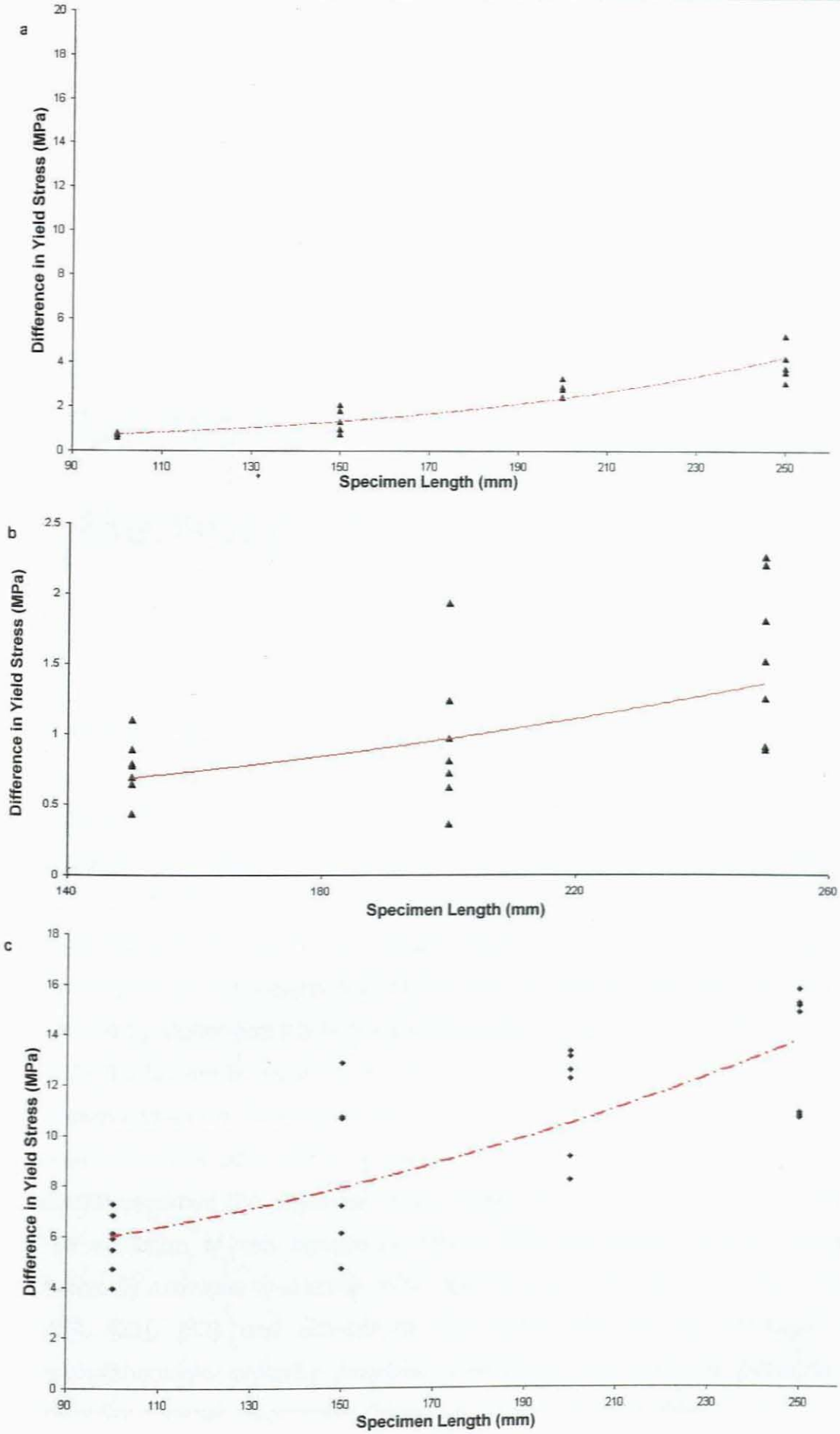


Figure 2.10: Effect of Specimen Length on the Difference in Yield Stress for (a) 1mm, (b) 2mm and (c) 3mm diameter NiTi Shape Memory Alloys Wire.

Chapter 3

Constitutive Model for Shape Memory Alloy Line Actuators

3.1 Introduction

This section describes the Multi-Well or Triple Well constitutive approach to simulate the complex material response, i.e., the shape memory effect and the pseudo-elastic effect, exhibited by shape memory alloy materials. The idea behind this particular shape memory alloy model was originally developed by Achenbach and Müller and Achenbach and was then further refined by Müller and Seelecke and Seelecke and Kastner [40-42]. The model finds it's footing firmly in the foundations of Thermodynamics and Statistical Thermodynamics. The model is particularly appropriate to study the case of a shape memory alloy element subjected to uniaxial loading conditions. The model assumes the existence of an austenitic phase fraction and describes the evolution of two martensitic phase fractions based on the theory of thermally activated processes. What follows is a brief overview gathered from [20], [21], [32] and [39-42] of the steps followed to formulate this comprehensible uniquely powerful constitutive approach to simulate the complex material responses exhibited by shape memory alloys.

3.2 Mechanical Properties of the Model

The basic element of the model is the lattice particle, i.e., a small piece of metallic lattice, shown in Fig. 3.1. The figure shows the three equilibrium configurations (M_+ and M_- for the martensitic twins and A for the austenite) that a shape memory alloy line actuator can find itself in. Form this description it is easy to see that the two martensitic variants are sheared versions of the austenite with the shear lengths, $\Delta = \pm J$. Other shear lengths are however also possible.

The specific potential energy, Π, of the lattice particle corresponds to a given shear length. Fig. 3.1 also shows the postulated form of that specific potential energy as a train of three convex parabolas. It is characterised by two stable minima for the martensitic twins and a metastable minimum for the austenite. Analytically this given by

$$\Pi_\alpha(\Delta) = \begin{cases} \frac{1}{2}E_M(\Delta - J)^2 \\ \frac{1}{2}E_M\Delta^2 + \phi_0 \\ \frac{1}{2}E_M(\Delta + J)^2 \end{cases} \qquad (3.1)$$

The intersection points for these minima are the potential barriers given by $\Delta = \pm\Delta_S$. If the lattice particle is subjected to a shear load P, the potential energy of the load must be added. This is a linearly decreasing function of Δ and therefore the potential energy of the particle becomes distorted as shown in Fig. 3.2. The right minimum becomes deeper and the left minimum becomes shallower. The potential barriers also change their heights due to this distortion.

In order to construct a model for the line actuator as a whole, the lattice particles are arranged in layers to form a stack of N such layers thus representing a single crystal tensile specimen. The layers are arranged at an angle, θ, to the direction of the tensile load according to the crystallographic axes of the specimen (see Fig. 3.3a). This angle is chosen as 45^0 so that the layers experience the maximum shear load. On the left of Fig. 3.3a we see the body at low temperature, in the martensitic state. It has alternating layers of the two martensitic variants, M_+ and M_- at 50% each. This defines the natural state of the body that has an original length of L_0. The application of a tensile load will subject the layers to a shear load. As seen in the figure, the M_+ layers become flatter while the M_- layers will become steeper.

The vertical component of the shear lengths, $\frac{\Delta_i}{\sqrt{2}}$ (summed over all N layers) will provide the total deformation of the body and is given by

$$D = L - L_0 = \frac{1}{\sqrt{2}} \sum_{\Delta} \Delta N_{\Delta} \qquad (3.2)$$

N_{Δ} is the number of layers with the shear length Δ and the summation extends over all possible shear lengths. Unloading lets the specimen fall back to the natural state so that a small deformation is elastic. If the load is increased however, there comes a point where the M_- layers will flip over and become M_+ layers (see Fig. 3.3a at load F_2). The flipped layers provide a large shear length and consequently, the deformation increases drastically. Upon subsequent unloading, the layers settle into the M_+ minimum so that a large deformation remains.

Fig. 3.3b is a schematic representation of the load versus deformation behaviour of the shape memory alloy layers. It is considered that while the specimen is loaded between 0 and F_1, its behaviour is still purely elastic. As

F_2 is reached however, the layers are preparing to flip. F_2 thus represents our yield load and flipping or de-twinning, as we shall call it from now on, commences at this constant load. This is also the region where we observe the greatest deformation. The region between 0 and F_1 signifies the initial elastic region and F_2 the initial yield stress. It is noted that not all the layers flip at once but instead go one after the other. This then simulates the inhomogeneous deformation during the transformation region as observed in the experimental investigation of Chapter 2.

When all the M_- layers have de-twinned to M_+ we see a linear relationship of the load versus deformation behaviour of the M_+ layers. Unloading from this point, F_3, shows some degree of elastic unloading and the deformation that remains appear to be plastic. The behaviour described above is purely mechanical and ssubsequent heating will cause a phase transformation to the austenitic phase and the lattice layers straighten up and accordingly the specimen contracts to its original length. This is seen on the far right hand side of Fig. 3.3a. At this point the specimen has the outward shape of the natural state, but internally it is different (more symmetric/ordered austenitic phase). The internal and external shape recovery is completed by a cooling phase where at a certain point the austenite becomes unstable and the layers revert back to their martensitic state having equal proportions of M_+ and M_- that the specimen started out with, i.e., the original zig-zag internal structure of alternating martensitic variants. This then simulates the shape memory effect exhibited by shape memory alloys.

At high temperatures however the model assumes an initial austenitic neutral state (see Fig. 3.4a and b). If the model body is now subjected to a uniaxial load while in the region 0 and F_2, it will perform elastically. When F_2 is reached, certain austenitic layers will transform to M_+ variants and induce the apparent macroscopic permanent deformation. Again these austenitic layers flip or de-twin one at a time and a macroscopic inhomogeneous deformation

is observed. Once all these layers have transformed to M_+ variants, the model body exhibits the elastic behaviour of these variants up to F_3.

Unloading from this region will now guide the model body through a hysteretic path. When F_1 is now reached the M_+ variants of martensite will now transform back to austenite. When this transformation is complete the model body now follows the initial elastic path back to 0 and the model body reassumes its original internal and external properties.

We can conclude that the model is able to simulate:

The shape memory effect through

- The initial elastic deformation from the low temperature neutral state,
- The yield through de-twinning,
- The residual deformation, and
- The shape recovery upon heating by conversion of the layers to austenite..

The pseudo-elastic material response through

- The initial elastic deformation from the high temperature neutral state,
- The yield through de-twinning,
- Elastic deformation after complete de-twinning
- Hysteretic unloading path, and
- The complete high temperature shape recovery.

The mechanical features of this model are easily interpreted. The thermally induced phase transformations however require thermodynamic arguments, which will now be discussed.

3.3 Thermodynamic Properties of the Model

Any change in the thermodynamic state of a system is ascribed to four thermodynamic potentials. These thermodynamic potentials are useful in the chemical thermodynamics of reactions and non-cyclic processes. They are internal energy, e, enthalpy, η, the Helmholtz free energy, ψ, and the Gibbs free energy, φ. The Helmholtz and Gibbs free energies are respectively defined as:

$$\psi = e - T\chi, \tag{3.3}$$

and

$$\varphi = \eta - T\chi \tag{3.4}$$

where $\eta = e - PD$, χ is the entropy, and T is the temperature. For this model description, the internal energy considers potential energy of the lattice structure as well as the kinetic energy of their random motion and is given as

$$e = \sum_{\Delta} \Pi(\Delta) N_{\Delta} + A(T - T_R) + e_R \tag{3.5}$$

where e_R is the energy of the reference state at $T = T_R$. Since thermal fluctuations are taken into account the model deformation can no longer be described by energetic considerations only and entropic effects has to be considered due to its importance with an increase in temperature. The relevant thermodynamic potential is given by the Gibbs free energy of (3.4) and the entropy term is given by a term for the entropy of thermal fluctuation, an entropy constant given for the reference state, and the Boltzmann formula. Seelecke and Müller [42] now give the Gibbs free energy in the form

$$\varphi = \sum_{\Delta}\left(\Pi(\Delta) - P\Delta + kT\ln\frac{N_\Delta}{N}\right)N_\Delta + A(T - T_R) + U_R - T\left(A\ln\frac{T}{T_R} + \chi_R\right) \tag{3.6}$$

and the equilibrium conditions for a given temperature and applied load occurs when the Gibbs free energy is a minimum. Since our model body consists of layers that may belong to different metallurgical phases, which will be denoted by α, partial equilibrium states may exist. Under these conditions one might find that these metallurgical phases are in equilibrium for a given external load P and the temperature, T, of an external heat bath but the number of layers of metallurgical phases, N_α, might not be dictated by the minimum of the Gibbs free energy as given by (3.6). This is quite plausible since thermal and dynamic equilibrium establish themselves quickly, whereas the process of obtaining phase equilibrium is considered slower because potential barriers have to be overcome. This is particularly true for solid bodies, where the thermal activation energy, kT, is low compared to potential energy barriers. The Gibbs free energy for each particular phase, α, in partial equilibrium is thus given by

$$\varphi_\alpha = -N_\alpha kT \sum_{\Delta\in\alpha} \exp-\left(\frac{\Pi_\alpha(\Delta) - P\Delta}{kT}\right) + A_\alpha(T - T_R) + U_R^\alpha - T\left(A_\alpha \ln\frac{T}{T_R} + S_R^\alpha\right) \tag{3.7}$$

Partial equilibrium conditions dictate that fluctuation of the layers occurs only in the immediate neighbourhood of the different potential minima. The sums in (7) can then be approximated by integrating over all possible shear lengths, Δ, from $-\infty$ to $+\infty$. Considering Π_α given by (3.1), the specific Gibbs free energy, $\hat{\varphi}_\alpha = \frac{\varphi_\alpha}{N_\alpha}$, now reduces to

$$\hat{\varphi}_\alpha(P,T)=\begin{cases} -\dfrac{P^2}{2E_M}-PJ+\lambda_\alpha \\ -\dfrac{P^2}{2E_A}+\lambda_\alpha \\ -\dfrac{P^2}{2E_M}+PJ+\lambda_\alpha \end{cases} \tag{3.8}$$

where

$$\lambda_\alpha = c_\alpha(T-T_R)+\underbrace{\left(u_R^\alpha+\frac{1}{2}kT_R\right)}_{=:\varepsilon_\alpha} - T\left(c_\alpha \ln\frac{T}{T_R}+\underbrace{\hat{\chi}_R^\alpha-\ln\left(\frac{1}{Y}\sqrt{\frac{E_\alpha}{2\pi kT}}\right)}_{=:\eta_\alpha}\right) \tag{3.9}$$

the specific heat of each metallurgical phase, c_α, is introduced in (3.9) and from thermodynamics the specific deformation, $d_\alpha=\frac{D_\alpha}{N_\alpha}$, and the specific entropy, $\hat{\chi}_\alpha=\frac{\chi_\alpha}{N_\alpha}$, is obtained from the Gibbs free energy. We thus have

$$d_\alpha=\begin{cases} \dfrac{P}{E_M}+J \\ \dfrac{P}{E_A} \\ \dfrac{P}{E_M}-J \end{cases} \tag{3.10}$$

and

$$\hat{\chi}_\alpha = c_\alpha \ln\frac{T}{T_R}+\eta_\alpha \tag{3.11}$$

The specific Helmholtz free energies and the specific enthalpies of each metallurgical phase may now be determined from $\hat{\psi}_\alpha(T,P) = \hat{\varphi}_\alpha + P d_\alpha$ and $\hat{h}_\alpha = \hat{\varphi}_\alpha + T\hat{\chi}_\alpha$, and given as

$$\hat{\psi}_\alpha(d_\alpha, T) = \begin{cases} \frac{E_M}{2}(d_{M+} - J)^2 + \lambda_\alpha \\ \frac{E_A}{2} d_A{}^2 + \lambda_\alpha \\ \frac{E_M}{2}(d_{M-} + J)^2 + \lambda_\alpha \end{cases} \tag{3.12}$$

where λ_α is given by (9) and

$$\hat{h}_\alpha(P,T) = \begin{cases} -\frac{P^2}{2E_M} - PJ + c_{M+}(T - T_R) + \varepsilon_{M+} \\ -\frac{P^2}{2E_A} + c_A(T - T_R) + \varepsilon_A \\ -\frac{P^2}{2E_M} + PJ + c_{M-}(T - T_R) + \varepsilon_{M-} \end{cases} \tag{3.13}$$

It should be noted that apart from the temperature dependant terms, the specific Helmholtz free energies, $\hat{\psi}_\alpha$, depend on d_α by the same functions as the potential energies, Π_α, depend on Δ. Since the specific deformation $d = \frac{D}{N}$ of the entire model body is the weighted sum of the specific deformation of each of the metallurgical phases, d_α, and the phase proportions, $\xi_\alpha = \frac{N_\alpha}{N}$, given as weighting factors we may write the total specific deformation as

$$d = \sum_\alpha \xi_\alpha d_\alpha \tag{3.14}$$

and similarly this additive decomposition can be applied to the other thermodynamic potentials reveals the following

$$\hat{\varphi} = \sum_{\alpha} \xi_{\alpha} \hat{\varphi}_{\alpha}, \qquad \hat{\chi} = \sum_{\alpha} \xi_{\alpha} \hat{\chi}_{\alpha}$$
$$\hat{\psi} = \sum_{\alpha} \xi_{\alpha} \hat{\psi}_{\alpha}, \qquad \hat{h} = \sum_{\alpha} \xi_{\alpha} \hat{h}_{\alpha} \tag{3.15}$$

3.4 The Effective Potential Energy of the Model Body

Seelecke, Müller and Seelecke now modified this constitutive approach under the premise that because of the existence of thermal fluctuations, there will always be a few layers even while in partial equilibrium that will have enough kinetic energy to overcome the potential barriers. These layers will be able to sample the energies of a neighbouring potential well and they will thus settle so as to permit the body to lower its Gibbs free energy. These researchers now made the assumption that in an average sense a layer moves in an effective potential energy field, which is given by Gibbs free energy, or, in the absence of a load, the Helmholtz free energy of the form given by (3.12).

Acceptance of this conjecture now lead to a smoothing out of the free energy function in (3.12) by connecting the three convex parabola of Fig. 3.1 continuously and differentiably by two concave parabola of the form $a\Delta^2 + b_{\pm}\Delta + c$. The specific Helmholtz free energy $\hat{\psi}(\Delta, T)$ is thus given by

$$\hat{\psi}(\Delta, T) = \begin{cases} \frac{E_A}{2}\Delta^2 + \phi_0 + \beta_{\alpha} & |\Delta| \le \Delta_L \\ a\Delta^2 + b_{\pm}\Delta + c & \Delta_L \le \Delta \le \Delta_R \\ \frac{E_M}{2}(\Delta \mp J)^2 + \beta_{\alpha} & \Delta_R \le \pm\Delta \end{cases} \tag{3.16}$$

with

$$\beta_\alpha = c_\alpha(T - T_R) + \varepsilon_\alpha - T\left(c_\alpha \ln\frac{T}{T_R} + \eta_\alpha\right) \tag{3.17}$$

the horizontal coordinates of the points where the parabola merge are denoted by $\pm\Delta_R$ and $\pm\Delta_L$. If J, E_A and E_M are known, the parameters a, $b_\pm$ and c, of the concave parabola may be given as

$$a = -\frac{E_M(J - \Delta_R) + E_A\Delta_L}{2(\Delta_R - \Delta_L)} \tag{3.18}$$

$$b_\pm = \pm\frac{E_M(J - \Delta_R) + E_A\Delta_R}{(\Delta_R - \Delta_L)}\Delta_L \tag{3.19}$$

$$\begin{aligned} c = {} & \frac{1}{2}E_M J^2 - \frac{E_M(J - \Delta_L) + E_A\Delta_L}{2(\Delta_R - \Delta_L)}\Delta_R{}^2 + c_M(T - T_R) \\ & + \varepsilon_M - T\left(c_M \ln\frac{T}{T_R} + \eta_M\right) \end{aligned} \tag{3.20}$$

The fourth condition for the smooth connection of the parabola relates to the difference in the heights of the minima to Δ_R and Δ_L, i.e.,

$$\begin{aligned} & \phi_0 + (c_A - c_M)(T - T_R) + \varepsilon_A - \varepsilon_M - \\ & T\left((c_A - c_M)\ln\frac{T}{T_R} + \eta_A - \eta_M\right) = \frac{1}{2}E_M J^2 + \\ & \frac{1}{2}(E_M - E_A)\Delta_R\Delta_L - \frac{1}{2}E_M J(\Delta_R + \Delta_L) \end{aligned} \tag{3.21}$$

The connection points, $\pm\Delta_R$ and $\pm\Delta_L$, as defined in eqs. 3.16 – 3.21 are dependant on the temperature of the body. Fig. 3.5 a, b, c shows this dependence as well as the temperature dependence of the heights of the minima of the convex and concave parabola. These graphs are drawn under the assumption that

$$\varepsilon_M - T\eta_M > \varepsilon_A - T\eta_A \tag{3.22}$$

At high temperatures the minimum potential energy lies with the austenitic well at $\Delta = 0$. For intermediate temperature three minima is apparent, i.e., one for austenite and one for each of the two variants of martensite. When the temperature is below the shape memory alloy's transition temperature, the austenitic minima ceases to be relevant and the martensitic wells themselves are connected by a single concave parabola. This situation arises at a temperature where $\pm\Delta_L(T) = 0$. This temperature is the martensitic start M_S temperature.

Under a load, the effective potential energy is given by the Gibbs free, $\hat{\varphi}(\Delta, T) = \hat{\psi}(\Delta, T) - P\Delta$ (see Fig. 3.6). This then shows that under a load, the potential energy may exhibit a barrier at high temperature. This barrier may be eliminated by the loads

$$P_L = E_A \Delta_L(T) \tag{3.23}$$

and

$$P_R = E_M(\Delta_R(T) - J) \tag{3.24}$$

These two loads are identified with the pseudo-elastic yield and recovery loads in a tensile test.

3.5 Phase transition of the model

Phase transitions in the model is governed by the rate laws for the phase fractions $\xi_{M_\pm}$, ξ_A and is given by

$$\dot{\xi}_{M_-} = -\left(\overset{-A}{\rho} + \overset{A-}{\rho}\right)\xi_{M_-} - \overset{A+}{\rho}\,\xi_{M_+} + \overset{A-}{\rho} \tag{3.25}$$

and

$$\dot{\xi}_{M_+} = -\overset{A+}{\rho}\,\xi_{M_-} - \left(\overset{+A}{\rho} + \overset{A+}{\rho}\right)\xi_{M_+} + \overset{A+}{\rho} \tag{3.26}$$

under the constraint $1 = \xi_{M+} + \xi_{M_-} + \xi_A$. For a typical transformation, the rate of change of ξ_{M_-} consists of a loss, due to layers that jump from M_- to A, and a gain due to layers that jump in the reverse direction. The number of jumps is proportional to the phase fractions in the phase of origin.

The quantities $\overset{\alpha\beta}{\rho}$ in (3.25) and (3.26) are the transition probabilities from phase α to phase β and are calculated using statistical thermodynamics. They are determined by the Boltzman factor corresponding to $\hat{\psi}(\Delta_B, T) - P\Delta_B$ where Δ_B are shear lengths at the barriers of the effective potential energy (see Fig. 3.6). These quantities may then be given as

$$\overset{-A}{\rho} = \frac{1}{\tau}\frac{e^{-\frac{\hat{\psi}\left(\Delta_B^L, T\right) - P\Delta_B^L}{kT}}}{\sum_{\Delta \in M_-} e^{-\frac{\hat{\psi}(\Delta, T) - P\Delta}{kT}}} \tag{3.27}$$

or

$$\overset{A+}{\rho} = \frac{1}{\tau}\frac{e^{-\frac{\hat{\psi}\left(\Delta_B^R,T\right)-P\Delta_B^R}{kT}}}{\sum_{\Delta\in M_-} e^{-\frac{\hat{\psi}(\Delta,T)-P\Delta}{kT}}} \tag{3.28}$$

so that, for example, $\overset{-A}{\rho}$ is proportional to the fraction of M_- layers that have the energy of the left barrier. The probabilities $\overset{A+}{\rho}$, and $\overset{A-}{\rho}$, $\overset{+A}{\rho}$ are constructed in the same way with obvious changes appropriate to the different phases of origin and different barriers. The idea behind (3.27) and (3.28) is that once a layer has the energy of the barrier, it may pass over it. The relaxation time of this process is denoted by τ, and needs to be adjusted to yield rate of recovery rate as observed in an experiment. T represents the temperature of the model body and initially we will assume it to be a constant. The system of ordinary differential equations in time (3.25) and (3.26) permits the calculation of phase fractions for a given initial conditions provided the temperature, loading condition, and environmental temperature are known. The total specific deformation of the body is given by the sum of deformation of all the layers (derived from (3.10)) and is given by

$$d = d_A + \xi_{M_+}\left(d_{M_+} - d_A\right) + \xi_{M_-}\left(d_{M_-} - d_A\right) \tag{3.29}$$

(3.29) can also be considered the load-deformation relation which is complemented by the system of non-linear ordinary differential equations for the martensitic phase fractions.

3.6 Numerical Implementation of the Model

This section is concluded with numerical simulations to illustrate the models capability to reproduce the quasi-plastic material response exhibited by shape memory alloy specimens observed in the experimental investigation of Chapter 2. The preceding section shows that the mathematical structure of the model is given by the system of non-linear ordinary differential equations

in time (3.25) and (3.26) and the algebraic relation (3.29). It was observed that this system exhibits extreme numerical stiffness. To this end an implicit method was employed to obtain a stable solution [58]. A FORTRAN program was constructed that uses the RADAU5 routine for solution of the ODE's. This routine was developed by Harrier and Wanner and uses an implicit Runga-Kutta solution scheme of order five with step size control and continuous output. This routine can be used for the numerical solution of a stiff system of first order ordinary differential equations. This system can be implicit or explicit. An example of this code is given in Appendix C.

Since the model assumes an equal amount of martensitic variants (50% of each) as an initial condition, Fig. 3.8a shows the evolution of the phase transformation as the given load is applied (see Fig. 3.7). It shows how the ξ_{M_+} variant increases to 100% and the ξ_{M_-} variant decreases to 0%. The characteristic s-shaped curve of transformation is also observed in Fig. 3.8a.

The computational variables (see Table 3.1) were obtained from experimental data (from Chapter 2) for a NiTi shape memory alloy rod of 3mm diameter and a length of 200mm. The load-extension data was then plotted with the results of the numerical simulation. These results compared favourably as seen in Fig. 3.8b. Although there is some variation in the initial loading region, the model shows its appropriateness in handling the shape memory effect observed in 3mm diameter NiTi shape memory alloy rods.

Computational Variables	*Symbol*	*Value*
Interfacial Energy	IE	169.0
Total Transformation Strain	J	1.83E-3
Potential Energy at Barrier	Psi	170.0
Potential Energy of Austenitic Well	Psi0	To Be Calculated
Elastic Stiffness of Austenite	Ea	291.80E6
Elastic Stiffness of Martensite	Em	119.53E6
Transformation Strain at Barrier	deltaS	0.0 (Below Transition Temperature)
Transformation Strain at Left Smoothing Point	deltaL	0.0 (Below Transition Temperatre)
Transformation Strain at Right Smoothing Point	deltaR	0.267E-3
Boltzmann Constant	k	1.3804E-23
Absolute Temperature	Temp	298
Transformation Volume	V	17.710E-23
Rate of Transformation	Omega	19.5E-1
Reference Temparature	Tr	298
Reference Free Energy of Martensite	Epm	0.0
Reference Free Energy of Austenite	Epa	12.375
Specific Heat of Martensite	SHM	395.0
Specfic Heat of Astenite	SHA	395.0
Reference Entropy of Martensite	ENM	0.0
Reference Entropy of Austenite	ENA	57.84
Smoothing Parameters of Effective Potential Energy	a, b, c	To Be Calculated
Phase Fraction of Martensite +	ξ^{M+}	To Be Calculated
Phase Fraction of Martensite -	ξ^{M-}	To Be Calculated
Phase Fraction of Austenite	ξ^{A}	To Be Calculated

Table 3.1: Computational variables used in numerical simulations

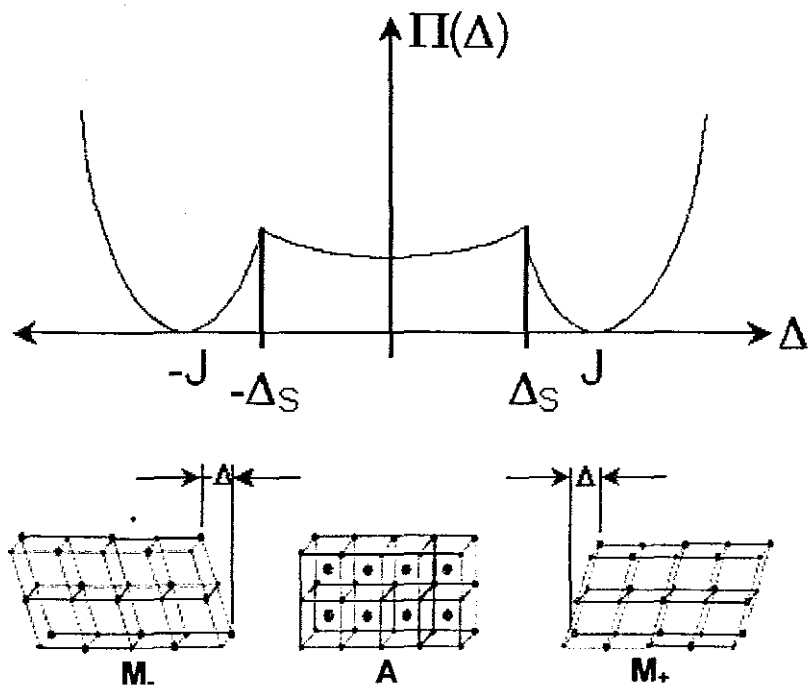


Figure. 3.1: Three configuration of a lattice particle (M_+, M_- and A) and the postulated form of the potential energy

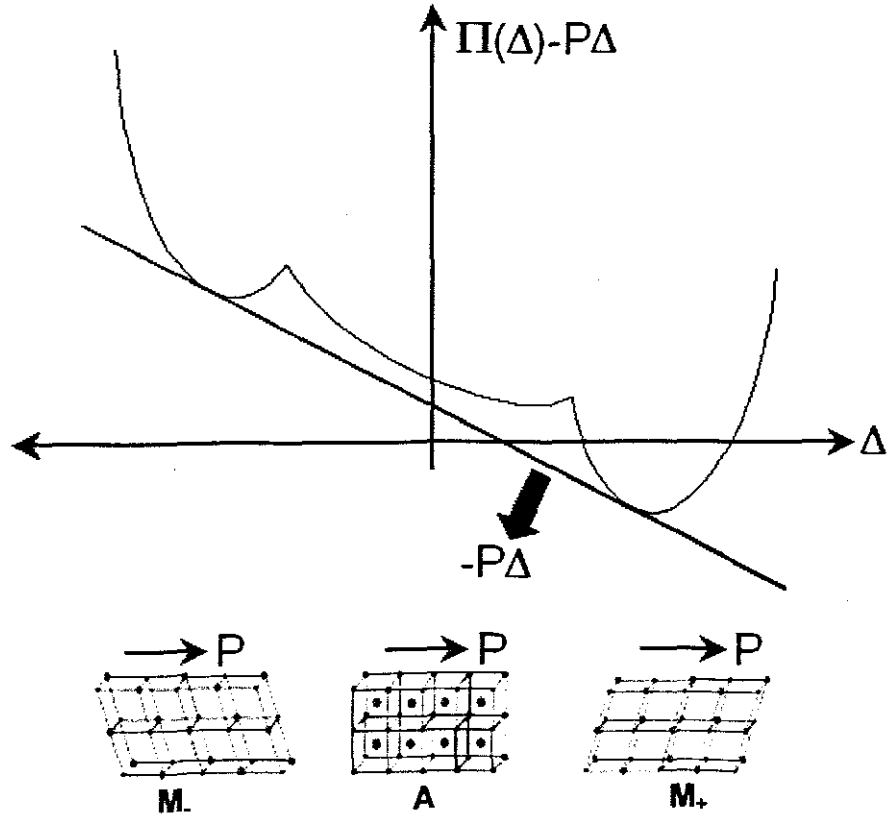
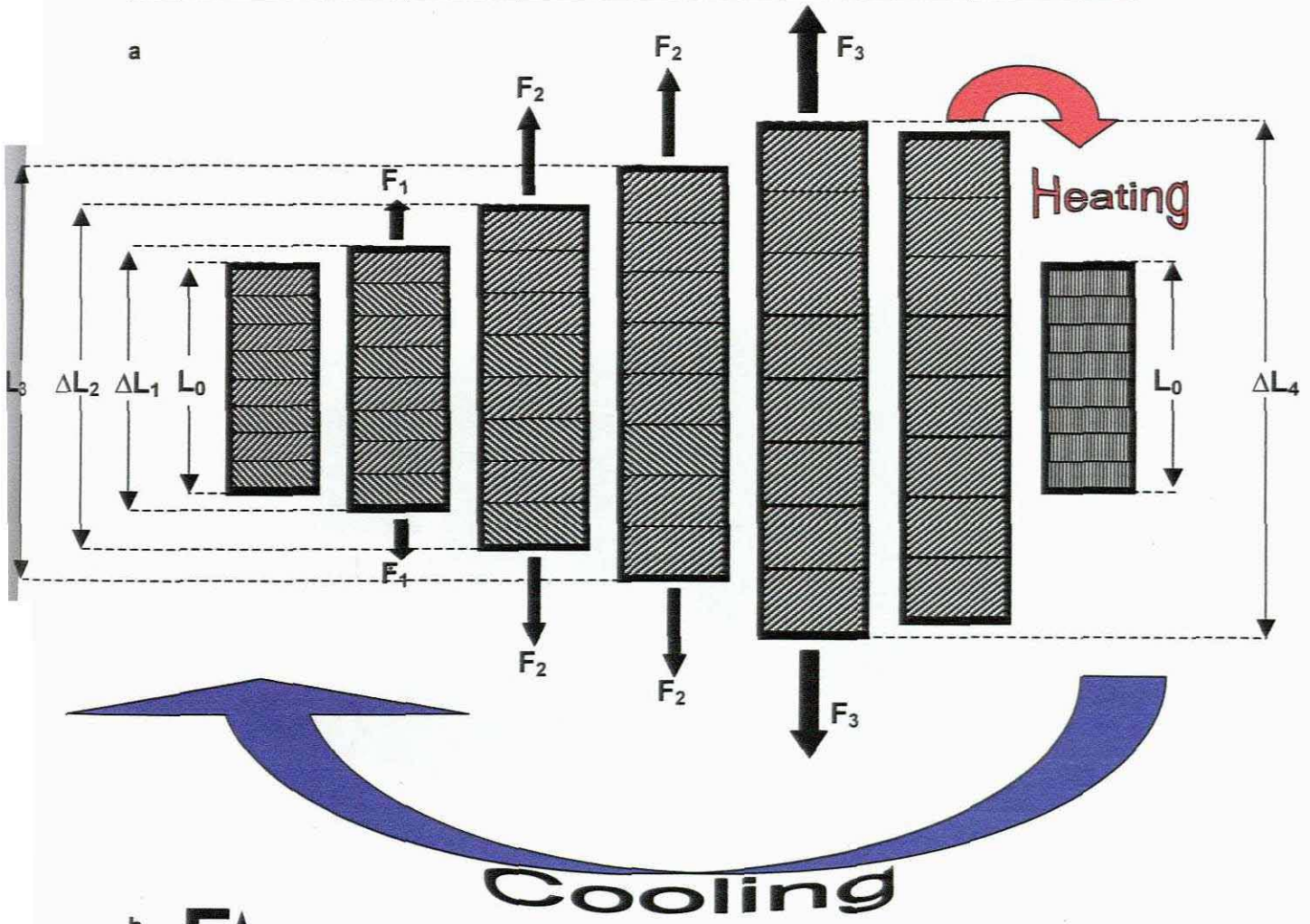


Figure. 3.2: Lattice particle subjected to shear load and distorted potential energy



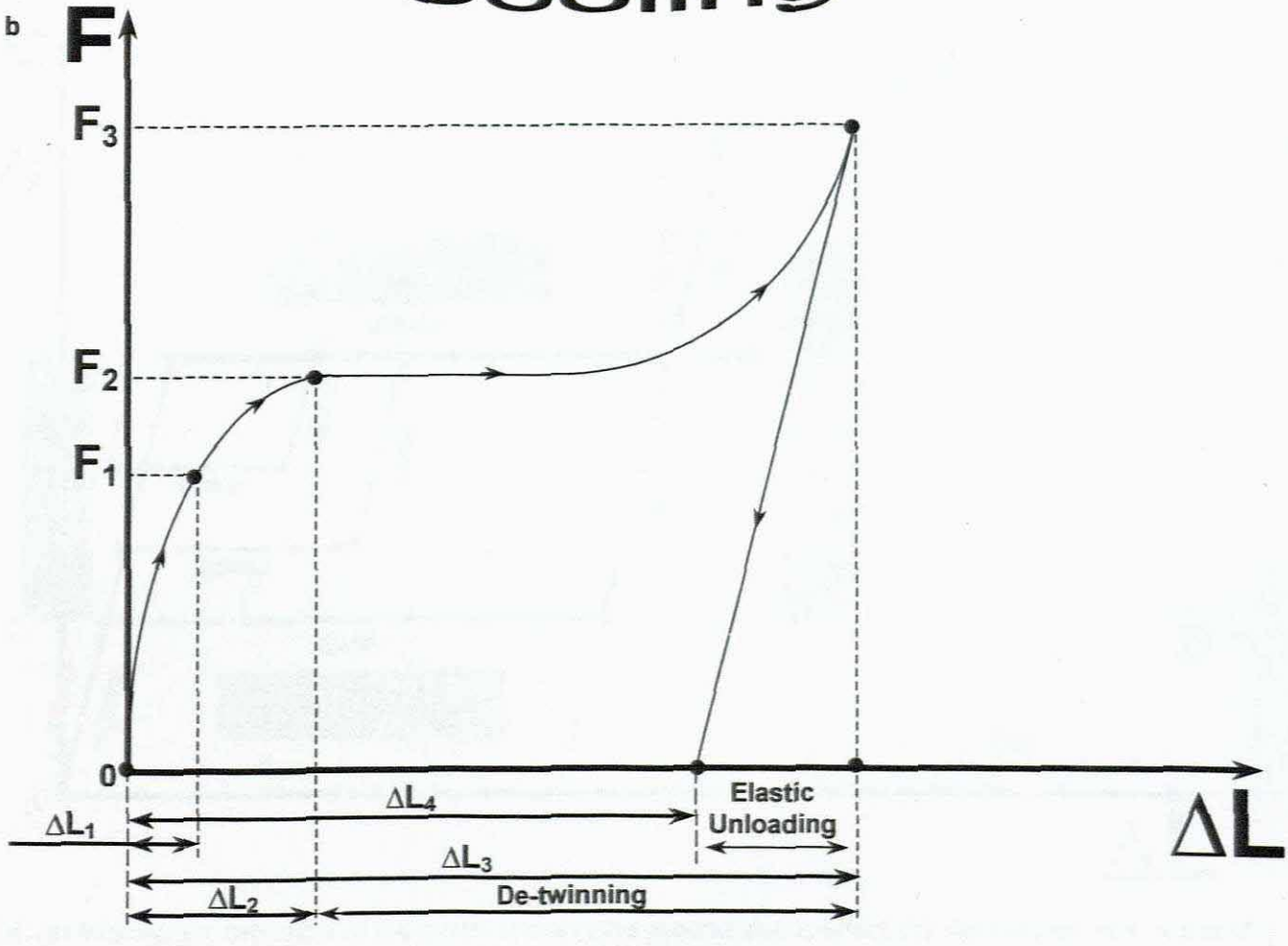


Figure 3.3: (a) Microscopic behaviour of constitutive model showing the shape memory effect. (b) Macroscopic description of the constitutive model corresponding to (a).

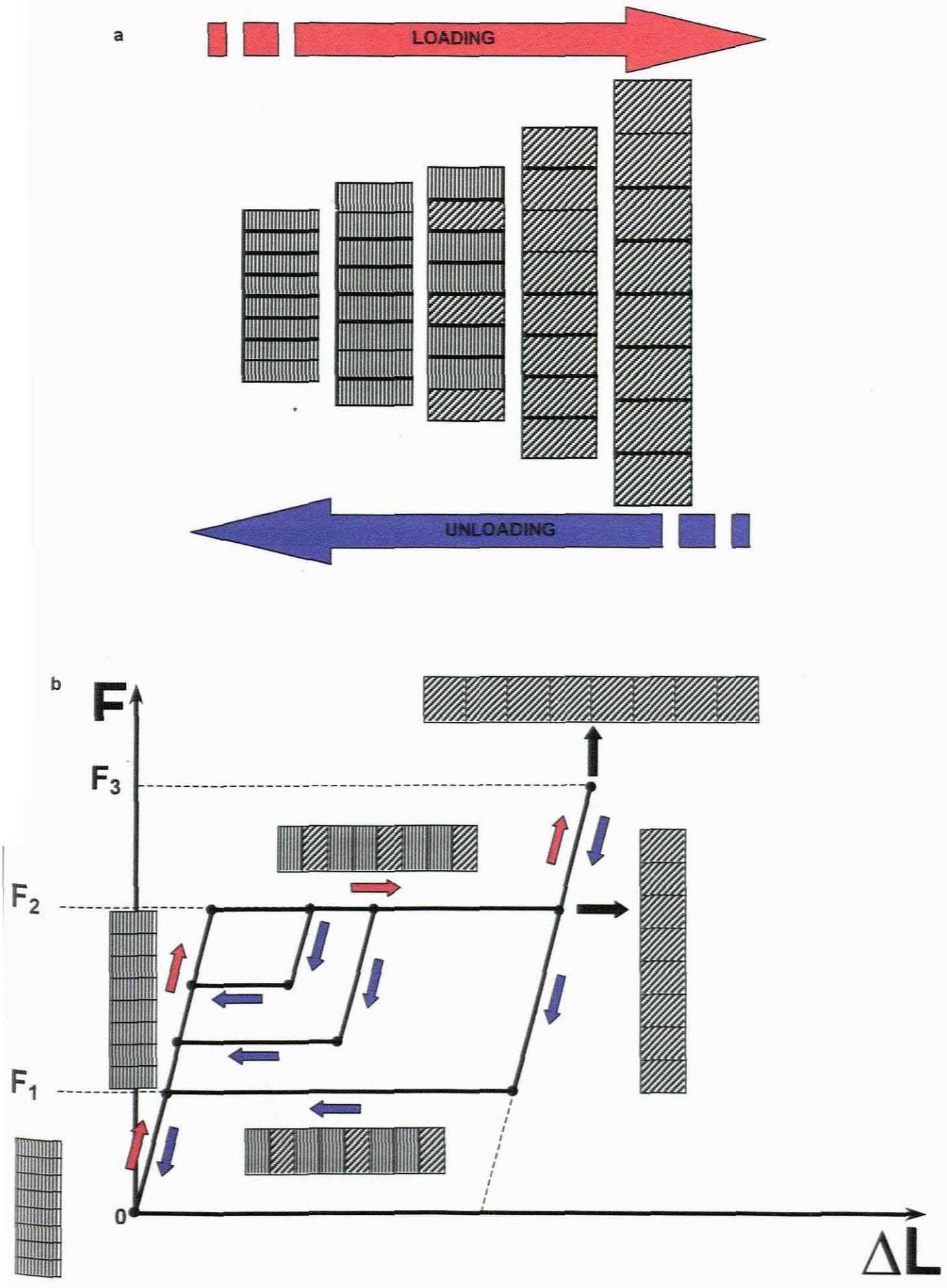


Figure 3.4: (a) Microscopic behaviour of the model showing the pseudo-elastic effect. (b) Macroscopic description of the model corresponding to (a).

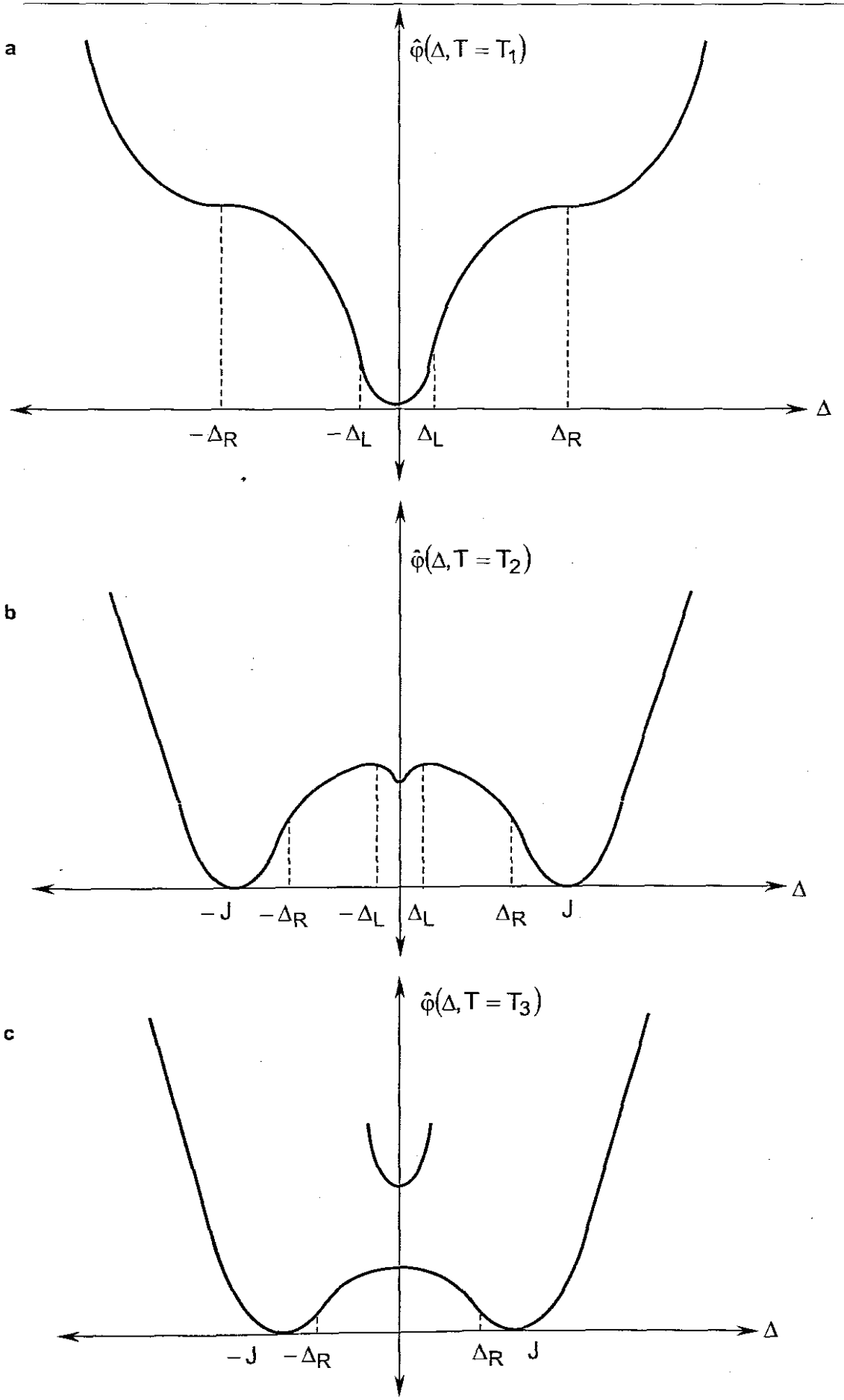


Figure 3.5: Helmholtz free energy of a layer at three different temperatures: (a) High, (b) Intermediate, and (c) Low temperature.

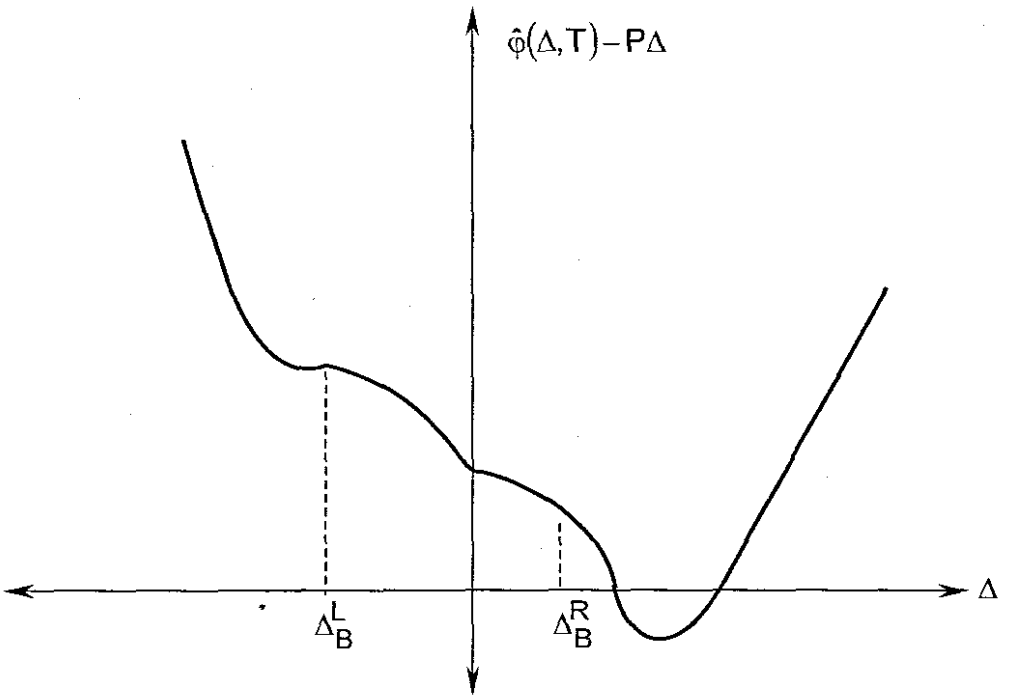


Figure 3.6: Gibbs free energy of a layer under a load at an intermediate temperature.

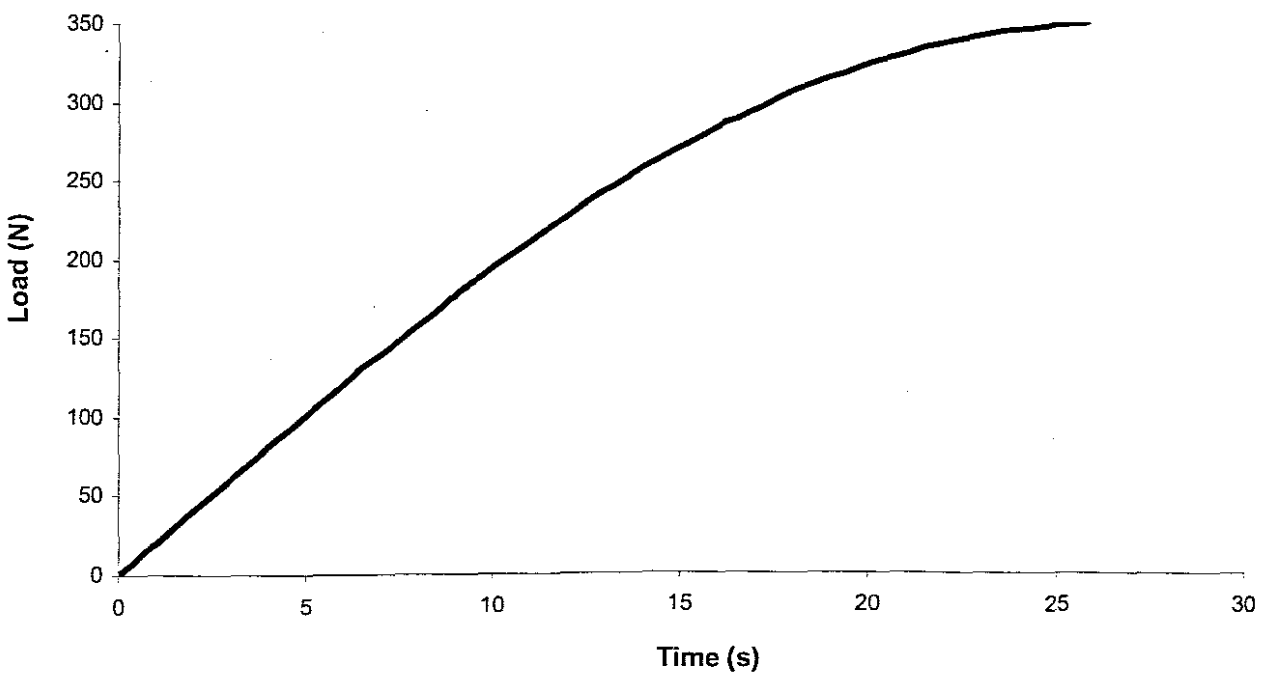


Figure 3.7: Load-Time relation used for model simulation.

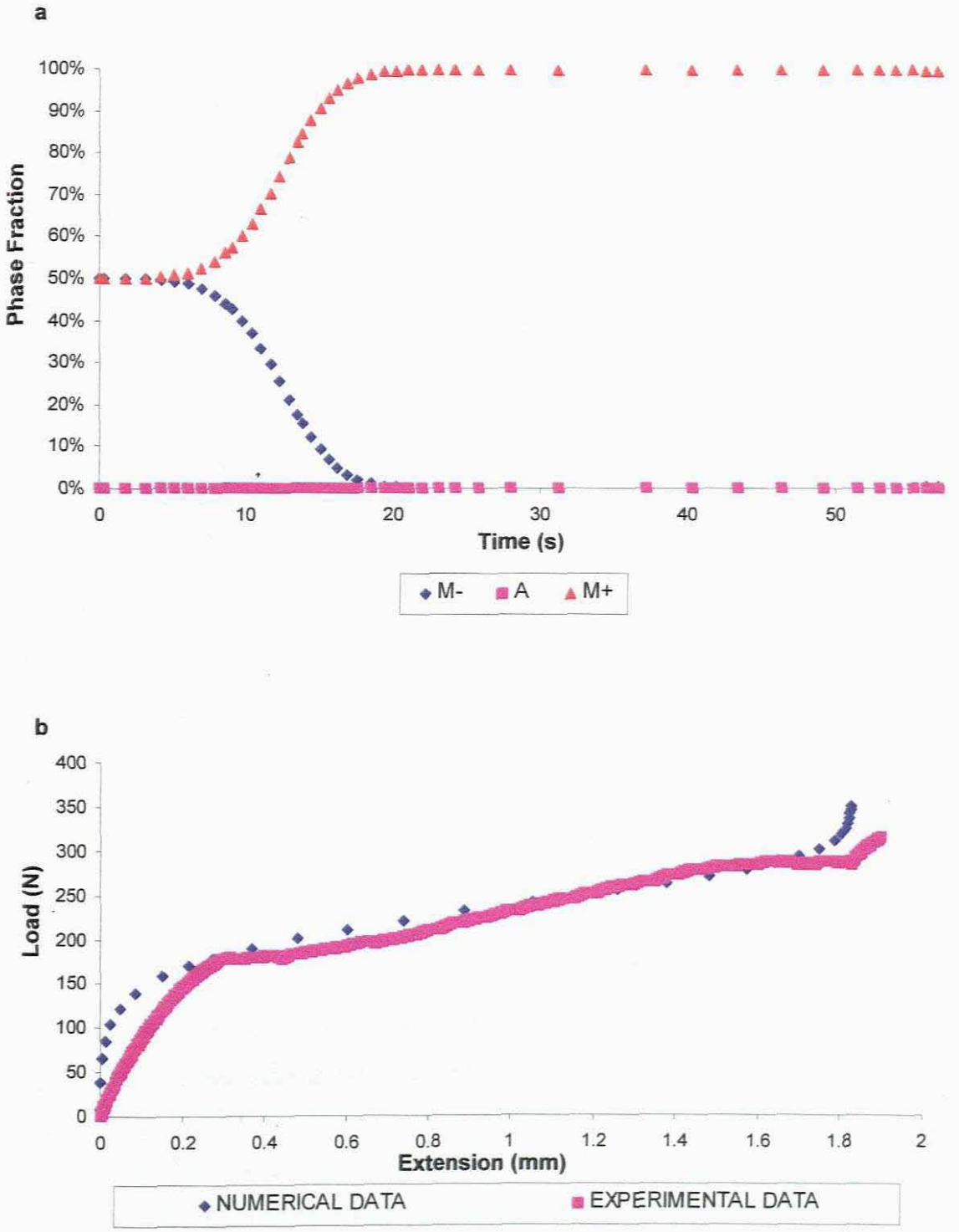


Figure 3.8: (a) Numerical interpretation of load induced martensitic phase transformation during quasi-plastic material response. (b) Mechanical load cycle of a NiTi shape memory alloy wire specimen of 3mm diameter and length 200mm showing numerical fit.

Chapter 4

Finite Element Formulation for a Shape Memory Alloy Truss (SMAT) Element

In the development of a Finite Element formulation for our shape memory alloy element, we will use the Total Lagrangian formulation as presented by Bathe [59], [60], to describe its unique material behaviour. This type of analysis describes large displacements, large rotations, and large strains and uses the Second Piola-Kirchhoff stress measure, which is work-conjugate with the Green-Lagrange strain measure.

4.1 Incremental Finite Element Equation

In a non-linear analysis such as ours, the main objective is to find the state of equilibrium of our shape memory alloy body that corresponds to a given externally applied load. These externally applied loads are functions of time and the equilibrium conditions of the system of finite elements used for our shape memory alloy body under consideration is given by

$$^{t}R - {}^{t}F = 0 \qquad (4.1)$$

where ^{t}R and ^{t}F represent the vector of externally applied nodal forces and nodal point forces corresponding to element stresses in the configuration at time, t, respectively. The relation given by (4.1) must satisfy equilibrium of the shape memory alloy body in the current deformed configuration, taking

cognisance of all material non-linearities. In the solution of the non-linear response, it should be recognised that (4.1) must be satisfied throughout the complete history of the applied load, i.e., t can take on any value ranging from zero to a maximum value. In our static analysis with material time effects caused by either temperature or load induced phase transformations, the time variable should be considered as an actual variable to be properly included in the modelling of the physical situation.

An incremental step-by-step solution will be used to calculate the analysis results. This approach requires the solution for the discrete time, t, to be known, while we require the solution for the discrete time, $t+\Delta t$, where Δt, represents a suitably chosen time increment. At time $t+\Delta t$, 4.1 is given by

$$^{t+\Delta t}\mathbf{R}-{}^{t+\Delta t}\mathbf{F}=\mathbf{0} \tag{4.2}$$

with $^{t+\Delta t}\mathbf{R}$ being independent of deformations. Since the solution is known at time t, $^{t+\Delta t}\mathbf{F}$ is given by

$$^{t+\Delta t}\mathbf{F}={}^{t}\mathbf{F}+\mathbf{F} \tag{4.3}$$

where $\mathbf{F}$ represents an increment of nodal point forces corresponding to the increment in element displacements and stresses from time t to time $t+\Delta t$. It can be approximated using a tangent stiffness matrix, $^{t}\mathbf{K}$, which corresponds to the geometric and material conditions of our shape memory alloy at time t and is given by

$$\mathbf{F}={}^{t}\mathbf{K}\mathbf{U} \tag{4.4}$$

where $\mathbf{U}$ represents a vector of incremental nodal displacements and the tangent stiffness matrix, $^{t}\mathbf{K}$, is calculated by differentiating the internal element nodal point forces, $^{t}\mathbf{F}$, with respect to the nodal point displacements, $^{t}\mathbf{U}$. Substituting these relations into (4.2) produces

$$^{t+\Delta t}\mathbf{R} - {}^{t+\Delta t}\mathbf{F} = {}^{t}\mathbf{K}\mathbf{U}. \tag{4.5}$$

The solution of $\mathbf{U}$ in (4.5), calculates an approximation to the displacements at time $t+\Delta t$, i.e.,

$$^{t+\Delta t}\mathbf{U} = {}^{t}\mathbf{U} + \mathbf{U} \tag{4.6}$$

The exact displacements at time $t+\Delta t$, corresponds to the externally applied load, $^{t+\Delta t}\mathbf{R}$. It should be noted that an approximation of these displacements are calculated in (4.6) since (4.4) was used. The assumptions used in (4.4) may be subject to significant calculation errors, and depending on time and load step size used, it may even become unstable. To counter this, an iterative method, the Newton-Raphson technique will be employed to smooth the calculation. This method is an extension of the incremental technique given in (4.5) and (4.6), i.e., having calculated an increment in the nodal point displacements that defines the new total displacement vector, we can repeat the incremental solution presented above using the currently known total displacements instead of the displacements at time t. The equations used for the Newton-Raphson iteration are for $i = 1,2,3,\ldots$,

$$^{t+\Delta t}\mathbf{K}^{(i-1)}\,\Delta\mathbf{U}^{(i)} = {}^{t+\Delta t}\mathbf{R} + {}^{t+\Delta t}\mathbf{F}^{(i-1)} \tag{4.7}$$

and

$$^{t+\Delta t}\mathbf{U}^{(i)} = {}^{t+\Delta t}\mathbf{U}^{(i-1)} + \Delta\mathbf{U}^{(i)} \tag{4.8}$$

with the initial conditions

$$^{t+\Delta t}\mathbf{U}^{(0)} = {}^{t}\mathbf{U}; \qquad ^{t+\Delta t}\mathbf{K}^{(0)} = {}^{t}\mathbf{K}; \qquad ^{t+\Delta t}\mathbf{F}^{(0)} = {}^{t}\mathbf{F} \tag{4.9}$$

In the first iteration, (4.8) and (4.9), reduces the equations to (4.5) and (4.6). In subsequent iterations, the latest estimates for the nodal point

displacements are used to evaluate the corresponding element stresses and nodal point forces, ${}^{t+\Delta t}F^{(i-1)}$, and the tangent stiffness matrix, ${}^{t+\Delta t}K^{(i-1)}$. The out-of-balance load vector, ${}^{t+\Delta t}R - {}^{t+\Delta t}F^{(i-1)}$, correspond to a load vector that is not yet balanced by element stresses, and hence an increment in the nodal point displacements is required. This updating of the nodal point displacements in the iteration is continued until the out-of-balance loads and incremental displacements are very small.

4.2 Continuum-Mechanics-Based Approach

A consistent continuum-mechanics-based approach is used to develop the governing finite element equation to be used for the simulation of our non-linear shape memory alloy material response. The continuum mechanics equation is developed for a displacement-based finite element solution. The principle of virtual work is used and is formulated for large displacement, large rotation, and large strain with a non-linear constitutive material response.

The Lagrangian description of body motion is used here, i.e., the particles of our shape memory alloy body will be followed in a stationary Cartesian coordinate system from an original configuration at time $t = 0$ to some final configuration at time $t = t + \Delta t$ (See Fig. 4.1).

The aim is to evaluate the equilibrium positions of the complete body at discrete time points 0, Δt, $2\Delta t$, $3\Delta t$, ..., where Δt is an increment in time. The solution strategy assumes that the solutions for the static and kinematic variables for all time steps from time 0 to t, inclusive, have been obtained. The solution process for the next required equilibrium position corresponding to time $t + \Delta t$ is typical and is applied repetitively until the complete path has been solved for.

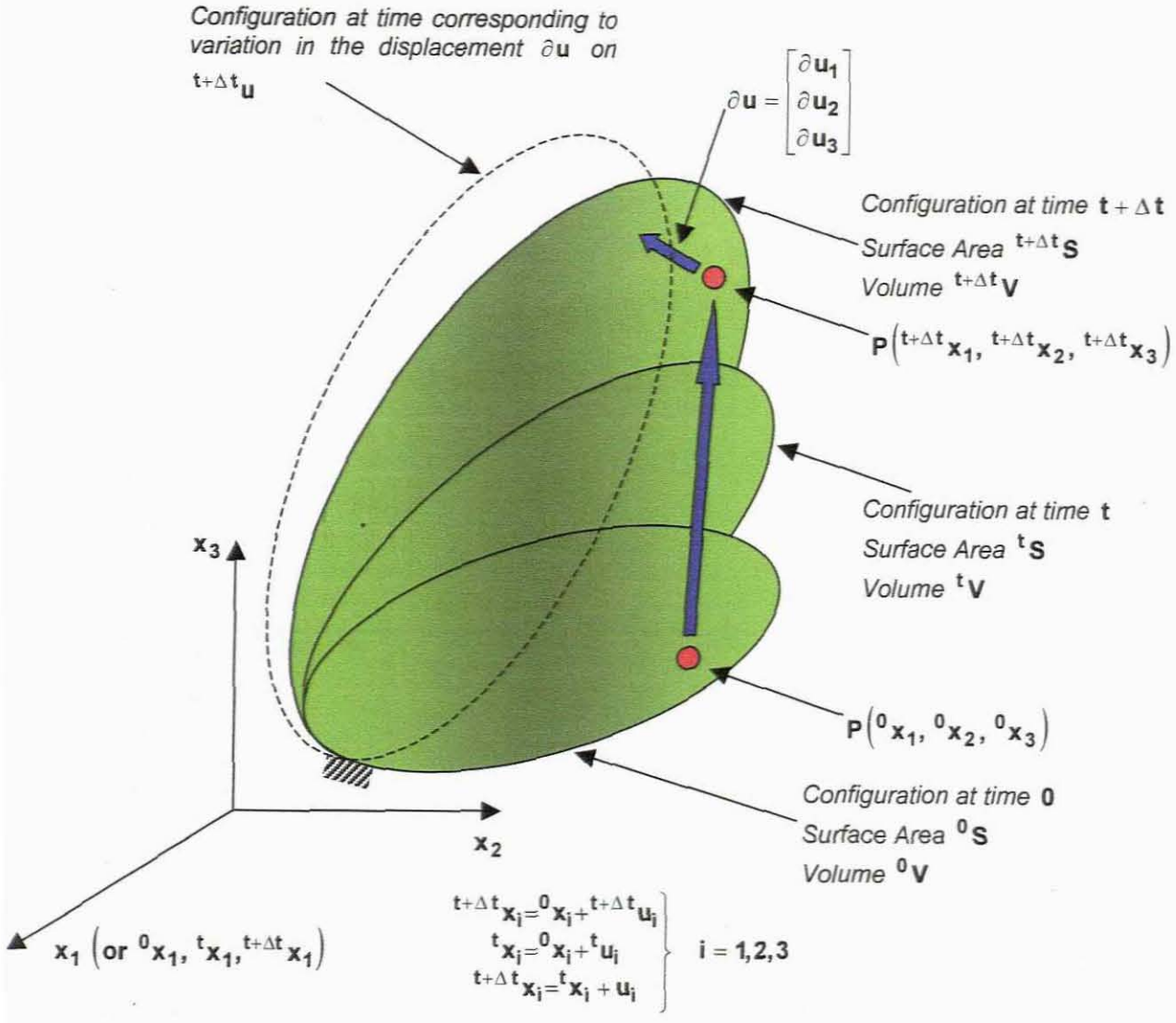


Figure 4.1: Motion of a body in a Stationary Cartesian Coordinate Frame.

4.3 Relevant Notation, Kinematic Descriptions and Strain-Stress Measures.

A key point in presenting our continuum level finite element equations to simulate the non-linear shape memory alloy material responses lies in the successful representation of all relevant state variables which will be measured in a stationary Cartesian coordinate system (See Fig. 4.1). A summary of the notation used in this work is presented next.

The coordinates of a generic point P in the body at time 0 are $\left({}^{0}x_1, {}^{0}x_2, {}^{0}x_3\right)$; at time t they are $\left({}^{t}x_1, {}^{t}x_2, {}^{t}x_3\right)$; and at time $t+\Delta t$ they are $\left({}^{t+\Delta t}x_1, {}^{t+\Delta t}x_2, {}^{t+\Delta t}x_3\right)$. The left superscript denotes the configuration that the body finds itself in while the right subscript represents the coordinate axis. The notation for the displacements is given in an analogous manner to the coordinates: at time t the displacement is ${}^{t}u_i, i = 1, 2, 3$, and at time $t+\Delta t$ the displacement is ${}^{t+\Delta t}u_i, i = 1, 2, 3$. We then have the following relations

$$\left.\begin{aligned} {}^{t}x_i &= {}^{0}x_i + {}^{t}u_i \\ {}^{t+\Delta t}x_i &= {}^{0}x_i + {}^{t+\Delta t}u_i \end{aligned}\right\} \quad i = 1, 2, 3. \tag{4.10}$$

with incremental representation of the displacement from time t to $t+\Delta t$ given by

$$u_i = {}^{t+\Delta t}u_i - {}^{t}u \tag{4.11}$$

The motion of the body causes continuous changes in its specific mass, area, and volume and thus for our body these values are given respectively for times 0, t, and $t+\Delta t$ is given as ${}^{0}\rho$, ${}^{t}\rho$, ${}^{t+\Delta t}\rho$, ${}^{0}A$, ${}^{t}A$, ${}^{t+\Delta t}A$, ${}^{0}V$, ${}^{t}V$, and ${}^{t+\Delta t}V$. The configuration of the body at the time $t+\Delta t$ is unknown and therefore we will refer applied forces, stresses measures, and strain measures to a known equilibrium configuration. In analogy to the notation used for coordinates and displacements a left superscript indicates in which configuration the quantity occurs with a left subscript indicating the configuration with respect to which the quantity is measured. If for example G is a tensor representing a given quantity and finds itself in the configuration at time $t+\Delta t$, but will be measured in the configuration at time 0 it will be given as ${}^{t+\Delta t}_{0}G$.

The strain measure with value to the finite element analysis is the Green-Lagrange Strain Tensor, ${}^{t}_{0}\varepsilon$. All derivatives of this strain measure are taken

with respect to the initial coordinates of a particular body in question. It is therefore said the Green-Lagrange Strain Tensor is defined with respect to the initial coordinates of the body. In this formulation we will use the Second Piola-Kirchoff Stress Tensor, ${}^{t}_{0}\mathbf{S}$, as the work-conjugate to the Green-Lagrange Strain Tensor. Table 4.1 summarizes all the relevant stress and strain measures that will be used in our continuum level non-linear finite element formulation.

${}^{t}\mathbf{x}^{T} = \begin{bmatrix} {}^{t}x_1 & {}^{t}x_3 & {}^{t}x_3 \end{bmatrix}$	Coordinates of a particle at time t
${}_{0}\nabla^{T} = \begin{bmatrix} \frac{\partial}{\partial {}^{0}x_1} & \frac{\partial}{\partial {}^{0}x_1} & \frac{\partial}{\partial {}^{0}x_1} \end{bmatrix}^{T}$	Gradient Operator
${}^{t}_{0}\mathbf{X} = \left({}_{0}\nabla\ {}^{t}\mathbf{x}^{T}\right)^{T}$	Deformation Gradient
${}^{t}\mathbf{J} = \det\left({}^{t}_{0}\mathbf{X}\right)$	Jacobian Determinant / Volume Ratio
${}^{t}\rho = {}^{0}\rho / \det\left({}^{t}_{0}\mathbf{X}\right)$	Mass Density of the Body at Time t
${}^{t}_{0}\mathbf{C} = {}^{t}_{0}\mathbf{X}^{T}\,{}^{t}_{0}\mathbf{X}$	Right Cauchy-Green Deformation Tensor / Green Deformation Tensor
${}^{t}_{0}\mathbf{B} = {}^{t}_{0}\mathbf{X}\,{}^{t}_{0}\mathbf{X}^{T}$	Left Cauchy-Green Deformation Tensor / Piola Deformation Tensor
${}^{t}_{0}\mathbf{R}$	Orthogonal (Rotation) Tensor
${}^{t}_{0}\mathbf{U}$	Symmetric Stretch Tensor
${}^{t}_{0}\mathbf{X} = {}^{t}_{0}\mathbf{R}\,{}^{t}_{0}\mathbf{U}$	Polar Decomposition of ${}^{t}_{0}\mathbf{X}$ ${}^{t}_{0}\mathbf{U}^{2} = {}^{t}_{0}\mathbf{C}$ ${}^{t}_{0}\mathbf{U} = {}^{t}_{0}\mathbf{U}^{T}$ ${}^{t}_{0}\mathbf{R}^{T}\,{}^{t}_{0}\mathbf{R} = \mathbf{I}$
${}^{t}_{0}\mathbf{D}$	Velocity Strain Tensor
${}^{t}_{0}\mathbf{W}$	Spin Tensor
${}^{t}_{0}\mathbf{L} = {}^{t}_{0}\mathbf{D} + {}^{t}_{0}\mathbf{W}$	Velocity Gradient
${}^{t}_{0}\varepsilon = \frac{1}{2}\left({}^{t}_{0}\mathbf{C} - \mathbf{I}\right)$	Green-Lagrange Strain Tensor
${}^{t}_{0}\dot{\varepsilon} = {}^{t}_{0}\mathbf{X}^{T}\,{}^{t}_{0}\mathbf{D}\,{}^{t}_{0}\mathbf{X}$	Rate of change of Green-Lagrange Strain Tensor
${}^{t}\tau$	Cauchy Stress Tensor
${}^{t}_{0}\mathbf{S} = \frac{{}^{0}\rho}{{}^{t}\rho}\,{}^{t}_{0}\mathbf{X}\ {}^{t}\tau\ {}^{t}_{0}\mathbf{X}^{T}$	Second Piola-Kirchoff Stress Tensor

Table 4.1: Summary of Continuum Level Stress and Strain Measures

4.4 Continuum Mechanics Incremental Total Lagrangian Formulation

The solution approach to effective incremental analysis of non-linear problems relies heavily on the selection of appropriate stress and strain measures that will be employed. The equation that we want to solve should express the equilibrium and compatibility requirement of our shape memory alloy body in the configuration at time $t+\Delta t$. This equation is generally given as

$$\int_{t+\Delta t_V} {}^{t+\Delta t}\tau_{ij}\,\delta_{t+\Delta t}e_{ij}\,d^{t+\Delta t}V = {}^{t+\Delta t}\wp \qquad (4.12)$$

where

${}^{t+\Delta t}\tau_{ij}$: Cartesian components of the Cauchy Stress Tensor

$\delta_{t+\Delta t}e_{ij}$: Strain Tensor corresponding to virtual work

${}^{t+\Delta t}V$: volume at time $t+\Delta t$

and

$${}^{t+\Delta t}\wp = \int_{t+\Delta t_V} {}^{t+\Delta t}f_i^B\,\delta u_i\,d^{t+\Delta t}V + \int_{t+\Delta t_{S_f}} {}^{t+\Delta t}f_i^S\,\delta u_i^S\,d^{t+\Delta t}S \qquad (4.13)$$

where

${}^{t+\Delta t}f_i^B$: components of externally applied forces per unit volume at time $t+\Delta t$

${}^{t+\Delta t}f_i^S$: components of externally applied surface tractions per unit surface area at time $t+\Delta t$

${}^{t+\Delta t}S_f$: surface at time $t+\Delta t$ on which external tractions are applied

$\delta u_i^S = \delta u_i$ evaluated on the surface ${}^{t+\Delta t}S_f$

In (4.12), the left-hand side represents the internal virtual work while the right-hand side represents the external virtual work and has the following equilibrium equations

Within ${}^{t+\Delta t}V$ for $i = 1, 2, 3,$

$$ {}^{t+\Delta t}\tau_{ij,j} + {}^{t+\Delta t}f_i^B = 0 \qquad \text{sum over } j = 1, 2, 3, \tag{4.14} $$

and on the surface ${}^{t+\Delta t}S_f$, for $i = 1, 2, 3,$

$$ {}^{t+\Delta t}\tau_{ij}\,{}^{t+\Delta t}n_j = {}^{t+\Delta t}f_i^S \qquad \text{sum over } j = 1, 2, 3, \tag{4.15} $$

where the ${}^{t+\Delta t}n_j$ are the components of the unit normal to the surface ${}^{t+\Delta t}S_f$ at time $t+\Delta t$.

Under the assumption that our body undergoes large displacement and large strains with non-linear constitutive relations, (4.12) cannot be solved directly. It can however be overcome by determining an approximate solution that refers all variable to a previously calculated equilibrium configuration and then linearizing the resulting equation. This solution may then be improved by iterations.

In developing this linearized equation, recall that the solutions for times 0, Δt, $2\Delta t$, $3\Delta t$, ..., t have been solved for already. The appropriate stress and strain measures can then be referred to any one of these known configurations. The Total Lagrangian Formulation is used in this analysis to develop the desired approximate solution. All static and kinematic variables are referred to the initial configuration at time 0 for this solution scheme. It also includes all kinematic non-linear effects due to large displacement, large rotation and large strains. The successful implementation of this solution scheme depends on the appropriate modelling of the particular material's constitutive description. Use of the Total Lagrangian formulation reduces (4.12) to the following form

$$ \int_{{}_0V} {}_0C_{ijrs}\,\delta\,{}_0e_{rs}\,d{}^0V + \int_{{}_0V} {}_0^tS_{ij}\,\delta\,{}_0q_{ij}\,d{}^0V = {}^{t+\Delta t}\wp - \int_{{}_0V} {}_0^tS_{ij}\,\delta\,{}_0e_{ij}\,d{}^0V \tag{4.16} $$

where ${}_0C_{ijrs}$ is the incremental stress-strain tensor at time t referred to the configuration at time 0. ${}^t_0S_{ij}$ is known at time t; and ${}_0e_{ij}$, ${}_0q_{ij}$ are the linear and non-linear strains, which are referred to the configuration at, time 0. Table 4.2 summarize the relations used to arrive at the linearized equation of motion for the state at time t in the Total Lagrangian formulation.

1. **Equation of Motion** $$\int_{{}^0V} {}^{t+\Delta t}_0S_{ij}\,\delta\,{}^{t+\Delta t}_0\varepsilon_{ij}\,dV^0 = {}^{t+\Delta t}\wp$$ *where* $${}^{t+\Delta t}_0S_{ij} = \frac{{}^0\rho}{{}^t\rho}\,{}^0_{t+\Delta t}x_{i,m}\,{}^t\tau_{mn}\,{}^0_{t+\Delta t}x_{j,n}\,; \qquad \delta\,{}^{t+\Delta t}_0\varepsilon_{ij} = \delta\frac{1}{2}\left({}^{t+\Delta t}_0u_{i,j}+{}^{t+\Delta t}_0u_{j,i}+{}^{t+\Delta t}_0u_{k,i}\,{}^{t+\Delta t}_0u_{k,j}\right)$$
2. **Incremental Decomposition** *(a) Stresses* $${}^{t+\Delta t}_0S_{ij} = {}^t_0S_{ij} + {}_0S_{ij}$$ *(b) Strains* $${}^{t+\Delta t}_0\varepsilon_{ij} = {}^t_0\varepsilon_{ij} + {}_0\varepsilon_{ij}\,; \qquad {}_0\varepsilon_{ij} = {}_0e_{ij} + {}_0q_{ij}$$ $${}_0e_{ij} = \frac{1}{2}\left({}_0u_{i,j}+{}_0u_{j,i}+{}^t_0u_{k,i}\,{}_0u_{k,j}+{}_0u_{k,i}\,{}^t_0u_{k,j}\right); \qquad {}_0q_{ij} = \frac{1}{2}\,{}_0u_{k,i}\,{}_0u_{k,j}$$
3. **Equation of Motion with Incremental Decomposition** (*with* ${}^{t+\Delta t}_0\varepsilon_{ij} = \delta\,{}_0\varepsilon_{ij}$) $$\int_{{}_0V} {}_0S_{ij}\,\delta\,{}_0\varepsilon_{ij}\,dV^0 + \int_{{}_0V} {}^t_0S_{ij}\,\delta\,{}_0q_{ij}\,dV^0 = {}^{t+\Delta t}\wp - \int_{{}_0V} {}^t_0S_{ij}\,\delta\,{}_0e_{ij}\,dV^0$$
4. **Linearization of Equation of Motion** (*with the approximation* ${}_0S_{ij} = {}_0C_{ijrs}\,{}_0e_{rs}$ *and* ${}^{t+\Delta t}_0\varepsilon_{ij} = \delta\,{}_0\varepsilon_{ij}$) $$\int_{{}_0V} {}_0C_{ijrs}\,\delta\,{}_0e_{rs}\,dV^0 + \int_{{}_0V} {}^t_0S_{ij}\,\delta\,{}_0q_{ij}\,dV^0 = {}^{t+\Delta t}\wp - \int_{{}_0V} {}^t_0S_{ij}\,\delta\,{}_0e_{ij}\,dV^0$$

Table 4.2: Continuum Mechanics incremental Decomposition of the Total Lagrangian Formulation

The relation in (4.16) is employed to calculate an increment in the displacement, which is then used to evaluate approximations to the displacements, strains and stresses corresponding to time $t+\Delta t$. The displacement approximations corresponding to time $t+\Delta t$ are obtained by adding the calculated increments to the displacements at time t. The strain approximations are evaluated from displacements using the kinematic

relations shown in Table 4.1. The calculations of stresses corresponding to time $t+\Delta t$ depend on the correct implementation of the constitutive relations to be employed.

If we now assume that the approximate displacements, strains, and stresses have been obtained, a check for the error between internal virtual work (evaluated using static and kinematic variables for time $t+\Delta t$) and external virtual work. Denoting approximate values with a right superscript (1), this error due to linearization is given by

$$\text{Error} = \int_{{}_0V} {}^{t+\Delta t}_{\ 0}S^{(1)}_{ij}\, \delta\, {}^{t+\Delta t}_{\ 0}\dot{\varepsilon}^{(1)}_{ij}\, d^0V \tag{4.17}$$

and noting that the right-hand side of (4.17) is equivalent to (4.16) (since ${}^{t+\Delta t}_{\ 0}\varepsilon_{ij} = \delta\, {}_0\varepsilon_{ij}$). In each case the current configurations with corresponding stress and strain variables are used. It shows that the right-hand side of (4.16) represent an out-of-balance virtual work prior to the calculations of increments in displacements whereas (4.17) represent an out-of-balance virtual work after the solution, which is due to the linearizations performed on (4.16).

In order to further reduce this out-of-balance virtual work we need to perform an iteration in which the above solution step is repeated until the difference between the external virtual work and internal virtual work is negligible within a certain convergence measure. A similar procedure to that used in section 4.1 is now employed to develop a Newton-Raphson iterative solution scheme for the above problem. The equation solved repetitively for $I = 1,2,3,\ldots$, is

$$\int_{{}_0V} {}_0C^{(I-1)}_{ijrs}\, \Delta\, {}_0e^{(I)}_{rs}\, dV^0 + \int_{{}_0V} {}^{t+\Delta t}_{\ 0}S^{(I-1)}_{ij}\, \delta\Delta\, {}_0q^{I}_{ij}\, dV^0 = {}^{t+\Delta t}\wp - \int_{{}_0V} {}^{t+\Delta t}_{\ 0}S^{(I-1)}_{ij}\, \delta\, {}^{t+\Delta t}_{\ 0}e^{(I-1)}_{ij}\, dV^0 \tag{4.18}$$

where in the case $I = 1$ (4.18) corresponds to (4.16) with displacements being updated as follows

$$ {}^{t+\Delta t}_{0}u_i^l = {}^{t+\Delta t}u_i^{(l-1)} + \Delta u_i^{(l)} \tag{4.19} $$

with initial condition

$$ {}^{t+\Delta t}u_i^{(0)} = {}^{t}u_i \tag{4.20} $$

4.5 Matrix Equations for a Displacement-Based Continuum Element used for Finite Element Analysis in Structural Analysis

The governing finite element matrices for an isoparametric continuum finite element with displacement degrees of freedom are presented here. The derivation of the governing finite element equations depend on the selection of the interpolation functions and the interpolation of the element coordinates and displacements with these functions in the governing continuum mechanics equations. We only consider a single element in this derivation since the governing equilibrium equation of an assemblage of these elements is easily constructed using the direct stiffness method.

Substituting element coordinates and displacement interpolation functions into the linearized equations derived in Table 4.2, we obtain (for either a single element or an assemblage of elements) for a static analysis using the Total Lagrangian Formulation the matrix equation of the following form

$$ \left({}^{t}_{0}K_L + {}^{t}_{0}K_{NL} \right) U = {}^{t+\Delta t}R + {}^{t}_{0}F \tag{4.21} $$

(4.21) considers non-linear constitutive material responses and assumes that externally applied loads are deformation-independent, i.e., the load vector for all load (time) steps are known before the incremental analysis is performed. Table 4.3 summarizes (for a single element) the integrals considered with their corresponding matrix evaluations.

Integral	Matrix Evaluation
$\int_{0V} {}_0C_{ijrs}\, {}_0e_{rs}\, \delta_0 e_{ij}\, d^0V$	${}^t_0K_L\hat{u} = \int_{0V} \left({}^t_0B_L^T\, {}_0C\, {}^t_0B_L\, d^0V \right)\hat{u}$
$\int_{0V} {}^t_0S_{ij}\, \delta_0 q_{ij}\, d^0V$	${}^t_0K_{NL}\hat{u} = \int_{0V} \left({}^t_0B_{NL}^T\, {}^t_0S\, {}^t_0B_{NL}\, d^0V \right)\hat{u}$
$\int_{0V} {}^t_0S_{ij}\, \delta_0 e_{ij}\, d^0V$	${}^t_0F = \int_{0V} {}^t_0B_L^T\, {}^t_0\hat{S}\, d^0V$

where

${}^t_0B_L$	*: linear strain-displacement transformation matrix*
${}^t_0B_{NL}$	*: non-linear strain-displacement transformation matrix*
${}_0C$	*: incremental stress-strain material properties*
${}^t_0S$	*: Matrix of Second Piola-Kirchoff stresses*
${}^t_0\hat{S}$	*: Vector of Second Piola-Kirchoff stresses*

Table 4.3: Finite Element Matrices

4.6 Constitutive Relations

The kinematic descriptions used the Total Lagrangian Formulation representing large displacement and large strain is for a general element. To accurately predict a desired material response, it is necessary to use the appropriate constitutive relations. In the Total Lagrangian Formulation it is assumed that the Second Piola-Kirchoff stress tensor is given in terms of the Green-Lagrange strain tensor and that we have

$$ {}^t_0S = \frac{\partial \psi}{\partial\, {}^t_0\varepsilon} \tag{4.22} $$

and

$$ {}_0C = \frac{\partial^2 \psi}{\partial\, {}^t_0\varepsilon^2} \tag{4.23} $$

where, for our current shape memory alloy material description, ψ, represents the total sum of the Helmholtz Potential of the body. This kind of constitutive description is often employed to model Hyperelastic material responses to simulate Rubber-like material behaviour. Examples of these models are the Mooney-Rivlin and Ogden models. For shape memory alloys, this approach has been used by Govindjee and Hall [17] and Auricchio and Sacco [2]. The Helmholtz Potential may be expressed as the sum of contributions of each phase present in a given volume of the spatial continuum [17]. It is given as

$$\hat{\psi} = \xi^{M+}\psi^{M+} + \xi^{A}\psi^{A} + \xi^{M-}\psi^{M-} \tag{4.24}$$

with the following fundamental restrictions applied to the volume fractions at every point in the spatial continuum [17]

$$\mathbf{1} = \xi^{M+} + \xi^{A} + \xi^{M-} \tag{4.25}$$

and

$$\mathbf{0} = \dot{\xi}^{M+} + \dot{\xi}^{A} + \dot{\xi}^{M-} \tag{4.26}$$

Using the description of the Helmholtz Potential given in section 3.3, we re-write these Helmholtz contributions to include thermal effects as

$$\psi^{M+} = \frac{{}_0C^{M+}}{2}\left({}_0^t\varepsilon - \varepsilon^{M+}\right)^2 - {}_0^t\varepsilon\alpha^{M+}(T-T_R) + c_{M+}(T-T_R) + \underbrace{\left(u_R^{M+} + \frac{1}{2}kT_R\right)}_{=:\varepsilon_{M+}}$$
$$-T\left(c_{M+}\ln\frac{T}{T_R} + \underbrace{\tilde{\chi}_R^{M+} - \ln\left(\frac{1}{Y}\sqrt{\frac{{}_0C_{M+}}{2\pi kT}}\right)}_{=:\eta_{M+}}\right) \tag{4.27}$$

$$\psi^{A} = \frac{{}_0C^{A}}{2}\left({}^t_0\varepsilon\right)^2 - {}^t_0\varepsilon\,\alpha^{A}\left(T - T_R\right) + c_A\left(T - T_R\right) + \underbrace{\left(u_R^{A} + \frac{1}{2}kT_R\right)}_{=:\varepsilon_A}$$
$$- T\left(c_A \ln\frac{T}{T_R} + \underbrace{\hat{\chi}_R^{A} - \ln\left(\frac{1}{Y}\sqrt{\frac{{}_0C_A}{2\pi kT}}\right)}_{=:\eta_A}\right) \tag{4.28}$$

$$\psi^{M-} = \frac{{}_0C^{M-}}{2}\left({}^t_0\varepsilon - \varepsilon^{M-}\right)^2 - {}^t_0\varepsilon\,\alpha^{M-}\left(T - T_R\right) + c_{M-}\left(T - T_R\right) + \underbrace{\left(u_R^{M-} + \frac{1}{2}kT_R\right)}_{=:\varepsilon_-}$$
$$- T\left(c_{M-} \ln\frac{T}{T_R} + \underbrace{\hat{\chi}_R^{M-} - \ln\left(\frac{1}{Y}\sqrt{\frac{{}_0C_{M-}}{2\pi kT}}\right)}_{=:\eta_{M-}}\right) \tag{4.29}$$

Using (4.22) and (4.23) and assuming $\alpha^{M} = \alpha^{M+} = \alpha^{M-}$, we find respectively the Second Piola-Kirchoff stress tensor and the effective material modulus tensor to be given by

$${}^t_0S = \xi^{A}\;{}_0C^{A}\left({}^t_0\varepsilon - \alpha^{A}\left(T - T_R\right)\right) + \xi^{M-}\;{}_0C^{M-}\,\varepsilon^{M-} - \xi^{M+}\;{}_0C^{M+}\,\varepsilon^{M+}$$
$$+ \left({}^t_0\varepsilon - \alpha^{M}\left(T - T_R\right)\right)\left(\xi^{M-}\;{}_0C^{M-} + \xi^{M+}\;{}_0C^{M+}\right) \tag{4.30}$$

and

$${}_0C = \xi^{A}\;{}_0C^{A} + \xi^{M-}\;{}_0C^{M-} + \xi^{M+}\;{}_0C^{M+} \tag{4.31}$$

4.7 Development of a Shape Memory Alloy Truss (SMAT) Element

We will now proceed with the derivation of the tangent stiffness matrix and force vector for our one-dimensional non-linear elastic, constant cross-sectional area Shape Memory Alloy Truss element (SMAT) that is subjected to large displacements and large strains. The truss element is formulated using an isoparametric formulation with the natural coordinate, s, attached as shown in Fig. 4.2(a). This element is subjected to tensile loads directed along

the axis of the SMAT element and applied at nodes 1 and 2. The nodal degrees of freedom are the local axial displacements represented by u_1 and u_2 at the ends of the each node of the SMAT element and associated to a global coordinate in the reference configuration ${}^0x_1$, ${}^0x_2$ and the current configuration ${}^tx_1$, ${}^tx_2$ (See Fig. 4.2(b)). The global coordinates of the nodal points of the SMAT element are at time 0, ${}^0x_1^k$, ${}^0x_2^k$ and at time t, ${}^tx_1^k$, ${}^tx_2^k$, where $k = 1,...,N$, with N equal to the number of nodes ($N = 2$). The nodal point coordinates are assumed to determine the spatial configuration of the SMAT element at time 0 and t using

$$ {}^0x_1(s) = \sum_{k=1}^{N} h_k \, {}^0x_1^k \; ; \; {}^0x_2(s) = \sum_{k=1}^{N} h_k \, {}^0x_2^k \qquad \textit{for reference configuration} \quad (4.32) $$

and

$$ {}^tx_1(s) = \sum_{k=1}^{N} h_k \, {}^tx_1^k \; ; \; {}^tx_2(s) = \sum_{k=1}^{N} h_k \, {}^tx_2^k \qquad \textit{for current configuration} \quad (4.33) $$

The interpolation functions h_k for the two nodal positions are $h_1 = \frac{1}{2}(1-s)$ and $h_2 = \frac{1}{2}(1+s)$ and using (4.32) and (4.33) it follows that

$$ {}^tu_i(s) = \sum_{k=1}^{N} h_k \, {}^tu_i^k \,, \qquad i = 1, 2 \qquad (4.34) $$

and

$$ u_i(s) = \sum_{k=1}^{N} h_k u_i^k \,, \qquad i = 1, 2 \qquad (4.35) $$

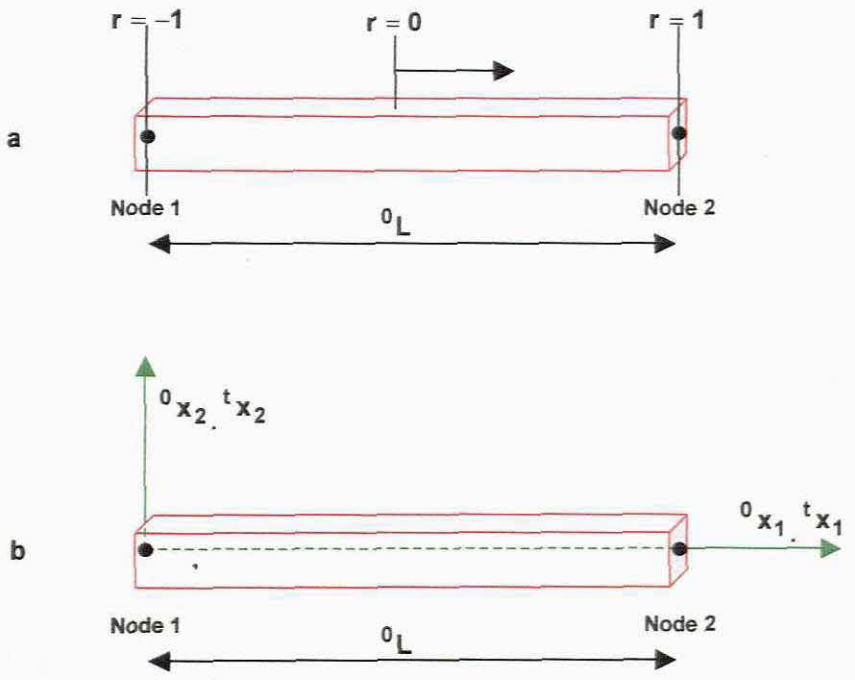


Figure 4.2: Two-node Shape Memory Alloy Truss Element in (a) natural coordinate system and in (b) global coordinate system

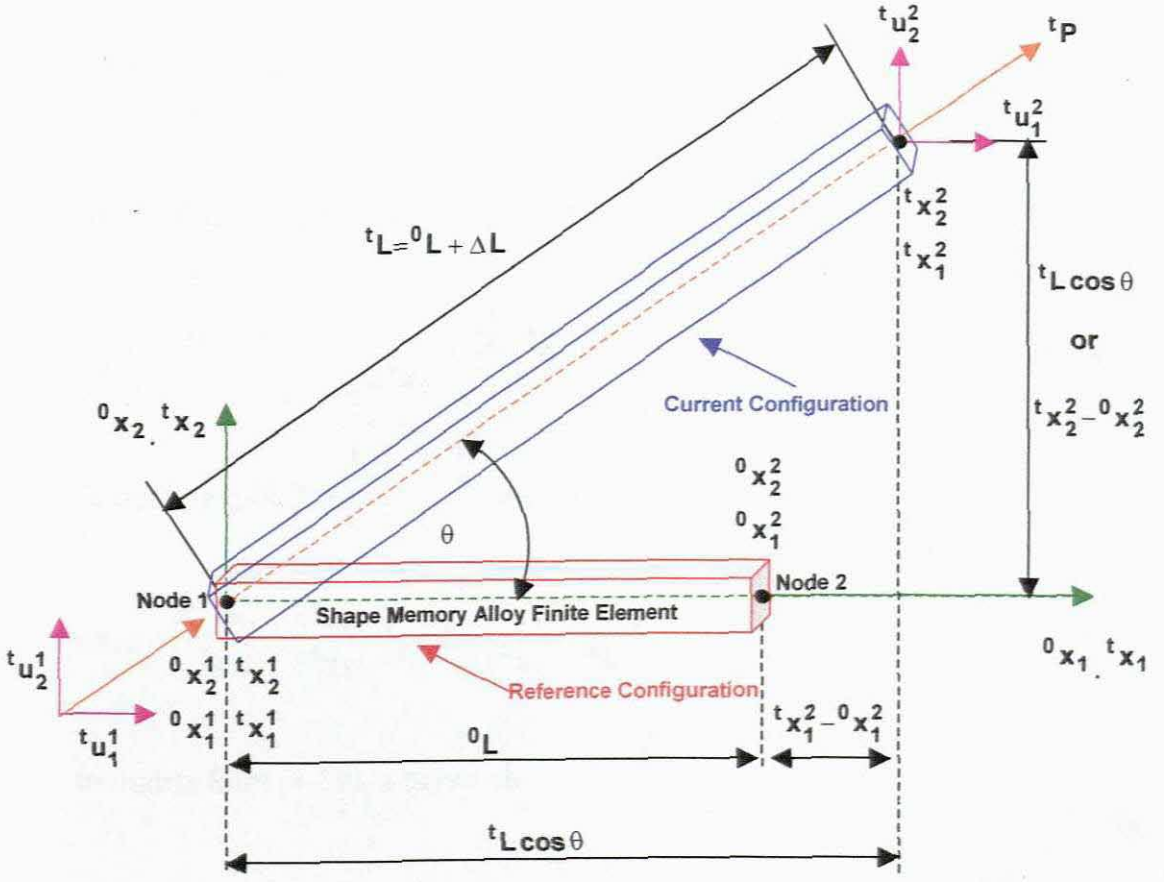


Figure 4.3: Two-node Shape Memory Alloy Truss Element in the reference/initial and current/deformed configurations

Fig. 4.2 shows the SMAT element in its reference and current configurations. It should be noted that the element is straight at time 0 and aligned with the ${}^0x_1$. A load, ${}^tP$, is applied to the element and causes an axial deformation while rotating it through an angle θ to bring it to the current configuration at time t. Using the Total Lagrangian formulation we need to express the linear (${}_0e_{ij}$) and non-linear (${}_0q_{ij}$) strains given in Table 4.2 in terms of the element displacement functions. Our SMAT element only undergoes displacements in the ${}^0x_1$, ${}^0x_2$ plane and therefore we have

$$ {}_0e_{11} = \frac{\partial u_1}{\partial {}^0x_1} + \frac{\partial {}^tu_1}{\partial {}^0x_1}\frac{\partial u_1}{\partial {}^0x_1} + \frac{\partial {}^tu_2}{\partial {}^0x_1}\frac{\partial u_2}{\partial {}^0x_1} \tag{4.36} $$

and

$$ {}_0q_{11} = \frac{1}{2}\left[\left(\frac{\partial u_1}{\partial {}^0x_1}\right)^2 + \left(\frac{\partial u_2}{\partial {}^0x_1}\right)^2\right] \tag{4.37} $$

Using relations given in (4.32) and (4.34) we obtain

$$ \frac{\partial u_1}{\partial {}^0x_1} = \frac{u_1^2 - u_1^1}{{}^0L}; \qquad \frac{\partial u_2}{\partial {}^0x_1} = \frac{u_2^2 - u_2^1}{{}^0L}. \tag{4.38} $$

Substituting (4.38) into (4.36) we obtain

$$ {}_0e_{11} = \frac{u_1^2 - u_1^1}{{}^0L} + \frac{\partial {}^tu_1}{\partial {}^0x_1}\frac{u_1^2 - u_1^1}{{}^0L} + \frac{\partial {}^tu_2}{\partial {}^0x_1}\frac{u_2^2 - u_2^1}{{}^0L} \tag{4.39} $$

In matrix form (4.39) is given as

$$ {}_0e_{11} = \frac{1}{{}^0L}\left[-\left(1+\frac{\partial^t u_1}{\partial^0 x_1}\right) \quad -\frac{\partial^t u_2}{\partial^0 x_1} \quad \left(1+\frac{\partial^t u_1}{\partial^0 x_1}\right) \quad \frac{\partial^t u_2}{\partial^0 x_1}\right]\cdot\begin{Bmatrix} u_1^1 \\ u_2^1 \\ u_1^2 \\ u_2^2 \end{Bmatrix} \tag{4.40} $$

With the aid of Fig. 4.2 and since the displacements are known at time $\mathbf{0}$ and $\mathbf{t}$, we easily obtain

$$ \frac{\partial^t u_1}{\partial^0 x_1} = \frac{{}^t x_1^2 - {}^0 x_1^2}{{}^0 x_1^2 - {}^0 x_1^1} - \frac{{}^t x_1^1 - {}^0 x_1^1}{{}^0 x_1^2 - {}^0 x_1^1}; \qquad \frac{\partial^t u_1}{\partial^0 x_1} = \frac{{}^t L\cos\theta - {}^0 L}{{}^0 L} \tag{4.41} $$

0 due to geometry

and

$$ \frac{\partial^t u_2}{\partial^0 x_1} = \frac{{}^t x_2^2 - {}^0 x_2^2}{{}^0 x_1^2 - {}^0 x_1^1} - \frac{{}^t x_2^1 - {}^0 x_2^1}{{}^0 x_1^2 - {}^0 x_1^1}; \qquad \frac{\partial^t u_1}{\partial^0 x_1} = \frac{{}^t L\cos\theta}{{}^0 L} \tag{4.42} $$

0 due to geometry

with ${}^t L = {}^0 L + \Delta L$ and substituting (4.41) and (4.42) into (4.40) gives

$$ {}_0e_{11} = \frac{{}^t L + \Delta L}{\left({}^0 L\right)^2}\left[-\cos\theta \quad -\sin\theta \quad \cos\theta \quad \sin\theta\right]\cdot\begin{Bmatrix} u_1^1 \\ u_2^1 \\ u_1^2 \\ u_2^2 \end{Bmatrix} \tag{4.43} $$

and hence,

$$ {}_0^t B_L = \frac{{}^t L + \Delta L}{\left({}^0 L\right)^2}\left[-\cos\theta \quad -\sin\theta \quad \cos\theta \quad \sin\theta\right] \tag{4.44} $$

The non-linear strain-displacement matrix is obtained in an analogous manner and is given as

$$ {}_0^tB_{NL} = \frac{1}{{}^0L}\begin{bmatrix} -1 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix} \tag{4.45} $$

In the Total Lagrangian formulation we assume that ${}_0^tS_{11}$ is given in terms of ${}_0^t\varepsilon_{11}$, and from section 4.6 we have

$$ {}_0C_{1111} = \frac{\partial {}_0^tS_{11}}{\partial {}_0^t\varepsilon_{11}} \tag{4.46} $$

The tangent stiffness matrix and force vector is therefore given as

$$ {}_0^tK = {}_0C_{1111}\frac{\left({}^0L+\Delta L\right)^2}{\left({}^0L\right)^3}\begin{bmatrix} \cos^2\theta & \sin\theta\cos\theta & -\cos^2\theta & -\sin\theta\cos\theta \\ \sin\theta\cos\theta & \sin^2\theta & -\sin\theta\cos\theta & -\sin^2\theta \\ \cos^2\theta & \sin\theta\cos\theta & \cos^2\theta & \sin\theta\cos\theta \\ -\sin\theta\cos\theta & -\sin^2\theta & \sin\theta\cos\theta & \sin^2\theta \end{bmatrix} + \frac{{}^tP}{{}^0L+\Delta L}\begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ -1 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix} \tag{4.47} $$

$$ {}_0^tF = {}^tP\begin{Bmatrix} -\cos\theta \\ -\sin\theta \\ \cos\theta \\ \sin\theta \end{Bmatrix} \tag{4.48} $$

where ${}^tP$ is the current force applied in the SMAT element. The Cauchy stress equals ${}^tP/{}^tA$, giving the rest of the relations as

$$ {}_0^tS_{11} = \frac{{}^0\rho}{{}^t\rho}\left(\frac{{}^0L}{{}^0L+\Delta L}\right)^2\frac{{}^tP}{{}^tA}; \quad {}_0^t\varepsilon_{11} = \frac{\Delta L}{{}^0L} + \frac{1}{2}\left(\frac{\Delta L}{{}^0L}\right)^2 \tag{4.49} $$

$$ {}^0\rho\,{}^0L\,{}^0A = {}^t\rho\left({}^0L+\Delta L\right){}^tA; \qquad {}_0^tS_{11} = \frac{{}^0L}{{}^0L+\Delta L}\frac{{}^tP}{{}^0A} \tag{4.50} $$

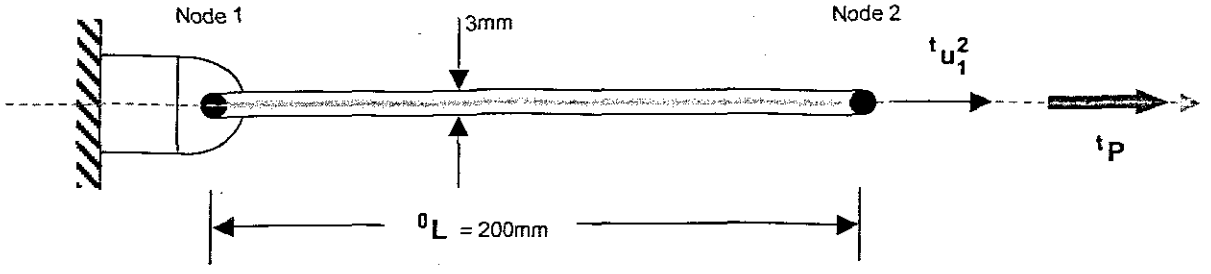
$$ {}^{t}P = {}^{t}_{0}S_{11}\,{}^{0}A\,\frac{{}^{0}L + \Delta L}{{}^{0}L} \qquad (4.51) $$

4.8 Solution of a Shape Memory Alloy Truss (SMAT) element

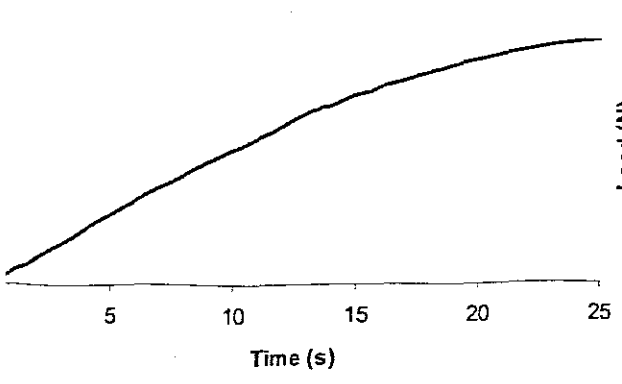
We will now illustrate the use of the equations developed in sections 4.5-4.8. The direct stiffness method will be used to assemble the total stiffness matrix and equations for the following plane shape memory alloy truss example problems.

Problem 1: Single SMAT element

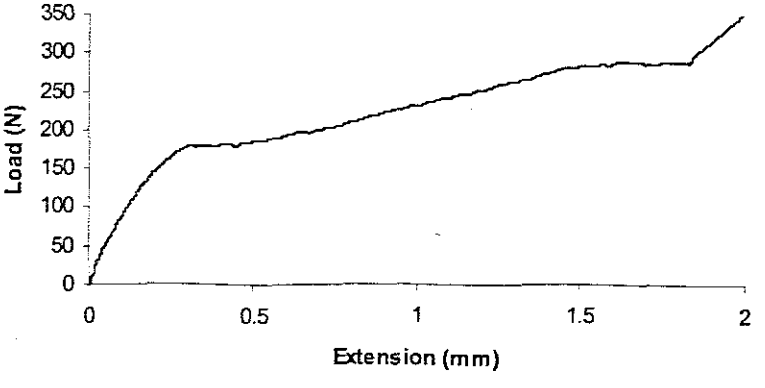
The plane SMAT element shown in Fig. 4.4a is subjected to an axially applied load. This load is given as a function of time, ${}^{t}P(t)$ and is applied at node 2 of the element (See Fig 4.4.b). The element represents a NiTi shape memory alloy and has a constant cross-sectional area of 7.068E-6m^2 with a length of 200mm. Let ${}_{0}C^{M+} = {}_{0}C^{M-} = 34$GPa and ${}_{0}C^{A} = 89$GPa.



(a) Plane SMAT element subjected to an axially applied load



(b) Load-Time relation



(c) Load-Extension relation

Figure 4.4: (a) Plane SMAT element subjected to an axially applied load. (b) Load-Time relation (c) Load-Extension relation

Considering the geometry of the problem and using (4.7)-(4.9) and (4.30)-(4.31) with the terms given in (4.47) an algorithm (See Table 4.4) is developed to solve for the load steps of this SMAT element. The results of $\mathbf{u_1^2(P,t)}$ are presented in Table 4.5.

Given: u_1^2 , F , *tolerance, maximum number of iterations*

1. *Set:* $newu_1^2 = u_1^2$
   $newF = F$
2. **Repeat**
   Set: $oldu_1^2 = newu_1^2$
   Set: $oldF = newF$
   $newu_1^2$ = ***function(*** $oldu_1^2$ ***)***
   $newF$ = ***function(*** $oldF$ ***)***
   if *[R- newF] is less than or equal to tolerance* **then**
   Note: 'solved'
   Until *'solved' (or limit on number of iterations)*
3. **If** *'solved'* **then**
   Record: u_1^2 , F , *number of iterations*
   If *'limit'* **then**
   Record: $lastu_1^2$

Table 4.4: Algorithm for solving displacements for the plane SMAT element of Problem 1.

u_1^2 (mm)	^tF (N)	u_1^2 (mm)	^tF (N)	u_1^2 (mm)	^tF (N)
1.58E-02	3.81E+01	5.65E-01	2.01E+02	1.67E+00	2.79E+02
3.06E-02	6.48E+01	6.89E-01	2.11E+02	1.74E+00	2.86E+02
4.65E-02	1.04E+02	8.27E-01	2.21E+02	1.80E+00	2.93E+02
9.83E-02	1.22E+02	9.80E-01	2.31E+02	1.85E+00	3.02E+02
1.42E-01	1.39E+02	1.15E+00	2.42E+02	1.89E+00	3.11E+02
2.15E-01	1.57E+02	1.28E+00	2.50E+02	1.91E+00	3.18E+02
2.90E-01	1.70E+02	1.35E+00	2.55E+02	1.93E+00	3.25E+02
3.48E-01	1.78E+02	1.47E+00	2.63E+02	1.94E+00	3.31E+02
4.48E-01	1.90E+02	1.58E+00	2.71E+02	1.95E+00	3.42E+02

Table 4.5: Results of Finite Element solution for the plane SMAT element of Problem 1.

The results found in Table 4.5 were plotted (see Fig. 4.5) with the experimental data of a 3mm diameter NiTi shape memory alloy wire of length 200mm, and the numerical results of the model presented in Chapter 3. The finite element method shows better agreement with the experimental data.

This is shown in Fig. 4.6a and b where the percentage error of FEM and Numerical data is plotted with respect to the experimental data. The negative values shown in Fig. 3a and b represent the positions where the FEM and Numerical data crosses the experimental data. From Figs. 4.5 and 4.6 it is evident that the both methods experience difficulties in modelling the initial material response of our NiTi shape memory alloy wires and produces %errors of 97% and 56% respectively for the Numerical and FEM models. This initial material response is where the material shows its initial elastic response as discussed in Chapter 2.

When the initial yield point is reached (Load=178N and Extension=0.32mm), these errors decrease to 8% and -17% respectively for the Numerical and FEM data. During the transformation (de-twinning) from its initial elastic to final elastic regions the %error remains small. The FEM model then shows better agreement with the experimental data as the final elastic region is reached.

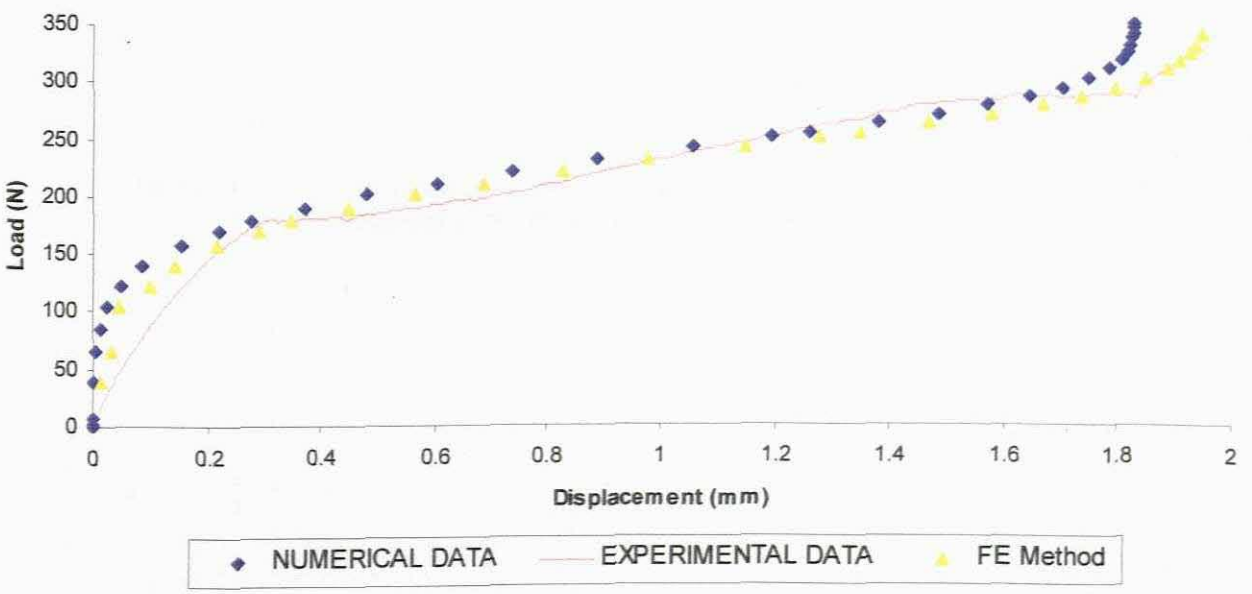
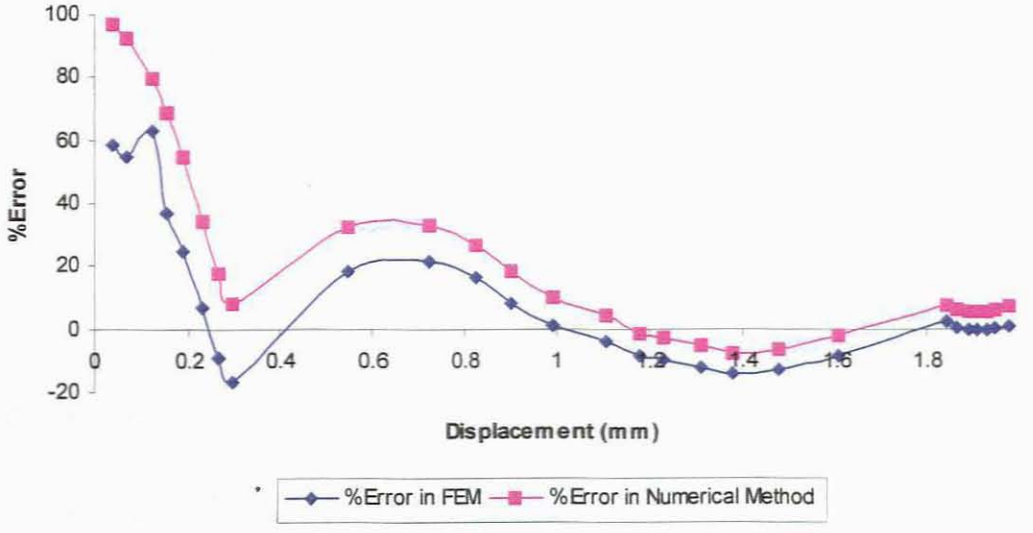


Figure 4.5: Comparison of Experimental, Numerical and Finite Element Data

a



b

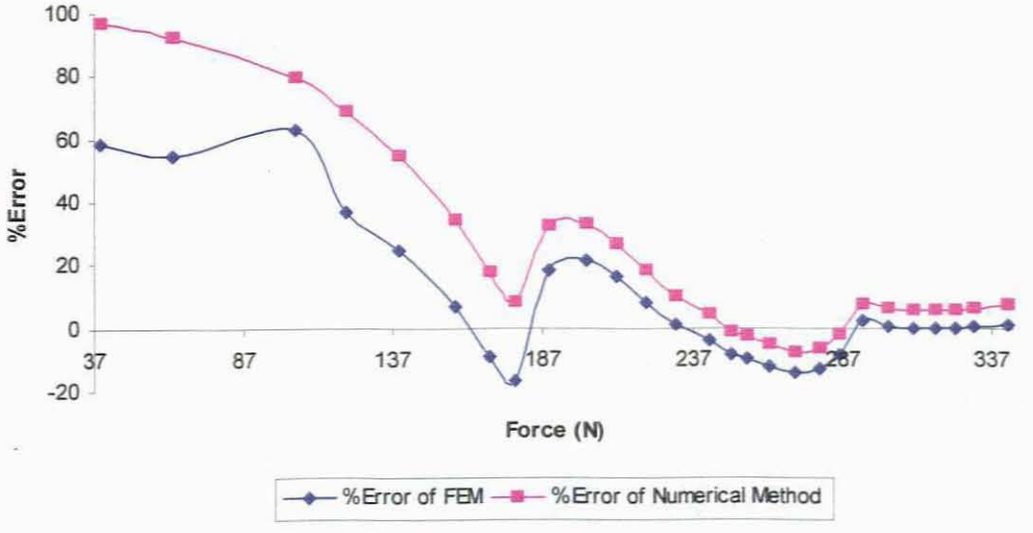


Figure 4.6: Representation of the %Error of (a) Displacement and (b) Load of FEM and Numerical Model compared to Experimental data.

Chapter 5

Development of a Computational Design Tool to Simulate the Behaviour of NiTi Shape Memory Alloys

In Chapters 3 and 4 we successfully managed to simulate the complex quasi-plastic material response exhibited by shape memory alloys at low temperatures. This response is inherent to the shape memory effect. The design of actuators comprising of shape memory alloys that harness the shape memory effect is dependent on the reliable simulation of this material response.

In this section we show the development of a computational design tool to simulate the behaviour of NiTi shape memory alloys. This design tool is specifically suited to design actuators consisting of NiTi shape memory alloys that harness the shape memory effect for their operation. The framework of this tool is written in the Language of C++ and C++ Builder was used to construct the software program. The computational framework of this tool will consist of a calibrated constitutive model (a modified version of the Müller-Achenbach model [32], [39]-[42]) to simulate the complex material responses exhibited by shape memory alloys, which is incorporated into a finite element formulation with a control algorithm to study multi-dimensional uniaxial shape memory alloy line

actuators as structural members in dynamic mechanical systems. This computational tool enhances the design of shape memory alloy line actuators by providing necessary information for actuator heating and cooling, actuator force and stroke, and actuator efficiency.

Finite element implementations have been have been done by Brinson and Lammering and Lagoudas et al. Seelecke used the finite element method to simulate the control of adaptive beams using shape memory alloy wires, which were all orientated in the same direction (one-dimensional) [40]. Recently Masuda et al. used the finite element method in the design of base isolation devices consisting of shape memory alloy springs the harnessed the pseudo-elastic effect for their operation [63].

The general applicability of the finite element method makes it the most widely used simulation tool [40]. The technique has unique capabilities and can be cost effective if properly used. In Chapter 4 we have shown how the constitutive model could be implemented into a finite element formulation and successfully simulated the material response of single SMAT element being subjected to an externally applied load that is dependent of time.

5.1 Numerical Solution of Stiff Sets of Equations

Our solution technique had to be altered when we started considering multi-dimensional uniaxial shape memory alloy systems. We found that the RADAU5 routine (described in Chapter 3) was not suited for the finite element method and a search was conducted to find a suitable numerical scheme that could be easily implemented into the finite element method. The generic problem in ordinary differential equations is to study a set of N coupled first-order differential equations for the functions y_i, $i = 1,2,...,N$, having the general form

$$\frac{\partial y_i(x)}{\partial x} = f_i(x, y_1, \ldots, y_N), \qquad i = 1, \ldots, N \tag{5.1}$$

where the functions f_i, on the right-hand side are known. We settled for the Semi-implicit Euler Method of the form (as presented by Press et al. [62]):

$$y_{n+1} = y_n + \lambda\left[1 - \lambda\frac{\partial f}{\partial y}\right]^{-1} \cdot f(y_n) \tag{5.2}$$

where λ is the interval through which the solution is advanced. Here we only consider two functions, i.e., rate laws for the two martensitic phase fractions $\xi_{M_\pm}$. A Jacobian matrix, $\mathbf{A}$, was constructed to aid in the solution and is given in the form

$$\mathbf{A} = \begin{bmatrix} \left(1 - \lambda\frac{\partial f_1}{\partial y_1}\right) & \left(1 - \lambda\frac{\partial f_1}{\partial y_2}\right) \\ \left(1 - \lambda\frac{\partial f_2}{\partial y_1}\right) & \left(1 - \lambda\frac{\partial f_2}{\partial y_2}\right) \end{bmatrix} \tag{5.3}$$

with its inverse given by

$$\mathbf{A}^{-1} = \frac{1}{\det \mathbf{A}} \begin{bmatrix} \left(1 - \lambda\frac{\partial f_2}{\partial y_2}\right) & -\left(1 - \lambda\frac{\partial f_1}{\partial y_2}\right) \\ -\left(1 - \lambda\frac{\partial f_2}{\partial y_1}\right) & \left(1 - \lambda\frac{\partial f_1}{\partial y_1}\right) \end{bmatrix} \tag{5.4}$$

considering (5.3) and (5.4), (5.2) can now be rewritten in the form

$$y_i^{n+1} = y_i^n + \lambda A_{ij}^{-1} \cdot f_j\left(y_i^n\right) \tag{5.5}$$

The implementation of this numerical solution scheme into C++ is given in Appendix C.

5.2 Program Description

The solution algorithm used in this program is presented in Figure 5.1. It starts off by calculating the nodal displacements, (4.21) of a given structural system being subjected to an externally applied load. This load can be dependent or independent on time. These displacements are then used to determine the strains, which is then substituted into the relation for the Second Piola-Kirchoff stress (4.30). The latter relation is given in terms of the phase fractions, which is calculated using the phase transformation kinetic equations (3.25 – 3.28). The Second Piola-Kirchoff stress is now used to determine the internal force that the element is being subjected to. This force value is used to determine if the structural system is in equilibrium or not (4.1). If so, the calculation is stopped and the results plotted. If not, the result for the force is substituted back into the solution until equilibrium has been achieved.

The program makes use of an input file (see Fig. 5.2) for reading in nodal coordinate data, element connectivity and displacement and load boundary conditions. The user is prompted to open this input file when the program is initialized (see Fig. 5.3). The data is now read in and a plot of the input data is presented (see Fig. 5.4) in terms of the position of nodes and elements. At the start of the analysis a window appears asking the user for the initial time, time increment and the final time (see Fig. 5.5). When these values have been entered and the analysis is run, the user has the option of viewing the deformation of the entire structure, the x-displacment, y-displacement, uniaxial stress and strain, the deformed structure, and the two variants of martensite (see Figs. 5.6 – 5.12). The contour plots in Figs. 5.11 and 5.12 shows the unique nature of this program. Since we start off with equal amounts of martensitic variants (50% of each) these plots show the transformation of twinned martensite to de-twinned martensite. Fig. 5.12 shows elements 3 and 6 experienced complete transformation, while elements 1,2,4,5,7 and 8 only partially

transformed, with element 9 showing no transformation at all. It shows in fact that element 9 is redundant in this position. In the next section we will demonstrate the program's use the design of other smart actuators.

5.3 Program usage in Actuator design

The preceding section shows the ease at which the program can be used to design multi-dimensional actuators consisting of NiTi shape memory alloy wires. The results presented shows the complex quasi-plastic material behaviour exhibited by this material. These devices can be used in a variety of applications ranging from very sophisticated space deployment actuators to simple switches that may be used to control valves used in fluid pipelines.

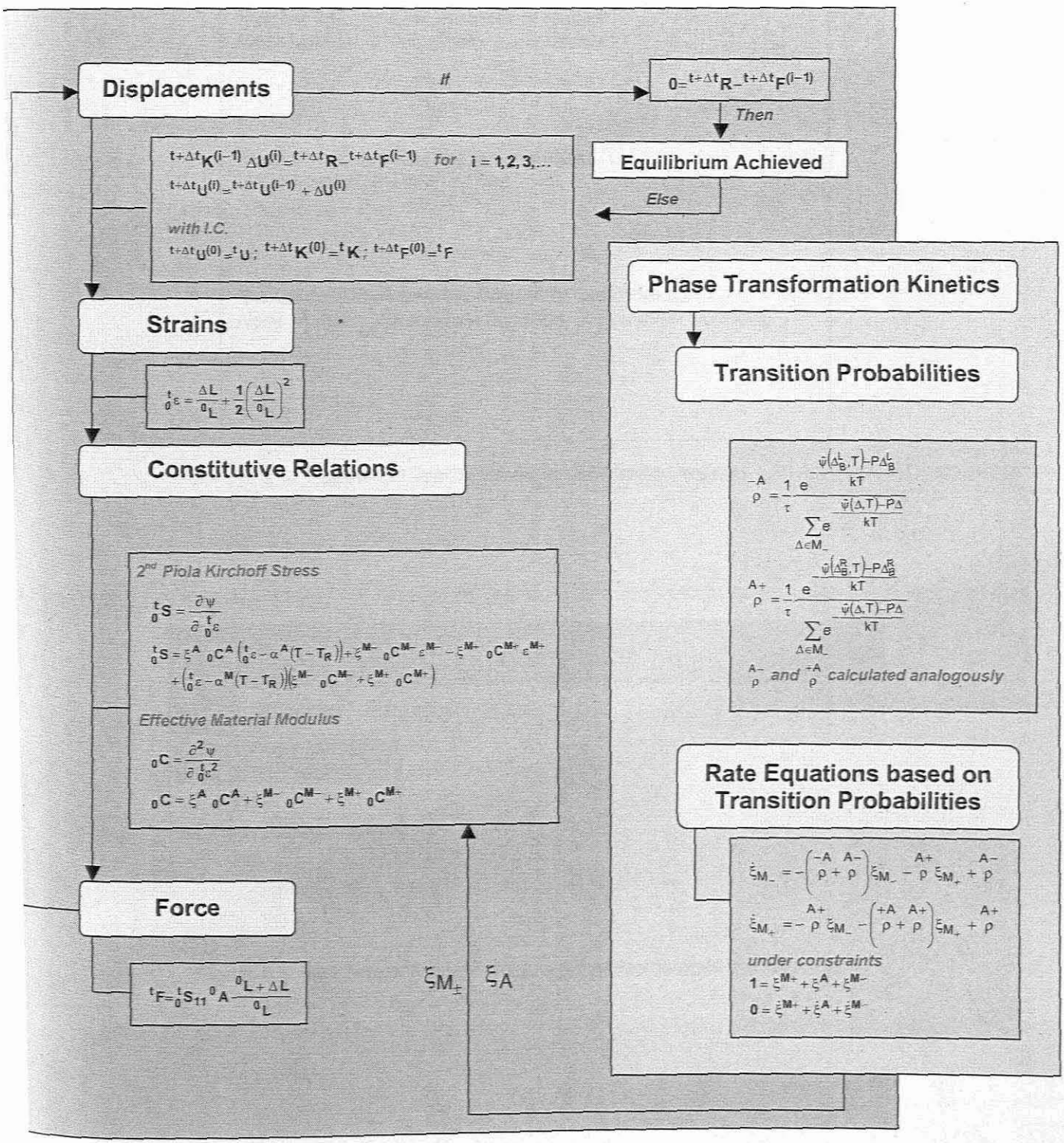


Figure 5.1: Solution algorithm used in the program

Input File used for the Program

Number of Elements, Number of Nodes
Node Number, X-coordinate, Y-coordinate
:
:
:
:
Number of Load Boundary Conditions
Node Number, X-Load, Y-Load
:
:
:
:
Number of Displacement Boundary Conditions
Node Number, X-Boundary Condition, Y-Boundary Condition
:
:
:
:

Figure 5.2: Typical Input File used in the Program

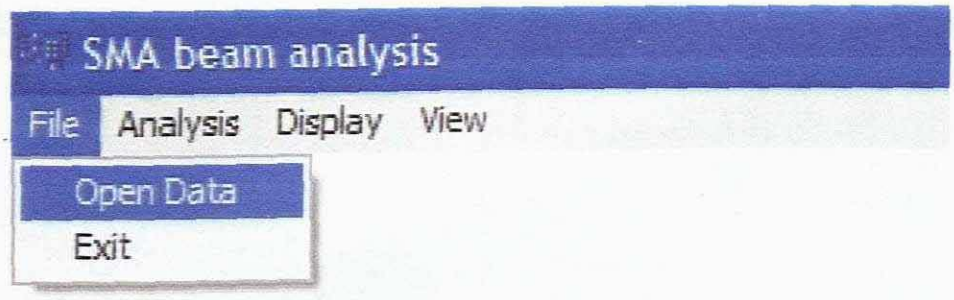


Figure 5.3: Inter-Active Window prompting the User to open the input File

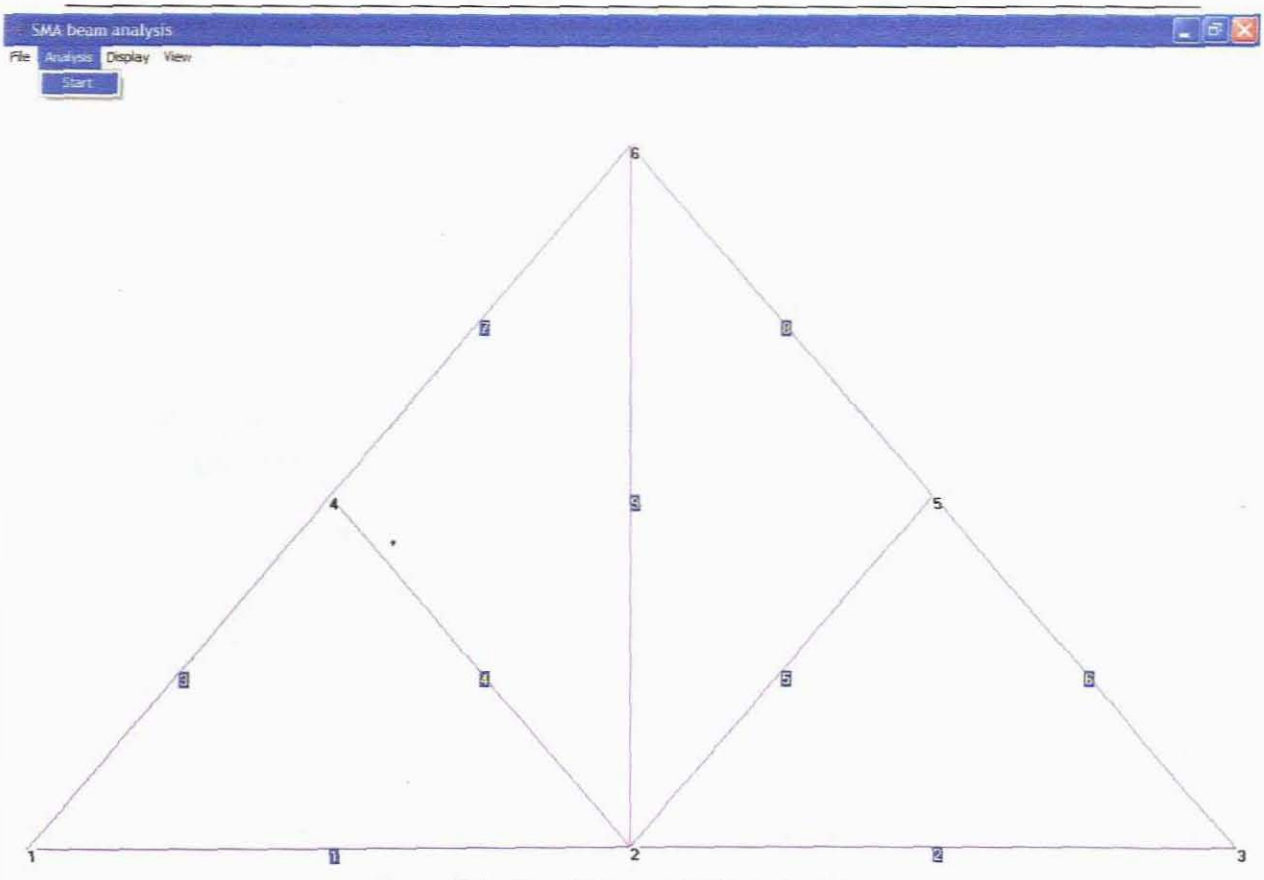


Figure 5.4: Plot of the input data presented.

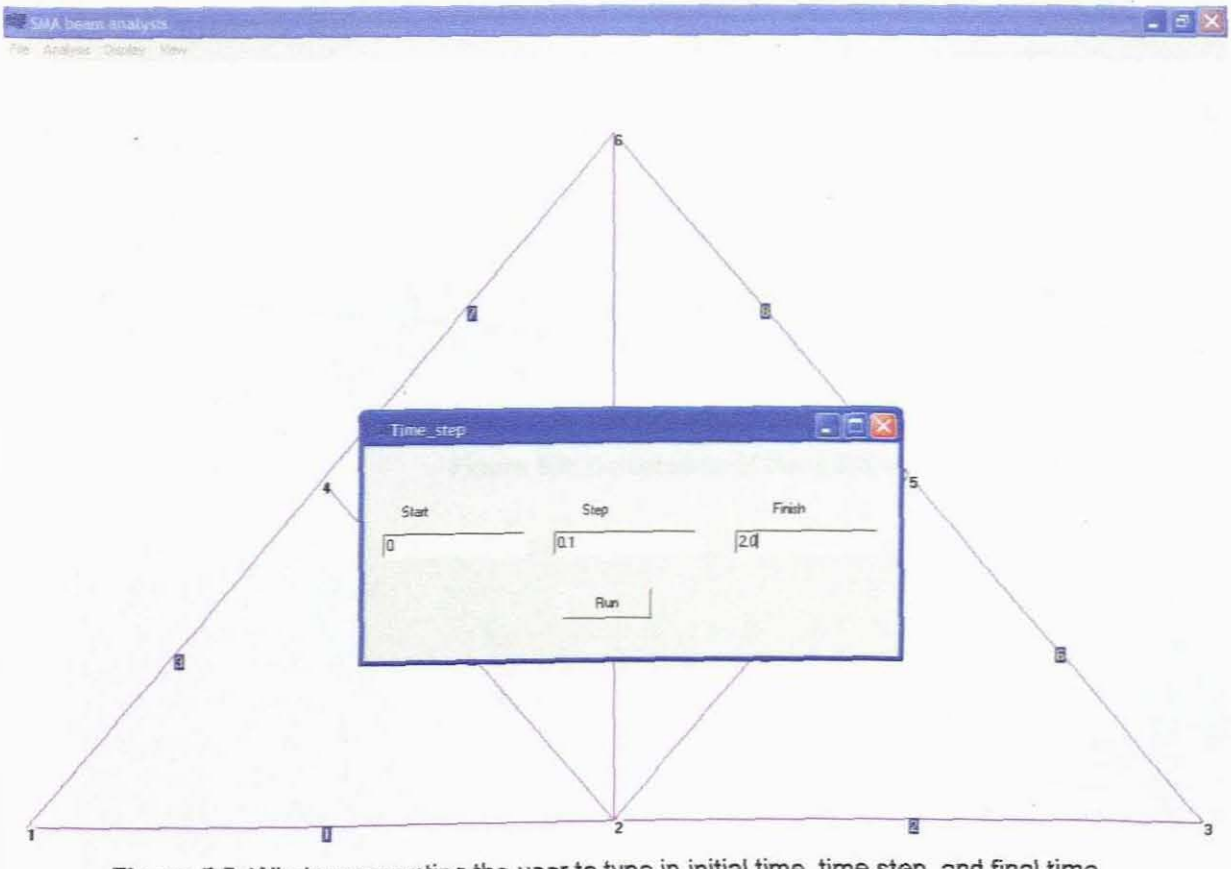


Figure 5.5: Window prompting the user to type in initial time, time step, and final time

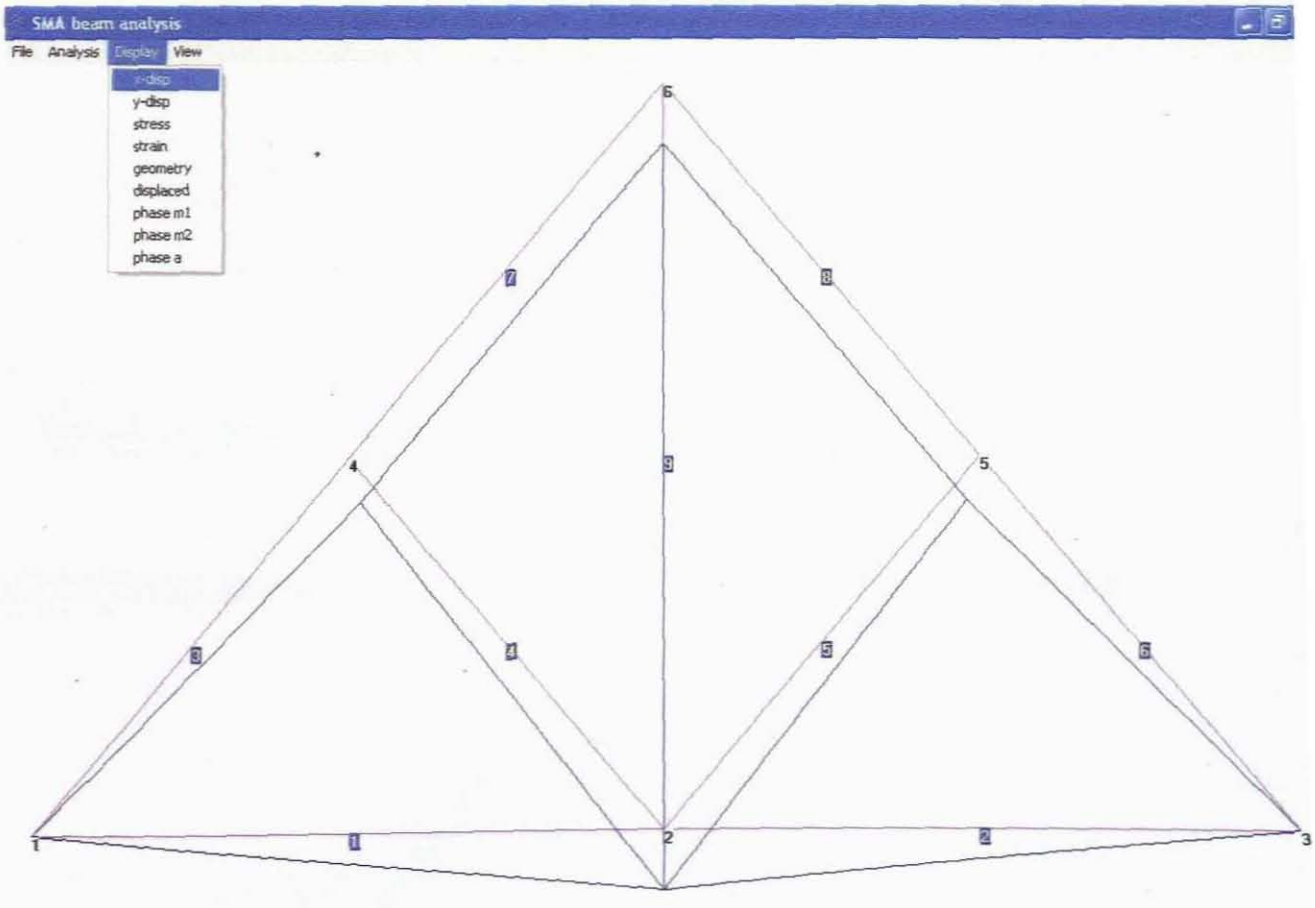


Figure 5.6: Deformation of the entire structure

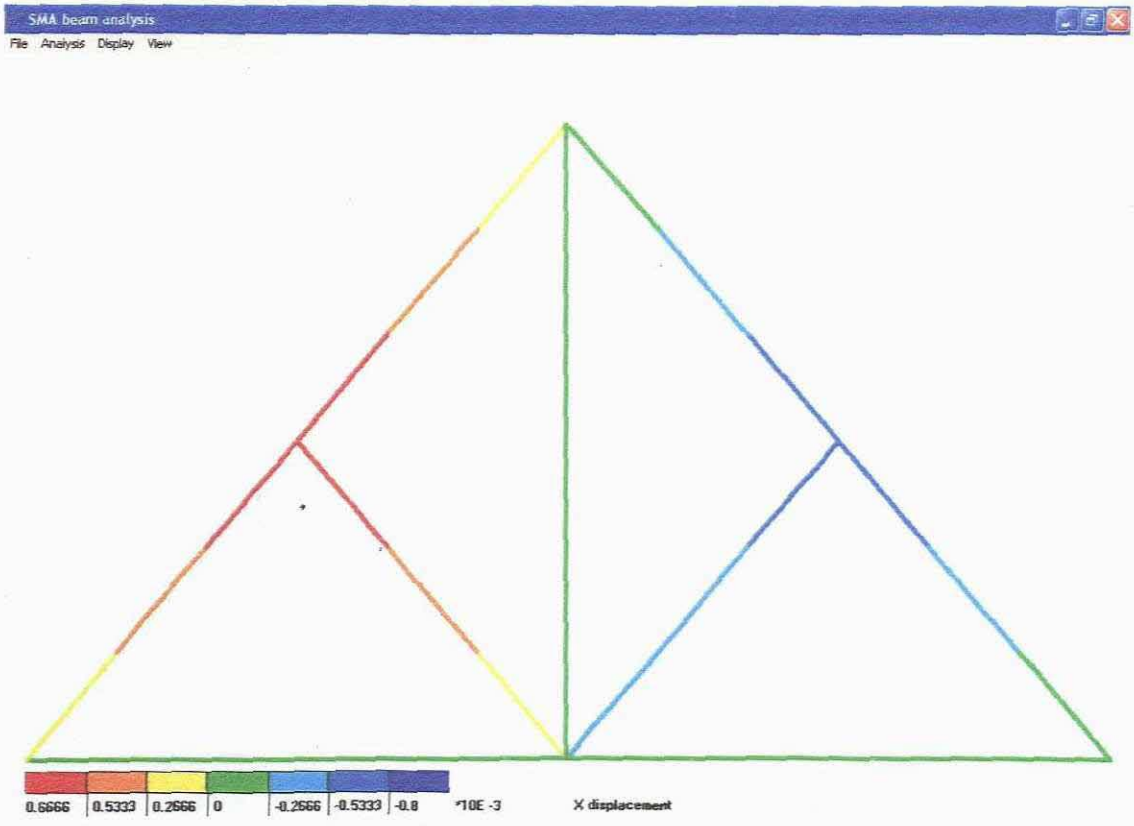


Figure 5.7: Contour plot of x-displacement

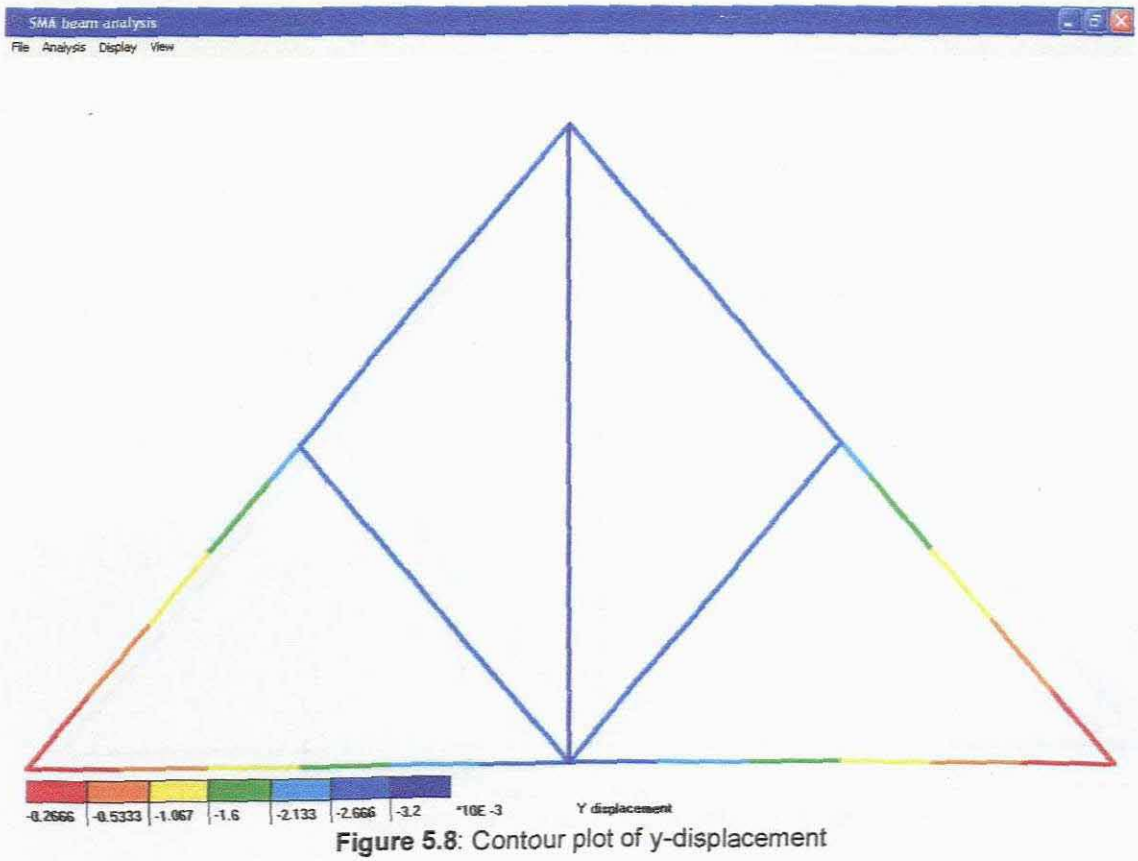


Figure 5.8: Contour plot of y-displacement

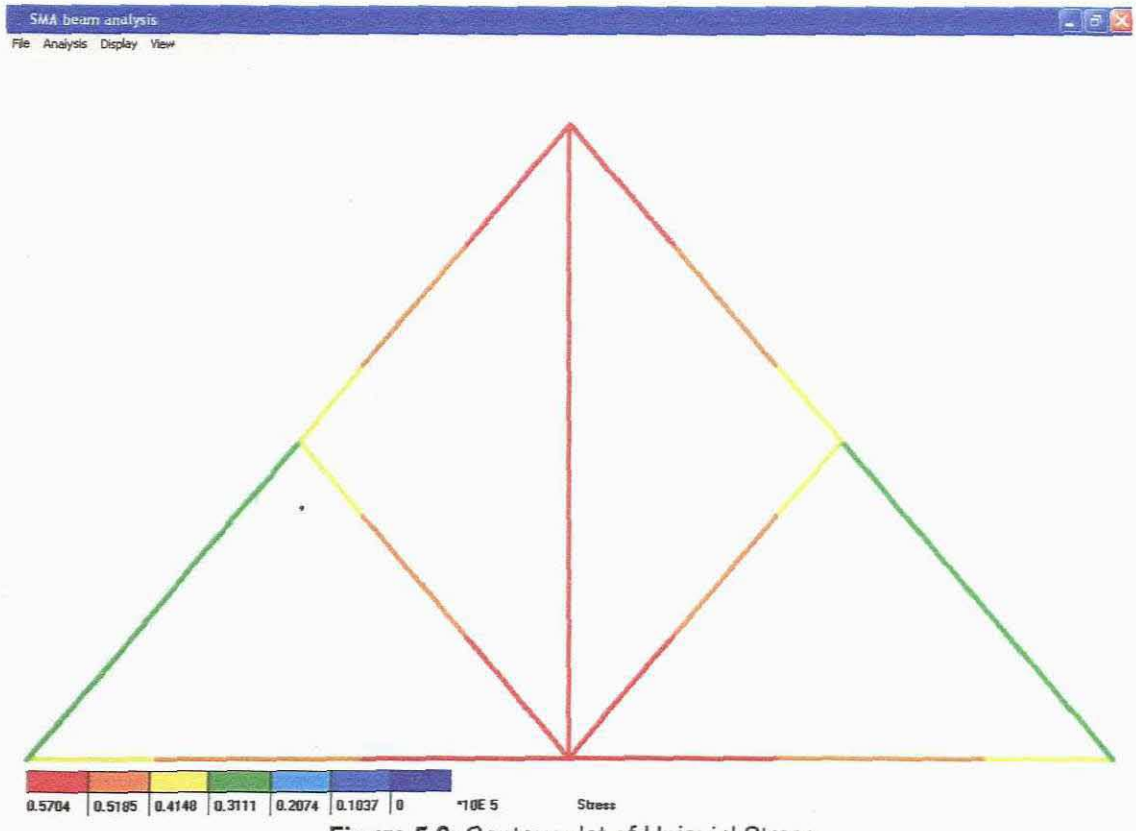


Figure 5.9: Contour plot of Uniaxial Stress

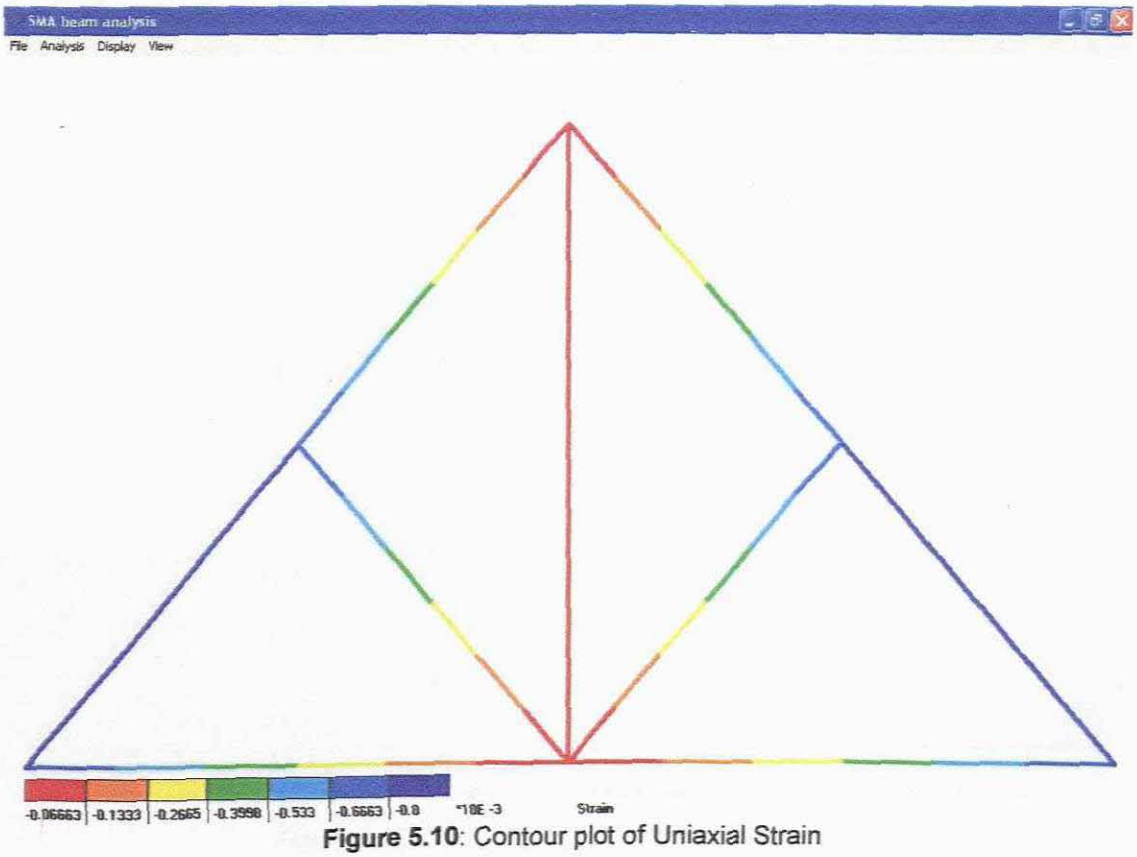


Figure 5.10: Contour plot of Uniaxial Strain

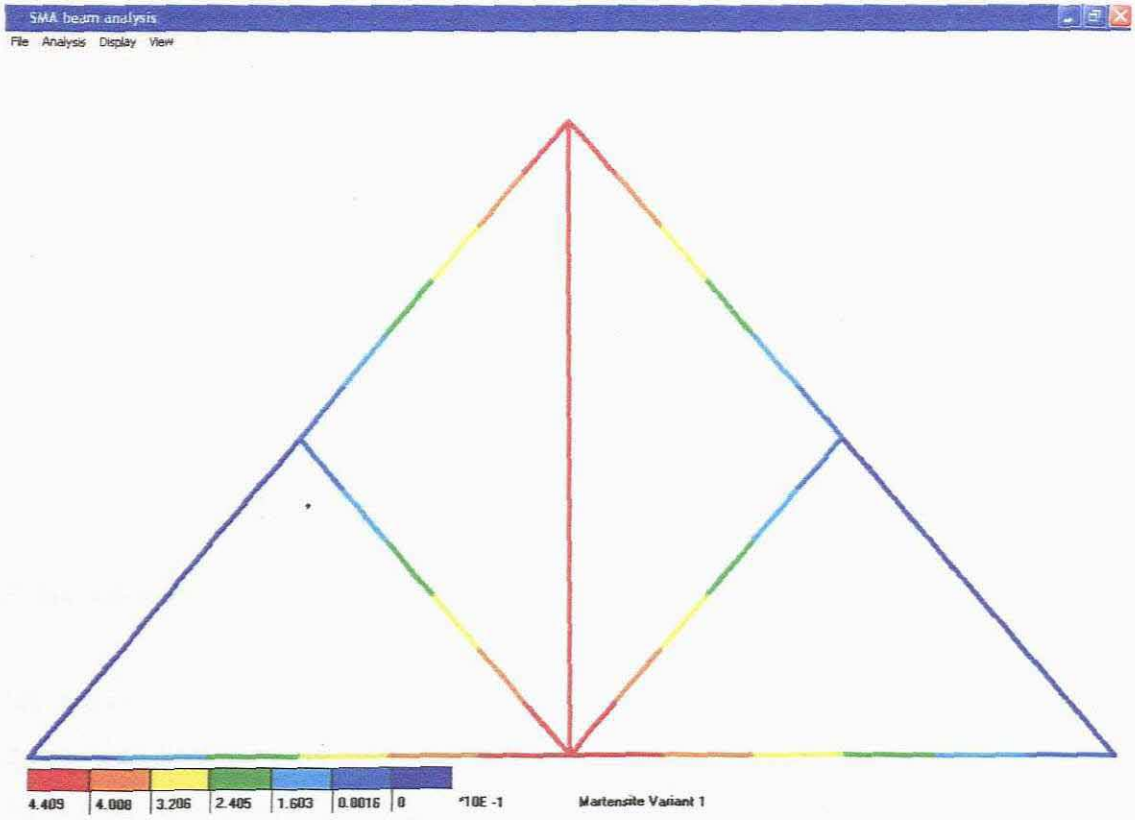


Figure 5.11: Contour plot of Martensite: Variant 1

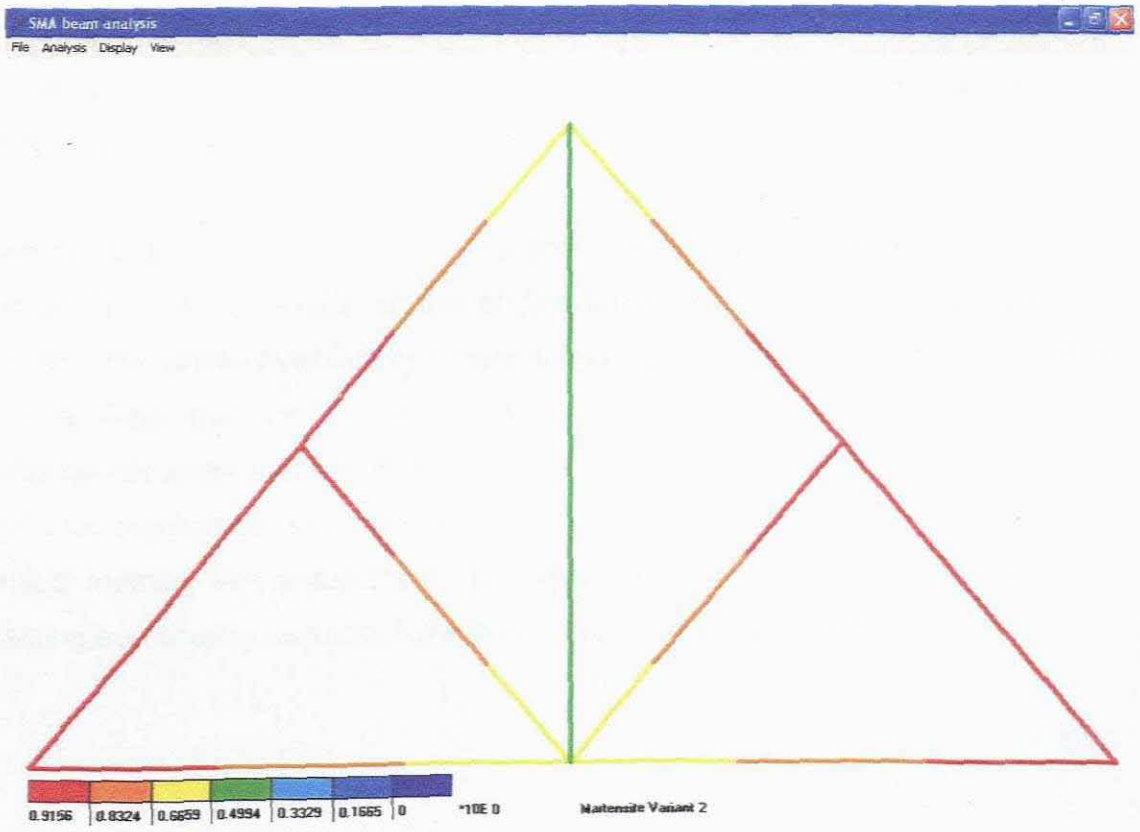


Figure 5.12: Contour plot of Martensite: Variant 2

Chapter 6

Conclusion

6.1 Conclusions

This thesis presents the successful development an engineering computational design tool that can be used for the design of smart or intelligent actuators comprising of NiTi shape memory alloy wires that harness the shape memory effect for their operation. An experimental database pertaining to one aspect of the shape memory effect, the quasi-plastic material response was developed to serve as an engineering design aid but also to serve as a verification tool for the simulation of this unique material response.

The computational framework of this tool consists of a calibrated constitutive model (a modified version of the Müller-Achenbach) to simulate the complex material responses exhibited by shape memory alloys, which is then incorporated into a finite element formulation with a control algorithm to study multi-dimensional uniaxial shape memory alloy line actuators as structural members in dynamic mechanical systems. This computational tool enhances the design of shape memory line actuators by providing necessary information for actuator heating and cooling, actuator force and stroke, and actuator efficiency.

6.1.1 Experimental Investigation

Shape memory alloys exhibit two unique material responses ascribed to solid-solid thermo-elastic martensitic forward and reverse phase transformations, i.e., the shape memory effect and the pseudo-elastic effect. The experimental investigation was performed to study only one aspect of the shape memory effect, the quasi-plastic material response under tensile loading conditions. Shape memory alloy while in its low temperature martensitic state exhibit this behaviour under different loading conditions (tension, compression, torsion, tension-torsion, compression-torsion, etc.). The experimental results obtained were consistent with what other researchers found in the field and even show some interesting findings regarding this unique material response. It is hoped that these results will add to the body of work performed in this area of research but also more importantly it provided better insight into the different approaches used in the constitutive modeling of shape memory alloys. All experiments were conducted at the Strength of Materials Laboratory of the Department of Mechanical Engineering at the Peninsula Technikon.

The experimental investigation was performed to determine the effect that different displacement rates have on geometric and mechanical properties of 1, 2 and 3mm diameter NiTi shape memory alloys wires specimens of varying lengths. A summary of these findings will now follow.

- Quasi-plastic material response displays three distinct regions when plotted graphically, i.e., an initial elastic region (IE), a nearly horizontal region (NH), and finally another elastic region (FE).

- Elastic regions suggest that the specimen changes from its initial state to some final state through a transformation and the slopes of these elastic regions are similar.

- The nearly horizontal regions on these graphs thus determine the location of the transformation behaviour from one elastic state to the other.

- Transformation region of the 1mm diameter shape memory alloy wire specimens shows very small fluctuations of the load with regions of constant load between them. These load fluctuations can be regarded as the flipping or de-twinning of the martensitic twins.

- Increase in the displacement rate causes a decrease in the frequency of these fluctuations and thus produce a more homogeneous deformation.

- The 2 and 3mm diameter NiTi shape memory alloy wires does not show the fluctuations observed in the transformation region and show a steeper transformation region.

- 1mm diameter wire transforms from one state to the other with a seemingly perfectly plastic behaviour while the 2 and 3mm diameter wires can take the form of plasticity models that exhibit hardening behaviours.

- The steeper transformation region observed in the 2 and 3mm diameter NiTi shape memory alloy wire could be ascribed to the geometric changes of the material only.

- Thinner wires (1mm or less) can thus be considered as uniaxial test specimens showing the actual horizontal transformation behaviour of shape memory alloys.

- Thicker wires thus show a transformational behaviour with hardening-like behaviour.

- Quasi-plasticity has preferred displacement rates.

- For quasi-plasticity, the nucleation load of transformation is higher than that of the transformation load.

- Lüders-like deformation was observed but disappears with an increase in the displacement rate for certain lengths of wire.

- Increase in the displacement rate had no effect on the total transformation strain for the 1mm diameter wire specimens but showed slight decreases for the 2 and 3mm diameter wires.

- Decreases in the value of the initial yield stress of the different wire specimens as the displacement rate is increased.

- The difference in yield stress values (second yield – initial yield) indicates the way in which the material tries to maintain the total transformation strain that the wire specimens produce.

6.1.2 Constitutive Modeling

The Multi-Well or Triple Well constitutive approach was used to simulate the complex material response, i.e., the shape memory effect, exhibited by NiTi shape memory alloy wires. The idea behind this particular shape memory alloy model was originally developed by Achenbach and Müller and Achenbach and was then further refined by Müller and Seelecke and Seelecke and Kastner. The model finds it's footing firmly in the foundations of Thermodynamics and Statistical Thermodynamics. The model is particularly appropriate to study the case of a shape memory alloy element subjected to uniaxial loading conditions. The model assumes the existence of an austenitic phase fraction and describes

the evolution of two martensitic phase fractions based on the theory of thermally activated processes.

Numerical simulations to illustrate the model's capability to reproduce the quasi-plastic material response exhibited by NiTi shape memory alloy wires were performed. The mathematical structure of the model is given by the system of non-linear ordinary differential equations in time coupled with an algebraic relation for the displacement of the shape memory alloy specimen. A FORTRAN program was constructed that uses the RADAU5 routine for solution of the ODE's.

The computational variables were obtained from experimental data (from Chapter 2) for a NiTi shape memory alloy rod of 3mm diameter and a length of 200mm. The load-extension data was then plotted with the results of the numerical simulation. These results compared favourably. Although there is some variation in the initial loading region, the model shows its appropriateness in handling the shape memory effect observed in 3mm diameter NiTi shape memory alloy rods.

6.1.3 Finite Element Modeling

The Finite Element formulation of our shape memory alloy element follows the Total Lagrangian formulation to describe its unique material behaviour. This type of analysis describes large displacements, large rotations, and large strains and uses the Second Piola-Kirchhoff stress measure, which is work-conjugate with the Green-Lagrange strain measure. The governing finite element matrices for an isoparametric continuum finite element with displacement degrees of freedom were presented. Constitutive relations were presented using the descriptions given in Chapter 3. The derivation of the tangent stiffness matrix and force vector for our one-dimensional non-linear elastic, constant cross-sectional area Shape

Memory Alloy Truss element (SMAT) subjected to large displacements and large strains is also given. The finite element equations are used to solve for a simple one-dimensional truss subjected to a tensile load. The direct stiffness method was used to assemble the total stiffness matrix and equations.

The same computational variables as those used in Chapter 3 were used. The results obtained from the solution were plotted with the experimental data of a 3mm diameter NiTi shape memory alloy wire of length 200mm, and the numerical results of the model presented in Chapter 3. The finite element method showed better agreement with the experimental data.

6.1.4 Implementation of Finite Element Formulation into the Computation Design Tool

The finite element formulation described in Chapter 4 was successfully implemented into a computational design tool that may be used for the design of multi-dimensional actuator systems comprising of NiTi shape memory alloy wires. The program uses the Semi-Implicit Euler method to solve the differential equations used for the phase transformation kinetics. The phase fractions that are calculated using this method, is substituted into the constitutive relations for this material. The program gives results for the x-displacement, y-displacement, uniaxial stress and strain, the deformed structure, and the evolution of the two variants of martensite. To the best of the author's knowledge, this kind of analysis approach has not yet been attempted.

6.2 Recommendations

Although this program shows its versatility for its use in the design of smart actuator systems comprising of NiTi shape memory alloy wires, there is however more work to be done in refining it. Currently the program only simulates the load

induced phase transformation or quasi-plastic material response thus producing the actuator stroke for a given actuator system. Work still has to be done to include temperature effects to complete the full shape memory effect.

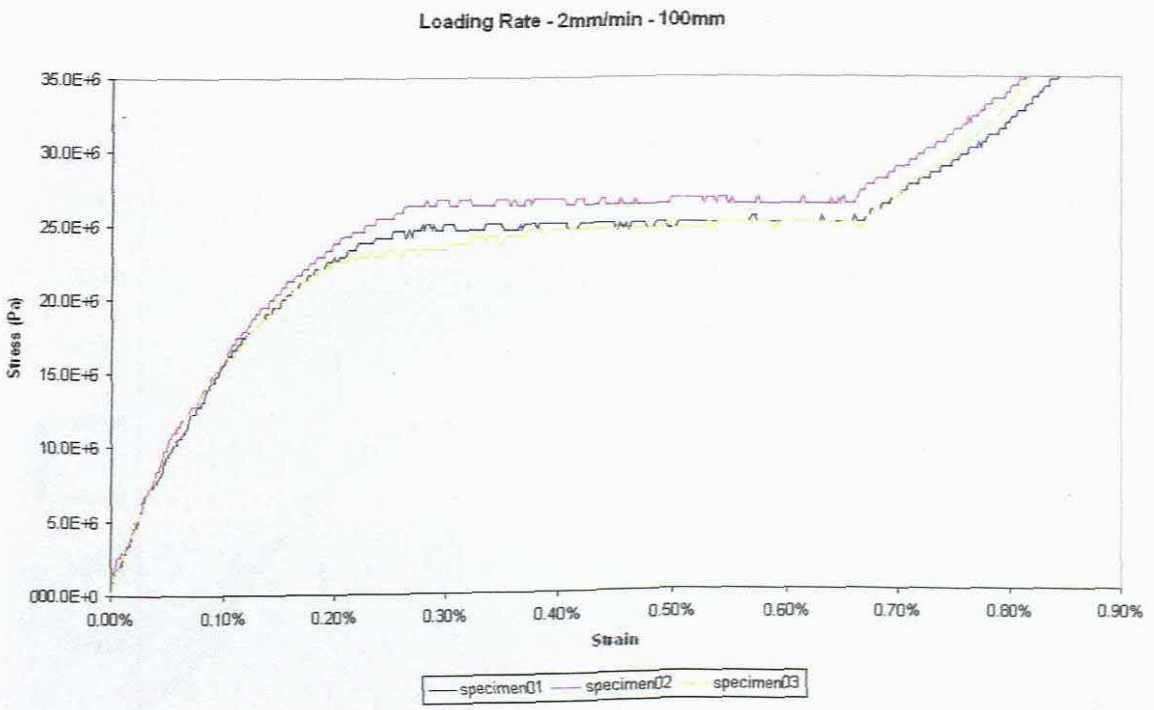
In terms of experimental investigation, we still need to investigate the complex pseudo-elastic material behaviour. Once this is completed, we may include this material response in the program.

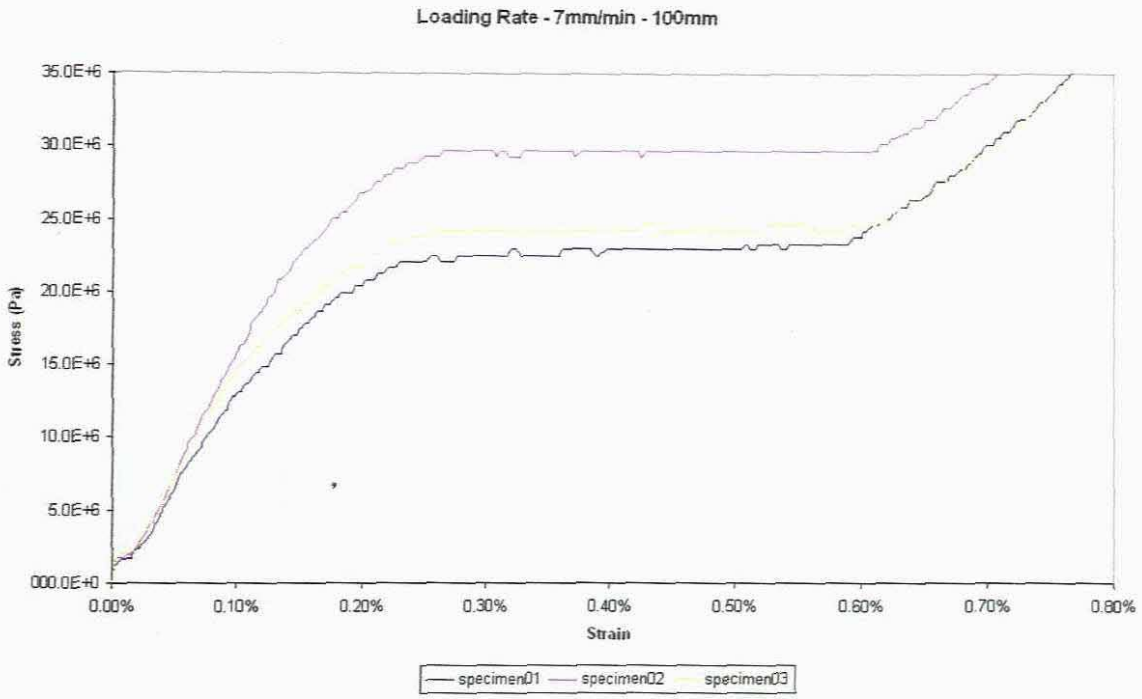
It has been observed that the constitutive approach used can simulate the behaviour of 3mm diameter wire quite well. This is however not the case for 1 and 2mm diameter wires and thus a further investigation needs to be conducted to improve the numerical results obtained for 1 and 2mm diameter wires.

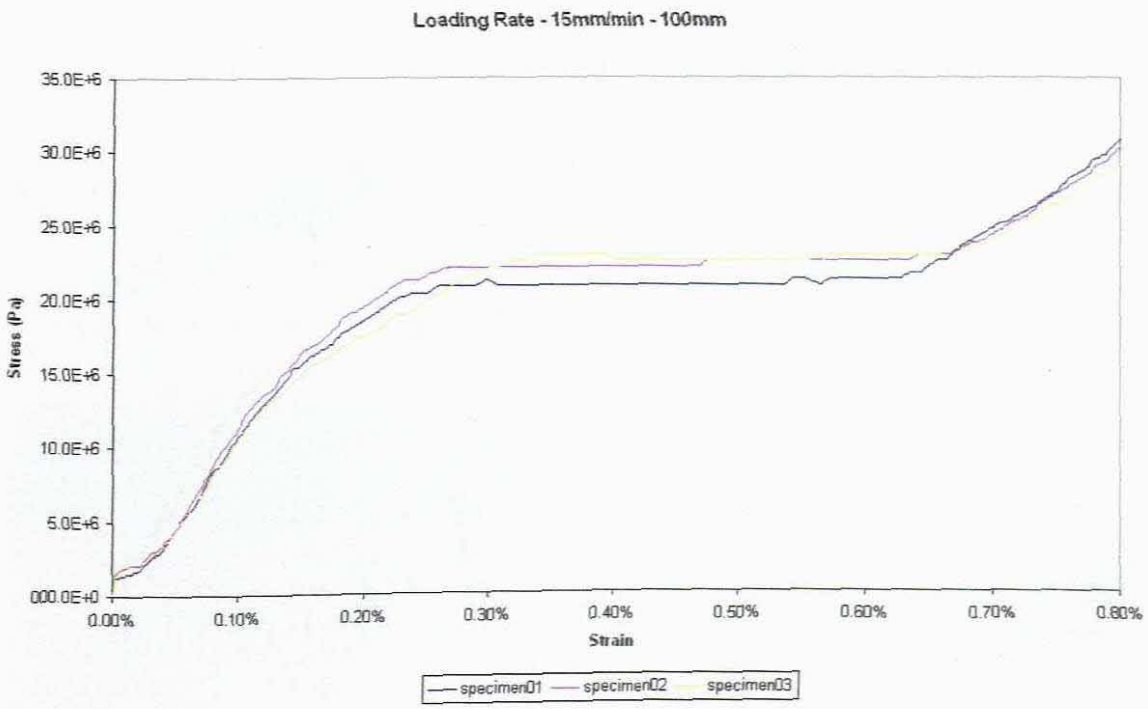
Appendix A

Selected Stress-Strain Graphs of Experimental Results presented in Chapter 2

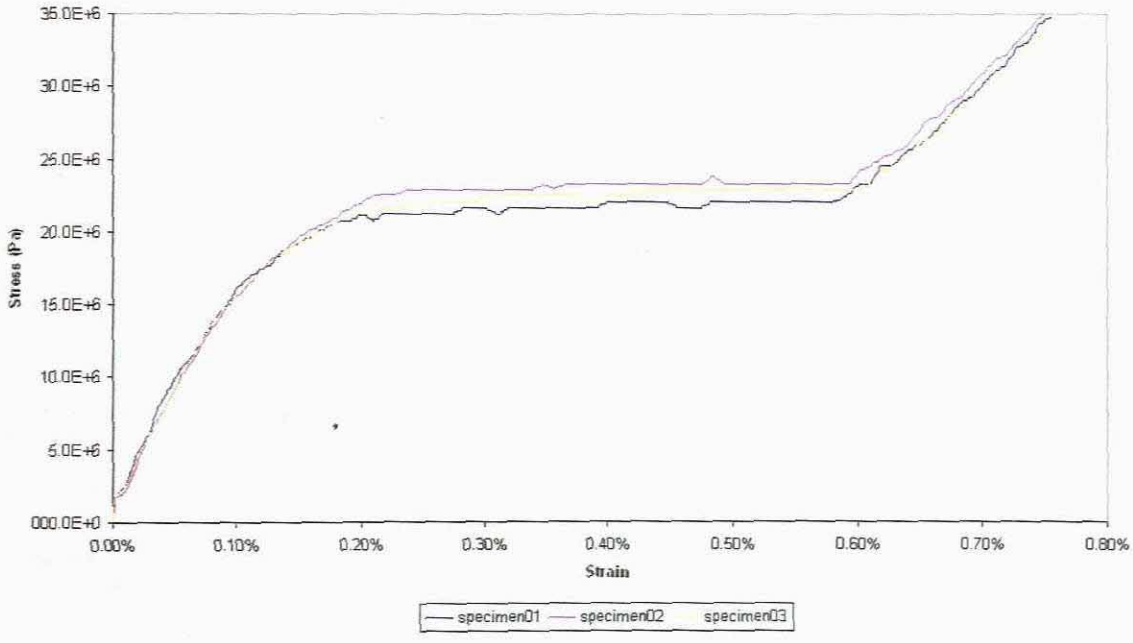
1. 1mm diameter NiTi Shape memory alloy Wires



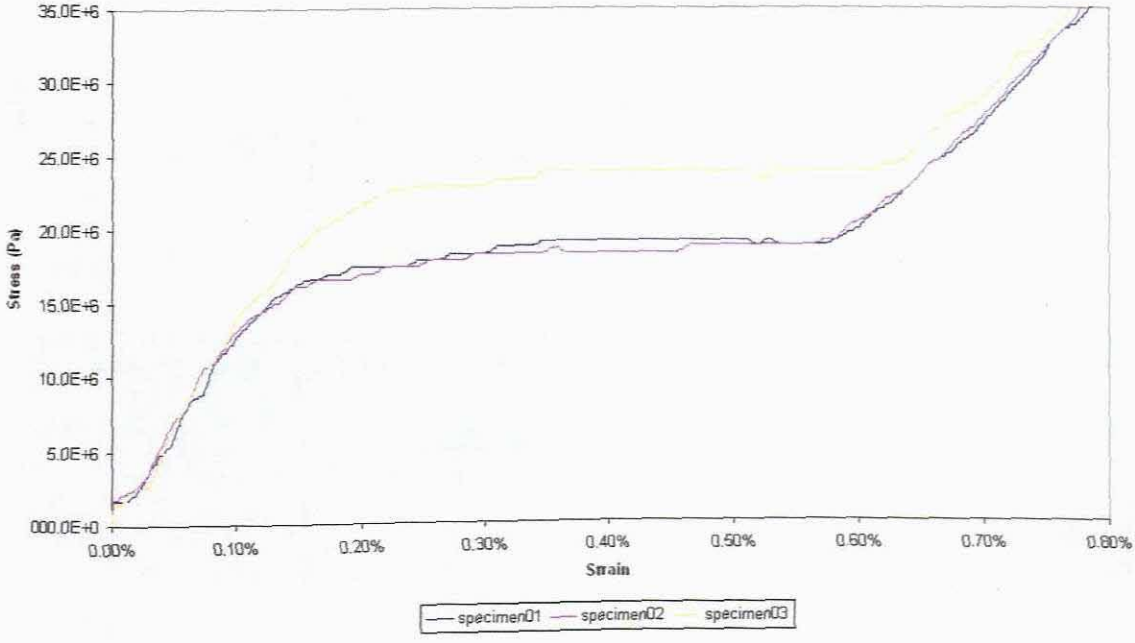

Loading Rate - 7mm/min - 100mm
35.0E+6
30.0E+6
25.0E+6
20.0E+6
15.0E+6
10.0E+6
5.0E+6
000.0E+0
Stress (Pa)
0.00%
0.10%
0.20%
0.30%
0.40%
0.50%
0.60%
0.70%
0.80%
Strain
specimen01
specimen02
specimen03


Loading Rate - 15mm/min - 100mm
35.0E+6
30.0E+6
25.0E+6
20.0E+6
15.0E+6
10.0E+6
5.0E+6
000.0E+0
Stress (Pa)
0.00%
0.10%
0.20%
0.30%
0.40%
0.50%
0.60%
0.70%
0.80%
Strain
specimen01
specimen02
specimen03

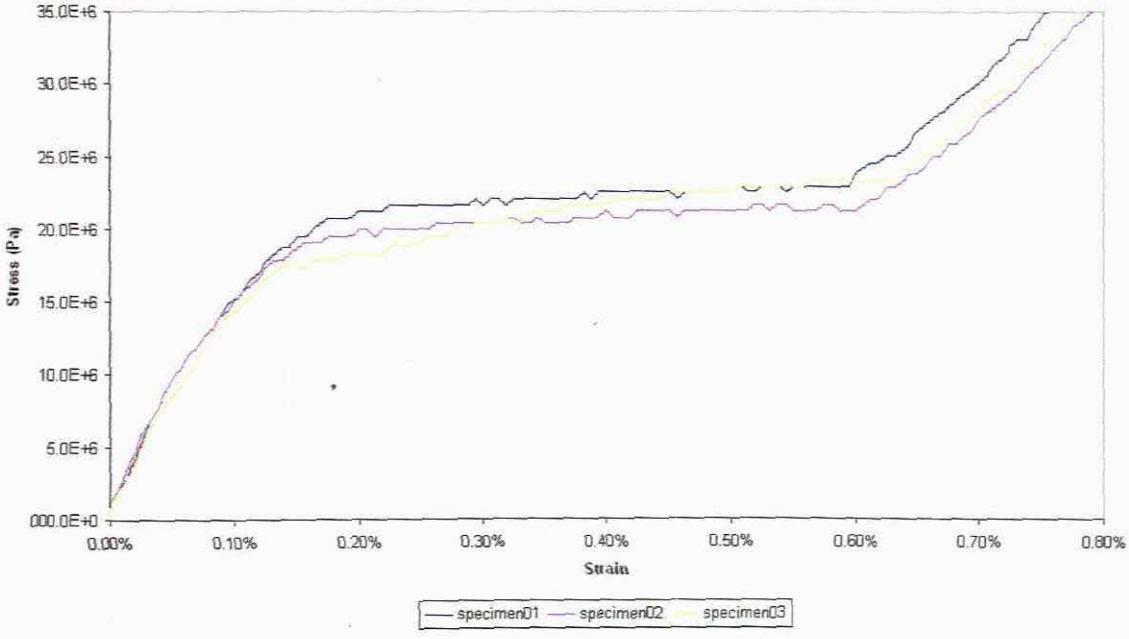
Loading Rate - 7mm/min - 150mm



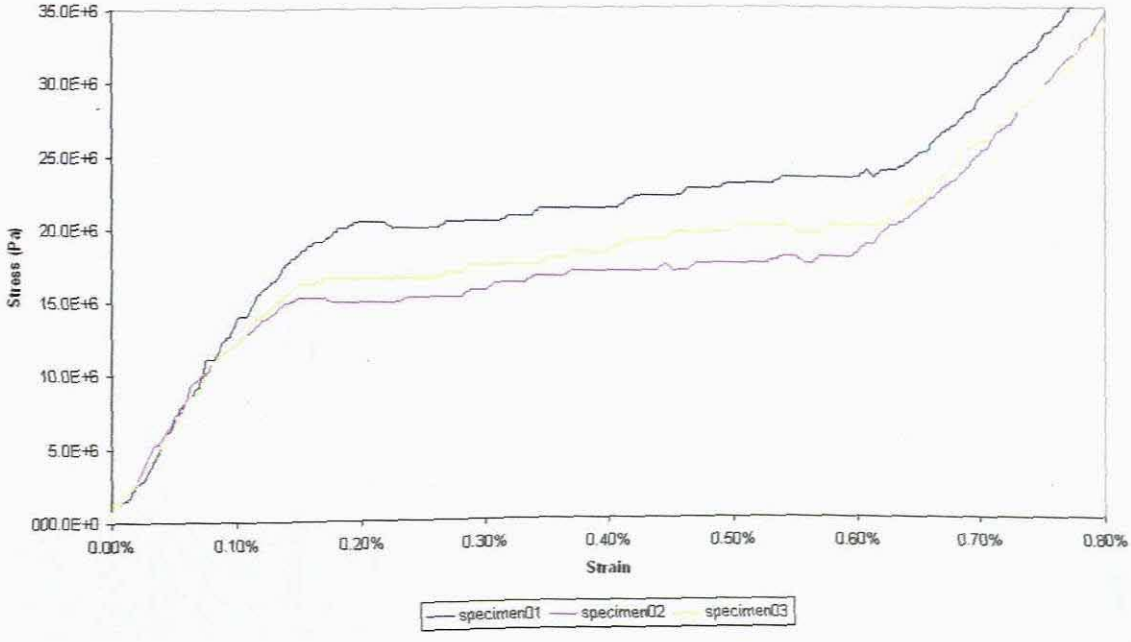
Loading Rate - 15mm/min - 150mm



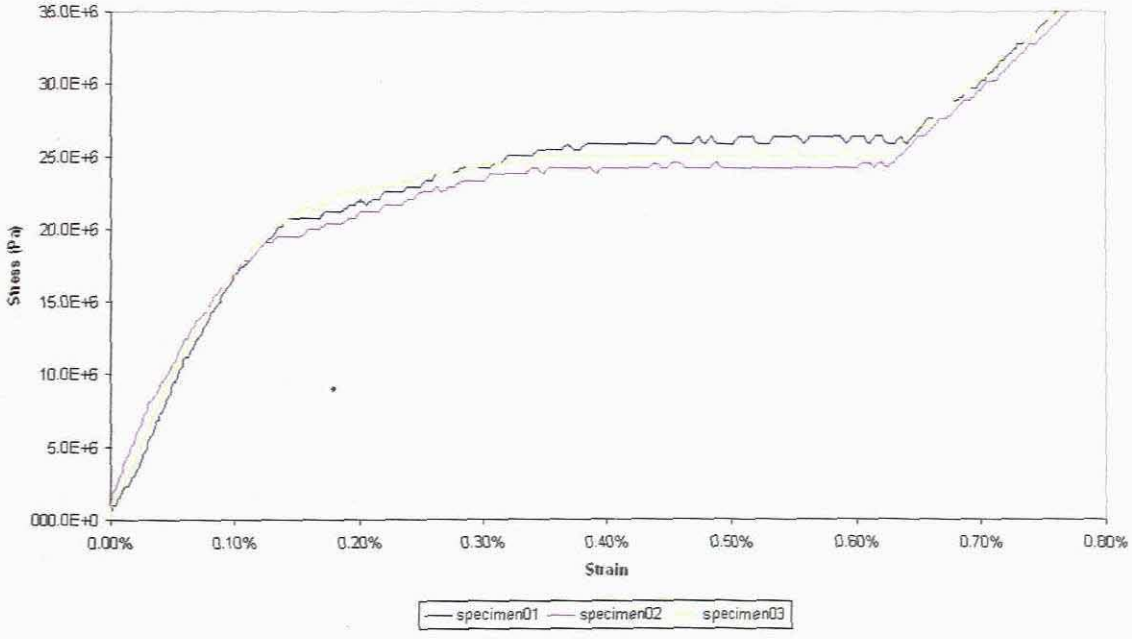
Loading Rate - 4mm/min - 200mm



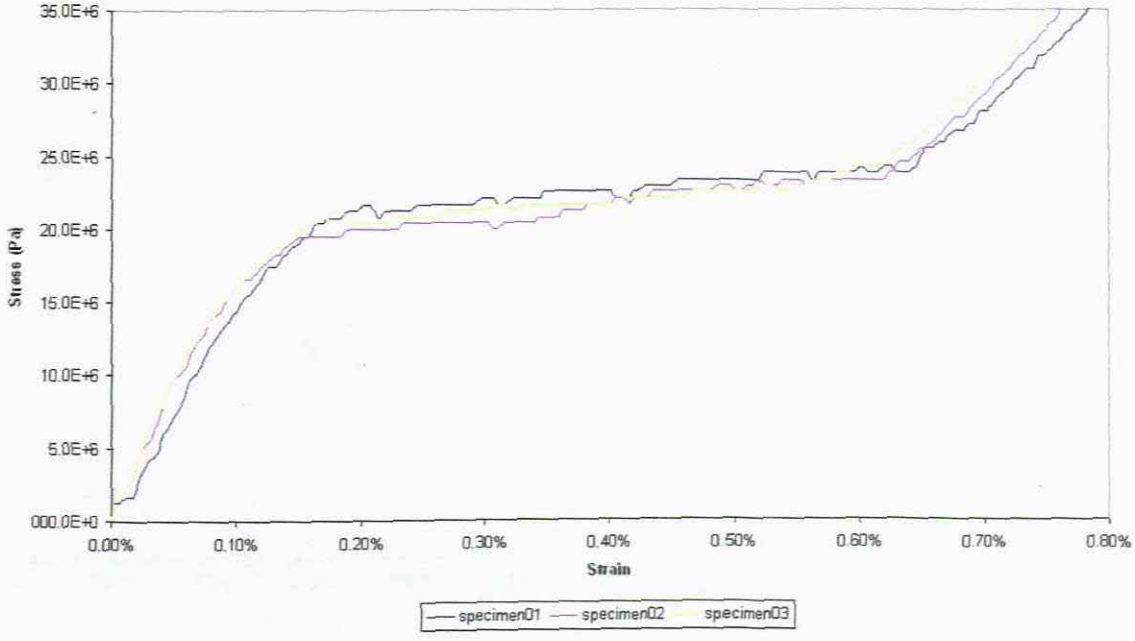
Loading Rate - 15mm/min - 200mm



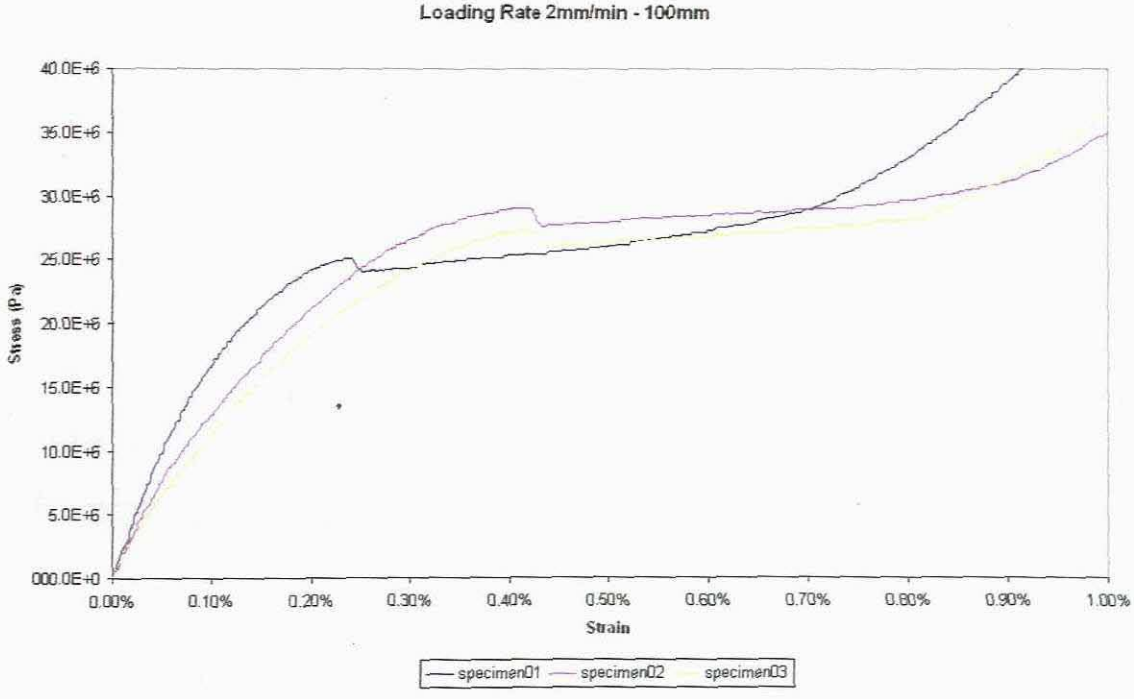
Loading Rate - 2mm/min - 250mm

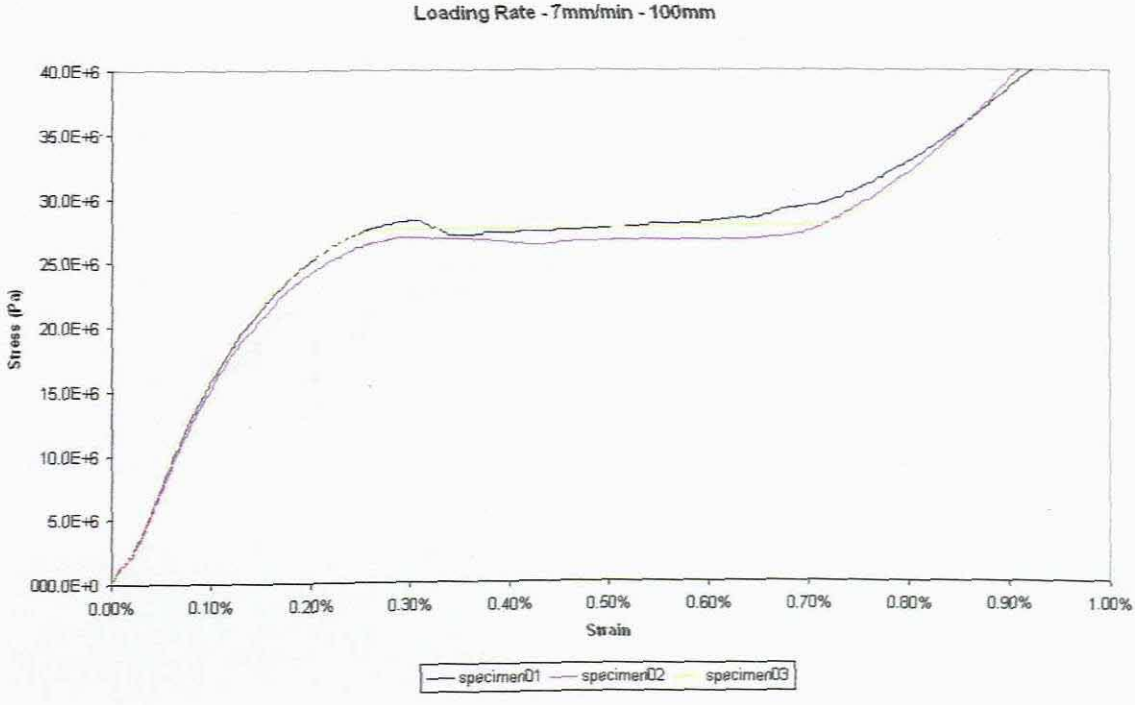


Loading Rate - 10mm/min - 250mm

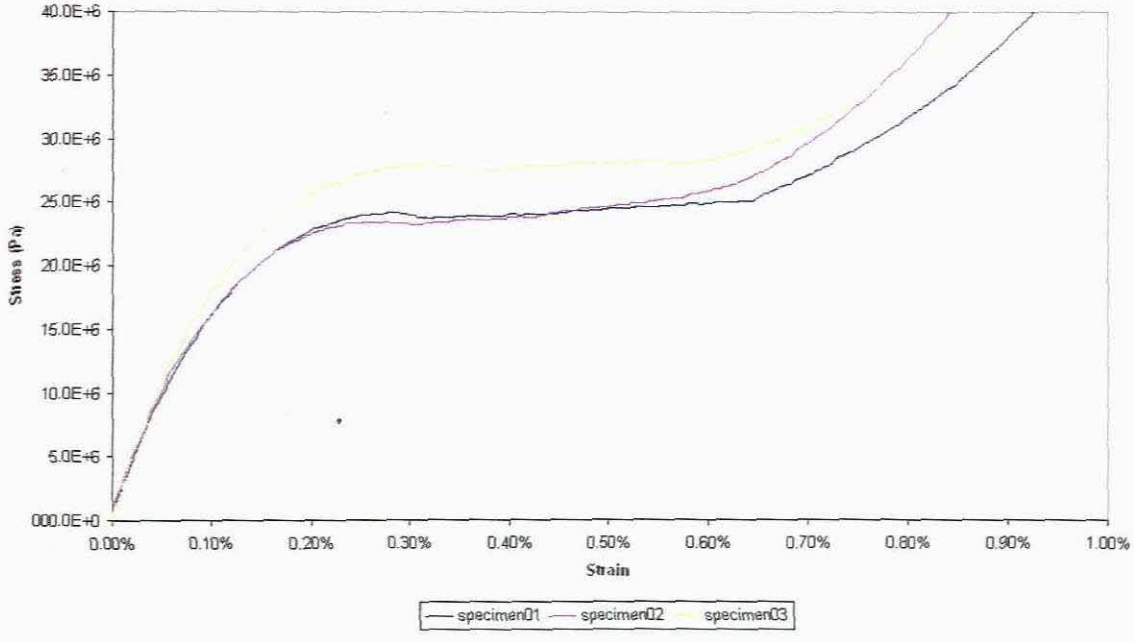


2. 2mm diameter NiTi Shape Memory Alloy Wires

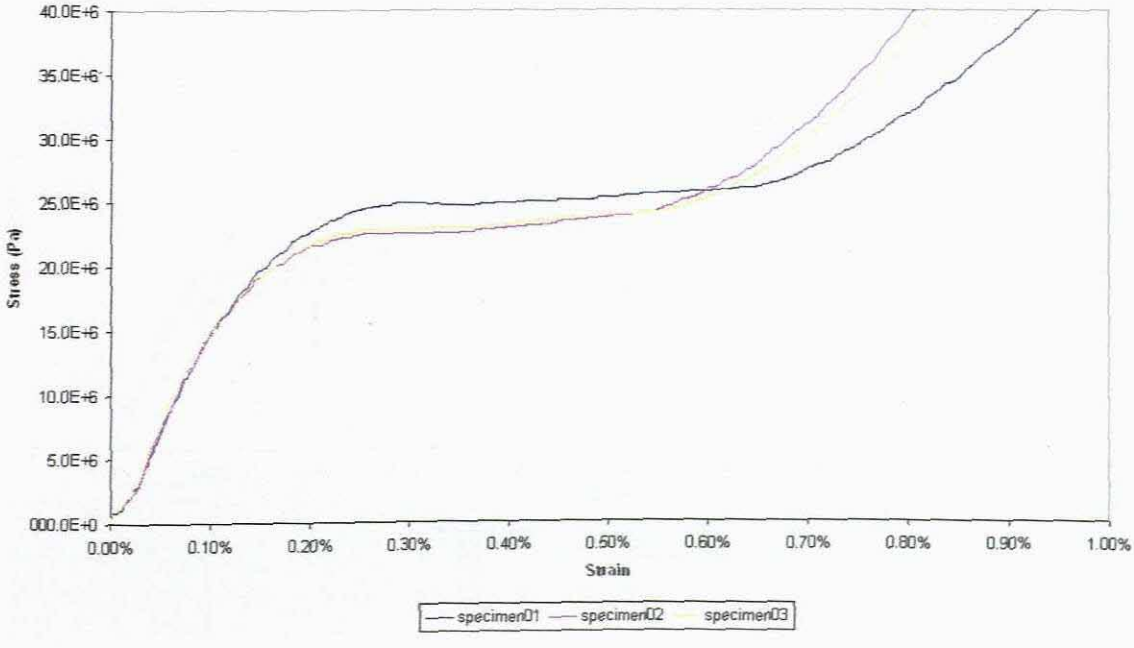


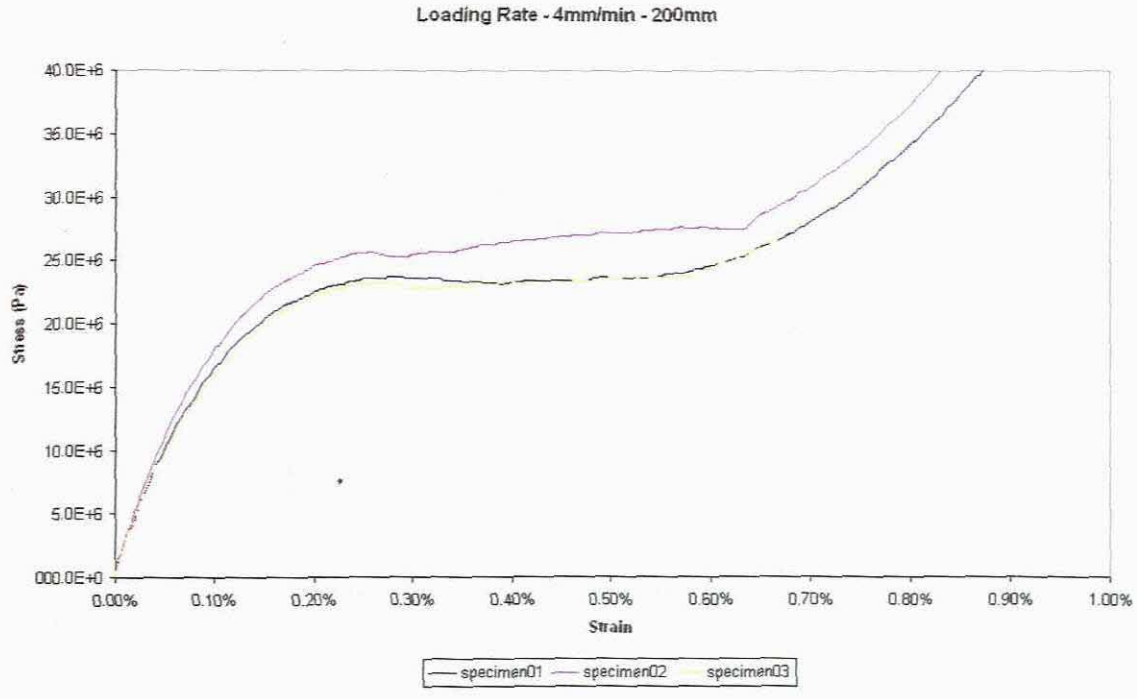


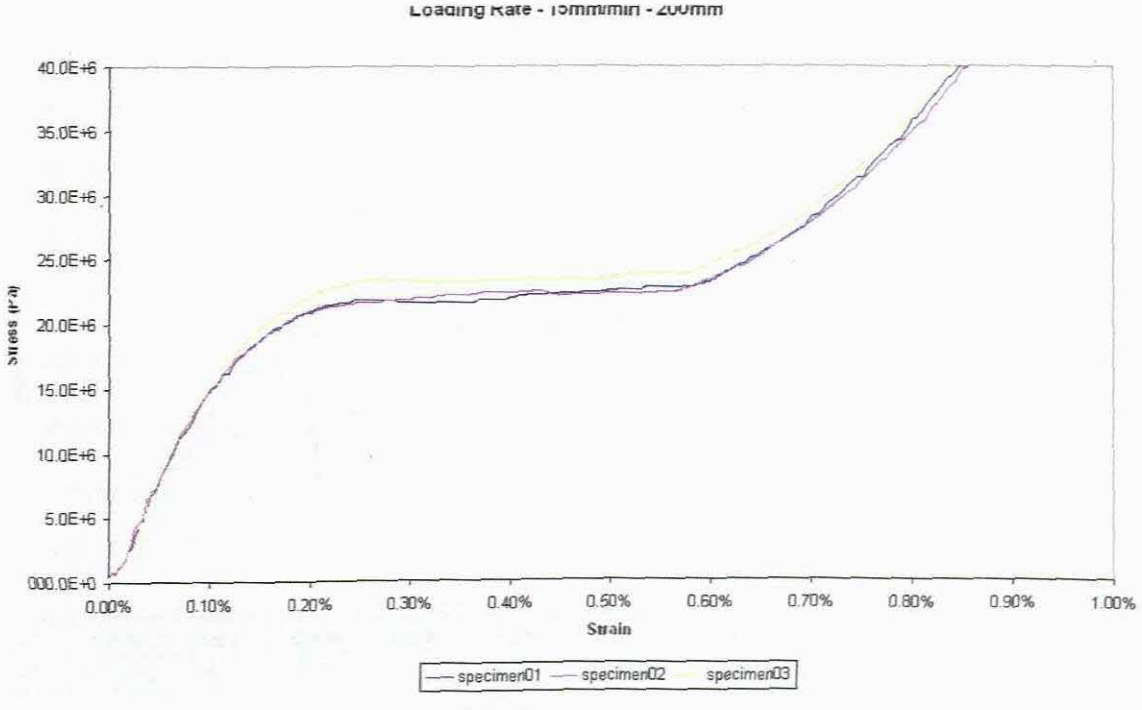
Loading Rate - 4mm/min - 150mm



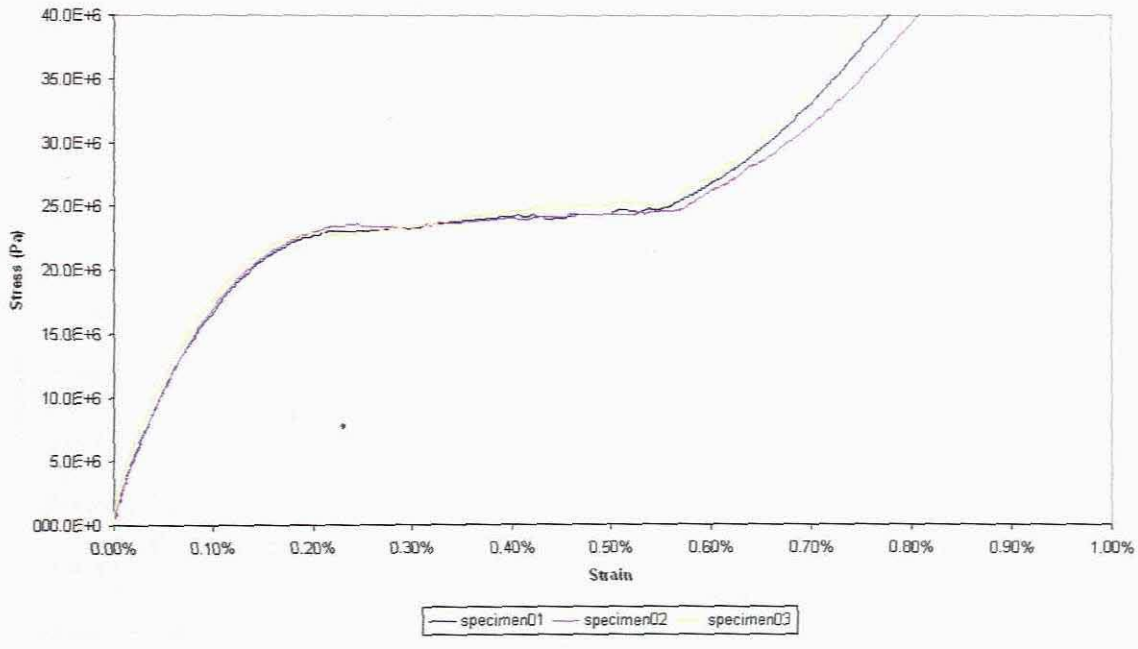
Loading Rate - 15mm/min - 150mm



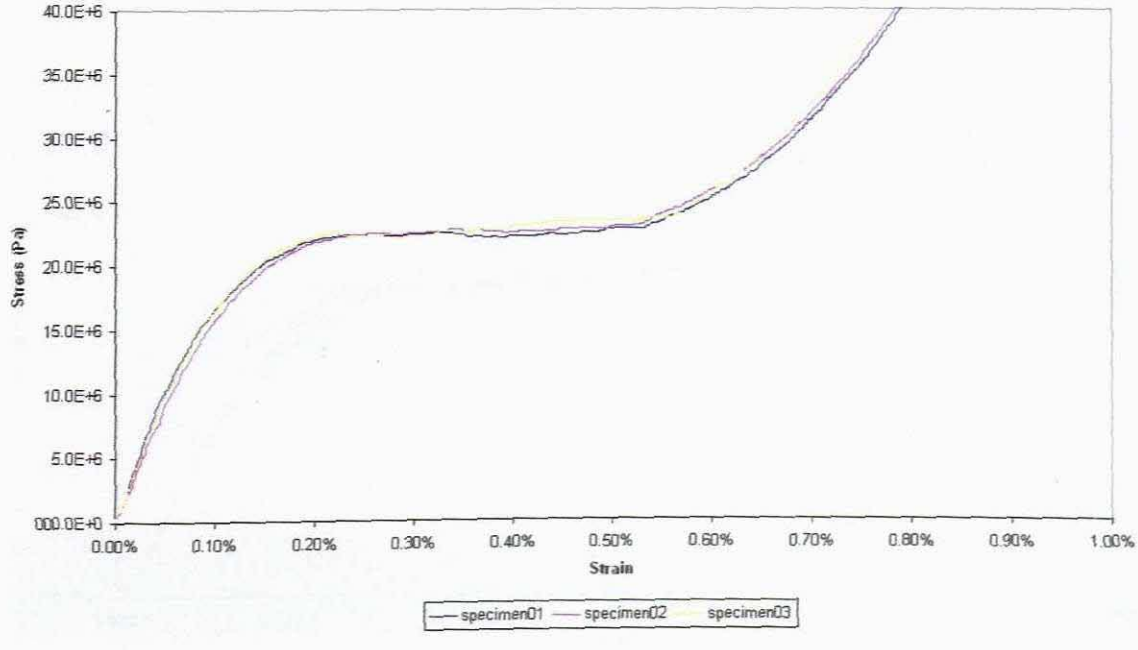

Loading Rate - 4mm/min - 200mm
40.0E+6
35.0E+6
30.0E+6
25.0E+6
20.0E+6
15.0E+6
10.0E+6
5.0E+6
000.0E+0
Stress (Pa)
0.00%
0.10%
0.20%
0.30%
0.40%
0.50%
0.60%
0.70%
0.80%
0.90%
1.00%
Strain
specimen01
specimen02
specimen03


Loading Rate - 15mm/min - 200mm
40.0E+6
35.0E+6
30.0E+6
25.0E+6
20.0E+6
15.0E+6
10.0E+6
5.0E+6
000.0E+0
Stress (Pa)
0.00%
0.10%
0.20%
0.30%
0.40%
0.50%
0.60%
0.70%
0.80%
0.90%
1.00%
Strain
specimen01
specimen02
specimen03

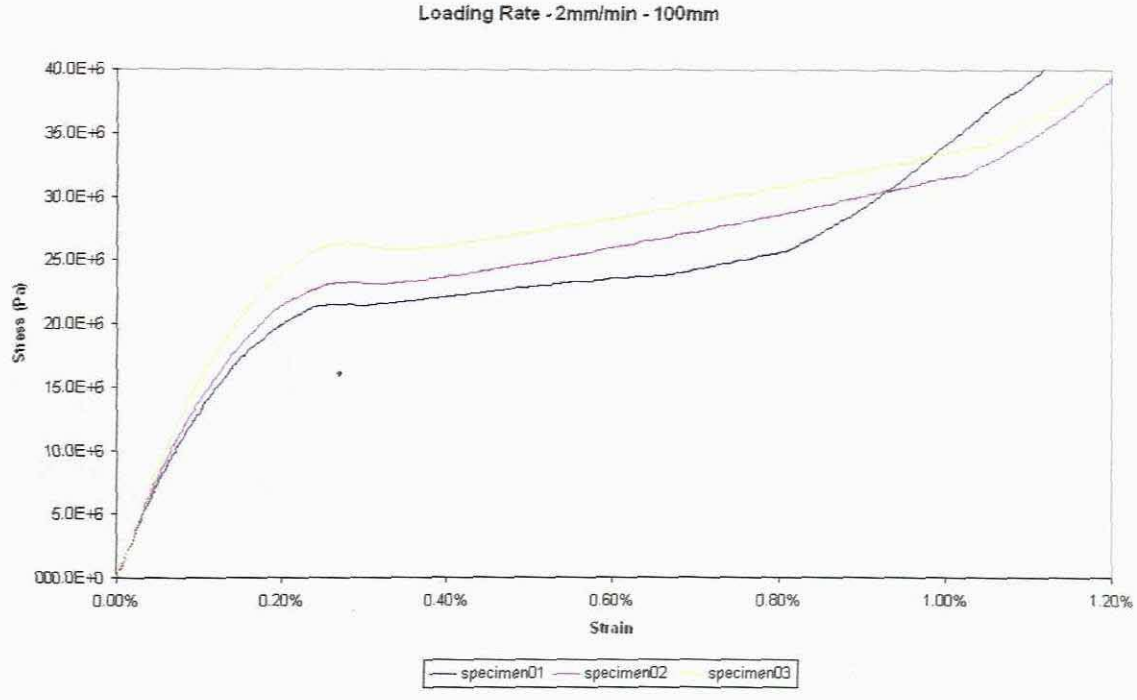
Loading Rate - 4mm/min - 250mm

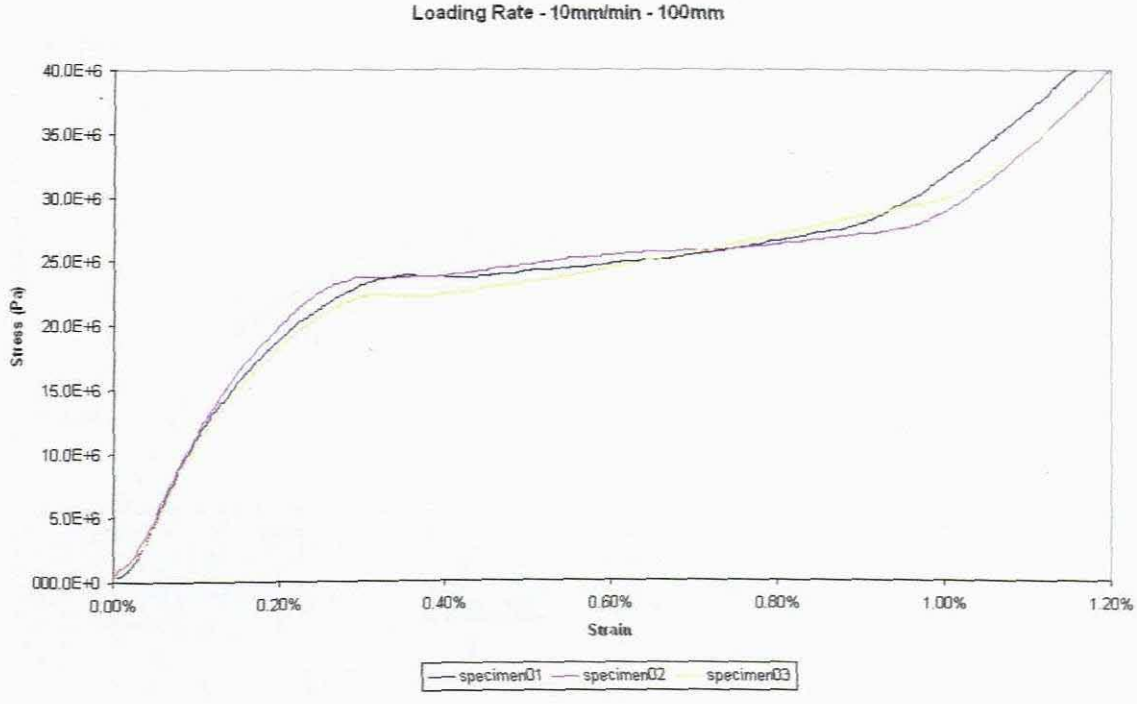


Loading Rate - 10mm/min - 250mm

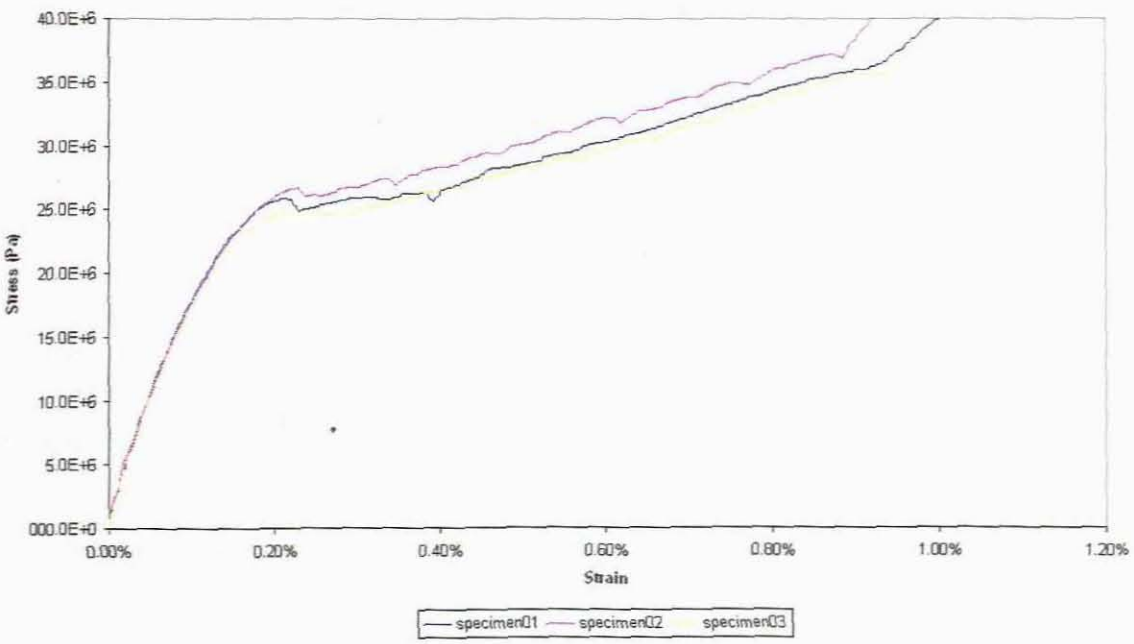


3. 3mm diameter NiTi Shape Memory Alloy Wires

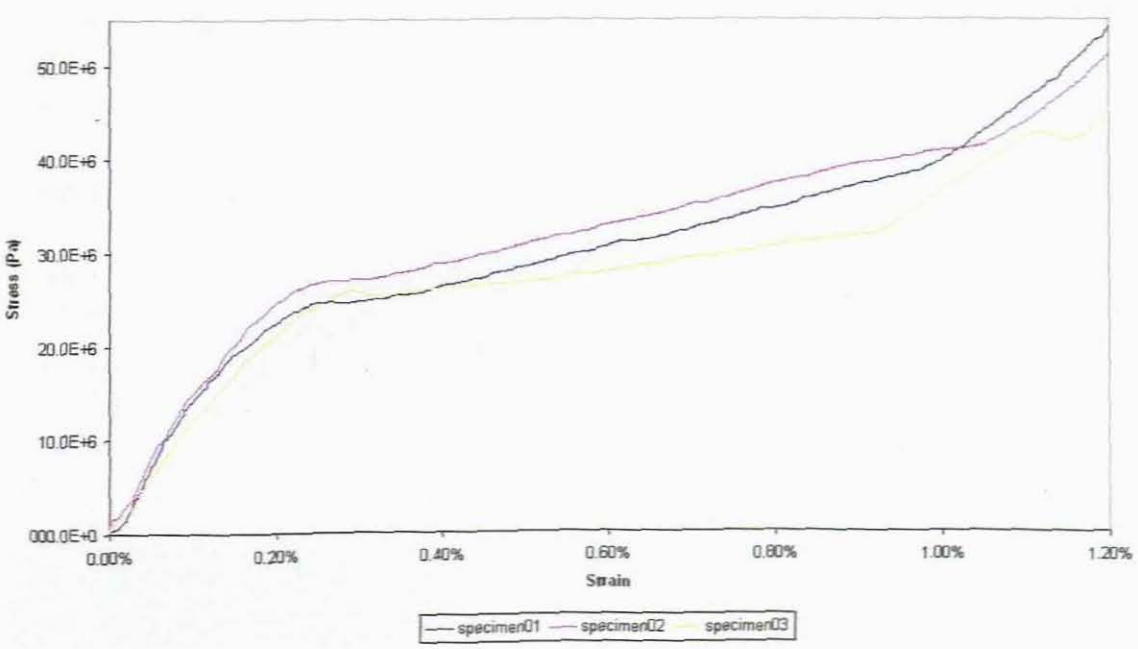




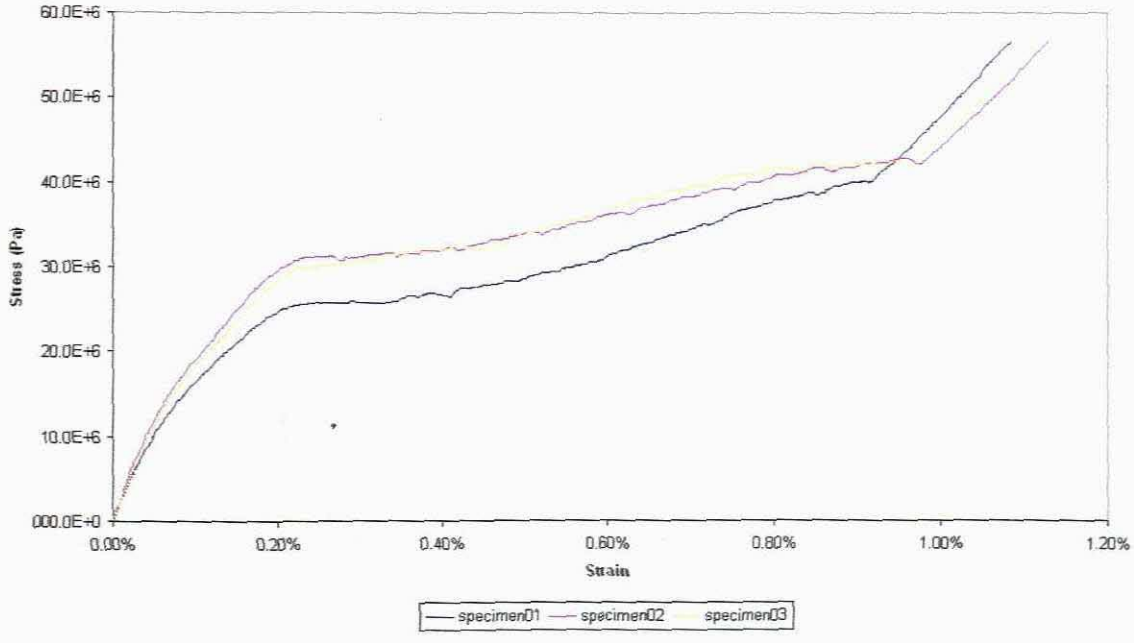
Loading Rate - 2mm/min - 150mm



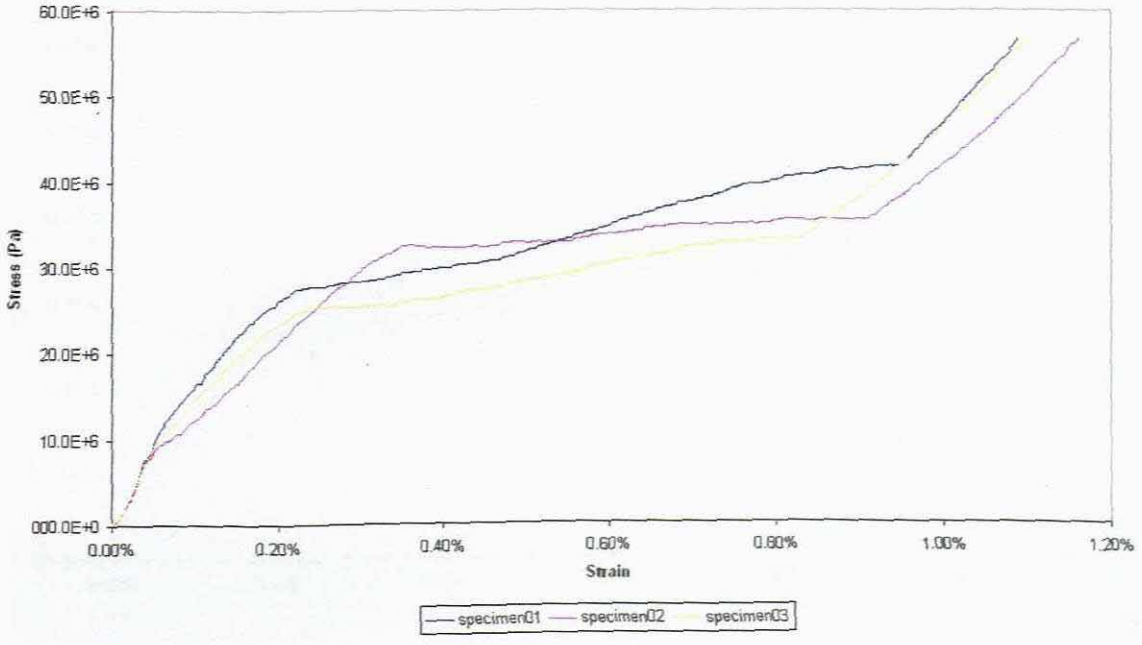
Loading Rate - 15mm/min - 150mm



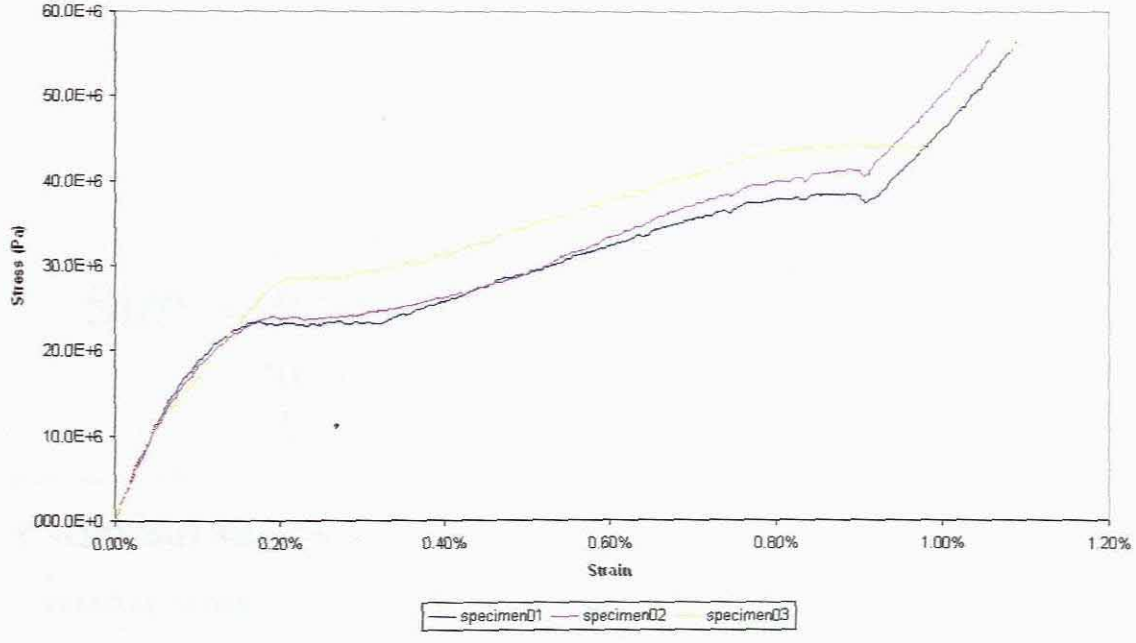
Loading Rate - 4mm/min - 200mm



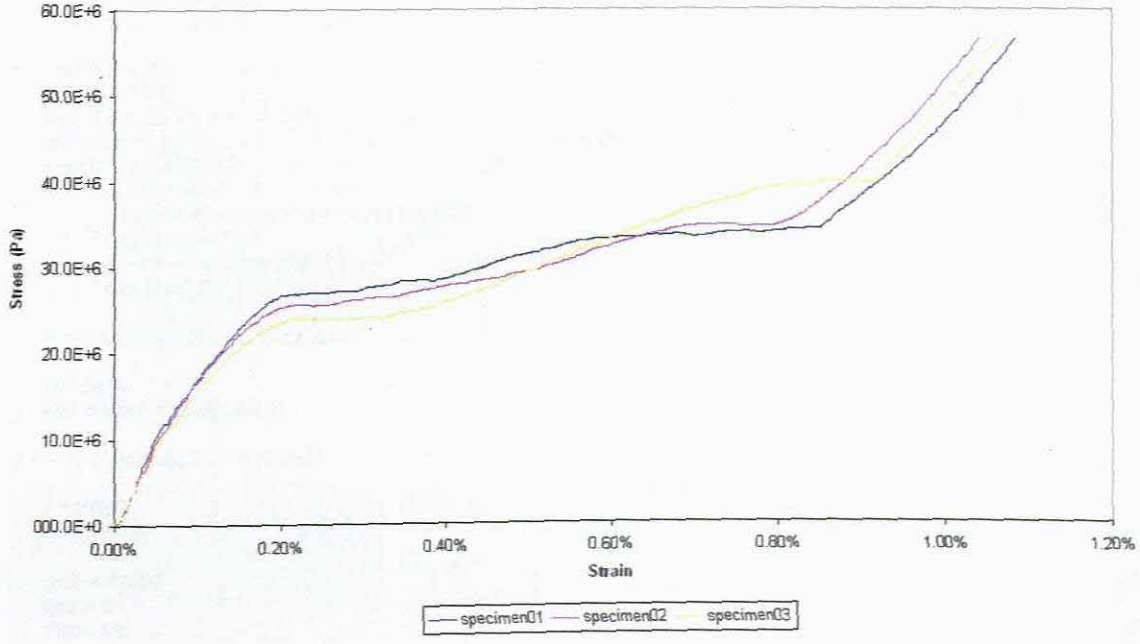
Loading Rate - 15mm/min - 200mm



Loading Rate - 4mm/min - 250mm



Loading Rate - 15mm/min - 250mm



Appendix B

FORTRAN Codes used for Numerical Simulations and RADAU5 routine as discussed in Chapter 3.

1. FORTRAN SMA Code

```
c   Simulation of NiTi wire with Sinisoidal Load

    program SMA01

    implicit none

    integer n,lwork,liwork
    parameter (n=3,lwork=4*n*n+12*n+20,liwork=3*n+20)

    integer ijac,mljac,imas,iout,itol,mlmas,mumas,idid,i,mujac,ipar
    real*8 y(n),work(lwork)
          integer iwork(liwork)
    real*8 t,h,rpar,rtol(n),atol(n),abstol,reltol

    real*8 IE,J,PSI,PSI0,EM,EA,DELTAS,DELTAL,DELTAR,k,Temp,V,OMEGA
    common /v/ IE,J,PSI,PSI0,EM,EA,DELTAS,DELTAL,DELTAR,k,Temp,V,OMEGA
    real*8 a,b,d,c,TR,FM,FA,EPM,EPA,SHM,SHA,ENM,ENA
    common /v/ a,b,d,c,TR,FM,FA,EPM,EPA,SHM,SHA,ENM,ENA
    real*8 THETA,MINUSA,AMINUS,PLUSSA,APLUSS
          real*8 PMA,PAM,PPA,PAP
          common /tranprob/ PMA,PAM,PPA,PAP
          real*8 tende,DEF
          common /v/ tende,DEF
          real*8 AP,DP

    external fcn,jac,mas,solout,radau5

    call opfile
    call initial(y,n,abstol,reltol)

c ---> Preparation for RADAU5

    t = 0.0d0
    h = 1.0d-5

    rpar = 1.0d-7
    ijac = 0
    mljac = n
    imas = 0
    iout = 1
    itol = 1
    imas = 0
    mlmas = 0
    mumas = 0
```

```
      idid = 0


      do i=1,liwork
              iwork(i) = 0
            enddo

            do i=1,lwork
          work (i) = 0.0d0
            enddo

            iwork(2)= 500000

            do i=1,n
            atol(i) = abstol
            rtol(i) = reltol
            enddo

      call radau5(n,fcn,t,y,tende,h,
     &     rtol,atol,itol,
     &     jac,ijac,mljac,mujac,
     &     mas,imas,mlmas,mumas,
     &     solout,iout,
     &     work,lwork,iwork,liwork,rpar,ipar,idid)

      call clfile

      stop '.....End of Calculation .....'
      end
c ***************************************************

            real*8 function AP(t)
            real*8 z,t
            z=0.058823529
            AP = 17.8d+04 * sin(z * t)
            return
            end

            real*8 function DP(t)
            real*8 z,t
            z=0.058823529
            DP = 350 * sin(z * t)
            return
            end
c ***************************************************

      subroutine initial(y,n,atol,rtol)

      implicit none

      integer nplot,n,i

      real*8 rtol,atol,y(n)

      real*8 IE,J,PSI,PSI0,EM,EA,DELTAS,DELTAL,DELTAR,k,Temp,V,OMEGA
      common /v/ IE,J,PSI,PSI0,EM,EA,DELTAS,DELTAL,DELTAR,k,Temp,V,OMEGA
      real*8 a,b,d,c,TR,FM,FA,EPM,EPA,SHM,SHA,ENM,ENA
      common /v/ a,b,d,c,TR,FM,FA,EPM,EPA,SHM,SHA,ENM,ENA
      real*8 THETA,MINUSA,AMINUS,PLUSSA,APLUSS
            real*8 PMA,PAM,PPA,PAP
            common /tranprob/ PMA,PAM,PPA,PAP
            real*8 tende,DEF
            common /v/ tende,DEF
            real*8 AP,DP
            integer n2

      read (20,1001)
      read (20,1001)
      read (20,1001)
```

```
      read (20,1001)
      read (20,1001) IE,J,PSI
      read (20,1001)
      read (20,1001)
      read (20,1001)
      read (20,1001) EM,EA,DELTAS,k
      read (20,1001)
      read (20,1001)
      read (20,1001)
      read (20,1001) Temp,V,OMEGA
      read (20,1001)
      read (20,1001)
      read (20,1001)
      read (20,1001) tende,DELTAL,DELTAR
      read (20,1001)
      read (20,1001)
      read (20,1001)
      read (20,1001) TR
      read (20,1001)
      read (20,1001)
      read (20,1001)
      read (20,1001) EPM,EPA,SHM,SHA
      read (20,1001)
      read (20,1001)
      read (20,1001)
      read (20,1001) ENM,ENA


ct ************************
c... Initial Inputs
            nplot = 0
            atol=1.0d-07
            rtol=1.0d-07

            print*, 'These are the inputs'
            print*
            print*, 'INTERFACIAL ENERGY=',IE
            print*, 'TOTAL TRANSFORMATION STRAIN=',J
            print*, 'POTENTIAL ENERGY AT BARRIER=',PSI
            print*, 'ELASTIC STIFFNESS OF MARTENSITE=',EM
            print*, 'ELASTIC STIFFNESS OF AUSTENITE=',EA
            print*, 'TRANSFORMATION STRAIN AT BARRIER=',DELTAS
            print*, 'TRANSFORMATION STRAIN AT SMOOTHING POINT L=',DELTAL
            print*, 'TRANSFORMATION STRAIN AT SMOOTHING POINT R=',DELTAR
            print*, 'BOLTZMANN CONSTANT=',k
            print*, 'ABSOLUTE TEMPERATURE=',Temp
            print*, 'TRANSFORMATION VOLUME=',V
            print*, 'TRANSFORMATION RATE=',OMEGA
            print*, 'Final Time=',tende
            print*, 'Reference Temperature=',TR
      print*, 'Reference Energy for Martensite=',EPM
            print*, 'Reference Energy for Austenite=',EPA
            print*, 'Specific Heat for Martensite=',SHM
            print*, 'Specific Heat for Austenite=',SHA
            print*, 'Reference Entropy for Martensite=',ENM
            print*, 'Reference Entropy for Austenite=',ENA
            print*

      y(1)=0.5
            y(2)=0.0
            y(3)=0.5
ct ************************
 1001 format (5d14.6)
ct ************************
            return
            end
c ***************************************************
      subroutine mas(n,am,lmas)

      return
```

```
      end
c ***************************************************
      subroutine jac(n,t,y,dfy,ldfy)

      return
      end
c ***************************************************
      subroutine fcn (n,t,y,f)

      implicit none

      integer n,i,n2
      real*8 t,y(n),f(n)

      real*8 IE,J,PSI,PSI0,EM,EA,DELTAS,DELTAL,DELTAR,k,Temp,V,OMEGA
      common /v/ IE,J,PSI,PSI0,EM,EA,DELTAS,DELTAL,DELTAR,k,Temp,V,OMEGA
      real*8 a,b,d,c,TR,FM,FA,EPM,EPA,SHM,SHA,ENM,ENA
      common /v/ a,b,d,c,TR,FM,FA,EPM,EPA,SHM,SHA,ENM,ENA
      real*8 THETA,MINUSA,AMINUS,PLUSSA,APLUSS
            real*8 PMA,PAM,PPA,PAP
            common /tranprob/ PMA,PAM,PPA,PAP
            real*8 tende,DEF
            common /v/ tende,DEF
            real*8 AP,DP
c
c           y(2)=(1-y(1)-y(3))
c
      a=-(DELTAL*EA + EM*(-DELTAR + J))/(2.*(-DELTAL + DELTAR))
            b=(DELTAL*(DELTAR*EA + EM*(-DELTAR + J)))/(-DELTAL + DELTAR)
            c=(EM*J**2)/2. - (DELTAR**2*(DELTAL*EA + EM*(-DELTAL + J)))/(2.*
     &(-DELTAL + DELTAR)) + SHM*(Temp - TR) + EPM - Temp*
     &(ENM + SHM*Log(Temp/TR))
      PSI0=(DELTAL*DELTAR*(-EA + EM))/2. - ((DELTAL + DELTAR)*EM*J)/2.
     &+ (EM*J**2)/2. - (SHA - SHM)*(Temp - TR) - EPA + EPM + Temp*
     &(ENA - ENM + (SHA - SHM)*Log(Temp/TR))
      FM=SHM*(Temp - TR) + EPM - Temp*(ENM + SHM*Log(Temp/TR))
      FA=SHA*(Temp - TR) + EPA - Temp*(ENA + SHA*Log(Temp/TR))
c
      MINUSA=b**2/(4.*a) - c - FM + IE + DELTAS*AP(t) - J*AP(t) + AP(t)
     &**2/(2.*EM) + PSI - 2*IE*y(2)
      AMINUS=b**2/(4.*a) - c - FA - IE + DELTAS*AP(t) + AP(t)**2/(2.*EA)
     &+ PSI - PSI0 + 2*IE*y(2)
      PLUSSA=b**2/(4.*a) - c - FM + IE - DELTAS*AP(t) + J*AP(t) + AP(t)
     &**2/(2.*EM) + PSI - 2*IE*y(2)
      APLUSS=b**2/(4.*a) - c - FA - IE - DELTAS*AP(t) + AP(t)**2/(2.*EA)
     &+ PSI - PSI0 + 2*IE*y(2)
c
            THETA = V/(k*Temp)
c
            PMA = (1/OMEGA)*EXP(-(MINUSA*THETA))
            PAM = (1/OMEGA)*EXP(-(AMINUS*THETA))
            PPA = (1/OMEGA)*EXP(-(PLUSSA*THETA))
            PAP = (1/OMEGA)*EXP(-(APLUSS*THETA))
c
      DEF=((y(1)/(EM*9.57E-3))+(y(3)/(EM*9.57E-3))+(y(2)/(EA*9.57E-3))
     &*(DP(t)))-(1.83)*(y(1)-y(3))
c
      f(1) = -y(1)*PMA+y(2)*PAM
      f(2) = PMA*y(1)-PAM*y(2)-PAP*y(2)+PPA*y(3)
            f(3) = -y(3)*PPA+y(2)*PAP
c
      return
      end
c ***************************************************************
      subroutine solout (nr,told,t,y,cont,lrc,n,rpar,ipar,irtrn)


      implicit none

      integer n,nr,irtrn,lrc,ipar,i
```

```
      real*8 told,t,y(n),cont(lrc),rpar

      real*8 IE,J,PSI,PSI0,EM,EA,DELTAS,DELTAL,DELTAR,k,Temp,V,OMEGA
      common /v/ IE,J,PSI,PSI0,EM,EA,DELTAS,DELTAL,DELTAR,k,Temp,V,OMEGA
      real*8 a,b,d,c,TR,FM,FA,EPM,EPA,SHM,SHA,ENM,ENA
      common /v/ a,b,d,c,TR,FM,FA,EPM,EPA,SHM,SHA,ENM,ENA
      real*8 THETA,MINUSA,AMINUS,PLUSSA,APLUSS
            real*8 PMA,PAM,PPA,PAP
            common /tranprob/ PMA,PAM,PPA,PAP
            real*8 tende,DEF
            common /v/ tende,DEF
            real*8 AP,DP

      write (30,3001) t,(y(i),i=1,n)
      write (31,3001) t,DEF
      write (32,3001) t,DP(t)
      write (33,3001) t,DEF,DP(t)
      write (34,3001) t,PMA,PAM,PPA,PAP
ct ************************************************************
3001  format(41e20.5)

      return
      end
c ************************************************************************
      subroutine opfile

      implicit none

      integer ioplot,iodat,iorand

      open (20,file = './Diminput12.inp',
     &      form = 'formatted',
     &      status = 'unknown',
     &      iostat = iodat )

      if (iodat.ne.0) then
        print*,'.....fehler beim oeffnen der Eingabedatei...'
        stop
      endif

      rewind (20)

      open (30,file = './Dimless50.out',
     &      form = 'formatted',
     &      status = 'unknown',
     &      iostat = ioplot )

      if (ioplot.ne.0) then
        print*,' .....fehler beim oeffnen der Plotdatei...'
        stop
      endif

      rewind (30)

      open (31,file = './Dimless51.out',
     &      form = 'formatted',
     &      status = 'unknown',
     &      iostat = ioplot )

      if (ioplot.ne.0) then
        print*,' .....fehler beim oeffnen der Plotdatei...'
        stop
      endif

      rewind (31)

      open (32,file = './Dimless52.out',
     &      form = 'formatted',
     &      status = 'unknown',
     &      iostat = ioplot )
```

```
      if (ioplot.ne.0) then
        print*,' .....fehler beim oeffnen der Plotdatei...'
        stop
      endif

      rewind (32)

      open (33,file = './Dimless53.out',
     &      form = 'formatted',
     &      status = 'unknown',
     &      iostat = ioplot )

      if (ioplot.ne.0) then
        print*,' .....fehler beim oeffnen der Plotdatei...'
        stop
      endif

      rewind (33)

      open (34,file = './Dimless54.out',
     &      form = 'formatted',
     &      status = 'unknown',
     &      iostat = ioplot )

      if (ioplot.ne.0) then
        print*,' .....fehler beim oeffnen der Plotdatei...'
        stop
      endif

      rewind (34)

      return
      end

c *****************************************************************

      subroutine clfile

      implicit none

      close (20)
      close (30)
            close (31)
            close (32)
            close (33)
            close (34)

      return
      end

c *****************************************************************
```

2. RADAU5 Routine

```
      SUBROUTINE RADAU5(N,FCN,X,Y,XEND,H,
     &            RTOL,ATOL,ITOL,
     &            JAC ,IJAC,MLJAC,MUJAC,
     &            MAS ,IMAS,MLMAS,MUMAS,
     &            SOLOUT,IOUT,
     &            WORK,LWORK,IWORK,LIWORK,RPAR,IPAR,IDID)
C ----------------------------------------------------------
C     NUMERICAL SOLUTION OF A STIFF (OR DIFFERENTIAL ALGEBRAIC)
C     SYSTEM OF FIRST ORDER ORDINARY DIFFERENTIAL EQUATIONS
C                M*Y'=F(X,Y).
C     THE SYSTEM CAN BE (LINEARLY) IMPLICIT (MASS-MATRIX M .NE. I)
C     OR EXPLICIT (M=I).
C     THE METHOD USED IS AN IMPLICIT RUNGE-KUTTA METHOD (RADAU IIA)
```

```
C   OF ORDER 5 WITH STEP SIZE CONTROL AND CONTINUOUS OUTPUT.
C   CF. SECTION IV.8
C
C   AUTHORS: E. HAIRER AND G. WANNER
C        UNIVERSITE DE GENEVE, DEPT. DE MATHEMATIQUES
C        CH-1211 GENEVE 24, SWITZERLAND
C        E-MAIL:  Ernst.Hairer@math.unige.ch
C                 Gerhard.Wanner@math.unige.ch
C
C   THIS CODE IS PART OF THE BOOK:
C     E. HAIRER AND G. WANNER, SOLVING ORDINARY DIFFERENTIAL
C     EQUATIONS II. STIFF AND DIFFERENTIAL-ALGEBRAIC PROBLEMS.
C     SPRINGER SERIES IN COMPUTATIONAL MATHEMATICS 14,
C     SPRINGER-VERLAG 1991, SECOND EDITION 1996.
C
C   VERSION OF JULY 9, 1996
C     (small correction April 14, 2000)
C
C   INPUT PARAMETERS
C   ----------
C   N        DIMENSION OF THE SYSTEM
C
C   FCN      NAME (EXTERNAL) OF SUBROUTINE COMPUTING THE
C            VALUE OF F(X,Y):
C              SUBROUTINE FCN(N,X,Y,F,RPAR,IPAR)
C              DOUBLE PRECISION X,Y(N),F(N)
C              F(1)=...  ETC.
C            RPAR, IPAR (SEE BELOW)
C
C   X        INITIAL X-VALUE
C
C   Y(N)     INITIAL VALUES FOR Y
C
C   XEND     FINAL X-VALUE (XEND-X MAY BE POSITIVE OR NEGATIVE)
C
C   H        INITIAL STEP SIZE GUESS;
C            FOR STIFF EQUATIONS WITH INITIAL TRANSIENT,
C            H=1.D0/(NORM OF F'), USUALLY 1.D-3 OR 1.D-5, IS GOOD.
C            THIS CHOICE IS NOT VERY IMPORTANT, THE STEP SIZE IS
C            QUICKLY ADAPTED. (IF H=0.D0, THE CODE PUTS H=1.D-6).
C
C   RTOL,ATOL  RELATIVE AND ABSOLUTE ERROR TOLERANCES. THEY
C            CAN BE BOTH SCALARS OR ELSE BOTH VECTORS OF LENGTH N.
C
C   ITOL     SWITCH FOR RTOL AND ATOL:
C             ITOL=0: BOTH RTOL AND ATOL ARE SCALARS.
C               THE CODE KEEPS, ROUGHLY, THE LOCAL ERROR OF
C               Y(I) BELOW RTOL*ABS(Y(I))+ATOL
C             ITOL=1: BOTH RTOL AND ATOL ARE VECTORS.
C               THE CODE KEEPS THE LOCAL ERROR OF Y(I) BELOW
C               RTOL(I)*ABS(Y(I))+ATOL(I).
C
C   JAC      NAME (EXTERNAL) OF THE SUBROUTINE WHICH COMPUTES
C            THE PARTIAL DERIVATIVES OF F(X,Y) WITH RESPECT TO Y
C            (THIS ROUTINE IS ONLY CALLED IF IJAC=1; SUPPLY
C            A DUMMY SUBROUTINE IN THE CASE IJAC=0).
C            FOR IJAC=1, THIS SUBROUTINE MUST HAVE THE FORM
C              SUBROUTINE JAC(N,X,Y,DFY,LDFY,RPAR,IPAR)
C              DOUBLE PRECISION X,Y(N),DFY(LDFY,N)
C              DFY(1,1)= ...
C            LDFY, THE COLUMN-LENGTH OF THE ARRAY, IS
C            FURNISHED BY THE CALLING PROGRAM.
C            IF (MLJAC.EQ.N) THE JACOBIAN IS SUPPOSED TO
C              BE FULL AND THE PARTIAL DERIVATIVES ARE
C              STORED IN DFY AS
C                DFY(I,J) = PARTIAL F(I) / PARTIAL Y(J)
C            ELSE, THE JACOBIAN IS TAKEN AS BANDED AND
C              THE PARTIAL DERIVATIVES ARE STORED
C              DIAGONAL-WISE AS
C                DFY(I-J+MUJAC+1,J) = PARTIAL F(I) / PARTIAL Y(J).
```

```
C
C   IJAC       SWITCH FOR THE COMPUTATION OF THE JACOBIAN:
C              IJAC=0: JACOBIAN IS COMPUTED INTERNALLY BY FINITE
C                DIFFERENCES, SUBROUTINE "JAC" IS NEVER CALLED.
C              IJAC=1: JACOBIAN IS SUPPLIED BY SUBROUTINE JAC.
C
C   MLJAC      SWITCH FOR THE BANDED STRUCTURE OF THE JACOBIAN:
C              MLJAC=N: JACOBIAN IS A FULL MATRIX. THE LINEAR
C                ALGEBRA IS DONE BY FULL-MATRIX GAUSS-ELIMINATION.
C              0<=MLJAC<N: MLJAC IS THE LOWER BANDWITH OF JACOBIAN
C                MATRIX (>= NUMBER OF NON-ZERO DIAGONALS BELOW
C                THE MAIN DIAGONAL).
C
C   MUJAC      UPPER BANDWITH OF JACOBIAN  MATRIX (>= NUMBER OF NON-
C          ZERO DIAGONALS ABOVE THE MAIN DIAGONAL).
C          NEED NOT BE DEFINED IF MLJAC=N.
C
C   ---  MAS,IMAS,MLMAS, AND MUMAS HAVE ANALOG MEANINGS      ---
C   ---  FOR THE "MASS MATRIX" (THE MATRIX "M" OF SECTION IV.8): -
C
C   MAS        NAME (EXTERNAL) OF SUBROUTINE COMPUTING THE MASS-
C          MATRIX M.
C          IF IMAS=0, THIS MATRIX IS ASSUMED TO BE THE IDENTITY
C          MATRIX AND NEEDS NOT TO BE DEFINED;
C          SUPPLY A DUMMY SUBROUTINE IN THIS CASE.
C          IF IMAS=1, THE SUBROUTINE MAS IS OF THE FORM
C            SUBROUTINE MAS(N,AM,LMAS,RPAR,IPAR)
C            DOUBLE PRECISION AM(LMAS,N)
C            AM(1,1)= ....
C            IF (MLMAS.EQ.N) THE MASS-MATRIX IS STORED
C            AS FULL MATRIX LIKE
C                 AM(I,J) = M(I,J)
C            ELSE, THE MATRIX IS TAKEN AS BANDED AND STORED
C            DIAGONAL-WISE AS
C                 AM(I-J+MUMAS+1,J) = M(I,J).
C
C   IMAS       GIVES INFORMATION ON THE MASS-MATRIX:
C              IMAS=0: M IS SUPPOSED TO BE THE IDENTITY
C                MATRIX, MAS IS NEVER CALLED.
C              IMAS=1: MASS-MATRIX  IS SUPPLIED.
C
C   MLMAS      SWITCH FOR THE BANDED STRUCTURE OF THE MASS-MATRIX:
C              MLMAS=N: THE FULL MATRIX CASE. THE LINEAR
C                ALGEBRA IS DONE BY FULL-MATRIX GAUSS-ELIMINATION.
C              0<=MLMAS<N: MLMAS IS THE LOWER BANDWITH OF THE
C                MATRIX (>= NUMBER OF NON-ZERO DIAGONALS BELOW
C                THE MAIN DIAGONAL).
C          MLMAS IS SUPPOSED TO BE .LE. MLJAC.
C
C   MUMAS      UPPER BANDWITH OF MASS-MATRIX (>= NUMBER OF NON-
C          ZERO DIAGONALS ABOVE THE MAIN DIAGONAL).
C          NEED NOT BE DEFINED IF MLMAS=N.
C          MUMAS IS SUPPOSED TO BE .LE. MUJAC.
C
C   SOLOUT     NAME (EXTERNAL) OF SUBROUTINE PROVIDING THE
C          NUMERICAL SOLUTION DURING INTEGRATION.
C          IF IOUT=1, IT IS CALLED AFTER EVERY SUCCESSFUL STEP.
C          SUPPLY A DUMMY SUBROUTINE IF IOUT=0.
C          IT MUST HAVE THE FORM
C            SUBROUTINE SOLOUT (NR,XOLD,X,Y,CONT,LRC,N,
C                       RPAR,IPAR,IRTRN)
C            DOUBLE PRECISION X,Y(N),CONT(LRC)
C            ....
C          SOLOUT FURNISHES THE SOLUTION "Y" AT THE NR-TH
C            GRID-POINT "X" (THEREBY THE INITIAL VALUE IS
C            THE FIRST GRID-POINT).
C          "XOLD" IS THE PRECEEDING GRID-POINT.
C          "IRTRN" SERVES TO INTERRUPT THE INTEGRATION. IF IRTRN
C            IS SET <0, RADAU5 RETURNS TO THE CALLING PROGRAM.
C
```

```
C      --- CONTINUOUS OUTPUT: ---
C          DURING CALLS TO "SOLOUT", A CONTINUOUS SOLUTION
C          FOR THE INTERVAL [XOLD,X] IS AVAILABLE THROUGH
C          THE FUNCTION
C                >>>  CONTR5(I,S,CONT,LRC)  <<<
C          WHICH PROVIDES AN APPROXIMATION TO THE I-TH
C          COMPONENT OF THE SOLUTION AT THE POINT S. THE VALUE
C          S SHOULD LIE IN THE INTERVAL [XOLD,X].
C          DO NOT CHANGE THE ENTRIES OF CONT(LRC), IF THE
C          DENSE OUTPUT FUNCTION IS USED.
C
C   IOUT     SWITCH FOR CALLING THE SUBROUTINE SOLOUT:
C            IOUT=0: SUBROUTINE IS NEVER CALLED
C            IOUT=1: SUBROUTINE IS AVAILABLE FOR OUTPUT.
C
C   WORK     ARRAY OF WORKING SPACE OF LENGTH "LWORK".
C          WORK(1), WORK(2),..., WORK(20) SERVE AS PARAMETERS
C          FOR THE CODE. FOR STANDARD USE OF THE CODE
C          WORK(1),...,WORK(20) MUST BE SET TO ZERO BEFORE
C          CALLING. SEE BELOW FOR A MORE SOPHISTICATED USE.
C          WORK(21),...,WORK(LWORK) SERVE AS WORKING SPACE
C          FOR ALL VECTORS AND MATRICES.
C          "LWORK" MUST BE AT LEAST
C                N*(LJAC+LMAS+3*LE+12)+20
C          WHERE
C            LJAC=N           IF MLJAC=N (FULL JACOBIAN)
C            LJAC=MLJAC+MUJAC+1  IF MLJAC<N (BANDED JAC.)
C          AND
C            LMAS=0           IF IMAS=0
C            LMAS=N           IF IMAS=1 AND MLMAS=N (FULL)
C            LMAS=MLMAS+MUMAS+1  IF MLMAS<N (BANDED MASS-M.)
C          AND
C            LE=N           IF MLJAC=N (FULL JACOBIAN)
C            LE=2*MLJAC+MUJAC+1 IF MLJAC<N (BANDED JAC.)
C
C          IN THE USUAL CASE WHERE THE JACOBIAN IS FULL AND THE
C          MASS-MATRIX IS THE INDENTITY (IMAS=0), THE MINIMUM
C          STORAGE REQUIREMENT IS
C                LWORK = 4*N*N+12*N+20.
C          IF IWORK(9)=M1>0 THEN "LWORK" MUST BE AT LEAST
C               N*(LJAC+12)+(N-M1)*(LMAS+3*LE)+20
C          WHERE IN THE DEFINITIONS OF LJAC, LMAS AND LE THE
C          NUMBER N CAN BE REPLACED BY N-M1.
C
C   LWORK    DECLARED LENGTH OF ARRAY "WORK".
C
C   IWORK    INTEGER WORKING SPACE OF LENGTH "LIWORK".
C          IWORK(1),IWORK(2),...,IWORK(20) SERVE AS PARAMETERS
C          FOR THE CODE. FOR STANDARD USE, SET IWORK(1),...,
C          IWORK(20) TO ZERO BEFORE CALLING.
C          IWORK(21),...,IWORK(LIWORK) SERVE AS WORKING AREA.
C          "LIWORK" MUST BE AT LEAST 3*N+20.
C
C   LIWORK   DECLARED LENGTH OF ARRAY "IWORK".
C
C   RPAR, IPAR  REAL AND INTEGER PARAMETERS (OR PARAMETER ARRAYS) WHICH
C          CAN BE USED FOR COMMUNICATION BETWEEN YOUR CALLING
C          PROGRAM AND THE FCN, JAC, MAS, SOLOUT SUBROUTINES.
C
C-----------------------------------------------------------
C
C   SOPHISTICATED SETTING OF PARAMETERS
C   -----------------------------
C        SEVERAL PARAMETERS OF THE CODE ARE TUNED TO MAKE IT WORK
C        WELL. THEY MAY BE DEFINED BY SETTING WORK(1),...
C        AS WELL AS IWORK(1),... DIFFERENT FROM ZERO.
C        FOR ZERO INPUT, THE CODE CHOOSES DEFAULT VALUES:
C
C   IWORK(1)  IF IWORK(1).NE.0, THE CODE TRANSFORMS THE JACOBIAN
C        MATRIX TO HESSENBERG FORM. THIS IS PARTICULARLY
```

```
C         ADVANTAGEOUS FOR LARGE SYSTEMS WITH FULL JACOBIAN.
C         IT DOES NOT WORK FOR BANDED JACOBIAN (MLJAC<N)
C         AND NOT FOR IMPLICIT SYSTEMS (IMAS=1).
C
C  IWORK(2)  THIS IS THE MAXIMAL NUMBER OF ALLOWED STEPS.
C         THE DEFAULT VALUE (FOR IWORK(2)=0) IS 100000.
C
C  IWORK(3)  THE MAXIMUM NUMBER OF NEWTON ITERATIONS FOR THE
C         SOLUTION OF THE IMPLICIT SYSTEM IN EACH STEP.
C         THE DEFAULT VALUE (FOR IWORK(3)=0) IS 7.
C
C  IWORK(4)  IF IWORK(4).EQ.0 THE EXTRAPOLATED COLLOCATION SOLUTION
C         IS TAKEN AS STARTING VALUE FOR NEWTON'S METHOD.
C         IF IWORK(4).NE.0 ZERO STARTING VALUES ARE USED.
C         THE LATTER IS RECOMMENDED IF NEWTON'S METHOD HAS
C         DIFFICULTIES WITH CONVERGENCE (THIS IS THE CASE WHEN
C         NSTEP IS LARGER THAN NACCPT + NREJCT; SEE OUTPUT PARAM.).
C         DEFAULT IS IWORK(4)=0.
C
C    THE FOLLOWING 3 PARAMETERS ARE IMPORTANT FOR
C    DIFFERENTIAL-ALGEBRAIC SYSTEMS OF INDEX > 1.
C    THE FUNCTION-SUBROUTINE SHOULD BE WRITTEN SUCH THAT
C    THE INDEX 1,2,3 VARIABLES APPEAR IN THIS ORDER.
C    IN ESTIMATING THE ERROR THE INDEX 2 VARIABLES ARE
C    MULTIPLIED BY H, THE INDEX 3 VARIABLES BY H**2.
C
C  IWORK(5)  DIMENSION OF THE INDEX 1 VARIABLES (MUST BE > 0). FOR
C         ODE'S THIS EQUALS THE DIMENSION OF THE SYSTEM.
C         DEFAULT IWORK(5)=N.
C
C  IWORK(6)  DIMENSION OF THE INDEX 2 VARIABLES. DEFAULT IWORK(6)=0.
C
C  IWORK(7)  DIMENSION OF THE INDEX 3 VARIABLES. DEFAULT IWORK(7)=0.
C
C  IWORK(8)  SWITCH FOR STEP SIZE STRATEGY
C         IF IWORK(8).EQ.1  MOD. PREDICTIVE CONTROLLER (GUSTAFSSON)
C         IF IWORK(8).EQ.2  CLASSICAL STEP SIZE CONTROL
C         THE DEFAULT VALUE (FOR IWORK(8)=0) IS IWORK(8)=1.
C         THE CHOICE IWORK(8).EQ.1 SEEMS TO PRODUCE SAFER RESULTS;
C         FOR SIMPLE PROBLEMS, THE CHOICE IWORK(8).EQ.2 PRODUCES
C         OFTEN SLIGHTLY FASTER RUNS
C
C    IF THE DIFFERENTIAL SYSTEM HAS THE SPECIAL STRUCTURE THAT
C       Y(I)' = Y(I+M2)   FOR  I=1,...,M1,
C    WITH M1 A MULTIPLE OF M2, A SUBSTANTIAL GAIN IN COMPUTERTIME
C    CAN BE ACHIEVED BY SETTING THE PARAMETERS IWORK(9) AND IWORK(10).
C    E.G., FOR SECOND ORDER SYSTEMS P'=V, V'=G(P,V), WHERE P AND V ARE
C    VECTORS OF DIMENSION N/2, ONE HAS TO PUT M1=M2=N/2.
C    FOR M1>0 SOME OF THE INPUT PARAMETERS HAVE DIFFERENT MEANINGS:
C    - JAC: ONLY THE ELEMENTS OF THE NON-TRIVIAL PART OF THE
C         JACOBIAN HAVE TO BE STORED
C         IF (MLJAC.EQ.N-M1) THE JACOBIAN IS SUPPOSED TO BE FULL
C           DFY(I,J) = PARTIAL F(I+M1) / PARTIAL Y(J)
C           FOR I=1,N-M1 AND J=1,N.
C         ELSE, THE JACOBIAN IS BANDED ( M1 = M2 * MM )
C           DFY(I-J+MUJAC+1,J+K*M2) = PARTIAL F(I+M1) / PARTIAL Y(J+K*M2)
C           FOR I=1,MLJAC+MUJAC+1 AND J=1,M2 AND K=0,MM.
C    - MLJAC: MLJAC=N-M1: IF THE NON-TRIVIAL PART OF THE JACOBIAN IS FULL
C         0<=MLJAC<N-M1: IF THE (MM+1) SUBMATRICES (FOR K=0,MM)
C             PARTIAL F(I+M1) / PARTIAL Y(J+K*M2),  I,J=1,M2
C            ARE BANDED, MLJAC IS THE MAXIMAL LOWER BANDWIDTH
C            OF THESE MM+1 SUBMATRICES
C    - MUJAC: MAXIMAL UPPER BANDWIDTH OF THESE MM+1 SUBMATRICES
C         NEED NOT BE DEFINED IF MLJAC=N-M1
C    - MAS: IF IMAS=0 THIS MATRIX IS ASSUMED TO BE THE IDENTITY AND
C         NEED NOT BE DEFINED. SUPPLY A DUMMY SUBROUTINE IN THIS CASE.
C         IT IS ASSUMED THAT ONLY THE ELEMENTS OF RIGHT LOWER BLOCK OF
C         DIMENSION N-M1 DIFFER FROM THAT OF THE IDENTITY MATRIX.
C         IF (MLMAS.EQ.N-M1) THIS SUBMATRIX IS SUPPOSED TO BE FULL
C           AM(I,J) = M(I+M1,J+M1)    FOR I=1,N-M1 AND J=1,N-M1.
```

```
C          ELSE, THE MASS MATRIX IS BANDED
C            AM(I-J+MUMAS+1,J) = M(I+M1,J+M1)
C     - MLMAS: MLMAS=N-M1: IF THE NON-TRIVIAL PART OF M IS FULL
C          0<=MLMAS<N-M1: LOWER BANDWIDTH OF THE MASS MATRIX
C     - MUMAS: UPPER BANDWIDTH OF THE MASS MATRIX
C          NEED NOT BE DEFINED IF MLMAS=N-M1
C
C   IWORK(9)  THE VALUE OF M1.  DEFAULT M1=0.
C
C   IWORK(10) THE VALUE OF M2.  DEFAULT M2=M1.
C
C-----------
C
C   WORK(1)  UROUND, THE ROUNDING UNIT, DEFAULT 1.D-16.
C
C   WORK(2)  THE SAFETY FACTOR IN STEP SIZE PREDICTION,
C          DEFAULT 0.9D0.
C
C   WORK(3)  DECIDES WHETHER THE JACOBIAN SHOULD BE RECOMPUTED;
C          INCREASE WORK(3), TO 0.1 SAY, WHEN JACOBIAN EVALUATIONS
C          ARE COSTLY. FOR SMALL SYSTEMS WORK(3) SHOULD BE SMALLER
C          (0.001D0, SAY). NEGATIV WORK(3) FORCES THE CODE TO
C          COMPUTE THE JACOBIAN AFTER EVERY ACCEPTED STEP.
C          DEFAULT 0.001D0.
C
C   WORK(4)  STOPPING CRITERION FOR NEWTON'S METHOD, USUALLY CHOSEN <1.
C          SMALLER VALUES OF WORK(4) MAKE THE CODE SLOWER, BUT SAFER.
C          DEFAULT MIN(0.03D0,RTOL(1)**0.5D0)
C
C   WORK(5) AND WORK(6) : IF WORK(5) < HNEW/HOLD < WORK(6), THEN THE
C          STEP SIZE IS NOT CHANGED. THIS SAVES, TOGETHER WITH A
C          LARGE WORK(3), LU-DECOMPOSITIONS AND COMPUTING TIME FOR
C          LARGE SYSTEMS. FOR SMALL SYSTEMS ONE MAY HAVE
C          WORK(5)=1.D0, WORK(6)=1.2D0, FOR LARGE FULL SYSTEMS
C          WORK(5)=0.99D0, WORK(6)=2.D0 MIGHT BE GOOD.
C          DEFAULTS WORK(5)=1.D0, WORK(6)=1.2D0 .
C
C   WORK(7)  MAXIMAL STEP SIZE, DEFAULT XEND-X.
C
C   WORK(8), WORK(9)  PARAMETERS FOR STEP SIZE SELECTION
C          THE NEW STEP SIZE IS CHOSEN SUBJECT TO THE RESTRICTION
C            WORK(8) <= HNEW/HOLD <= WORK(9)
C          DEFAULT VALUES: WORK(8)=0.2D0, WORK(9)=8.D0
C
C-----------------------------------------------------
C
C   OUTPUT PARAMETERS
C   -----------------
C   X        X-VALUE FOR WHICH THE SOLUTION HAS BEEN COMPUTED
C          (AFTER SUCCESSFUL RETURN X=XEND).
C
C   Y(N)      NUMERICAL SOLUTION AT X
C
C   H        PREDICTED STEP SIZE OF THE LAST ACCEPTED STEP
C
C   IDID      REPORTS ON SUCCESSFULNESS UPON RETURN:
C            IDID= 1  COMPUTATION SUCCESSFUL,
C            IDID= 2  COMPUT. SUCCESSFUL (INTERRUPTED BY SOLOUT)
C            IDID=-1  INPUT IS NOT CONSISTENT,
C            IDID=-2  LARGER NMAX IS NEEDED,
C            IDID=-3  STEP SIZE BECOMES TOO SMALL,
C            IDID=-4  MATRIX IS REPEATEDLY SINGULAR.
C
C  IWORK(14) NFCN    NUMBER OF FUNCTION EVALUATIONS (THOSE FOR NUMERICAL
C              EVALUATION OF THE JACOBIAN ARE NOT COUNTED)
C  IWORK(15) NJAC    NUMBER OF JACOBIAN EVALUATIONS (EITHER ANALYTICALLY
C              OR NUMERICALLY)
C  IWORK(16) NSTEP   NUMBER OF COMPUTED STEPS
C  IWORK(17) NACCPT  NUMBER OF ACCEPTED STEPS
C  IWORK(18) NREJCT  NUMBER OF REJECTED STEPS (DUE TO ERROR TEST),
```

```
C               (STEP REJECTIONS IN THE FIRST STEP ARE NOT COUNTED)
C  IWORK(19) NDEC   NUMBER OF LU-DECOMPOSITIONS OF BOTH MATRICES
C  IWORK(20) NSOL   NUMBER OF FORWARD-BACKWARD SUBSTITUTIONS, OF BOTH
C               SYSTEMS; THE NSTEP FORWARD-BACKWARD SUBSTITUTIONS,
C               NEEDED FOR STEP SIZE SELECTION, ARE NOT COUNTED
C-----------------------------------------------------------
C *** *** *** *** *** *** *** *** *** *** *** *** ***
C       DECLARATIONS
C *** *** *** *** *** *** *** *** *** *** *** *** ***
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION Y(N),ATOL(*),RTOL(*),WORK(LWORK),IWORK(LIWORK)
      DIMENSION RPAR(*),IPAR(*)
      LOGICAL IMPLCT,JBAND,ARRET,STARTN,PRED
      EXTERNAL FCN,JAC,MAS,SOLOUT
C *** *** *** *** *** *** ***
C      SETTING THE PARAMETERS
C *** *** *** *** *** *** ***
      NFCN=0
      NJAC=0
      NSTEP=0
      NACCPT=0
      NREJCT=0
      NDEC=0
      NSOL=0
      ARRET=.FALSE.
C ---- UROUND   SMALLEST NUMBER SATISFYING 1.0D0+UROUND>1.0D0
      IF (WORK(1).EQ.0.0D0) THEN
        UROUND=1.0D-16
      ELSE
        UROUND=WORK(1)
        IF (UROUND.LE.1.0D-19.OR.UROUND.GE.1.0D0) THEN
          WRITE(6,*)' COEFFICIENTS HAVE 20 DIGITS, UROUND=',WORK(1)
          ARRET=.TRUE.
        END IF
      END IF
C ---- CHECK AND CHANGE THE TOLERANCES
      EXPM=2.0D0/3.0D0
      IF (ITOL.EQ.0) THEN
        IF (ATOL(1).LE.0.D0.OR.RTOL(1).LE.10.D0*UROUND) THEN
          WRITE (6,*) ' TOLERANCES ARE TOO SMALL'
          ARRET=.TRUE.
        ELSE
          QUOT=ATOL(1)/RTOL(1)
          RTOL(1)=0.1D0*RTOL(1)**EXPM
          ATOL(1)=RTOL(1)*QUOT
        END IF
      ELSE
        DO I=1,N
        IF (ATOL(I).LE.0.D0.OR.RTOL(I).LE.10.D0*UROUND) THEN
          WRITE (6,*) ' TOLERANCES(',I,') ARE TOO SMALL'
          ARRET=.TRUE.
        ELSE
          QUOT=ATOL(I)/RTOL(I)
          RTOL(I)=0.1D0*RTOL(I)**EXPM
          ATOL(I)=RTOL(I)*QUOT
        END IF
        END DO
      END IF
C ---- NMAX , THE MAXIMAL NUMBER OF STEPS -----
      IF (IWORK(2).EQ.0) THEN
        NMAX=100000
      ELSE
        NMAX=IWORK(2)
        IF (NMAX.LE.0) THEN
          WRITE(6,*)' WRONG INPUT IWORK(2)=',IWORK(2)
          ARRET=.TRUE.
        END IF
      END IF
C ---- NIT    MAXIMAL NUMBER OF NEWTON ITERATIONS
      IF (IWORK(3).EQ.0) THEN
```

```
      NIT=7
    ELSE
      NIT=IWORK(3)
      IF (NIT.LE.0) THEN
        WRITE(6,*)' CURIOUS INPUT IWORK(3)=',IWORK(3)
        ARRET=.TRUE.
      END IF
    END IF
C ——— STARTN  SWITCH FOR STARTING VALUES OF NEWTON ITERATIONS
    IF(IWORK(4).EQ.0)THEN
      STARTN=.FALSE.
    ELSE
      STARTN=.TRUE.
    END IF
C ------ PARAMETER FOR DIFFERENTIAL-ALGEBRAIC COMPONENTS
    NIND1=IWORK(5)
    NIND2=IWORK(6)
    NIND3=IWORK(7)
    IF (NIND1.EQ.0) NIND1=N
    IF (NIND1+NIND2+NIND3.NE.N) THEN
     WRITE(6,*)' CURIOUS INPUT FOR IWORK(5,6,7)=',NIND1,NIND2,NIND3
     ARRET=.TRUE.
    END IF
C ------ PRED   STEP SIZE CONTROL
    IF(IWORK(8).LE.1)THEN
      PRED=.TRUE.
    ELSE
      PRED=.FALSE.
    END IF
C ——— PARAMETER FOR SECOND ORDER EQUATIONS
    M1=IWORK(9)
    M2=IWORK(10)
    NM1=N-M1
    IF (M1.EQ.0) M2=N
    IF (M2.EQ.0) M2=M1
    IF (M1.LT.0.OR.M2.LT.0.OR.M1+M2.GT.N) THEN
     WRITE(6,*)' CURIOUS INPUT FOR IWORK(9,10)=',M1,M2
     ARRET=.TRUE.
    END IF
C ——— SAFE    SAFETY FACTOR IN STEP SIZE PREDICTION
    IF (WORK(2).EQ.0.0D0) THEN
      SAFE=0.9D0
    ELSE
      SAFE=WORK(2)
      IF (SAFE.LE.0.001D0.OR.SAFE.GE.1.0D0) THEN
        WRITE(6,*)' CURIOUS INPUT FOR WORK(2)=',WORK(2)
        ARRET=.TRUE.
      END IF
    END IF
C —— THET    DECIDES WHETHER THE JACOBIAN SHOULD BE RECOMPUTED;
    IF (WORK(3).EQ.0.D0) THEN
      THET=0.001D0
    ELSE
      THET=WORK(3)
      IF (THET.GE.1.0D0) THEN
        WRITE(6,*)' CURIOUS INPUT FOR WORK(3)=',WORK(3)
        ARRET=.TRUE.
      END IF
    END IF
C — FNEWT   STOPPING CRITERION FOR NEWTON'S METHOD, USUALLY CHOSEN <1.
    TOLST=RTOL(1)
    IF (WORK(4).EQ.0.D0) THEN
      FNEWT=MAX(10*UROUND/TOLST,MIN(0.03D0,TOLST**0.5D0))
    ELSE
      FNEWT=WORK(4)
      IF (FNEWT.LE.UROUND/TOLST) THEN
        WRITE(6,*)' CURIOUS INPUT FOR WORK(4)=',WORK(4)
        ARRET=.TRUE.
      END IF
    END IF
```

```
C — QUOT1 AND QUOT2: IF QUOT1 < HNEW/HOLD < QUOT2, STEP SIZE = CONST.
      IF (WORK(5).EQ.0.D0) THEN
        QUOT1=1.D0
      ELSE
        QUOT1=WORK(5)
      END IF
      IF (WORK(6).EQ.0.D0) THEN
        QUOT2=1.2D0
      ELSE
        QUOT2=WORK(6)
      END IF
      IF (QUOT1.GT.1.0D0.OR.QUOT2.LT.1.0D0) THEN
        WRITE(6,*)' CURIOUS INPUT FOR WORK(5,6)=',QUOT1,QUOT2
        ARRET=.TRUE.
      END IF
C ——— MAXIMAL STEP SIZE
      IF (WORK(7).EQ.0.D0) THEN
        HMAX=XEND-X
      ELSE
        HMAX=WORK(7)
      END IF
C —— FACL,FACR  PARAMETERS FOR STEP SIZE SELECTION
      IF(WORK(8).EQ.0.D0)THEN
        FACL=5.D0
      ELSE
        FACL=1.D0/WORK(8)
      END IF
      IF(WORK(9).EQ.0.D0)THEN
        FACR=1.D0/8.0D0
      ELSE
        FACR=1.D0/WORK(9)
      END IF
      IF (FACL.LT.1.0D0.OR.FACR.GT.1.0D0) THEN
          WRITE(6,*)' CURIOUS INPUT WORK(8,9)=',WORK(8),WORK(9)
          ARRET=.TRUE.
        END IF
C *** *** *** *** *** *** *** *** *** *** *** *** *** ***
C       COMPUTATION OF ARRAY ENTRIES
C *** *** *** *** *** *** *** *** *** *** *** *** *** ***
C --- IMPLICIT, BANDED OR NOT ?
      IMPLCT=IMAS.NE.0
      JBAND=MLJAC.LT.NM1
C ——— COMPUTATION OF THE ROW-DIMENSIONS OF THE 2-ARRAYS ---
C – JACOBIAN  AND  MATRICES E1, E2
      IF (JBAND) THEN
        LDJAC=MLJAC+MUJAC+1
        LDE1=MLJAC+LDJAC
      ELSE
        MLJAC=NM1
        MUJAC=NM1
        LDJAC=NM1
        LDE1=NM1
      END IF
C -- MASS MATRIX
      IF (IMPLCT) THEN
        IF (MLMAS.NE.NM1) THEN
          LDMAS=MLMAS+MUMAS+1
          IF (JBAND) THEN
            IJOB=4
          ELSE
            IJOB=3
          END IF
        ELSE
          MUMAS=NM1
          LDMAS=NM1
          IJOB=5
        END IF
C —— BANDWITH OF "MAS" NOT SMALLER THAN BANDWITH OF "JAC"
        IF (MLMAS.GT.MLJAC.OR.MUMAS.GT.MUJAC) THEN
          WRITE (6,*) 'BANDWITH OF "MAS" NOT SMALLER THAN BANDWITH OF
```

```
     & "JAC"'
          ARRET=.TRUE.
        END IF
      ELSE
        LDMAS=0
        IF (JBAND) THEN
          IJOB=2
        ELSE
          IJOB=1
          IF (N.GT.2.AND.IWORK(1).NE.0) IJOB=7
        END IF
      END IF
      LDMAS2=MAX(1,LDMAS)
C ---- HESSENBERG OPTION ONLY FOR EXPLICIT EQU. WITH FULL JACOBIAN
      IF ((IMPLCT.OR.JBAND).AND.IJOB.EQ.7) THEN
        WRITE(6,*)' HESSENBERG OPTION ONLY FOR EXPLICIT EQUATIONS WITH
     &FULL JACOBIAN'
        ARRET=.TRUE.
      END IF
C ----- PREPARE THE ENTRY-POINTS FOR THE ARRAYS IN WORK -----
      IEZ1=21
      IEZ2=IEZ1+N
      IEZ3=IEZ2+N
      IEY0=IEZ3+N
      IESCAL=IEY0+N
      IEF1=IESCAL+N
      IEF2=IEF1+N
      IEF3=IEF2+N
      IECON=IEF3+N
      IEJAC=IECON+4*N
      IEMAS=IEJAC+N*LDJAC
      IEE1=IEMAS+NM1*LDMAS
      IEE2R=IEE1+NM1*LDE1
      IEE2I=IEE2R+NM1*LDE1
C ---- TOTAL STORAGE REQUIREMENT ---------
      ISTORE=IEE2I+NM1*LDE1-1
      IF(ISTORE.GT.LWORK)THEN
        WRITE(6,*)' INSUFFICIENT STORAGE FOR WORK, MIN. LWORK=',ISTORE
        ARRET=.TRUE.
      END IF
C ------ ENTRY POINTS FOR INTEGER WORKSPACE -----
      IEIP1=21
      IEIP2=IEIP1+NM1
      IEIPH=IEIP2+NM1
C ------- TOTAL REQUIREMENT ----------
      ISTORE=IEIPH+NM1-1
      IF (ISTORE.GT.LIWORK) THEN
        WRITE(6,*)' INSUFF. STORAGE FOR IWORK, MIN. LIWORK=',ISTORE
        ARRET=.TRUE.
      END IF
C ---- WHEN A FAIL HAS OCCURED, WE RETURN WITH IDID=-1
      IF (ARRET) THEN
        IDID=-1
        RETURN
      END IF
C ------ CALL TO CORE INTEGRATOR ---------
      CALL RADCOR(N,FCN,X,Y,XEND,HMAX,H,RTOL,ATOL,ITOL,
     &  JAC,IJAC,MLJAC,MUJAC,MAS,MLMAS,MUMAS,SOLOUT,IOUT,IDID,
     &  NMAX,UROUND,SAFE,THET,FNEWT,QUOT1,QUOT2,NIT,IJOB,STARTN,
     &  NIND1,NIND2,NIND3,PRED,FACL,FACR,M1,M2,NM1,
     &  IMPLCT,JBAND,LDJAC,LDE1,LDMAS2,WORK(IEZ1),WORK(IEZ2),
     &  WORK(IEZ3),WORK(IEY0),WORK(IESCAL),WORK(IEF1),WORK(IEF2),
     &  WORK(IEF3),WORK(IEJAC),WORK(IEE1),WORK(IEE2R),WORK(IEE2I),
     &  WORK(IEMAS),IWORK(IEIP1),IWORK(IEIP2),IWORK(IEIPH),
     &  WORK(IECON),NFCN,NJAC,NSTEP,NACCPT,NREJCT,NDEC,NSOL,RPAR,IPAR)
      IWORK(14)=NFCN
      IWORK(15)=NJAC
      IWORK(16)=NSTEP
      IWORK(17)=NACCPT
      IWORK(18)=NREJCT
```

```
      IWORK(19)=NDEC
      IWORK(20)=NSOL
C ---- RESTORE TOLERANCES
      EXPM=1.0D0/EXPM
      IF (ITOL.EQ.0) THEN
          QUOT=ATOL(1)/RTOL(1)
          RTOL(1)=(10.0D0*RTOL(1))**EXPM
          ATOL(1)=RTOL(1)*QUOT
      ELSE
        DO I=1,N
          QUOT=ATOL(I)/RTOL(I)
          RTOL(I)=(10.0D0*RTOL(I))**EXPM
          ATOL(I)=RTOL(I)*QUOT
        END DO
      END IF
C ------- RETURN --------
      RETURN
      END
C
C     END OF SUBROUTINE RADAU5
C
C *********************************************************
C
      SUBROUTINE RADCOR(N,FCN,X,Y,XEND,HMAX,H,RTOL,ATOL,ITOL,
     &  JAC,IJAC,MLJAC,MUJAC,MAS,MLMAS,MUMAS,SOLOUT,IOUT,IDID,
     &  NMAX,UROUND,SAFE,THET,FNEWT,QUOT1,QUOT2,NIT,IJOB,STARTN,
     &  NIND1,NIND2,NIND3,PRED,FACL,FACR,M1,M2,NM1,
     &  IMPLCT,BANDED,LDJAC,LDE1,LDMAS,Z1,Z2,Z3,
     &  Y0,SCAL,F1,F2,F3,FJAC,E1,E2R,E2I,FMAS,IP1,IP2,IPHES,
     &  CONT,NFCN,NJAC,NSTEP,NACCPT,NREJCT,NDEC,NSOL,RPAR,IPAR)
C ----------------------------------------------------------
C     CORE INTEGRATOR FOR RADAU5
C     PARAMETERS SAME AS IN RADAU5 WITH WORKSPACE ADDED
C ----------------------------------------------------------
C         DECLARATIONS
C ----------------------------------------------------------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION Y(N),Z1(N),Z2(N),Z3(N),Y0(N),SCAL(N),F1(N),F2(N),F3(N)
      DIMENSION FJAC(LDJAC,N),FMAS(LDMAS,NM1),CONT(4*N)
      DIMENSION E1(LDE1,NM1),E2R(LDE1,NM1),E2I(LDE1,NM1)
      DIMENSION ATOL(*),RTOL(*),RPAR(*),IPAR(*)
      INTEGER IP1(NM1),IP2(NM1),IPHES(NM1)
      COMMON /CONRA5/NN,NN2,NN3,NN4,XSOL,HSOL,C2M1,C1M1
      COMMON/LINAL/MLE,MUE,MBJAC,MBB,MDIAG,MDIFF,MBDIAG
      LOGICAL REJECT,FIRST,IMPLCT,BANDED,CALJAC,STARTN,CALHES
      LOGICAL INDEX1,INDEX2,INDEX3,LAST,PRED
      EXTERNAL FCN
C *** *** *** *** *** *** ***
C  INITIALISATIONS
C *** *** *** *** *** *** ***
C -------- DUPLIFY N FOR COMMON BLOCK CONT -----
      NN=N
      NN2=2*N
      NN3=3*N
      LRC=4*N
C ------ CHECK THE INDEX OF THE PROBLEM -----
      INDEX1=NIND1.NE.0
      INDEX2=NIND2.NE.0
      INDEX3=NIND3.NE.0
C ------ COMPUTE MASS MATRIX FOR IMPLICIT CASE --------
      IF (IMPLCT) CALL MAS(NM1,FMAS,LDMAS,RPAR,IPAR)
C ---------- CONSTANTS ---------
      SQ6=DSQRT(6.D0)
      C1=(4.D0-SQ6)/10.D0
      C2=(4.D0+SQ6)/10.D0
      C1M1=C1-1.D0
      C2M1=C2-1.D0
      C1MC2=C1-C2
      DD1=-(13.D0+7.D0*SQ6)/3.D0
      DD2=(-13.D0+7.D0*SQ6)/3.D0
```

```
      DD3=-1.D0/3.D0
      U1=(6.D0+81.D0**(1.D0/3.D0)-9.D0**(1.D0/3.D0))/30.D0
      ALPH=(12.D0-81.D0**(1.D0/3.D0)+9.D0**(1.D0/3.D0))/60.D0
      BETA=(81.D0**(1.D0/3.D0)+9.D0**(1.D0/3.D0))*DSQRT(3.D0)/60.D0
      CNO=ALPH**2+BETA**2
      U1=1.0D0/U1
      ALPH=ALPH/CNO
      BETA=BETA/CNO
      T11=9.123239487089294279 2D-02
      T12=-0.141255295020954208 43D0
      T13=-3.002919410514742449 2D-02
      T21=0.241717932707107018 96D0
      T22=0.204129352293799931 99D0
      T23=0.382942112757261937 79D0
      T31=0.966048182615092936 19D0
      TI11=4.325579890063155351 0D0
      TI12=0.339199251815809869 54D0
      TI13=0.541770539935874871 19D0
      TI21=-4.178718591551904727 3D0
      TI22=-0.327682820761062387 08D0
      TI23=0.476623554500550451 96D0
      TI31=-0.502872634945786875 95D0
      TI32=2.571926949855605429 2D0
      TI33=-0.596039204828224924 97D0
      IF (M1.GT.0) IJOB=IJOB+10
      POSNEG=SIGN(1.D0,XEND-X)
      HMAXN=MIN(ABS(HMAX),ABS(XEND-X))
      IF (ABS(H).LE.10.D0*UROUND) H=1.0D-6
      H=MIN(ABS(H),HMAXN)
      H=SIGN(H,POSNEG)
      HOLD=H
      REJECT=.FALSE.
      FIRST=.TRUE.
      LAST=.FALSE.
      IF ((X+H*1.0001D0-XEND)*POSNEG.GE.0.D0) THEN
        H=XEND-X
        LAST=.TRUE.
      END IF
      FACCON=1.D0
      CFAC=SAFE*(1+2*NIT)
      NSING=0
      XOLD=X
      IF (IOUT.NE.0) THEN
        IRTRN=1
        NRSOL=1
        XOSOL=XOLD
        XSOL=X
        DO I=1,N
          CONT(I)=Y(I)
        END DO
        NSOLU=N
        HSOL=HOLD
        CALL SOLOUT(NRSOL,XOSOL,XSOL,Y,CONT,LRC,NSOLU,
     &              RPAR,IPAR,IRTRN)
        IF (IRTRN.LT.0) GOTO 179
      END IF
      MLE=MLJAC
      MUE=MUJAC
      MBJAC=MLJAC+MUJAC+1
      MBB=MLMAS+MUMAS+1
      MDIAG=MLE+MUE+1
      MDIFF=MLE+MUE-MUMAS
      MBDIAG=MUMAS+1
      N2=2*N
      N3=3*N
      IF (ITOL.EQ.0) THEN
        DO I=1,N
          SCAL(I)=ATOL(1)+RTOL(1)*ABS(Y(I))
        END DO
      ELSE
```

```
      DO I=1,N
        SCAL(I)=ATOL(I)+RTOL(I)*ABS(Y(I))
      END DO
    END IF
    HHFAC=H
    CALL FCN(N,X,Y,Y0,RPAR,IPAR)
    NFCN=NFCN+1
C --- BASIC INTEGRATION STEP
 10 CONTINUE
C *** *** *** *** *** *** ***
C  COMPUTATION OF THE JACOBIAN
C *** *** *** *** *** *** ***
    NJAC=NJAC+1
    IF (IJAC.EQ.0) THEN
C --- COMPUTE JACOBIAN MATRIX NUMERICALLY
      IF (BANDED) THEN
C --- JACOBIAN IS BANDED
        MUJACP=MUJAC+1
        MD=MIN(MBJAC,M2)
        DO MM=1,M1/M2+1
          DO K=1,MD
            J=K+(MM-1)*M2
12            F1(J)=Y(J)
            F2(J)=DSQRT(UROUND*MAX(1.D-5,ABS(Y(J))))
            Y(J)=Y(J)+F2(J)
            J=J+MD
            IF (J.LE.MM*M2) GOTO 12
            CALL FCN(N,X,Y,CONT,RPAR,IPAR)
            J=K+(MM-1)*M2
            J1=K
            LBEG=MAX(1,J1-MUJAC)+M1
14            LEND=MIN(M2,J1+MLJAC)+M1
            Y(J)=F1(J)
            MUJACJ=MUJACP-J1-M1
            DO L=LBEG,LEND
              FJAC(L+MUJACJ,J)=(CONT(L)-Y0(L))/F2(J)
            END DO
            J=J+MD
            J1=J1+MD
            LBEG=LEND+1
            IF (J.LE.MM*M2) GOTO 14
          END DO
        END DO
      ELSE
C --- JACOBIAN IS FULL
        DO I=1,N
          YSAFE=Y(I)
          DELT=DSQRT(UROUND*MAX(1.D-5,ABS(YSAFE)))
          Y(I)=YSAFE+DELT
          CALL FCN(N,X,Y,CONT,RPAR,IPAR)
          DO J=M1+1,N
            FJAC(J-M1,I)=(CONT(J)-Y0(J))/DELT
          END DO
          Y(I)=YSAFE
        END DO
      END IF
    ELSE
C --- COMPUTE JACOBIAN MATRIX ANALYTICALLY
      CALL JAC(N,X,Y,FJAC,LDJAC,RPAR,IPAR)
    END IF
    CALJAC=.TRUE.
    CALHES=.TRUE.
 20 CONTINUE
C --- COMPUTE THE MATRICES E1 AND E2 AND THEIR DECOMPOSITIONS
    FAC1=U1/H
    ALPHN=ALPH/H
    BETAN=BETA/H
    CALL DECOMR(N,FJAC,LDJAC,FMAS,LDMAS,MLMAS,MUMAS,
   &            M1,M2,NM1,FAC1,E1,LDE1,IP1,IER,IJOB,CALHES,IPHES)
    IF (IER.NE.0) GOTO 78
```

```
      CALL DECOMC(N,FJAC,LDJAC,FMAS,LDMAS,MLMAS,MUMAS,
     &            M1,M2,NM1,ALPHN,BETAN,E2R,E2I,LDE1,IP2,IER,IJOB)
      IF (IER.NE.0) GOTO 78
      NDEC=NDEC+1
   30 CONTINUE
      NSTEP=NSTEP+1
      IF (NSTEP.GT.NMAX) GOTO 178
      IF (0.1D0*ABS(H).LE.ABS(X)*UROUND) GOTO 177
        IF (INDEX2) THEN
          DO I=NIND1+1,NIND1+NIND2
            SCAL(I)=SCAL(I)/HHFAC
          END DO
        END IF
        IF (INDEX3) THEN
          DO I=NIND1+NIND2+1,NIND1+NIND2+NIND3
            SCAL(I)=SCAL(I)/(HHFAC*HHFAC)
          END DO
        END IF
      XPH=X+H
C *** *** *** *** *** *** ***
C  STARTING VALUES FOR NEWTON ITERATION
C *** *** *** *** *** *** ***
      IF (FIRST.OR.STARTN) THEN
        DO I=1,N
          Z1(I)=0.D0
          Z2(I)=0.D0
          Z3(I)=0.D0
          F1(I)=0.D0
          F2(I)=0.D0
          F3(I)=0.D0
        END DO
      ELSE
        C3Q=H/HOLD
        C1Q=C1*C3Q
        C2Q=C2*C3Q
        DO I=1,N
          AK1=CONT(I+N)
          AK2=CONT(I+N2)
          AK3=CONT(I+N3)
          Z1I=C1Q*(AK1+(C1Q-C2M1)*(AK2+(C1Q-C1M1)*AK3))
          Z2I=C2Q*(AK1+(C2Q-C2M1)*(AK2+(C2Q-C1M1)*AK3))
          Z3I=C3Q*(AK1+(C3Q-C2M1)*(AK2+(C3Q-C1M1)*AK3))
          Z1(I)=Z1I
          Z2(I)=Z2I
          Z3(I)=Z3I
          F1(I)=TI11*Z1I+TI12*Z2I+TI13*Z3I
          F2(I)=TI21*Z1I+TI22*Z2I+TI23*Z3I
          F3(I)=TI31*Z1I+TI32*Z2I+TI33*Z3I
        END DO
      END IF
C *** *** *** *** *** *** ***
C  LOOP FOR THE SIMPLIFIED NEWTON ITERATION
C *** *** *** *** *** *** ***
        NEWT=0
        FACCON=MAX(FACCON,UROUND)**0.8D0
        THETA=ABS(THET)
   40   CONTINUE
        IF (NEWT.GE.NIT) GOTO 78
C ---   COMPUTE THE RIGHT-HAND SIDE
        DO I=1,N
          CONT(I)=Y(I)+Z1(I)
        END DO
        CALL FCN(N,X+C1*H,CONT,Z1,RPAR,IPAR)
        DO I=1,N
          CONT(I)=Y(I)+Z2(I)
        END DO
        CALL FCN(N,X+C2*H,CONT,Z2,RPAR,IPAR)
        DO I=1,N
          CONT(I)=Y(I)+Z3(I)
        END DO
```

```
      CALL FCN(N,XPH,CONT,Z3,RPAR,IPAR)
      NFCN=NFCN+3
C --- SOLVE THE LINEAR SYSTEMS
      DO I=1,N
        A1=Z1(I)
        A2=Z2(I)
        A3=Z3(I)
        Z1(I)=TI11*A1+TI12*A2+TI13*A3
        Z2(I)=TI21*A1+TI22*A2+TI23*A3
        Z3(I)=TI31*A1+TI32*A2+TI33*A3
      END DO
    CALL SLVRAD(N,FJAC,LDJAC,MLJAC,MUJAC,FMAS,LDMAS,MLMAS,MUMAS,
   &     M1,M2,NM1,FAC1,ALPHN,BETAN,E1,E2R,E2I,LDE1,Z1,Z2,Z3,
   &     F1,F2,F3,CONT,IP1,IP2,IPHES,IER,IJOB)
      NSOL=NSOL+1
      NEWT=NEWT+1
      DYNO=0.D0
      DO I=1,N
        DENOM=SCAL(I)
        DYNO=DYNO+(Z1(I)/DENOM)**2+(Z2(I)/DENOM)**2
   &      +(Z3(I)/DENOM)**2
      END DO
      DYNO=DSQRT(DYNO/N3)
C --- BAD CONVERGENCE OR NUMBER OF ITERATIONS TO LARGE
      IF (NEWT.GT.1.AND.NEWT.LT.NIT) THEN
        THQ=DYNO/DYNOLD
        IF (NEWT.EQ.2) THEN
          THETA=THQ
        ELSE
          THETA=SQRT(THQ*THQOLD)
        END IF
        THQOLD=THQ
        IF (THETA.LT.0.99D0) THEN
          FACCON=THETA/(1.0D0-THETA)
          DYTH=FACCON*DYNO*THETA**(NIT-1-NEWT)/FNEWT
          IF (DYTH.GE.1.0D0) THEN
            QNEWT=DMAX1(1.0D-4,DMIN1(20.0D0,DYTH))
            HHFAC=.8D0*QNEWT**(-1.0D0/(4.0D0+NIT-1-NEWT))
            H=HHFAC*H
            REJECT=.TRUE.
            LAST=.FALSE.
            IF (CALJAC) GOTO 20
            GOTO 10
          END IF
        ELSE
          GOTO 78
        END IF
      END IF
      DYNOLD=MAX(DYNO,UROUND)
      DO I=1,N
        F1I=F1(I)+Z1(I)
        F2I=F2(I)+Z2(I)
        F3I=F3(I)+Z3(I)
        F1(I)=F1I
        F2(I)=F2I
        F3(I)=F3I
        Z1(I)=T11*F1I+T12*F2I+T13*F3I
        Z2(I)=T21*F1I+T22*F2I+T23*F3I
        Z3(I)=T31*F1I+   F2I
      END DO
      IF (FACCON*DYNO.GT.FNEWT) GOTO 40
C --- ERROR ESTIMATION
    CALL ESTRAD (N,FJAC,LDJAC,MLJAC,MUJAC,FMAS,LDMAS,MLMAS,MUMAS,
   &     H,DD1,DD2,DD3,FCN,NFCN,Y0,Y,IJOB,X,M1,M2,NM1,
   &     E1,LDE1,Z1,Z2,Z3,CONT,F1,F2,IP1,IPHES,SCAL,ERR,
   &     FIRST,REJECT,FAC1,RPAR,IPAR)
C --- COMPUTATION OF HNEW
C --- WE REQUIRE .2<=HNEW/H<=8.
    FAC=MIN(SAFE,CFAC/(NEWT+2*NIT))
    QUOT=MAX(FACR,MIN(FACL,ERR**.25D0/FAC))
```

```
      HNEW=H/QUOT
C *** *** *** *** *** *** ***
C  IS THE ERROR SMALL ENOUGH ?
C *** *** *** *** *** *** ***
      IF (ERR.LT.1.D0) THEN
C --- STEP IS ACCEPTED
        FIRST=.FALSE.
        NACCPT=NACCPT+1
        IF (PRED) THEN
C       --- PREDICTIVE CONTROLLER OF GUSTAFSSON
          IF (NACCPT.GT.1) THEN
            FACGUS=(HACC/H)*(ERR**2/ERRACC)**0.25D0/SAFE
            FACGUS=MAX(FACR,MIN(FACL,FACGUS))
            QUOT=MAX(QUOT,FACGUS)
            HNEW=H/QUOT
          END IF
          HACC=H
          ERRACC=MAX(1.0D-2,ERR)
        END IF
        XOLD=X
        HOLD=H
        X=XPH
        DO I=1,N
          Y(I)=Y(I)+Z3(I)
          Z2I=Z2(I)
          Z1I=Z1(I)
          CONT(I+N)=(Z2I-Z3(I))/C2M1
          AK=(Z1I-Z2I)/C1MC2
          ACONT3=Z1I/C1
          ACONT3=(AK-ACONT3)/C2
          CONT(I+N2)=(AK-CONT(I+N))/C1M1
          CONT(I+N3)=CONT(I+N2)-ACONT3
        END DO
        IF (ITOL.EQ.0) THEN
          DO I=1,N
            SCAL(I)=ATOL(1)+RTOL(1)*ABS(Y(I))
          END DO
        ELSE
          DO I=1,N
            SCAL(I)=ATOL(I)+RTOL(I)*ABS(Y(I))
          END DO
        END IF
        IF (IOUT.NE.0) THEN
          NRSOL=NACCPT+1
          XSOL=X
          XOSOL=XOLD
          DO I=1,N
            CONT(I)=Y(I)
          END DO
          NSOLU=N
          HSOL=HOLD
          CALL SOLOUT(NRSOL,XOSOL,XSOL,Y,CONT,LRC,NSOLU,
     &                RPAR,IPAR,IRTRN)
          IF (IRTRN.LT.0) GOTO 179
        END IF
        CALJAC=.FALSE.
        IF (LAST) THEN
          H=HOPT
          IDID=1
          RETURN
        END IF
        CALL FCN(N,X,Y,Y0,RPAR,IPAR)
        NFCN=NFCN+1
        HNEW=POSNEG*MIN(ABS(HNEW),HMAXN)
        HOPT=HNEW
        HOPT=MIN(H,HNEW)
        IF (REJECT) HNEW=POSNEG*MIN(ABS(HNEW),ABS(H))
        REJECT=.FALSE.
        IF ((X+HNEW/QUOT1-XEND)*POSNEG.GE.0.D0) THEN
          H=XEND-X
```

```
          LAST=.TRUE.
        ELSE
          QT=HNEW/H
          HHFAC=H
          IF (THETA.LE.THET.AND.QT.GE.QUOT1.AND.QT.LE.QUOT2) GOTO 30
          H=HNEW
        END IF
        HHFAC=H
        IF (THETA.LE.THET) GOTO 20
        GOTO 10
      ELSE
C --- STEP IS REJECTED
        REJECT=.TRUE.
        LAST=.FALSE.
        IF (FIRST) THEN
          H=H*0.1D0
          HHFAC=0.1D0
        ELSE
          HHFAC=HNEW/H
          H=HNEW
        END IF
        IF (NACCPT.GE.1) NREJCT=NREJCT+1
        IF (CALJAC) GOTO 20
        GOTO 10
      END IF
C --- UNEXPECTED STEP-REJECTION
 78   CONTINUE
      IF (IER.NE.0) THEN
        NSING=NSING+1
        IF (NSING.GE.5) GOTO 176
      END IF
      H=H*0.5D0
      HHFAC=0.5D0
      REJECT=.TRUE.
      LAST=.FALSE.
      IF (CALJAC) GOTO 20
      GOTO 10
C --- FAIL EXIT
 176  CONTINUE
      WRITE(6,979)X
      WRITE(6,*) ' MATRIX IS REPEATEDLY SINGULAR, IER=',IER
      IDID=-4
      RETURN
 177  CONTINUE
      WRITE(6,979)X
      WRITE(6,*) ' STEP SIZE T0O SMALL, H=',H
      IDID=-3
      RETURN
 178  CONTINUE
      WRITE(6,979)X
      WRITE(6,*) ' MORE THAN NMAX =',NMAX,'STEPS ARE NEEDED'
      IDID=-2
      RETURN
C --- EXIT CAUSED BY SOLOUT
 179  CONTINUE
      WRITE(6,979)X
 979  FORMAT(' EXIT OF RADAU5 AT X=',E18.4)
      IDID=2
      RETURN
      END
C
C     END OF SUBROUTINE RADCOR
C
C ***********************************************************
C
      DOUBLE PRECISION FUNCTION CONTR5(I,X,CONT,LRC)
C ----------------------------------------------------------
C     THIS FUNCTION CAN BE USED FOR CONINUOUS OUTPUT. IT PROVIDES AN
C     APPROXIMATION TO THE I-TH COMPONENT OF THE SOLUTION AT X.
C     IT GIVES THE VALUE OF THE COLLOCATION POLYNOMIAL, DEFINED FOR
```

```
C   THE LAST SUCCESSFULLY COMPUTED STEP (BY RADAU5).
C ---------- ---------- ----------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION CONT(LRC)
      COMMON /CONRA5/NN,NN2,NN3,NN4,XSOL,HSOL,C2M1,C1M1
      S=(X-XSOL)/HSOL
      CONTR5=CONT(I)+S*(CONT(I+NN)+(S-C2M1)*(CONT(I+NN2)
     &     +(S-C1M1)*CONT(I+NN3)))
      RETURN
      END
C
C   END OF FUNCTION CONTR5
C
C ***********************************************************
```

Appendix C

C++ Code used to solve Stiff Ordinary Differential Equations Presented in Chapter 5

1. Semi-Implicit Euler Method

```
h=Time_Inc;
double det_A;
x=h;
 delta_n[1]=0.0;
 delta_n[2]=0.0;
 delta_0[1]=0.0;
 delta_0[2]=0.0;
  A[1][1]=1.0-h*df1dy1(x,yn,param);
  A[1][2]=1.0-h*df1dy2(x,yn,param);
  A[2][1]=1.0-h*df2dy1(x,yn,param);
  A[2][2]=1.0-h*df2dy2(x,yn,param);
  det_A=A[1][1]*A[2][2]-A[1][2]*A[2][1];
  A_inv[1][1]=A[2][2]/det_A;
  A_inv[1][2]=-A[1][2]/det_A;
  A_inv[2][1]=-A[2][1]/det_A;
  A_inv[2][2]=A[1][1]/det_A;
if(t!=0)
{
 //
 for(i=1;i<=2;i++)
 {
  delta_n[i]=delta_n[i]+A_inv[i][1]*h*f1(x,yn,param);
  delta_n[i]=delta_n[i]+A_inv[i][2]*h*f2(x,yn,param);
 }
  for(i=1;i<=2;i++)
  {
   yn[i]=yn[i]+delta_n[i];
  }
}
else if(t==Finish_Time)
{
 for(i=1;i<=2;i++)
 {
  temp_n[i]=0.0;
  temp_n[i]=temp_n[i]+A_inv[i][1]*(h*f1(x,yn,param)-delta_0[1]);
  temp_n[i]=temp_n[i]+A_inv[i][2]*(h*f2(x,yn,param)-delta_0[2]);
  delta_n[i]=temp_n[i];
 }
  for(i=1;i<=2;i++)
  {
```

```
    yn[i]=yn[i]+delta_n[i];
   }
  }
  else
  {
  //
   for(i=1;i<=2;i++)
   {
    temp_n[i]=0.0;
    temp_n[i]=temp_n[i]+A_inv[i][1]*(h*f1(x,yn,param)-delta_0[1]);
    temp_n[i]=temp_n[i]+A_inv[i][2]*(h*f2(x,yn,param)-delta_0[2]);
    delta_n[i]=delta_n[i]+2.0*temp_n[i];
   }
    for(i=1;i<=2;i++)
    {
     yn[i]=yn[i]+delta_n[i];
    }
  }
    x=x+h;
     phasem1[node1]=yn[1];
     phasem1[node2]=yn[1];
     phasem2[node1]=yn[2];
     phasem2[node2]=yn[2];
```

2. C++ Program

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop
# include <time.h>
#include "sma1.h"
#include <string.h>
#include <math.h>
#include "matrix.h"
#include "rk_fun3.h"
#include "Time_stepping.h"
#define SWAP(a,b) {dum=(a);(a)=(b);(b)=dum;}
#define TINY 1.0e-20
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
    : TForm(Owner)
{

}
//---------------------------------------------------------------------------
void __fastcall TForm1::OpenData1Click(TObject *Sender)
{
// use dialog to open data file for read
if(OpenDialog1->Execute())
 {
 // initialize maximum values for plotting
  max_x=-1E15;
  max_y=-1E15;
  min_x=1E15;
  min_y=1E15;
```

```
// get file name
file_name = OpenDialog1->FileName;
// open input file stream
 ifstream fef(file_name.c_str());
// read in number of nodes and elements
 fef>>nodes>>elements;
 out_file<<"number of nodes: "<<nodes<<"  number of elements: "<<elements<<"\n";
 int i,j;
 int el;
  // define arrays and matrix to store geometry
    elem=imatrix(1,elements,1,2);
    nx=vector(1,nodes);
    ny=vector(1,nodes);
    nx1=vector(1,nodes);
    ny1=vector(1,nodes);
   // displacment vector
   u=vector(1,nodes*2);
   R=vector(1,nodes*2);
    variable=vector(1,nodes);
   for(i=1;i<=nodes*2;i++) u[i]=0.0;
   for(i=1;i<=nodes*2;i++) R[i]=0.0;
   K=matrix(1,nodes*2,1,nodes*2);
   loadx=vector(1,nodes);
   loady=vector(1,nodes);
   bound=ivector(1,nodes*2);
   F=vector(1,nodes*2);
   P=vector(1,nodes*2);
   for(i=1;i<=nodes*2;i++)
   {
       for(j=1;j<=nodes*2;j++)
        {
          K[i][j]=0.0;
        }
   }
   for(i=1;i<=nodes;i++) loadx[i]=0.0;
   for(i=1;i<=nodes;i++) loady[i]=0.0;
   for(i=1;i<=nodes*2;i++) bound[i]=0;
    for(i=1;i<=nodes*2;i++) F[i]=0.0;
    for(i=1;i<=nodes*2;i++) P[i]=0.0;
 for(i=1;i<=nodes;i++)
 {
  fef>>node>>nx[i]>>ny[i];
  if(nx[i]>max_x) max_x=nx[i];
  if(ny[i]>max_y) max_y=ny[i];
  if(nx[i]<min_x) min_x=nx[i];
  if(ny[i]<min_y) min_y=ny[i];
 }
// read in element connectivity
 for(i=1;i<=elements;i++)
 {
  fef>>el;
  for(j=1;j<=2;j++) fef>>elem[i][j];
 }
// write out the data to file
out_file<<"*****************Nodes*****************\n";
 for(i=1;i<=nodes;i++)
 {
  out_file<<i<<"  "<<nx[i]<<","<<ny[i]<<"\n";
 }
  out_file<<"*************Elements*****************\n";
  for(i=1;i<=elements;i++)
 {
  out_file<<i<<"  "<<elem[i][1]<<","<<elem[i][2]<<"\n";
 }
 fef>>number_of_loads;
 for(i=1;i<=number_of_loads;i++)
 {
  fef>>node;
  fef>>loadx[node]>>loady[node];
 }
```

```
    for(i=1;i<=nodes;i++)
    {
     out_file<<"loadx:  "<<loadx[i]<<" loady: "<<loady[i]<<"\n";
    }
     fef>>number_of_bounds;
    num_bound=0;
    for(i=1;i<=number_of_bounds;i++)
    {
     fef>>node;
     fef>>bound[node*2-1]>>bound[node*2];
     if(bound[node*2-1]==1) num_bound++;
     if(bound[node*2]==1) num_bound++;
    }
     geom=true;
     contour_x=false;
     contour_y=false;
     contour_s=false;
     contour_e=false;
     Form1->Invalidate();
  }
 }
 //---------------------------------------------------------
 void __fastcall TForm1::FormCreate(TObject *Sender)
 {
 // open output file
    out_file.open("sma.out",ios::out);
    out_file.setf(ios::scientific);
     out_file<<"*************************************\n";
     out_file<<"****          SMA FE          ***\n";
     out_file<<"*************************************\n";
    out_file<<" Job Started:  "<<DateTimeToStr(Now()).c_str()<<"\n";
    contour_file.open("contour.out",ios::out);
    contour_file.setf(ios::scientific);
    contour_file<<"*************************************\n";
    contour_file<<"****          SMA FE          ***\n";
    contour_file<<"*************************************\n";
    contour_file<<" Job Started:  "<<DateTimeToStr(Now()).c_str()<<"\n";
    geom=false;
    result=false;
    dmag=100.0;
    variable=vector(1,nodes);
    dtype=0;
 }
 //---------------------------------------------------------

 void __fastcall TForm1::Exit1Click(TObject *Sender)
 {
 // exit
 Form1->Close();
 }
 //---------------------------------------------------------

 void __fastcall TForm1::Analysis1Click(TObject *Sender)
 {
 out_file<<"Start Time:"<<Start_Time<<"\n";
 out_file<<"Time Inc: "<<Time_Inc<<"\n";
 out_file<<"Finish_time: "<<Finish_Time<<"\n";
 // analyse
 int el,node1,node2,i,j;
 double X1,X2,Y1,Y2,L0,cos_phi0,sin_phi0;
 double x1,x2,y1,y2,L,cos_phi,sin_phi;
 double ax,ay;
 KM=matrix(1,4,1,4);
 KG=matrix(1,4,1,4);
 KB=matrix(1,nodes*2-num_bound,1,nodes*2-num_bound);
 KM_keep=matrix(1,nodes*2-num_bound,1,nodes*2-num_bound);
 FB=matrix(1,nodes*2-num_bound,1,1);
 UB=vector(1,nodes*2-num_bound);
 param=vector(1,20);
 A=matrix(1,2,1,2);
```

```
A_inv=matrix(1,2,1,2);
yn=vector(1,2);
delta_n=vector(1,2);
delta_0=vector(1,2);
temp_n=vector(1,2);
residual=vector(1,nodes*2-num_bound);
double k1,k2;
double estran,stress;
double E,A0;
E=34E6;
A0=7.8;
// while solution is not converged
iter=0;
max_iter=10;
TOL=1E-15;
not_converged=true;
stress_v=vector(1,nodes);
strain_v=vector(1,nodes);
phasem1=vector(1,nodes);
phasem2=vector(1,nodes);
// time step
double time;
long t,tn;
tn=ceil((Finish_Time-Start_Time)/Time_Inc);
time=Start_Time;
// intial phases
yn[1]=0.5;
yn[2]=0.5;
for(t=1;t<=tn;t++)
{
time=time+Time_Inc;
while(not_converged)
{
//———————— for each element start ————————
    for(el=1;el<=elements;el++)
    {
        // reference configuration
        node1=elem[el][1];
        node2=elem[el][2];
        X1=nx[node1];
        X2=nx[node2];
        Y1=ny[node1];
        Y2=ny[node2];
        L0=sqrt((X2-X1)*(X2-X1)+(Y2-Y1)*(Y2-Y1));
        // calculate cos theta
        cos_phi0=(X2-X1)/L0;
        // calculate cos theta
        sin_phi0=(Y2-Y1)/L0;
        out_file<<"X1: "<<X1<<" X2: "<<X2<<" Y1: "<<Y1<<" Y2: "
        <<Y2<<" L0: "<<L0<<"\n";
        out_file<<" cos_phi0: "<<cos_phi0<<" sin_phi0: "<<sin_phi0<<"\n";
        // current configuration
        x1=X1+u[node1*2-1];
        x2=X2+u[node2*2-1];
        y1=Y1+u[node1*2];
        y2=Y2+u[node2*2];
        L=sqrt((x2-x1)*(x2-x1)+(y2-y1)*(y2-y1));
        // calculate cos theta
        cos_phi=(x2-x1)/L;
        // calculate cos theta
        sin_phi=(y2-y1)/L;
        out_file<<"x1: "<<x1<<" x2: "<<x2<<" y1: "<<y1<<" y2: "
        <<y2<<" L: "<<L<<"\n";
        out_file<<"cos_phi: "<<cos_phi<<" sin_phi: "<<sin_phi<<"\n";
        // components of strain displacement matrix
        ax=L/L0*cos_phi;
        ay=L/L0*sin_phi;
        out_file<<"ax: "<<ax<<" ay: "<<ay<<"\n";
        // define material stiffness matrix
        k1=E*A0/L0;
```

```
KM[1][1]=k1*ax*ax;
KM[1][2]=k1*ax*ay;
KM[1][3]=-k1*ax*ax;
KM[1][4]=-k1*ax*ay;
KM[2][1]=k1*ax*ay;
KM[2][2]=k1*ay*ay;
KM[2][3]=-k1*ax*ay;
KM[2][4]=-k1*ay*ay;
KM[3][1]=-k1*ax*ax;
KM[3][2]=-k1*ax*ay;
KM[3][3]=k1*ax*ax;
KM[3][4]=k1*ax*ay;
KM[4][1]=-k1*ax*ay;
KM[4][2]=-k1*ay*ay;
KM[4][3]=k1*ax*ay;
KM[4][4]=k1*ay*ay;
 out_file<<" **********KM*********\n";
for(i=1;i<=4;i++)
{
 for(j=1;j<=4;j++)
 {
  out_file<<KM[i][j]<<" ";
 }
 out_file<<"\n";
}
 for(i=1;i<=nodes*2-num_bound;i++)
 {
     residual[i]=0;
 }
out_file<<"ax: "<<ax<<" ay: "<<ay<<"\n";
// define geomertic stiffness matrix
estran=(L*L-L0*L0)/(2.0*L0*L0);
stress=E*estran;
stress=stress+yn[1]*E*param[2]-yn[2]*E*param[2];
stress_v[node1]=stress;
stress_v[node2]=stress;
strain_v[node1]=estran;
strain_v[node2]=estran;
out_file<<"stran: "<<estran<<"stress: "<<stress<<"\n";
N=A0*stress;
k2=N/L0;
KG[1][1]=k2;
KG[1][2]=0;
KG[1][3]=-k2;
KG[1][4]=0;
KG[2][1]=0;
KG[2][2]=k2;
KG[2][3]=0;
KG[2][4]=-k2;
KG[3][1]=-k2;
KG[3][2]=0;
KG[3][3]=k2;
KG[3][4]=0;
KG[4][1]=0;
KG[4][2]=-k2;
KG[4][3]=0;
KG[4][4]=k2;
out_file<<" **********KG*********\n";
for(i=1;i<=4;i++)
{
 for(j=1;j<=4;j++)
 {
  out_file<<KG[i][j]<<" ";
 }
 out_file<<"\n";
}
// assemble stiffness matrix
 K[node1*2-1][node1*2-1]=K[node1*2-1][node1*2-1]+KM[1][1]+KG[1][1];
 K[node1*2-1][node1*2]  =K[node1*2-1][node1*2]  +KM[1][2]+KG[1][2];
 K[node1*2-1][node2*2-1]=K[node1*2-1][node2*2-1]+KM[1][3]+KG[1][3];
```

```
K[node1*2-1][node2*2] =K[node1*2-1][node2*2] +KM[1][4]+KG[1][4];

K[node1*2][node1*2-1]=K[node1*2][node1*2-1]+KM[2][1]+KG[2][1];
K[node1*2][node1*2] =K[node1*2][node1*2] +KM[2][2]+KG[2][2];
K[node1*2][node2*2-1]=K[node1*2][node2*2-1]+KM[2][3]+KG[2][3];
K[node1*2][node2*2] =K[node1*2][node2*2] +KM[2][4]+KG[2][4];

K[node2*2-1][node1*2-1]=K[node2*2-1][node1*2-1]+KM[3][1]+KG[3][1];
K[node2*2-1][node1*2] =K[node2*2-1][node1*2] +KM[3][2]+KG[3][2];
K[node2*2-1][node2*2-1]=K[node2*2-1][node2*2-1]+KM[3][3]+KG[3][3];
K[node2*2-1][node2*2] =K[node2*2-1][node2*2] +KM[3][4]+KG[3][4];

K[node2*2][node1*2-1]=K[node2*2][node1*2-1]+KM[4][1]+KG[4][1];
K[node2*2][node1*2] =K[node2*2][node1*2] +KM[4][2]+KG[4][2];
K[node2*2][node2*2-1]=K[node2*2][node2*2-1]+KM[4][3]+KG[4][3];
K[node2*2][node2*2] =K[node2*2][node2*2] +KM[4][4]+KG[4][4];
// force vector
double time_scale;
time_scale=Finish_Time/time;
F[node1*2-1]=loadx[node1]*time_scale;
F[node1*2]= loady[node1]*time_scale;
F[node2*2-1]=loadx[node2]*time_scale;
F[node2*2]=loady[node2]*time_scale;
// internal forces
P[node1*2-1]=P[node1*2-1]-ax*N;
P[node1*2]=P[node1*2]  -ay*N;
P[node2*2-1]=P[node2*2-1]+ax*N;
P[node2*2]=P[node2*2]  +ay*N;
///-----------------------------------------------------
  double c1,y_value;
double k1,k2,k3,k4,h;
double y1,y2;
int t,i,j,m;
x=0;
param[1]=169.0;
param[2]=1.83e-3;
param[3]=170.0;
param[4]=291.80e6;
param[5]=119.53e6;
param[6]=0.0;
param[7]=0.0;
param[8]=0.267e-3;
param[9]=1.3804e-23;
param[10]=298.0;
param[11]=17.71e-23;
param[12]=19.5E-1;
param[13]=298.0;
param[14]=0.0;
param[15]=12.3750;
param[16]=375.0;
param[17]=375.0;
param[18]=0.0;
param[19]=57.840;

//m=10000;
//double H=160.0;
h=Time_Inc;
double det_A;
x=h;
 delta_n[1]=0.0;
 delta_n[2]=0.0;
 delta_0[1]=0.0;
 delta_0[2]=0.0;
  A[1][1]=1.0-h*df1dy1(x,yn,param);
  A[1][2]=1.0-h*df1dy2(x,yn,param);
  A[2][1]=1.0-h*df2dy1(x,yn,param);
  A[2][2]=1.0-h*df2dy2(x,yn,param);
  det_A=A[1][1]*A[2][2]-A[1][2]*A[2][1];
  A_inv[1][1]=A[2][2]/det_A;
  A_inv[1][2]=-A[1][2]/det_A;
```

```
 A_inv[2][1]=-A[2][1]/det_A;
 A_inv[2][2]=A[1][1]/det_A;
if(t!=0)
{
 //
 for(i=1;i<=2;i++)
 {
  delta_n[i]=delta_n[i]+A_inv[i][1]*h*f1(x,yn,param);
  delta_n[i]=delta_n[i]+A_inv[i][2]*h*f2(x,yn,param);
 }
  for(i=1;i<=2;i++)
  {
   yn[i]=yn[i]+delta_n[i];
  }
}
else if(t==Finish_Time)
{
 for(i=1;i<=2;i++)
 {
  temp_n[i]=0.0;
  temp_n[i]=temp_n[i]+A_inv[i][1]*(h*f1(x,yn,param)-delta_0[1]);
  temp_n[i]=temp_n[i]+A_inv[i][2]*(h*f2(x,yn,param)-delta_0[2]);
  delta_n[i]=temp_n[i];
 }
  for(i=1;i<=2;i++)
  {
    yn[i]=yn[i]+delta_n[i];
  }
 }
else
{
//
 for(i=1;i<=2;i++)
 {
  temp_n[i]=0.0;
  temp_n[i]=temp_n[i]+A_inv[i][1]*(h*f1(x,yn,param)-delta_0[1]);
  temp_n[i]=temp_n[i]+A_inv[i][2]*(h*f2(x,yn,param)-delta_0[2]);
  delta_n[i]=delta_n[i]+2.0*temp_n[i];
 }
  for(i=1;i<=2;i++)
  {
    yn[i]=yn[i]+delta_n[i];
  }
}
   x=x+h;
    phasem1[node1]=yn[1];
    phasem1[node2]=yn[1];
    phasem2[node1]=yn[2];
    phasem2[node2]=yn[2];
 ///-------------------------------------------------
     }// elements
     out_file<<" ************K stiffness*********\n";
     for(i=1;i<=nodes*2;i++)
     {
       for(j=1;j<=nodes*2;j++)
       {
         out_file<<K[i][j]<<" ";
       }
        out_file<<"\n";
     }
     out_file<<"********** load F ***************\n";
     for(i=1;i<=nodes*2;i++)
     {

         out_file<<F[i]<<"\n";
     }
      out_file<<"********** internal load P ***************\n";
     for(i=1;i<=nodes*2;i++)
     {
         out_file<<P[i]<<"\n";
```

```
      }
// apply boundary conditions
int ib,jb;
 ib=0;
 for(i=1;i<=nodes*2;i++)
 {
   if(bound[i]==0)
   {
     ib++;
     jb=0;
     for(j=1;j<=nodes*2;j++)
     {
       if(bound[j]==0)
       {
        jb++;
        KB[ib][jb]=K[i][j];
       }
     }

   }
 }
  out_file<<" **********boundary conditions KB stiffness*********\n";
      for(i=1;i<=nodes*2-num_bound;i++)
      {
        for(j=1;j<=nodes*2-num_bound;j++)
        {
         out_file<<KB[i][j]<<" ";
        }
         out_file<<"\n";
      }
 // boundary conditions applied to load
 ib=0;
 for(i=1;i<=nodes*2;i++)
 {
   if(bound[i]==0)
   {
    ib++;
    FB[ib][1]=F[i]-R[ib];
   }
 }
out_file<<" **********boundary conditions load FB *********\n";
      for(i=1;i<=nodes*2-num_bound;i++)
      {

          out_file<<FB[i][1]<<"\n";

      }
//////////////////////////////////////////////////
///       Solve the reduced matrix and vector      /
//////////////////////////////////////////////////
  for(i=1;i<=nodes*2-num_bound;i++)
  {
   for(j=1;j<=nodes*2-num_bound;j++)
   {
    KM_keep[i][j]=KB[i][j];
   }
  }
  gaussj(KB,nodes*2-num_bound,FB,1);
  out_file<<" **********FB post *********\n";
   for(i=1;i<=nodes*2-num_bound;i++)
      {

          out_file<<FB[i][1]<<"\n";

      }
// store solution in displacement vector
     ib=0;
     for(i=1;i<=nodes*2;i++)
     {
      if(bound[i]==0)
```

```
        {
         ib++;
         u[i]=u[i]+FB[ib][1];
        }
       }
   out_file<<" **********solution *********\n";
        for(i=1;i<=nodes*2;i++)
        {

            out_file<<u[i]<<"\n";

        }
    // calculate residual
    for(i=1;i<=nodes*2-num_bound;i++)
    {
     for(j=1;j<=nodes*2-num_bound;j++)
     {
      R[i]=R[i]+(KM_keep[i][j]*FB[j][1]);
     }
    }
     out_file<<"R: \n";
    for(i=1;i<=nodes*2-num_bound;i++)
    {
     out_file<<R[i]<<"\n";
    }
    for(i=1;i<=nodes*2-num_bound;i++)
    {
     residual[i]=F[i]-R[i];
    }
        out_file<<"residual: \n";
    for(i=1;i<=nodes*2;i++)
    {
     out_file<<residual[i]<<"\n ";
    }
    // calculate maximum residual
    max_residual=0.0;
    for(i=1;i<=nodes*2;i++)
    {
     if(fabs(residual[i])>fabs(max_residual)) max_residual=residual[i];
    }
    iter++;
    out_file<<"iteration: "<<iter<<" maximum residual: "<<max_residual<<"\n";
    not_converged=false;
    if(fabs(max_residual)>TOL) not_converged=true;
    if(iter>max_iter) not_converged=false;

      result=true;
      Form1->Invalidate();
}// not converged
}// time
}
//---------------------------------------------------------------------------
void __fastcall TForm1::DisplayGeom()
{
//
int i;
   double Hmax;
   double Vmax;
   int j;
   // initialize maximum values for plotting
   max_x=-1E15;
   max_y=-1E15;
   min_x=1E15;
   min_y=1E15;
     for(i=1;i<=nodes;i++)
     {
         nx1[i]=nx[i]+u[(i-1)*2+1]*dmag;
         ny1[i]=ny[i]+u[(i-1)*2+2]*dmag;
     }
     for(i=1;i<=nodes;i++)
```

```
 {
  if(nx1[i]>max_x) max_x=nx1[i];
  if(ny1[i]>max_y) max_y=ny1[i];
  if(nx1[i]<min_x) min_x=nx1[i];
  if(ny1[i]<min_y) min_y=ny1[i];
 }
 max_xdime=max_x-min_x;
 max_ydime=max_y-min_y;
 if (min_x<0) shiftx=fabs(min_x);
  else shiftx=0;
 if (min_y<0) shifty=fabs(min_y);
  else shifty=0;
// drawing dimensions
 double lb,rb,tb,bb;
 double fx;
 double fy;
 double x,y;
// page dimensions
 Hmax=ClientWidth;
 Vmax=ClientHeight;
 // page borders
 lb=20;
 rb=0;
 tb=0;
 bb=60;
 Hmax=Hmax-lb-rb;
 Vmax=Vmax-tb-bb;
 Canvas->Pen->Width=1;
 Canvas->Pen->Color=clFuchsia;
 Canvas->Brush->Color=clWhite;
if(geom&&dtype==0)
{
 fx=(Hmax-lb-rb)/max_xdime;
 if(max_ydime!=0)
 {
  fy=(Vmax-tb-bb)/max_ydime;
 }
 else fy=fx;
  int node_number1,node_number2;
  double x1,y1;
    char str[10];
      Canvas->Font->Size=9;

    Canvas->Font->Style=TFontStyles()<<fsBold;
    int x0,y0;
  for(i=1;i<=elements;i++)
  {
        node_number1=elem[i][1];
        node_number2=elem[i][2];
        x=lb+(nx[elem[i][1]]+shiftx)*fx;
        y=(ny[elem[i][1]]+shifty)*fy;
         y=Vmax-y;
         x1=x;
         y1=y;
         x0=x;
         y0=y;
         Canvas->MoveTo(x,y);
         gcvt(node_number1, 4, str);
         Canvas->Font->Color=clBlack;
          Canvas->Brush->Color=clWhite;
         Canvas->TextOut(x,y,str);
         x=lb+(nx[elem[i][2]]+shiftx)*fx;
         y=(ny[elem[i][2]]+shifty)*fy;
         y=Vmax-y;
         gcvt(node_number2, 4, str);
          Canvas->Font->Color=clBlack;
           Canvas->Brush->Color=clWhite;
         Canvas->TextOut(x,y,str);
         Canvas->MoveTo(x1,y1);
         Canvas->LineTo(x,y);
```

```
        gcvt(i, 4, str);
        Canvas->Font->Color=clYellow;
        Canvas->Brush->Color=clBlue;
       Canvas->TextOut((x0+x)/2.0,(y0+y)/2.0,str);

  }
}
if(geom&&result&&dtype==0)
{
  fx=(Hmax-lb-rb)/max_xdime;
  fy=(Vmax-tb-bb)/max_ydime;
     Canvas->Pen->Color=clBlue      ;
     Canvas->Brush->Color=clWhite;
     for(i=1;i<=elements;i++)
     {
        x=lb+(nx1[elem[i][1]]+shiftx)*fx;
        y=(ny1[elem[i][1]]+shifty)*fy;
        y=Vmax-y;
        Canvas->MoveTo(x,y);
        x=lb+(nx1[elem[i][2]]+shiftx)*fx;
        y=(ny1[elem[i][2]]+shifty)*fy;
        y=Vmax-y;
        Canvas->LineTo(x,y);
    }
}
if(dtype>0)
{
char str[10];
Canvas->Font->Size=9;

  fx=(Hmax-lb-rb)/max_xdime;
  fy=(Vmax-tb-bb)/max_ydime;

int nc=7;
int e;
int node1,node2;
double cx1,cx2,cy1,cy2;
double cpx,cpy;
int levels;
double con_x,con_y;
int num_con;
int *c_list;
double *cx_list,*cy_list;
cx_list=vector(1,nc+2);
cy_list=vector(1,nc+2);
c_list=ivector(1,nc+2);
 // find maximum and maximum value of variable
double min_var=0;
double max_var=0;
double *contour_val;
double small_num=0.0;
double factor;
contour_val=vector(1,nc);
 // colour map for contours
 TColor col[9];
   col[0]=0x00FF0000;
   col[1]=0x00FF1000;
   col[2]=0x00FF8000;
   col[3]=0x00FFF000;
   col[4]=0x000FFF00;
   col[5]=0x0000FFFF;
   col[6]=0x00006FF8;
   col[7]=0x00000FF0;
   col[8]=0x00000EF0;
int direction;
for(i=1;i<=nodes;i++)
{
 if(variable[i]<min_var) min_var=variable[i];
 if(variable[i]>max_var) max_var=variable[i];
}
```

```
 if(fabs(min_var)<1E-15)min_var=0.0;
 contour_file<<"max variable: "<<max_var<<" min variable: "<<min_var<<"\n";
// contour levels
contour_file<<"contour_levels: \n";
double delta_temp=(max_var-min_var)/(nc-1) ;
 contour_val[1]=min_var;
 contour_val[2]=min_var+delta_temp;
 contour_val[3]=min_var+delta_temp*2;
 contour_val[4]=min_var+delta_temp*3;
 contour_val[5]=min_var+delta_temp*4;
 contour_val[6]=min_var+delta_temp*5;
 contour_val[7]=min_var+delta_temp*6-0.5*delta_temp;
 for(i=1;i<=nc;i++)
 {
  if(fabs(contour_val[i])<1E-15) contour_val[i]=0.0;
 }
 // contour legend
  Canvas->Font->Color=clBlack;
 Canvas->Pen->Color=clBlack;
 int height=ClientHeight;
   Canvas->Brush->Color=col[7];
   Canvas->Rectangle(lb,height-30,lb+55,height-50);
   Canvas->Brush->Color=col[6];
   Canvas->Rectangle(lb+55,height-30,lb+110,height-50);
   Canvas->Brush->Color=col[5];
   Canvas->Rectangle(lb+110,height-30,lb+165,height-50);
   Canvas->Brush->Color=col[4];
   Canvas->Rectangle(lb+165,height-30,lb+220,height-50);
   Canvas->Brush->Color=col[3];
   Canvas->Rectangle(lb+220,height-30,lb+275,height-50);
   Canvas->Brush->Color=col[2];
   Canvas->Rectangle(lb+275,height-30,lb+330,height-50);
    Canvas->Brush->Color=col[1];
   Canvas->Rectangle(lb+330,height-30,lb+385,height-50);
   Canvas->MoveTo(lb+55,height-30);
   Canvas->LineTo(lb+55,height-10);
   Canvas->MoveTo(lb+110,height-30);
   Canvas->LineTo(lb+110,height-10);
   Canvas->MoveTo(lb+165,height-30);
   Canvas->LineTo(lb+165,height-10);
   Canvas->MoveTo(lb+220,height-30);
   Canvas->LineTo(lb+220,height-10);
   Canvas->MoveTo(lb+275,height-30);
   Canvas->LineTo(lb+275,height-10);
    Canvas->MoveTo(lb+330,height-30);
   Canvas->LineTo(lb+330,height-10);
   Canvas->Brush->Color=clWhite;
     // format numbers for output

   int dec1,sgn1;
   double convert1;
   char *string;
   string=ecvt(delta_temp,4,&dec1,&sgn1);
   contour_file<<"ecvt:"<<string<<" "<<dec1<<" "<<sgn1<<" \n";
   convert1=pow(10,dec1);
   contour_file<<"convert1: "<<convert1<<"\n";
   gcvt(contour_val[7]/convert1, 4, str);
  Canvas->TextOut(lb,height-25,str);

   gcvt(contour_val[6]/convert1, 4, str);
  Canvas->TextOut(lb+60,height-25,str);

   gcvt(contour_val[5]/convert1, 4, str);
  Canvas->TextOut(lb+115,height-25,str);

   gcvt(contour_val[4]/convert1, 4, str);
  Canvas->TextOut(lb+170,height-25,str);

   gcvt(contour_val[3]/convert1, 4, str);
  Canvas->TextOut(lb+225,height-25,str);
```

```
    gcvt(contour_val[2]/convert1, 4, str);
   Canvas->TextOut(lb+280,height-25,str);

    gcvt(contour_val[1]/convert1, 3, str);
    Canvas->TextOut(lb+335,height-25,str);
    // mult factor
    Canvas->TextOut(lb+390,height-25,"*10E");
     Canvas->TextOut(lb+420,height-25,dec1);
     if(dtype==1)  Canvas->TextOut(lb+500,height-25,"X displacement");
 else if(dtype==2)  Canvas->TextOut(lb+500,height-25,"Y displacement");
 else if(dtype==3)  Canvas->TextOut(lb+500,height-25,"Stress");
 else if(dtype==4)  Canvas->TextOut(lb+500,height-25,"Strain");

//
 for(i=1;i<=nc;i++)
 {
  contour_file<<i<<" "<<contour_val[i]<<"\n";
 }
 int lcount=0;
 bool first;
 for(e=1;e<=elements;e++)
 {
   first=true;
 // get node 1 and 2 of element
   node1=elem[e][1];
   node2=elem[e][2];
   contour_file<<node1<<" "<<node2<<"\n";
   cx1=nx[node1];
   cy1=ny[node1];
   cx2=nx[node2];
   cy2=ny[node2];
   contour_file<<"x,y: "<<cx1<<" "<<cy1<<" "<<cx2<<" "<<cy2<<"\n";
   contour_file<<"variable: "<<variable[node1]<<" "<<variable[node2]<<"\n";
 // check within contour band
 double x_old,y_old;
 Canvas->Pen->Width=4;
 int counter=1;
 if(variable[node2]>variable[node1])
 {
   for(i=1;i<=nc;i++)
   {
    if (variable[node1]>=contour_val[i])
    {
     cx_list[1]=cx1;
     cy_list[1]=cy1;
     c_list[1]=i;
    }
   }
 }
 else
 {
   for(i=1;i<=nc;i++)
   {
    if (variable[node2]>=contour_val[i])
    {
      cx_list[1]=cx2;
     cy_list[1]=cy2;
     c_list[1]=i;
    }
   }
 }
 for(i=1;i<=nc;i++)
 {
   ////////////// linear interpolation case 1
   if((variable[node1]<contour_val[i])&&(variable[node2]>contour_val[i]))
   {
    if(fabs(variable[node2]-variable[node1])>small_num)
    {
     factor=(contour_val[i]-variable[node1])/(variable[node2]-variable[node1]);
```

```
    counter++;
   }

   else factor=0.0;
   cpx=cx1+(cx2-cx1)*factor;
   cpy=cy1+(cy2-cy1)*factor;
   cx_list[counter]=cpx;
   cy_list[counter]=cpy;
   c_list[counter]=i;
   direction=1;
   contour_file<<"cpx,cpy: "<<cpx<<" "<<cpy<<" direction 1 \n";
  contour_file<<"i: "<<i<<"\n";
  }
   ////////////// linear interpolation case 2
  else if((variable[node2]<contour_val[i])&&(variable[node1]>contour_val[i]))
  {
   if(fabs(variable[node2]-variable[node1])>small_num)
   {
    factor=(contour_val[i]-variable[node1])/(variable[node2]-variable[node1]);
     counter++;
   }
   else factor=0.0;

   cpx=cx1+(cx2-cx1)*factor;
   cpy=cy1+(cy2-cy1)*factor;
   cx_list[counter]=cpx;
   cy_list[counter]=cpy;
   c_list[counter]=i;
    contour_file<<"cpx,cpy: "<<cpx<<" "<<cpy<<" direction 2 \n";
    Canvas->Pen->Color=col[i];
   x=lb+(cpx)*fx;
   y=(cpy)*fy;
   y=Vmax-y;
   //Canvas->LineTo(x,y);
   contour_file<<"i: "<<i<<"\n";
  }
 }// i
if(variable[node2]>variable[node1])
{
  for(i=1;i<=nc;i++)
  {
   if (variable[node2]>=contour_val[i])
   {
     cx_list[counter+1]=cx2;
    cy_list[counter+1]=cy2;
    c_list[counter+1]=i;
   }
  }
}
else
{
  for(i=1;i<=nc;i++)
  {
  if (variable[node1]>=contour_val[i])
   {
     cx_list[counter+1]=cx1;
    cy_list[counter+1]=cy1;
    c_list[counter+1]=i;
   }
  }
}
contour_file<<"Point List for element "<<e<<": \n";

 int c;
for(c=1;c<=counter+1;c++)
{
 contour_file<<cx_list[c]<<" "<<cy_list[c]<<" "<<c_list[c]<<"\n";
   x=lb+cx_list[c]*fx;
   y=cy_list[c]*fy;
   y=Vmax-y;
```

```
      Canvas->Pen->Color=col[c_list[c]];
      if(first)
       {
         Canvas->MoveTo(x,y);
          first=false;
       }
      else
      {
       Canvas->LineTo(x,y);
      }

    }
   }// e
  }
}

//---------------------------------------------------------

void __fastcall TForm1::FormPaint(TObject *Sender)
{
 Form1->DisplayGeom();
}
//---------------------------------------------------------

void __fastcall TForm1::FormResize(TObject *Sender)
{
 Form1->Invalidate();
}

//---------------------------------------------------------
void __fastcall TForm1::gaussj(double **a, int n, double **b, int m)
{
 int *indxc,*indxr, *ipiv;
 int i,icol,irow,j,k,l,ll;
 double big,dum,pivinv;
 indxc=ivector(1,n);
 indxr=ivector(1,n);
 ipiv=ivector(1,n);
 for(j=1;j<=n;j++) ipiv[j]=0;
 for(i=1;i<=n;i++)
 {
   big=0.0;
   for(j=1;j<=n;j++)
    if(ipiv[j]!=1)
     for(k=1;k<=n;k++)
     {
      if(ipiv[k]==0)
      {
       if (fabs(a[j][k])>=big)
       {
        big=fabs(a[j][k]);
        irow=j;
        icol=k;
       }
      }
     }
    ++(ipiv[icol]);
    if(irow!=icol)
     {
      for(l=1;l<=n;l++) SWAP(a[irow][l],a[icol][l])
      for(l=1;l<=m;l++) SWAP(b[irow][l],b[icol][l])
     }
    indxr[i]=irow;
    indxc[i]=icol;
    pivinv=1.0/a[icol][icol];
    a[icol][icol]=1.0;
    for(l=1;l<=n;l++) a[icol][l]*=pivinv;
    for(l=1;l<=m;l++) b[icol][l]*=pivinv;
    for(ll=1;ll<=n;ll++)
     if(ll!=icol)
```

```
      {
        dum=a[l][icol];
        a[l][icol]=0.0;
        for(l=1;l<=n;l++) a[ll][l]-=a[icol][l]*dum;
        for(l=1;l<=m;l++) b[ll][l]-=b[icol][l]*dum;
      }
    }
    for(l=n;l>=1;l--)
    {
      if(indxr[l]!=indxc[l])
        for(k=1;k<=n;k++)
          SWAP(a[k][indxr[l]],a[k][indxc[l]]);
    }
    free_ivector(ipiv,1,n);
    free_ivector(indxr,1,n);
    free_ivector(indxc,1,n);
}
//---------------------------------------------------------------------------

void __fastcall TForm1::xdisp1Click(TObject *Sender)
{
int i;
 contour_x=false;
 contour_y=true;
 contour_s=false;
 contour_e=false;
  //transfer variable
  for(i=1;i<=nodes;i++)
  {
   variable[i]=u[(i-1)*2+1];
  }
  dtype=1;
 Form1->Invalidate();
}
//---------------------------------------------------------------------------

void __fastcall TForm1::geometry1Click(TObject *Sender)
{
// geometry
  dmag=0;
 dtype=0;
 Form1->Invalidate();
}
//---------------------------------------------------------------------------

void __fastcall TForm1::displaced1Click(TObject *Sender)
{
 dmag=100;
 dtype=0;
 Form1->Invalidate();
}
//---------------------------------------------------------------------------

void __fastcall TForm1::ydisp1Click(TObject *Sender)
{
int i;
  //transfer variable
  for(i=1;i<=nodes;i++)
  {
   variable[i]=u[(i-1)*2+2];
  }
  dtype=2;
 Form1->Invalidate();
}
//---------------------------------------------------------------------------

void __fastcall TForm1::stress1Click(TObject *Sender)
{
```

```
// display stress
int i;
  //transfer variable
  for(i=1;i<=nodes;i++)
  {
   variable[i]=stress_v[i];
  }
  dtype=3;
 Form1->Invalidate();
}
//---------------------------------------------------------------------------

void __fastcall TForm1::strain1Click(TObject *Sender)
{
 // display stress
int i;
  //transfer variable
  for(i=1;i<=nodes;i++)
  {
   variable[i]=strain_v[i];
  }
  dtype=4;
 Form1->Invalidate();
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Dmag1Click(TObject *Sender)
{
dmag=dmag*2.0;
Form1->Invalidate();
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Dmag2Click(TObject *Sender)
{
dmag=dmag/2.0;
Form1->Invalidate();
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Dmag01Click(TObject *Sender)
{
dmag=100;
Form1->Invalidate();
}
//---------------------------------------------------------------------------
//---------------------------------------------------------------------------
double f1(double t,double *yn,double *param)
{
 double a,deltaL,Ea,Em,deltaR,J;
      double b,d,SHM,Temp,TR,EPM,ENM,k0,V;
      double Psi0,SHA,EPA,ENA,FM,FA;
      double minus_A,A_minus,A_plus,plus_A,IE,deltaS,Psi;
      double PPA,PAP,PMA,PAM,omega,theta;
      // parameters
      IE=param[1];
      J=param[2];
      Psi=param[3];
      Ea=param[4];
      Em=param[5];
      deltaS=param[6];
      deltaL=param[7];
      deltaR=param[8];
      k0=param[9];
      Temp=param[10];
      V=param[11];
      omega=param[12];
      TR=param[13];
      EPM=param[14];
      EPA=param[15];
```

```
    SHM=param[16];
    SHA=param[17];
    ENM=param[18];
    ENA=param[19];
    char str[25];
    int sig = 8; /* significant digits */

  a=-(deltaL*Ea + Em*(-deltaR + J))/(2.*(-deltaL + deltaR));

  b=(deltaL*(deltaR*Ea + Em*(-deltaR + J)))/(-deltaL + deltaR);

  d=(Em*J*J)/2. - (deltaR*deltaR*(deltaL*Ea + Em*(-deltaL + J)))
  /(2.*(-deltaL + deltaR)) + SHM*(Temp - TR) + EPM
  - Temp*(ENM + SHM*log(Temp/TR));

  Psi0=(deltaL*deltaR*(-Ea + Em))/2. - ((deltaL + deltaR)*Em*J)/2.0
  +(Em*J*J)/2. - (SHA - SHM)*(Temp - TR) - EPA + EPM
  +Temp*(ENA - ENM + (SHA - SHM)*log(Temp/TR));

  FM=SHM*(Temp - TR) + EPM - Temp*(ENM + SHM*log(Temp/TR));
  FA=SHA*(Temp - TR) + EPA - Temp*(ENA + SHA*log(Temp/TR));

  minus_A=b*b/(4.*a) - d - FM + IE + deltaS*AP(t)
  - J*AP(t) + AP(t)*AP(t)/(2.*Em) + Psi;

  A_minus=b*b/(4.*a) - d - FA - IE + deltaS*AP(t)
  + AP(t)*AP(t)/(2.*Ea)+Psi - Psi0 ;

  plus_A=b*b/(4.*a) - d - FM + IE - deltaS*AP(t)
  + J*AP(t) + AP(t)*AP(t)/(2.*Em) + Psi ;

  A_plus=b*b/(4.*a) - d - FA - IE - deltaS*AP(t)
  + AP(t)*AP(t)/(2.*Ea)+Psi - Psi0 ;
  theta = V/(k0*Temp);

  PMA = (1/omega)*exp(-(minus_A*theta));

  PAM = (1/omega)*exp(-(A_minus*theta));

  PPA = (1/omega)*exp(-(plus_A*theta));

  PAP = (1/omega)*exp(-(A_plus*theta));

 double f;

   f=-PAP*yn[2]-(PPA+PAP)*yn[1]+PAP;
 return f;
}
//-------------------------------------
double f2(double t,double *yn,double *param)
{

    double a,deltaL,Ea,Em,deltaR,J;
    double b,d,SHM,Temp,TR,EPM,ENM,k0,V;
    double Psi0,SHA,EPA,ENA,FM,FA;
    double minus_A,A_minus,A_plus,plus_A,IE,deltaS,Psi;
    double PPA,PAP,PMA,PAM,omega,theta;
    // parameters
    IE=param[1];
    J=param[2];
    Psi=param[3];
    Ea=param[4];
    Em=param[5];
    deltaS=param[6];
    deltaL=param[7];
    deltaR=param[8];
    k0=param[9];
    Temp=param[10];
    V=param[11];
    omega=param[12];
```

```
    TR=param[13];
    EPM=param[14];
    EPA=param[15];
    SHM=param[16];
    SHA=param[17];
    ENM=param[18];
    ENA=param[19];
    char str[25];
    int sig = 8; /* significant digits */
  a=-(deltaL*Ea + Em*(-deltaR + J))/(2.*(-deltaL + deltaR));

  b=(deltaL*(deltaR*Ea + Em*(-deltaR + J)))/(-deltaL + deltaR);

  d=(Em*J*J)/2. - (deltaR*deltaR*(deltaL*Ea + Em*(-deltaL + J)))
  /(2.*(-deltaL + deltaR)) + SHM*(Temp - TR) + EPM
  - Temp*(ENM + SHM*log(Temp/TR));

  Psi0=(deltaL*deltaR*(-Ea + Em))/2. - ((deltaL + deltaR)*Em*J)/2.0
  +(Em*J*J)/2. - (SHA - SHM)*(Temp - TR) - EPA + EPM
  +Temp*(ENA - ENM + (SHA - SHM)*log(Temp/TR));

  FM=SHM*(Temp - TR) + EPM - Temp*(ENM + SHM*log(Temp/TR));
  FA=SHA*(Temp - TR) + EPA - Temp*(ENA + SHA*log(Temp/TR));

  minus_A=b*b/(4.*a) - d - FM + IE + deltaS*AP(t)
  - J*AP(t) + AP(t)*AP(t)/(2.*Em) + Psi ;

  A_minus=b*b/(4.*a) - d - FA - IE + deltaS*AP(t)
  + AP(t)*AP(t)/(2.*Ea)+Psi - Psi0 ;

  plus_A=b*b/(4.*a) - d - FM + IE - deltaS*AP(t)
  + J*AP(t) + AP(t)*AP(t)/(2.*Em) + Psi ;

  A_plus=b*b/(4.*a) - d - FA - IE - deltaS*AP(t)
  + AP(t)*AP(t)/(2.*Ea)+Psi - Psi0 ;
  theta = V/(k0*Temp);

  PMA = (1/omega)*exp(-(minus_A*theta));

  PAM = (1/omega)*exp(-(A_minus*theta));

  PPA = (1/omega)*exp(-(plus_A*theta));

  PAP = (1/omega)*exp(-(A_plus*theta));

 double f;

   f=-(PMA+PAM)*yn[2]-PAP*yn[1]+PAM;
 return f;
}
double AP(double t)
{
// define load
double F;
//F=19E3*sin(0.05823529*t);
F=N;
return F;
}
//---------------------------------------------------------
double df1dy1(double t,double *yn,double *param)
{
 double a,deltaL,Ea,Em,deltaR,J;
      double b,d,SHM,Temp,TR,EPM,ENM,k0,V;
      double Psi0,SHA,EPA,ENA,FM,FA;
      double minus_A,A_minus,A_plus,plus_A,IE,deltaS,Psi;
      double PPA,PAP,PMA,PAM,omega,theta;
      // parameters
      IE=param[1];
      J=param[2];
      Psi=param[3];
```

```
    Ea=param[4];
    Em=param[5];
    deltaS=param[6];
    deltaL=param[7];
    deltaR=param[8];
    k0=param[9];
    Temp=param[10];
    V=param[11];
    omega=param[12];
    TR=param[13];
    EPM=param[14];
    EPA=param[15];
    SHM=param[16];
    SHA=param[17];
    ENM=param[18];
    ENA=param[19];
    char str[25];
    int sig = 8; /* significant digits */

  a=-(deltaL*Ea + Em*(-deltaR + J))/(2.*(-deltaL + deltaR));

  b=(deltaL*(deltaR*Ea + Em*(-deltaR + J)))/(-deltaL + deltaR);

  d=(Em*J*J)/2. - (deltaR*deltaR*(deltaL*Ea + Em*(-deltaL + J)))
  /(2.*(-deltaL + deltaR)) + SHM*(Temp - TR) + EPM
  - Temp*(ENM + SHM*log(Temp/TR));

  Psi0=(deltaL*deltaR*(-Ea + Em))/2. - ((deltaL + deltaR)*Em*J)/2.0
  +(Em*J*J)/2. - (SHA - SHM)*(Temp - TR) - EPA + EPM
  +Temp*(ENA - ENM + (SHA - SHM)*log(Temp/TR));

  FM=SHM*(Temp - TR) + EPM - Temp*(ENM + SHM*log(Temp/TR));
  FA=SHA*(Temp - TR) + EPA - Temp*(ENA + SHA*log(Temp/TR));

  minus_A=b*b/(4.*a) - d - FM + IE + deltaS*AP(t)
  - J*AP(t) + AP(t)*AP(t)/(2.*Em) + Psi;

  A_minus=b*b/(4.*a) - d - FA - IE + deltaS*AP(t)
  + AP(t)*AP(t)/(2.*Ea)+Psi - Psi0 ;

  plus_A=b*b/(4.*a) - d - FM + IE - deltaS*AP(t)
  + J*AP(t) + AP(t)*AP(t)/(2.*Em) + Psi ;

  A_plus=b*b/(4.*a) - d - FA - IE - deltaS*AP(t)
  + AP(t)*AP(t)/(2.*Ea)+Psi - Psi0 ;
  theta = V/(k0*Temp);

  PMA = (1/omega)*exp(-(minus_A*theta));

  PAM = (1/omega)*exp(-(A_minus*theta));

  PPA = (1/omega)*exp(-(plus_A*theta));

  PAP = (1/omega)*exp(-(A_plus*theta));

 double f;
  f=-(PPA+PAP);
 return f;
}
//-------------------------------
double df1dy2(double t,double *yn,double *param)
{

    double a,deltaL,Ea,Em,deltaR,J;
    double b,d,SHM,Temp,TR,EPM,ENM,k0,V;
    double Psi0,SHA,EPA,ENA,FM,FA;
    double minus_A,A_minus,A_plus,plus_A,IE,deltaS,Psi;
    double PPA,PAP,PMA,PAM,omega,theta;
    // parameters
    IE=param[1];
```

```
    J=param[2];
    Psi=param[3];
    Ea=param[4];
    Em=param[5];
    deltaS=param[6];
    deltaL=param[7];
    deltaR=param[8];
    k0=param[9];
    Temp=param[10];
    V=param[11];
    omega=param[12];
    TR=param[13];
    EPM=param[14];
    EPA=param[15];
    SHM=param[16];
    SHA=param[17];
    ENM=param[18];
    ENA=param[19];
    char str[25];
    int sig = 8; /* significant digits */
  a=-(deltaL*Ea + Em*(-deltaR + J))/(2.*(-deltaL + deltaR));

  b=(deltaL*(deltaR*Ea + Em*(-deltaR + J)))/(-deltaL + deltaR);

  d=(Em*J*J)/2. - (deltaR*deltaR*(deltaL*Ea + Em*(-deltaL + J)))
  /(2.*(-deltaL + deltaR)) + SHM*(Temp - TR) + EPM
  - Temp*(ENM + SHM*log(Temp/TR));

  Psi0=(deltaL*deltaR*(-Ea + Em))/2. - ((deltaL + deltaR)*Em*J)/2.0
  +(Em*J*J)/2. - (SHA - SHM)*(Temp - TR) - EPA + EPM
  +Temp*(ENA - ENM + (SHA - SHM)*log(Temp/TR));

  FM=SHM*(Temp - TR) + EPM - Temp*(ENM + SHM*log(Temp/TR));
  FA=SHA*(Temp - TR) + EPA - Temp*(ENA + SHA*log(Temp/TR));

  minus_A=b*b/(4.*a) - d - FM + IE + deltaS*AP(t)
  - J*AP(t) + AP(t)*AP(t)/(2.*Em) + Psi ;

  A_minus=b*b/(4.*a) - d - FA - IE + deltaS*AP(t)
  + AP(t)*AP(t)/(2.*Ea)+Psi - Psi0 ;

  plus_A=b*b/(4.*a) - d - FM + IE - deltaS*AP(t)
  + J*AP(t) + AP(t)*AP(t)/(2.*Em) + Psi ;

  A_plus=b*b/(4.*a) - d - FA - IE - deltaS*AP(t)
  + AP(t)*AP(t)/(2.*Ea)+Psi - Psi0 ;
  theta = V/(k0*Temp);

  PMA = (1/omega)*exp(-(minus_A*theta));

  PAM = (1/omega)*exp(-(A_minus*theta));

  PPA = (1/omega)*exp(-(plus_A*theta));

  PAP = (1/omega)*exp(-(A_plus*theta));

 double f;
 f=-PAP;

 return f;
}

//-------------------------------------------------------
double df2dy1(double t,double *yn,double *param)
{
 double a,deltaL,Ea,Em,deltaR,J;
      double b,d,SHM,Temp,TR,EPM,ENM,k0,V;
      double Psi0,SHA,EPA,ENA,FM,FA;
      double minus_A,A_minus,A_plus,plus_A,IE,deltaS,Psi;
      double PPA,PAP,PMA,PAM,omega,theta;
```

```
    // parameters
    IE=param[1];
    J=param[2];
    Psi=param[3];
    Ea=param[4];
    Em=param[5];
    deltaS=param[6];
    deltaL=param[7];
    deltaR=param[8];
    k0=param[9];
    Temp=param[10];
    V=param[11];
    omega=param[12];
    TR=param[13];
    EPM=param[14];
    EPA=param[15];
    SHM=param[16];
    SHA=param[17];
    ENM=param[18];
    ENA=param[19];
    char str[25];
    int sig = 8; /* significant digits */

  a=-(deltaL*Ea + Em*(-deltaR + J))/(2.*(-deltaL + deltaR));

  b=(deltaL*(deltaR*Ea + Em*(-deltaR + J)))/(-deltaL + deltaR);

  d=(Em*J*J)/2. - (deltaR*deltaR*(deltaL*Ea + Em*(-deltaL + J)))
  /(2.*(-deltaL + deltaR)) + SHM*(Temp - TR) + EPM
  - Temp*(ENM + SHM*log(Temp/TR));

  Psi0=(deltaL*deltaR*(-Ea + Em))/2. - ((deltaL + deltaR)*Em*J)/2.0
  +(Em*J*J)/2. - (SHA - SHM)*(Temp - TR) - EPA + EPM
  +Temp*(ENA - ENM + (SHA - SHM)*log(Temp/TR));

  FM=SHM*(Temp - TR) + EPM - Temp*(ENM + SHM*log(Temp/TR));
  FA=SHA*(Temp - TR) + EPA - Temp*(ENA + SHA*log(Temp/TR));

  minus_A=b*b/(4.*a) - d - FM + IE + deltaS*AP(t)
  - J*AP(t) + AP(t)*AP(t)/(2.*Em) + Psi;

  A_minus=b*b/(4.*a) - d - FA - IE + deltaS*AP(t)
  + AP(t)*AP(t)/(2.*Ea)+Psi - Psi0 ;

  plus_A=b*b/(4.*a) - d - FM + IE - deltaS*AP(t)
  + J*AP(t) + AP(t)*AP(t)/(2.*Em) + Psi ;

  A_plus=b*b/(4.*a) - d - FA - IE - deltaS*AP(t)
  + AP(t)*AP(t)/(2.*Ea)+Psi - Psi0 ;
  theta = V/(k0*Temp);

  PMA = (1/omega)*exp(-(minus_A*theta));

  PAM = (1/omega)*exp(-(A_minus*theta));

  PPA = (1/omega)*exp(-(plus_A*theta));

  PAP = (1/omega)*exp(-(A_plus*theta));

 double f;

   f=-PAP;

 return f;
}
//-------------------------------------
double df2dy2(double t,double *yn,double *param)
{

     double a,deltaL,Ea,Em,deltaR,J;
```

```
    double b,d,SHM,Temp,TR,EPM,ENM,k0,V;
    double Psi0,SHA,EPA,ENA,FM,FA;
    double minus_A,A_minus,A_plus,plus_A,IE,deltaS,Psi;
    double PPA,PAP,PMA,PAM,omega,theta;
    // parameters
    IE=param[1];
    J=param[2];
    Psi=param[3];
    Ea=param[4];
    Em=param[5];
    deltaS=param[6];
    deltaL=param[7];
    deltaR=param[8];
    k0=param[9];
    Temp=param[10];
    V=param[11];
    omega=param[12];
    TR=param[13];
    EPM=param[14];
    EPA=param[15];
    SHM=param[16];
    SHA=param[17];
    ENM=param[18];
    ENA=param[19];
    char str[25];
    int sig = 8; /* significant digits */
  a=-(deltaL*Ea + Em*(-deltaR + J))/(2.*(-deltaL + deltaR));

  b=(deltaL*(deltaR*Ea + Em*(-deltaR + J)))/(-deltaL + deltaR);

  d=(Em*J*J)/2. - (deltaR*deltaR*(deltaL*Ea + Em*(-deltaL + J)))
  /(2.*(-deltaL + deltaR)) + SHM*(Temp - TR) + EPM
  - Temp*(ENM + SHM*log(Temp/TR));

  Psi0=(deltaL*deltaR*(-Ea + Em))/2. - ((deltaL + deltaR)*Em*J)/2.0
  +(Em*J*J)/2. - (SHA - SHM)*(Temp - TR) - EPA + EPM
  +Temp*(ENA - ENM + (SHA - SHM)*log(Temp/TR));

  FM=SHM*(Temp - TR) + EPM - Temp*(ENM + SHM*log(Temp/TR));
  FA=SHA*(Temp - TR) + EPA - Temp*(ENA + SHA*log(Temp/TR));

  minus_A=b*b/(4.*a) - d - FM + IE + deltaS*AP(t)
  - J*AP(t) + AP(t)*AP(t)/(2.*Em) + Psi ;

  A_minus=b*b/(4.*a) - d - FA - IE + deltaS*AP(t)
  + AP(t)*AP(t)/(2.*Ea)+Psi - Psi0 ;

  plus_A=b*b/(4.*a) - d - FM + IE - deltaS*AP(t)
  + J*AP(t) + AP(t)*AP(t)/(2.*Em) + Psi ;

  A_plus=b*b/(4.*a) - d - FA - IE - deltaS*AP(t)
  + AP(t)*AP(t)/(2.*Ea)+Psi - Psi0 ;
  theta = V/(k0*Temp);

  PMA = (1/omega)*exp(-(minus_A*theta));

  PAM = (1/omega)*exp(-(A_minus*theta));

  PPA = (1/omega)*exp(-(plus_A*theta));

  PAP = (1/omega)*exp(-(A_plus*theta));

 double f;
  f=-(PMA+PAM);
 return f;
}

void __fastcall TForm1::Start1Click(TObject *Sender)
{
// start the analysis
```

```
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Start2Click(TObject *Sender)
{
// time stepping for the analysis
Time_step->Show();
}
//---------------------------------------------------------------------------


void __fastcall TForm1::phasem11Click(TObject *Sender)
{
//
// display phase
int i;
  //transfer variable
  for(i=1;i<=nodes;i++)
  {
   variable[i]=phasem1[i];
  }
  dtype=5;
 Form1->Invalidate();
}
//---------------------------------------------------------------------------

void __fastcall TForm1::phasem21Click(TObject *Sender)
{
 // display phase
int i;
  //transfer variable
  for(i=1;i<=nodes;i++)
  {
   variable[i]=phasem2[i];
  }
  dtype=6;
 Form1->Invalidate();
}
//---------------------------------------------------------------------------
```

REFERENCES

1. Allafi, J.K, Ren, X. & Eggeler, G. 2002: The mechanism of multistage martensitic transformations in aged Ni-rich NiTi Shape Memory Alloys, Acta Materialia, 50, pp. 793-803.
2. Auricchio, F. & Sacco, E. 2001: Thermo-mechanical modelling of a superelastic shape-memory wire under cyclic stretching-bending loadings, International Journal of Solids and Structures, 38, pp. 6123-6145.
3. Bekker, A. & Brinson, L.C 1998: Phase diagram based description of the hysteresis behavior of shape memory alloys, Acta Materialia, Vol. 46, No. 10, pp. 3649-3665.
4. Bhattacharyya, A., Lagoudas, D.C, Wang, Y. & Kinra, V.K: On the role of thermoelectric heat transfer in the design of SMA actuators: Theoretical modelling and Experiment, Smart Mater. Struct., 4, 252.
5. Bhattacharyya, A. & Lagoudas, D.C 1997: A stochastic thermodynamic model for the gradual thermal transformation of SMA polycrystals, Journal of Solids and Structures, 6, pp. 235-250.
6. Bhattacharyya, K: Theory of martensitic microstructure and the shape memory effect, Internal Publication, Division of Engineering & Applied Science, 104-44, California Institute of Technology, Pasadena, CA 91125, USA.
7. Bo, Z., Lagoudas, D.C & Miller, D 1998: Material characterization of SMA Actuators under non-proportional thermomechanical loading, Submitted to Journal of Engineering Materials & Technology, July 31, 1998.

8. Boyd, J.G & Lagoudas, D.C 1996: A Thermodynamic constitutive model for the shape memory material Part I. The monolithic shape memory alloy, International Journal of Plasticity, Vol. 12, No. 6, pp. 805-842.
9. Brocca, M., Brinson, L.C & Bazant, Z.P 2002: Three-dimensional constitutive model for shape memory alloys based on microplane model, Journal of the Mechanics and Physics of Solids, 50, pp. 1051-1077.
10. Chen, W.C, QuiPing, W, Kang, J.H & Winfree, N. A 2001: Compressive superelastic behavior of a NiTi shape memory alloy at strain rates of 0.001-750 s^{-1}, International Journal of Solids and Structures, 38, pp. 8989-8998.
11. Chen, Q. & Levy, E 1996: Active vibration control of elastic beam by means of shape memory alloy layers, Smart Mater. Struct., 5, pp. 400-406.
12. Edelen, D.G.B & Lagoudas, D.C 1999: Matching the inner and outer solutions in the continuum theory of dislocations, International Journal of Engineering Science, 37, pp. 59-73.
13. Faciu, C & Mihailescu-Suliciu, M 2002: On modelling phase propagation in SMAs by a Maxwellian thermo-viscoelastic approach, International Journal of Solids and Structures, 39, pp. 3811-3830.
14. Fukuda, T, Kakeshita, T, Houjoh, H, Shiraishi, S & Saburi, T 1999: Electronic structure and stability of intermetallic compounds in the Ti-Ni System, Materials Science and Engineering, A273-275, pp.166-169.
15. Gall, K., Sehitoglu, H., Anderson, R., Karaman, I., Chumlyakov, Y.I & Kireeva, I.V 2001: On the behavior of single crystal NiTi shape memory alloys and related polycrystalline phenomenon, Materials Science and Engineering, A317, pp. 85-92.
16. Gall, K., Juntunen, K., Maier, H.J, Sehitoglu, H.J & Chumlyakov, Y.I 2001: Instrumented micro-indentation of NiTi shape memory alloys, Acta mater. 49, pp. 3205-3217.
17. Govindjee, S. & Hall, G.J 2000: A computational model for shape memory alloys, International Journal of Solids and Structures, 37, pp. 735-760.
18. Helm, D. & Haupt, P. 2003: Shape memory behaviour: modelling within continuum thermomechanics, International Journal of Solids and Structures, 40, pp. 827-849.

19. Huang, W. 1999: "Yield" surface of shape memory alloys and their applications, Acta mater., Vol. 47, No. 9, pp. 2769-2776.
20. Huo, Y. 1993: On the thermodynamics of pseudoelasticity, Thesis submitted towards the degree of Dr.-Ing., Technical University of Berlin.
21. Huo, Y. & Müller, I. 1993: Nonequilibrium thermodynamics of pseudoelasticity, Continuum Mechanics and Thermodynamics, 5, pp. 163-204.
22. Khelfaoui, F., Thollet, G. & Guenin, G. 2002: Microstructural kinetics after plastic deformation of equiatomic Ti-Ni alloy during isothermal annealings, Materials Science and Engineering, A00, pp. 1-8.
23. Lagoudas, D.C & Bo, Z. 1994: The cylindrical bending of composite plates with piezoelectric and SMA layers, Smart Mater, Struct., 3, pp. 309-317.
24. Lagoudas, D.C, Garner, L.J, Rediniotis, O.K & Wilson, N.: Modeling and experiments of the hysteretic response of an active hydrofoil actuated by SMA line actuators, Internal Publication, Aerospace Engineering Department, Texas A & M University, College Station, TX 77843-3141.
25. Lagoudas, D.C & Bhattacharyya, A.: Modelling of thin layer thermoelectric SMA actuators, Internal Publication, Aerospace Engineering Department, Texas A & M University, College Station, TX 77843-3141.
26. Lagoudas, D.C, Bo, Z. & Qidwai, M. A 1996: A unified thermodynamic model for SMA and Finite Element Analysis of Active Metal Matrix, Composite Materials and Structures, Vol. 3, pp. 153-179.
27. Leclercq, S. & Lexcellent, C. 1996: A general macroscopic description of the thermomechanical behavior of shape memory alloys, J. Mech. Phys. Solids, Vol. 44, No. 6, pp. 953-980.
28. Liu, Y., Xie, Z.L, Van Hambeeck, J. & Delaey, L. 1999: Effect of texture orientation on the martensite deformation of NiTi shape memory alloy sheet, Acta mater., Vol. 47, No. 2, pp. 645-660.
29. Lu, T.J, Hutchinson, J.W & Evans, A.G 2001: Optimal design of a flexural actuator, Journal of Mechanics and Physics of Solids, 49, pp. 2071-2093.
30. Lu, Z.K & Weng, G.J 1997: Martensitic transformation and stress-strain relations of shape-memory alloys, Journal of Mechanics and Physics of Solids, Vol. 45, No. 11/12, pp. 1905-1928.

31. Miller, D.A & Lagoudas, D.C 2001: Influence of cold work and heat treatment on the shape memory effect and plastic strain development of NiTi, Materials Science and Engineering, A 308, pp. 161-175.
32. Müller, I. & Seelecke, S. 2001: Analysis, Modeling and Simulation of Multiscale Problems, Thermodynamic Aspects of Shape Memory Alloys, Preprint 38, Math. & Comp. Modeling, 34, Peragon Press, Oxford, New York.
33. Ohta, T. 2001: Theory of rubber-like elasticity in shape memory alloys, Materials Science and Engineering, A 312, pp. 57-65.
34. Park, C., Walz, C. & Chorpa, I. 1996:Bending and torsion models of beams with induced strain actuators, Smart Mater. Struct., 5, pp. 98-113.
35. Peng, X., Yang, Y. & Haung, S. 2001: A comprehensive description of shape memory alloys with a two phase constitutive model, International Journal of Solids and Structures, 38, pp. 6925-6940.
36. Qidwai, M.A & Lagoudas, D.C 2000: On thermomechanics and transformation surfaces of polycrystalline NiTi shape memory alloy material, International Journal of Plasticity, 16, pp. 1309-1343.
37. Qidwai, M.A & Lagoudas, D.C 1999: Numerical implementation of a shape memory alloy thermomechanical constitutive model using return mapping algorithms, Journal of Numerical Methods in Engineering, 47, pp. 1123-1168.
38. Sawada, T., Tobushi, H., Kimura, K., Hattori, T., Tanaka, K. & Lin, P. 1993: Stress-strain-temperature relationship associated with the R-phase transformation in TiNi shape memory alloy (Influence of shape memory processing temperature), JSME International Journal, Series A, Vol. 36, No. 4, pp. 395-401.
39. Seelecke, S. 1997: Torsional vibration of a shape memory wire, Cont. Mech. Therm., 9, pp. 165-173.
40. Seelecke, S. 1997: FE Simulation and Optimal Control of Adaptive Beams Using SMA Wires, Internal Publication, Institut für Verfahrenstechnik, Technical University of Berlin, Sekr. HF2, Straße des 17 Juni 135, D-10623 Berlin.
41. Seelecke, S. & Büskens, C. 1997: Optimal control of beam structures by shape memory wires, in Proceedings of the OPTI 97, Computer Aided

Optimum Design of Structures, Hernàndez, S., Brebbia, C.A. (eds.), September 8-10, 1997, Rome, Italy, Comp. Mech. Press, 1997.

42. Seelecke, S. & Müller, I. 2000: SMA actuators in smart structures, Internal Publication, Institut für Verfahrenstechnik, Technical University of Berlin, Sekr. HF2, Straβe des 17 Juni 135, D-10623 Berlin.
43. Shakeri, C., Noori, M.N & Hou, Z.: Smart Materials and Structures, A Review, http://me.wpi.edu/~cirrus/Publications/SmartMaterials/SmartmaterialExtension.html
44. Shaw, J.A 2000: Simulations of localized thermo-mechanical behavior in a NiTi shape memory alloy, International Journal of Plasticity, Vol. 16, pp. 541-652.
45. Shaw, J.A 2002: A thermomechanical model for a 1-D shape memory alloy wire with propagating instabilities, International Journal of Solids and Structures, 39, pp. 1275-1305.
46. Shaw, J.A & Kyriakides, S. 1995: Thermomechanical aspects of NiTi, J. Mech. Phys. Solids, Vol. 43, No. 8, pp. 1243-1281.
47. Shaw, J.A & Kyriakides, S. 1997: On the nucleation and propogation of phase transformation fronts in NiTi alloy, Acta mater., Vol 45, No. 2, pp. 683-700.
48. Shaw, J.A & Kyriakides, S. 1998: Initiation and propogation of localized deformation in elasto-plastic strips under uniaxial tension, International Journal of Plasticity, Vol. 13, No. 10, pp. 837-871.
49. Shu, S.G, Lagoudas, D.C, Hughes, D. & Wen, J,T 1997: Modeling of a flexible beam actuated by shape memory alloy wires, Smart Mater. Struct., 6, pp. 265-277.
50. Smith, R.C, Seelecke, S., Dapino, M. & Ounaies, Z.: A unified framework for modelling hysteresis in ferroic materials, Internal Publication, Centre for Research in Scientific Computation, North Carolina State University, Raleigh, NC 27695.
51. Sun, Q. & Li, Z. 2002: Phase transformation in superelastic NiTi polycrystalline micro-tubes under tension and torsion – from localization to homogeneous deformation, International Journal of Solids and Structures, 39, pp. 3797-3809.

52. Tanaka, K., Kobayashi, S. & Sato, Y. 1986: Thermomechanics of transformation pseudoelasticity and shape memory effect in alloys, International Journal of Plasticity, Vol. 2, pp. 59-72.
53. Tsoi, K.A, Stalmans, R. & Schrooten, J. 2002: Transformational behaviour of constrained shape memory alloys, Acta Materialia, 50, pp. 3535-3544.
54. Van Hambeeck, J. 1999: Non-medical applications of shape memory alloys, Materials Science and Engineering, A273-275, pp. 134-148.
55. Webb, G.V, Kurdila, A.J & Lagoudas, D.C 1998: Hysteresis modelling of SMA actuators for control applications, Final paper sent to JIMSS.
56. Xie, Z., Liu, Y. & Van Hambeeck, J. 2000: Microstructure of NiTi shape memory alloy due to tension-compression cyclic deformation, Acta mater., Vol. 46, No. 6, pp. 1989-2000.
57. Zhu, J., Liang, N., Huang, W., Liew, K.M & Liu, Z. 2002: A thermodynamic constitutive modles for stress induced phase transformation in shape memory alloys, International Journal of Solids and Structures, 39, pp. 741-763.
58. http://www.thermodynamik.tu-berlin.de/haupt/simulation/Sma_Sim_Background.html
59. Bathe, K.-J. 1982: Finite Element Procedures in Engineering Analysis, Prentice-Hall, Inc., Englewood Cliffs, New Jersey 07632.
60. Bathe, K.-J. 1996: Finite Element Procedures, Prentice-Hall, Englewood Cliffs, New Jersey 07632.
61. Atkinson, L.V., Harley, P.J. & Hudson, J.D. 1989: Numerical Methods with FORTRAN 77, A Practical Introduction, T.J. Press (Padstow), Cornwall, Great Britain.
62. Press, W.H., Teukolsky, S.A., Vetterling, W.T. & Flannery, B.P. 1994: Numerical Recipes in C, The Art of Scientific Computing, Second Edition, Press Syndicate of the University of Cambridge, The Pitt Building, Trumpington Street, Cambridge CB2 1RP, 40 West 20th Street, New York, NY 10011-4211, USA, 10 Stanford Road, Oakleigh, Melbourne 3166, Australia.
63. Masuda, A., Sone, A., Kamata, S. & Yamashita, Y. 2003: Finite element analysis of shape memory alloy springs designed for base isolation,

Internal publication: Department of Mechanical and System Engineering, Kyoto Institute of Technology.