IMPROVED FLEXIBLE ARMOR DESIGN

William Houghton
David Roylance

Advanced Composites Laboratories
224 Calvary Street
Waltham, MA 02154

First Phase Final Report

Contract DAAK60-87-R-0042
U.S. Army Natick RD&E Center

August 1989

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Advanced Composites Laboratories 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION U.S. Army Natick Research, Development and Engineering Center
6c. ADDRESS (City, State, and ZIP Code) 224 Calvary Street Waltham, MA 02154	7b. ADDRESS (City, State, and ZIP Code) Natick, MA 01760
8a. NAME OF FUNDING/SPONSORING ORGANIZATION U.S. Army Natick R,D&E Center 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code) Natick, MA 01760	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO. 015

11. TITLE (Include Security Classification)
Improved Flexible Armor Design (U)

12. PERSONAL AUTHOR(S) William Houghton and David Roylance

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
First Phase - Final	FROM TO	1989, August 28	49

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

A simple but effective numerical method for analysis of ballistic impact on a woven fabric panels has been extended to treat the case in which the fabric is protected against uong slender projectiles by means of hard covering plates. The principles and implementation of the numerical method are described, following which the computer code is used in a series of numerical experiments to assess the influence of these plates on the wave propagation patterns in impacted panels. Based on these results, three armor configurations are recommended for further study: Kevlar fabric fronted by titanium plates, Spectra fabric fronted by titanium plates, and Spectra fabric fronted by B_4C plates.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified	
22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE (Include Area Code)	22c. OFFICE SYMBOL

DD FORM 1473, 84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

Unclassified

Contents

List of Figures

Abstract

A simple but effective numerical method for analysis of ballistic impact on a woven fabric panel has been extended to treat the case in which the fabric is protected against long slender projectiles by means of hard covering plates. The principles and implementation of the numerical method are described, following which the computer code is used in a series of numerical experiments to assess the influence of these plates on the wave propagation patterns in impacted panels. Based on these results, three armor configurations are recommended for further study: Kevlar fabric fronted by titanium plates, Spectra fabric fronted by titanium plates, and Spectra fabric fronted by B_4C plates.

Chapter 1

Introduction

1.1 Current Status of Personnel Armor Design

The history of personnel armor has been marked by advances in armor design counteracted by advances in armaments, requiring further development in armor systems to meet the new threats. The effectiveness of fabric armor for personnel protection against fragments and lower velocity projectiles increased significantly with the development of aramid fibers, due to their remarkable combination of high modulus and high failure strain. Since the incorporation of these fibers into body armor and helmets, several new projectiles have appeared which must be countered by advances in the design of the fabric armor system.

Among the newer armament threats which have been developed in recent years are projectiles which have a low friction coefficient coating such as teflon, and high aspect ratio projectiles such as flechettes which are able to defeat the fabric armor by passing through the interstices of the fabric without losing a significant amount of kinetic energy.

Techniques were developed during the 1970's to model the response of fabrics to ballistic impact and to clarify the details of the projectile-fabric interaction. These analytical techniques succeeded in large part in accounting for the increased armor protection afforded by aramid fabrics. These same techniques have been utilized and extended in this study in an effort to upgrade the ballistic resistance of fabric armor to the newer threats.

1.2 Stress Wave Propagation in Fibers.

Much of the numerical modeling to be described in this report is based on the closed-form analysis of wave propagation in fibers which is available in the literature (e.g. Roylance, 1977).[1] Salient points from these earlier analyses are included here to establish a context for the computer model.

Wave propagation phenomena in the fibrous elements of the fabric are considerably less complicated than in a general medium, since the possibility of unrestrained

[1] Literature citations are listed alphabetically in Appendix A on page 28.

transverse contraction in fibers eliminates to a good approximation the simultaneous propagation of independent dilatational and distortional waves which are present in general. Previous work has shown that longitudinal tensile stress wave propagation in fabric armor can be predicted accurately using these simplifications.

The equation of motion for fibers or rods is simply:

$$\frac{\partial^2 u}{\partial t^2} = \frac{E}{\rho}\left(\frac{\partial^2 u}{\partial x^2}\right) \tag{1.1}$$

where u is the longitudinal particle displacement, ρ is the material density, E is the longitudinal Young's modulus and x and t are the space and time coordinates. This is the well known wave equation, whose solution represents a disturbance traveling at a velocity

$$c = \sqrt{E/\rho} \tag{1.2}$$

Conventional textile units employing stiffness per unit linear density are very convenient in wave propagation analyses, since the factor is included implicitly in the modulus. For modulus expressed in grams per denier and wavespeed in meters per second, the equation for wavespeed becomes:

$$c = \sqrt{kE} \tag{1.3}$$

where $k = 88,260$ is the necessary units-conversion factor. The appropriate value of modulus in these equations is the "dynamic" stiffness relevant to the high strain rates corresponding to wave-propagation tests.

The relations above provide a simple means of estimating the stress which will be generated upon impact. Consider a fiber fixed at one end whose free end is suddenly subjected to a constant outward velocity V in the longitudinal (fiber) direction. After a time t, the strain wave will have propagated into the fiber a distance ct, while the free end will have displaced outward an amount Vt. The strain resulting from the impact of then the displacement Vt divided by the effected length ct:

$$\epsilon = \frac{Vt}{ct} = \frac{V}{c} = V\sqrt{\rho/E} \tag{1.4}$$

The corresponding stress is

$$\sigma = E\epsilon = V\sqrt{\rho E} \tag{1.5}$$

These equations assume a linear elastic material whose stiffness E is independent of the strain. In this case the wavefront will propagate as a sharp discontinuity (a shock wave) at which the strain rises instantaneously from zero to the value given by the strain equation. The effect of fiber nonlinearity gives rise to complications in the above equations, causing among other effects a rounding of the shock front. Although this treatment represents a simplification, it remains a good general description of the nature of the propagation of longitudinal stress waves in the fibers of a textile material.

Additional features must be added to the one-dimensional wave theory when the fiber is impacted transversely, as indicated in *Fig. 1.1*. In additional to an outward-propagating tensile wave which acts as for longitudinal impact, there is a transverse or

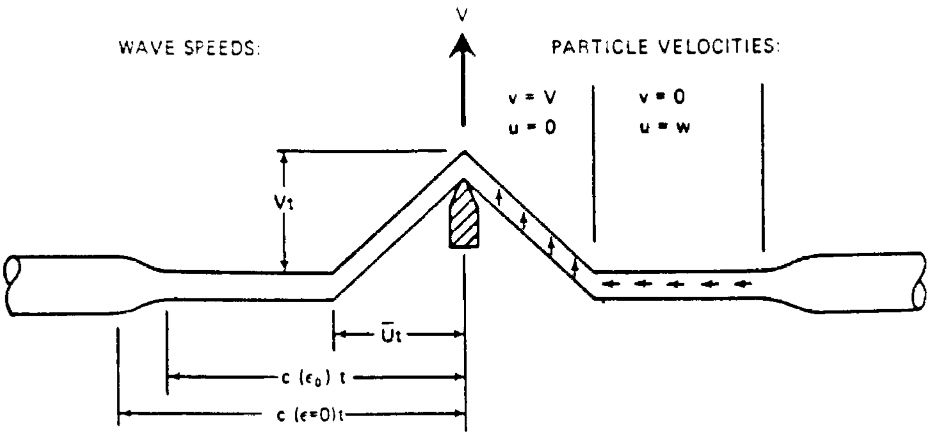

Figure 1.1: Schematic of wave propagation in transversely impacted fiber.

kink wave which is also propagated outward from the impact point. At the transverse wavefront the inward flow of material due to the longitudinal wave ceases abruptly and is replaced by a transverse particle velocity equal in magnitude and direction to that of the projectile. The strain and tension are unchanged across the transverse wavefront, but both the longitudinal and transverse particle velocities experience discontinuities there; in this sense the transverse wave is a geometrical shock. The apparently unbalanced tensions on either side of the transverse wavefront are compensated by the change in particle momentum as the wave propagates. Behind the transverse wavefront all particles velocities are equal in magnitude and direction to the projectile velocity, and the fiber configuration is a straight line at a constant inclination from the longitudinal direction.

The mechanics of transverse impact of single fibers, at times sufficiently short to avoid interactions of reflected wave components, have been elaborated by in a number of papers by Smith (e.g. Smith *et al.*, 1955) and Roylance (e.g. Roylance, 1977). Relations among the principal problem variables can be listed as follows:

- Longitudinal wave speed:

$$c(\epsilon) = \sqrt{E(\epsilon)/\rho} = \sqrt{kE(\epsilon), \mathrm{gpd}}$$

- Particle velocity:

$$w = \int_0^{\epsilon_0} c(\epsilon) d\epsilon$$

- Transverse wave speed:

$$U = \sqrt{T_0 k/(1+\epsilon_0)}$$

$$\overline{U} = (1+\epsilon_0)U - w$$

- Relation to impact velocity:

$$V = \sqrt{(1+\epsilon_0)^2 - \overline{U}^2}$$

- Transverse wave angle:

$$\theta = \sin^{-1} \frac{1}{1+\epsilon_0} \frac{V}{U}$$

These relations are useful in assessing the role of fiber modulus on wave propagation parameters. In the case of linear materials (E = constant), they can be combined to give:

$$V = \sqrt{\epsilon_0 k E \left[2\sqrt{\epsilon_0(1+\epsilon_0} - \epsilon_0 \right]} \quad (1.6)$$

which provides a relation for the strain ϵ_0 developed by impact at a velocity V in terms of the fiber modulus. The relation can be solved numerically if one withes to compute ϵ_0 for a given V, or it can be used directly to plot ϵ_0 versus V for the purpose of developing design curves.

These closed-form relations show (Roylance, 1977) that the rate at which the fiber absorbs energy from the projectile increases monotonically with modulus, so from this point of view one seeks the highest possible modulus. However, increases in stiffness are usually accompanied by decreases in breaking strain, and a point may be reached where this reduced ductility overshadows the beneficial reduction in impact-generated strain. Much of the numerical modeling to be discussed in this report is aimed at determining this balance between energy absorption rate and toughness.

Chapter 2

The Numerical Model

2.1 The "Direct Analysis" Method

Given the importance of textile armor in military and police applications, it is not surprising that a number of techniques have been developed for the purpose of simulating analytically the ballistic impact of these systems. In the 1960's and 1970's, Prof. Norman Davids and his collaborators at Pennsylvania State University (e.g. Mehta & Davids, 1966) developed an approach to transient field problems which was termed "direct" analysis. This method consists of taking incremental statements of the governing equations directly and coding them for computer execution, as opposed to carrying the incremental statements to a limit so as to develop differential equations.

The direct approach was used by Lynch (1970) to analyze transverse impact of single fibers and later extended by Roylance (1973) to the study of viscoelastic fiber impact and by Roylance, Wilde, and Tocci (1973) for impact on woven textile panels. This method appears to be well suited for flexible armor analysis and design: it is relatively simple to understand and implement, yet seems to capture the important physics of the impact event with good accuracy. Without claiming that direct analysis is the *only* effective approach to the fabric impact problem, this report will describe the extension of this method to an armor design for flechette-type projectiles.

2.2 Basic Numerical Algorithm

Referring to *Fig. 2.1*, the woven panel is idealized as an assemblage of pin-jointed, flexible fiber elements, each having a mass which makes the areal density of the idealized mesh equal to that of the panel being simulated. (The density of crossovers in the numerical model need not necessarily be the same as that in the actual panel.) The initial projectile velocity is imposed on the node at the impact point, which causes a strain to develop in the adjacent elements. The tension resulting from this strain is computed from a material "constitutive" (stress-strain) relation, and this tension is used to calculate an acceleration in the neighboring elements. The computer proceeds outward from the impact point, successively using a momentum-impulse balance, a strain-displacement condition, and a constitutive equation to compute for each element

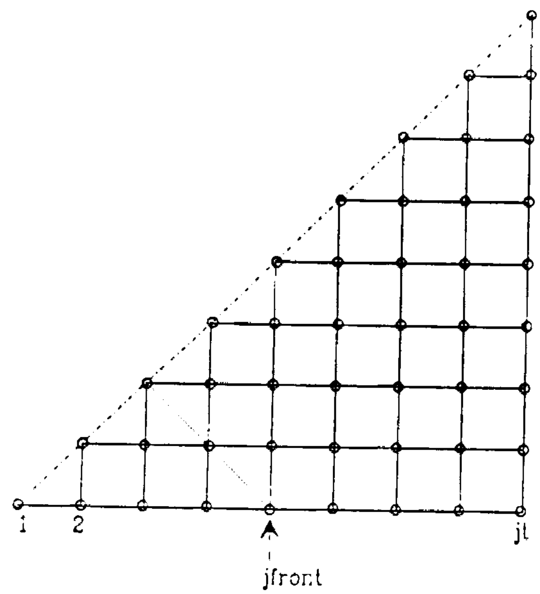

Figure 2.1: Idealization of fabric.

the current values of tension, strain, velocity, position, and such ancillary but important quantities as strain energy and kinetic energy. At the end of these calculations, a new projectile velocity is computed from the tensions exerted on the projectile by the fibers, and the process is repeated for a new increment of time.

At each time step, the computer code "marches" from node to node, beginning at the impact point and working outward. At each nodal position, it determines the values of the problem variables at the next node outward in terms of those at the present node. The algorithms for this are written in terms of using values at node (j, k) to determine variables at node $(j + 1, k + 1)$. The indices j and k are stepped from 1 (at the impact point) to a value corresponding to the edge of the plate; this latter is an input parameter named `jt` in the code. The indices are incremented so as to follow diagonal paths at successive distances from the impact point, as indicated by the dotted lines in *Fig. 2.1*.

The beginning point for these "space loop" calculations is on the horizontal fiber running through the impact point, and is denoted in the code by the Fortran variable `jfront`. This starting node is initially at $j = 2, k = 1$, and at each successive time increment is moved one node to the right; the `jfront` position noted in *Fig. 2.1* is the fourth starting point for the space loop. After `jstart` reaches the edge of the panel, it is then moved upward along the vertical fiber at the panel's edge.

The masses of the fiber elements are taken to be lumped at the crossover "nodal" points, so that each node incorporates half the mass of the four fiber elements meeting at that node:

$$m_n = \frac{1}{2} \cdot 4\varsigma\rho_l\Delta l \tag{2.1}$$

The parameter ρ_l is the lineal fiber density, read into the Fortran variable denyrn in textile units of denier (mass in grams of a 9,000 m length of yarn); this is usually converted to gm/cm by dividing by 9×10^5. Δl is the length of the fiber element connecting the nodes, and ς is a numerical factor associated with the crimping and weave of the fabric structure. The crimping factor ς (named `crimp` in the computer code) is in turn computed by dividing the actual mass of the experimental fabric panel by that which would have been obtained simply by multiplying the lengths of the various fiber elements by their density. This crimping factor provides a means of simulating multiple-layer fabrics; the user provides a value for the actual multiple-layer panel mass (**fmassa** in the code), and the appropriate mass adjustment is incorporated into the crimping factor. The crimping factor allows the use of numerical meshes less fine than the actual weave, since it serves to make the areal densities of the actual and simulated fabrics the same.

These considerations have been programmed in Fortran as follows:

```
denyrn = denyrn*clyr
fmassa = fmassa*clyr
jtm1=jt-1
fmassm = 4.d0*float(jtm1)*(2.d0*denyrn*xl)/9.d5
crimp  = fmassa/fmassm
unitm  = crimp*2.d0*denyrn*dxl
```

In the above, an effective yarn density (denyrn, in gm/den), and fabric mass (**fmassa**, in gm) is obtained by multiplying the single-yarn denier and the single-layer fabric mass by the number of layers `clyr`. A "model" fabric mass, named **fmassm**, is computed by multiplying the yarn denier by the fabric half-length (xl, in cm), dividing by 9×10^5 to obtain grams mass, multiplying by two to obtain the mass of a fiber running the full width of the fabric, times the number of horizontal fibers in the entire square panel ($2 \times [\texttt{jt} - 1]$), and finally times two again to account for the vertical fibers. The model fabric mass is then divided into **fmassa** to obtain the crimp factor. The effective nodal mass m_n, named `unitm` in the code, is computed as in *Eqn. 2.1*.

The tensions in the fiber elements meeting at the node cause an acceleration of the nodal mass, and this effect is described by a straightforward application of Newton's Second Law:

$$\mathbf{T} = m_n \frac{\Delta \mathbf{v}}{\Delta t} \tag{2.2}$$

Here $\Delta \mathbf{v}$ is the change in velocity in a time increment Δt, where boldface type is used to indicate vector quantities. This relation provides a means of calculating current velocity from field variables in the previous time increment. For instance, at node $(j+1, k+1)$, the velocity at time t_{m+1} may be expressed in a finite difference form as

$$\mathbf{v}^{m+1}_{j+1,k+1} = \mathbf{v}^{m}_{j+1,k+1} + \mathbf{T}^{m}_{j+1,k+1} \frac{\Delta t}{2\varsigma \rho_l \Delta l} \tag{2.3}$$

In this expression subscripts are used to indicate nodal positions, and superscripts denote the time increment.

The units in this expression warrant some discussion, as the code is written to use common but mixed units: cm for position and displacement, m/s for velocity, μs for time increment, denier for lineal density, gmf/den for fiber tension and stiffness. A units equation for the momentum balance might be written as:

$$\Delta \mathbf{v}\left(\frac{\mathrm{m}}{\mathrm{s}}\right) = \frac{\mathbf{T}\left(\frac{\mathrm{gmf}}{\mathrm{den}}\right)\cdot\rho_l(\mathrm{den})\cdot 980.7\left(\frac{\mathrm{dyne}}{\mathrm{gmf}}\right)\cdot\Delta t(\mu s)\cdot 10^{-6}\left(\frac{\mathrm{s}}{\mu\mathrm{s}}\right)}{2\varsigma\cdot\frac{\rho_l}{9\times10^5}\left(\frac{\mathrm{gm}}{\mathrm{cm}}\right)\cdot\Delta l(\mathrm{cm})}\cdot 10^{-2}\frac{\mathrm{m}}{\mathrm{cm}}$$

$$= \frac{8.826\mathbf{T}\Delta t}{2\varsigma\Delta l} \qquad (2.4)$$

To obtain the final expression above, dyne-sec/gm are noted to be equivalent to cm/sec (since 1 dyne = 1 gm cm/s^2). Note also that the yarn denier ρ_l cancels from the expression, which is one reason for using lineal density in the formulation.

The tensile force $\mathbf{T}_{j+1,k+1}$ has five contributing components, one from each of the four fiber elements meeting at the node, and one more – potentially – from any "backup" material lying behind the impacted fabric. These components can be denoted as:

$$\mathbf{T}^m_{j+1,k+1} = \mathbf{T10}^m_{j+1,k+1} - \mathbf{T10}^m_{j,k+1} + \mathbf{T01}^m_{j+1,k+1} - \mathbf{T01}^m_{j+1,k} - k_b\mathbf{u}^m_{j+1,k+1} \qquad (2.5)$$

in which k_b is a backup spring constant and **u** is the displacement vector. **T10** and **T01** are the tensions in fiber elements oriented originally along the x and y axes. Of course, as the fabric deforms during impact, these originally orthogonal fibers will deflect, so that **T10** and **T10** will develop force components in all three coordinate directions. The tensions are those "outboard" of the node; for instance a tension $\mathbf{T10}_{j,k}$ is the (vector) tension in the element running outward from the point of impact, connecting node j,k and $j+1,k$. The backup spring can serve to simulate the influence of an actual backup layer in the armor, or the effect of the human torso. It might also serve to stiffen the fabric so as to simulate the role of an impregnating resin, such as the butyral resin used in current military helmets.

The Fortran statements for these velocity calculations compute each velocity component separately, as follows:

```
      dvx=(tx10(j+1,k+1)-tx10(j,k+1)+tx01(j+1,k+1)-tx01(j+1,k))
     &        *dtm*8.82d0/(2.d0*dxl*crimp)
      dvy=(ty10(j+1,k+1)-ty10(j,k+1)+ty01(j+1,k+1)-ty01(j+1,k))
     &        *dtm*8.82d0/(2.d0*dxl*crimp)
      dvz=(tz10(j+1,k+1)-tz10(j,k+1)+tz01(j+1,k+1)-tz01(j+1,k))
     &        *dtm*8.82d0/(2.d0*dxl*crimp)
     &        +xk*dxl*zcd(j+1,k+1)*dtm*9.8d-6/(unitm/9.d5)
```

The coordinates representing nodal positions can be updated at each time increment by using current values of the nodal velocities; this relation can be written as:

$$\mathbf{x}^{m+1}_{j+1,k+1} = \mathbf{x}^m_{j+1,k+1} + \mathbf{v}^{m+1}_{j+1,k+1}\Delta t \qquad (2.6)$$

The corresponding Fortran statements are:

```
xcd(j+1,k)=xcd(j+1,k)+vx(j+1,k)*dtm/1.d4
ycd(j+1,k)=ycd(j+1,k)+vy(j+1,k)*dtm/1.d4
zcd(j+1,k)=zcd(j+1,k)+vz(j+1,k)*dtm/1.d4

xcd(j,k+1)=xcd(j,k+1)+vx(j,k+1)*dtm/1.d4
ycd(j,k+1)=ycd(j,k+1)+vy(j,k+1)*dtm/1.d4
zcd(j,k+1)=zcd(j,k+1)+vz(j,k+1)*dtm/1.d4
```

The `1.d4` factor compensates for the mixed units: coordinates are in cm, velocities in m/s, and the time increment `dtm` is μs.

The current strain ϵ can then be computed from the updated displacements. For instance the strain $\epsilon 01^{m+1}_{j+1,k}$ of the element extending in the y-direction from node $(j+1,k)$ is:

$$\epsilon 01^{m+1}_{j+1,k} = \epsilon 01^{m}_{j+1,k} + \left(\frac{\mathbf{x}^{m+1}_{j+1,k+1} - \mathbf{x}^{m+1}_{j+1,k}}{\mathbf{x}^{m}_{j+1,k+1} - \mathbf{x}^{m}_{j+1,k}} - 1 \right) \tag{2.7}$$

This produces an "incremental" or logarithmic strain measure, in which each increment of strain is based on the current length of the fiber element as opposed to its original unstrained length. The Fortran computations are done in a series of steps:

```
 dsq01=(xcd(j+1,k+1)-xcd(j+1,k))**2
&      +(ycd(j+1,k+1)-ycd(j+1,k))**2
&      +(zcd(j+1,k+1)-zcd(j+1,k))**2

 dist01=dsqrt(dsq01)

 dsqx01=(xcd(j+1,k+1)+vx(j+1,k+1)*dtm/1.d4
&         -xcd(j+1,k  )-vx(j+1,k  )*dtm/1.d4)**2
 dsqy01=(ycd(j+1,k+1)+vy(j+1,k+1)*dtm/1.d4
&         -ycd(j+1,k  )-vy(j+1,k  )*dtm/1.d4)**2
 dsqz01=(zcd(j+1,k+1)+vz(j+1,k+1)*dtm/1.d4
&         -zcd(j+1,k  )-vz(j+1,k  )*dtm/1.d4)**2

 ddst01=dsqrt(dsqx01+dsqy01+dsqz01)
 deps=ddst01/dist01-1.d0
 eps01(j+1,k)=eps01(j+1,k)+deps
```

The fiber tension T can be related to the strain ϵ by the material's dynamic constitutive relationship, and the code has a number of material models available. For the case of a simple elastic fiber of modulus E, stress and strain may be calculated by the simple form of Hooke's law:

$$T = E\epsilon \tag{2.8}$$

Here the modulus E is in units of gm/den. In addition to linear elastic materials, the code is able to simulate power-law nonlinear elasticity:

$$T = k\epsilon^{n} \tag{2.9}$$

or a polynomial nonlinear elasticity:

$$T = E_0 + E_1\epsilon + E_2\epsilon^2 + E_3\epsilon^3 \tag{2.10}$$

or "Standard Linear Solid" viscoelasticity:

$$T^t = g(\epsilon^t - \epsilon^{t-1}) + \frac{g(1-\lambda)}{\tau}\epsilon^t \Delta t + T^{t-1} \tag{2.11}$$

Here g is the instantaneous or "glassy" modulus, τ is the characteristic time for the relaxation, and λ is the ratio of the magnitude of the relaxation to the equilibrium modulus (Roylance, 1973).

The linear viscoelastic model has also been extended (Roylance & Wang, 1978) to include nonlinear viscoelasticity by using a power-law spring and an Eyring dashpot, defined as

$$T_s = k\epsilon_s^n, \qquad \dot{\epsilon}_d = A\sinh(\alpha T_d) \tag{2.12}$$

The finite-difference equation relating tension to strain in the j^{th} arm of the model is

$$\frac{\epsilon^t - \epsilon^{t-1}}{\Delta t} = \frac{1}{n_j k_j}\left(\frac{T_j^t}{k_j}\right)^{\frac{1}{n_j}-1}\left(\frac{T_j^t - T_j^{t-1}}{\Delta t}\right) + A_j \sinh(\alpha_j T_j^t). \tag{2.13}$$

This last relation requires an iterative numerical solution for each T_j^t at each element and at each time step; the computer effort is increased but the principles of the impact algorithm are unchanged. The principal obstacle to the use of nonlinear models in impact analysis is not the incorporation of the models into the solution scheme, but deciding which of many possible approaches is reasonable. Of course, it is also difficult to determine the many material parameters applicable in the microsecond time scale of polymer relaxations.

The subroutine `tensn.for` which computes the element tension from the strain also computes the increase in fabric strain energy. The incremental increase in strain energy per unit volume is $\sigma\Delta\epsilon$, and the code uses the average stress during the time increment:

$$\Delta U = \left(\frac{\sigma^t + \sigma^{t-1}}{2}\right)\left(\epsilon^t - \epsilon^{t-1}\right)$$

The units of this expression are in gmf/den; it is converted to units of Joules per gram of fabric by multiplying by the factors:

$$\rho_l(\text{den}) \cdot 980.7\frac{\text{dyne}}{\text{gmf}} \cdot 10^{-5}\frac{\text{N}}{\text{dyne}} \cdot \Delta l(\text{cm}) \cdot 10^{-2}\frac{\text{m}}{\text{cm}}$$

The corresponding Fortran statement, which includes a correction for crimping and a normalization by the fabric mass `fmassa`, is:

```
      dsen=(4.d0 * 0.5d0 * (ts+tslold) * (epss-epsold)
     &    * dxl * denyrn * crimp * 9.8d-5 * symctr) / fmassa
```

The factor 4.0 accounts for the energy deposited symmetrically in all four quadrants of the fabric.

After indexing the node counters from the impact point to the upper-right node in the half-quadrant, the projectile velocity V_p is updated and the loop over the fabric nodes is begun again. The projectile velocity is corrected by means of a momentum balance between the projectile deceleration and the tensions acting on the projectile by the four fiber elements connected to the impact point, and also the decelerating force provided by the backup medium:

$$V_p^{m+1} = V_p^m + \left[4(T_z)10_{0,0}^{m+1} \cdot \sin\theta_p + k_b(u_z)_{0,0}^{m+1}\right]\frac{\Delta t}{m_p}, \tag{2.14}$$

where m_p is the projectile mass and θ_p is the fabric inclination at the impact point, which may be evaluated from the nodal coordinates as:

$$\theta_p = \tan^{-1}\left[\frac{(x_z)_{0,0}^{m+1} - (x_z)_{1,0}^{m+1}}{\Delta l}\right]. \tag{2.15}$$

Here x_z is the z-component of the node's position vector; i.e. its deflection along the direction parallel to the projectile velocity.

The length increment Δl is user-specified, by selecting the number of nodes jt spanning the distance from the impact point to the edge of the fabric. However, the time increment Δt cannot be specified arbitrarily, since the ratio of the length and time increments must satisfy a numerical stability criterion:

$$\frac{\Delta l}{\Delta t} < c_f, \tag{2.16}$$

where c_f is the wavespeed in the fabric. This wavespeed is less than the well-known value $c = \sqrt{E/\rho}$ expected for single fibers, since the crossovers act effectively to double the mass density sensed by the propagating wave. This line of reasoning, which is admittedly conjectural and in need of further study, leads to $c_f = \sqrt{2}c$. To permit alternative relations, the time increment is computed by the code as:

$$\Delta t = \frac{\Delta l}{c_f} = \frac{\Delta l}{\alpha\sqrt{E/\rho}}. \tag{2.17}$$

where usually we take $\alpha = \sqrt{2}$; α is named ctdm in the code.

The numerical method requires appropriate initial and boundary conditions in order to proceed with the computation. The initial condition is that all nodal points are at rest except that the initial projectile velocity is imposed at the center of the panel, i.e. $\mathbf{v}_{0,0}^0 = V_p$. The boundaries of the fabric are assumed to be rigidly clamped during impact, thus $\mathbf{v}_{z,L/2}^t = 0$.

2.3 Extension to Covering-Plate Designs

As presently developed, fabric-based flexible personnel armor is able to provide good protection against relatively blunt projectile impacting at low to moderate velocities, such as are developed by common fragmentation threats. However, fabrics are notably less effective against slender needle-like projectiles, such as the "flechettes" being incorporated in a variety of "improved conventional munitions" (ICM's). These projectiles are able to insinuate themselves through the fabric by passing between fibers, so the ability of the fibers to absorb strain energy is not brought to bear in defeating the projectile.

One natural means of defeating such a threat might be to provide an overlapping arrangement of small plates covering the outward-facing side of the fabric armor; these would act either to defeat the projectile themselves, or at least to deflect the projectile away from a zero-obliquity impact. The direct-analysis fabric code was therefore extended to make a new version, named PLATE, which incorporated this feature. While keeping the basic nature of the direct analysis algorithm – a space loop embedded within a time loop – several features must be considered in adding a covering plate to the model:

- A decision must be made concerning the nature of the projectile-plate interaction. This might range from simply assuming the plate to impact on the fabric at a velocity equal to that of the projectile, to a full and detailed finite-difference simulation of plate perforation.

- Whereas in the FABRIC code only a single nodal point remained in contact with the projectile, in PLATE the velocity of the covering plate is imposed on all those nodes lying beneath it.

- At the end of the space loop over the fabric nodes, the projectile is decelerated by the forces exerted on it by the fibers and the backup support. When the plate is added, decelerating forces are imposed on it by all of the fibers connected to its edges.

In a first model for covering-plate simulation, the FABRIC code was updated simply by replacing the point-impact scenario by one in which the plate and projectile were assumed to impact the fabric together, with a velocity equal to that of the initial projectile velocity. This provided some indication as to plate effects, but it is clearly unrealistic to assume the plate can accelerate instantaneously and without any slowdown of the projectile.

A number of trial developments were then investigated which considered more realistic mechanics of the plate-projectile interaction. One approach would be the use of one of several "hydro" impact codes now avaliable (e.g. HEMP, PUFF, DYNA, EPIC, etc.) These finite-difference or finite-element simulations could provide a very high level of detail as to the impact dynamics of the projectile and plate, not only near the impact point but also the back-face displacements which could then be imposed on the underlying fabric. However, these codes are substantially larger and more complex than our fabric code, and in keeping with our goal of simplicity and economy we have sought less elaborate approaches.

As an intermediate approach, we developed several code versions which estimated the dynamic stress state at the impact point between the projectile and the plate, and then used the corresponding force to compute an acceleration of the plate. This acceleration was in turn used to start the iterative fabric calculations.

One such estimate was provided by the ARAP model (Aeronautical Research Associates of Princeton, 1977), which uses hydrodynamic reasoning to estimate the impact-point force to be of the form:

$$F = \rho l^2 \left(\frac{C_D}{2} V^2 + E_p \right) = A_1 V^2 + A_2 \tag{2.18}$$

where ARAP has estimated values for the constants A_1 and A_2 from impact data. While this model has several attractive features, its use in beginning FABRIC calculations is problematic due to the need to estimate the region over which the impact stress occurs. When the impact force as calculated from *Eqn. 2.18* is used to initiate the code, an overly rapid deceleration of the projectile occurs. This is because the computed force does not act on the entire projectile, but is confined to a small region of influence near the contact point. This "region of influence" is itself an important dynamic variable, but to compute it exactly would appear to require an elaborate penetration code.

A similar problem arises even in using very simple estimates of the impact stress, such as the expression $\sigma = V\sqrt{\rho E}$ given by one-dimensional wave theory *(Eqn. 1.5)*. The stress is easy to compute, but its region of influence is a variable which increases as waves propagate outward from the point of impact. Use of the stress from this equation directly in an expression for plate acceleration, assuming the stress acts on the entire projectile, produces unreasonably large values. It appears that any stress-wave oriented approach will suffer from this difficulty, and it would be necessary to include an overly complex "front-end" code in order to compute the forces transmitted by the plate to the fabric. In keeping with our stated intention to keep the computer model as simple as possible, other approaches are indicated.

After several trials, we have selected a simple approach in which the initial plate velocity is computed by partitioning the initial momentum in the impacting projectile such that the plate and projectile move together at a common velocity after impact. This is a "totally inelastic" collision, in which the kinetic energy is *not* conserved and the velocity is that which conserves momentum, as given by:

$$V^{t=0} = V_{impact} \left(\frac{m_{prj}}{m_{prj} + m_{plt}} \right), \tag{2.19}$$

where $V^{t=0}$ is the initial velocity of the projectile-plate combination which is to be imposed on the fabric, V_{impact} is the impact velocity of the projectile, and m_{prj} and m_{plt} are the masses (in gm) of the projectile and plate respectively. This computation is done in the following Fortran expressions:

```
totmass=prjmass+pltmass
vplate = vproj*prjmass/totmass
vproj=vplate
```

At present, the failure time for the fabric-panel combination is computed by monitoring the strain in each fiber element, and predicting panel failure when any element exceeds the specified fiber breaking strain. More elaborate failure models are possible, of course. One elaboration which should be considered for future work is that of predicting the breakup of the covering plate. This might be done by using ARAP or other plate-penetration models. Once the plate is predicted to have failed, the PLATE code might then reassign jplate to be unity and set the plate mass to zero. The fabric could then continue to respond as in the point-impact case to whatever velocity remains of the impoacting projectile.

The computation of the deceleration of the projectile-plate combination is done at the end of each space loop, just as before. However, the plate will now be influenced by tensions in fiber elements all along the plate edges, as well as by backup forces from all nodes lying underneath the plate. These computations are performed by the following statements:

```
c       compute plate/projectile slowdown

49      zforce=0.d0
        do 50 k=1,jplate
           aproj=atan((zcd(jplate+1,k)-zcd(jplate,k))
     &              /(xcd(jplate+1,k)-xcd(jplate,k)))
           zforce=zforce + t10(jplate,k)*sin(aproj)
50      continue
        bforce=xk*zcd(1,1) * ((2*jplate-1)**2)
           vplate=vplate+8.d0*zforce*dtm*denyrn*9.80e-06*crimp/totmass
     &             -bforce*dtm*9.80e-06/totmass
           vproj=vplate
```

Chapter 3

Numerical Design Studies

3.1 Code Predictions

Many performance aspects of the basic numerical algorithm have been discussed in previous reports and papers. However, certain features will be elaborated here, both to place the further numerical developments in context and to elaborate points which warrant further discussion.

3.1.1 Wave Propagation Plots.

As the code simulates the propagation of strain waves away from the point of impact, it generates a large quantity of data representing the problem variables at each node and at each increment of time. It is difficult to observe trends from printed numerical tabulations alone, and some means of graphical interpretation is very useful. Although the code is not written with an explicit graphical output capability, it does create a series of output datafiles which can be used as input to independent graphics routines. Most of these files are indicated with a `.csv` suffix, which denotes a "comma-separated value" format; these can be imported into a variety of graphics and spreadsheet programs.

Figure 3.1 displays a three-dimensional map of the z-displacements approximately $10\mu s$ after a 300 m/s impact on a simulated nylon fabric; this map was created from the file `zmap10.csv` using the "Golden Graphics" software package. A similar map, but depicting the strains in the horizontally-running fibers, is shown if *Fig. 3.2;* this was generated from the file `smap10.csv`. A visual comparision of these two figures shows that the strains are propagated outward from the impact point more rapidly than is the transverse displacement.

The files `splot.csv` and `zplot.csv` depict the strains and z-displacements, respectively, along the fiber passing through the impact point, at each increment of time after impact. *Figure 3.3* shows a plot obtained by processing `splot.csv` with the "SuperCalc 4" spreadsheet program; here the strain at three nodal positions, successively further from the impact point, are plotted versus time after impact of a simulated nylon fabric at 300 m/s.

Several aspects of wave propagation in fabrics are evident in this plot: the strain

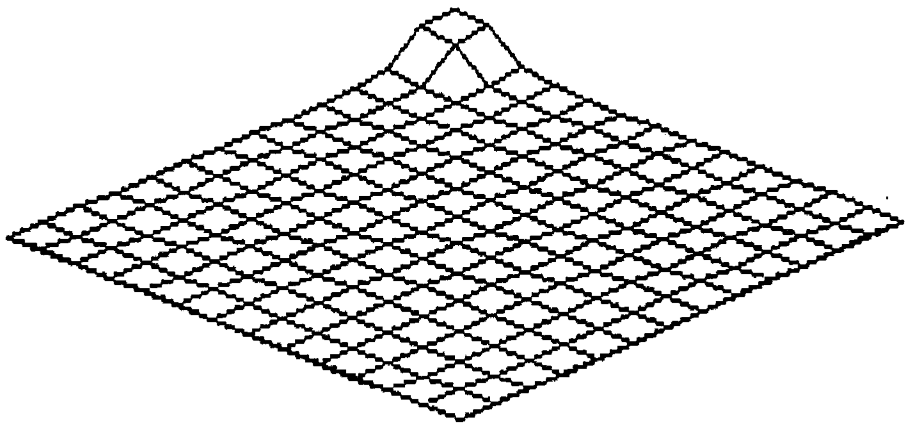

Figure 3.1: Map of z-displacements.

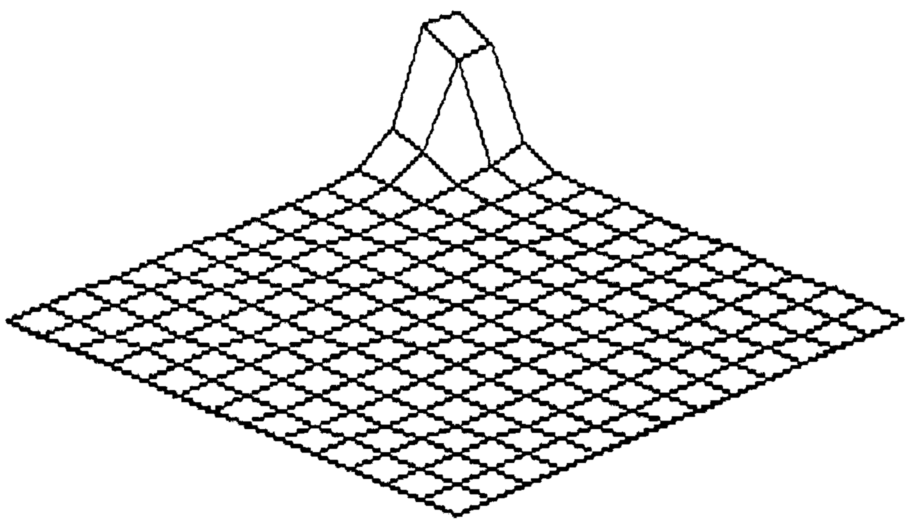

Figure 3.2: Map of strain in horizontal fibers.

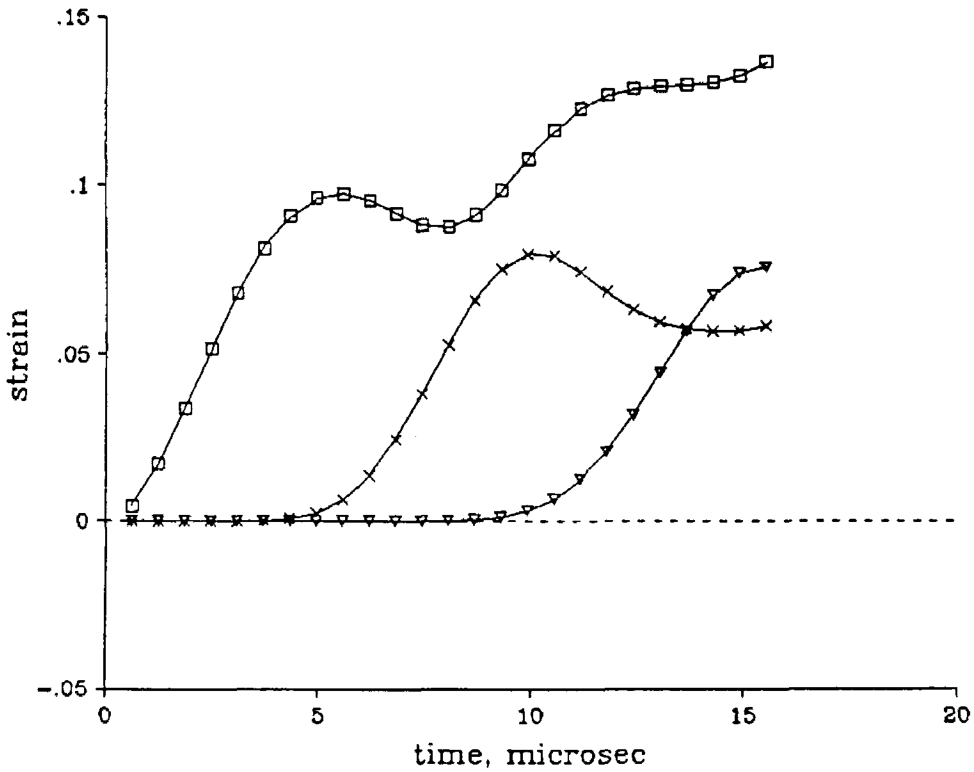

Figure 3.3: Development of strain in fiber passing through impact point.

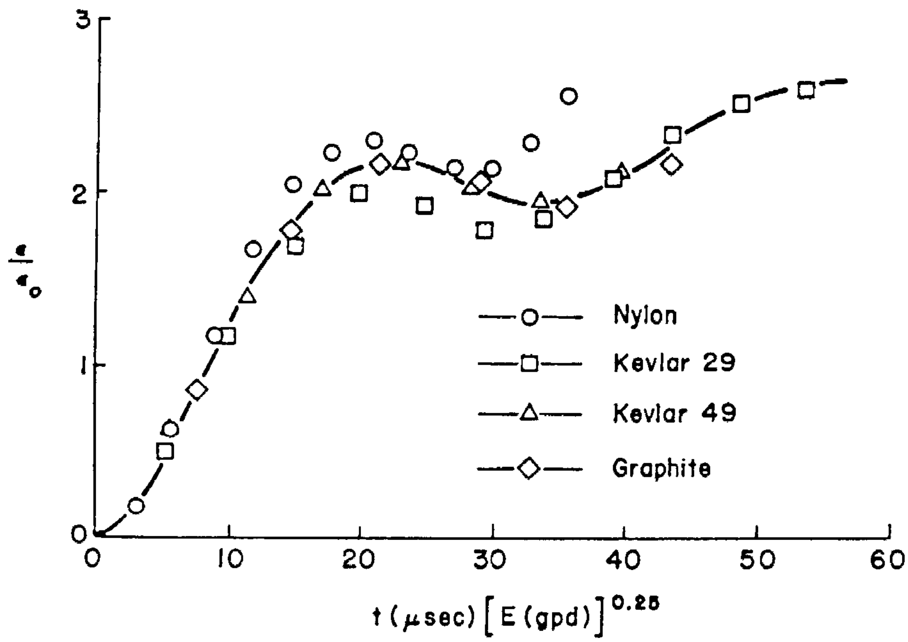

Figure 3.5: Master curve for strain at impact point.

3.1.2 Influence of Fiber Properties.

The principal influences of fiber properties on ballistic response of textile panels have been described in an earlier reports by Roylance and Wang (1979, 1981). As in the case of a single transversely impacted fiber described in the introduction of this report, two properties are especially important: the fiber modulus and the fiber breaking strain. As the modulus increases, so does the wavespeed according to $c = \sqrt{E/\rho}$, and the fabric will be able to propagate the ballistic impact energy away from the point of impact more rapidly. The influence of modulus is demonstrated in *Fig. 3.5,* in which the ordinate is strain at the impact point, normalized by the strain which would have been developed in a single transversely impacted fiber (ϵ_0 in *Eqn. 1.6*). The single-fiber strain is itself modulus-dependent; higher-modulus fibers develop higher strain when impacted at a given velocity. The abscissa is time after impact, normalized by the fourth root of the fiber modulus. The resulting plot is a "master-curve" depiction of strain history, made independent of fiber modulus by the normalization factors.

As breaking strain increases, longer times will be required for the strain at the impact point to rise to the breaking value, and the fabric will be able to survive a longer time after impact. While it survives, the outward-propagating strain waves will develop strain energy in a region of influence which increases in size as time proceeds, and this increasing fabric strain energy is extracted from the kinetic energy of the projectile. Clearly, higher breaking strains will improve ballistic performance, Unfortunately, increasing the fiber modulus – usually by drawing it to a greater extent after spinning – normally reduces its breaking strain. This forces the armor designer to compromise these two conflicting effects, and one principal use for such computer analyses as the

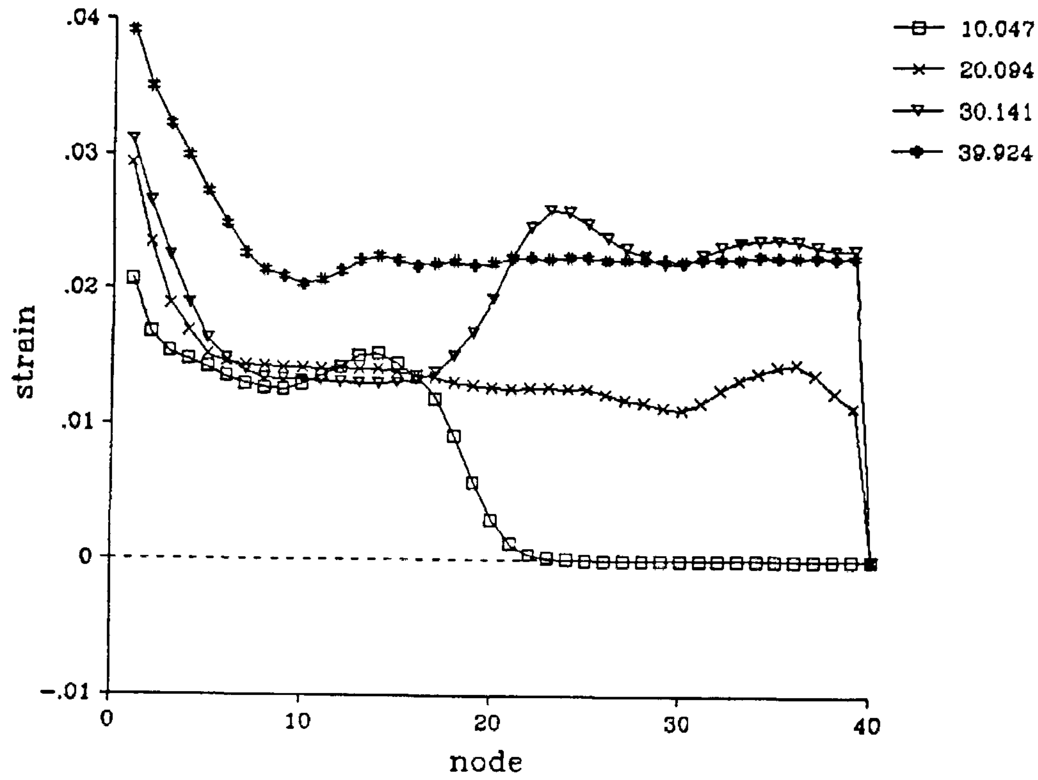

Figure 3.4: Strain distribution along fiber passing through impact point.

at the impact point (node # 1) generally rises with time, due to the arrival there of wavelets reflected from fiber crossovers. This is unlike the single-fiber event, in which the strain at the impact point remains constant (in the absence of appreciable viscoelastic relaxation) until the strain wave propagates along the fiber, reflects from the clamp, and travels back to the impact point.

It is also seen in *Fig. 3.3* that the strain history does not increase monotonically, but rather with a series of local maxima separated by a fairly regular period. It is tempting to ascribe this periodicity to some sort of constructive or destructive interference of wavelets travelling in the fabric. However, the same waviness is seen in simulations of single-fiber impact as well (Cunniff, 1989), and is probably an artifact of the numerical algorithm.

It is also useful to crossplot the data so as to examine the distribution of strain along the fiber passing through the impoact point at a sequence of times. *Figure 3.4* shows such a plot for a simulated Kevlar 29 fabric impacted at 300 m/s. At high impact velocities, fracture can occur at the impact point without any reflection of the main wavefront from the fabric clamp. But as can clearly be seen in *Fig. 3.4*, eventually the wave can reflect from the clamp and travel back toward the impaact point. It is also possible for the reflected strain to exceed the breaking strain, in which case the model will predict fiber rupture at the clamp.

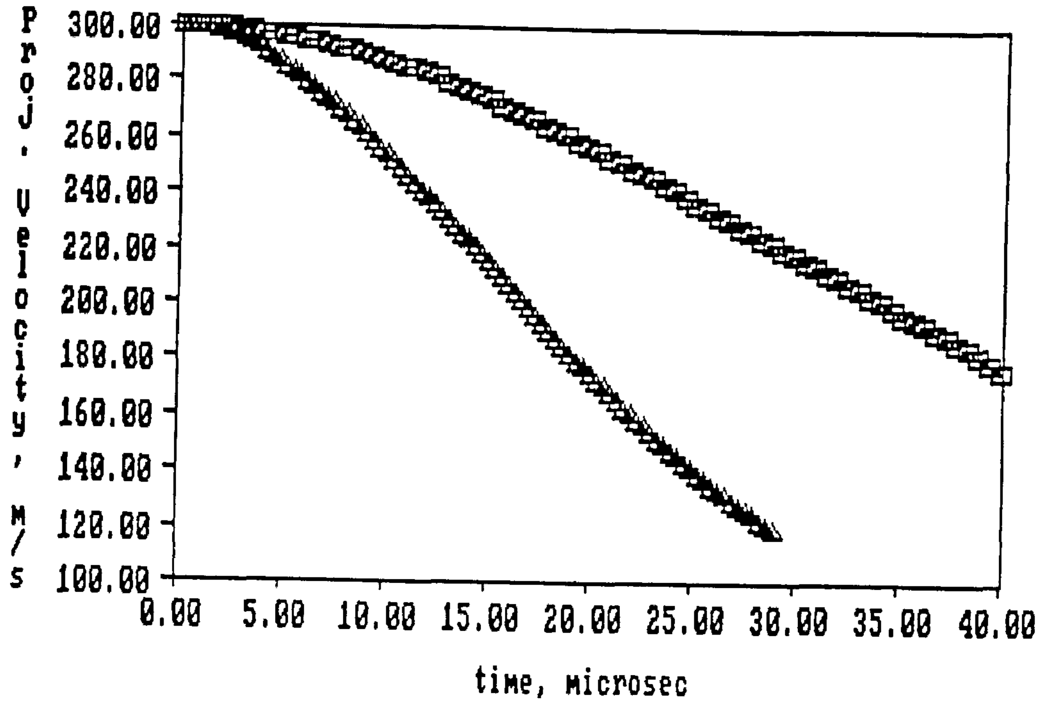

Figure 3.6: Projectile deceleration after 300 m/s impact.

one developed in this study is in determining the optimal balance.

The output file `plate.csv` contains values of current projectile velocity, as well as the fabric strain and kinetic energies, at each time increment after impact. Plots of projectile velocity versus time are useful in assessing armor effectiveness, and *Fig. 3.6* compares the relative ability of Kevlar 29 (denoted by the □ symbols) and Spectra 900 (△ symbols) in decelerating the projectile after a 300 m/s impact. The Kevlar survives longer, but its lower wavespeed and concomitantly lower rate of energy absorption leads to a higher residual projectile velocity after failure. *This plot indicates that Spectra would be expected to provide ballistic resistance superior to Kevlar.* (This conclusion is based only on wave propagation characteristics and breaking strain, without regard to cost, ease of fabrication, or other potentially important aspects of armor design.)

3.1.3 Influence of Node Spacing

As was mentioned earlier, it is not necessary to have as many crossover nodes in the numerical model as are actually present in the fabric, since the crimp factor acts to make the areal densities of the numerical and actual fabrics equal. Woven fabrics have on the order of 40 crossovers per inch, and using a numerical mesh density this fine would overwhelm the storage capacity of many small computers. However, code accuracy would be expected to improve as the number of nodes increases. To assess this aspect of code performance, a number of simulations were completed in which the projectile velocity was kept constant and only the number of nodes `jt` was varied. One

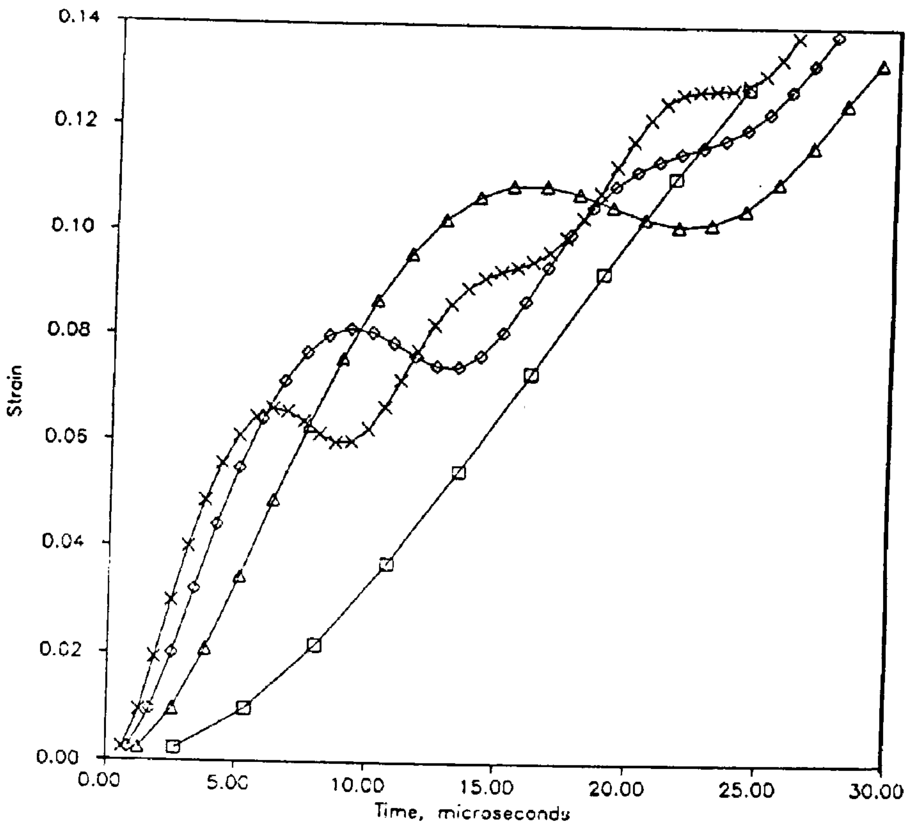

Figure 3.7: Influence of node spacing on impact-point strain histories.

useful measure of code response is to plot the strain history at the impact point, as is done in *Fig. 3.7.* Here the □ symbol denotes a simulation for jt = 10, while the symbols △, ◇, and × indicate jt = 20, 30, and 40, respectively.

Clearly, the node spacing has a substantial effect on the computed strain history. *Figure 3.8* provides one way of quantifying this effect, by plotting the time taken for the strain at the impact point to rise to a given value (4%) versus the node spacing. (This figure is for impact at constant 300 m/s velocity on a ballistic nylon fabric.) It is seen that the dependence on node spacing diminishes, with relatively little dependence after jt = 40. This is also the limit which can be accomplished with our present hardware, and the flattening out of the curve provides some assurance that further refinement beyond jt = 40 might not be necessary for preliminary design work.

3.1.4 Influence of Covering Plate

When a covering plate is impacted by a projectile, the highest strains are developed initially in the fiber elements around the edge of the plate. *Figure 3.9* is a portrayal of the strains developed in the "10" fibers (those running in the x-direction) approximately 10μs after impact on an idealized ballistic nylon panel covered with a 2 cm plate is impacted at a constant velocity (300 m/s). Another view is provided by *Fig. 3.10,*

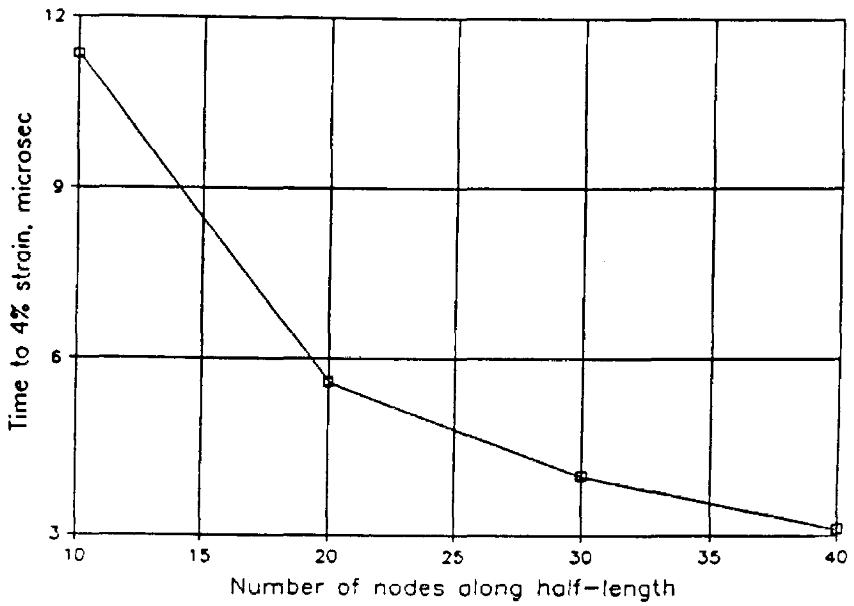

Figure 3.8: Influence of node spacing on time to reach 4% strain.

which plots the strain in the fiber running through the impact point at two different times after impact.

These strains are propagated both outward, away from the plate, and inward, underneath the plate. In time, the strain directly under the projectile impact point becomes the largest in the fabric, just as in the fabric-only case. This can be seen in *Fig. 3.11*, in which an idealized ballistic nylon panel covered with a 3 cm plate is impacted at a constant velocity (300 m/s). Here × indicates values at the node nearest the edge of plate, and □ represents the fabric under the center of the plate (at impact point).

3.2 Design Recommendations.

The principal design recommendation of this report, taken in response to the threat of slender flechette-type projectiles, is to provide an array of adjacent hard plates which prevent the projectile from simply passing between the fibers of the fabric weave. These plates could be held in place by a fabric weave, or pinned so as to make a chain-mail-like sheet which could be placed in a vest in front of the fabric layers. In the sections to follow, we use insight provided by the PLATE code to develop "first-cut" recommendations as to the various materials and geometric parameters which make up such a design.

3.2.1 Fabric Material Selection.

Any study aimed at furthering fabric armor would naturally begin with DuPont's Kevlar 29 as a baseline material; this material provided a breakthrough in armor effec-

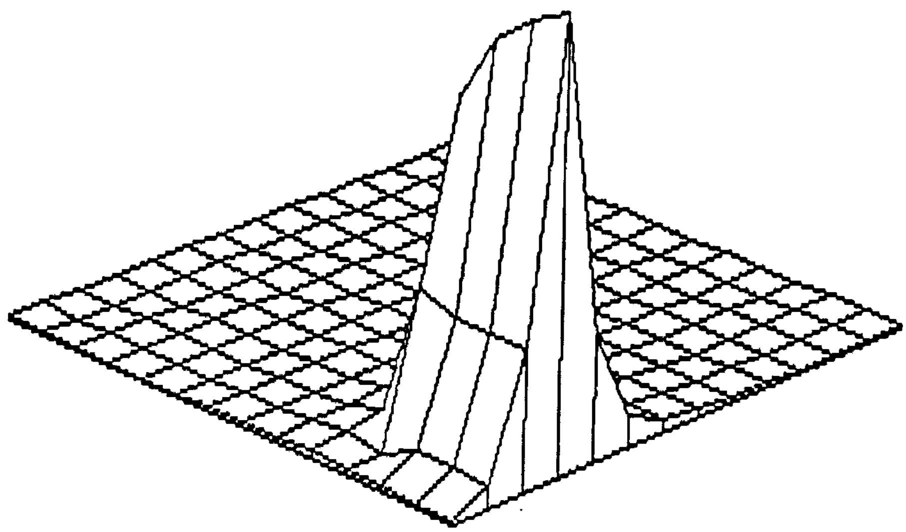

Figure 3.9: Map of strain in horizontally-running fibers.

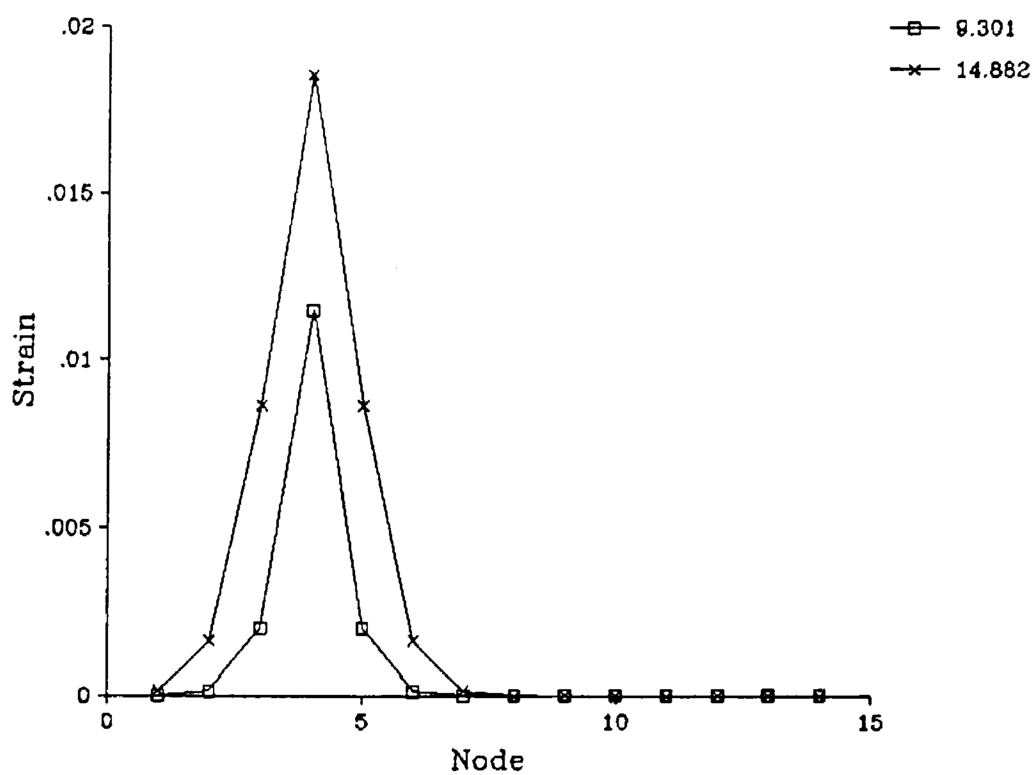

Figure 3.10: Strain profile along fiber passing through impact point.

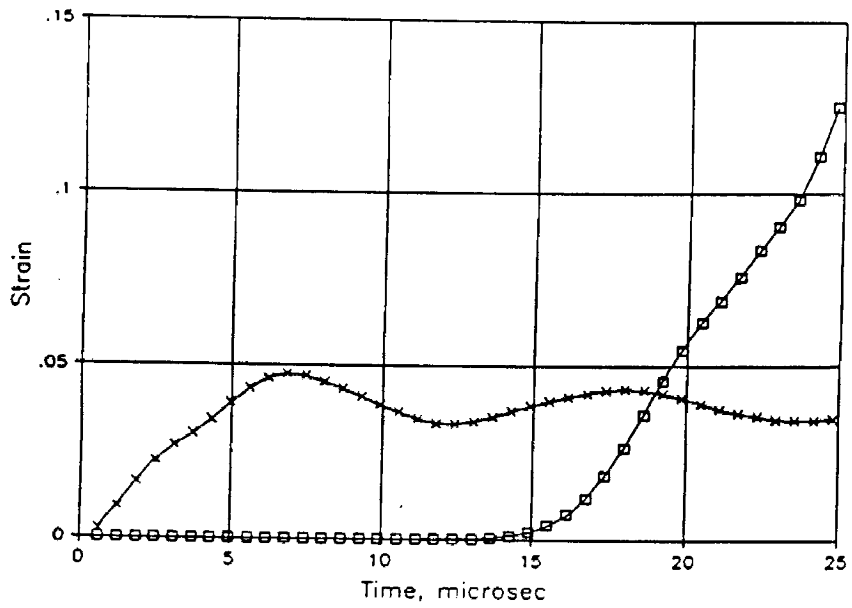

Figure 3.11: Development of strain in fabric with 3 cm covering plate.

	Spectra 900	Kevlar 29
Areal Density, gm/cm^2	0.00237	0.0421
Modulus, gmf/den	1400	550
Yarn Denier	1200	5835
Breaking Strain, %	0.035	0.04

Table 3.1: Properties of candidate ballistic fabrics.

tiveness in the 1970's and is now almost synonymous with impact protection devices. However, as demonstrated in *Fig. 3.6,* Spectra fibers are an obvious choice as an alternative material. With its very high wavespeed ($c = \sqrt{kE} = \sqrt{88,260 \times 1400 \text{ gpd}} =$ 11,116 m/s for Spectra 900), a very rapid rate of projectile energy absorption can be expected. This high modulus and low density are achieved while retaining a high breaking strain of 0.035%. Spectra 900 is tougher than the higher-modulus Spectra 1000, so further design variations will assume a 7 oz/yd^2 plain-weave fabric of Spectra 900 fibers. *Table 3.1* lists certain properties for this fabric, taken from Allied-Signal literature, which are needed in the PLATE code. Data for the Kevlar fabric used in the 1979 report by Roylance & Wang is included for comparison.

3.2.2 Plate Size.

Figure 3.12 illustrates the effect of plate size on the projectile deceleration for a single layer of Spectra 900 fabric impacted at 450 m/s. Here the □ symbol indicates numerical

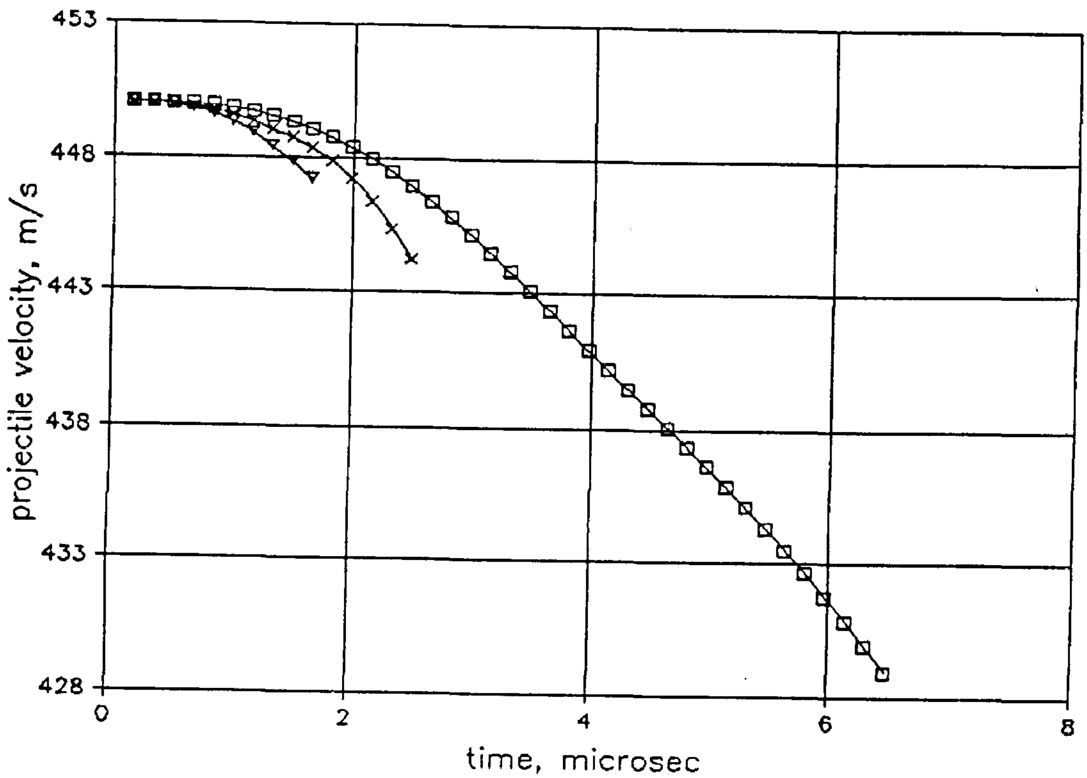

Figure 3.12: Effect of plate width on projectile slowdown.

data for no plate at all, while × and ▽ are for plates 1 and 2 cm wide, respectively. The mass of these plates was taken as zero, so their effect is to spread the impact stresses over a wider area of the fabric without initially reducing the projectile velocity by momentum transfer. As the plate is made wider, the projectile deceleration is increased; this is to be expected since a larger number of fibers is brought to bear in retarding the plate-projectile combination. However, the strain level in the fabric also increases more rapidly as the plate size is increased, so that the time to failure is reduced.

As seen in *Fig. 3.12*, the largest overall reduction in projectile velocity occurs when no plate is present at all. Of course, the inertia of the plate will assist in lowering the projectile velocity by momentum partitioning, and this will help offset the strain-intensifying effect. But in order to minimize this effect the plate size should clearly be taken as small as practicable. A reasonable size might be approximately 2 cm, as this would result in an overlay of adjacent plates without undue fabrication difficulties.

3.2.3 Selection of Plate Material.

Choice of plate material goes somewhat beyond the scope of this study, as it involves considerations of ballistic resistance of homogeneous armor material. This is a very large field of study, with an extensive literature and a sizeable community of active

	Ti	B_4C
Hardness	Br 200	Vi 2800
Modulus, Mpsi	14	
Yield stress, kpsi	150	
Density, gm/ml	4.5	2.52
Cost, $/lb	5	3

Table 3.2: Physical properties of candidate plate materials.

researchers. The ARAP Integral Theory of Impact mentioned earlier is an example of such studies; that work provides a means of ranking materials based on their capacity for ballistic energy absorption. The ARAP classifications showed titanium to be a near optimal armor from several points of view: it requires somewhat more thickness of plate than steel in order to stop a specified projectile, but is some 36% lighter. At approximately $5/lb, it is moderate in cost. It is hard in comparison with most metals (Brinell 200 in comparison with Br 40 for aluminum), which contributes to its ability to blunt or shatter an incoming projectile. *Table 3.2* lists various properties of this material which are relevant to armor design.

Titanium is a natural choice as one candidate for the covering plate material, but several attractive alternatives exist as well. For instance, ceramic materials offer a number of very attractive features in this application. ARAP (1977) studies have shown that B_4C, with a Vickers hardness of 2800, a potential future cost of approximately $3/lb, and a plastic energy absorption capacity of 500 BTU/lb (compared with 328 BTU/lb for titanium) could become a superior alternative as the pertinent costs and technologies improve. B_4C will be a second plate material for design purposes, and its properties are included in *Table 3.2*.

The above materials considerations lead us to propose the following three design configurations:

1. Kevlar fabric fronted by titanium plates.

2. Spectra fabric fronted by titanium plates.

3. Spectra fabric fronted by B_4C plates.

3.2.4 Selection of Armor Thicknesses.

Once a given threat mass and velocity is specified, a thickness of plate and number of fabric layers is chosen to defeat the projectile (or reduce its residual velocity to a specified level). This level of design detail will require collaborative guidance from Natick personnel as to current views of threat distributions. Accordingly, plate/fabric areal density specification will be among the features to be completed in the final phase of the study, to be conducted in collaboration with Natick personnel. The topics to be addressed in this final phase will be listed in the following section.

Chapter 4

Recommendations for Future Work

The work to date in this study has developed a numerical method for simulating a fabric armor with an overlay of flechette-defeating hard plates. It remains in the final phase of the project to use the code, in conjunction with guidance as to current Army performance goals, to develop a detailed armor design which will then be fabricated by the Army and evaluated ballistically. These tasks can be listed as:

- Natick and ACL personnel will work in collaboration, using current threat distributions and casualty reduction goals, to specify the plate and fabric thickness needed for the desired level of protection. The design will be done by performing PLATE code runs to assess the initial-versus-residual projectile velocity response of various plate and fabric thicknesses.

- Test specimens of the designed armor will be procured by the government. If funding permits, it would be desirable to fabricate panels over a range of parameters, to permit corroboration and fine-tuning of the computer model. For instance, the suggested plate size of 2 cm is an estimate based largely on ease of fabrication. It would be of interest to procure panels of varying covering plate sizes, so as to determine experimentally the degree to which this size is critical.

- The fabricated panels will be evaluated by the Army for ballistic performance, by means of firing tests. ACL will collaborate in the planning and execution of both the panel fabrication and firing tests.

- The experimental ballistic observations will be compared with the PLATE code predictions, and directions for future code development will be suggested.

Appendix A

References

Aeronautical Research Associates of Princeton, Inc., "The Development of a Theory for the Design of Lightweight Armor," Technical Report AFFDL-TR-77-114, Air Force Flight Dynamics Laboratory, Wright-Patterson Air Force Base, Ohio, November 1977.

Cunniff, P., personal communication (to be published), U.S. Army Natick Laboratories, 1989.

Mehta, P., and N. Davids, "A Direct Numerical Analysis Method for Cylindrical and Spherical Elastic Waves," *AIAA Journal,* Vol. 4, pp. 112–117, 1966.

Roylance, D., and S.S. Wang, "Penetration Mechanics of Textile Structures," Technical Report CEMEL-218, U.S. Army Natick Research and Development Command, Natick, MA, June 1979.

Lynch, F. deS., "Dynamic Response of a Constrained Fibrous System Subjected to Transverse Impact, Part II: A Mechanical Model, Technical Report TR70-16, Army Materials and Mechanics Research Center, Watertown, MA, July 1970.

Roylance, D., "Wave Propagation in a Viscoelastic Fiber Subjected to Transverse Impact," *Journal of Applied Mechanics,* vol. 40, pp. 143–148, 1973.

Roylance, D., A.F. Wilde, and G.C. Tocci, "Ballistic Impact of Textile Structures," *Textile Research Journal,* Vol. 43, pp. 34–41, 1973.

Roylance, D., "Ballistics of Transversely Impacted Fibers," *Textile Research Journal,* Vol. 47, pp. 679–684, 1977.

Roylance, D., and S.S. Wang, "Penetration Mechanics of Textile Structures: Influence of Non-Linear Viscoelastic Relaxation," *Polymer Engineering and Science,* vol. 18, pp. 1068–1072, 1978.

Roylance, D., and S.S. Wang, "Influence of Fiber Properties on Ballistic Penetration of Textile Panels," *Fibre Science and Technology,* Vol. 14, pp. 183–190, 1981.

Smith, J.C., F.L. McCrackin, and J.F. Schiefer, "Stress-Strain Relationships in Yarns Subjected to Rapid Impact Loading: 3. Effect of Wave Propagation," *Journal of Research of the National Bureau of Standards*, Vol. 55, No. 1, July 1955.

Appendix B

Code User Instructions

The PLATE code is written so as to accept input data on Fortan logical unit #5. On Unix-type systems, this allows the input dataset to be provided to the code by redirection; for instance the file `spec1` can be run by issuing the command `plate <spec1`. As the code runs, it creates a number of output datafiles as described earlier which can be printed or imported into independent graphics routines.

A typical input datafile is shown below:

```
spec1: Spectra 900  450 m/s, 1cm 0gm plate
1.1         | prjmass   projectile mass (gm)
450.        | vproj     initial projectile velocity (m/sec)
9.8         | fmassa    actual fabric mass (gm)
1.          | psize     plate size (cm)
0.          | pltmass   plate mass (gm)
10.16       | xl        fabric panel length (cm)
1200.       | denyrn    yarn denier
.035        | epsb      yarn breaking strain
0.          | xk        elastic spring constant (gmf/sq cm)
100.        | tmax      maximum time (microsec)
1.4142      | cdtm      stability coefficient (.ge. 1)
           1| lyr       number of fabric layers in panel
            |
          40| jt        number of nodes along panel length
           1| inc       print skip increment
           1| execution mode (0 for check, 1 for execution)
            |
           1| flag for nonlinear (exponential) strain hardening
1400.       | eyrn    preexponential factor (gpd)
1.          | chard   strain hardening exponent
            |
           0| flag for nonnlinear polymomial
            | eo      first term (gpd)
            | e1      second term (gpd)
```

```
          | e2      third term (gpd)
          | e3      fourth term (gpd)
          |
         0| flag for standard linear solid viscoelasticity
100.      | g       glassy modulus (gpd)
0.        | vlamda  viscous fraction (.le. 1.)
5.        | tau     relaxation time (microsec)
          |
         0| flag for nonlinear viscoelasticity
          | en1    equilibrium spring constant (gpd)
          | en2    second spring constant (gpd)
          | a      eyring coefficient (1/sec)
          | alp    eyring exponential (1/gpd)
          | vlbd   model viscous fraction (.le. 1)
```

The first line of this file is a comment line which can be filled out as desired; the remaining lines contain numerical input parameters. Only the first ten columns of these parameter lines are read by the computer; the remaining columns are used to annotate the meanings of the various parameters. This provides a template to facilitate the preparation of input files, by copying the file to another name and then editing it as desired. It is important only to respect the ten-column field expected by the Fortran format commands; for instance, integers and exponent fields must be right-justified against column ten. Values with explicit decimals and without exponent fields can be placed anywhere in the first ten columns.

Appendix C

PLATE Source Code

The Fortran module `plate.for` is the main program for the covering-plate model.

```
c     P L A T E

c        A direct numerical solution of the field equations
c        for normal-obliquity ballistic impact of an orthogonal
c        fabric mesh, with an overlying plate.

c        Modified 1988-89 for implementation on MS-DOS and UNIX
c        systems.  Presently written to accept input on unit 5
c        as standard input.  A datafile can then be provided by
c        redirection, e.g.  plate < pdata

c        Output files:
c           unit 6 - log file to be sent to standard output
c           unit 7 - full numerical output, sent to file "plate.out"
c           unit 8 - selected data, sent to file "plate.csv" as
c              comma-separated values
c           unit 9 - zplot.csv, z-displacements of nodes along
c              primary fiber
c           unit 10 - splot.csv, strains along primary fiber
c           unit 11 - smap10 - x-y map of strain in 10 fibers
c           unit 12 - zmap10 - x-y map of z-disp in 10 fibers

c-----------------------------------------------------------------------
c---- DECLARATIONS -----------------------------------------------------

      implicit real*8 (a-h,o-z)
      character*80 hed

      common/ixmatl/ipt
      common/cmatl/eyrn,chard,e0,e1,e2,e3
      common/visco/g,vlamda,tau
      common/nonlr/en1,en2,a,alp,vlbd
      common/varbl/jt,xl,dxl,dtm,denyrn,prjmass,crimp,fmassa,fmassm,
     &   unitm,hunitm,jplate,vproj,epsmax,epsb,tim
      common/tsnold/t10old(40,40),t01old(40,40),e10old(40,40),
     &   e01old(40,40)
```

```
      common/backup/xk
      common/cmesh/xcd(40,40),ycd(40,40),zcd(40,40),xcdold,ycdold,zcdold
      common/ten/t10(40,40),t01(40,40),tx10(40,40),tx01(40,40),
     &   ty10(40,40),ty01(40,40),tz10(40,40),tz01(40,40),
     &   eps10(40,40),eps01(40,40)
      common/veloc/vx(40,40),vy(40,40),vz(40,40)
      common/ergcum/xknrgy,yknrgy,zknrgy,snrgy
      common/ergelem/elke(40,40),else(40,40),tlelen(40,40)

c---- INPUT AND INITIALIZATION----------------------------------------

c     open output files

      open ( 7,file='plate.out')
      open ( 8,file='plate.csv')
      open ( 9,file='zplot.csv')
      open (10,file='splot.csv')
      open (11,file='smap10   ')
      open (12,file='zmap10   ')

c     read and print input parameters
c     and compute internal control parameters

1     format(f10.0)
2     format(i10)
3     format (a80)

      read (5,3) hed
      read (5,1) prjmass
      read (5,1) vproj
      read (5,1) fmassa
      read (5,1) psize
      read (5,1) pltmass
      read (5,1) xl
      read (5,1) denyrn
      read (5,1) epsb
      read (5,1) xk
      read (5,1) tmax
      read (5,1) cdtm
      read (5,2) lyr
      clyr =lyr

      read (5,1) skipln
      read (5,2) jt
      read (5,2) inc
      read (5,2) mode

c  exponential strain hardening
      read (5,1) skipln
      read (5,2) iflag
      if (iflag.eq.1) ipt  = 2
      read (5,1) eyrn
      read (5,1) chard
```

```
c  nonlinear polynomial
      read (5,1) skipln
      read (5,2) iflag
      if (iflag.eq.1) ipt  = 1
      read (5,1) e0
      read (5,1) e1
      read (5,1) e2
      read (5,1) e3

c  standard linear solid
      read (5,1) skipln
      read (5,2) iflag
      if (iflag.eq.1) ipt  = 3
      read (5,1) g
      read (5,1) vlamda
      read (5,1) tau

c  nonlinear viscoelasticity
      read (5,1) skipln
      read (5,2) iflag
      if (iflag.eq.1) ipt  = 4
      read (5,1) en1
      read (5,1) en2
      read (5,1) a
      read (5,1) alp
      read (5,1) vlbd

      cwave  = dsqrt(88260.d0*eyrn)
      jtm1   = jt-1
      dxl    = xl/float(jtm1)
      dtm    = 1.d4*dxl/cwave
      dtm    = dtm/cdtm
      ntinc  = idint(tmax/dtm)
      idiag  = 0
      jplate = 1 + int( .5+( (psize/2.) /dxl) )
      jprt   = jt
      jprty  = 1
      iplast = 0
      tprt   = 0

      denyrn = denyrn*clyr
      fmassa = fmassa*clyr
      fmassm = 4.d0*float(jtm1)*(2.d0*denyrn*xl)/9.d5

      crimp  = fmassa/fmassm
      unitm  = crimp*2.d0*denyrn*dxl/9.d5
      hunitm = 0.5d0*unitm

      totmass= prjmass+pltmass
      vplate = vproj*prjmass/totmass
      vproj  = vplate
      xke    = 0.5d-3*totmass*vproj**2/fmassa

      write (6,1010) hed
```

```
1010 format (1x,a80,/2x,'      time',T15,'   proj. vel.',T30,
     &  '        max. strain',4x,'% error'/)

      write (7,121) hed
  121 format (1x,a80)

      write (8,121) hed

      write(8,1030)
1030  format( /'time,proj vel,proj engy loss,',
     & 'fab en gain,backup engy,strain engy,',
     & 'x - k.e.,y - k.e.,z - k.e.,max strn,'/)

      write(7,61) denyrn,vproj,prjmass,xl,dxl,tmax,cdtm,dtm,lyr
61    format(1x,'input parameters:'//
     &    5x,'yarn denier, denyrn (den) =',f8.1/
     &    5x,'initial projectile velocity, vproj (m/sec) =',f7.1/
     &    5x,'projectile mass, prjmass (gm) =',f8.3/
     &    5x,'fabric panel length, xl (cm) =',f8.2/
     &    5x,'element length dxl, (cm) =',f7.3/
     &    5x,'maximum impact duration, tmax (microsec) =',f7.3/
     &    5x,'stability coefficient, cdtm =',f7.3/
     &    5x,'time increment, dtm (microsec) =',f6.3/
     &    5x,'numbers of layers, lyr =',i3)

      write(7,62) cwave,eyrn,xk
62    format(/5x,'strain wave velocity, cwave (m/sec) =',f8.1/
     &    5x,'initial modulus, eyrn (gr/den) =',f8.2/
     &    5x,'backup spring constant, xk (gr/cm/cm) =',f8.2)

      write(7,63) jt, ntinc,inc
63    format(/5x,'number of nodes along model panel, jt =',i3/
     &    5x,'maximum number of time increments, ntinc =',i4/
     &    5x,'printing frequency, inc =',i3)

      write(7,64) fmassa, fmassm,crimp,unitm,hunitm
64    format (/5x,'actual fabric mass, fmassa (gm) =',f8.2/
     &    5x,'model fabric mass, fmassm (gm) =',f8.2/
     &    5x,'crimp = fmassa/fmassm =', f7.2/
     &    5x,'unit element mass (gm) =',g11.4/
     &    5x,'half-unit element mass (gm) =',g11.4)

      write(7,65) xke
65    format(/5x,'initial projectile kinetic energy, xke (joule/gm) =',
     &    f7.3)

      write(7,652) psize,pltmass,jplate
652   format(/5x,'plate size (cm) =',f8.2,
     &        /5x,'plate mass (gm) =', f8.2,
     &        /5x,'node at edge of plate (jplate) =', i2)


      write(7,66) ipt,eyrn
66    format(/3x,'material properties:'/
```

```
     &   5x,'material model option, ipt =',i2/
     &   5x,'initial modulus, eyrn(gr/den) =',f8.2)

      if (ipt.eq.2) write (7,661) eyrn, chard
  661 format (/1x,'exponential strain hardening'/
     &   5x,'coefficient, eyrn (gr/den) =',f8.2/
     &   5x,'strain hardening exponent, chard =',f8.2/)

      if (ipt.eq.1) write (7,662) e0,e1,e2,e3
  662 format (/1x,'nonlinear elastic model -- cubic polynomial'/
     &   5x,'e0 (gr/den) =',f8.2/
     &   5x,'e1 (gr/den) =',f8.2/
     &   5x,'e2 (gr/den) =',f8.2/
     &   5x,'e3 (gr/den) =',f8.2)

      if (ipt.eq.3) write(7,68) g,vlamda,tau
 68   format(/5x,'viscoelastic - standard linear solid parameters:'/
     &   5x,'model glassy modulus (gpd) =',f8.2/
     &   5x,'model viscous fraction =',f7.2/
     &   5x,'model relaxation time (microseconds) =',f8.2)

      if (ipt.eq.4) write(7,69) en1,en2,a,alp,vlbd
69    format(/1x,'eyring nonlinear model, e1 (gpd) =',d14.5/
     &   5x,'Eyring nonlinear model, e2 (gpd) =',d14.5/
     &   5x,'nonlinear dashpot, a (1/sec) =',d14.5/
     &   5x,'nonlinear dashpot, alp (den/gf) =',d14.5/
     &   5x,'nonlinear elastic arm ratio, vlbd =',d14.5)

      write(7,67) epsb
67    format(/5x,'fiber rupture strain, epsb =',f7.3/)

c     initialize field variables
c     establish initial nodal coordinates

      do 11 k=1,jt
      do 11 j=1,jt

         xcd(j,k)=(j-1)*dxl
         ycd(j,k)=(k-1)*dxl
         zcd(j,k)=0.d0

         vx(j,k)=0.d0
         vy(j,k)=0.d0
         vz(j,k)=0.d0

         t10 (j,k)=0.d0
         t01 (j,k)=0.d0
         tx10(j,k)=0.d0
         ty10(j,k)=0.d0
         tz10(j,k)=0.d0
         tx01(j,k)=0.d0
         ty01(j,k)=0.d0
         tz01(j,k)=0.d0
```

```
            eps10(j,k)=0.d0
            eps01(j,k)=0.d0

            t10old(j,k)=0.d0
            t01old(j,k)=0.d0
            e10old(j,k)=0.d0
            e01old(j,k)=0.d0

11    continue

      inc1=0
      nfront=jtm1*2

c---- LOOP OVER SUCCESSIVE TIME STEPS ---------------------------------

      do 12 it=1,ntinc

      vz(1,1)=vplate
      xknrgy=0.d0
      yknrgy=0.d0
      zknrgy=0.d0
      snrgy=0.d0
      tim=dtm*float(it)
      inc1=inc1+1
      epsmax=0.d0

c---- LOOP OVER SUCCESSIVE DIAGONAL WAVEFRONT POSITIONS ---------------
c          (starting point moves horizontally from node to node away from
c           the impact point, then up the vertical line of nodes
c           at the clamp.  There are thus nfront=(jt-1)*2
c           wavefront positions.)

      call bndry(it,1,1)

c     limit space loop to region of influence
      imax = jplate+it+2
      if (imax.gt.nfront) imax=nfront

      do 14 ifront=2,imax

c     locate nodes along front

         if(ifront.lt.jt)  call bndry(it,ifront,1)
         iflag=0
         nodes=ifront/2
         if(ifront-(jt+1).lt.0) then
            kstart=1
         else
            kstart =ifront-(jt-1)
         endif


c---- INNER LOOP ALONG NODES ON DIAGONAL WAVEFRONT --------------------
```

```
20      do 22 k=kstart,nodes

c       get nodal indices: j = column, k = row

        j=ifront-k
        if(k.eq.j) idiag=1

c       compute velocity changes from inpulse-momentum balance

        dvx=(tx10(j+1,k+1)-tx10(j,k+1)+tx01(j+1,k+1)-tx01(j+1,k))
     &        *dtm*8.82d0/(2.d0*dxl*crimp)
        dvy=(ty10(j+1,k+1)-ty10(j,k+1)+ty01(j+1,k+1)-ty01(j+1,k))
     &        *dtm*8.82d0/(2.d0*dxl*crimp)
        dvz=(tz10(j+1,k+1)-tz10(j,k+1)+tz01(j+1,k+1)-tz01(j+1,k))
     &        *dtm*8.82d0/(2.d0*dxl*crimp)
     &        +xk*dxl*zcd(j+1,k+1)*dtm*9.8d-6/(unitm/9.d5)
        dvz=-dvz

        if( abs(dvx).ge.0.0001.or.abs(dvy).ge.0.0001
     &                   .or.abs(dvz).ge.0.0001) iflag=iflag+1

c       compute new velocities

        vx(j+1,k+1)=vx(j+1,k+1)+dvx
        vy(j+1,k+1)=vy(j+1,k+1)+dvy
        vz(j+1,k+1)=vz(j+1,k+1)+dvz

c       fix nodal velocities along panel edge

           vx(jt,k+1)=0.d0
           vy(jt,k+1)=0.d0
           vz(jt,k+1)=0.d0

c       fix z-velocities of nodes under plate

        if(j.lt.jplate) vz(j+1,k+1)=vplate

c       set velocities in second octant by symmetry

        vx(k+1,j+1)=vy(j+1,k+1)
        vy(k+1,j+1)=vx(j+1,k+1)
        vz(k+1,j+1)=vz(j+1,k+1)

        if((ifront.gt.jtm1).and.((j+1).eq.jt)) go to 24

c       compute change of nodal coordinates from velocities

        xcd(j+1,k)=xcdold
        ycd(j+1,k)=ycdold
        zcd(j+1,k)=zcdold

        xcd(k,j+1)=ycdold
        ycd(k,j+1)=xcdold
        zcd(k,j+1)=zcdold
```

```
c        compute change in element strain from new coordinates
c           first, vertical element from (j+1,k) to (j+1,k+1)

24       dsq01=(xcd(j+1,k+1)-xcd(j+1,k))**2
     &       +(ycd(j+1,k+1)-ycd(j+1,k))**2
     &       +(zcd(j+1,k+1)-zcd(j+1,k))**2

         dist01=dsqrt(dsq01)

         dsqx01=(xcd(j+1,k+1)+vx(j+1,k+1)*dtm/1.d4
     &          -xcd(j+1,k  )-vx(j+1,k  )*dtm/1.d4)**2
         dsqy01=(ycd(j+1,k+1)+vy(j+1,k+1)*dtm/1.d4
     &          -ycd(j+1,k  )-vy(j+1,k  )*dtm/1.d4)**2
         dsqz01=(zcd(j+1,k+1)+vz(j+1,k+1)*dtm/1.d4
     &          -zcd(j+1,k  )-vz(j+1,k  )*dtm/1.d4)**2

         ddst01=dsqrt(dsqx01+dsqy01+dsqz01)
         deps=ddst01/dist01-1.d0
         eps01(j+1,k)=eps01(j+1,k)+deps

c        check for max strain

         if(eps01(j+1,k).gt.epsb) then
            write(7,242) tim,vproj
            write(*,242) tim,vproj
242         format(//1x,'strain rupture at t = ',g14.5,' vproj =',g14.5)
            stop
         endif
         epsmax=dmax1( epsmax, dabs(eps01(j+1,k)) )

         if(idiag.eq.1) go to 26
         eps10(k,j+1)=eps01(j+1,k)

c           next, horizontal element from (j,k+1) to (j+1,k+1)

 26      dsq10=(xcd(j+1,k+1)-xcd(j,k+1))**2
     &       +(ycd(j+1,k+1)-ycd(j,k+1))**2
     &       +(zcd(j+1,k+1)-zcd(j,k+1))**2
         dist10=dsqrt(dsq10)
         dsqx10=(xcd(j+1,k+1)+vx(j+1,k+1)*dtm/1.d4
     &          -xcd(j  ,k+1)-vx(j  ,k+1)*dtm/1.d4)**2
         dsqy10=(ycd(j+1,k+1)+vy(j+1,k+1)*dtm/1.d4
     &          -ycd(j  ,k+1)-vy(j  ,k+1)*dtm/1.d4)**2
         dsqz10=(zcd(j+1,k+1)+vz(j+1,k+1)*dtm/1.d4
     &          -zcd(j  ,k+1)-vz(j  ,k+1)*dtm/1.d4)**2

         ddst10=dsqrt(dsqx10+dsqy10+dsqz10)
         deps=ddst10/dist10-1.d0
         eps10(j,k+1)=eps10(j,k+1)+deps
         eps01(k+1,j)=eps10(j,k+1)

c        check for max strain
```

```
        if(eps10(j,k+1).gt.epsb) then
           write(7,242) tim,vproj
           write(*,242) tim,vproj
           stop
        endif
        epsmax=dmax1( epsmax, dabs(eps10(j,k+1)) )

c       call tensn subroutine to compute element
c       tension and strain energy from element strain
c       or strain history.
c          first, vertical element from (j+1,k) to (j+1,k+1)

        epslny=eps01(j+1,k)
        epyold=e01old(j+1,k)
        tsyold=t01old(j+1,k)
        jp1=j+1
        call tensn(it,jp1,k,epslny,tsly,epyold,tsyold,dsnrgy)

        t01   (j+1,k)=tsly
        t01old(j+1,k)=tsyold
        t10old(k,j+1)=t01old(j+1,k)
        e01old(j+1,k)=epyold
        e10old(k,j+1)=e01old(j+1,k)

        bsem=1.d0
        if(j.gt.1.and.k.eq.1) bsem=0.5d0
        snrgy=snrgy+dsnrgy*(2-idiag)*bsem

c          next, horizontal element from (j,k+1) to (j+1,k+1)
        epslnx=eps10(j,k+1)
        epxold=e10old(j,k+1)
        tsxold=t10old(j,k+1)

        kp1=k+1
        call tensn(it,j,kp1,epslnx,tslx,epxold,tsxold,dsnrgy)

        t10   (j,k+1)=tslx
        t10old(j,k+1)=tsxold
        t01old(k+1,j)=t10old(j,k+1)
        e10old(j,k+1)=epxold
        e01old(k+1,j)=e10old(j,k+1)
        if(j.eq.1.and.k.eq.1) go to 51
          snrgy=snrgy+dsnrgy*(2-idiag)

51      t10(k,j+1)=t01(j+1,k)
        t01(k+1,j)=t10(j,k+1)

        xcdold=xcd(j,k+1)
        ycdold=ycd(j,k+1)
        zcdold=zcd(j,k+1)

        elke(j,k+1)=elke(j,k+1)+.5d-3*hunitm*
     &     (vx(j,k+1)**2+vy(j,k+1)**2+vz(j,k+1)**2)/fmassa
        elke(j+1,k)=elke(j+1,k)+.5d-3*hunitm*
```

```
     &      (vx(j+1,k)**2+vy(j+1,k)**2+vz(j+1,k)**2)/fmassa
            elke(k+1,j)=elke(j,k+1)
            elke(k,j+1)=elke(j+1,k)
            tlelen(j,k+1)=else(j,k+1)+elke(j,k+1)
            tlelen(j+1,k)=else(j+1,k)+elke(j+1,k)
            tlelen(k,j+1)=tlelen(j+1,k)
            tlelen(k+1,j)=tlelen(j,k+1)

            bkexyk=1.d0
            bkexyj=1.d0

            if(k.eq.1.and.j.lt.jt) bkexyk=0.50d0
            if(j.eq.1.and.k.lt.jt) bkexyj=0.50d0

            xknrgy=xknrgy+4.d0*0.50d-3*hunitm
     &            *(bkexyk*vx(j+1,k)**2+bkexyj*vx(j,k+1)**2)
     &            /(fmassa)
            yknrgy=yknrgy+4.d0*0.50d-3*hunitm
     &            *(bkexyk*vy(j+1,k)**2+bkexyj*vy(j,k+1)**2)
     &            /(fmassa)
            zknrgy=zknrgy+(2.0d0-idiag)*4.d0*0.50d-3*hunitm
     &            *(bkexyk*vz(j+1,k)**2+bkexyj*vz(j,k+1)**2)
     &            /(fmassa)


c           add symmetric components to energy unless on diagonal

            if(idiag.eq.1) go to 28
            xknrgy=xknrgy+4.d0*0.5d-3*hunitm
     &            *(bkexyk*vx(k,j+1)**2+bkexyj*vx(k+1,j)**2)
     &            /(fmassa)
            yknrgy=yknrgy+4.d0*0.5d-3*hunitm
     &            *(bkexyk*vy(k,j+1)**2+bkexyj*vy(k+1,j)**2)
     &            /(fmassa)

c           store new nodal coordinates

28          xcd(j+1,k)=xcd(j+1,k)+vx(j+1,k)*dtm/1.d4
            ycd(j+1,k)=ycd(j+1,k)+vy(j+1,k)*dtm/1.d4
            zcd(j+1,k)=zcd(j+1,k)+vz(j+1,k)*dtm/1.d4

            xcd(j,k+1)=xcd(j,k+1)+vx(j,k+1)*dtm/1.d4
            ycd(j,k+1)=ycd(j,k+1)+vy(j,k+1)*dtm/1.d4
            zcd(j,k+1)=zcd(j,k+1)+vz(j,k+1)*dtm/1.d4

            if(j.eq.k) go to 15
            xcd(k,j+1)=xcd(k,j+1)+vx(k,j+1)*dtm/1.d4
            ycd(k,j+1)=ycd(k,j+1)+vy(k,j+1)*dtm/1.d4
            zcd(k,j+1)=zcd(k,j+1)+vz(k,j+1)*dtm/1.d4

            if(j.eq.(k+1)) go to 15
            xcd(k+1,j)=xcd(k+1,j)+vx(k+1,j)*dtm/1.d4
            ycd(k+1,j)=ycd(k+1,j)+vy(k+1,j)*dtm/1.d4
            zcd(k+1,j)=zcd(k+1,j)+vz(k+1,j)*dtm/1.d4
```

```
c       obtain vector components of element tensions

 15     a1=atan((ycd(j+1,k+1)-ycd(j,k+1))
     &         /(xcd(j+1,k+1)-xcd(j,k+1)))
        cosa1=cos(a1)
        sina1=sin(a1)

        if(abs(zcd(j+1,k+1)-zcd(j,k+1))- 0.0001) 90,90,91
 90     cosb1=0.d0
        sinb1=1.d0

        go to 92
 91     b1=atan((((xcd(j+1,k+1)-xcd(j  ,k+1))**2
     &          +(ycd(j+1,k+1)-ycd(j  ,k+1))**2)**0.50
     &          /(zcd(j  ,k+1)-zcd(j+1,k+1)))
        cosb1=cos(b1)
        sinb1=sin(b1)

 92     if(abs(xcd(j+1,k+1)-xcd(j+1,k))- 0.0001) 96,96,97
 96     cosa2=0.d0
        sina2=1.d0

        go to 98
 97     a2=atan((ycd(j+1,k+1)-ycd(j+1,k))
     &         /(xcd(j+1,k+1)-xcd(j+1,k)))
        cosa2=cos(a2)
        sina2=sin(a2)

 98     if(abs(zcd(j+1,k+1)-zcd(j+1,k))- 0.0001) 93,93,94
 93     cosb2=0.d0
        sinb2=1.d0

        go to 95
 94     b2=atan((((xcd(j+1,k+1)-xcd(j+1,k))**2
     &          +(ycd(j+1,k+1)-ycd(j+1,k))**2)**0.50
     &          /(zcd(j+1,k  )-zcd(j+1,k+1)))
        cosb2=cos(b2)
        sinb2=sin(b2)

 95     tx10(j,k+1)=t10(j,k+1)*sinb1*cosa1
        ty10(j,k+1)=t10(j,k+1)*sinb1*sina1
        tz10(j,k+1)=t10(j,k+1)*cosb1

        tx01(j+1,k)=t01(j+1,k)*sinb2*cosa2
        ty01(j+1,k)=t01(j+1,k)*sinb2*sina2
        tz01(j+1,k)=t01(j+1,k)*cosb2

        t10(k,j+1)=t01(j+1,k)
        t01(k+1,j)=t10(j,k+1)

        t10old(k,j+1)=t01old(j+1,k)
        t01old(k+1,j)=t10old(j,k+1)
```

```
      tx10(k,j+1)=ty01(j+1,k)
      ty10(k,j+1)=tx01(j+1,k)
      tz10(k,j+1)=tz01(j+1,k)

      tx01(k+1,j)=ty10(j,k+1)
      ty01(k+1,j)=tx10(j,k+1)
      tz01(k+1,j)=tz10(j,k+1)

      idiag=0

c---- END LOOP OVER NODES ON DIAGONAL --------------------------------

22    continue

c---- END LOOP OVER DIAGONAL WAVEFRONT POSITIONS ----------------------

14    continue

c     compute plate/projectile slowdown

49    zforce=0.d0
      do 50 k=1,jplate
         aproj=atan((zcd(jplate+1,k)-zcd(jplate,k))
     &              /(xcd(jplate+1,k)-xcd(jplate,k)))
         zforce=zforce + t10(jplate,k)*sin(aproj)
50    continue
      bforce=xk*zcd(1,1) * ((2*jplate-1)**2)
         vplate=vplate+8.d0*zforce*dtm*denyrn*9.80e-06*crimp/totmass
     &          -bforce*dtm*9.80e-06/totmass
         vproj=vplate

      dke=xke-0.50d-3*totmass*vproj**2/(fmassa)
      tlegfr=snrgy+xknrgy+yknrgy+zknrgy
      error = 100.*(1.-2.*tlegfr)/dke
      engybk=dke-tlegfr

c     check for end of run

      if(vproj.le.0.d0) then
         write(7,82) tim,vproj,dke,tlegfr,engybk,snrgy,
     &               xknrgy,yknrgy,zknrgy
82       format(//1x,'at projectile stop:'/1x,4x,9e14.5)
         stop
      endif

      if(xk.lt.0.001) engybk=0.d0

c     print current field variables if at selected time increment

c     entire strain and z-displacement fields at 10 microsec

      if ( tim.gt.10. .and. tprt.eq.0 ) then
         tprt=1
         do 86 j=1,jt
```

```
         do 86 k=1,jt
            write (11,84) xcd(j,k),ycd(j,k),eps10(j,k)
            write (12,84) xcd(j,k),ycd(j,k),  zcd(j,k)
84          format(1x,3f15.4)
86       continue
      endif

      if(incl.eq.inc) then

c        full data to plate.out file (unit 7):

         write (7,2000) tim,vproj,dke,tlegfr,engybk,snrgy,xknrgy,
     &                  yknrgy,zknrgy,error,epsmax
2000     format(/1x,'time = ',g12.4,' microseconds',
     &          /4x,'projectile velocity =',g12.4,' m/sec',
     &          /4x,'projectile energy loss =',g12.4,' joules',
     &          /4x,'fabric energy gain =',g12.4,' joules',
     &          /4x,'energy in backup =',g12.4,' joules',
     &          /4x,'fabric strain energy =',g12.4,' joules',
     &          /4x,'x - kinetic energy =',g12.4,' joules',
     &          /4x,'y - kinetic energy =',g12.4,' joules',
     &          /4x,'z - kinetic energy =',g12.4,' joules',
     &          /4x,'energy discrepancy, % =',g12.4/,
     &          /4x,'maximum strain =',g12.4/)

802      write(7,72) ((t10(ix,iy),ix=1,jprt),iy=1,jprty)
72       format(1x,'t10(ix,iy)='/8(g12.4))

         write(7,74)((t01(jx,jy),jx=1,jprt),jy=1,jprty)
74       format(1x,'t01(jx,jy)='/8(g12.4))

         write (7,702) (( vz(nx,ny),nx=1,jprt),ny=1,jprty)
702      format(1x,' vz(i,j)='/9(g12.4))

         write(7,701) ((xcd(mx,my),mx=1,jprt),my=1,jprty)
701      format(1x,'xcd(i,j)='/9(g12.4))

         write(7,703) ((zcd(icx,icy),icx=1,jprt),icy=1,jprty)
703      format(1x,'zcd(i,j)='/9(g12.4))

         write(7,76) ((eps10(kx,ky),kx=1,jprt),ky=1,jprty)
76       format(1x,'eps10(kx,ky)='/8(g12.4))

         write(7,78) ((eps01(lx,ly),lx=1,jprt),ly=1,jprty)
78       format(1x,'eps01(lx,ly)='/8(g12.4))

c        brief summary to screen (unit 6):

         write(6,1020) tim,vproj,epsmax,error
1020     format (1x,2g15.4,f15.4,g15.4)

c        comma-separated values to units 8,9,10:

c        overall values for armor at this time increment...
```

```
        write(8,1015) tim,vproj,dke,tlegfr,engybk,snrgy,xknrgy,
     &                yknrgy,zknrgy,epsmax
1015    format( 4x,10(g14.5,','))

c       values of z-coordinate and strain along primary fiber...

        if(it.eq.1) write (9,1034) (i,i=1,jt)
1034    format(4x,41(i5,','))
        write( 9,1035) tim,(  zcd(i,1),i=1,jt)
        if(it.eq.1) write (10,1034) (i,i=1,jt)
        write(10,1035) tim,(eps10(i,1),i=1,jt)
1035    format(4x,41(g14.5,','))

        inc1=0

      endif

c---- END TIME LOOP ------------------------------------------------

12    continue

      write(7,102) tmax
102   format(//2x,'stopped at max time: t = ', 2x,e14.5)
      stop
      end
```

The subroutine bndry.for is used to perform computations for nodes lying along the horizontal fiber passing through the impact point.

```
      subroutine bndry(it,j,k)

      implicit real*8 (a-h,o-z)
      common/ixmatl/ipt
      common/cmatl/eyrn,chard,e0,e1,e2,e3
      common/visco/g,vlamda,tau
      common/varbl/jt,xl,dxl,dtm,denyrn,prjmass,crimp,fmassa,fmassm,
     &   unitm,hunitm,jplate,vproj,epsmax,epsb,tim
      common/tsnold/t10old(40,40),t01old(40,40),
     &             e10old(40,40),e01old(40,40)
      common/backup/xk
      common/cmesh/xcd(40,40),ycd(40,40),zcd(40,40),xcdold,ycdold,zcdold
      common/ten/t10(40,40),t01(40,40),tx10(40,40),tx01(40,40),
     &          ty10(40,40),ty01(40,40),tz10(40,40),tz01(40,40),
     &          eps10(40,40),eps01(40,40)
      common/veloc/vx(40,40),vy(40,40),vz(40,40)
      common/ergcum/xknrgy,yknrgy,zknrgy,snrgy
      common/ergelem/elke(40,40),else(40,40),tlelen(40,40)

      dvx=(tx10(j+1,k)-tx10(j,k)+2.d0*tx01(j+1,k))*dtm
     &       *8.82d0/(2.d0*dxl*crimp)
      dvz=(tz10(j+1,k)-tz10(j,k)+2.d0*tz01(j+1,k))*dtm
     &       *8.82d0/(2.d0*dxl*crimp)
```

```
&        +xk*dxl*zcd(j+1,k)*dtm*9.8d-6/(unitm/9.0d5)
      dvz=-dvz

      vx(j+1,k)=vx(j+1,k)+dvx
      vy(j+1,k)=0.d0
      vz(j+1,k)=vz(j+1,k)+dvz

         if(j.lt.jplate) vz(j+1,k)=vproj

      vx(jt,k)=0.d0
      vy(jt,k)=0.d0
      vz(jt,k)=0.d0

      vx(k,j+1)=vy(j+1,k)
      vy(k,j+1)=vx(j+1,k)
      vz(k,j+1)=vz(j+1,k)

      dsqx=(xcd(j+1,k)+vx(j+1,k)*dtm/1.d4-xcd(j,k)
     &               -vx(j,k)*dtm/1.d4)**2
      dsqy=(ycd(j+1,k)+vy(j+1,k)*dtm/1.d4-ycd(j,k)
     &               -vy(j,k)*dtm/1.d4)**2
      dsqz=(zcd(j+1,k)+vz(j+1,k)*dtm/1.d4-zcd(j,k)
     &               -vz(j,k)*dtm/1.d4)**2

      ddl= dsqrt(dsqx+dsqy+dsqz)
      dsq=(xcd(j+1,k)-xcd(j,k))**2+(ycd(j+1,k)-ycd(j,k))**2
     &   +(zcd(j+1,k)-zcd(j,k))**2
      dist= dsqrt(dsq)

      deps=ddl/dist-1.d0
      eps10(j,k)=eps10(j,k)+deps

c     check for max strain

      if(eps10(j,k).gt.epsb) then
         write(7,242) tim,vproj
         write(*,242) tim,vproj
242      format(//1x,'strain rupture at t = ',g14.5,' vproj =',g14.5)
         stop
      endif
      epsmax=dmax1( epsmax, dabs(eps10(j,k)) )

      eps01(k,j)=eps10(j,k)
      tslold=t10old(j,k)
      epsold=e10old(j,k)
      epsln=eps10(j,k)
      call tensn(it,j,k,epsln,tsl,epsold,tslold,dsnrgy)

      t10(j,k)=tsl
      t01(k,j)=t10(j,k)
      t10old(j,k)=tslold
      t01old(k,j)=t10old(j,k)
      e10old(j,k)=epsold
      e01old(k,j)=e10old(j,k)
```

```
      bkexy=2.d0
      if(j.eq.1.and.k.eq.1) bkexy=1.d0

      xknrgy=xknrgy+bkexy*0.5d-3*hunitm*vx(j,k)**2/(fmassa)
     &             +bkexy*0.5d-3*hunitm*vx(k,j)**2/(fmassa)
      yknrgy=yknrgy+bkexy*0.5d-3*hunitm*vy(j,k)**2/(fmassa)
     &             +bkexy*0.5d-3*hunitm*vy(k,j)**2/(fmassa)

      bkezz=4.d0
      if(j.eq.1.and.k.eq.1) bkezz=2.d0
      zknrgy=zknrgy+bkezz*0.5d-3*hunitm*vz(j,k)**2/(fmassa)
      snrgy=snrgy+dsnrgy

      ctrse=4.d0
      if(j.eq.1 .and. k.eq.1) ctrse=1.d0
      else(j,k)=else(j,k)+dsnrgy/ctrse
      else(k,j)=else(j,k)
      ctrke=1.d0
      if(j.eq.1 .and. k.eq.1) ctrke=2.d0
      elke(j,k)=elke(j,k)+ctrke*.5d-3*hunitm*
     &   (vx(j,k)**2+vy(j,k)**2+vz(j,k)**2)/fmassa
      elke(k,j)=elke(j,k)
      tlelen(j,k)=else(j,k)+elke(j,k)
      tlelen(k,j)=tlelen(j,k)

      xcdold=xcd(j,k)
      ycdold=ycd(j,k)
      zcdold=zcd(j,k)

      xcd(j,k)=xcd(j,k)+vx(j,k)*dtm/1.d4
      ycd(j,k)=ycd(j,k)+vy(j,k)*dtm/1.d4
      zcd(j,k)=zcd(j,k)+vz(j,k)*dtm/1.d4

      if(j.eq.k) go to 20
      xcd(k,j)=xcd(k,j)+vx(k,j)*dtm/1.d4
      ycd(k,j)=ycd(k,j)+vy(k,j)*dtm/1.d4
      zcd(k,j)=zcd(k,j)+vz(k,j)*dtm/1.d4

20    a=atan((ycd(j+1,k)-ycd(j,k))/(xcd(j+1,k)-xcd(j,k)))
      cosa=cos(a)
      sina=sin(a)
      if(abs(zcd(j+1,k)-zcd(j,k))-0.0001) 10,10,11
10    cosb=0.d0
      sinb=1.d0
      go to 12

11    b=atan((xcd(j+1,k)-xcd(j,k))/(zcd(j,k)-zcd(j+1,k)))
      cosb=cos(b)
      sinb=sin(b)

12    tx10(j,k)=t10(j,k)*sinb*cosa
      ty10(j,k)=t10(j,k)*sinb*sina
      tz10(j,k)=t10(j,k)*cosb
      tx01(k,j)=ty10(j,k)
```

```
      ty01(k,j)=tx10(j,k)
      tz01(k,j)=tz10(j,k)

      return
      end
```

The subroutine tensn.for computes the tension resulting from a given strain, and also computes the strain energy in the fiber element.

```
      subroutine tensn(it,j,k,epss,ts,epsold,tslold,dsen)

c        use appropriate constitutive model for
c        computation of element tension from strain
c        or strain history

      implicit real*8 (a-h,o-z)
      external f
      common/ixmatl/ipt
      common/cmatl/eyrn,chard,e0,e1,e2,e3
      common/visco/g,vlamda,tau
      common/nonlr/en1,en2,a,alp,vlbd
      common/noneq/a1,a2,a3
      common/varbl/jt,xl,dxl,dtm,denyrn,prjmass,crimp,fmassa,fmassm,
     &   unitm,humitm,jplate,vproj
      common/tsnold/t10old(40,40),t01old(40,40),
     &   e10old(40,40),e01old(40,40)
      common/backup/xk
      dimension x(1)

      if(epss.le.0.d0) go to 1
      symctr=1.00e 00
      if(j.eq.1 .and. k.eq.1) symctr=0.5d0

      if(ipt.eq.2) go to 12
      if(ipt.eq.3) go to 14
      if(ipt.eq.4) go to 18

c     nonlinear (cubic polynomial) elastic model
c
      ts = e0 + e1*epss + e2*epss**2 + e3*epss**3
      dsen=4.d0*(e0*epss+0.5d0*epss**2
     &   +0.33333d0*epss**3+0.25d00*epss**4)
     &   *denyrn*dxl*crimp*9.8d-5*symctr/fmassa
      go to 2
c
c     nonlinear (exponential strain hardening) elastic model
c
12    ts=eyrn*epss**chard
      dsen=(4.d0/(1.d0+chard))*eyrn * epss**(1.d0+chard)
     &   *denyrn*dxl*crimp*9.80e-05*symctr/fmassa
      go to 2
c
c     linear (standard linear solid) viscoelastic model
```

```
c
14    b=g*(1.d0-vlamda)/tau
      d=1.d0 + dtm/tau
      if(it.gt.1) go to 16
         epsold=0.d0
         tslold=0.d0
16    ts=(g*(epss-epsold) + b*epss*dtm+tslold) / d
      dsen=(4.d0 * 0.5d0 * (ts+tslold) * (epss-epsold)
     &   * dxl * denyrn * crimp * 9.8d-5 * symctr) / fmassa
      epsold=epss
      tslold=ts
      go to 2
c
c     nonlinear (Eyring) viscoelastic model
c
18    if(it.gt.1) go to 20
      epsold=0.00e 00
      tslold=0.00e 00
20    b1=en2*a*dtm
      b2=en2*(1.00d 00+vlbd)
      b3=alp*en1*epss
      b4=epss-epsold
      a1=b1*dcosh(b3)
      a2=-b1*dsinh(b3)
      a3=-b2*b4-tslold
      eps=1.00d-06
      eps2=1.00d-06
      eta=1.00d-02
      nsig=6
      n=1
      x(1)=tslold
      itmax=100
      call znonlr(f,eps,eps2,eta,nsig,n,x,itmax,ier)
      ts=x(1)
      dsen=(4.00e 00*0.50e 00*(ts+tslold)*(epss-epsold)*dxl*denyrn*crimp
     &*9.80e-05*symctr)/fmassa
      epsold=epss
      tslold=ts
      go to 2
1     ts=0.00e 00
      dsen=0.00e 00
2     return
      end


      real*8 function f(s)
      implicit real*8 (a-h,o-z)
      common/nonlr/en1,en2,a,alp,vlbd
      common/noneq/a1,a2,a3
      f=s+a1*dsinh(alp*s)+a2*dcosh(alp*s)+a3
      return
      end
```