MULTI-TIME-STEP DOMAIN DECOMPOSITION AND COUPLING METHODS FOR NON-LINEAR STRUCTURAL DYNAMICS

BY

ARUN PRAKASH

B.Tech., Indian Institute of Technology, 1999
M.S., University of Illinois at Urbana-Champaign, 2001

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Civil Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2007

Urbana, Illinois

Abstract

The need for better and more efficient computational methods to study large-scale coupled physical phenomena has grown significantly over the last couple of decades. Coupled multi-physics problems are extremely challenging because of the disparity in the length and time scales that are usually involved. Researchers have devoted significant effort to address the coupling of multi-physics phenomena in space through techniques such as domain decomposition and multi-scale methods but a commensurate effort to couple multiple scales in time is lacking in the current literature.

This dissertation addresses the problem of efficiently coupling multiple time scales for non-linear dynamic analysis of large structures with complex geometries. Such problems have been solved, in the past, by discretizing the structural domain in space with finite elements and using a time-stepping scheme for numerical time integration. However, using a uniform time step for the entire mesh that meets that stability and accuracy requirements of all the elements is computationally very inefficient.

Early attempts to overcome this problem used domain decomposition to divide a large structural mesh into two or more subdomains. This allowed one to formulate multi-time-step methods where the time step and/or the time stepping scheme could be chosen in accord with the requirements of individual subdomains. However, multi-time-step methods in the literature thus far, usually fail to preserve the stability and accuracy of solutions and are computationally inefficient.

The present multi-time-step coupling method is based on a dual Schur domain decomposition method that uses Lagrange multipliers to enforce the continuity of the solution across

the interfaces between the subdomains. The subdomains can be integrated with different time steps and/or time stepping schemes. The method was shown to be unconditionally stable, energy preserving and computationally very efficient. The multi-time-step method has also been extended for non-linear problems and a recursive coupling method was developed to couple multiple subdomains with multiple levels of time steps. An efficient parallelization based on message passing that uses a recursive tree topology for distributing subdomains between processors is also presented.

To a lifetime of learning mechanics ...

and to my family and friends.

Acknowledgements

First, I would like to thank Professor Keith D. Hjelmstad for being an incredible advisor to me and for teaching me mechanics. I am indebted to him for his guidance and encouragement through tough times. He has helped me not only shape my career and grow professionally, but has also had an indelible effect on my personality.

I would also like to thank the distinguished members of my examination committee, Professor Michael T. Heath, Professor Robert H. Dodds Jr. and Professor Daniel A. Tortorelli for their time and effort in going through my work and for their valuable suggestions. A special thanks is due to Professor Tortorelli for his insightful feedback on a variety of topics that came up during discussions in our research group meetings.

I am fortunate to be associated with a strong and active research group that has enabled me to appreciate and explore various problems at the cutting edge of research in computational mechanics. It has made research and academics an enjoyable experience for me over the last few years. I thank Dr. Alireza Namazifard, Wei Xu, Kalyanababu Nakshatrala, Ghadir Haikal, Kristine Cochran, Daniel Turner, Sudhir Singamsethi and other past and present members of Professor Hjelmstad's research group for their interest and lively discussions. I particularly enjoyed the numerous occasions on which the Hjelmstads invited us to their lovely home to spend time with their family and friends.

During the early part of my graduate studies, I came in contact with some researchers who helped me in various ways to get started on my problem. I thank Dr. Dennis Parsons, Professor Philippe Geubelle, Professor Amit Acharya, Professor Ertugrul Taciroglu, Dr. Jason Hales and Dr. Nathan Crane for the same.

I am grateful for the generous support I have received from the Center for Simulation of Advanced Rockets throughout my graduate studies at Illinois. I thank the staff of the Computational Science and Engineering department and the Civil and Environmental Engineering department for their help with various issues that evolved from time to time.

I cannot thank my family enough for all they have done for me. Without their care and nurturing, I would not be where I am today. I thank my parents, Dr. Prem Prakash and Urmila Gupta, who always put the good of their children ahead of their own and provided us with the best education they could. I thank my sister, Pratibha Gupta, and my brother, Ravi Prakash, for their love and support. While in Urbana-Champaign, my cousins Dr. Abhay Vardhan, Dr. Varsha Vardhan and their daughter Mridula also provided me with the comfort and security of a family away from home.

Last but not the least, I thank all my colleagues and close friends, especially Kapil Dev, Ashish Jagmohan, Akhil Jain, Nitin Pande, Sanya Johnson, Ghadir Haikal, Kalyanababu Nakshatrala and Siddhartha Narra with whom I shared the ups and downs of life over the last few years.

Table of Contents

List of Figures

Chapter 1

Introduction

Dynamics is regarded as the study of objects and systems whose behavior changes with time. Such systems can be found everywhere, from the smallest sub-atomic scale, governing the behavior of minute particles, to the grand celestial scale, describing the evolution of the universe itself. Dynamics of physical systems has been an active research area for several decades, perhaps, even centuries. Despite significant advances, understanding the dynamics of most practical systems remains a challenge to this day.

Structural dynamics is the study of time varying response of everyday structures under dynamic loads. Common examples of such structures are buildings, bridges, dams etc. but structures can also span a wide range of length scales. Structures can be as small as micro-electro-mechanical devices (MEMs) or computer chips or even biological cells or as large as mountain ranges or tectonic plates of the earth's crust. Even though the governing differential equations of structural dynamics are well established, obtaining exact closed form solutions for the dynamic behavior of most structures is usually not possible. Researchers have come up with different ways to gain insight into the dynamics of structures by approximating them with idealized systems. Such idealizations usually involve replacing the continuous structure with a representative *discrete* structure.

The finite element method (FEM) is the most widely used tool for discretization. The basic idea of FEM is to replace a continuous structure with a collection of *elements* which, when *assembled* together into a *mesh*, closely approximate the original structure. Within each element, the solution is assumed to be an linear combination of certain *shape* functions. The unknown coefficients of these shape functions represent the degrees of freedom (DOFs)

of the discretized structure and comprise the solution to be computed.

For linear structural dynamics, the solution is time dependent and is obtained from the familiar equation of motion:

$$\boldsymbol{M}\,\ddot{\boldsymbol{U}}(t) + \boldsymbol{D}\,\dot{\boldsymbol{U}}(t) + \boldsymbol{K}\,\boldsymbol{U}(t) = \boldsymbol{P}(t) \tag{1.1}$$

where $\boldsymbol{U}(t)$, $\dot{\boldsymbol{U}}(t)$ and $\ddot{\boldsymbol{U}}(t)$ represent the nodal displacements, velocities and accelerations respectively at time t. $\boldsymbol{M}$, $\boldsymbol{D}$ and $\boldsymbol{K}$ denote the mass, damping and stiffness matrices of the structure respectively and $\boldsymbol{P}$ is a vector representing loads on the structure. The initial state of the structure along with appropriate boundary conditions need to be specified to complete the problem statement.

Structural dynamics is primarily studied with two methods of analysis, namely, the frequency domain and the time domain. Frequency domain analysis is commonly used when one is studying the long term *steady state* behavior of a structure under a sustained smooth loading. For instance, problems involving wind induced loading on high-rise buildings or bridges, or machine induced vibrations on a structure, are better suited for frequency domain analysis. However, for cases where the *initial transient* response of structure is critical to the solution, one must conduct a time domain analysis. Examples of such problems are scenarios involving impact, blast or crash loads on structures. Time domain analysis is also better suited for problems that are highly non-linear.

Analysis in the time domain is usually conducted by direct numerical time integration of the 2nd order system of ordinary differential equations (ODEs) (1.1). Most popular time integration methods in the literature are usually based on finite difference (FD) which leads to *time stepping*. One may convert the system (1.1) to a 1st order system with twice the number of unknowns and use 1st order methods such as the Euler method or Runge-Kutta method to integrate the same. However, methods such as the Newmark family of schemes that integrate 2nd order ODEs directly are preferred over the 1st order integrators. Other

methods based on variational formulations in time that lead to time-finite-elements or space-time elements are also continually being researched in the literature.

Time-stepping schemes, usually, advance a known solution at some instant of time t_i by a small increment Δt, called a time step. This is achieved by enforcing the equation of motion (1.1), with displacement, velocity and acceleration as the unknowns, at $t = t_{i+1}$. Difference formulas are used to express two of the unknowns in terms of the third which is then solved from the resulting equation. The three unknowns are updated to reflect the state at t_{i+1} and the process is repeated successively to obtain the response of the structure at following instants of time.

1.1 Implicit vs. Explicit Time Integration

Time-stepping schemes can broadly be classified into two categories namely, explicit and implicit. Most commercial codes use either explicit or implicit time integration exclusively with a uniform time step for the entire mesh. A comparison of the two methods with respect to their computational characteristics such as solution time per time step, stability and accuracy can serve as a basis for this choice.

Implicit time integration usually involves the solution a system of equations at each time step. For the explicit schemes, the system to be solved is usually the diagonal mass matrix, which can be inverted very easily. If the number of degrees of freedom in the problem are n then the computational effort required per time step is usually proportional to n^α. For implicit methods, α takes a value between 2 and 3 depending upon the type of solver, whereas an explicit system can be solved in time proportional to n ($\alpha = 1$). Thus, for each time step, implicit schemes, in general, require more computational effort than explicit schemes.

This advantage of the explicit methods is lost when comparing stability characteristics. A method is said to be stable if the computed solution remains bounded at all times. Explicit methods, in general, have a stringent requirement, governed by the Courant condition, on

the maximum allowable time step for stability:

$$\Delta t \leq \Delta t_{cr} = \frac{L}{c} \tag{1.2}$$

where L is a characteristic length of the *smallest* element in the mesh and c denotes the wave speed in the material. Clearly, the time step for the entire mesh is restricted by the size of the smallest element which is a severe disadvantage. Implicit methods, on the other hand, are usually stable, even for relatively large time steps and some members of the implicit family are actually *unconditionally* stable.

Accuracy is the ability of a method to replicate the exact the solution to within certain quantifiable error. In general, if the error tends to zero as $\Delta t \to 0$, the method is said to be *consistent*. For instance if the *trunctation* error $\boldsymbol{\tau}$ at any time t can be expressed as:

$$|\boldsymbol{\tau}(t)| \leq c\Delta t^k \tag{1.3}$$

then the scheme is consistent and together with stability, this results in accuracy. k is said to be the order of accuracy or the the rate of convergence. The explicit central difference and the implicit constant average acceleration are two of the most commonly used schemes in practice and are both 2nd order accurate. Other schemes with higher order of accuracy have also been developed in the literature and are currently being researched.

1.2 Motivation

As mentioned previously, almost all time-stepping schemes are formulated using a uniform time-step for the entire mesh. For large-scale nonlinear problems with complex geometries such as crash, impact or blast analysis, the range of element sizes in a mesh usually varies over several orders of magnitude. Certain parts of the mesh may contain very small elements, perhaps to capture high stress gradients while large parts of the mesh may still be relatively

coarse. Using an explicit or an implicit scheme exclusively with a uniform time-step, for such problems, is computationally very inefficient. If one were to use an explicit scheme, the time-step would be restricted by the size of the smallest element in the mesh and it would take a very large number of steps to compute the response of the structure for the desired interval of time. On the other hand, using an implicit scheme with a large time-step, one would not be able to capture, accurately, the response in regions of the mesh with high gradients with respect to time.

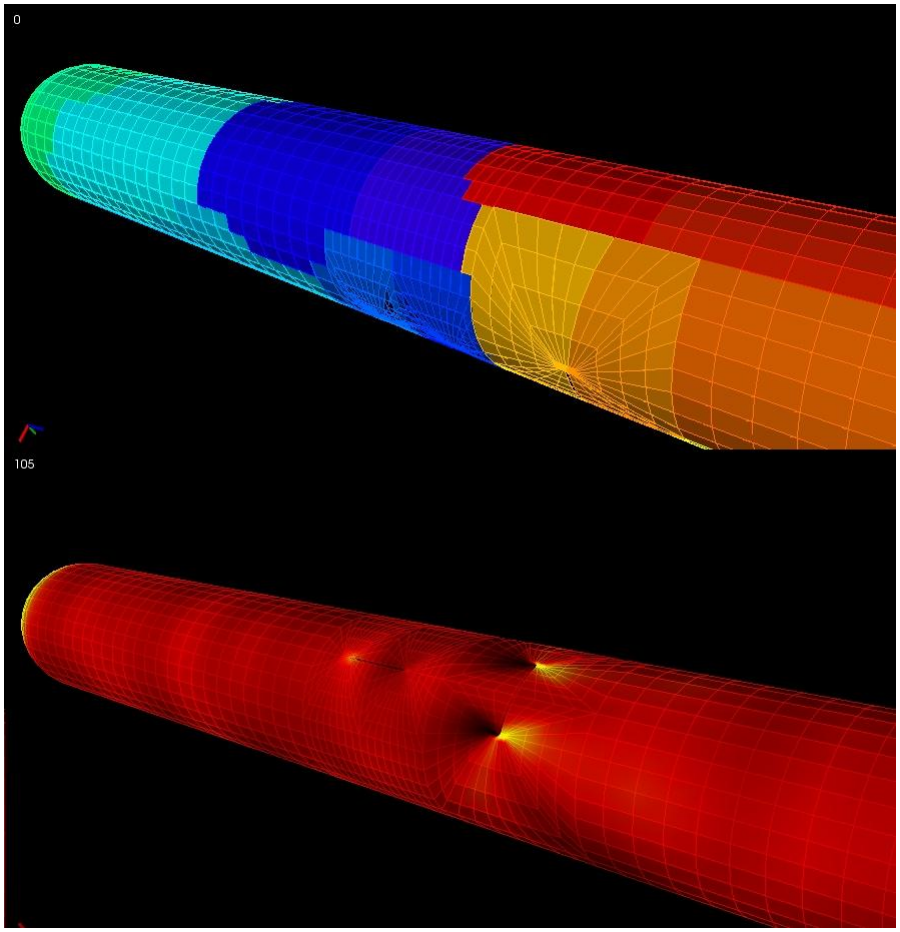

Figure 1.1: Axial stress response of a rocket casing with cracks to sudden head-end pressure; Analyzed with 32 subdomains partitioned with METIS; Coupled with recursive multi-time-step method with a time step ratio 10.

An intuitive solution to this problem is to integrate different parts of the mesh with time-steps and stepping schemes that are appropriate for the requirements of stability and

accuracy within the particular regions of the mesh. Researchers have experimented with several methods to accomplish this goal with limited success. A promising approach adopted in the present work uses domain decomposition to divide a large finite element mesh into several smaller parts called subdomains. The decomposition is done to ensure that each part contains elements with similar requirements with respect to stability and accuracy. The different subdomains are then solved as independent problems separately, possibly in parallel, and the individual solutions are coupled together to ensure continuity of the global solution across interface boundaries between the subdomains. It is shown that the present approach enables the use of different time-steps and stepping schemes in different parts of the mesh and still preserves the accuracy and stability of the individual subdomains with minimal computational overhead associated with coupling.

Chapter 2

Formulation

The dynamic behavior of a structure is governed by well known partial differential equations describing the kinematics of deformation, point-wise equilibrium of the body and the constitutive material response along with appropriate boundary and initial conditions. Texts on continuum and structural mechanics [1, 2, 3, 4] provide details of the formulation presented briefly here.

2.1 Kinematics

In order to formulate the equations describing the time dependent deformation of a structure, consider a body that occupies a region $\Omega(t)$ with boundary $\Gamma(t)$ at the current instant of time t as shown in Figure 2.1. Its initial configuration (usually undeformed) at time $t = t_0$, $\Omega(t_0) = \Omega_0 \subseteq \mathbb{R}^d$ where d is number of spatial dimensions of the body, is assumed known. Let the time interval of interest be $I = [t_0, t_F]$ and the current time $t \in I$. The motion of each point $\boldsymbol{X}$ in Ω_0 is completely described through a time dependent mapping $\boldsymbol{x} = \boldsymbol{\phi}(\boldsymbol{X}, t)$ from the initial configuration to the current configuration. The inverse map gives $\boldsymbol{X} = \boldsymbol{\phi}^{-1}(\boldsymbol{x}, t)$. Note that the same co-ordinate system has been chosen to describe vectors in both, initial and current, configurations. Standard notation, denoting quantities in the initial configuration with upper case letters and quantities in the current configuration with lower case letters, has been adopted.

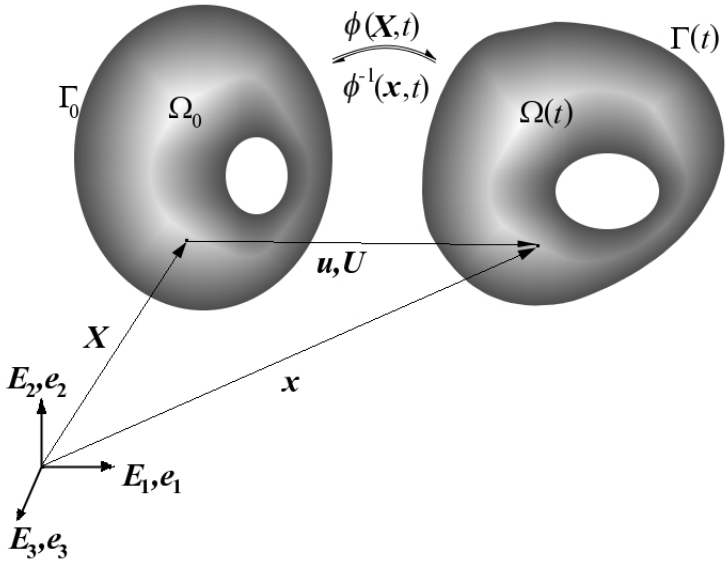

Figure 2.1: Kinematics of deformation.

The displacement in the current configuration is given by

$$\boldsymbol{U}(\boldsymbol{X},t) = \boldsymbol{x} - \boldsymbol{X} = \boldsymbol{\phi}(\boldsymbol{X},t) - \boldsymbol{X} \quad = \quad \boldsymbol{u}(\boldsymbol{x},t) = \boldsymbol{x} - \boldsymbol{X} = \boldsymbol{x} - \boldsymbol{\phi}^{-1}(\boldsymbol{x},t) \tag{2.1}$$

The velocity is defined as the time derivative of the map $\boldsymbol{\phi}$ and is expressed in material and spatial co-ordinates as:

$$\begin{aligned} \boldsymbol{V}(\boldsymbol{X},t) = \dot{\boldsymbol{\phi}}(\boldsymbol{X},t) = \frac{d\boldsymbol{\phi}(\boldsymbol{X},t)}{dt} = \frac{\partial \boldsymbol{\phi}(\boldsymbol{X},t)}{\partial t} = \frac{\partial \boldsymbol{U}(\boldsymbol{X},t)}{\partial t} = \frac{d\boldsymbol{U}(\boldsymbol{X},t)}{dt} = \dot{\boldsymbol{U}}(\boldsymbol{X},t) \\ = \boldsymbol{v}(\boldsymbol{x},t) = \dot{\boldsymbol{u}}(\boldsymbol{x},t) = \frac{d\boldsymbol{u}(\boldsymbol{x},t)}{dt} = \frac{\partial \boldsymbol{u}(\boldsymbol{x},t)}{\partial \boldsymbol{x}} \frac{\partial \boldsymbol{x}(\boldsymbol{X},t)}{\partial t} + \frac{\partial \boldsymbol{u}(\boldsymbol{x},t)}{\partial t} = \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}} \boldsymbol{v} + \frac{\partial \boldsymbol{u}}{\partial t} \end{aligned} \tag{2.2}$$

respectively. Similarly acceleration is defined as the time derivatives of the velocity:

$$\begin{aligned} \boldsymbol{A}(\boldsymbol{X},t) = \dot{\boldsymbol{V}}(\boldsymbol{X},t) = \frac{d\boldsymbol{V}(\boldsymbol{X},t)}{dt} = \frac{\partial \boldsymbol{V}(\boldsymbol{X},t)}{\partial t} = \ddot{\boldsymbol{U}}(\boldsymbol{X},t) = \ddot{\boldsymbol{\phi}}(\boldsymbol{X},t) \\ = \boldsymbol{a}(\boldsymbol{x},t) = \dot{\boldsymbol{v}}(\boldsymbol{x},t) = \frac{\partial \boldsymbol{v}(\boldsymbol{x},t)}{\partial \boldsymbol{x}} \frac{\partial \boldsymbol{x}(\boldsymbol{X},t)}{\partial t} + \frac{\partial \boldsymbol{v}(\boldsymbol{x},t)}{\partial t} = \frac{\partial \boldsymbol{v}}{\partial \boldsymbol{x}} \boldsymbol{v} + \frac{\partial \boldsymbol{v}}{\partial t} = \ddot{\boldsymbol{u}}(\boldsymbol{x},t) \end{aligned} \tag{2.3}$$

Note that $\boldsymbol{u} = \boldsymbol{U} \circ \boldsymbol{\phi}^{-1}$, $\boldsymbol{v} = \boldsymbol{V} \circ \boldsymbol{\phi}^{-1}$ and $\boldsymbol{a} = \boldsymbol{A} \circ \boldsymbol{\phi}^{-1}$ where $\circ$ denotes the composition of functions.

An important measure of the deformation is the deformation gradient, defined by:

$$\boldsymbol{F}(\boldsymbol{X},t)=\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{X}} \qquad \Rightarrow \qquad F_{iI}(\boldsymbol{e}_i\otimes\boldsymbol{E}_I)=\frac{\partial x_i}{\partial X_I}(\boldsymbol{e}_i\otimes\boldsymbol{E}_I) \tag{2.4}$$

where the summation convention is implied on the repeated indices i and I ($i, I \in [1, d]$) and $\otimes$ denotes the tensor product of two vectors. The explicit dependence of quantities on $\boldsymbol{X}$ and t will be dropped henceforth for brevity. The polar decomposition of $\boldsymbol{F}$, ($\boldsymbol{F}=\boldsymbol{F}^R\,\boldsymbol{F}^U=\boldsymbol{F}^V\boldsymbol{F}^R$), shows that it contains information about both, the stretching ($\boldsymbol{F}^U$, $\boldsymbol{F}^V$) and rigid body rotation ($\boldsymbol{F}^R$), components of the deformation. In order to obtain a measure of strain that is independent of rigid body motion, the right and left Cauchy-Green deformation tensors is defined as:

$$\boldsymbol{C}=\boldsymbol{F}^T\boldsymbol{F}=(\boldsymbol{F}^U)^2 \qquad ; \qquad \boldsymbol{b}=\boldsymbol{F}\boldsymbol{F}^T=(\boldsymbol{F}^V)^2 \tag{2.5}$$

respectively. The corresponding Green-Lagrangian strain and Almansi-Eulerian strain tensors are defined as:

$$\boldsymbol{E}=\frac{1}{2}(\boldsymbol{C}-\boldsymbol{I}) \qquad ; \qquad \boldsymbol{e}=\frac{1}{2}(\boldsymbol{I}-\boldsymbol{b}^{-1}) \tag{2.6}$$

2.2 Balance Laws and Equilibrium

Fundamental balance laws include conservation of mass, momentum and energy. The balance of linear and angular momentum leads to structural equilibrium. A configuration is said to be in equilibrium if the sum of internal and external forces and moments is zero at all points of the body in that configuration. The internal forces in a body are characterized through stress. A convenient expression for stress at a point $\boldsymbol{x}$ in the current configuration is the Cauchy stress tensor $\boldsymbol{\sigma}(\boldsymbol{x})$. The traction (force per unit area) $\boldsymbol{t_n}$ acting on a plane normal to the unit vector $\boldsymbol{n}$ and the Cauchy stress $\boldsymbol{\sigma}$, at a point $\boldsymbol{x}$ in the current configuration, are

related through the Cauchy formula:

$$\boldsymbol{t_n} = \boldsymbol{\sigma n} \qquad \text{where} \qquad \boldsymbol{\sigma} = \sigma_{ij}(\boldsymbol{e}_i \otimes \boldsymbol{e}_j) \tag{2.7}$$

If the body $\Omega(t)$ is in equilibrium, all possible subregions $\Omega_t^s \subset \Omega(t)$ must also be in equilibrium:

$$\begin{aligned} &\int_{\Gamma_t^s} \boldsymbol{t_n}\, da \; + \int_{\Omega_t^s} (-\rho \ddot{\boldsymbol{u}} + \bar{\boldsymbol{b}})\, dv = 0 \\ \Rightarrow \qquad &\int_{\Omega_t^s} (-\rho \ddot{\boldsymbol{u}} + \text{div}\boldsymbol{\sigma} + \bar{\boldsymbol{b}})\, dv = 0 \end{aligned} \tag{2.8}$$

where Γ_t^s denotes the boundary of the subregion Ω_t^s, ρ is the density, $\bar{\boldsymbol{b}}$ is the body force per unit volume and $-\rho\ddot{\boldsymbol{u}}$ is the inertial force obtained using the *d'Alembert's* principle. Since equation (2.8) must hold for all possible subregions Ω_t^s, the instantaneous residual equations for dynamic equilibrium in the current configuration is given by:

$$\boldsymbol{r} = -\rho\ddot{\boldsymbol{u}} + \text{div}\boldsymbol{\sigma} + \bar{\boldsymbol{b}} = \boldsymbol{0} \qquad \forall \boldsymbol{x} \in \Omega(t) \tag{2.9}$$

where the divergence (div) is taken with respect to the current coordinates $\boldsymbol{x}$. A similar equilibrium of moments leads to symmetry of the Cauchy stress $\boldsymbol{\sigma} = \boldsymbol{\sigma}^T$.

Alternate measures of stress defined on the initial configuration are the first and second Piola-Kirchhoff stress tensors $\boldsymbol{P}$ and $\boldsymbol{S}$. The first Piola-Kirchhoff stress tensor is defined such that the traction $\boldsymbol{T_N}$ at a point $\boldsymbol{X}$ in the initial configuration, obtained from the Cauchy relationship $\boldsymbol{T_N} = \boldsymbol{P}\,\boldsymbol{N}$, satisfies:

$$\boldsymbol{T_N}\, dA = \boldsymbol{t_n}\, da \qquad \Rightarrow \qquad \boldsymbol{P}\,\boldsymbol{N}\, dA = \boldsymbol{\sigma n}\, da \tag{2.10}$$

where dA is an infinitesimal area in the initial configuration and da is the corresponding mapped area in the current configuration. The oriented areas $\boldsymbol{N}dA$ and $\boldsymbol{n}da$ are obtained

as vector products: $(\boldsymbol{N}_1 dS_1) \times (\boldsymbol{N}_2 dS_2)$ and $(\boldsymbol{n}_1 ds_1) \times (\boldsymbol{n}_2 ds_2)$ respectively. The infinitesimal vectors are transformed as $\boldsymbol{n}_1 ds_1 = \boldsymbol{F}\boldsymbol{N}_1 dS_1$ and $\boldsymbol{n}_2 ds_2 = \boldsymbol{F}\boldsymbol{N}_2 dS_2$ and the area transformation is given by:

$$\begin{aligned} \boldsymbol{n}\, da &= (\boldsymbol{n}_1 ds_1) \times (\boldsymbol{n}_2 ds_2) \\ &= (\boldsymbol{F}\boldsymbol{N}_1 dS_1) \times (\boldsymbol{F}\boldsymbol{N}_2 dS_2) = J\boldsymbol{F}^{-T}(\boldsymbol{N}_1 dS_1) \times (\boldsymbol{N}_2 dS_2) = J\boldsymbol{F}^{-T}\boldsymbol{N}\, dA \end{aligned} \tag{2.11}$$

where $J = \det(\boldsymbol{F})$. Using the above relations one obtains the definition of the first Piola-Kirchhoff stress tensor as:

$$\boldsymbol{P} = J\boldsymbol{\sigma}\boldsymbol{F}^{-T} \tag{2.12}$$

which is not symmetric. A symmetric measure of stress in the initial configuration is the second Piola-Kirchhoff stress tensor, defined as:

$$\boldsymbol{S} = J\boldsymbol{F}^{-1}\boldsymbol{\sigma}\boldsymbol{F}^{-T} \tag{2.13}$$

Equation (2.9) written in terms of quantities defined on the initial configuration is:

$$\boldsymbol{r}_0 = -\rho_0 \ddot{\boldsymbol{U}} + \text{DIV}\boldsymbol{P} + \bar{\boldsymbol{B}} = \boldsymbol{0} \qquad \forall \boldsymbol{X} \in \Omega_0 \tag{2.14}$$

where $\rho_0 = \rho J$ is the initial density and $\bar{\boldsymbol{B}} = \bar{\boldsymbol{b}} J$ is equivalent body force in initial configuration and the two residuals are related as $\boldsymbol{r}_0 = \boldsymbol{r} J$. The divergence (DIV) is taken with respect to initial co-ordinates $\boldsymbol{X}$.

2.3 Constitutive Relationships

The behavior of materials is governed through constitutive relationships, which in general, relate a measure of material response to applied stimuli, such as relating strain to stress. As yet, it has not been possible to characterize the behavior of any material purely

from first principles. Physicists have been actively engaged in state-of-the-art research to formulate a universal theory to describe the behavior of all matter, space and time but have not yet succeeded. Till the time such a theory can be realized, if ever, researchers will continue to characterize material response through empirical relationships based on observed phenomena. Since an unimaginable number of factors might possibly affect material response, such empirical models are usually restricted to certain range of behavior and applied stimuli represented through internal variables. Important material models that characterize material behavior for mechanical problems coupled with a variety of physical phenomena include thermo-mechanical, electro-mechanical, photo-mechanical and magneto-rheological models. Physical phenomena such as fracture and phase-change are also modeled in constitutive theories. It must be pointed out that any constitutive model must not contradict any fundamental balance law for the quantities that it is trying to model. For instance, a thermo-mechanical material model must not violate the second law of thermo-dynamics. In general, the problem of ensuring consistency of a constitutive model with governing balance laws remains an open problem.

In structural mechanics we are primarily concerned with stress-strain constitutive relationships. Basic principles of material behavior such as *determinism*, *locality* and *frame-indifference* are widely accepted [5, 6, 7]. Haupt [8] classifies constitutive theories into four categories:

- *Rate-independent Elasticity* includes Euler-perfect fluids and isotropic-linear-elastic solids.
- *Rate-independent Plasticity* includes elastic-perfectly-plastic solids.
- *Rate-dependent Elasticity* includes Newton-linear-viscous fluids and visco-elastic solids.
- *Rate-dependent Plasticity* includes visco-plastic solids.

It is, however, conceivable that rate-dependent models would encompass rate-independent models and that elasticity may be treated as a special case of plasticity. In this scenario, rate-dependent plasticity may be generalized to cover all classical constitutive theories.

2.3.1 Hyperelasticity

A simple rate-independent elastic constitutive theory is *hyperelasticity*. A material is said to be hyperelastic if there exists a strain energy density function $\hat{\Psi}$ which depends only on the initial and current configurations, Ω_0 and $\Omega(t)$, of the body and *not* on the actual *path* of the deformation process. In this case, the work done by the stresses at a material point $\boldsymbol{X}$ (stored as strain energy) can be expressed as:

$$\begin{aligned} & \hat{\Psi}(\boldsymbol{F}(\boldsymbol{X},t),\boldsymbol{X}) = \int_{t_0}^{t} \boldsymbol{P}(\boldsymbol{F}(\boldsymbol{X},t),\boldsymbol{X}) : \dot{\boldsymbol{F}}(\boldsymbol{X},t)\, dt \\ \Rightarrow \quad & \dot{\hat{\Psi}}(\boldsymbol{F}(\boldsymbol{X},t),\boldsymbol{X}) = \boldsymbol{P}(\boldsymbol{F}(\boldsymbol{X},t),\boldsymbol{X}) : \dot{\boldsymbol{F}}(\boldsymbol{X},t) \\ \Rightarrow \quad & \boldsymbol{P}(\boldsymbol{F}(\boldsymbol{X},t),\boldsymbol{X}) = \frac{\partial \hat{\Psi}(\boldsymbol{F}(\boldsymbol{X},t),\boldsymbol{X})}{\partial \boldsymbol{F}} \qquad \Rightarrow \qquad P_{iI} = \frac{\partial \hat{\Psi}}{\partial F_{iI}} \end{aligned} \tag{2.15}$$

Alternatively, the strain energy can be expressed in terms of the second Piola Kirchhoff stress tensor $\boldsymbol{S}$ and the Cauchy-Green deformation tensor $\boldsymbol{C}$ or the Lagrangian strain tensor $\boldsymbol{E}$, signifying its independence from rigid body rotation $\boldsymbol{F}^R$:

$$\begin{aligned} & \hat{\Psi}(\boldsymbol{F}) = \Psi(\boldsymbol{C}) = \int_{t_0}^{t} \frac{1}{2}\boldsymbol{S} : \dot{\boldsymbol{C}}\, dt = \int_{t_0}^{t} \boldsymbol{S} : \dot{\boldsymbol{E}}\, dt = \bar{\Psi}(\boldsymbol{E}) \\ \Rightarrow \quad & \dot{\Psi} = \frac{1}{2}\boldsymbol{S} : \dot{\boldsymbol{C}} = \boldsymbol{S} : \dot{\boldsymbol{E}} = \dot{\bar{\Psi}}(\boldsymbol{E}) \\ \Rightarrow \quad & \boldsymbol{S} = 2\frac{\partial \Psi(\boldsymbol{C})}{\partial \boldsymbol{C}} = \frac{\partial \bar{\Psi}(\boldsymbol{E})}{\partial \boldsymbol{E}} \end{aligned} \tag{2.16}$$

where the explicit dependence on $\boldsymbol{X}$ and t has been dropped for brevity. In addition to the stress-strain relationship, the rate of change of stress with strain is also needed, as we shall see in section §2.7. This rate is quantified by the fourth order *elasticity* tensor $\mathcal{C}$ defined as:

$$\mathcal{C} = \frac{\partial \boldsymbol{S}}{\partial \boldsymbol{E}} = \frac{\partial^2 \bar{\Psi}}{\partial \boldsymbol{E}\, \partial \boldsymbol{E}} \tag{2.17}$$

A further simplification of hyperelasticity is *isotropy*, which states that material properties are independent of direction. The strain energy density function of an isotropic hypere-

lastic material is usually expressed in terms of the invariants of $\boldsymbol{C}$ as $\Psi(I_C, II_C, III_C)$ where $I_C = \mathrm{tr}(\boldsymbol{C})$ is the first invariant, $II_C = \mathrm{tr}(\boldsymbol{C}^2)$ is the second invariant and $III_C = \det(\boldsymbol{C}) = J^2$ is the third invariant. An example of a *linear* hyperelastic isotropic material model is the St. Venant-Kirchhoff model:

$$\begin{aligned} \bar{\Psi}(\boldsymbol{E}) &= \frac{1}{2}\lambda(\mathrm{tr}(\boldsymbol{E}))^2 + \mu \mathrm{tr}(\boldsymbol{E}^2) \\ \Rightarrow \qquad \boldsymbol{S} &= \lambda(\mathrm{tr}(\boldsymbol{E}))\boldsymbol{I} + 2\mu\boldsymbol{E} \\ \Rightarrow \qquad \mathcal{C} &= \lambda\, \boldsymbol{I} \otimes \boldsymbol{I} + 2\mu\, \mathcal{I} \end{aligned} \tag{2.18}$$

where λ and μ are the Lamé parameters. $\mathcal{I}$ and $\boldsymbol{I} \otimes \boldsymbol{I}$ are fourth order tensors with components $\mathcal{I}_{IJKL} = \delta_{IK}\delta_{JL}$ and $(\boldsymbol{I} \otimes \boldsymbol{I})_{IJKL} = \delta_{IJ}\delta_{KL}$ respectively.

Often one encounters situations that require modeling of *incompressible* materials. An incompressible deformation is characterized by the restriction $J = 1$. In such cases it is helpful to split the Cauchy-Green deformation tensor into volumetric and deviatoric parts:

$$\boldsymbol{C} = \boldsymbol{C}^{\mathrm{VOL}}\boldsymbol{C}^{\mathrm{DEV}} = (J^{\frac{2}{3}}\boldsymbol{I})(J^{-\frac{2}{3}}\boldsymbol{C}) \tag{2.19}$$

and note that $\det(\boldsymbol{C}^{\mathrm{DEV}}) = 1$. Now the strain energy density function can be expressed in terms of the deviatoric part only:

$$\begin{aligned} \Psi'(\boldsymbol{C}) &= \Psi(\boldsymbol{C}^{\mathrm{DEV}}) \\ \Rightarrow \quad \boldsymbol{S} &= \frac{\partial \Psi'}{\partial \boldsymbol{C}} + pJ\boldsymbol{C}^{-1} \end{aligned} \tag{2.20}$$

where p is an independent variable denoting hydrostatic pressure and is *not* determinable through constitutive relationships alone. p is determined from minimization of the energy subject to the constraint $J = 1$.

An example of a model used for incompressible rubbers is the generalized Mooney-Rivlin-

Ogden material:

$$\Psi(\boldsymbol{C}) = \sum_{\alpha=0}^{M}\sum_{\beta=0}^{N}\mu_{\alpha\beta}(I_C - 3)^{\alpha}(I_C^2 - II_C - 3)^{\beta} \tag{2.21}$$

where $\mu_{\alpha\beta}$ are positive constants. A special case when $M = 1$ and $N = 0$ is called Neo-Hookean material.

$$\Psi(\boldsymbol{C}) = \frac{\mu}{2}(I_C - 3) \tag{2.22}$$

For compressible and *almost* incompressible materials, a convex function U that depends only on the volumetric part is added:

$$\Psi(\boldsymbol{C}) = \Psi'(\boldsymbol{C}) + U(J) \tag{2.23}$$

An example of a compressible Neo-Hookean material model is:

$$\begin{aligned} \Psi(\boldsymbol{C}) &= \frac{\mu}{2}(I_C - 3) - \mu\ln(J) + \frac{\lambda}{2}(\ln(J))^2 \\ \Rightarrow \qquad \boldsymbol{S} &= \mu(\boldsymbol{I} - \boldsymbol{C}^{-1}) + \lambda\ln(J)\,\boldsymbol{C}^{-1} \\ \Rightarrow \qquad \mathcal{C}_{IJKL} &= \lambda\boldsymbol{C}^{-1}_{IJ}\boldsymbol{C}^{-1}_{KL} + 2(\mu - \lambda\ln J)\boldsymbol{C}^{-1}_{IK}\boldsymbol{C}^{-1}_{JL} \end{aligned} \tag{2.24}$$

2.3.2 Small-strain Rate-independent Elasto-plasticity

For modeling metals, researchers have traditionally used *small-strain* plasticity with an additive split of the strain into an elastic and plastic part ($\boldsymbol{\varepsilon} = \boldsymbol{\varepsilon}^e + \boldsymbol{\varepsilon}^p$). Since only linearized strains are considered, the distinction between stresses in different configuration is dropped. The *yield* function for an elasto-plastic material model with isotropic-kinematic hardening is defined as:

$$f(\boldsymbol{\sigma}, \alpha, \boldsymbol{\beta}) = \|\boldsymbol{\xi}\| - (\sigma_Y + K_1\alpha) \tag{2.25}$$

where σ_Y denotes the uniaxial yield stress, K_1 is a positive constant, α and $\boldsymbol{\beta}$ are internal variables representing effective plastic strain and back stress respectively. $\boldsymbol{\xi} = \boldsymbol{\sigma}' - \boldsymbol{\beta}$ where

$\boldsymbol{\sigma}'$ is the deviatoric stress $\boldsymbol{\sigma} - \frac{1}{3}(\mathrm{tr}(\boldsymbol{\sigma}))\boldsymbol{I}$. The evolution equations are given by:

$$\dot{\boldsymbol{\varepsilon}}^p = \gamma \frac{\boldsymbol{\xi}}{\|\boldsymbol{\xi}\|} = \gamma \frac{\partial f}{\partial \boldsymbol{\sigma}} \tag{2.26}$$

$$\dot{\alpha} = \gamma \tag{2.27}$$

$$\dot{\boldsymbol{\beta}} = \gamma\, H\, \frac{\boldsymbol{\xi}}{\|\boldsymbol{\xi}\|} \tag{2.28}$$

$$\dot{\boldsymbol{\sigma}} = \mathcal{C}\, \dot{\boldsymbol{\varepsilon}}^e = \mathcal{C}\, (\dot{\boldsymbol{\varepsilon}} - \dot{\boldsymbol{\varepsilon}}^p) \tag{2.29}$$

where $\gamma \geq 0$ and $f \leq 0$ implying the Kuhn-Tucker *complementarity* condition $\gamma f = 0$. In addition, we have the *persistency* condition $\gamma \dot{f} = 0$ which states that for continual plastic loading, stress must be on the yield surface.

2.4 Initial Boundary Value Problem

The kinematics, equilibrium and constitutive relationships together result in a system of partial differential equations governing the behavior of a structural domain. To complete the problem statement, either an applied traction or an imposed displacement needs to be specified at each point of the boundary of the entire domain. The imposed displacement and applied traction boundary conditions are known as Dirichlet and Neumann boundary conditions respectively.

Let the boundary $\Gamma(t)$ at any instant of time t be divided into two parts $\Gamma_D(t)$ and $\Gamma_N(t)$, denoting the Dirichlet and Neumann boundaries respectively, such that $\Gamma_D(t) \cup \Gamma_N(t) = \Gamma(t)$ and $\Gamma_D(t) \cap \Gamma_N(t) = \phi$, the null set. The respective boundary conditions can be specified as:

$$\boldsymbol{\phi}(\boldsymbol{X}, t) = \bar{\boldsymbol{\phi}}(\boldsymbol{X}, t) \qquad \forall\, t \in I \text{ and } \forall\, \boldsymbol{X} \in \Gamma_D(0) \tag{2.30}$$

$$\boldsymbol{t_n}(\boldsymbol{x}, t) = \bar{\boldsymbol{t}}(\boldsymbol{x}, t) \qquad \forall\, t \in I \text{ and } \forall\, \boldsymbol{x} \in \Gamma_N(t) \tag{2.31}$$

where $\bar{\boldsymbol{\phi}}$ and $\bar{\boldsymbol{t}}$ are given, and $\boldsymbol{n}$ is the outward normal to the Neumann boundary $\Gamma_N(t)$.

Finally, the initial boundary value problem of structural dynamics can be stated as follows: Given the initial configuration Ω_0 of a structure, find the map $\boldsymbol{\phi}(\boldsymbol{X}, t)$ that satisfies, the kinematic strain-displacement relationships in section § 2.1, equilibrium equation (2.9) or (2.14), the chosen constitutive relationships and boundary conditions (2.30) and (2.31).

2.5 Virtual Work and Weak Form

The equations presented in the preceding sections are known as the *strong* form. Classical solutions to the strong form of the equations are almost never available, even for very simple problems. A method for obtaining approximate solutions is facilitated by the *weak* form of the equations which is obtained by applying the fundamental theorem of calculus of variations. For instance, consider the spaces $\mathcal{V} = \{\boldsymbol{\phi} : \boldsymbol{\phi} \in H^1(\Omega(t)); \boldsymbol{\phi}|_{\Gamma_D(t)} = \bar{\boldsymbol{\phi}}\}$ and $\tilde{\mathcal{V}} = \{\tilde{\boldsymbol{u}} : \tilde{\boldsymbol{u}} \in H^1(\Omega(t)); \tilde{\boldsymbol{u}}|_{\Gamma_D(t)} = \boldsymbol{0}\}$ (see [9] for details). Define the functional $\mathscr{G}(\boldsymbol{\phi}, \tilde{\boldsymbol{u}})$ on the fields $\boldsymbol{\phi} \in \mathcal{V}$ and $\tilde{\boldsymbol{u}} \in \tilde{\mathcal{V}}$:

$$
\begin{aligned}
\mathscr{G}(\boldsymbol{\phi}, \tilde{\boldsymbol{u}}) &\equiv -\int_{\Omega(t)} \tilde{\boldsymbol{u}} \cdot \left(-\rho\ddot{\boldsymbol{u}} + \operatorname{div}\boldsymbol{\sigma} + \bar{\boldsymbol{b}}\right) dv + \int_{\Gamma_N(t)} \tilde{\boldsymbol{u}} \cdot (\boldsymbol{t_n} - \bar{\boldsymbol{t}})\, da \\
&= \int_{\Omega(t)} \tilde{\boldsymbol{u}} \cdot \rho\ddot{\boldsymbol{u}} - \left(\operatorname{div}(\boldsymbol{\sigma}^T\tilde{\boldsymbol{u}}) - \boldsymbol{\sigma} : \frac{\partial\tilde{\boldsymbol{u}}}{\partial\boldsymbol{x}}\right) - \tilde{\boldsymbol{u}} \cdot \bar{\boldsymbol{b}}\, dv + \int_{\Gamma_N(t)} \tilde{\boldsymbol{u}} \cdot (\boldsymbol{t_n} - \bar{\boldsymbol{t}})\, da \\
&= \int_{\Omega(t)} \tilde{\boldsymbol{u}} \cdot \rho\ddot{\boldsymbol{u}} + (\boldsymbol{\sigma} : \nabla_x\tilde{\boldsymbol{u}}) - \tilde{\boldsymbol{u}} \cdot \bar{\boldsymbol{b}}\, dv + \int_{\Gamma_N(t)} -\tilde{\boldsymbol{u}} \cdot (\boldsymbol{\sigma}\boldsymbol{n}) + \tilde{\boldsymbol{u}} \cdot (\boldsymbol{t_n} - \bar{\boldsymbol{t}})\, da \\
&= \int_{\Omega(t)} \tilde{\boldsymbol{u}} \cdot \rho\ddot{\boldsymbol{u}}\, dv + \int_{\Omega(t)} \boldsymbol{\sigma} : \nabla_x\tilde{\boldsymbol{u}}\, dv - \int_{\Omega(t)} \tilde{\boldsymbol{u}} \cdot \bar{\boldsymbol{b}}\, dv - \int_{\Gamma_N(t)} \tilde{\boldsymbol{u}} \cdot \bar{\boldsymbol{t}}\, da
\end{aligned}
\tag{2.32}
$$

The expressions in equation (2.32) are obtained using the product rule $\operatorname{div}(\boldsymbol{\sigma}^T\tilde{\boldsymbol{u}}) = \operatorname{div}(\boldsymbol{\sigma}) \cdot \tilde{\boldsymbol{u}} + \boldsymbol{\sigma} : \nabla_x\tilde{\boldsymbol{u}}$, the divergence theorem $\int_{\Omega(t)}\operatorname{div}(\boldsymbol{\sigma}^T\tilde{\boldsymbol{u}})\, dv = \int_{\Gamma(t)}(\boldsymbol{\sigma}^T\tilde{\boldsymbol{u}}) \cdot \boldsymbol{n}\, da$, symmetry of the Cauchy stress $\boldsymbol{\sigma} = \boldsymbol{\sigma}^T$ and the Cauchy relation $\boldsymbol{t_n} = \boldsymbol{\sigma}\boldsymbol{n}$.

A direct consequence of the fundamental theorem of calculus of variations is the principle of virtual work, which states if $\mathscr{G}(\boldsymbol{\phi}, \tilde{\boldsymbol{u}}) = 0$ for all $\tilde{\boldsymbol{u}} \in \tilde{\mathcal{V}}$ then $\boldsymbol{\phi}$ results in an equilibrium

configuration. The virtual work functional is equivalently defined as:

$$\mathscr{G}(\boldsymbol{\phi}, \tilde{\boldsymbol{u}}) = \int_{\Omega(t)} \tilde{\boldsymbol{u}} \cdot \rho \ddot{\boldsymbol{u}} \, dv + \mathscr{W}_I(\boldsymbol{\phi}, \tilde{\boldsymbol{u}}) - \mathscr{W}_E(\boldsymbol{\phi}, \tilde{\boldsymbol{u}}) \tag{2.33}$$

where $\mathscr{W}_E$ and $\mathscr{W}_I$ denote the external and internal virtual work respectively, defined as:

$$\mathscr{W}_E(\boldsymbol{\phi}, \tilde{\boldsymbol{u}}) = \int_{\Omega(t)} \tilde{\boldsymbol{u}} \cdot \bar{\boldsymbol{b}} \, dv + \int_{\Gamma_N(t)} \tilde{\boldsymbol{u}} \cdot \bar{\boldsymbol{t}} \, da \tag{2.34}$$

$$\mathscr{W}_I(\boldsymbol{\phi}, \tilde{\boldsymbol{u}}) = \int_{\Omega(t)} \tilde{\boldsymbol{\varepsilon}} : \boldsymbol{\sigma} \, dv \tag{2.35}$$

$\tilde{\boldsymbol{\varepsilon}}$ is the linearized (small) strain tensor corresponding to the virtual displacement $\tilde{\boldsymbol{u}}$:

$$\tilde{\boldsymbol{\varepsilon}} = \frac{1}{2}(\nabla_x \tilde{\boldsymbol{u}} + \nabla_x \tilde{\boldsymbol{u}}^T) \tag{2.36}$$

The virtual work functional $\mathscr{G}$ can also be expressed in terms of quantities in the initial configuration:

$$\mathscr{G}(\boldsymbol{\phi}, \tilde{\boldsymbol{U}}) = \int_{\Omega_0} \tilde{\boldsymbol{U}} \cdot \rho_0 \ddot{\boldsymbol{U}} \, dV + \mathscr{W}_I(\boldsymbol{\phi}, \tilde{\boldsymbol{U}}) - \mathscr{W}_E(\boldsymbol{\phi}, \tilde{\boldsymbol{U}}) \tag{2.37}$$

where the actual functional form of $\mathscr{G}(\boldsymbol{\phi}, \tilde{\boldsymbol{U}})$ may be different from $\mathscr{G}(\boldsymbol{\phi}, \tilde{\boldsymbol{u}})$ but, for simplicity, we will use the same symbol $\mathscr{G}$ for both (since $\tilde{\boldsymbol{U}}(\boldsymbol{X}) = \tilde{u}(\boldsymbol{x})$) and the distinction between them will be clear from context. The external virtual work is given by:

$$\mathscr{W}_E(\boldsymbol{\phi}, \tilde{\boldsymbol{U}}) = \int_{\Omega_0} \tilde{\boldsymbol{U}} \cdot \bar{\boldsymbol{B}}_0 \, dV + \int_{\Gamma_{N0}} \tilde{\boldsymbol{U}} \cdot \bar{\boldsymbol{T}}_0 \, dA \tag{2.38}$$

The internal work $\mathscr{W}_I$ can be defined in terms of the first Piola Kirchhoff stress tensor:

$$\begin{aligned} \mathscr{W}_I(\boldsymbol{\phi}, \tilde{\boldsymbol{U}}) &= \int_{\Omega_0} (J\boldsymbol{\sigma}\boldsymbol{F}^{-T}\boldsymbol{F}^T) : \left(\frac{\partial \tilde{\boldsymbol{U}}}{\partial \boldsymbol{X}} \frac{\partial \boldsymbol{X}}{\partial \boldsymbol{x}}\right) dV = \int_{\Omega_0} (\boldsymbol{P}\boldsymbol{F}^T) : \left(\frac{\partial \tilde{\boldsymbol{U}}}{\partial \boldsymbol{X}} \boldsymbol{F}^{-1}\right) dV \\ &= \int_{\Omega_0} \boldsymbol{P} : \frac{\partial \tilde{\boldsymbol{U}}}{\partial \boldsymbol{X}} \, dV \end{aligned} \tag{2.39}$$

or the second Piola Kirchhoff stress tensor:

$$\begin{aligned} \mathscr{W}_I(\boldsymbol{\phi}, \tilde{\boldsymbol{U}}) &= \int_{\Omega_0} \boldsymbol{P} : \frac{\partial \tilde{\boldsymbol{U}}}{\partial \boldsymbol{X}} \, dV = \int_{\Omega_0} (\boldsymbol{F}\boldsymbol{S}) : \tilde{\boldsymbol{F}} \, dV \\ &= \int_{\Omega_0} \tilde{\boldsymbol{E}} : \boldsymbol{S} \, dV \end{aligned} \tag{2.40}$$

where

$$\tilde{\boldsymbol{F}} = \nabla_X \tilde{\boldsymbol{U}} = \frac{\partial \tilde{\boldsymbol{U}}}{\partial \boldsymbol{X}} \tag{2.41}$$

$$\tilde{\boldsymbol{E}} = \frac{1}{2}(\boldsymbol{F}^T \tilde{\boldsymbol{F}} + \tilde{\boldsymbol{F}}^T \boldsymbol{F}) \tag{2.42}$$

Note that in equation (2.32), only the equation of equilibrium was enforced weakly. One can obtain alternate *mixed* formulations by enforcing any or all of the governing partial differential equations weakly.

2.6 Variational Methods and Energy Functionals

The virtual work functional can always be constructed from the strong form of any set of partial differential equations. An alternate statement of equilibrium can sometimes be obtained using energy principles. The kinetic and potential energies are defined as:

$$T(\dot{\boldsymbol{u}}) = \int_{\Omega(t)} \frac{1}{2} \rho (\dot{\boldsymbol{u}} \cdot \dot{\boldsymbol{u}}) \, dv \tag{2.43}$$

$$V(\boldsymbol{u}) = \int_{\Omega(t)} \boldsymbol{\sigma} : \boldsymbol{\varepsilon} - \boldsymbol{u} \cdot \bar{\boldsymbol{b}} \, dv - \int_{\Gamma_N(t)} \boldsymbol{u} \cdot \bar{\boldsymbol{t}} \, da \tag{2.44}$$

respectively. The potential V exists only if the symmetry conditions of the *Vainberg theorem* (see Chapter 9, [2]) are satisfied and the formulation is said to have a variational basis. The *Lagrangian* $L(\boldsymbol{u}, \dot{\boldsymbol{u}})$ is defined as

$$L(\boldsymbol{u}, \dot{\boldsymbol{u}}) = T(\dot{\boldsymbol{u}}) - V(\boldsymbol{u}) \tag{2.45}$$

and its action integral I as:

$$I = \int_{t0}^{t} L(\boldsymbol{u}, \dot{\boldsymbol{u}})\, dt \tag{2.46}$$

The *Hamilton's* principle of critical action (see [10]) states:

$$\delta I = 0 \tag{2.47}$$

One may verify that the *Euler-Lagrange equation*:

$$\frac{d}{dt}\left(D_2 L(\boldsymbol{u}, \dot{\boldsymbol{u}})\right) - D_1 L(\boldsymbol{u}, \dot{\boldsymbol{u}}) = \frac{d}{dt}\left(\frac{\delta L}{\delta \dot{\boldsymbol{u}}}\right) - \frac{\delta L}{\delta \boldsymbol{u}} = 0 \tag{2.48}$$

obtained from the Hamilton's principle, for the Lagrangian given above, is identical to the equation of equilibrium (2.9). Another form of the Euler-Lagrange equations can be obtained by defining the *Hamiltonian*:

$$H(\boldsymbol{u}, \boldsymbol{p}) = \int_{\Omega(t)} \boldsymbol{p} \cdot \dot{\boldsymbol{u}}\, dv - L(\boldsymbol{u}, \dot{\boldsymbol{u}}) = T(\dot{\boldsymbol{u}}) + V(\boldsymbol{u}) \tag{2.49}$$

where $\boldsymbol{p} = \rho \dot{\boldsymbol{u}}$ denotes linear momentum. This leads to the *Hamilton's equations*:

$$\frac{\partial \boldsymbol{u}}{\partial t} = \frac{\delta H}{\delta \boldsymbol{p}} \tag{2.50}$$

$$\frac{\partial \boldsymbol{p}}{\partial t} = -\frac{\delta H}{\delta \boldsymbol{u}} \tag{2.51}$$

where $\dfrac{\delta H}{\delta \boldsymbol{u}}$ and $\dfrac{\delta H}{\delta \boldsymbol{p}}$ represent the *functional derivatives* of H (see [10]).

2.7 Linearization

The governing equations of the initial boundary value problem are, in general, non-linear in the constitutive relationships (material non-linearity) and/or the strain-displacement relations (geometric non-linearity). In order to facilitate an iterative solution using Newton's

method, one needs to linearize these equations.

The functional $\mathscr{G}(\boldsymbol{\phi}, \tilde{\boldsymbol{u}})$ is linearized around an assumed solution $\boldsymbol{\phi}^i$, at iteration i, as follows:

$$\begin{aligned} & \mathscr{G}((\boldsymbol{\phi}^i + \Delta\boldsymbol{u}), \tilde{\boldsymbol{u}}) \approx \mathscr{G}(\boldsymbol{\phi}^i, \tilde{\boldsymbol{u}}) + D\mathscr{G}(\boldsymbol{\phi}, \tilde{\boldsymbol{u}})\Big|_{\boldsymbol{\phi}^i}[\Delta\boldsymbol{u}] = 0 \\ \Rightarrow \qquad & D\mathscr{G}(\boldsymbol{\phi}, \tilde{\boldsymbol{u}})\Big|_{\boldsymbol{\phi}^i}[\Delta\boldsymbol{u}] = -\mathscr{G}(\boldsymbol{\phi}^i, \tilde{\boldsymbol{u}}) \end{aligned} \tag{2.52}$$

where $\Delta\boldsymbol{u}$ is the displacement update. The assumed solution for the next iteration $i+1$ is given by $\boldsymbol{\phi}^{i+1} = \boldsymbol{\phi}^i + \Delta\boldsymbol{u}$ and the procedure is repeated until $|\mathscr{G}(\boldsymbol{\phi}^{i^*}, \tilde{\boldsymbol{u}})| \leq \epsilon_{tol}$ for some final iteration i^*.

The term $D\mathscr{G}(\boldsymbol{\phi}, \tilde{\boldsymbol{u}})\Big|_{\boldsymbol{\phi}^i}[\Delta\boldsymbol{u}]$ can be computed from equation (2.32) as follows:

$$D\mathscr{G}(\boldsymbol{\phi}, \tilde{\boldsymbol{u}})[\Delta\boldsymbol{u}] = D\left(\int_{\Omega(t)} \tilde{\boldsymbol{u}} \cdot \rho\ddot{\boldsymbol{u}}\, dv\right)[\Delta\boldsymbol{u}] + D\mathscr{W}_I[\Delta\boldsymbol{u}] - D\mathscr{W}_E[\Delta\boldsymbol{u}] \tag{2.53}$$

Note that if the applied loads are not deformation dependent then last term $D\mathscr{W}_E[\Delta\boldsymbol{u}]$ does not contribute. The integrals in the first two terms are computed on a domain that is deformation dependent and in order to simplify the calculation of derivatives for the linearization, the integrals may be transformed onto domains that are not deformation dependent. Any prior known (or assumed) configuration of the domain is sufficient for this purpose. Usually, either the initial undeformed configuration Ω_0 or the current assumed configuration $\Omega(t)^i$ is chosen, leading to *total* and *updated* Lagrangian formulations respectively. Choosing the total Lagrangian approach, the first term is computed as:

$$D\left(\int_{\Omega_0} \tilde{\boldsymbol{u}} \cdot \rho\ddot{\boldsymbol{u}}\, J dV\right)[\Delta\boldsymbol{u}] = \int_{\Omega_0} \rho_0\, \tilde{\boldsymbol{u}} \cdot \Delta\ddot{\boldsymbol{u}}\, dV = \int_{\Omega(t)} \rho\, \tilde{\boldsymbol{u}} \cdot \Delta\ddot{\boldsymbol{u}}\, dv \tag{2.54}$$

The second term in equation (2.53):

$$\begin{aligned} D\mathscr{W}_I[\Delta \boldsymbol{u}] &= D\left(\int_{\Omega_0} \boldsymbol{P} : \frac{\partial \tilde{\boldsymbol{u}}}{\partial \boldsymbol{X}}\, dV\right)[\Delta \boldsymbol{u}] \\ &= \int_{\Omega_0} \frac{\partial \tilde{\boldsymbol{u}}}{\partial \boldsymbol{X}} : \left(\frac{\partial \boldsymbol{P}}{\partial \boldsymbol{F}} D\boldsymbol{F}[\Delta \boldsymbol{u}]\right) dV \\ &= \int_{\Omega_0} \frac{\partial \tilde{\boldsymbol{u}}}{\partial \boldsymbol{X}} : \left(\frac{\partial \boldsymbol{P}}{\partial \boldsymbol{F}} \frac{\partial \Delta \boldsymbol{u}}{\partial \boldsymbol{X}}\right) dV \end{aligned} \tag{2.55}$$

where we have used the definition of the first Piola Kirchhoff stress tensor from (2.12) and the chain rule to expand $\nabla_x \tilde{\boldsymbol{u}}$. An alternative, symmetric form can be obtained in terms of the second Piola Kirchhoff stress tensor (2.13) as follows:

$$\begin{aligned} D\mathscr{W}_I[\Delta \boldsymbol{u}] &= D\left(\int_{\Omega_0} \boldsymbol{S} : \tilde{\boldsymbol{E}}\, dV\right)[\Delta \boldsymbol{u}] \\ &= \int_{\Omega_0} (D\boldsymbol{S}[\Delta \boldsymbol{u}]) : \tilde{\boldsymbol{E}} + \boldsymbol{S} : (D\tilde{\boldsymbol{E}}[\Delta \boldsymbol{u}])\, dV \\ &= \int_{\Omega_0} \tilde{\boldsymbol{E}} : \left(\frac{\partial \boldsymbol{S}}{\partial \boldsymbol{E}}\, D\boldsymbol{E}[\Delta \boldsymbol{u}]\right) dV + \int_{\Omega_0} \boldsymbol{S} : (D\tilde{\boldsymbol{E}}[\Delta \boldsymbol{u}])\, dV \\ &= \int_{\Omega_0} \tilde{\boldsymbol{E}} : \mathcal{C}\, \Delta \boldsymbol{E}\, dV + \int_{\Omega_0} \boldsymbol{S} : \frac{1}{2}\left((\Delta \boldsymbol{F})^T \tilde{\boldsymbol{F}} + \tilde{\boldsymbol{F}}^T (\Delta \boldsymbol{F})\right) dV \end{aligned} \tag{2.56}$$

where, in addition to the symmetry of second Piola Kirchhoff stress tensor, we have used the following definitions:

$$\Delta \boldsymbol{F} = D\boldsymbol{F}[\Delta \boldsymbol{u}] = \nabla_X (\Delta \boldsymbol{u}) = \frac{\partial (\Delta \boldsymbol{u})}{\partial \boldsymbol{X}} \tag{2.57}$$

$$\Delta \boldsymbol{E} = D\boldsymbol{E}[\Delta \boldsymbol{u}] = \frac{1}{2}\left(\boldsymbol{F}^T (\Delta \boldsymbol{F}) + (\Delta \boldsymbol{F})^T \boldsymbol{F}\right) \tag{2.58}$$

$$D\tilde{\boldsymbol{E}}[\Delta \boldsymbol{u}] = \frac{1}{2}\left((\Delta \boldsymbol{F})^T \tilde{\boldsymbol{F}} + \tilde{\boldsymbol{F}}^T (\Delta \boldsymbol{F})\right) \tag{2.59}$$

Noting that $\tilde{\boldsymbol{F}} = \nabla_X \tilde{\boldsymbol{u}} = \nabla_x \tilde{\boldsymbol{u}}\, \boldsymbol{F}$ and $\Delta \boldsymbol{F} = \nabla_X (\Delta \boldsymbol{u}) = \nabla_x (\Delta \boldsymbol{u})\, \boldsymbol{F}$, the expression (2.56)

can now be transformed onto the current assumed configuration $\Omega(t)^i$:

$$\begin{aligned} D\mathscr{W}_I[\Delta\boldsymbol{u}] &= \int_{\Omega_0} \frac{1}{2}(\boldsymbol{F}^T\tilde{\boldsymbol{F}} + \tilde{\boldsymbol{F}}^T\boldsymbol{F}) : \mathcal{C}\frac{1}{2}\left(\boldsymbol{F}^T(\Delta\boldsymbol{F}) + (\Delta\boldsymbol{F})^T\boldsymbol{F}\right)\, dV \\ &\qquad + \int_{\Omega_0} \boldsymbol{S} : \boldsymbol{F}^T\frac{1}{2}\left((\nabla_x(\Delta\boldsymbol{u}))^T\nabla_x\tilde{\boldsymbol{u}} + (\nabla_x\tilde{\boldsymbol{u}})^T\nabla_x(\Delta\boldsymbol{u})\right)\boldsymbol{F}\, dV \\ &= \int_{\Omega_0} (\boldsymbol{F}^T\tilde{\boldsymbol{\varepsilon}}\boldsymbol{F}) : \frac{1}{J}\mathcal{C}(\boldsymbol{F}^T\Delta\boldsymbol{\varepsilon}\boldsymbol{F})\, J\, dV \\ &\qquad + \int_{\Omega_0} \frac{1}{J}(\boldsymbol{F}\boldsymbol{S}\boldsymbol{F}^T) : \frac{1}{2}\left((\nabla_x(\Delta\boldsymbol{u}))^T\nabla_x\tilde{\boldsymbol{u}} + (\nabla_x\tilde{\boldsymbol{u}})^T\nabla_x(\Delta\boldsymbol{u})\right)\, J\, dV \\ &= \int_{\Omega(t)} \tilde{\boldsymbol{\varepsilon}} : c\,\Delta\boldsymbol{\varepsilon}\, dv + \int_{\Omega(t)} \boldsymbol{\sigma} : \frac{1}{2}\left((\nabla_x(\Delta\boldsymbol{u}))^T\nabla_x\tilde{\boldsymbol{u}} + (\nabla_x\tilde{\boldsymbol{u}})^T\nabla_x(\Delta\boldsymbol{u})\right)\, dv \end{aligned} \tag{2.60}$$

where

$$\Delta\boldsymbol{\varepsilon} = \frac{1}{2}\left(\nabla_x(\Delta\boldsymbol{u}) + \nabla_x(\Delta\boldsymbol{u})^T\right) \tag{2.61}$$

$$c_{ijkl} = J^{-1}F_{iI}F_{jJ}F_{kK}F_{lL}\mathcal{C}_{IJKL} \tag{2.62}$$

2.8 Discretization

The virtual work equation (2.32) is expressed in terms of continuous *field* variables. In general, one cannot find exact solutions for $\boldsymbol{\phi} \in \mathcal{V}$ that will satisfy the equation for all $\tilde{\boldsymbol{u}} \in \tilde{\mathcal{V}}$. However, approximate solutions can be found by suitably restricting the spaces $\mathcal{V}$ and $\tilde{\mathcal{V}}$ to a linear combination of finite number of functions. This approach is referred to as the Raleigh-Ritz method. Restricting the spaces, effectively reduces an infinite dimensional problem to a discrete one that can be solved using the Newton's update equation (2.52).

2.8.1 Spatial Discretization

The finite element method (FEM) is one of the most common choices for discretization (see [11],[12]). Using this approach, the structural domain $\Omega(t)$ is divided into a number of *elements* connected at *nodes* to form a *mesh* as shown in figure 2.2. We will denote the

number of elements in the mesh by m and the total number of nodes in the mesh by n.

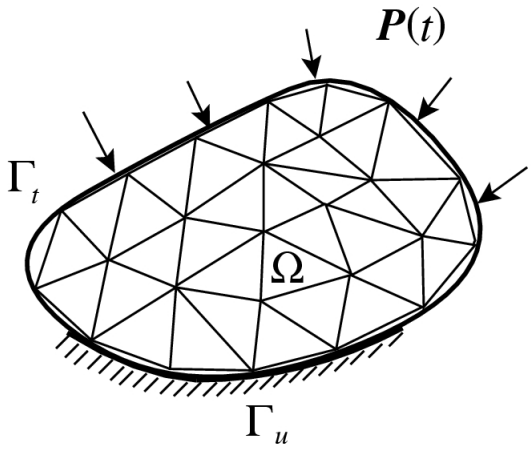

Figure 2.2: Finite element discretization of a structural domain.

Now all the integrals in the equations (2.32) and (2.52) can be transformed into a sum of integrals over all the elements:

$$\begin{aligned} \int_{\Omega(t)} [\cdot](\boldsymbol{x})\, dv = \int_{\Omega_0} [\cdot](\boldsymbol{X})\, dV \quad &= \quad \sum_{e=1}^{m} \int_{\Omega^e(t)} [\cdot](\boldsymbol{x})\, dv = \sum_{e=1}^{m} \int_{\Omega_0^e} [\cdot](\boldsymbol{X})\, dV \\ \int_{\Gamma_N(t)} [\cdot](\boldsymbol{x})\, da = \int_{\Gamma_{N_0}} [\cdot](\boldsymbol{X})\, dA \quad &= \quad \sum_{e=1}^{m} \int_{\Gamma_N^e(t)} [\cdot](\boldsymbol{x})\, da = \sum_{e=1}^{m} \int_{\Gamma_{N_0}^e} [\cdot](\boldsymbol{X})\, dA \end{aligned} \tag{2.63}$$

Within each element e, the spaces $\mathcal{V}$ and $\tilde{\mathcal{V}}$ are restricted to certain class of functions, usually polynomials, called *shape* or *basis* functions. Finite element basis functions, $\bar{N}_\alpha^e(\boldsymbol{X})$ on Ω_0 or $\hat{N}_\alpha^e(\boldsymbol{x})$ on $\Omega(t)$, are associated with each *node* α within element e and take the value unity at node α and zero at all other nodes.

The calculation of individual element integrals can be further simplified by mapping each element to a single *parent* element using *isoparametric* mapping. The domain of the parent element, denoted by ■ with boundary □ in (ξ_1, ξ_2, ξ_3) coordinates, is the same for all the elements in the mesh allowing the coordinates $\boldsymbol{X}$ and $\boldsymbol{x}$ to be interpolated using the parent

element basis functions $N_\alpha(\boldsymbol{\xi})$ as:

$$\begin{aligned} \boldsymbol{X} = \sum_{\alpha=1}^{e_n} \boldsymbol{X}_\alpha^e \, N_\alpha(\boldsymbol{\xi}) \qquad ; \qquad \boldsymbol{x} = \sum_{\alpha=1}^{e_n} \boldsymbol{x}_\alpha^e \, N_\alpha(\boldsymbol{\xi}) \\ X_I = X_{I\alpha}^e N_\alpha \qquad ; \qquad x_i = x_{i\alpha}^e N_\alpha \end{aligned} \tag{2.64}$$

where e_n is the number of nodes in element e and $\boldsymbol{X}_\alpha^e$ and $\boldsymbol{x}_\alpha^e$ represent the nodal co-ordinates of node α of element e in the initial and current configurations respectively. Summation over $1 \leq \alpha \leq e_n$ is implied. In conventional matrix form (for 3D: d=3), the same is expressed as:

$$\begin{aligned} \boldsymbol{X} = [\boldsymbol{N}]\,\{\boldsymbol{X}^e\} \qquad ; \qquad \boldsymbol{x} = [\boldsymbol{N}]\,\{\boldsymbol{x}^e\} \\ [\boldsymbol{N}] = \left[\begin{array}{c|ccc|c} & N_\alpha & 0 & 0 & \\ \cdots & 0 & N_\alpha & 0 & \cdots \\ & 0 & 0 & N_\alpha & \end{array}\right] \\ \{\boldsymbol{X}^e\} = \left[\begin{array}{c} \vdots \\ \hline X_{1\alpha}^e \\ X_{2\alpha}^e \\ X_{3\alpha}^e \\ \hline \vdots \end{array}\right] \qquad \{\boldsymbol{x}^e\} = \left[\begin{array}{c} \vdots \\ \hline x_{1\alpha}^e \\ x_{2\alpha}^e \\ x_{3\alpha}^e \\ \hline \vdots \end{array}\right] \end{aligned} \tag{2.65}$$

The real, virtual and incremental displacements within element e are also interpolated using the same basis functions:

$$\begin{aligned} \boldsymbol{U}^e(\boldsymbol{X}(\boldsymbol{\xi})) = \boldsymbol{u}^e(\boldsymbol{x}(\boldsymbol{\xi})) = \boldsymbol{U}_\alpha^e \, \bar{N}_\alpha^e(\boldsymbol{X}) = \boldsymbol{U}_\alpha^e \, \hat{N}_\alpha^e(\boldsymbol{x}) = \boldsymbol{U}_\alpha^e \, N_\alpha(\boldsymbol{\xi}) = [\boldsymbol{N}]\{\boldsymbol{U}^e\} \\ \tilde{\boldsymbol{U}}^e(\boldsymbol{X}(\boldsymbol{\xi})) = \tilde{\boldsymbol{u}}^e(\boldsymbol{x}(\boldsymbol{\xi})) = \tilde{\boldsymbol{U}}_\alpha^e \, \bar{N}_\alpha^e(\boldsymbol{X}) = \tilde{\boldsymbol{U}}_\alpha^e \, \hat{N}_\alpha^e(\boldsymbol{x}) = \tilde{\boldsymbol{U}}_\alpha^e \, N_\alpha(\boldsymbol{\xi}) = [\boldsymbol{N}]\{\tilde{\boldsymbol{U}}^e\} \\ \Delta\boldsymbol{U}^e(\boldsymbol{X}(\boldsymbol{\xi})) = \Delta\boldsymbol{u}^e(\boldsymbol{x}(\boldsymbol{\xi})) = \Delta\boldsymbol{U}_\alpha^e \, \bar{N}_\alpha^e(\boldsymbol{X}) = \Delta\boldsymbol{U}_\alpha^e \, \hat{N}_\alpha^e(\boldsymbol{x}) = \Delta\boldsymbol{U}_\alpha^e \, N_\alpha(\boldsymbol{\xi}) = [\boldsymbol{N}]\{\Delta\boldsymbol{U}^e\} \end{aligned} \tag{2.66}$$

where $\boldsymbol{U}_\alpha^e$, $\Delta\boldsymbol{U}_\alpha^e$ and $\tilde{\boldsymbol{U}}_\alpha^e$ denote the nodal *degrees of freedom* (dofs) of node α of element

e representing the real, incremental and virtual displacements respectively. The nodal dofs $\{\boldsymbol{U}^e\}$ of element e are related to the global dofs $\boldsymbol{U}$ through a boolean matrix $\boldsymbol{A}^e$ as:

$$\{\boldsymbol{U}^e\} = \boldsymbol{A}^e \boldsymbol{U} \tag{2.67}$$

$\boldsymbol{A}^e$ denotes the *assembly* matrix that picks out the components of $\{\boldsymbol{U}^e\}$ from the vector of global dofs $\boldsymbol{U}$. Note that along the Dirichlet boundary Γ_D, the components of $\tilde{\boldsymbol{U}}$ are zero and the components of $\boldsymbol{U}$ are specified.

The element integrals (2.63) can be expressed over the parent element as:

$$\begin{aligned}
\sum_{e=1}^{m} \int_{\Omega^e(t)} [\cdot](\boldsymbol{x})\, dv &= \sum_{e=1}^{m} \int_{\blacksquare} [\cdot](\boldsymbol{x}(\boldsymbol{\xi}))\, J_{x\xi}\, d\blacksquare \approx \sum_{e=1}^{m} \left[\sum_{k} w_k\, [\cdot](\boldsymbol{x}(\boldsymbol{\xi}_k)) J_{x\xi}(\boldsymbol{\xi}_k) \right] \\
\sum_{e=1}^{m} \int_{\Omega_0^e} [\cdot](\boldsymbol{X})\, dV &= \sum_{e=1}^{m} \int_{\blacksquare} [\cdot](\boldsymbol{X}(\boldsymbol{\xi})) J_{X\xi}\, d\blacksquare \approx \sum_{e=1}^{m} \left[\sum_{k} w_k [\cdot](\boldsymbol{X}(\boldsymbol{\xi}_k)) J_{X\xi}(\boldsymbol{\xi}_k) \right] \\
\sum_{e=1}^{m} \int_{\Gamma_N^e(t)} [\cdot](\boldsymbol{x})\, da &= \sum_{e=1}^{m} \int_{\square_N} [\cdot](\boldsymbol{x}(\boldsymbol{\xi}))\, J_{x\xi} \boldsymbol{F}_{x\xi}^{-T}\, d\square \\
&\approx \sum_{e=1}^{m} \left[\sum_{l}^{\square_N} w_l\, [\cdot](\boldsymbol{x}(\boldsymbol{\xi}_l)) J_{x\xi}(\boldsymbol{\xi}_l) \boldsymbol{F}_{x\xi}^{-T}(\boldsymbol{\xi}_l) \right] \\
\sum_{e=1}^{m} \int_{\Gamma_{N_0}^e} [\cdot](\boldsymbol{X})\, dA &= \sum_{e=1}^{m} \int_{\square_N} [\cdot](\boldsymbol{X}(\boldsymbol{\xi})) J_{X\xi} \boldsymbol{F}_{X\xi}^{-T}\, d\square \\
&\approx \sum_{e=1}^{m} \left[\sum_{l}^{\square_N} w_l [\cdot](\boldsymbol{X}(\boldsymbol{\xi}_l)) J_{X\xi}(\boldsymbol{\xi}_l) \boldsymbol{F}_{X\xi}^{-T}(\boldsymbol{\xi}_l) \right]
\end{aligned} \tag{2.68}$$

where Gauss quadrature has been used to approximate the integrals with weights and locations $(w_k,\, \boldsymbol{\xi}_k)$ for the domain and $(w_l,\, \boldsymbol{\xi}_l)$ for the boundary. $\boldsymbol{F}_{x\xi}$ and $\boldsymbol{F}_{X\xi}$ are given by:

$$\begin{aligned}
\boldsymbol{F}_{x\xi} &= \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\xi}} = \frac{\partial N_\alpha}{\partial \xi_j} x_{i\alpha}^e\, \boldsymbol{e}_i \otimes \hat{\boldsymbol{i}}_j \qquad J_{x\xi} = \det(\boldsymbol{F}_{x\xi}) \\
\boldsymbol{F}_{X\xi} &= \frac{\partial \boldsymbol{X}}{\partial \boldsymbol{\xi}} = \frac{\partial N_\alpha}{\partial \xi_j} X_{I\alpha}^e\, \boldsymbol{E}_I \otimes \hat{\boldsymbol{i}}_j \qquad J_{X\xi} = \det(\boldsymbol{F}_{X\xi})
\end{aligned} \tag{2.69}$$

where $\hat{\boldsymbol{i}}$ are the unit vectors for the parent element.

In order to discretize the quantities in equation (2.52), computation of $\boldsymbol{F}$ is also required:

$$\boldsymbol{F} = F_{iI}\,\boldsymbol{e}_i \otimes \boldsymbol{E}_I = \frac{\partial x_i}{\partial X_I}\,\boldsymbol{e}_i \otimes \boldsymbol{E}_I = x^e_{i\alpha}\frac{\partial N_\alpha}{\partial X_I}\,\boldsymbol{e}_i \otimes \boldsymbol{E}_I \tag{2.70}$$

where $\dfrac{\partial N_\alpha}{\partial X_I}$ and $\dfrac{\partial N_\alpha}{\partial x_i}$ can be computed separately as:

$$\begin{aligned}
\frac{\partial N_\alpha}{\partial X_I} &= \frac{\partial N_\alpha}{\partial \xi_k}\left(\frac{\partial X_I}{\partial \xi_k}\right)^{-1} = \frac{\partial N_\alpha}{\partial \xi_k}\left(X^e_{I\beta}\frac{\partial N_\beta}{\partial \xi_k}\right)^{-1} \\
\frac{\partial N_\alpha}{\partial x_i} &= \frac{\partial N_\alpha}{\partial \xi_k}\left(\frac{\partial x_i}{\partial \xi_k}\right)^{-1} = \frac{\partial N_\alpha}{\partial \xi_k}\left(x^e_{i\beta}\frac{\partial N_\beta}{\partial \xi_k}\right)^{-1}
\end{aligned} \tag{2.71}$$

Note that once a configuration $\Omega(t)^i$ is assumed, all the quantities above are known and $\boldsymbol{F} = \boldsymbol{F}_{x\xi}\boldsymbol{F}^{-1}_{X\xi}$.

Quantities for the updated Lagrangian expressions (2.33), (2.34), (2.35) and (2.60):

$$\begin{aligned}
\nabla_x \tilde{\boldsymbol{u}} &= \frac{\partial \tilde{u}^e_i}{\partial x_j}\,\boldsymbol{e}_i \otimes \boldsymbol{e}_j = \tilde{u}^e_{i\alpha}\frac{\partial N_\alpha}{\partial x_j}\,\boldsymbol{e}_i \otimes \boldsymbol{e}_j \\
\tilde{\boldsymbol{\epsilon}} &= \frac{1}{2}\left(\tilde{u}^e_{i\alpha}\frac{\partial N_\alpha}{\partial x_j} + \tilde{u}^e_{j\alpha}\frac{\partial N_\alpha}{\partial x_i}\right)\boldsymbol{e}_i \otimes \boldsymbol{e}_j \\
\nabla_x(\Delta \boldsymbol{u}) &= \Delta u^e_{i\alpha}\frac{\partial N_\alpha}{\partial x_j}\,\boldsymbol{e}_i \otimes \boldsymbol{e}_j \\
\Delta\boldsymbol{\epsilon} &= \frac{1}{2}\left((\Delta u^e_{i\alpha})\frac{\partial N_\alpha}{\partial x_j} + (\Delta u^e_{j\alpha})\frac{\partial N_\alpha}{\partial x_i}\right)\boldsymbol{e}_i \otimes \boldsymbol{e}_j
\end{aligned} \tag{2.72}$$

In the conventional matrix notation $\tilde{\boldsymbol{\epsilon}}$ and $\Delta\boldsymbol{\epsilon}$ are expressed as:

$$\{\tilde{\boldsymbol{\epsilon}}\} = \left[\boldsymbol{B}^{UL}_\alpha\right]\{\tilde{\boldsymbol{U}}_\alpha\} \qquad ; \qquad \{\Delta\boldsymbol{\epsilon}\} = \left[\boldsymbol{B}^{UL}_\alpha\right]\{\Delta\boldsymbol{U}_\alpha\} \tag{2.73}$$

where

$$\{\tilde{\boldsymbol{\epsilon}}\} = \text{vec}(\tilde{\boldsymbol{\epsilon}}) = [\tilde{\epsilon}_{11},\, \tilde{\epsilon}_{22},\, \tilde{\epsilon}_{33},\, 2\tilde{\epsilon}_{12},\, 2\tilde{\epsilon}_{23},\, 2\tilde{\epsilon}_{31}]^T$$

$$\{\Delta\boldsymbol{\epsilon}\} = \text{vec}(\Delta\boldsymbol{\epsilon}) = [\Delta\epsilon_{11},\, \Delta\epsilon_{22},\, \Delta\epsilon_{33},\, 2\Delta\epsilon_{12},\, 2\Delta\epsilon_{23},\, 2\Delta\epsilon_{31}]^T$$

$$\left[\boldsymbol{B}_\alpha^{UL}\right] = \begin{bmatrix} \dfrac{\partial N_\alpha}{\partial x_1} & 0 & 0 \\ 0 & \dfrac{\partial N_\alpha}{\partial x_2} & 0 \\ 0 & 0 & \dfrac{\partial N_\alpha}{\partial x_3} \\ \dfrac{\partial N_\alpha}{\partial x_2} & \dfrac{\partial N_\alpha}{\partial x_1} & 0 \\ 0 & \dfrac{\partial N_\alpha}{\partial x_3} & \dfrac{\partial N_\alpha}{\partial x_2} \\ \dfrac{\partial N_\alpha}{\partial x_3} & 0 & \dfrac{\partial N_\alpha}{\partial x_1} \end{bmatrix} \tag{2.74}$$

Similarly quantities required for the total Lagrangian expressions (2.37), (2.38), (2.40) and (2.56):

$$\begin{aligned} \tilde{\boldsymbol{F}} &= \tilde{u}_{i\alpha}^e \frac{\partial N_\alpha}{\partial X_I}\, \boldsymbol{e}_i \otimes \boldsymbol{E}_I \\ \tilde{\boldsymbol{E}} &= \frac{1}{2}\left(F_{iI}\tilde{F}_{iJ} + \tilde{F}_{iI}F_{iJ}\right) \boldsymbol{e}_i \otimes \boldsymbol{E}_I = \frac{1}{2}\left(F_{iI}\frac{\partial N_\alpha}{\partial X_J} + F_{iJ}\frac{\partial N_\alpha}{\partial X_I}\right) \tilde{u}_{i\alpha}^e\, \boldsymbol{e}_i \otimes \boldsymbol{E}_I \\ \Delta\boldsymbol{F} &= \Delta u_{i\alpha}^e \frac{\partial N_\alpha}{\partial X_I}\, \boldsymbol{e}_i \otimes \boldsymbol{E}_I \\ \Delta\boldsymbol{E} &= \frac{1}{2}\left(F_{iI}\,\Delta F_{iJ} + \Delta F_{iI}\,F_{iJ}\right) \boldsymbol{e}_i \otimes \boldsymbol{E}_I = \frac{1}{2}\left(F_{iI}\frac{\partial N_\alpha}{\partial X_J} + F_{iJ}\frac{\partial N_\alpha}{\partial X_I}\right) \Delta u_{i\alpha}^e\, \boldsymbol{e}_i \otimes \boldsymbol{E}_I \end{aligned} \tag{2.75}$$

where $\tilde{u}_{i\alpha}^e$ is arbitrary and $\Delta u_{i\alpha}^e$ is the unknown to be computed. In the conventional matrix notation $\tilde{\boldsymbol{E}}$ and $\Delta\boldsymbol{E}$ are expressed as:

$$\{\tilde{\boldsymbol{E}}\} = \left[\boldsymbol{B}_\alpha^{TL}\right]\{\tilde{\boldsymbol{U}}_\alpha\} \qquad ; \qquad \{\Delta\boldsymbol{E}\} = \left[\boldsymbol{B}_\alpha^{TL}\right]\{\Delta\boldsymbol{U}_\alpha\} \tag{2.76}$$

where

$$\{\tilde{\boldsymbol{E}}\} = \text{vec}(\tilde{\boldsymbol{E}}) = [\tilde{E}_{11},\, \tilde{E}_{22},\, \tilde{E}_{33},\, 2\tilde{E}_{12},\, 2\tilde{E}_{23},\, 2\tilde{E}_{31}]^T$$

$$\{\Delta\boldsymbol{E}\} = \text{vec}(\Delta\boldsymbol{E}) = [\Delta E_{11},\, \Delta E_{22},\, \Delta E_{33},\, 2\Delta E_{12},\, 2\Delta E_{23},\, 2\Delta E_{31}]^T$$

$$\left[\boldsymbol{B}_\alpha^{TL}\right] = \begin{bmatrix} F_{11}\dfrac{\partial N_\alpha}{\partial X_1} & F_{21}\dfrac{\partial N_\alpha}{\partial X_1} & F_{31}\dfrac{\partial N_\alpha}{\partial X_1} \\ F_{12}\dfrac{\partial N_\alpha}{\partial X_2} & F_{22}\dfrac{\partial N_\alpha}{\partial X_2} & F_{32}\dfrac{\partial N_\alpha}{\partial X_2} \\ F_{13}\dfrac{\partial N_\alpha}{\partial X_3} & F_{23}\dfrac{\partial N_\alpha}{\partial X_3} & F_{33}\dfrac{\partial N_\alpha}{\partial X_3} \\ F_{11}\dfrac{\partial N_\alpha}{\partial X_2} + F_{12}\dfrac{\partial N_\alpha}{\partial X_1} & F_{21}\dfrac{\partial N_\alpha}{\partial X_2} + F_{22}\dfrac{\partial N_\alpha}{\partial X_1} & F_{31}\dfrac{\partial N_\alpha}{\partial X_2} + F_{32}\dfrac{\partial N_\alpha}{\partial X_1} \\ F_{12}\dfrac{\partial N_\alpha}{\partial X_3} + F_{13}\dfrac{\partial N_\alpha}{\partial X_2} & F_{22}\dfrac{\partial N_\alpha}{\partial X_3} + F_{23}\dfrac{\partial N_\alpha}{\partial X_2} & F_{32}\dfrac{\partial N_\alpha}{\partial X_3} + F_{33}\dfrac{\partial N_\alpha}{\partial X_2} \\ F_{13}\dfrac{\partial N_\alpha}{\partial X_1} + F_{11}\dfrac{\partial N_\alpha}{\partial X_3} & F_{23}\dfrac{\partial N_\alpha}{\partial X_1} + F_{21}\dfrac{\partial N_\alpha}{\partial X_3} & F_{33}\dfrac{\partial N_\alpha}{\partial X_1} + F_{31}\dfrac{\partial N_\alpha}{\partial X_3} \end{bmatrix} \tag{2.77}$$

The stress tensors $\boldsymbol{S}$ and $\boldsymbol{\sigma}$ are assembled into vectors as:

$$\begin{aligned} \{\boldsymbol{\sigma}\} &= \text{vec}(\boldsymbol{\sigma}) = [\sigma_{11}, \sigma_{22}, \sigma_{33}, \sigma_{12}, \sigma_{23}, \sigma_{31}]^T \\ \{\boldsymbol{S}\} &= \text{vec}(\boldsymbol{S}) = [S_{11}, S_{22}, S_{33}, S_{12}, S_{23}, S_{31}]^T \end{aligned} \tag{2.78}$$

The elasticity tensors $\mathcal{C}$ and c can be assembled in matrix form as:

$$[\boldsymbol{\mathcal{C}}] = \text{mat}(\mathcal{C}) = \begin{bmatrix} \mathcal{C}_{1111} & \mathcal{C}_{1122} & \mathcal{C}_{1133} & \mathcal{C}_{1112} & \mathcal{C}_{1123} & \mathcal{C}_{1131} \\ & \mathcal{C}_{2222} & \mathcal{C}_{2233} & \mathcal{C}_{2212} & \mathcal{C}_{2223} & \mathcal{C}_{2231} \\ & & \mathcal{C}_{3333} & \mathcal{C}_{3312} & \mathcal{C}_{3323} & \mathcal{C}_{3331} \\ & & & \mathcal{C}_{1212} & \mathcal{C}_{1223} & \mathcal{C}_{1231} \\ & \text{SYM} & & & \mathcal{C}_{2323} & \mathcal{C}_{2331} \\ & & & & & \mathcal{C}_{3131} \end{bmatrix} \tag{2.79}$$

Quantities in the Newton update equation (2.52) can now be expressed in terms of the

discrete variables above. The right hand side is obtained from equation (2.33) or (2.37):

$$-\mathscr{G}(\boldsymbol{\phi}^i, \tilde{u}) \approx -\sum_{e=1}^{m} \{\tilde{\boldsymbol{U}}^e\}^T \left([\boldsymbol{M}^e]\{\ddot{\boldsymbol{U}}^e\} + \{\boldsymbol{Q}^e(\boldsymbol{U}(t))\} - \{\boldsymbol{P}^e\} \right) \tag{2.80}$$

where:

$$[\boldsymbol{M}^e] = \sum_k w_k \rho [\boldsymbol{N}]^T [\boldsymbol{N}] J_{x\xi}(\boldsymbol{\xi}_k) = \sum_k w_k \rho_0 [\boldsymbol{N}]^T [\boldsymbol{N}] J_{X\xi}(\boldsymbol{\xi}_k) \tag{2.81}$$

$$\{\boldsymbol{Q}^e(\boldsymbol{U}(t))\} = \sum_k w_k [\boldsymbol{B}^{UL}]^T \{\boldsymbol{\sigma}\} J_{x\xi}(\boldsymbol{\xi}_k) = \sum_k w_k [\boldsymbol{B}^{TL}]^T \{\boldsymbol{S}\} J_{X\xi}(\boldsymbol{\xi}_k) \tag{2.82}$$

$$\begin{aligned} \{\boldsymbol{P}^e\} &= \sum_k w_k [\boldsymbol{N}]^T \bar{\boldsymbol{b}} J_{x\xi}(\boldsymbol{\xi}_k) + \sum_l^{\square_N} w_l [\boldsymbol{N}]^T \bar{\boldsymbol{t}} J_{x\xi} \boldsymbol{F}_{x\xi}^{-T}(\boldsymbol{\xi}_l) \\ &= \sum_k w_k [\boldsymbol{N}]^T \bar{\boldsymbol{B}}_0 J_{X\xi}(\boldsymbol{\xi}_k) + \sum_l^{\square_N} w_l [\boldsymbol{N}]^T \bar{\boldsymbol{T}}_0 J_{X\xi} \boldsymbol{F}_{X\xi}^{-T}(\boldsymbol{\xi}_l) \end{aligned} \tag{2.83}$$

The directional derivatives on the left hand side of equation (2.52) are given by:

$$D\left(\int_{\Omega_0} \tilde{\boldsymbol{u}} \cdot \rho_0 \ddot{\boldsymbol{u}} \, dV \right) [\Delta \boldsymbol{u}] = D\left(\int_{\Omega(t)} \tilde{\boldsymbol{u}} \cdot \rho \ddot{\boldsymbol{u}} \, dv \right) [\Delta \boldsymbol{u}] = \sum_{e=1}^{m} \{\tilde{\boldsymbol{U}}^e\}^T [\boldsymbol{M}^e] \{\Delta \ddot{\boldsymbol{U}}^e\} \tag{2.84}$$

$$D\mathscr{W}_I[\Delta \boldsymbol{u}] = \sum_{e=1}^{m} \{\tilde{\boldsymbol{U}}^e\}^T [\boldsymbol{K}^e] \{\Delta \boldsymbol{U}^e\} \tag{2.85}$$

where the tangent stiffness matrix $[\boldsymbol{K}^e] = [\boldsymbol{K}^e_M] + [\boldsymbol{K}^e_G]$ is the sum of the material and geometric stiffness matrices given by:

$$[\boldsymbol{K}^e_M] = \sum_k w_k [\boldsymbol{B}^{UL}]^T [\boldsymbol{c}] [\boldsymbol{B}^{UL}] J_{x\xi}(\boldsymbol{\xi}_k) = \sum_k w_k [\boldsymbol{B}^{TL}]^T [\boldsymbol{\mathcal{C}}] [\boldsymbol{B}^{TL}] J_{X\xi}(\boldsymbol{\xi}_k) \tag{2.86}$$

$$[\boldsymbol{K}^e_G] = \sum_k w_k [\boldsymbol{G}^{UL}] J_{x\xi}(\boldsymbol{\xi}_k) = \sum_k w_k [\boldsymbol{G}^{TL}] J_{X\xi}(\boldsymbol{\xi}_k) \tag{2.87}$$

The geometric stiffness components $[\boldsymbol{G}^{UL}]$ and $[\boldsymbol{G}^{TL}]$ are given by:

$$[\boldsymbol{G}] = \begin{bmatrix} \ddots & & & \\ \ddots & \ddots & & \\ & [\boldsymbol{G}_{\alpha\beta}] & \ddots & \\ & & \ddots & \ddots \end{bmatrix} \quad ; \quad [\boldsymbol{G}_{\alpha\beta}] = \begin{bmatrix} G_{\alpha\beta} & 0 & 0 \\ 0 & G_{\alpha\beta} & 0 \\ 0 & 0 & G_{\alpha\beta} \end{bmatrix} \tag{2.88}$$

where

$$\begin{aligned} G^{UL}_{\alpha\beta} &= \sigma_{ij} \frac{1}{2} \left(\frac{\partial N_\beta}{\partial x_i} \frac{\partial N_\alpha}{\partial x_j} + \frac{\partial N_\alpha}{\partial x_i} \frac{\partial N_\beta}{\partial x_j} \right) \\ G^{TL}_{\alpha\beta} &= S_{IJ} \frac{1}{2} \left(\frac{\partial N_\beta}{\partial X_I} \frac{\partial N_\alpha}{\partial X_J} + \frac{\partial N_\alpha}{\partial X_I} \frac{\partial N_\beta}{\partial X_J} \right) \end{aligned} \tag{2.89}$$

These expressions for the geometric stiffness components are obtained from the second term in the expressions (2.56) and (2.60):

$$\begin{aligned} \boldsymbol{S} : \frac{1}{2} \left((\Delta \boldsymbol{F})^T \tilde{\boldsymbol{F}} + \tilde{\boldsymbol{F}}^T (\Delta \boldsymbol{F}) \right) &= S_{IJ} \frac{1}{2} \left((\Delta F)_{iI} \tilde{F}_{iJ} + \tilde{F}_{iI} (\Delta F)_{iJ} \right) \\ &= S_{IJ} \frac{1}{2} \left((\Delta u)^e_{i\beta} \frac{\partial N_\beta}{\partial X_I} \tilde{u}^e_{i\alpha} \frac{\partial N_\alpha}{\partial X_J} + \tilde{u}^e_{i\alpha} \frac{\partial N_\alpha}{\partial X_I} (\Delta u)^e_{i\beta} \frac{\partial N_\beta}{\partial X_J} \right) \\ &= \tilde{u}^e_{i\alpha} \left[S_{IJ} \frac{1}{2} \left(\frac{\partial N_\beta}{\partial X_I} \frac{\partial N_\alpha}{\partial X_J} + \frac{\partial N_\alpha}{\partial X_I} \frac{\partial N_\beta}{\partial X_J} \right) \right] (\Delta u)^e_{i\beta} \end{aligned} \tag{2.90}$$

$$\begin{aligned} \boldsymbol{\sigma} : \frac{1}{2} &\left((\nabla_x (\Delta \boldsymbol{u}))^T \nabla_x \tilde{\boldsymbol{u}} + (\nabla_x \tilde{\boldsymbol{u}})^T \nabla_x (\Delta \boldsymbol{u}) \right) \\ &= \sigma_{ij} \frac{1}{2} \left((\Delta u)^e_{k\beta} \frac{\partial N_\beta}{\partial x_i} \tilde{u}^e_{k\alpha} \frac{\partial N_\alpha}{\partial x_j} + \tilde{u}^e_{k\alpha} \frac{\partial N_\alpha}{\partial x_i} (\Delta u)^e_{k\beta} \frac{\partial N_\beta}{\partial x_j} \right) \\ &= \tilde{u}^e_{k\alpha} \left[\sigma_{ij} \frac{1}{2} \left(\frac{\partial N_\beta}{\partial x_i} \frac{\partial N_\alpha}{\partial x_j} + \frac{\partial N_\alpha}{\partial x_i} \frac{\partial N_\beta}{\partial x_j} \right) \right] (\Delta u)^e_{k\beta} \end{aligned} \tag{2.91}$$

The FEM discretization leads to the following semi-discrete form of the principle of virtual work (2.33) and (2.37):

$$\mathscr{G}(\boldsymbol{U}, \tilde{\boldsymbol{U}}) \approx \bar{\mathscr{G}}(\boldsymbol{U}, \tilde{\boldsymbol{U}}) = \tilde{\boldsymbol{U}}^T \boldsymbol{R}(\boldsymbol{U}(t)) = 0 \qquad \forall \tilde{\boldsymbol{U}} \in \mathbb{R}^{nd} \tag{2.92}$$

where $\boldsymbol{R}(\boldsymbol{U}(t))$ and $\boldsymbol{U}(t)$ represent the global residual and nodal displacement vectors respectively and nd is the total number of dofs. Satisfying the principle of virtual work now becomes a matter of solving a system of non-linear ordinary differential equations (ODEs):

$$\boldsymbol{R}(\boldsymbol{U}(t)) = \boldsymbol{M}\ddot{\boldsymbol{U}}(t) + \boldsymbol{Q}(\boldsymbol{U}(t)) - \boldsymbol{P}(t) = 0 \tag{2.93}$$

where $\boldsymbol{M}$, $\boldsymbol{Q}$ and $\boldsymbol{P}$ represent the global mass matrix, internal and external force vectors respectively obtained by assembling the element contributions:

$$\begin{aligned} \boldsymbol{M} &= \sum_{e=1}^{m} \boldsymbol{A}^{e^T}[\boldsymbol{M}^e]\boldsymbol{A}^e \\ \boldsymbol{Q}(\boldsymbol{U}(t)) &= \sum_{e=1}^{m} \boldsymbol{A}^{e^T}\{\boldsymbol{Q}^e(\boldsymbol{U}(t))\} \\ \boldsymbol{P} &= \sum_{e=1}^{m} \boldsymbol{A}^{e^T}\{\boldsymbol{P}^e\} \end{aligned} \tag{2.94}$$

For *linear* problems, the internal force, in general, depends on both displacement and velocity and can be obtained as:

$$\boldsymbol{Q}(\dot{\boldsymbol{U}}, \boldsymbol{U}) = \boldsymbol{D}\,\dot{\boldsymbol{U}} + \boldsymbol{K}\,\boldsymbol{U} \tag{2.95}$$

where $\boldsymbol{K} = \dfrac{\partial \boldsymbol{Q}}{\partial \boldsymbol{U}}$ denotes the stiffness matrix and $\boldsymbol{D} = \dfrac{\partial \boldsymbol{Q}}{\partial \dot{\boldsymbol{U}}}$ denotes the damping matrix. The stiffness matrix $\boldsymbol{K}$ is obtained by assembling the elemental contributions:

$$\boldsymbol{K} = \sum_{e=1}^{m} \boldsymbol{A}^{e^T}[\boldsymbol{K}^e]\boldsymbol{A}^e \tag{2.96}$$

Ideally, the damping matrix should also be obtained from a *rate-dependent* constitutive relationship in a manner similar to the stiffness matrix, but other forms of damping such as *Raleigh* damping $\boldsymbol{D} = \alpha_1\boldsymbol{M} + \alpha_2\boldsymbol{K}$ ($\alpha_1,\ \alpha_2 \in \mathbb{R}$) are also commonly employed. The

resulting system of ordinary differential equations (ODEs) for a linear problem is:

$$\boldsymbol{M}\,\ddot{\boldsymbol{U}}(t) + \boldsymbol{D}\,\dot{\boldsymbol{U}}(t) + \boldsymbol{K}\,\boldsymbol{U}(t) = \boldsymbol{P}(t) \tag{2.97}$$

In addition, one needs initial conditions and boundary conditions to complete the problem statement:

$$\begin{aligned} &\boldsymbol{U}(0) = \boldsymbol{U}_0 \qquad \dot{\boldsymbol{U}}(0) = \boldsymbol{V}_0 \\ &\boldsymbol{U}(t) = \bar{\boldsymbol{U}}(t) \quad \text{on } \Gamma_D(t) \qquad \forall t \in [t_0, t_F] \end{aligned} \tag{2.98}$$

where $\boldsymbol{U}_0$ and $\boldsymbol{V}_0$ are the specified initial displacement and velocity vectors respectively and $\bar{\boldsymbol{U}}$ denotes the specified displacement for the Dirichlet boundary condition (2.30) from section §2.4.

2.8.2 Temporal Discretization with Newmark Method

The semi-discrete system of ordinary differential equations (2.93) or (2.97) are solved approximately by numerical time integration algorithms. Most commonly, a finite difference (FD) scheme is used to enforce the equations at discrete instants of time. The time interval of interest $I = [t_0, t_F]$ is divided into N time steps of sizes Δt_i for $1 \leq i \leq N$ and the equations are enforced at discrete instants of time t_n given by $t_n = t_0 + \sum_{i=1}^{n} \Delta t_i$ with $t_0 = 0$ and $t_N = t_F$. Assuming that the complete state of the system is known at the instant t_n, advancing the state by the time step Δt_{n+1} to obtain the state at t_{n+1} is known as *time-stepping*.

The fully discrete system of algebraic equations at t_{n+1} can be written as:

$$\boldsymbol{R}_{n+1}(\boldsymbol{U}_{n+1}, \dot{\boldsymbol{U}}_{n+1}) = \boldsymbol{M}\ddot{\boldsymbol{U}}_{n+1} + \boldsymbol{Q}_{n+1}(\boldsymbol{U}_{n+1}, \dot{\boldsymbol{U}}_{n+1}) - \boldsymbol{P}_{n+1} = 0 \tag{2.99}$$

In addition, two more equations are needed to approximate the time derivatives for velocity

$\dot{\boldsymbol{U}}_{n+1}$ and acceleration $\ddot{\boldsymbol{U}}_{n+1}$. The *Newmark* family of time integration schemes [13] are widely used in structural dynamics to approximate this relationship as:

$$\dot{\boldsymbol{U}}_{n+1} = \hat{\dot{\boldsymbol{U}}}_{n+1} + \Delta t\, \gamma\, \ddot{\boldsymbol{U}}_{n+1} \quad ; \quad \hat{\dot{\boldsymbol{U}}}_{n+1} = \dot{\boldsymbol{U}}_n + \Delta t\, (1-\gamma)\, \ddot{\boldsymbol{U}}_n \tag{2.100}$$

$$\boldsymbol{U}_{n+1} = \hat{\boldsymbol{U}}_{n+1} + \Delta t^2\, \beta\, \ddot{\boldsymbol{U}}_{n+1} \quad ; \quad \hat{\boldsymbol{U}}_{n+1} = \boldsymbol{U}_n + \Delta t\, \dot{\boldsymbol{U}}_n + \Delta t^2\, (\tfrac{1}{2} - \beta) \ddot{\boldsymbol{U}}_n \tag{2.101}$$

where γ and β are algorithmic parameters. The fully discrete, non-linear systems of algebraic equations (2.99), (2.100) and (2.101) are solved using the Newton's method. An initial guess (iteration $i = 0$) for the state at t_{n+1} is assumed as:

$$\ddot{\boldsymbol{U}}^0_{n+1} = \ddot{\boldsymbol{U}}^*_n \tag{2.102}$$

$$\dot{\boldsymbol{U}}^0_{n+1} = \dot{\boldsymbol{U}}^*_n + \Delta t \left[(1-\gamma) \ddot{\boldsymbol{U}}^*_n + \gamma\, \ddot{\boldsymbol{U}}^0_{n+1} \right] \tag{2.103}$$

$$\boldsymbol{U}^0_{n+1} = \boldsymbol{U}^*_n + \Delta t\, \dot{\boldsymbol{U}}^*_n + \Delta t^2 \left[(\tfrac{1}{2} - \beta) \ddot{\boldsymbol{U}}^*_n + \beta\, \ddot{\boldsymbol{U}}^0_{n+1} \right] \tag{2.104}$$

from the converged iteration $i = *$ of the previous time step t_n. The linearized system of equations for the Newton update:

$$\boldsymbol{M}\, \Delta \ddot{\boldsymbol{U}}_{n+1} + \boldsymbol{D}^i_{n+1}\, \Delta \dot{\boldsymbol{U}}_{n+1} + \boldsymbol{K}^i_{n+1}\, \Delta \boldsymbol{U}_{n+1} = -\boldsymbol{R}^i_{n+1}(\boldsymbol{U}^i_{n+1}, \dot{\boldsymbol{U}}^i_{n+1}) \tag{2.105}$$

$$\Delta \dot{\boldsymbol{U}}_{n+1} \;=\; \Delta t\, \gamma\, \Delta \ddot{\boldsymbol{U}}_{n+1} \tag{2.106}$$

$$\Delta \boldsymbol{U}_{n+1} \;=\; \Delta t^2\, \beta\, \Delta \ddot{\boldsymbol{U}}_{n+1} \tag{2.107}$$

can be solved for $\Delta \ddot{\boldsymbol{U}}_{n+1}$, $\Delta \dot{\boldsymbol{U}}_{n+1}$ and $\Delta \boldsymbol{U}_{n+1}$ and the state for iteration $i+1$ can be updated using:

$$\ddot{\boldsymbol{U}}^{i+1}_{n+1} \;=\; \ddot{\boldsymbol{U}}^i_{n+1} + \Delta \ddot{\boldsymbol{U}}_{n+1} \tag{2.108}$$

$$\dot{\boldsymbol{U}}^{i+1}_{n+1} \;=\; \dot{\boldsymbol{U}}^i_{n+1} + \Delta \dot{\boldsymbol{U}}_{n+1} \tag{2.109}$$

$$\boldsymbol{U}^{i+1}_{n+1} \;=\; \boldsymbol{U}^i_{n+1} + \Delta \boldsymbol{U}_{n+1} \tag{2.110}$$

This iterative process is repeated until the residual $\|\boldsymbol{R}^i_{n+1}(\boldsymbol{U}^i_{n+1}, \dot{\boldsymbol{U}}_{n+1})\|$ is less than some tolerance ε_{TOL}. In contrast, the system of equations for a linear structural dynamics problem is:

$$\boldsymbol{M}\ddot{\boldsymbol{U}}_{n+1} + \boldsymbol{D}\,\dot{\boldsymbol{U}}_{n+1} + \boldsymbol{K}\,\boldsymbol{U}_{n+1} = \boldsymbol{P}_{n+1} \tag{2.111}$$

together with the Newmark relations (2.100) and (2.101). This system can be solved for $\ddot{\boldsymbol{U}}_{n+1}$ from the equation

$$\tilde{\boldsymbol{M}}\,\ddot{\boldsymbol{U}}_{n+1} = \boldsymbol{P}_{n+1} - \boldsymbol{D}\hat{\dot{\boldsymbol{U}}}_{n+1} - \boldsymbol{K}\hat{\boldsymbol{U}}_{n+1} \tag{2.112}$$

$$\text{where} \qquad \tilde{\boldsymbol{M}} = \big(\boldsymbol{M} + \gamma\Delta t\boldsymbol{D} + \beta\Delta t^2\boldsymbol{K}\big) \tag{2.113}$$

without the need for iteration.

Note that to start the time stepping one must compute the initial acceleration $\ddot{\boldsymbol{U}}_0$ by solving the residual equation at t_0:

$$\boldsymbol{M}\ddot{\boldsymbol{U}}_0 = \boldsymbol{P}_0 - \boldsymbol{Q}(\boldsymbol{U}_0, \boldsymbol{V}_0) \tag{2.114}$$

Newmark schemes are divided into two basic classes namely explicit and implicit. For explicit schemes usually $\beta = 0$ and the displacement is predicted solely on the basis of known values from the previous time step. Implicit schemes have $\beta > 0$ and require one to solve a system of equations at every time step. Explicit central difference with ($\beta = 0$; $\gamma = \frac{1}{2}$) and implicit constant-average-acceleration ($\beta = \frac{1}{4}$; $\gamma = \frac{1}{2}$) are the two most widely used schemes of the Newmark family. Other values of $\beta = \frac{1}{6}$ and $\frac{1}{12}$ result in the linear acceleration and the Fox-Goodwin schemes respectively.

An alternative implementation of the Newmark method to solve directly for $\boldsymbol{U}_{n+1}$ can

also be obtained:

$$\frac{1}{\Delta t^2}\boldsymbol{M}\boldsymbol{U}_{n+1} = \boldsymbol{P}_n - \left(\boldsymbol{K} - \frac{2}{\Delta t^2}\boldsymbol{M}\right)\boldsymbol{U}_n - \left(\frac{1}{\Delta t^2}\boldsymbol{M}\right)\boldsymbol{U}_{n-1} \tag{2.115}$$

$$\left(\boldsymbol{K} + \frac{4}{\Delta t^2}\boldsymbol{M}\right)\boldsymbol{U}_{n+1} = \boldsymbol{P}_{n+1} + \boldsymbol{M}\left(\frac{4}{\Delta t^2}\boldsymbol{U}_n + \frac{4}{\Delta t}\dot{\boldsymbol{U}}_n + \ddot{\boldsymbol{U}}_n\right) \tag{2.116}$$

are the systems of equations for the explicit central difference and the implicit constant average acceleration schemes respectively.

2.9 Review of Time Integration Schemes

Traditionally ODEs involving time derivatives have been treated as initial value problems (see Chapter 9, [14]). A general system of ODEs can be written in the form:

$$\{\dot{\boldsymbol{y}}\} = \{\boldsymbol{f}\,(t, \{\boldsymbol{y}\})\} \tag{2.117}$$

where $\{\boldsymbol{y}\}$ is vector of quantities $\boldsymbol{y}_1$, $\boldsymbol{y}_2$, ... $\boldsymbol{y}_n$ which themselves may be vectors. A system of ODEs of any order may be converted into the form (2.117) by introducing new variables $\boldsymbol{y}_{n+1} = \dot{\boldsymbol{y}}_n$. In particular, the system of ODEs (2.93) and (2.97) are of second order and can be converted into first order form (2.117) by substituting $\boldsymbol{y}_1 = \boldsymbol{U}$ and $\boldsymbol{y}_2 = \dot{\boldsymbol{U}}$.

There are a variety of methods available in the literature for solving a system of ODEs with FD schemes based on Taylor series expansions of the unknown quantities. One such method as the *Euler* method:

$$\begin{aligned} \{\boldsymbol{y}\}_{n+1} &= \{\boldsymbol{y}\}_n + \Delta t\ \{\dot{\boldsymbol{y}}\}_{n+\alpha} \\ \{\dot{\boldsymbol{y}}\}_{n+\alpha} &= [(1-\alpha)\{\dot{\boldsymbol{y}}\}_n + \alpha\ \{\dot{\boldsymbol{y}}\}_{n+1}] \end{aligned} \tag{2.118}$$

Substituting expressions (2.118) into equation (2.117), one can solve for $\{\boldsymbol{y}\}_{n+1}$ or $\{\dot{\boldsymbol{y}}\}_{n+\alpha}$ depending upon the implementation. The Euler method is a particular *one-step* case of a

more general family of methods known as *Linear Multi-step* (LMS) methods (see Chapter 9, [11]). LMS methods allow one to use information from several previous time-steps (t_{n-k}, t_{n-k+1} ... t_n) to compute the next state at t_{n+1}:

$$\sum_{i=0}^{k} [\alpha_i \{\boldsymbol{y}\}_{n-i+1} + \Delta t \; \beta_i \{\boldsymbol{f}\left(t_{n-i+1}, \{\boldsymbol{y}\}_{n-i+1}\right)\}] = 0 \tag{2.119}$$

where α_i and β_i are algorithmic parameters. Another class of schemes based on FD are the *Runge-Kutta* Methods which approximate the effect of the derivatives $\dot{\boldsymbol{y}}_k$ by evaluating the function $\boldsymbol{f}$ at several locations within the time step t_n to t_{n+1}. A popular choice is the fourth-order accurate Runge-Kutta method:

$$\begin{aligned}
\{\boldsymbol{f}_1\} &= \Delta t \, \{\boldsymbol{f}(t_n, \{\boldsymbol{y}\}_n)\} \\
\{\boldsymbol{f}_2\} &= \Delta t \, \{\boldsymbol{f}(t_n + \frac{\Delta t}{2}, \{\boldsymbol{y}\}_n + \frac{\{\boldsymbol{f}_1\}}{2})\} \\
\{\boldsymbol{f}_3\} &= \Delta t \, \{\boldsymbol{f}(t_n + \frac{\Delta t}{2}, \{\boldsymbol{y}\}_n + \frac{\{\boldsymbol{f}_2\}}{2})\} \\
\{\boldsymbol{f}_4\} &= \Delta t \, \{\boldsymbol{f}(t_n + \Delta t, \{\boldsymbol{y}\}_n + \{\boldsymbol{f}_3\})\} \\
\{\boldsymbol{y}\}_{n+1} &= \{\boldsymbol{y}\}_n + \frac{1}{6} \left[\{\boldsymbol{f}_1\} + 2\{\boldsymbol{f}_2\} + 2\{\boldsymbol{f}_3\} + \{\boldsymbol{f}_4\}\right] \qquad + O(\Delta t^5)
\end{aligned} \tag{2.120}$$

The time step Δt can be adapted using an approximate error measure by re-solving for $\{\boldsymbol{y}\}_{n+1}$ by taking two steps of $\Delta t/2$ each and comparing the results.

In structural dynamics, however, methods that directly integrate second order ODEs are preferred over the solvers for the first order ODEs since they require less storage and compute the desired quantities directly. Such methods are also abundant in the literature. One may refer to the texts by Hughes (Chapter 9, [11]), Zienkiewicz and Taylor (Chapter 18, [12]) and Bathe [15] for a summary and analysis of methods such as the Houbolt method, Wilson-θ method, Park's method, Generalized Newmark methods etc.

More recently, Chung & Hulbert [16] presented a Generalized-α method:

$$\boldsymbol{M}\,\ddot{\boldsymbol{U}}_{n+1-\alpha_m} + \boldsymbol{K}\,\boldsymbol{U}_{n+1-\alpha_f} = \boldsymbol{P}_{n+1-\alpha_f} \tag{2.121}$$

$$(\cdot)_{n+1-\alpha} = (1-\alpha)(\cdot)_{n+1} + \alpha(\cdot)_n \tag{2.122}$$

which has controllable numerical dissipation to damp out artificial high-frequency oscillations in the response. $\alpha_m = \alpha_f = 1/2$ gives the second order accurate, unconditionally stable midpoint rule. Several other algorithms such as HHT-α [17], θ-Collocation schemes, WBZ-α [18] are contained in this method as special cases.

2.9.1 Analysis of Time Integrators

Traditionally, the performance of time integrators, with respect to *stability* and *accuracy*, is usually studied for linear or linearized systems. For linear systems, one can use *modal decomposition* to reduce the global coupled nd equations to nd independent scalar equations through the generalized eigen-value problem:

$$[\boldsymbol{K} - \boldsymbol{\Lambda}\boldsymbol{M}]\,\boldsymbol{\Phi} = 0 \tag{2.123}$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix of the square of the frequencies ω_i^2 with the corresponding eigen-modes $\boldsymbol{\phi}_i$ contained in the columns of the matrix $\boldsymbol{\Phi}$. Each scalar equation can be integrated using a time-stepping scheme of interest to get a relation between the state $\boldsymbol{x}_n = \{\dot{u}_n, u_n\}^T$ and the state $\boldsymbol{x}_{n+1} = \{\dot{u}_{n+1}, u_{n+1}\}^T$:

$$\boldsymbol{x}_{n+1} = \boldsymbol{A}\boldsymbol{x}_n + \boldsymbol{L}_n \tag{2.124}$$

where $\boldsymbol{A}$ is the *amplification matrix*. The condition for *stability* is that the maximum eigen-value (spectral radius) $\rho(\boldsymbol{A}) \leq 1$. In order to study the accuracy of a method, one can

substitute the exact continous solution which, in general, does not satisfy the discrete equation (2.124) and one gets a residual error term:

$$\boldsymbol{x}(t_{n+1}) = \boldsymbol{A}\boldsymbol{x}(t_n) + \boldsymbol{L}_n + \Delta t\, \boldsymbol{\tau}(t_n) \tag{2.125}$$

where $\boldsymbol{\tau}(t)$ is the called the *local truncation error*. Subtracting equation (2.125) from (2.124) one obtains the error equation:

$$\boldsymbol{e}(t_{n+1}) = \boldsymbol{A}\, \boldsymbol{e}(t_n) - \Delta t\, \boldsymbol{\tau}(t_n) \tag{2.126}$$

which after successive substitution for $\boldsymbol{e}(t_n)$ $\boldsymbol{e}(t_{n-1})$ ... $\boldsymbol{e}(t_0)$ leads to the general error equation:

$$\boldsymbol{e}(t_{n+1}) = \boldsymbol{A}^{n+1}\, \boldsymbol{e}(t_0) - \sum_{i=0}^{n} \Delta t\, \boldsymbol{A}^i\, \boldsymbol{\tau}(t_{n-i}) \tag{2.127}$$

If $\boldsymbol{e}(t_0) = 0$ and $\rho(\boldsymbol{A}) \leq 1$ then the error estimate reduces to:

$$\boldsymbol{e}(t_{n+1}) \leq t_n \max|\boldsymbol{\tau}(t)| \tag{2.128}$$

The expression for $\boldsymbol{\tau}(t)$ is obtained by comparing Taylor series expansions of $\dot{u}_n(t)$ and $u(t)$ around t_n and t_{n+1} respectively. If $|\boldsymbol{\tau}(t)| \leq c\Delta t^k$ where c is independent of Δt and $k \geq 1$ then the scheme is said to be *consistent*. *Stability* together with *consistency* results in *accuracy* or *convergence* and k is the called the order of accuracy or the rate of convergence.

Using this analysis one concludes that Newmark-β methods are 2nd order accurate for $\gamma = 1/2$. Trapezoidal rule is *unconditionally stable* and the time step is restricted only by accuracy requirements. The stable time step for the explicit Central difference method is given by the *Courant* condition:

$$\Delta t \leq \Delta t_{cr} = \frac{2}{\omega_{max}} \tag{2.129}$$

where ω_{max} is the highest undamped natural frequency. For most practical meshes, this is rarely known so an approximation based on the highest undamped natural frequency of its constituent elements ω^{el}_{max} is adopted. It can be proved [11] that $\omega^{el}_{max} > \omega_{max}$ so replacing ω_{max} by ω^{el}_{max} in (2.129) results in a conservative estimate for the stable time step. ω^{el}_{max} can be computed for most elements as:

$$\omega^{el}_{max} = \frac{2c}{L} \tag{2.130}$$

where c is the wave speed given by $\sqrt{E/\rho}$ where E is the Young's Modulus, ρ is the density and L is some characteristic length associated with the element. Substituting (2.130) into (2.129) one obtains the estimate (1.2).

2.9.2 Recent Trends in Time Integration

Researchers in the field of time integration continue to investigate and develop methods that have better properties such as stability, accuracy with respect to certain underlying mathematical structure and computational efficiency. A very brief list of some works worth note is presented.

Simo and co-workers [19, 20, 21] developed *Energy, Momentum, Symplectic* form preserving integrators. They have shown that the mid-point rule exactly conserves energy and the norm of the angular momentum but fails to conserve the direction of the angular momentum vector and propose alternative family of algorithms which conserve total energy and total angular momentum. Armero and Romero [22, 23] extended this energy-momentum method to include controllable numerical dissipation.

Discrete Variational integrators were introduced by Veselov [24] and studied further by Marsden and co-workers [25]. The text by Hairer [26] provides extensive literature on numerical and variational integrators from a variety of fields. Kane et al. [27] provide an introduction to the variational integrators and show that the Newmark family of algorithms

is variational. Central Difference ($\gamma = 1/2$, $\beta = 0$) is symplectic and momentum preserving. They also state that energy-momentum algorithms proposed by Simo and others generally fail to be symplectic. Lew, Ortiz, Marsden and West [28] have extended variational integrators to elasto-dynamics and introduced the concept of *asynchronous* variational integrators (AVI). This allows individual elements to be integrated with different time steps and leads to a formulation that resembles *spacetime* formulations. The AVI corresponding to the explicit central difference can be integrated element-wise and leads to efficient computation, but in general, implicit AVIs are coupled in time and lead to huge computational costs.

Researchers have also studied *spacetime* formulations using time-discontinuous Galerkin finite elements in time. Building on the work of Johnson [29], Hughes and Hulbert [30, 31] extended the idea of space-time finite elements to elasto-dynamics. Li and Wiberg [32], Mancuso and Ubertini [33] and Haber and co-workers [34] have also contributed to this area.

Betsch and Steinmann [35] have developed *time finite elements* and compared the same with Energy-Momentum and Variational integrators. Tamma and co-workers [36] have developed a methodology to construct general time integration operators that lead to certain desired properties of the discrete integrator.

Chapter 3

Domain Decomposition and Coupling Methods

Large-scale problems commonly involve elements with sizes varying over several orders of magnitude. The choice of a particular time-stepping scheme and time-step size that would be suitable for the entire mesh in such cases is very difficult. In a large mesh, it is usually easier to identify zones of elements with similar time step requirements. A coupled approach based on domain decomposition (DD) that facilitates the use of different time stepping schemes and/or time steps in different regions of a mesh while preserving the global accuracy and stability of the solution is the natural way to solve large-scale problems.

Using domain decomposition, a large mesh can be divided into several subdomains depending upon element sizes and time step requirements. For dynamics, implicit integration can be used for those subdomains of the mesh where the time step is not limited by solution accuracy and explicit integration can be used elsewhere to save computational time. Since actual time-steps can vary widely between different subdomains, the computational overhead involved with coupling them bears greatly on the efficiency of this approach.

3.1 Domain Decomposition Methods

Historically domain decomposition (DD) methods have been used to solve coupled multi-field problems such as fluid-structure interaction (FSI). However, recently, with the advent of parallel computers, researchers have started using DD methods to divide their computation into multiple smaller problems which can be distributed over several processors and solved in parallel. The text by Toselli and Widlund [37], the survey article by Fragakis and Pa-

padrakakis [38] and several conference proceedings [39] provide a good summary of various DD methods in the literature.

A basic classification of DD methods is done into *overlapping* and *non-overlapping* DD methods. As the name suggest, overlapping DD methods divide the problem domain into subdomains that have a partial overlap between them such as the *Schwarz alternating* method [40]. The subdomains are divided into two groups which are solved one at a time using boundary conditions from the other. The process is repeated until the solution in the overlap converges. The presence of an overlap between the subdomains usually helps in stability of such coupling methods. Non-overlapping methods lead to interfaces of a lower dimension between subdomains. A *coarse* problem on the interface is solved to enforce the continuity of the solution across these interfaces. The discussion in this document is restricted to non-overlapping DD methods.

Non-overlapping domain decomposition methods are commonly found in structural anlysis like the substructuring techniques [41] based on the *Schur* complement approach [42]. Substructuring falls under the class of methods commonly known as *primal* Schur complement methods where the interface variable for the coarse problem is a primal variable such as the displacement, velocity or acceleration. Another primal Schur complement method is the *balancing* domain decomposition (BDD) method [43] where the coarse problem is solved iteratively using a preconditioned conjugate gradient (PCG) method for better convergence. An alternative to the primal method is the *dual* Schur complement method where the variable for the interface problem is a dual variable such as inter-subdomain traction or a Lagrange multiplier. Dual Schur complement methods will be discussed in greater detail in the following sections.

3.1.1 Partitioning Approaches

In structural dynamics, researchers have used non-overlapping methods to couple implicit and explicit schemes within the same mesh to avoid using a very small time step in the fine

regions of the mesh. Such approaches use either node partitioning or element partitioning to divide the mesh into subdomains.

In node partitioning, the variables associated with each node are taken to be either explicit or implicit as depicted in figure 3.1. The interface elements, associated with both

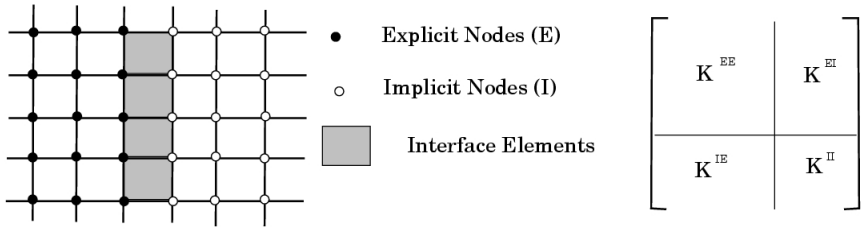

Figure 3.1: Node partitioning.

implicit and explicit nodes are responsible for coupling and have to be treated separately from the normal explicit and implicit elements. This procedure amounts to simply dividing the system matrix into disjoint halves with the off diagonal terms representing the interface elements. Node partitioning, in general, results in complex coupling algorithms which often include additional book-keeping for the interface elements and component systems which are asymmetric and hence more expensive to solve. In a particular case [44], this method requires the use of a zone of interface elements where the size of this zone grows with the ratio of the time steps between the subdomains. The situation is further complicated if there are interface elements which are shared by more than two subdomains as might be the case in 2-D and 3-D problems. In addition, computational overheads are large if the interface has to be modified continuously as in the case of adaptive mesh refinement. Hence this approach is not suitable for coupling explicit and implicit schemes for a large time step ratio between the subdomains.

In element partitioning, the structure is divided into two or more subdomains and all the elements in a particular subdomain are treated either explicitly or implicitly. The boundary between any two subdomains comprises *shared nodes* as shown in figure 3.2. The component system matrices can be assembled from the elements of the corresponding subdomains where

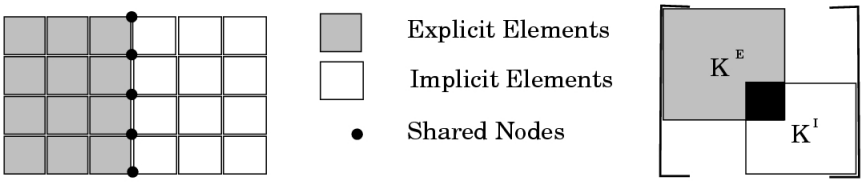

Figure 3.2: Element partitioning.

the overlap represents shared nodes which couple the individual solutions together. Coupling ensures continuity of the solution across the interface and equilibrates tractions between the subdomains. Element partitioning can handle shared nodes with more than two subdomains easily because the component system matrices are still assembled the same way. One may also use separate existing codes for solving different subdomains and write a small additional subroutine for coupling. Another important advantage of this method is that the interface can be moved easily by simply changing a few elements from explicit to implicit and vice versa. The process of subdomain selection may also be automated based on some set criterion such as element size which is beneficial in adaptive mesh refinement. As one adaptively refines the mesh, the selection of different zones can be done automatically without human intervention.

3.2 Primal Coupling Methods

In literature one finds several methods for coupled analysis that use primal substructuring. Most coupling methods based on substructuring are designed using a uniform time step for all the subdomains. Such methods, often called *mixed time* or *implicit-explicit* (*I-E*) integration methods, use the same time step throughout the mesh but different integration schemes in different subdomains. For example, implicit schemes are used for subdomains with stiff meshes to avoid the Courant time step limit and explicit integration is used for subdomains with flexible meshes to reduce computational cost. This approach is usually adopted for hyperbolic problems like dynamic structure-media interaction where the stiff

structural mesh is integrated implicitly and the flexible media mesh is integrated explicitly. However, as discussed earlier, for large complex meshes this approach is inadequate and one needs to use different time steps in different subdomains.

Multi-time-step and *Subcycling* methods allow the use of different time steps in different subdomains. Substructuring based multi-time step and subcycling methods usually lead to complex procedures for time step ratios more than one. Moreover, the stability and accuracy analysis for such methods is substantially more involved.

I-E finite elements: Hughes and Liu [45] developed a coupling method that uses element partitioning and an explicit predictor-corrector scheme constructed from an implicit Newmark scheme. In this method, the acceleration is computed from the modified equation of motion as follows:

$$\boldsymbol{M}\ddot{\boldsymbol{U}}_{n+1} + \boldsymbol{C}\hat{\dot{\boldsymbol{U}}}_{n+1} + \boldsymbol{K}\hat{\boldsymbol{U}}_{n+1} = \boldsymbol{P}_{n+1} \tag{3.1}$$

which is used to update the displacement and velocity vectors using the Newmark relations (2.100) and (2.101). The I-E algorithm is then formulated using another modified equation of motion:

$$\boldsymbol{M}\ddot{\boldsymbol{U}}_{n+1} + \boldsymbol{C}^I\dot{\boldsymbol{U}}_{n+1} + \boldsymbol{C}^E\hat{\dot{\boldsymbol{U}}}_{n+1} + \boldsymbol{K}^I\boldsymbol{U}_{n+1} + \boldsymbol{K}^E\hat{\boldsymbol{U}}_{n+1} = \boldsymbol{P}_{n+1} \tag{3.2}$$

which is solved for $\ddot{\boldsymbol{U}}_{n+1}$ and the displacement and velocity vectors updated using Newmark relations.

The authors conducted a stability analysis using the energy method (see Chapter 9, [11]) for both, the explicit predictor-corrector scheme and the I-E scheme. They concluded that, for the explicit predictor corrector method, the limiting time step was inversely proportional to the damping. The I-E method reduced to the coupling of central difference and average acceleration methods with limiting time step in the explicit subdomain governed by the

Courant limit for the undamped case. These analytical results were verified in [46] and extended the method for non-linear systems in [47].

A proof of convergence was given in [48] showing that the norm of the error remained bounded and the rate of convergence for undamped systems with $\gamma = \frac{1}{2}$ was 2^{nd} order. 2^{nd} order convergence could also be achieved for damped systems with additional modifications. Another improvement of the I-E method was proposed in [49] to control numerical dissipation of the unwanted high frequency oscillations using a method similar to the HHT-α method [17]. Belytschko and Lu [50] presented a general methodology of stability analysis for the same using Fourier methods.

Mixed-Time I-E finite elements: Liu and Belytschko [51] outlined an mE-I partition where the implicit elements are updated once using a big time step ΔT and the explicit elements are updated m times using a small time step Δt where $\Delta T = m\Delta t$. The predictor corrector algorithm developed by Hughes and Liu [45] was used for explicit computations and the Newmark method for implicit computations. Stability was verified numerically.

I-E mesh partitions: Belytschko and Mullen [44] proposed an I-E coupling algorithm using node partitioning. In this method, the explicit partition is updated first followed by the implicit partition. The implicit scheme uses the computed displacements from the explicit nodes in the interface elements to correctly update the implicit nodes:

$$\begin{bmatrix} \boldsymbol{M}^E & 0 \\ 0 & \boldsymbol{M}^I \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{U}}^E_{n+1} \\ \ddot{\boldsymbol{U}}^I_{n+1} \end{bmatrix} + \begin{bmatrix} \boldsymbol{K}^{EE} & \boldsymbol{K}^{EI} \\ \boldsymbol{K}^{IE} & \boldsymbol{K}^{II} \end{bmatrix} \begin{bmatrix} \boldsymbol{U}^E_{n+1} \\ \boldsymbol{U}^I_{n+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{P}^E_{n+1} \\ \boldsymbol{P}^I_{n+1} \end{bmatrix} \tag{3.3}$$

The central difference scheme (2.115) applied to the explicit partition (first row above) for time step t_n gives:

$$\boldsymbol{U}^E_{n+1} = 2\boldsymbol{U}^E_n - \boldsymbol{U}^E_{n-1} - \Delta t^2 \boldsymbol{M}^{E^{-1}} (\boldsymbol{P}^E_n - \boldsymbol{K}^{EE}\boldsymbol{U}^E_n - \boldsymbol{K}^{EI}\boldsymbol{U}^I_n) \tag{3.4}$$

This solution for the explicit partition $\boldsymbol{U}^E_{n+1}$ is then used for the constant average acceleration

method in the implicit partition as:

$$\left(\boldsymbol{M}^I + \beta\Delta t^2 \boldsymbol{K}^{II}\right) \ddot{\boldsymbol{U}}^I_{n+1} = \boldsymbol{P}^I_{n+1} - \boldsymbol{K}^{IE}\boldsymbol{U}^E_{n+1} - \boldsymbol{K}^{II}\hat{\boldsymbol{U}}^I_{n+1} \tag{3.5}$$

In this manner, the solution is advanced one step at a time. Note that this procedure has to be modified to include more interface elements for the case of time step ratio > 1. Belytschko and Mullen [52] carried out a stability analysis of the method for a time step ratio of 1 using the energy method (see Chapter 9, [11]) and concluded that the procedure is stable if the Courant limit is satisfied for the explicit partition.

Mixed and Multi-time Methods: Belytschko, Yen and Mullen [53] described E-E, I-E, E^m-E and I-I partitions using linear multi-step (LMS) formulae of the form:

$$\begin{aligned} \boldsymbol{U}_{n+1} &= \beta_0 \ddot{\boldsymbol{U}}_{n+1} + \boldsymbol{h}^u_n \quad \text{for implicit} \\ \boldsymbol{U}_{n+1} &= \beta_1 \ddot{\boldsymbol{U}}_n + \alpha_1 \boldsymbol{U}_n + \boldsymbol{h}^u_{n-1} \quad \text{for explicit} \end{aligned} \tag{3.6}$$

where $\boldsymbol{h}$ vectors depend only on the solution computed at earlier time steps. They show that E-E partitions can be computed in any order since the next state depends only on the previous time steps, however, for I-E partitions, explicit must be computed before implicit. They also present a new E^m-E partition which allows one of the subdomains to be integrated with a time step that is an integer multiple of the other i.e. $\Delta T = m\Delta t$. Here the subdomain (E^m) with time step ΔT is computed first followed by linear interpolation of the solution at the interface and solutio of the subdomain (E) with time step Δt. Lastly, they show that the I-I partition cannot be solved without extrapolating the interface quantities for one of the subdomains.

Their stability analysis confirms the results from previous I-E mesh partitions and shows that E^m-E partitions have more stringent stability requirements than the I-E method. In addition, the critical time step also depends on the relative stiffness of the two subdomains which makes this method unsuitable in general cases. They also found that the I-I method

with extrapolation is unconditionally *unstable* unless damping terms are included.

Partitioned Procedures: Park [54] described I-E and I-I partitioned procedures for coupled problems. The I-E procedure was demonstrated for structural dynamics and the I-I procedure for fluid-structure interaction problems. The I-E methods of both Belytschko and Hughes were shown to be special cases of the partitioned procedure. A general stability analysis considering the spectral radius of the amplification matrix was also conducted to find stable time steps.

Park and Felippa [55] also carried out an accuracy analysis based on the results from a two degree of freedom model problem. They show that the partitioned procedures are 2^{nd} order accurate but the frequency distortion (period elongation/contraction) is least for node-by-node partitions followed by element-by-element partitions and then for dof-by-dof partitions introduced in [54].

Multi-Stepping Subcycling Procedures: Belytschko, Smolinski and Liu [56] describe an I-E procedure for the first-order transient heat conduction problem using the Euler difference formula (2.118):

$$\boldsymbol{M}\dot{\boldsymbol{U}} + \boldsymbol{K}\boldsymbol{U} = \boldsymbol{P} \tag{3.7}$$

$$\boldsymbol{U}_{n+1} = \boldsymbol{U}_n + (1-\alpha)\Delta t\dot{\boldsymbol{U}}_n + \alpha\Delta t\dot{\boldsymbol{U}}_{n+1} \tag{3.8}$$

The domain equations are partitioned into two subdomains A and B with time steps ΔT and Δt respectively:

$$\dot{\boldsymbol{U}} = \dot{\boldsymbol{U}}^A + \dot{\boldsymbol{U}}^B; \quad \boldsymbol{P} = \boldsymbol{P}^A + \boldsymbol{P}^B; \quad \boldsymbol{K} = \boldsymbol{K}^A + \boldsymbol{K}^B \tag{3.9}$$

where $\Delta T = m\Delta t$. The resulting update equations are:

$$\begin{aligned} \boldsymbol{M}\dot{\boldsymbol{U}}^A &= \boldsymbol{P}^A - \boldsymbol{K}^A\boldsymbol{U} \\ \boldsymbol{M}\dot{\boldsymbol{U}}^B &= \boldsymbol{P}^B - \boldsymbol{K}^B\boldsymbol{U} \end{aligned} \tag{3.10}$$

Note that the mass matrix is not decomposed. This leads to a subcycling procedure where the solution is updated as:

Subcycling: for $(n+1) \bmod m \neq 0$

$$\boldsymbol{U}_{n+1} = \boldsymbol{U}_n + (1-\alpha_B)\Delta t \dot{\boldsymbol{U}}_n^B + \alpha_B \Delta t \dot{\boldsymbol{U}}_{n+1}^B \tag{3.11}$$

Total Update: for $(n+1) \bmod m = 0$

$$\begin{aligned} \boldsymbol{U}_{n+1} = \boldsymbol{U}_n &+ (1-\alpha_B)\Delta t \dot{\boldsymbol{U}}_n^B + \alpha_B \Delta t \dot{\boldsymbol{U}}_{n+1}^B \\ &+ (1-\alpha_A) m \Delta t \dot{\boldsymbol{U}}_n^A + \alpha_A m \Delta t \dot{\boldsymbol{U}}_{n+1}^A \end{aligned} \tag{3.12}$$

Stability analysis of this method showed that the subcycling procedure is stable as long as the Courant time step limits are satisfied within each subdomain individually. Belytschko and Lu [57] extended the first order subcycling method for second order problems. Neal and Belytschko [58] considered the case of subcycling for non-integer time-step ratios between the subdomains.

Smolinski [59] extended this subcycling technique using E^m-E mesh partitions for the 2^{nd} order equations of structural dynamics and conducted a stability analysis based on the energy method in [60] and [61]. The stability analysis concluded that the method is stable as long as the Courant time step limits are met within each subdomain. Further enhancements of the subcycling method for non-integer time-step ratios [62] and to implicit subcycling [63, 64] have also been explored.

Subcycled Newmark Scheme: Daniel [65] illustrated a subcycling procedure based on an I-I node partition using the Newmark method. The algorithm is implemented only in terms of the displacements as:

$$\begin{aligned} [\boldsymbol{M} + \beta \Delta t^2 \boldsymbol{K}]\boldsymbol{U}_{n+1} = &- [\boldsymbol{M} + (\tfrac{1}{2} - \gamma + \beta)\Delta t^2 \boldsymbol{K}]\boldsymbol{U}_{n-1} \\ &+ [2\boldsymbol{M} - (\tfrac{1}{2} + \gamma - 2\beta)\Delta t^2 \boldsymbol{K}]\boldsymbol{U}_n + \Delta t^2 \boldsymbol{P}_{eq} \end{aligned} \tag{3.13}$$

where $\boldsymbol{P}_{eq} = (\frac{1}{2} - \gamma + \beta)\boldsymbol{P}_{n-1} + (\frac{1}{2} + \gamma - 2\beta)\boldsymbol{P}_n + \beta\boldsymbol{P}_{n+1}$. This algorithm permits the use of different time steps in different subdomains of the mesh. Time stepping by ΔT requires one total update and $m-1$ subcycles. The total update for the complete system is done first by extrapolating the interface displacements of subdomain B from time step $t_n + \Delta t$ to $t_n + \Delta T$. This time steps the subdomain A to $t_n + \Delta T$ while subdomain B is time stepped only to $t_n + \Delta t$. Subdomain B is then updated in $m-1$ small time step subcycles which involve computations only over the subdomain B.

Stability analysis [66] using the conventional approach of amplification matrices showed that the stable time step depends upon the relative stiffnesses of the subdomains. Even though stable regions exist, there are bands of instabilities which become smaller for bigger meshes. The subcycling procedure was also extended for the generalized-α method [67].

Concurrent Procedures: Ortiz and Nour-Omid [68] presented a *concurrent* procedure (computable in parallel) for dynamic structural analysis. Here the partitioned subdomains are solved independently one time step at a time. The resulting global solution is obtained by weighted averaging of the computed solutions where the corresponding mass matrices are used for weighting. They also propose a subclass of the algorithm where the effective stiffness matrices of the resulting systems are modified to give rise to an unconditionally stable method. An accuracy analysis comparing wave speeds (frequency distortion) is done in [69] and the computational efficiency of this method on parallel computers is demonstrated in [70].

Sotelino [71] described an I-E scheme similar to I-E mesh partitions by Belytschko except that all the subdomains are treated implicitly and only the interface nodes are treated explicitly. This allows the interface to be updated first explicitly and this interface solution is used as boundary condition for the implicit solve of the interior nodes.

Modak and Sotelino [72] described an iterative *group-implicit* procedure where the solutions to the individual subdomains are computed using an assumed interface reaction force from the adjoining subdomains and iterations are carried out till the reaction force converges.

Stability and accuracy of this algorithm is demonstrated numerically. Dere and Sotelino [73] proposed additional modifications to the iterative group implicit algorithm to improve its convergence and stability properties.

Asynchronous Variational Integrators: Lew, Marsden, Ortiz and West [74, 28] developed Asynchronous Variational Integrators (AVIs) which allow the integration of each element with its own time step so that the time step for the entire mesh is not governed by the time step of the smallest element. This method does not involve domain decomposition but it addresses the issue of multi-time-stepping. The maximum allowable time step in the mesh is still constrained by the Courant limit for the largest element in the mesh. However, AVIs are not be suitable for implicit computations such as evaluating the overall dynamic response of a large structure since implicit AVIs are coupled in the entire space-time domain that leads to huge computational costs.

3.3 Dual Coupling Methods

Coupling methods based on dual Schur complement approach have recently received significant attention from researchers in domain decomposition. Dual coupling methods enforce continuity of the solution across the interface between subdomains by introducing constraints that are usually imposed using Lagrange multipliers. This makes them suitable for coupling two or more subdomains almost completely independent of each other. Applications of the dual Schur complement method such as fluid structure coupling using partitioned procedures by Park and co-workers [75, 76], non-matching mesh tying using mortar methods [77, 78] and discontinuous interfaces [79] have also been explored. The discussion here will be limited to dual Schur complement methods for domain decomposition in particular for structural dynamics.

3.3.1 Finite Element Tearing and Interconnecting - FETI

One of the most popular and well-researched dual Schur methods is the finite element tearing and interconnecting (FETI) method proposed by Farhat and Roux [80, 81]. A brief formulation of the FETI method is presented here for problems in linear statics which can be extended further to non-linear structural dynamics.

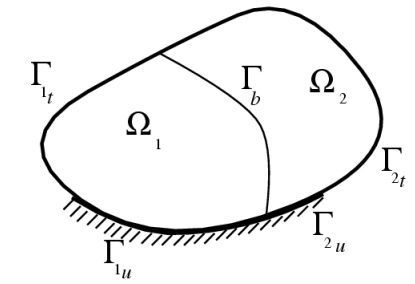

Figure 3.3: Domain decomposition for finite element tearing and interconnecting.

A continuous problem domain Ω is partitioned into subdomains Ω_1 and Ω_2 creating an interface boundary between the two subdomains Γ_b as shown in figure 3.3. Γ_u and Γ_t denote the Dirichlet and Neumann boundaries respectively. The variational form of the original boundary value problem is written as follows. For given $\boldsymbol{f}$ and $\boldsymbol{t}$, find the displacement $\boldsymbol{u}$ which is a stationary point of the energy functional:

$$\mathscr{E}(\boldsymbol{v}) = \tfrac{1}{2}a(\boldsymbol{v},\boldsymbol{v}) - (\boldsymbol{v},\boldsymbol{f}) - (\boldsymbol{v},\boldsymbol{t})_\Gamma \tag{3.14}$$

where

$$\begin{aligned} a(\boldsymbol{v},\boldsymbol{w}) &= \int_\Omega v_{i,j}\, C_{ijkl}\, w_{k,l}\, d\Omega \\ (\boldsymbol{v},\boldsymbol{f}) &= \int_\Omega v_i f_i\, d\Omega \\ (\boldsymbol{v},\boldsymbol{t})_\Gamma &= \int_{\Gamma_t} v_i t_i\, d\Gamma \end{aligned} \tag{3.15}$$

The energy functionals for the partitioned systems $\mathscr{E}_1(\boldsymbol{v}_1)$ and $\mathscr{E}_2(\boldsymbol{v}_2)$ are analogous to the

definition of $\mathscr{E}(\boldsymbol{v})$. In addition the solution should satisfy continuity across the interface:

$$\boldsymbol{u}_1 = \boldsymbol{u}_2 \qquad \text{on } \Gamma_b \tag{3.16}$$

Solution of the partitioned variational problem with the constraint (3.16) is equivalent to finding the stationary point of the Lagrangian:

$$\tilde{\mathscr{E}}(\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{\mu}) = \mathscr{E}_1(\boldsymbol{v}_1) + \mathscr{E}_2(\boldsymbol{v}_2) + (\boldsymbol{v}_1 - \boldsymbol{v}_2, \boldsymbol{\mu})_{\Gamma_b} \tag{3.17}$$

where

$$(\boldsymbol{v}_1 - \boldsymbol{v}_2, \boldsymbol{\mu})_{\Gamma_b} = \int_{\Gamma_b} \boldsymbol{\mu} \cdot (\boldsymbol{v}_1 - \boldsymbol{v}_2)\, d\Gamma \tag{3.18}$$

This reduces to finding the displacement fields $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ and Lagrange multipliers $\boldsymbol{\lambda}$ such that:

$$\tilde{\mathscr{E}}(\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{\mu}) \leq \tilde{\mathscr{E}}(\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{\lambda}) \leq \tilde{\mathscr{E}}(\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{\lambda}) \tag{3.19}$$

hold for any admissible $\boldsymbol{v}_1$, $\boldsymbol{v}_2$ and $\boldsymbol{\mu}$.

Applying the principle of virtual work and the finite element discretization one obtains:

$$\begin{aligned} \boldsymbol{K}^1\boldsymbol{U}^1 &= \boldsymbol{P}^1 + \boldsymbol{C}^{1^T}\boldsymbol{\Lambda} \\ \boldsymbol{K}^2\boldsymbol{U}^2 &= \boldsymbol{P}^2 + \boldsymbol{C}^{2^T}\boldsymbol{\Lambda} \\ \boldsymbol{C}^1\boldsymbol{U}^1 + \boldsymbol{C}^2\boldsymbol{U}^2 &= \boldsymbol{0} \end{aligned} \tag{3.20}$$

where $\boldsymbol{C}^1$ and $\boldsymbol{C}^2$ are boolean connectivity matrices for subdomains Ω_1 and Ω_2 respectively which pick out components corresponding to the degrees of freedom lying along the interface Γ_b from the subdomain vectors. This is a system of three equations and three unknowns

which can be solved using a bordered procedure leading to:

$$\begin{aligned}\left[\boldsymbol{C}^1\boldsymbol{K}^{1^{-1}}\boldsymbol{C}^{1^T}+\boldsymbol{C}^2\boldsymbol{K}^{2^{-1}}\boldsymbol{C}^{2^T}\right]\boldsymbol{\Lambda} &= -(\boldsymbol{C}^1\boldsymbol{K}^{1^{-1}}\boldsymbol{P}^1+\boldsymbol{C}^2\boldsymbol{K}^{2^{-1}}\boldsymbol{P}^2)\\ \boldsymbol{U}^1 &= \boldsymbol{K}^{1^{-1}}(\boldsymbol{P}^1+\boldsymbol{C}^{1^T}\boldsymbol{\Lambda})\\ \boldsymbol{U}^2 &= \boldsymbol{K}^{2^{-1}}(\boldsymbol{P}^2+\boldsymbol{C}^{2^T}\boldsymbol{\Lambda})\end{aligned} \tag{3.21}$$

where $\boldsymbol{H}=\left[\boldsymbol{C}^1\boldsymbol{K}^{1^{-1}}\boldsymbol{C}^{1^T}+\boldsymbol{C}^2\boldsymbol{K}^{2^{-1}}\boldsymbol{C}^{2^T}\right]$ is called the Schur complement matrix.

For static problems singularities arise if the decomposition creates floating subdomains. These singularities have to be treated using the psuedo-inverses for individual subdomains which remove the rigid body modes from the subdomain solution:

$$\begin{bmatrix}\boldsymbol{H} & -\boldsymbol{G}_I\\ -\boldsymbol{G}_I^T & 0\end{bmatrix}\begin{bmatrix}\boldsymbol{\Lambda}\\ \boldsymbol{\alpha}\end{bmatrix}=\begin{bmatrix}\boldsymbol{d}\\ -\boldsymbol{e}\end{bmatrix} \tag{3.22}$$

where, for a multi-subdomain case with S subdomains:

$$\boldsymbol{H}=\sum_{k=1}^{S}\boldsymbol{C}^k\left[\boldsymbol{K}^k\right]^+\boldsymbol{C}^{k^T}\quad;\qquad \boldsymbol{d}=\sum_{k=1}^{S}\boldsymbol{C}^k\left[\boldsymbol{K}^k\right]^+\boldsymbol{P}^k \tag{3.23}$$

The second row of (3.22) enforces the self-equilibriating condition for floating subdomains and $\boldsymbol{G}_I$ stores their rigid body modes.

The Lagrange multiplier $\boldsymbol{\Lambda}$ is usually solved using an iterative preconditioned conjugate gradient (PCG) approach as:

1. Initialize $i=0$

$$\begin{aligned}\boldsymbol{\Lambda}^0 &= \boldsymbol{0}\\ \boldsymbol{r}^0 &= \boldsymbol{f}-\boldsymbol{H}\boldsymbol{\Lambda}^0\end{aligned} \tag{3.24}$$

2. Iterate for $i = 0, 1, 2 \cdots$ until convergence

$$\begin{aligned} \boldsymbol{d}^i &= \tilde{\boldsymbol{H}}^{-1}\boldsymbol{r}^i \\ \boldsymbol{p}^i &= \boldsymbol{d}^i - \sum_{j=i}^{i-1} \frac{\boldsymbol{d}^{i^T}\boldsymbol{H}\boldsymbol{p}^j}{\boldsymbol{p}^{j^T}\boldsymbol{H}\boldsymbol{p}^j}\boldsymbol{p}^j \\ \eta^i &= \frac{\boldsymbol{p}^{i^T}\boldsymbol{r}^i}{\boldsymbol{p}^{i^T}\boldsymbol{H}\boldsymbol{p}^i} \\ \boldsymbol{\Lambda}^{i+1} &= \boldsymbol{\Lambda}^i + \eta^i\boldsymbol{p}^i \\ \boldsymbol{r}^{i+1} &= \boldsymbol{r}^i - \eta^i\boldsymbol{H}\boldsymbol{p}^i \end{aligned} \tag{3.25}$$

where $\tilde{\boldsymbol{H}}^{-1}$ is a suitable preconditioner. A computationally economical lumped preconditioner [82] is defined as:

$$\tilde{\boldsymbol{H}}^{-1} \equiv \sum_{k=1}^{S} \boldsymbol{C}^k\boldsymbol{K}^k\boldsymbol{C}^{k^T} \tag{3.26}$$

The PCG procedure helps avoid explicit inversion of the Schur complement matrix in the interface equation (3.21). The matrix-vector products $\boldsymbol{H}\boldsymbol{p}^i$ and $\tilde{\boldsymbol{H}}^{-1}\boldsymbol{r}^i$ can be computed subdomain-wise one at a time and the matrices themselves need not be assembled. For the case with floating subdomains, one needs to project out the rigid body modes of the floating subdomains from the residuals $\boldsymbol{r}^i$ and the search directions $\boldsymbol{p}^i$ at every PCG iteration.

Further enhancements in terms of implementation of the FETI method were proposed in [83]. Park et al. [84] proposed an alternative *algebraic* form of the FETI method (A-FETI) which was later shown to be equivalent to the original FETI method for a specific choice of the projection operator by Rixen et al. [85]. The two approaches differ in their treatment of *cross-points* which are nodes in the mesh that are common to three or more subdomains. The original FETI method was shown to have better convergence if the continuity constraints at such cross points included redundant dofs which leads to a singular but solvable system of equations on the interface. The A-FETI method on the other hand can handle cross points between any number of subdomains without such redundancies and does not lead to

a singular system of equations on the interface.

A two-level FETI procedure for fourth order bending problems was described by Farhat and Mandel [86] and additional implementation refinements were provided in [87]. A FETI-DP approach which uses a combination of dual-primal Schur complements was proposed in [88] to improve the convergence of the iterative interface problem.

3.3.2 FETI Method for Structural Dynamics

The FETI method was also extended for structural dynamics by Farhat et al. [89]. The semi-discrete coupled equations of motion for dynamics, assuming $\Omega_1 = A$ and $\Omega_2 = B$ are:

$$\boldsymbol{M}^A\ddot{\boldsymbol{U}}^A + \boldsymbol{K}^A\boldsymbol{U}^A + \boldsymbol{C}^{A^T}\boldsymbol{\Lambda} - \boldsymbol{P}^A = \boldsymbol{0} \tag{3.27}$$

$$\boldsymbol{M}^B\ddot{\boldsymbol{U}}^B + \boldsymbol{K}^B\boldsymbol{U}^B + \boldsymbol{C}^{B^T}\boldsymbol{\Lambda} - \boldsymbol{P}^B = \boldsymbol{0} \tag{3.28}$$

$$\boldsymbol{C}^A\boldsymbol{X}^A + \boldsymbol{C}^B\boldsymbol{X}^B = \boldsymbol{0} \tag{3.29}$$

where $\boldsymbol{X}$ denotes the kinematic quantity whose continuity is enforced on the interface. Using this formulation one *cannot* enforce the continuity of all the kinematic variables namely, displacements, velocities and accelerations.

A spectral stability analysis [90] within the framework of the Generalized-α method proved that the the dynamic FETI algorithm is only *weakly* stable and predicted a linear growth instability of the decomposed problem for any constraint on the interface. Farhat and co-workers also proposed a *Time Parallel* iterative method for dynamics [91, 92].

3.3.3 Multi-time-stepping for the FETI Method

Gravouil and Combescure [93, 94] extended the dynamic FETI algorithm to include multiple time-steps based on equality of velocities across the interface and proved the stability of their method using the energy approach. This method is referred to as the GC method in

this document. Subdomains A and B are integrated with time steps ΔT and Δt respectively where $\Delta T = \tau \Delta t$. One can think of subdomain A as being integrated implicitly and B integrated explicitly since the implicit time step is usually greater than the explicit time step. However, theoretically there is no restriction on the time step ratios and the explicit time step may be a multiple of the implicit time step as long as it satisfies the Courant condition (Eq. 2.129) in its subdomain.

Equations (3.27) - (3.29) may be discretized in time as:

$$\boldsymbol{M}^{\mathrm{A}}\ddot{\boldsymbol{U}}^{\mathrm{A}}_{n+\tau} + K^{\mathrm{A}}\boldsymbol{U}^{\mathrm{A}}_{n+\tau} = \boldsymbol{P}^{\mathrm{A}}_{n+\tau} + \boldsymbol{C}^{\mathrm{A}^{\mathrm{T}}}\boldsymbol{\Lambda}_{n+\tau} \tag{3.30}$$

$$\boldsymbol{M}^{\mathrm{B}}\ddot{\boldsymbol{U}}^{\mathrm{B}}_{n+j} + K^{\mathrm{B}}\boldsymbol{U}^{\mathrm{B}}_{n+j} = \boldsymbol{P}^{\mathrm{B}}_{n+j} + \boldsymbol{C}^{\mathrm{B}^{\mathrm{T}}}\boldsymbol{\Lambda}_{n+j} \qquad \forall j : 1 \leq j \leq \tau \tag{3.31}$$

$$\boldsymbol{C}^{\mathrm{A}}\dot{\boldsymbol{U}}^{\mathrm{A}}_{n+j} + \boldsymbol{C}^{\mathrm{B}}\dot{\boldsymbol{U}}^{\mathrm{B}}_{n+j} = \boldsymbol{0} \qquad \forall j : 1 \leq j \leq \tau \tag{3.32}$$

for subdomains A and B, where the subscripts $n+j$ and $n+\tau$ represent the instants t_{n+j} and $t_{n+\tau}$ respectively. Using the Newmark relations (2.100) and (2.101) with parameters $(\beta_{\mathrm{A}}, \gamma_{\mathrm{A}})$ and $(\beta_{\mathrm{B}}, \gamma_{\mathrm{B}})$ for subdomains A and B respectively, equations (3.30-3.32) yield:

$$\tilde{\boldsymbol{M}}^{\mathrm{A}}\ddot{\boldsymbol{U}}^{\mathrm{A}}_{n+\tau} = \boldsymbol{P}^{\mathrm{A}}_{n+\tau} - K^{\mathrm{A}}\hat{\boldsymbol{U}}^{\mathrm{A}}_{n+\tau} + \boldsymbol{C}^{\mathrm{A}^{\mathrm{T}}}\boldsymbol{\Lambda}_{n+\tau} \tag{3.33}$$

$$\tilde{\boldsymbol{M}}^{\mathrm{B}}\ddot{\boldsymbol{U}}^{\mathrm{B}}_{n+j} = \boldsymbol{P}^{\mathrm{B}}_{n+j} - K^{\mathrm{B}}\hat{\boldsymbol{U}}^{\mathrm{B}}_{n+j} + \boldsymbol{C}^{\mathrm{B}^{\mathrm{T}}}\boldsymbol{\Lambda}_{n+j} \qquad \forall j : 1 \leq j \leq \tau \tag{3.34}$$

$$\boldsymbol{C}^{\mathrm{A}}\dot{\boldsymbol{U}}^{\mathrm{A}}_{n+j} + \boldsymbol{C}^{\mathrm{B}}\dot{\boldsymbol{U}}^{\mathrm{B}}_{n+j} = \boldsymbol{0} \qquad \forall j : 1 \leq j \leq \tau \tag{3.35}$$

where for subdomain A:

$$\hat{\boldsymbol{U}}^{\mathrm{A}}_{n+\tau} = \boldsymbol{U}^{\mathrm{A}}_{n} + \Delta T\dot{\boldsymbol{U}}^{\mathrm{A}}_{n} + \Delta T^2\left(\tfrac{1}{2} - \beta_{\mathrm{A}}\right)\ddot{\boldsymbol{U}}^{\mathrm{A}}_{n} \tag{3.36}$$

$$\tilde{\boldsymbol{M}}^{\mathrm{A}} = \boldsymbol{M}^{\mathrm{A}} + \beta_{\mathrm{A}}\Delta T^2 K^{\mathrm{A}} \tag{3.37}$$

and for subdomain B:

$$\hat{\boldsymbol{U}}^{\mathrm{B}}_{n+j} = \boldsymbol{U}^{\mathrm{B}}_{n+j-1} + \Delta t \dot{\boldsymbol{U}}^{\mathrm{B}}_{n+j-1} + \Delta t^2 \left(\tfrac{1}{2} - \beta_{\mathrm{B}}\right) \ddot{\boldsymbol{U}}^{\mathrm{B}}_{n+j-1} \tag{3.38}$$

$$\tilde{\boldsymbol{M}}^{\mathrm{B}} = \boldsymbol{M}^{\mathrm{B}} + \beta_{\mathrm{B}} \Delta t^2 K^{\mathrm{B}} \tag{3.39}$$

One may note that the system of equations (3.33-3.35) are coupled by the unknowns $\dot{\boldsymbol{U}}^{\mathrm{A}}_{n+j}$, $\dot{\boldsymbol{U}}^{\mathrm{B}}_{n+j}$ and $\boldsymbol{\Lambda}_{n+j}$ $\forall j \in [1, \tau]$.

Equations (3.33-3.35) can be decoupled by splitting the kinematic quantities into two parts as follows:

$$\boldsymbol{U}_{n+j} = \boldsymbol{D}_{n+j} + \boldsymbol{d}_{n+j} \tag{3.40}$$

$$\dot{\boldsymbol{U}}_{n+j} = \dot{\boldsymbol{D}}_{n+j} + \dot{\boldsymbol{d}}_{n+j} \tag{3.41}$$

$$\ddot{\boldsymbol{U}}_{n+j} = \ddot{\boldsymbol{D}}_{n+j} + \ddot{\boldsymbol{d}}_{n+j} \tag{3.42}$$

where $\boldsymbol{D}_{n+j}$ is the displacement under external loads only and $\boldsymbol{d}_{n+j}$ is the displacement under interface tractions only. The Newmark scheme in equations (2.100,2.101) can then be expressed as:

$$\boldsymbol{D}_{n+1} = \hat{\boldsymbol{U}}_{n+1} + \beta \Delta t^2 \ddot{\boldsymbol{D}}_{n+1} \tag{3.43}$$

$$\boldsymbol{d}_{n+1} = \beta \Delta t^2 \ddot{\boldsymbol{d}}_{n+1} \tag{3.44}$$

$$\dot{\boldsymbol{D}}_{n+1} = \hat{\dot{\boldsymbol{U}}}_{n+1} + \gamma \Delta t \ddot{\boldsymbol{D}}_{n+1} \tag{3.45}$$

$$\dot{\boldsymbol{d}}_{n+1} = \gamma \Delta t \ddot{\boldsymbol{d}}_{n+1} \tag{3.46}$$

where $\hat{\boldsymbol{U}}_{n+1}$ and $\hat{\dot{\boldsymbol{U}}}_{n+1}$ are defined in equations (2.101) and (2.100). Substituting equations

(3.41,3.46) into equations (3.33-3.35) and splitting equations (3.33) and (3.34) we get:

$$\tilde{\boldsymbol{M}}^{\mathrm{A}}\ddot{\boldsymbol{D}}^{\mathrm{A}}_{n+\tau} = \boldsymbol{P}^{\mathrm{A}}_{n+\tau} - K^{\mathrm{A}}\hat{\boldsymbol{U}}^{\mathrm{A}}_{n+\tau} \tag{3.47}$$

$$\tilde{\boldsymbol{M}}^{\mathrm{B}}\ddot{\boldsymbol{D}}^{\mathrm{B}}_{n+j} = \boldsymbol{P}^{\mathrm{B}}_{n+j} - K^{\mathrm{B}}\hat{\boldsymbol{U}}^{\mathrm{B}}_{n+j} \qquad \forall j : 1 \leq j \leq \tau \tag{3.48}$$

$$\tilde{\boldsymbol{M}}^{\mathrm{A}}\ddot{\boldsymbol{d}}^{\mathrm{A}}_{n+\tau} = \boldsymbol{C}^{\mathrm{A}^{\mathrm{T}}}\boldsymbol{\Lambda}_{n+\tau} \tag{3.49}$$

$$\tilde{\boldsymbol{M}}^{\mathrm{B}}\ddot{\boldsymbol{d}}^{\mathrm{B}}_{n+j} = \boldsymbol{C}^{\mathrm{B}^{\mathrm{T}}}\boldsymbol{\Lambda}_{n+j} \qquad \forall j : 1 \leq j \leq \tau \tag{3.50}$$

$$\boldsymbol{C}^{\mathrm{A}}\left(\dot{\boldsymbol{D}}^{\mathrm{A}}_{n+j} + \dot{\boldsymbol{d}}^{\mathrm{A}}_{n+j}\right) + \boldsymbol{C}^{\mathrm{B}}\left(\dot{\boldsymbol{D}}^{\mathrm{B}}_{n+j} + \dot{\boldsymbol{d}}^{\mathrm{B}}_{n+j}\right) = \boldsymbol{0} \qquad \forall j : 1 \leq j \leq \tau \tag{3.51}$$

Note that equations (3.47,3.48) are now decoupled from the system and may be solved for $\ddot{\boldsymbol{D}}^{\mathrm{A}}_{n+\tau}$ and $\ddot{\boldsymbol{D}}^{\mathrm{B}}_{n+j}$ (initially for $j = 1$) independently of the others.

To solve the coupled system (3.49-3.51), $\dot{\boldsymbol{D}}^{\mathrm{A}}_{n+j}$ and $\dot{\boldsymbol{d}}^{\mathrm{A}}_{n+j}$ must be computed at the instant t_{n+j} by linearly interpolating the velocities between t_n and $t_{n+\tau}$ as:

$$\dot{\boldsymbol{D}}^{\mathrm{A}}_{n+j} = \left(1 - \frac{j}{\tau}\right)\dot{\boldsymbol{D}}^{\mathrm{A}}_{n} + \frac{j}{\tau}\dot{\boldsymbol{D}}^{\mathrm{A}}_{n+\tau} \tag{3.52}$$

$$\dot{\boldsymbol{d}}^{\mathrm{A}}_{n+j} = \left(1 - \frac{j}{\tau}\right)\dot{\boldsymbol{d}}^{\mathrm{A}}_{n} + \frac{j}{\tau}\dot{\boldsymbol{d}}^{\mathrm{A}}_{n+\tau} \tag{3.53}$$

Consider (3.51) in the form:

$$-\boldsymbol{C}^{\mathrm{A}}\dot{\boldsymbol{d}}^{\mathrm{A}}_{n+j} - \boldsymbol{C}^{\mathrm{B}}\dot{\boldsymbol{d}}^{\mathrm{B}}_{n+j} = \boldsymbol{C}^{\mathrm{A}}\dot{\boldsymbol{D}}^{\mathrm{A}}_{n+j} + \boldsymbol{C}^{\mathrm{B}}\dot{\boldsymbol{D}}^{\mathrm{B}}_{n+j} \quad \forall j : 1 \leq j \leq \tau \tag{3.54}$$

Substituting equation (3.53) above we get:

$$-\boldsymbol{C}^{\mathrm{A}}\left[\left(1 - \frac{j}{\tau}\right)\dot{\boldsymbol{d}}^{\mathrm{A}}_{n} + \frac{j}{\tau}\dot{\boldsymbol{d}}^{\mathrm{A}}_{n+\tau}\right] - \boldsymbol{C}^{\mathrm{B}}\dot{\boldsymbol{d}}^{\mathrm{B}}_{n+j} = \boldsymbol{C}^{\mathrm{A}}\dot{\boldsymbol{D}}^{\mathrm{A}}_{n+j} + \boldsymbol{C}^{\mathrm{B}}\dot{\boldsymbol{D}}^{\mathrm{B}}_{n+j} \qquad \forall j : 1 \leq j \leq \tau \tag{3.55}$$

Utilizing equation (3.46), $\dot{\boldsymbol{d}}^{\mathrm{A}}_{n}$, $\dot{\boldsymbol{d}}^{\mathrm{A}}_{n+\tau}$ and $\dot{\boldsymbol{d}}^{\mathrm{B}}_{n+j}$ can be expressed in terms of $\ddot{\boldsymbol{d}}^{\mathrm{A}}_{n}$, $\ddot{\boldsymbol{d}}^{\mathrm{A}}_{n+\tau}$ and $\ddot{\boldsymbol{d}}^{\mathrm{B}}_{n+j}$

as:

$$\dot{\boldsymbol{d}}_n^{\mathrm{A}} = \gamma_{\mathrm{A}} \Delta T \ddot{\boldsymbol{d}}_n^{\mathrm{A}} \tag{3.56}$$

$$\dot{\boldsymbol{d}}_{n+\tau}^{\mathrm{A}} = \gamma_{\mathrm{A}} \Delta T \ddot{\boldsymbol{d}}_{n+\tau}^{\mathrm{A}} \tag{3.57}$$

$$\dot{\boldsymbol{d}}_{n+j}^{\mathrm{B}} = \gamma_{\mathrm{B}} \Delta t \ddot{\boldsymbol{d}}_{n+j}^{\mathrm{B}} \tag{3.58}$$

Substituting the above expressions in equation (3.55):

$$\begin{aligned} -\left(\gamma_{\mathrm{A}} \Delta T\right) \boldsymbol{C}^{\mathrm{A}} \left[\left(1 - \frac{j}{\tau}\right) \ddot{\boldsymbol{d}}_n^{\mathrm{A}} + \frac{j}{\tau} \ddot{\boldsymbol{d}}_{n+\tau}^{\mathrm{A}} \right] - \left(\gamma_{\mathrm{B}} \Delta t\right) \boldsymbol{C}^{\mathrm{B}} \ddot{\boldsymbol{d}}_{n+j}^{\mathrm{B}} \\ = \boldsymbol{C}^{\mathrm{A}} \dot{\boldsymbol{D}}_{n+j}^{\mathrm{A}} + \boldsymbol{C}^{\mathrm{B}} \dot{\boldsymbol{D}}_{n+j}^{\mathrm{B}} \qquad \forall j : 1 \leq j \leq \tau \end{aligned} \tag{3.59}$$

Using equations (3.49) and (3.50) one obtains:

$$\ddot{\boldsymbol{d}}_n^{\mathrm{A}} = \tilde{\boldsymbol{M}}^{\mathrm{A}^{-1}} \boldsymbol{C}^{\mathrm{A}^{\mathrm{T}}} \boldsymbol{\Lambda}_n \tag{3.60}$$

$$\ddot{\boldsymbol{d}}_{n+\tau}^{\mathrm{A}} = \tilde{\boldsymbol{M}}^{\mathrm{A}^{-1}} \boldsymbol{C}^{\mathrm{A}^{\mathrm{T}}} \boldsymbol{\Lambda}_{n+\tau} \tag{3.61}$$

$$\ddot{\boldsymbol{d}}_{n+j}^{\mathrm{B}} = \tilde{\boldsymbol{M}}^{\mathrm{B}^{-1}} \boldsymbol{C}^{\mathrm{B}^{\mathrm{T}}} \boldsymbol{\Lambda}_{n+j} \tag{3.62}$$

Substituting equations (3.60-3.62) into (3.59) we get:

$$\begin{aligned} -\left(\gamma_{\mathrm{A}} \Delta T\right) \boldsymbol{C}^{\mathrm{A}} \tilde{\boldsymbol{M}}^{\mathrm{A}^{-1}} \boldsymbol{C}^{\mathrm{A}^{\mathrm{T}}} \left[\left(1 - \frac{j}{\tau}\right) \boldsymbol{\Lambda}_n + \frac{j}{\tau} \boldsymbol{\Lambda}_{n+\tau} \right] \\ -\left(\gamma_{\mathrm{B}} \Delta t\right) \boldsymbol{C}^{\mathrm{B}} \tilde{\boldsymbol{M}}^{\mathrm{B}^{-1}} \boldsymbol{C}^{\mathrm{B}^{\mathrm{T}}} \boldsymbol{\Lambda}_{n+j} \qquad = \qquad \boldsymbol{C}^{\mathrm{A}} \dot{\boldsymbol{D}}_{n+j}^{\mathrm{A}} + \boldsymbol{C}^{\mathrm{B}} \dot{\boldsymbol{D}}_{n+j}^{\mathrm{B}} \end{aligned} \tag{3.63}$$

If one assumes that the interface tractions are also interpolated linearly as:

$$\boldsymbol{\Lambda}_{n+j} = \left(1 - \frac{j}{\tau}\right) \boldsymbol{\Lambda}_n + \frac{j}{\tau} \boldsymbol{\Lambda}_{n+\tau} \tag{3.64}$$

then equation (3.63) reduces to:

$$\boldsymbol{H}\boldsymbol{\Lambda}_{n+j} = -\left(\boldsymbol{C}^{\mathrm{A}}\dot{\boldsymbol{D}}^{\mathrm{A}}_{n+j} + \boldsymbol{C}^{\mathrm{B}}\dot{\boldsymbol{D}}^{\mathrm{B}}_{n+j}\right) \qquad \forall j : 1 \leq j \leq \tau \tag{3.65}$$

where

$$\boldsymbol{H} = \left[(\gamma_{\mathrm{A}}\Delta T)\,\boldsymbol{C}^{\mathrm{A}}\tilde{\boldsymbol{M}}^{\mathrm{A}^{-1}}\boldsymbol{C}^{\mathrm{A}^{\mathrm{T}}} + (\gamma_{\mathrm{B}}\Delta t)\,\boldsymbol{C}^{\mathrm{B}}\tilde{\boldsymbol{M}}^{\mathrm{B}^{-1}}\boldsymbol{C}^{\mathrm{B}^{\mathrm{T}}}\right] \tag{3.66}$$

Thus, the system of equations to be solved in (3.47-3.51) reduces to:

$$\tilde{\boldsymbol{M}}^{\mathrm{A}}\ddot{\boldsymbol{D}}^{\mathrm{A}}_{n+\tau} = \boldsymbol{P}^{\mathrm{A}}_{n+\tau} - K^{\mathrm{A}}\hat{\boldsymbol{U}}^{\mathrm{A}}_{n+\tau} \tag{3.67}$$

$$\tilde{\boldsymbol{M}}^{\mathrm{B}}\ddot{\boldsymbol{D}}^{\mathrm{B}}_{n+j} = \boldsymbol{P}^{\mathrm{B}}_{n+j} - K^{\mathrm{B}}\hat{\boldsymbol{U}}^{\mathrm{B}}_{n+j} \qquad \forall j : 1 \leq j \leq \tau \tag{3.68}$$

$$\tilde{\boldsymbol{M}}^{\mathrm{B}}\ddot{\boldsymbol{d}}^{\mathrm{B}}_{n+j} = \boldsymbol{C}^{\mathrm{B}^{\mathrm{T}}}\boldsymbol{\Lambda}_{n+j} \qquad \forall j : 1 \leq j \leq \tau \tag{3.69}$$

$$\tilde{\boldsymbol{M}}^{\mathrm{A}}\ddot{\boldsymbol{d}}^{\mathrm{A}}_{n+\tau} = \boldsymbol{C}^{\mathrm{A}^{\mathrm{T}}}\boldsymbol{\Lambda}_{n+\tau} \tag{3.70}$$

$$\boldsymbol{H}\boldsymbol{\Lambda}_{n+j} = -\left(\boldsymbol{C}^{\mathrm{A}}\dot{\boldsymbol{D}}^{\mathrm{A}}_{n+j} + \boldsymbol{C}^{\mathrm{B}}\dot{\boldsymbol{D}}^{\mathrm{B}}_{n+j}\right) \qquad \forall j : 1 \leq j \leq \tau \tag{3.71}$$

This system can be solved by first solving for $\ddot{\boldsymbol{D}}^{\mathrm{A}}_{n+\tau}$ from equation (3.67) and $\ddot{\boldsymbol{D}}^{\mathrm{B}}_{n+j}$ for a particular value of j from equation (3.68). For every time step in subdomain B, one can solve for $\boldsymbol{\Lambda}_{n+j}$ from equation (3.71) by interpolating for $\dot{\boldsymbol{D}}^{\mathrm{A}}_{n+j}$ between $\dot{\boldsymbol{D}}^{\mathrm{A}}_{n}$ and $\dot{\boldsymbol{D}}^{\mathrm{A}}_{n+\tau}$. This value of $\boldsymbol{\Lambda}_{n+j}$ can then be used to solve for $\ddot{\boldsymbol{d}}^{\mathrm{B}}_{n+j}$ from equation (3.69) and repeating this process by incrementing j every time until $j = \tau$ gives $\boldsymbol{\Lambda}_{n+\tau}$. Lastly $\ddot{\boldsymbol{d}}^{\mathrm{A}}_{n+\tau}$ can be obtained from equation (3.70) using $\boldsymbol{\Lambda}_{n+\tau}$.

This algorithm can be generalized for more than two subdomains and for non-linear problems. Stability analysis using the energy method shows [94] that the coupling algorithm is unconditionally stable for continuity of velocities and the critical time step for each explicit subdomain is governed by the Courant limit. It is also shown that 2^{nd} order accuracy is preserved only if the velocity of subdomain A is constant over the larger time step ΔT and

one encounters numerical damping if that is not the case.

It should be noted that the interface solve needs to be done at every fine time step Δt. The interface matrix is full and involves inverses of the subdomain system matrices. This is quite taxing on the efficiency of the algorithm. Possible directions for improvement may consider different constraint equations and/or treating the discretized equations differently for time stepping cycles that are multiples of the larger time step ΔT. Accuracy may also be improved by choosing to implement constraints other than kinematic such as conservation of momentum.

Prakash and Hjelmstad [95] presented a multi-time-step coupling method for Newmark schemes which is unconditionally stable coupling, energy preserving and computationally very efficient. Details of this multi-time-step method will be discussed in Chapter 4.

3.4 Preliminary Coupling Methods Investigated

A basic problem, one may consider, to develop a multi-time-step coupling method is a *split* single degree of freedom (SDOF) problem shown in figure 3.4. Here, a simple mass ($\mathbf{m}$) and spring ($\mathbf{k}$) system is split into a system of two masses ($\mathbf{m}_A$, $\mathbf{m}_B$ where $\mathbf{m}_A + \mathbf{m}_B = \mathbf{m}$) and two springs ($\mathbf{k}_A$, $\mathbf{k}_B$ where $\mathbf{k}_A + \mathbf{k}_B = \mathbf{k}$) held together by an interface reaction force ($\mathbf{\Lambda}$). The load is also split into $\mathbf{f}_A$ and $\mathbf{f}_B$ such that $\mathbf{f}_A + \mathbf{f}_B = \mathbf{f}$. The two masses (nodes) can be integrated separately using different time steps and/or schemes and the solutions can be coupled by computing the interface reaction. The crux of this problem lies in computing the interface reaction correctly. One may visualize the two masses as being held together with a link and the interface reaction as the force developed in the link during the motion.

As part of the current research, a coding framework to investigate alternative approaches for multi-time-step coupling of the split SDOF problem was developed. Finite element codes were also written for linear 1-D and 2-D problems using bar and 4-node quadrilateral elements respectively. Subroutines for time integration include explicit central difference,

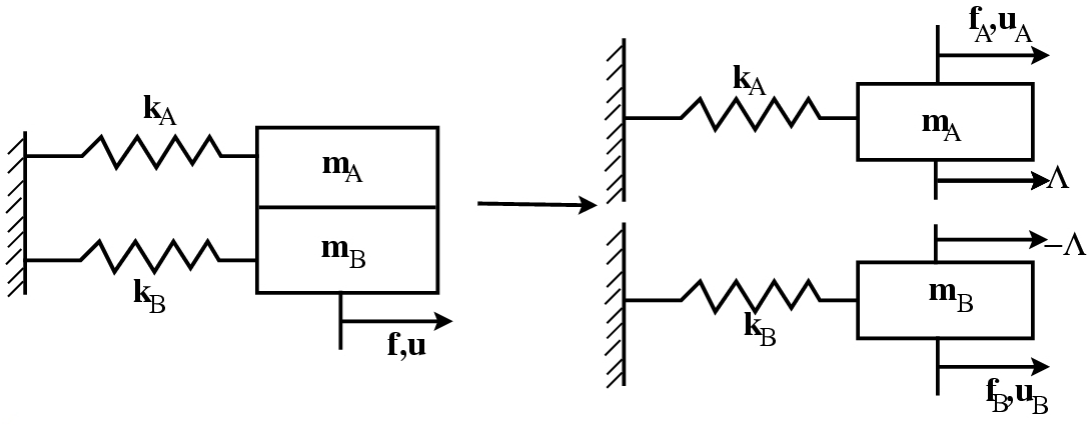

Figure 3.4: A split SDOF problem.

implicit average acceleration and a general Newmark method based on γ and β. The GC method was also implemented in both 1-D and 2-D codes for comparison with the current coupling method. Both, 1-D and 2-D codes are capable of analyzing a system with any number of subdomains and any integer time step ratios between them. Results obtained from a few approaches for multi-time-step coupling are discussed below.

3.4.1 Iterative I-E Coupling

This idea is based on element partitioning with implicit time step ΔT and explicit time step Δt where $\Delta T = \tau \Delta t$. The algorithm for advancing the solution by ΔT is as follows:

1. Save the current state
2. Loop until convergence
   (a) Solve the Implicit partition
   (b) Loop for j = 1 to τ
      i. Linearly interpolate displacement for shared nodes from implicit solution.
      ii. Solve Explicit partition with specified displacement on the shared nodes.
   (c) Compute the traction for the shared nodes from adjacent explicit elements.
   (d) Apply the computed traction to the shared nodes on implicit partition.
   (e) Check for convergence

- No: load the initial state stored before first iteration and loop again for the next iteration.
- Yes: save computed state and move to next time step.

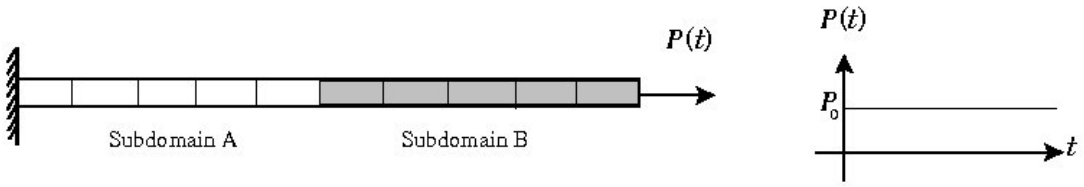

Figure 3.5: 1-D bar problem.

Tests were run on a 1-D bar, fixed at one end and loaded with a step load at the other. Half the elements of the bar were integrated explicitly and the other half integrated implicitly as shown in figure 3.5.

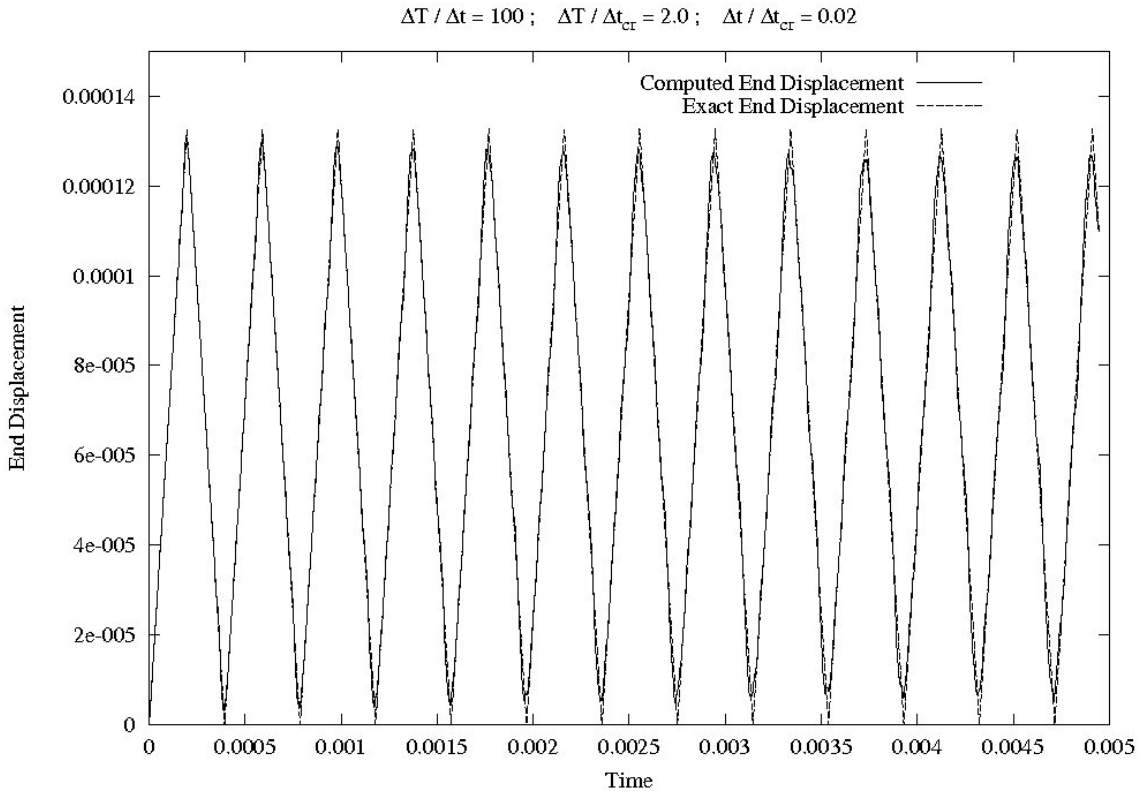

Figure 3.6: End displacement for $\Delta T/\Delta t = 100$. Stable

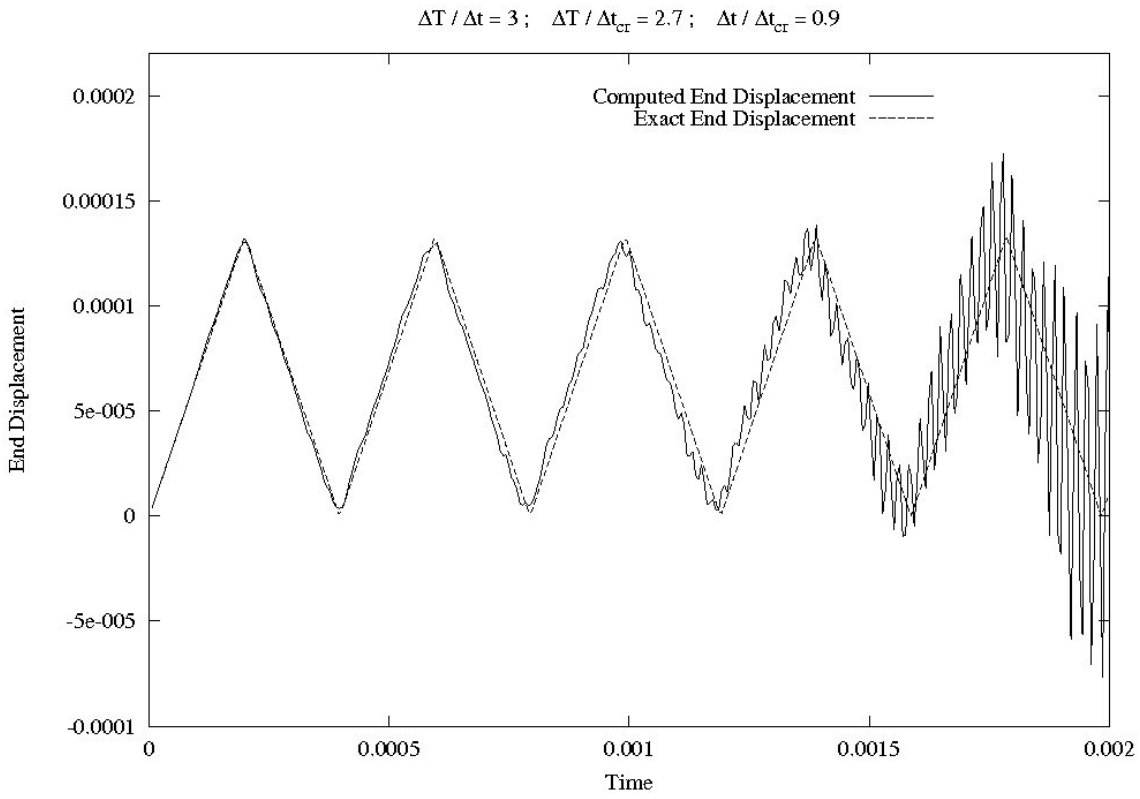

Figure 3.7: End displacement for $\Delta T/\Delta t = 3$. Unstable

Numerical results from this algorithm show that stability can be achieved for the case $\tau = 1$. However stability is questionable for higher time step ratios. The number of iterations to convergence also increases with the time step ratio. A case with $\tau = 100; \Delta t = 5 \times 10^{-9} \approx \Delta t_{cr}/50$ was computed for 10000 ΔT and shows good results without instability (figure 3.6). For another case with $\tau = 3; \Delta t = 2 \times 10^{-6} \approx \Delta t_{cr}$ stability is lost very quickly (figure 3.7). Analytical stability analysis may shed some light on the observed instability. Intuitively, one may argue that this method fails for multiple time step ratio cases because the interface reaction force at the intermediate time steps is not computed correctly which leads to instability.

3.4.2 GC Multi-time-step Coupling Using Bordered Solution

A modification of the GC method was devised to solve the $(2\tau + 1)$ equations in (3.30) - (3.32) as a whole in order to avoid the assumption of linear variation of interface reaction forces at the intermediate time steps. The entire system of equations was solved using a bordered solution procedure to advance the solution by ΔT. It was found that time stepping could still be used for computing the uncoupled part of the solution from equations (3.67) and (3.68). In order to use time-stepping for the interface solve and the subsequent update one has to consider the coupled interface equation obtained from applying the bordered procedure to the complete system:

$$\begin{bmatrix} \boldsymbol{A}_1 & & & \frac{1}{\tau}\boldsymbol{B} \\ \boldsymbol{A}_2 & \boldsymbol{A}_1 & & \frac{2}{\tau}\boldsymbol{B} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{A}_\tau & \boldsymbol{A}_{\tau-1} & \cdots & \boldsymbol{A}_1 + \frac{1}{\tau}\boldsymbol{B} \end{bmatrix} \begin{bmatrix} \boldsymbol{\lambda}_1 \\ \boldsymbol{\lambda}_2 \\ \vdots \\ \boldsymbol{\lambda}_\tau \end{bmatrix} = \begin{bmatrix} \sum_{AB} \boldsymbol{C}^k \dot{\boldsymbol{V}}_1^k \\ \sum_{AB} \boldsymbol{C}^k \dot{\boldsymbol{V}}_2^k \\ \vdots \\ \sum_{AB} \boldsymbol{C}^k \dot{\boldsymbol{V}}_\tau^k \end{bmatrix} \tag{3.72}$$

which can be solved using the transformation:

$$\begin{aligned} & \boldsymbol{\lambda}_j = \hat{\boldsymbol{\lambda}}_j - \boldsymbol{\Lambda}_j \lambda_\tau \\ \text{where} \quad & \boldsymbol{\Lambda}_j = \boldsymbol{A}_1^{-1} \left[\tfrac{j}{\tau}\boldsymbol{B} - \textstyle\sum_{i=1}^{j-1} \boldsymbol{A}_{j-i+1} \boldsymbol{\Lambda}_i \right] \\ \text{and} \quad & \hat{\boldsymbol{\lambda}}_j = \boldsymbol{A}_1^{-1} \left[\textstyle\sum_k \boldsymbol{C}^k \dot{\boldsymbol{V}}_j^k - \sum_{i=1}^{j-1} \boldsymbol{A}_{j-i+1} \hat{\boldsymbol{\lambda}}_i \right] \end{aligned} \tag{3.73}$$

This transformation permits the solution of Lagrange multipliers in a time stepping fashion along with the subdomain solutions. At the end of the τ^{th} time step the Lagrange multipliers are known exactly and the solution for both subdomains A and B can be updated correctly. Note that the solution for subdomain B has to be stored for all the intermediate time steps 1 to $(\tau - 1)$. Stability was studied numerically.

A modification of this scheme to eliminate the terms above the diagonal in the interface matrix, by using extrapolation instead of interpolation of velocities, was also considered.

This helps in faster interface solves and enables one to compute the entire solution in a time stepping fashion: one Δt time step at a time. Figure 3.8 shows a comparison of the results from the GC method (GC), the Bordered GC method using interpolation (BorInt) and the Bordered GC method using extrapolation (BorExt). It was observed that the assumption of linear variation of interface reactions at the intermediate time steps not only simplifies the interface solve but also gives more accurate results. In addition, the Bordered GC method using extrapolation results in spurious oscillations for higher time step ratios and sometimes displays instability.

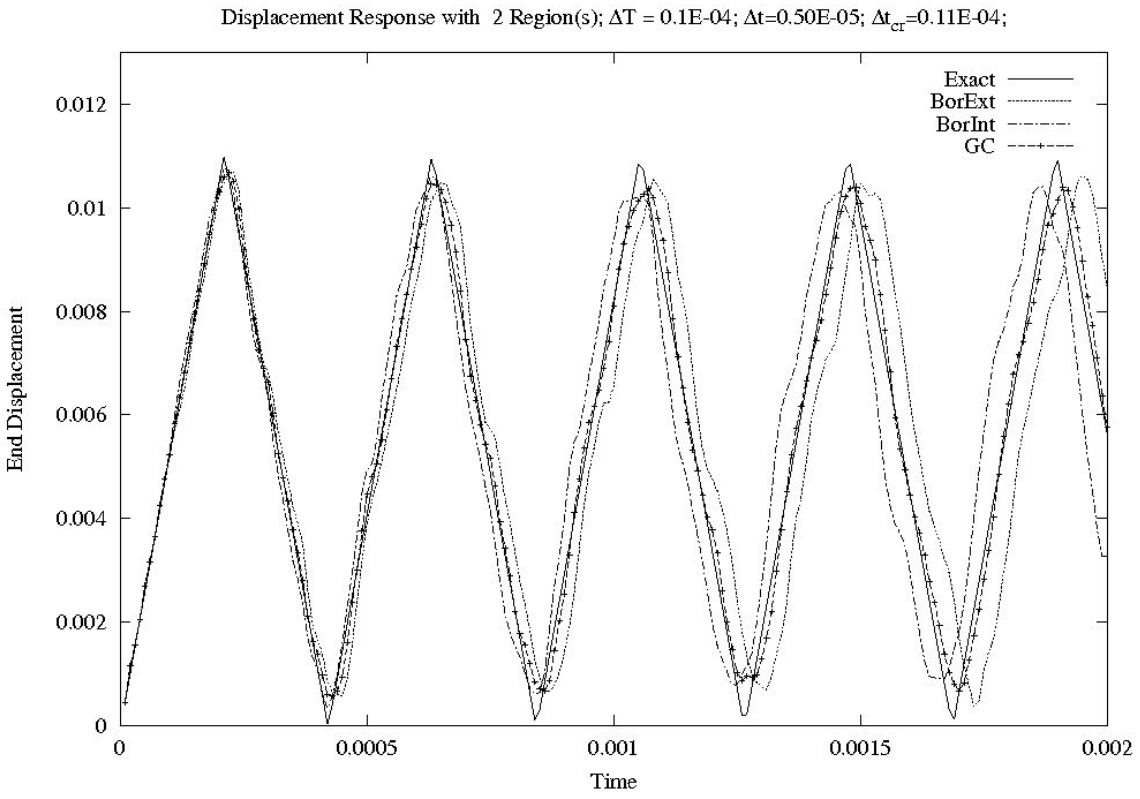

Figure 3.8: Algorithm comparison.

3.4.3 Minimization of Error Approach

This approach is based on minimizing a discretized least squares error functional that has a quadratic form in terms of the unknowns. Consider the SDOF problem shown in figure 3.4 where the time step ratio between A and B is 2 i.e. $\frac{1}{2}\Delta t_a = \Delta t_b = \Delta t$.

The system to be solved is:

$$\begin{aligned} \boldsymbol{g}_1^b &: \quad \boldsymbol{m}^b\ddot{\boldsymbol{u}}_1^b + \boldsymbol{k}^b\boldsymbol{u}_1^b + \boldsymbol{R}_1 - \boldsymbol{f}_1^b = \boldsymbol{0} \\ \boldsymbol{g}_2^b &: \quad \boldsymbol{m}^b\ddot{\boldsymbol{u}}_2^b + \boldsymbol{k}^b\boldsymbol{u}_2^b + \boldsymbol{R}_2 - \boldsymbol{f}_2^b = \boldsymbol{0} \\ \boldsymbol{g}_2^a &: \quad \boldsymbol{m}^a\ddot{\boldsymbol{u}}_2^a + \boldsymbol{k}^a\boldsymbol{u}_2^a - \boldsymbol{R}_2 - \boldsymbol{f}_2^a = \boldsymbol{0} \end{aligned} \tag{3.74}$$

where the subscripts denote the time steps starting from 0. Thus $t_0 = 0$, $t_1 = \Delta t$ and $t_2 = 2\Delta t$. A cubic polynomial is chosen as the *model* displacement for the interface:

$$\begin{aligned} \boldsymbol{u}^I(t) &= d\,t^3 + a\,t^2 + b\,t + c \\ \dot{\boldsymbol{u}}^I(t) &= 3\,d\,t^2 + 2\,a\,t + b \\ \ddot{\boldsymbol{u}}^I(t) &= 6\,d\,t + 2\,a \end{aligned} \tag{3.75}$$

where d, a, b and c are unknown parameters. Thus, the values of the interface quantities at time steps t_0, t_1 and t_2 are:

$$\begin{array}{c|c|c} \boldsymbol{u}_0^I = c & \boldsymbol{u}_1^I = d\,t_1^3 + a\,t_1^2 + b\,t_1 + c & \boldsymbol{u}_2^I = d\,t_2^3 + a\,t_2^2 + b\,t_2 + c \\ \dot{\boldsymbol{u}}_0^I = b & \dot{\boldsymbol{u}}_1^I = 3\,d\,t_1^2 + 2\,a\,t_1 + b & \dot{\boldsymbol{u}}_2^I = 3\,d\,t_2^2 + 2\,a\,t_2 + b \\ \ddot{\boldsymbol{u}}_0^I = 2\,a & \ddot{\boldsymbol{u}}_1^I = 6\,d\,t_1 + 2\,a & \ddot{\boldsymbol{u}}_2^I = 6\,d\,t_2 + 2\,a \end{array} \tag{3.76}$$

Now, we define an error functional as:

$$\begin{aligned}\mathscr{J} =& \tfrac{1}{2}\xi_{\ddot{U}}[(\ddot{\boldsymbol{u}}_0^a - \ddot{\boldsymbol{u}}_0^I)^2 + (\ddot{\boldsymbol{u}}_2^a - \ddot{\boldsymbol{u}}_2^I)^2 + \textstyle\sum_{j=0}^{2}(\ddot{\boldsymbol{u}}_j^b - \ddot{\boldsymbol{u}}_j^I)^2] \\ &+ \tfrac{1}{2}\xi_{\dot{U}}[(\dot{\boldsymbol{u}}_0^a - \dot{\boldsymbol{u}}_0^I)^2 + (\dot{\boldsymbol{u}}_2^a - \dot{\boldsymbol{u}}_2^I)^2 + \textstyle\sum_{j=0}^{2}(\dot{\boldsymbol{u}}_j^b - \dot{\boldsymbol{u}}_j^I)^2] \\ &+ \tfrac{1}{2}\xi_{U}[(\boldsymbol{u}_0^a - \boldsymbol{u}_0^I)^2 + (\boldsymbol{u}_2^a - \boldsymbol{u}_2^I)^2 + \textstyle\sum_{j=0}^{2}(\boldsymbol{u}_j^b - \boldsymbol{u}_j^I)^2] \\ &+ \lambda_2^a \boldsymbol{g}_2^a + \lambda_1^b \boldsymbol{g}_1^b + \lambda_2^b \boldsymbol{g}_2^b\end{aligned} \tag{3.77}$$

This functional is minimized in terms of the 18 unknowns : d, a, b, c, $\ddot{\boldsymbol{u}}_1^b$, $\ddot{\boldsymbol{u}}_2^b$, $\ddot{\boldsymbol{u}}_2^a$, $\dot{\boldsymbol{u}}_1^b$, $\dot{\boldsymbol{u}}_2^b$, $\dot{\boldsymbol{u}}_2^a$, $\boldsymbol{u}_1^b$, $\boldsymbol{u}_2^b$, $\boldsymbol{u}_2^a$, $\boldsymbol{R}_1$, $\boldsymbol{R}_2$, λ_1^b, λ_2^b and λ_2^a yielding an 18×18 system which can then be solved as a whole. The above system results in a stable computation for multi-time-step coupling. However, the interface reaction forces are still not computed correctly (figure 3.9).

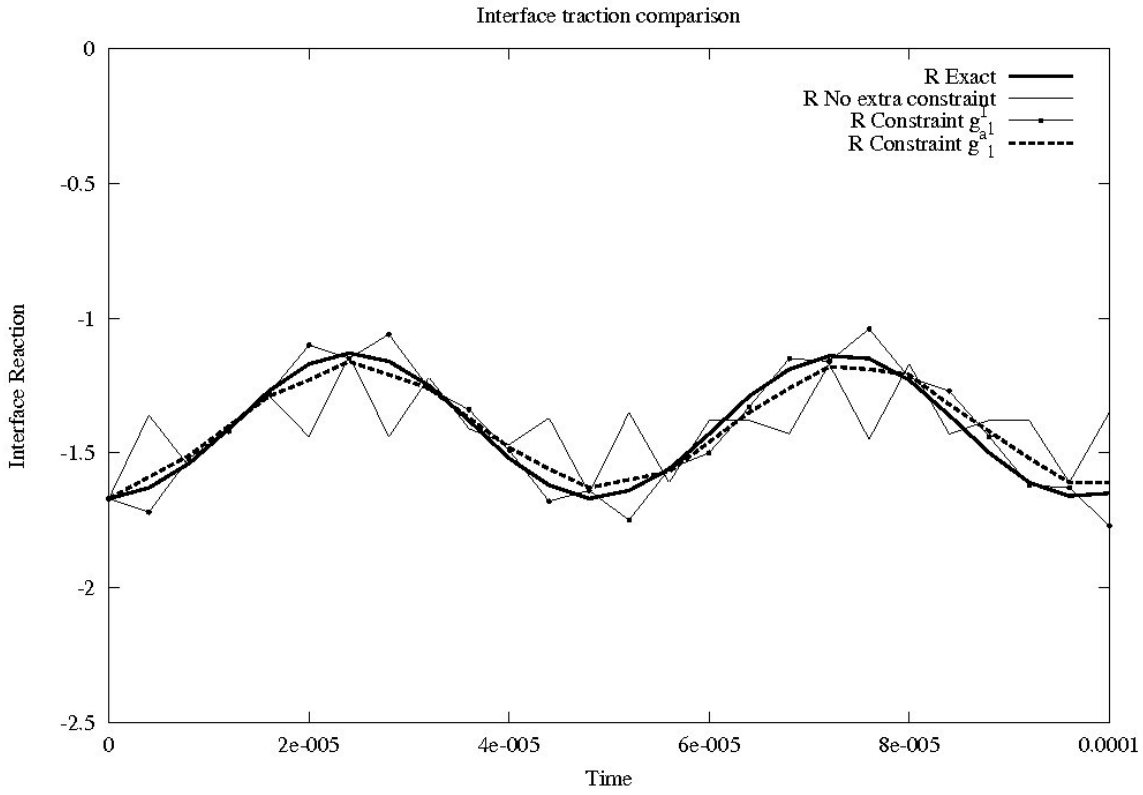

Figure 3.9: Comparison of computed interface reactions by using different constraints.

To overcome this anomaly, the following constraints were added to the above system one

at a time:

$$\begin{aligned} \boldsymbol{g}_1^I \quad &: \quad \boldsymbol{m}^a \ddot{\boldsymbol{u}}_1^I + \boldsymbol{k}^a \boldsymbol{u}_1^I - \boldsymbol{R}_1 - \boldsymbol{f}_1^a = \boldsymbol{0} \\ \boldsymbol{g}_1^a \quad &: \quad \boldsymbol{m}^a \ddot{\boldsymbol{u}}_1^a + \boldsymbol{k}^a \boldsymbol{u}_1^a - \boldsymbol{R}_1 - \boldsymbol{f}_1^a = \boldsymbol{0} \end{aligned} \tag{3.78}$$

where the quantities $\ddot{\boldsymbol{u}}_1^a$, $\dot{\boldsymbol{u}}_1^a$ and $\boldsymbol{u}_1^a$ in the second constraint are obtained by linearly interpolating between t_0 and t_2. As evident from the plot in Figure 3.9, adding the constraint $\boldsymbol{g}_1^a$ resulted in a very accurate computation of the interface traction. This observation led to the idea used in the following chapter for the development of a new, efficient and accurate multi-time-step coupling method for Newmark schemes.

Chapter 4

A New Multi-time-step Coupling Method

An efficient and accurate multi-time-step coupling method using FETI domain decomposition for structural dynamics is presented. Using this method one can divide a large structural mesh into a number of smaller subdomains, solve the individual subdomains separately and couple the solutions together to obtain the solution to the original problem. The various subdomains can be integrated in time using different time steps and/or different Newmark schemes. This approach will be most effective for very large-scale simulations on complex geometries.

The present coupling method builds upon the multi-time-step method previously proposed by Gravouil and Combescure [93] (GC method) who demonstrated that imposing continuity of velocities at the interface led to a stable algorithm. They proposed a multi-time-step coupling method to couple arbitrary Newmark schemes with different time steps in different subdomains. They proved that the GC method is unconditionally stable as long as all of the individual subdomains satisfy their own stability requirements. They also showed that for the case of a single time step in all the subdomains, the GC method is energy preserving in that it does not add or remove energy from the coupled system. However, for multi-time-step cases the GC method is dissipative. Another drawback of the GC method for multi-time-step cases is that one needs to compute the interface reactions at the smallest time step in the mesh which is a significant computational effort.

It is shown that for the simplest case, when the same time step is used for all subdomains in the mesh, the present method simplifies to the GC method and is unconditionally stable and energy preserving. In addition, the present method is shown to posses these properties

of unconditional stability and energy preservation for general multi-time-step cases too. The present method is also computationally more efficient than the GC method since the computation of interface forces is required only at the largest time step in the mesh as opposed to the smallest time step for the GC method.

4.1 Another Look at Newmark Time Stepping Scheme

To facilitate the derivation of the present multi-time-step coupling method it is helpful to visualize the Newmark time-stepping scheme in the following way. Consider the fully discretized system of equations and Newmark relations:

$$\boldsymbol{M}\ddot{\boldsymbol{U}}_{n+1} + \boldsymbol{K}\boldsymbol{U}_{n+1} = \boldsymbol{P}_{n+1} \tag{4.1}$$

$$\dot{\boldsymbol{U}}_{n+1} = \dot{\boldsymbol{U}}_n + \Delta t\,(1-\gamma)\,\ddot{\boldsymbol{U}}_n + \gamma\Delta t\ddot{\boldsymbol{U}}_{n+1} \tag{4.2}$$

$$\boldsymbol{U}_{n+1} = \boldsymbol{U}_n + \Delta t\dot{\boldsymbol{U}}_n + \Delta t^2\left(\tfrac{1}{2}-\beta\right)\ddot{\boldsymbol{U}}_n + \beta\Delta t^2\ddot{\boldsymbol{U}}_{n+1} \tag{4.3}$$

Writing the same in matrix form, one obtains:

$$\begin{bmatrix} \boldsymbol{M} & 0 & \boldsymbol{K} \\ -\gamma\Delta t\boldsymbol{I} & \boldsymbol{I} & 0 \\ -\beta\Delta t^2\boldsymbol{I} & 0 & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{U}}_{n+1} \\ \dot{\boldsymbol{U}}_{n+1} \\ \boldsymbol{U}_{n+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{P}_{n+1} \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} 0 \\ \hat{\dot{\boldsymbol{U}}}_{n+1} \\ \hat{\boldsymbol{U}}_{n+1} \end{bmatrix} \tag{4.4}$$

This can be expressed compactly as:

$$\mathbb{M}\,\mathbb{U}_{n+1} = \mathbb{P}_{n+1} - \mathbb{N}\,\mathbb{U}_n \tag{4.5}$$

where

$$\mathbb{M} = \begin{bmatrix} \boldsymbol{M} & 0 & \boldsymbol{K} \\ -\gamma\Delta t\boldsymbol{I} & \boldsymbol{I} & 0 \\ -\beta\Delta t^2\boldsymbol{I} & 0 & \boldsymbol{I} \end{bmatrix} \;;\quad \mathbb{N} = \begin{bmatrix} 0 & 0 & 0 \\ -\Delta t(1-\gamma)\boldsymbol{I} & -\boldsymbol{I} & 0 \\ -\Delta t^2(\tfrac{1}{2}-\beta)\boldsymbol{I} & -\Delta t\boldsymbol{I} & -\boldsymbol{I} \end{bmatrix} \tag{4.6}$$

$$\mathbb{U}_n = \begin{bmatrix} \ddot{\boldsymbol{U}}_n \\ \dot{\boldsymbol{U}}_n \\ \boldsymbol{U}_n \end{bmatrix} \quad ; \quad \mathbb{P}_{n+1} = \begin{bmatrix} \boldsymbol{P}_{n+1} \\ 0 \\ 0 \end{bmatrix} \tag{4.7}$$

One may verify that solving the system (4.5) is the same as advancing the solution by a single time step from t_n to t_{n+1}. Knowing the solution at time t_{n+1} one can now advance the solution by another time step as:

$$\left[\begin{array}{ccc|ccc} \boldsymbol{M} & 0 & \boldsymbol{K} & & & \\ -\gamma\Delta t\boldsymbol{I} & \boldsymbol{I} & 0 & & 0 & \\ -\beta\Delta t^2\boldsymbol{I} & 0 & \boldsymbol{I} & & & \\ \hline 0 & 0 & 0 & \boldsymbol{M} & 0 & \boldsymbol{K} \\ -\Delta t(1-\gamma)\boldsymbol{I} & -\boldsymbol{I} & 0 & -\gamma\Delta t\boldsymbol{I} & \boldsymbol{I} & 0 \\ -\Delta t^2(\frac{1}{2}-\beta)\boldsymbol{I} & -\Delta t\boldsymbol{I} & -\boldsymbol{I} & -\beta\Delta t^2\boldsymbol{I} & 0 & \boldsymbol{I} \end{array}\right] \left[\begin{array}{c} \ddot{\boldsymbol{U}}_{n+1} \\ \dot{\boldsymbol{U}}_{n+1} \\ \boldsymbol{U}_{n+1} \\ \hline \ddot{\boldsymbol{U}}_{n+2} \\ \dot{\boldsymbol{U}}_{n+2} \\ \boldsymbol{U}_{n+2} \end{array}\right] = \left[\begin{array}{c} \boldsymbol{P}_{n+1} \\ \hat{\dot{\boldsymbol{U}}}_{n+1} \\ \hat{\boldsymbol{U}}_{n+1} \\ \hline \boldsymbol{P}_{n+2} \\ 0 \\ 0 \end{array}\right] \tag{4.8}$$

This can be expressed compactly as:

$$\begin{bmatrix} \mathbb{M} & 0 \\ \mathbb{N} & \mathbb{M} \end{bmatrix} \begin{bmatrix} \mathbb{U}_{n+1} \\ \mathbb{U}_{n+2} \end{bmatrix} = \begin{bmatrix} \mathbb{P}_{n+1} \\ \mathbb{P}_{n+2} \end{bmatrix} \tag{4.9}$$

Extending the analogy to m time steps one obtains the following system of equations:

$$\begin{bmatrix} \mathbb{M} & & & \\ \mathbb{N} & \mathbb{M} & & \\ & \ddots & \ddots & \\ & & \mathbb{N} & \mathbb{M} \end{bmatrix} \begin{bmatrix} \mathbb{U}_{n+1} \\ \mathbb{U}_{n+2} \\ \vdots \\ \mathbb{U}_{n+m} \end{bmatrix} = \begin{bmatrix} \mathbb{P}_{n+1} - \mathbb{N}\,\mathbb{U}_n \\ \mathbb{P}_{n+2} \\ \vdots \\ \mathbb{P}_{n+m} \end{bmatrix} \tag{4.10}$$

where

$$\mathbb{P}_{n+j} = \begin{bmatrix} \boldsymbol{P}_{n+j} \\ 0 \\ 0 \end{bmatrix} \quad \mathbb{U}_{n+j} = \begin{bmatrix} \ddot{\boldsymbol{U}}_{n+j} \\ \dot{\boldsymbol{U}}_{n+j} \\ \boldsymbol{U}_{n+j} \end{bmatrix} \quad \forall j \in [1, m] \tag{4.11}$$

Note that the system (4.10) is lower triangular and can be solved by forward substitution row-wise from top to bottom which simply amounts to time-stepping the initial state $\mathbb{U}_n$ by m time steps to obtain $\mathbb{U}_{n+m}$. For non-linear problems, this way of visualizing the Newmark time-stepping scheme is valid for a single time step of the linearized system (2.105) within one iteration.

4.2 Coupled Equations for Structural Dynamics

The coupled equations of motion for dual Schur domain decomposition, based on the conventional FETI approach and the algebraic FETI (A-FETI) approach, can be derived from energy principles (see section 2.6). The two approaches have been shown to be equivalent [85] but have their own merits and demerits . The FETI-DP method [88] combines the two approaches by treating part of the interface that is shared exclusively between two subdomains in the conventional manner and other parts of the interface that include cross points using the A-FETI approach.

4.2.1 Conventional FETI

Consider a continuous problem domain Ω which is decomposed into S subdomains Ω_k for $1 \leq k \leq S$ as shown in Figure 4.1(a). This partitioning creates the interface Γ_b of shared nodes between the subdomains. Let the number of degrees of freedom in subdomain Ω_k be N_k and the total number of degrees of freedom along Γ_b be L. As shown in Figure 4.1(b), there are additional forces $\boldsymbol{R}(t)$ acting on a subdomain resulting from its interaction with the rest of the mesh.

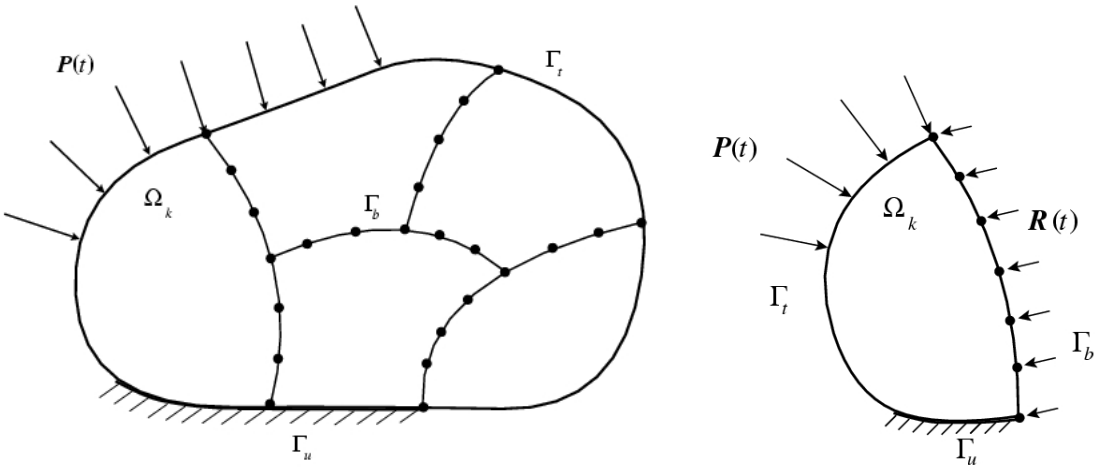

Figure 4.1: (a) A partitioned problem domain showing shared nodes. (b) A typical subdomain.

For multiple subdomains, the Lagrangian can be written as:

$$\mathcal{L} \equiv \sum_{k=1}^{S} \left[\frac{1}{2} \dot{\boldsymbol{U}}^{k\mathrm{T}} \boldsymbol{M}^k \dot{\boldsymbol{U}}^k - \frac{1}{2} \boldsymbol{U}^{k\mathrm{T}} \boldsymbol{K}^k \boldsymbol{U}^k \right] \tag{4.12}$$

where the superscript on a quantity indicates its subdomain. In addition the solution must be continuous across the interface Γ_b. We impose continuity of velocities at the interface as:

$$\sum_{k=1}^{S} \boldsymbol{C}^k \dot{\boldsymbol{U}}^k = \boldsymbol{0} \tag{4.13}$$

where $\boldsymbol{C}^k$ is a boolean matrix of dimension $L \times N_k$ that operates on nodal vectors from subdomain Ω_k, picks out the degrees of freedom lying along Γ_b and assembles them in an *interface* vector. Note that conjugate to $\boldsymbol{C}^k$, the $\boldsymbol{C}^{k^T}$ matrix operates on interface vectors, picks out the degrees of freedom that belong to Ω_k and assembles them at their corresponding position in the nodal vector for the subdomain.

The Lagrangian is augmented with the constraint equation (4.13) using a multiplier $\boldsymbol{\Lambda}$

as:

$$\tilde{\mathcal{L}} = \sum_{k=1}^{S} \left[\frac{1}{2} \dot{\boldsymbol{U}}^{k^{\mathrm{T}}} \boldsymbol{M}^k \dot{\boldsymbol{U}}^k - \frac{1}{2} \boldsymbol{U}^{k^{\mathrm{T}}} \boldsymbol{K}^k \boldsymbol{U}^k \right] + \boldsymbol{\Lambda}^{\mathrm{T}} \left[\sum_{k=1}^{S} \boldsymbol{C}^k \dot{\boldsymbol{U}}^k \right] \tag{4.14}$$

Taking the variation of $\tilde{\mathcal{L}}$:

$$\delta\tilde{\mathcal{L}} = \sum_{k=1}^{S} \left[\delta\dot{\boldsymbol{U}}^{k^{\mathrm{T}}} \boldsymbol{M}^k \dot{\boldsymbol{U}}^k - \delta\boldsymbol{U}^{k^{\mathrm{T}}} \boldsymbol{K}^k \boldsymbol{U}^k + \delta\dot{\boldsymbol{U}}^{k^{\mathrm{T}}} \boldsymbol{C}^{k^{\mathrm{T}}} \boldsymbol{\Lambda} \right] + \delta\boldsymbol{\Lambda}^{\mathrm{T}} \left[\sum_{k=1}^{S} \boldsymbol{C}^k \dot{\boldsymbol{U}}^k \right] \tag{4.15}$$

The external virtual work done on the system is given by:

$$\delta\mathcal{W} = \sum_{k=1}^{S} \delta\boldsymbol{U}^{k^{\mathrm{T}}} \boldsymbol{P}^k \tag{4.16}$$

Hamilton's principle states:

$$\int_{t_1}^{t_2} \left(\delta\tilde{\mathcal{L}} + \delta\mathcal{W} \right) \, dt = 0 \tag{4.17}$$

Upon substitution we obtain:

$$\int_{t_1}^{t_2} \sum_{k=1}^{S} \left[\delta\dot{\boldsymbol{U}}^{k^{\mathrm{T}}} \left(\boldsymbol{M}^k \dot{\boldsymbol{U}}^k + \boldsymbol{C}^{k^{\mathrm{T}}} \boldsymbol{\Lambda} \right) - \delta\boldsymbol{U}^{k^{\mathrm{T}}} \left(\boldsymbol{K}^k \boldsymbol{U}^k - \boldsymbol{P}^k \right) \right] \, dt + \int_{t_1}^{t_2} \delta\boldsymbol{\Lambda}^{\mathrm{T}} \left[\sum_{k=1}^{S} \boldsymbol{C}^k \dot{\boldsymbol{U}}^k \right] \, dt = 0 \tag{4.18}$$

Integrating the first term by parts assuming $\delta\boldsymbol{U}^k(t_1) = \delta\boldsymbol{U}^k(t_2) = 0$ we get:

$$\int_{t_1}^{t_2} \sum_{k=1}^{S} - \delta\boldsymbol{U}^{k^{\mathrm{T}}} \left[\boldsymbol{M}^k \ddot{\boldsymbol{U}}^k + \boldsymbol{K}^k \boldsymbol{U}^k + \boldsymbol{C}^{k^{\mathrm{T}}} \dot{\boldsymbol{\Lambda}} - \boldsymbol{P}^k \right] + \delta\boldsymbol{\Lambda}^{\mathrm{T}} \left[\sum_{k=1}^{S} \boldsymbol{C}^k \dot{\boldsymbol{U}}^k \right] \, dt = 0 \tag{4.19}$$

Using the fundamental theorem of the calculus of variations, and replacing the variable $\dot{\boldsymbol{\Lambda}}$ by $\boldsymbol{\Lambda}$ the semidiscrete equations of motion for a coupled system are obtained as:

$$\boldsymbol{M}^k \ddot{\boldsymbol{U}}^k + \boldsymbol{K}^k \boldsymbol{U}^k + \boldsymbol{C}^{k^{\mathrm{T}}} \boldsymbol{\Lambda} = \boldsymbol{P}^k \qquad \forall k : 1 \leq k \leq S \tag{4.20}$$

$$\sum_{k=1}^{S} \boldsymbol{C}^k \dot{\boldsymbol{U}}^k = \boldsymbol{0} \tag{4.21}$$

where the unknowns are the kinematic quantities from all the subdomains and the Lagrange

multipliers $\mathbf{\Lambda}$ which can be interpreted as the interface reaction forces acting internally between the various subdomains.

4.2.2 Algebraically Partitioned FETI

An alternate derivation of the coupled equations of motion can be obtained by algebraically partitioning the domain Ω into S subdomains. The global interface, denoted Γ_I, is union of all the subdomain interfaces Γ_I^k. Let the Boolean connectivity matrix $\boldsymbol{C}^k$ pick out dofs corresponding to Γ_I^k from Ω_k and the Boolean connectivity matrix $\boldsymbol{B}^k$ pick out dofs corresponding to Γ_I^k from Γ_I.

For illustration purposes, we will choose to impose continuity of displacement across the interface Γ_I:

$$\boldsymbol{g}^k(\boldsymbol{U}^k, \boldsymbol{U}^I) \equiv \boldsymbol{C}^k\dot{\boldsymbol{U}}^k - \boldsymbol{B}^k\dot{\boldsymbol{U}}^I = 0 \qquad \forall k : 1 \leq k \leq S \tag{4.22}$$

Defining the constrained Lagrangian:

$$\tilde{\mathcal{L}} = \sum_{k=1}^{S} \left[\frac{1}{2}\dot{\boldsymbol{U}}^{k^{\mathrm{T}}}\boldsymbol{M}^k\dot{\boldsymbol{U}}^k - \frac{1}{2}\boldsymbol{U}^{k^{\mathrm{T}}}\boldsymbol{K}^k\boldsymbol{U}^k + \boldsymbol{\Lambda}^{k^T}\boldsymbol{g}^k \right] \tag{4.23}$$

Taking the variation of $\tilde{\mathcal{L}}$:

$$\delta\tilde{\mathcal{L}} = \sum_{k=1}^{S} \left[\delta\dot{\boldsymbol{U}}^{k^{\mathrm{T}}}\boldsymbol{M}^k\dot{\boldsymbol{U}}^k - \delta\boldsymbol{U}^{k^{\mathrm{T}}}\boldsymbol{K}^k\boldsymbol{U}^k + \delta\dot{\boldsymbol{U}}^{k^{\mathrm{T}}}\boldsymbol{C}^{k^{\mathrm{T}}}\boldsymbol{\Lambda}^k + \delta\dot{\boldsymbol{U}}^{I^{\mathrm{T}}}\boldsymbol{B}^{k^{\mathrm{T}}}\boldsymbol{\Lambda}^k + \delta\boldsymbol{\Lambda}^{\mathrm{T}}\boldsymbol{g}^k \right] \tag{4.24}$$

Using the Hamilton's principle and integrating the first term by parts, we get:

$$\begin{aligned} &\int_{t_1}^{t_2} \sum_{k=1}^{S} -\delta\boldsymbol{U}^{k^{\mathrm{T}}} \left[\boldsymbol{M}^k\ddot{\boldsymbol{U}}^k + \boldsymbol{K}^k\boldsymbol{U}^k + \boldsymbol{C}^{k^{\mathrm{T}}}\dot{\boldsymbol{\Lambda}}^k - \boldsymbol{P}^k \right] dt \\ &+ \int_{t_1}^{t_2} \sum_{k=1}^{S} \delta\boldsymbol{\Lambda}^{k^{\mathrm{T}}} \left[\boldsymbol{C}^k\dot{\boldsymbol{U}}^k - \boldsymbol{B}^k\dot{\boldsymbol{U}}^I \right] dt + \int_{t_1}^{t_2} \delta\dot{\boldsymbol{U}}^{I^{\mathrm{T}}} \left[\sum_{k=1}^{S} \boldsymbol{B}^{k^{\mathrm{T}}}\boldsymbol{\Lambda}^k \right] dt = 0 \end{aligned} \tag{4.25}$$

Using the fundamental theorem of the calculus of variations, and replacing the variable $\dot{\boldsymbol{\Lambda}}$

by $\mathbf{\Lambda}$, the semidiscrete equations of motion for an algebraically partitioned system are:

$$\boldsymbol{M}^k\ddot{\boldsymbol{U}}^k + \boldsymbol{K}^k\boldsymbol{U}^k + \boldsymbol{C}^{k^{\mathrm{T}}}\boldsymbol{\Lambda}^k = \boldsymbol{P}^k \qquad \forall k : 1 \leq k \leq S \tag{4.26}$$

$$\boldsymbol{C}^k\dot{\boldsymbol{U}}^k = \boldsymbol{B}^k\dot{\boldsymbol{U}}^I \qquad \forall k : 1 \leq k \leq S \tag{4.27}$$

$$\sum_{k=1}^{S} \boldsymbol{B}^{k^T}\boldsymbol{\Lambda}^k = \mathbf{0} \tag{4.28}$$

Note that in the semi-discrete form, continuity of displacements *implies* continuity of velocities and accelerations. However, in the time discretized form, one can impose the continuity of only one these kinematic quantities and continuity of velocities is shown to have stable behavior.

4.3 Implementation of FETI Interfaces

Let the continuous problem domain Ω be decomposed into S subdomains Ω_k for $1 \leq k \leq S$ as shown in figure 4.1(a). This partitioning creates the interface Γ_b of shared nodes between the subdomains. Each subdomain can then be solved using the appropriate time integration scheme with the optimal time step. As shown in figure 4.1(b), there are additional forces $\boldsymbol{R}(t)$ acting on a subdomain resulting from its interaction with the rest of the mesh.

The global degrees of freedom can be decomposed into degrees of freedom of the component subdomains as shown in figure 4.2. Let N be the total number of degrees of freedom in the unpartitioned structure and N^k be the number of degrees of freedom in subdomain Ω_k. Similarly, let L be the number of degrees of freedom in Ω lying along Γ_b and L^k be the number of degrees of freedom in subdomain Ω_k lying along Γ_b. A degree of freedom lying on the boundary Γ_b may be shared by two or more subdomains and may have to be replicated for each of the corresponding subdomains.

Let $P^{k,j}$ be the number of degrees of freedom in subdomain Ω_k that are shared with subdomain Ω_j as depicted in figure 4.3(a). Continuity constraints for the shared degrees of

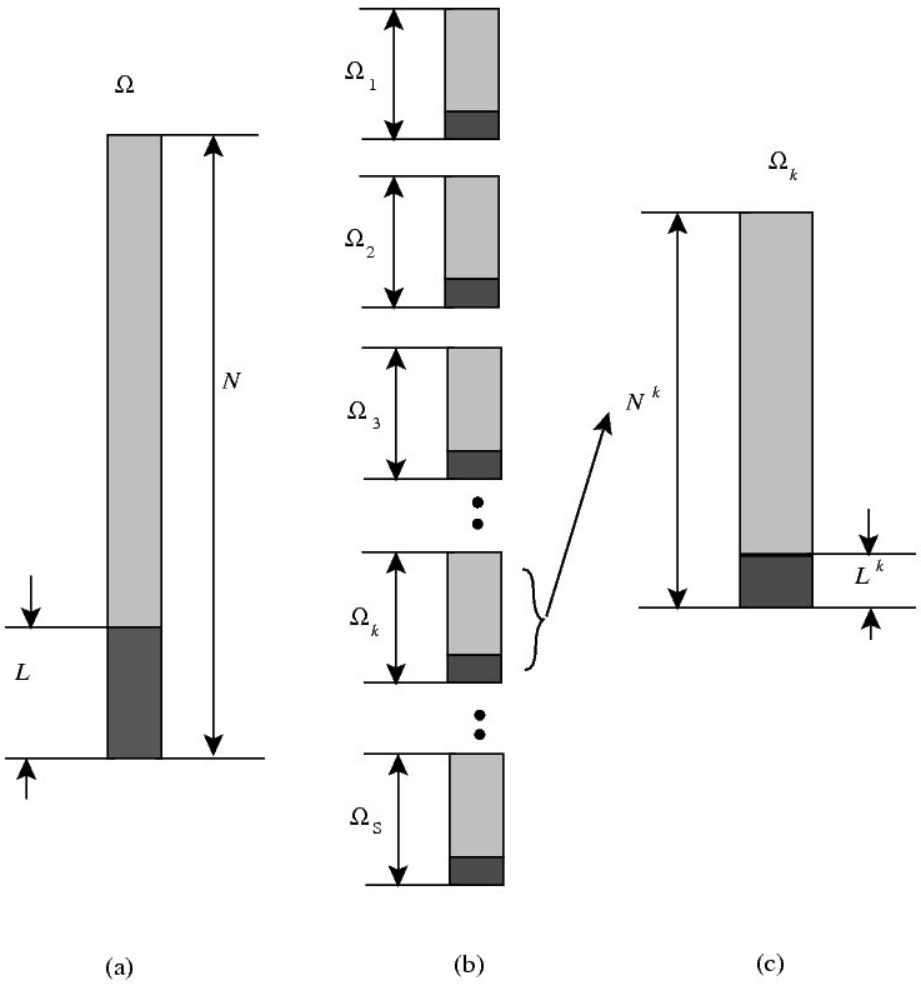

Figure 4.2: (a) Degrees of freedom for the unpartitioned system. (b) Degrees of freedom partitioned among subdomains. (c) Degrees of freedom for one subdomain.

freedom $P^{k,j}(= P^{j,k})$ may be written as:

$$\boldsymbol{V}^k|_{P^{k,j}} = \boldsymbol{V}^j|_{P^{j,k}} \quad \forall k : 1 \leq k \leq S \quad \text{and} \quad \forall j : k < j \leq S \tag{4.29}$$

where $\boldsymbol{V}^k$ and $\boldsymbol{V}^j$ are the solution quantities obtained from subdomains Ω_k and Ω_j respectively and the equality is assumed to hold only between the corresponding degrees of freedom. Note that j varies between $(k+1)$ and S to avoid duplication of constraints (shaded degrees of freedom in figure 4.3(a)). Each row in figure 4.3(a) represents one constraint corresponding to a single shared degree of freedom. Note that for those degrees of freedom shared

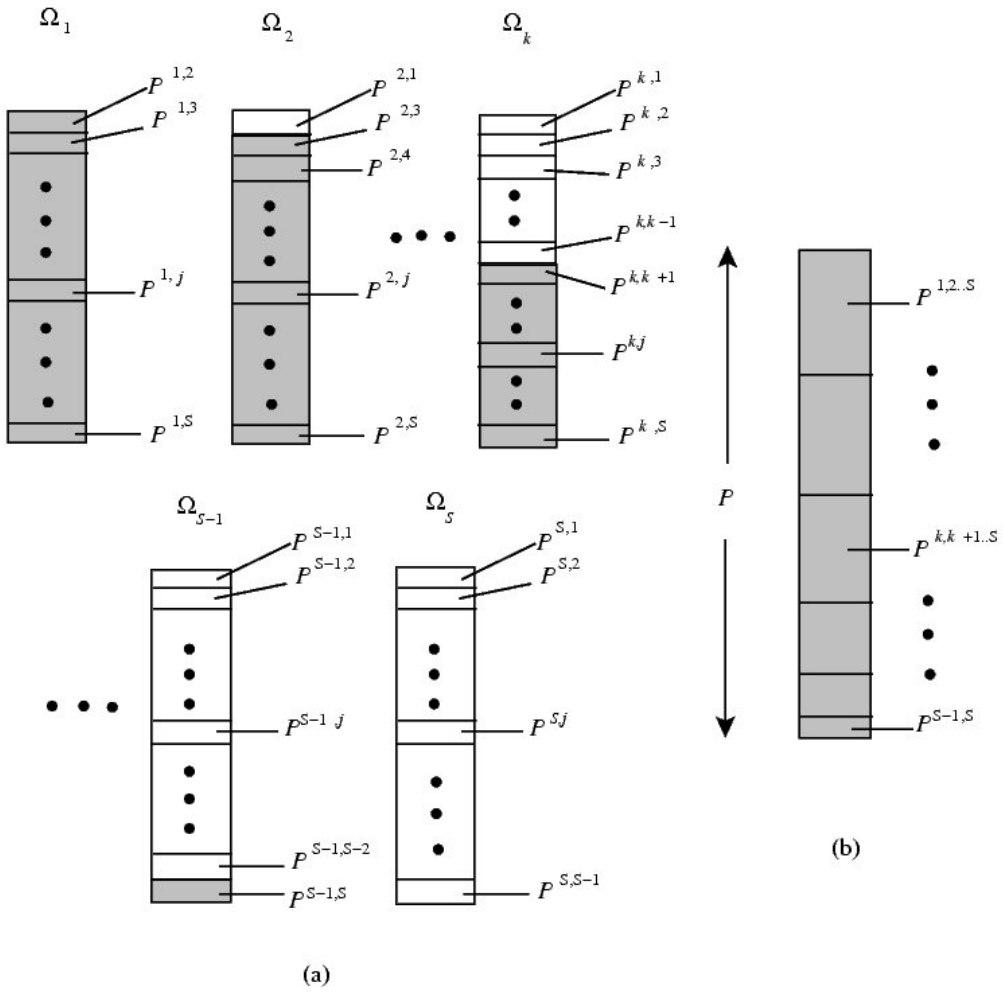

Figure 4.3: (a) Shared degrees of freedom for each subdomain. Shaded degrees of freedom represent constraints. (b) Shared degrees of freedom concatenated for Γ_b.

by three or more subdomains, the continuity constraints need to be enforced between all combinations of the participating subdomains taken two at a time. For example, if a node is shared between subdomains numbered 1,2 and 3 then the constraints to be enforced would be:

$$\boldsymbol{V}^1|_{P^{1,2}} = \boldsymbol{V}^2|_{P^{2,1}} \quad \boldsymbol{V}^1|_{P^{1,3}} = \boldsymbol{V}^3|_{P^{3,1}} \quad \boldsymbol{V}^2|_{P^{2,3}} = \boldsymbol{V}^3|_{P^{3,2}} \tag{4.30}$$

The shared degrees of freedom from all the subdomains may be concatenated together to

form a global interface vector for the entire boundary Γ_b as shown in figure 4.3(b) where P is total number of constraints to be enforced.

To enforce the constraints in equation (4.29) we introduce a matrix operator $\boldsymbol{C}^k$, one for each subdomain, that picks out those degrees of freedom from the subdomain that are shared with other subdomains and places them in the appropriate rows of the global interface vector. A simple example for two subdomains A and B is shown in figure 4.4. The $\boldsymbol{C}$ matrices for this decomposition are:

$$
\begin{array}{c|cc}
 & \boldsymbol{C}^{\mathrm{A}} & \boldsymbol{C}^{\mathrm{B}} \\
\begin{matrix}1\\2\\3\end{matrix} & \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} & \begin{bmatrix} 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{matrix}1\\2\\3\end{matrix} \\
 & \begin{matrix}1&2&3&4&5&6\end{matrix} & \begin{matrix}1&2&3&4&5&6\end{matrix}
\end{array}
$$

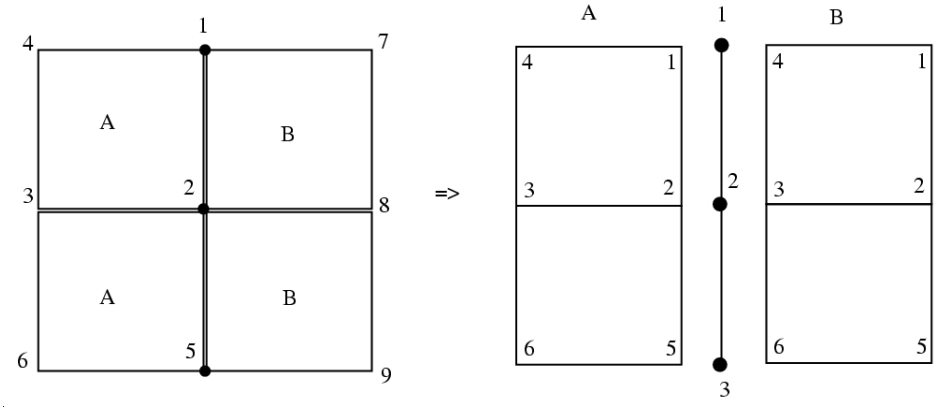

Figure 4.4: Decomposition of a domain into two subdomains A and B.

Note that the $\boldsymbol{C}$ matrices transform a subdomain vector into an interface vector and $\boldsymbol{C}^T$ matrices transform an interface vector into a subdomain vector. For the example in figure 4.4, continuity constraints are given by:

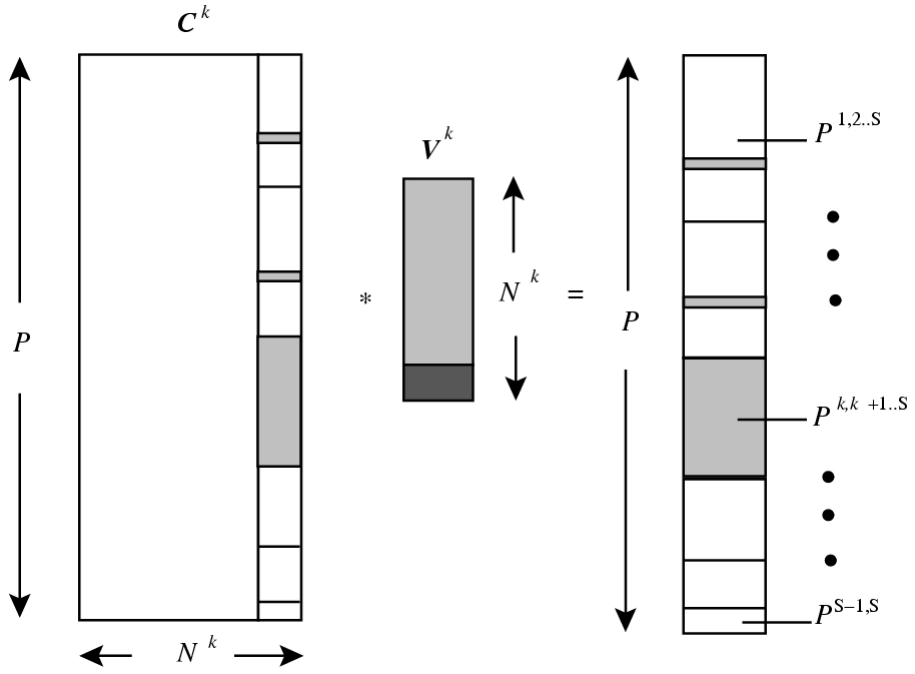

Figure 4.5: Structure of a typical $\boldsymbol{C}$ matrix for subdomain Ω_k.

$$\boldsymbol{C}^{\mathrm{A}}\boldsymbol{V}^{\mathrm{A}} + \boldsymbol{C}^{\mathrm{B}}\boldsymbol{V}^{\mathrm{B}} = \boldsymbol{0}$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} V_1^{\mathrm{A}} \\ V_2^{\mathrm{A}} \\ V_3^{\mathrm{A}} \\ V_4^{\mathrm{A}} \\ V_5^{\mathrm{A}} \\ V_6^{\mathrm{A}} \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} V_1^{\mathrm{B}} \\ V_2^{\mathrm{B}} \\ V_3^{\mathrm{B}} \\ V_4^{\mathrm{B}} \\ V_5^{\mathrm{B}} \\ V_6^{\mathrm{B}} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{aligned} V_1^{\mathrm{A}} &= V_4^{\mathrm{B}} \\ \Rightarrow\ V_2^{\mathrm{A}} &= V_3^{\mathrm{B}} \\ V_5^{\mathrm{A}} &= V_6^{\mathrm{B}} \end{aligned}$$

Interface forces on each subdomain are given by:

$$
\begin{array}{c|c}
\boldsymbol{C}^{\mathrm{A}^{\mathrm{T}}}\boldsymbol{\Lambda} & \boldsymbol{C}^{\mathrm{B}^{\mathrm{T}}}\boldsymbol{\Lambda} \\
\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{bmatrix} = \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ 0 \\ 0 \\ \lambda_3 \\ 0 \end{bmatrix} &
\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -\lambda_2 \\ -\lambda_1 \\ 0 \\ -\lambda_3 \end{bmatrix}
\end{array}
$$

The $\boldsymbol{C}^k$ matrix consists of binary elements and typically has the structure shown in figure 4.5 where the blank regions represent "zeros" and the shaded regions represent possible "ones" (or negative ones). Note that $\boldsymbol{C}^{k^{\mathrm{T}}}$ operates on vector quantities defined on the global interface, picks out components corresponding to the shared degrees of freedom in the subdomain Ω_k and places them in the appropriate rows of the subdomain vector.

4.4 The Multi-time-step Coupling Method

For simplicity consider a problem with two subdomains A and B. Let the subdomains be integrated with time steps ΔT and Δt respectively where $\Delta T = m\Delta t$. The Newmark parameters for the two subdomains are given by (γ_A, β_A) and (γ_B, β_B) respectively. For notational simplicity we will illustrate the coupling method for advancing the solution by ΔT from t_0 to $t_m = t_0 + \Delta T$ as shown in Figure 4.6. These ideas can be easily generalized for advancing the solution from t_n to t_{n+m}.

The semi discretized equations of motion that one obtains from Hamilton's principle for the two subdomains with continuity of velocities enforced at the interface are:

$$\boldsymbol{M}^A\ddot{\boldsymbol{U}}^A + \boldsymbol{K}^A\boldsymbol{U}^A + \boldsymbol{C}^{A^T}\boldsymbol{\Lambda} - \boldsymbol{P}^A = \boldsymbol{0} \tag{4.31}$$

$$\boldsymbol{M}^B\ddot{\boldsymbol{U}}^B + \boldsymbol{K}^B\boldsymbol{U}^B + \boldsymbol{C}^{B^T}\boldsymbol{\Lambda} - \boldsymbol{P}^B = \boldsymbol{0} \tag{4.32}$$

$$\boldsymbol{C}^A\dot{\boldsymbol{U}}^A + \boldsymbol{C}^B\dot{\boldsymbol{U}}^B = \boldsymbol{0} \tag{4.33}$$

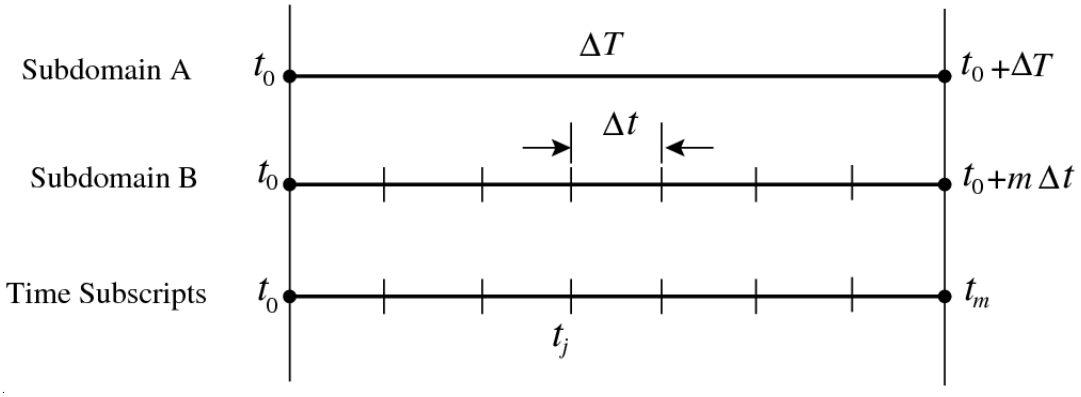

Figure 4.6: Representation of time steps for the two subdomain case.

The fully discretized equations for subdomain A can be written as:

$$\boldsymbol{M}^A\ddot{\boldsymbol{U}}_m^A + \boldsymbol{K}^A\boldsymbol{U}_m^A + \boldsymbol{C}^{A^T}\boldsymbol{\Lambda}_m - \boldsymbol{P}_m^A = \boldsymbol{0} \tag{4.34}$$

$$\dot{\boldsymbol{U}}_m^A - \left[\dot{\boldsymbol{U}}_0^A + \Delta T\,(1-\gamma_A)\,\ddot{\boldsymbol{U}}_0^A + \gamma_A \Delta T \ddot{\boldsymbol{U}}_m^A\right] = \boldsymbol{0} \tag{4.35}$$

$$\boldsymbol{U}_m^A - \left[\boldsymbol{U}_0^A + \Delta T\;\dot{\boldsymbol{U}}_0^A + \Delta T^2\left(\tfrac{1}{2} - \beta_A\right)\ddot{\boldsymbol{U}}_0^A + \beta_A \Delta T^2 \ddot{\boldsymbol{U}}_m^A\right] = \boldsymbol{0} \tag{4.36}$$

where the second and the third equations are obtained from the Newmark relations. Similarly for subdomain B the fully discretized set of equations $\forall j = 1, 2, \cdots, m$ can be written as:

$$\boldsymbol{M}^B\ddot{\boldsymbol{U}}_j^B + \boldsymbol{K}^B\boldsymbol{U}_j^B + \boldsymbol{C}^{B^T}\boldsymbol{\Lambda}_j - \boldsymbol{P}_j^B = \boldsymbol{0} \tag{4.37}$$

$$\dot{\boldsymbol{U}}_j^B - \left[\dot{\boldsymbol{U}}_{j-1}^B + \Delta t\,(1-\gamma_B)\,\ddot{\boldsymbol{U}}_{j-1}^B + \gamma_B \Delta t \ddot{\boldsymbol{U}}_j^B\right] = \boldsymbol{0} \tag{4.38}$$

$$\boldsymbol{U}_j^B - \left[\boldsymbol{U}_{j-1}^B + \Delta t\;\dot{\boldsymbol{U}}_{j-1}^B + \Delta t^2\left(\tfrac{1}{2} - \beta_B\right)\ddot{\boldsymbol{U}}_{j-1}^B + \beta_B \Delta t^2 \ddot{\boldsymbol{U}}_j^B\right] = \boldsymbol{0} \tag{4.39}$$

The equation of continuity of velocities at the interface which determines the interface reaction force $\boldsymbol{\Lambda}_m$ at the big time step ΔT is written as:

$$\boldsymbol{C}^A\dot{\boldsymbol{U}}_m^A + \boldsymbol{C}^B\dot{\boldsymbol{U}}_m^B = \boldsymbol{0} \tag{4.40}$$

In addition to the above equations, one needs $(m-1)$ equations to compute the interface reaction forces $\boldsymbol{\Lambda}_j \ \forall j = 1, 2, \cdots, (m-1)$ at all the intermediate time steps of Δt. Since the interface reaction forces at the intermediate time steps should be in balance between both the subdomains, therefore the following equations from subdomain A are required to balance the reactions from subdomain B:

$$\boldsymbol{C}^A \left[\boldsymbol{M}^A \ddot{\boldsymbol{U}}_j^A + \boldsymbol{K}^A \boldsymbol{U}_j^A + \boldsymbol{C}^{A^T} \boldsymbol{\Lambda}_j - \boldsymbol{P}_j^A \right] = \boldsymbol{0} \quad \forall j = 1, 2, \cdots, (m-1) \tag{4.41}$$

where the kinematic quantities for subdomain A at intermediate time steps are obtained by linearly interpolating between the end points as:

$$\ddot{\boldsymbol{U}}_j^A = \left(1 - \frac{j}{m}\right) \ddot{\boldsymbol{U}}_0^A + \left(\frac{j}{m}\right) \ddot{\boldsymbol{U}}_m^A \tag{4.42}$$

$$\dot{\boldsymbol{U}}_j^A = \left(1 - \frac{j}{m}\right) \dot{\boldsymbol{U}}_0^A + \left(\frac{j}{m}\right) \dot{\boldsymbol{U}}_m^A \tag{4.43}$$

$$\boldsymbol{U}_j^A = \left(1 - \frac{j}{m}\right) \boldsymbol{U}_0^A + \left(\frac{j}{m}\right) \boldsymbol{U}_m^A \tag{4.44}$$

Note that the system of equations (4.34)-(4.44) has $7m$ unknowns and the same number of equations and can now be solved. The vector of unknowns is:

$$\left[\mathbb{U}_1^B, \mathbb{U}_2^B, \cdots, \mathbb{U}_m^B \mid \mathbb{U}_m^A \mid \mathbb{U}_1^A, \mathbb{U}_2^A, \cdots, \mathbb{U}_{m-1}^A \mid \boldsymbol{\Lambda}_1, \boldsymbol{\Lambda}_2, \cdots, \boldsymbol{\Lambda}_{m-1} \mid \boldsymbol{\Lambda}_m \right]^T$$

Using the notation of section §4.1, one can define subdomain block matrices:

$$\mathbb{M}^k = \begin{bmatrix} \boldsymbol{M}^k & 0 & \boldsymbol{K}^k \\ -\gamma_k \Delta t_k \boldsymbol{I}^k & \boldsymbol{I}^k & 0 \\ -\beta_k \Delta t_k^2 \boldsymbol{I}^k & 0 & \boldsymbol{I}^k \end{bmatrix} \ ; \quad \mathbb{N}^k = \begin{bmatrix} 0 & 0 & 0 \\ -\Delta t_k (1 - \gamma_k) \boldsymbol{I}^k & -\boldsymbol{I}^k & 0 \\ -\Delta t_k^2 (\frac{1}{2} - \beta_k) \boldsymbol{I}^k & -\Delta t_k \boldsymbol{I}^k & -\boldsymbol{I}^k \end{bmatrix} \tag{4.45}$$

$$\mathbb{C}^k = \begin{bmatrix} \boldsymbol{C}^{k^T} \\ 0 \\ 0 \end{bmatrix} \; ; \quad \mathbb{U}_j^k = \begin{bmatrix} \ddot{\boldsymbol{U}}_j^k \\ \dot{\boldsymbol{U}}_j^k \\ \boldsymbol{U}_j^k \end{bmatrix} \; ; \quad \mathbb{I}^k = \begin{bmatrix} \boldsymbol{I}^k & 0 & 0 \\ 0 & \boldsymbol{I}^k & 0 \\ 0 & 0 & \boldsymbol{I}^k \end{bmatrix} \tag{4.46}$$

$$\mathbb{P}_j^B = \begin{bmatrix} \boldsymbol{P}_j^B \\ 0 \\ 0 \end{bmatrix} \; \forall j \in [1, m] \; ; \quad \mathbb{P}_m^A = \begin{bmatrix} \boldsymbol{P}_m^A \\ 0 \\ 0 \end{bmatrix} \; ; \quad \mathbb{P}_j^A = \boldsymbol{C}^A \boldsymbol{P}_j^A \;\; \forall j \in [1, (m-1)] \tag{4.47}$$

$$\mathbb{J}_j^A = -\frac{j}{m}\mathbb{I}^A \qquad \mathbb{L}_j^A = (1 - \frac{j}{m})\mathbb{U}_0^A \qquad \forall j \in [1, (m-1)] \tag{4.48}$$

$$\mathbb{R}^A = \begin{bmatrix} \boldsymbol{C}^A \boldsymbol{M}^A & 0 & \boldsymbol{C}^A \boldsymbol{K}^A \end{bmatrix} \qquad \mathbb{B}^k = \begin{bmatrix} 0 & \boldsymbol{C}^k & 0 \end{bmatrix} \tag{4.49}$$

One may verify that the system of equations (4.34)-(4.44) is completely represented in the matrix equation:

$$\left[\begin{array}{c|c|c||c|c}
\mathbb{M}^B & & & \mathbb{C}^B & \\
\mathbb{N}^B\ \mathbb{M}^B & & & \quad \mathbb{C}^B & \\
\quad \ddots\ \ddots & & & \qquad \ddots & \\
\quad \mathbb{N}^B\ \mathbb{M}^B & & & & \mathbb{C}^B \\
\hline
 & \mathbb{M}^A & & & \mathbb{C}^A \\
\hline
 & \mathbb{J}_1^A & \mathbb{I}^A & & \\
 & \mathbb{J}_2^A & \quad \mathbb{I}^A & & \\
 & \vdots & \qquad \ddots & & \\
 & \mathbb{J}_{m-1}^A & \qquad \mathbb{I}^A & & \\
\hline\hline
 & & \mathbb{R}^A & \boldsymbol{I}^R & \\
 & & \quad \mathbb{R}^A & \quad \boldsymbol{I}^R & \\
 & & \qquad \ddots & \qquad \ddots & \\
 & & \qquad \mathbb{R}^A & \qquad \boldsymbol{I}^R & \\
\hline
\mathbb{B}^B & \mathbb{B}^A & & & \mathbf{0}
\end{array}\right]
\left[\begin{array}{c}
\mathbb{U}_1^B \\ \mathbb{U}_2^B \\ \vdots \\ \mathbb{U}_m^B \\ \hline \mathbb{U}_m^A \\ \hline \mathbb{U}_1^A \\ \mathbb{U}_2^A \\ \vdots \\ \mathbb{U}_{m-1}^A \\ \hline\hline \boldsymbol{\Lambda}_1 \\ \boldsymbol{\Lambda}_2 \\ \vdots \\ \boldsymbol{\Lambda}_{m-1} \\ \hline \boldsymbol{\Lambda}_m
\end{array}\right]
=
\left[\begin{array}{c}
\mathbb{P}_1^B - \mathbb{N}^B \mathbb{U}_0^B \\ \mathbb{P}_2^B \\ \vdots \\ \mathbb{P}_m^B \\ \hline \mathbb{P}_m^A - \mathbb{N}^A \mathbb{U}_0^A \\ \hline \mathbb{L}_1^A \\ \mathbb{L}_2^A \\ \vdots \\ \mathbb{L}_{m-1}^A \\ \hline\hline \mathbb{P}_1^A \\ \mathbb{P}_2^A \\ \vdots \\ \mathbb{P}_{m-1}^A \\ \hline \mathbf{0}
\end{array}\right] \tag{4.50}$$

Here we have used the identity $\boldsymbol{C}^A\boldsymbol{C}^{A^T} = \boldsymbol{I}^R$ where $\boldsymbol{I}^R$ is an identity matrix of size equal to the degrees of freedom along the interface between A and B. For general cases with more than two subdomains having a common node, special care needs to be taken in formulating this matrix.

4.4.1 Bordered Solution Procedure

The matrix equation (4.50) can be solved by a bordered procedure by partitioning the matrix along the double lines in equation (4.50) as:

$$\begin{bmatrix} \mathsf{M} & \mathsf{B} \\ \mathsf{C} & \mathsf{G} \end{bmatrix} \begin{bmatrix} \mathsf{U} \\ \Lambda \end{bmatrix} = \begin{bmatrix} \mathsf{P} \\ \mathsf{Q} \end{bmatrix} \tag{4.51}$$

Let

$$\begin{aligned} \mathsf{U} &= \mathsf{V} + \mathsf{W} \\ \text{where} \quad \mathsf{V} &= \mathsf{M}^{-1}\mathsf{P} \\ \text{and} \quad \mathsf{W} &= -\mathsf{Y}\Lambda \quad ; \quad \mathsf{Y} = \mathsf{M}^{-1}\mathsf{B} \end{aligned} \tag{4.52}$$

Note the first row in equation (4.51) is satisfied:

$$\begin{aligned} \mathsf{M}\mathsf{U} + \mathsf{B}\Lambda &= \mathsf{P} \\ \mathsf{M}\mathsf{V} + \mathsf{M}\mathsf{W} + \mathsf{B}\Lambda &= \mathsf{P} \\ \mathsf{M}\mathsf{Y}\Lambda - \mathsf{B}\Lambda &= (\mathsf{M}\mathsf{V} - \mathsf{P}) \\ (\mathsf{M}\mathsf{Y} - \mathsf{B})\Lambda &= (\mathsf{M}\mathsf{V} - \mathsf{P}) \end{aligned} \tag{4.53}$$

The second row is used for computing the interface reactions:

$$\begin{aligned} \mathsf{CU} + \mathsf{G\Lambda} &= \mathsf{Q} \\ \mathsf{CV} + \mathsf{CW} + \mathsf{G\Lambda} &= \mathsf{Q} \\ [-\mathsf{CY} + \mathsf{G}]\mathsf{\Lambda} &= \mathsf{Q} - \mathsf{CV} \end{aligned} \tag{4.54}$$

The quantities in equation (4.52) can be computed as follows:

Computing $\mathsf{MV} = \mathsf{P}$:

$$\left[\begin{array}{ccc|c|c} \mathbb{M}^B & & & & \\ \mathbb{N}^B & \mathbb{M}^B & & & \\ & \ddots & \ddots & & \\ & & \mathbb{N}^B \;\; \mathbb{M}^B & & \\ \hline & & & \mathbb{M}^A & \\ \hline & & & \mathbb{J}_1^A & \mathbb{I}^A \\ & & & \mathbb{J}_2^A & \mathbb{I}^A \\ & & & \vdots & \ddots \\ & & & \mathbb{J}_{m-1}^A & \mathbb{I}^A \end{array}\right] \left[\begin{array}{c} \mathbb{V}_1^B \\ \mathbb{V}_2^B \\ \vdots \\ \mathbb{V}_m^B \\ \hline \mathbb{V}_m^A \\ \hline \mathbb{V}_1^A \\ \mathbb{V}_2^A \\ \vdots \\ \mathbb{V}_{m-1}^A \end{array}\right] = \left[\begin{array}{c} \mathbb{P}_1^B \\ \mathbb{P}_2^B \\ \vdots \\ \mathbb{P}_m^B \\ \hline \mathbb{P}_m^A \\ \hline \mathbb{L}_1^A \\ \mathbb{L}_2^A \\ \vdots \\ \mathbb{L}_{m-1}^A \end{array}\right] \tag{4.55}$$

This system is solved simply by time stepping through subdomain B by m time steps of size Δt each and time stepping subdomain A once through a time step of ΔT under external loads only. This time stepping yields the values for $\mathbb{V}_1^B$, $\mathbb{V}_2^B$ ... $\mathbb{V}_m^B$ and $\mathbb{V}_m^A$ respectively. The values for $\mathbb{V}_1^A$, $\mathbb{V}_2^A$ ... $\mathbb{V}_{m-1}^A$ at the intermediate time steps in subdomain A are obtained by linearly interpolating between $\mathbb{U}_0^A$ and $\mathbb{V}_m^A$.

Computing $\mathsf{MY} = \mathsf{B}$:

$$\left[\begin{array}{c|c|c} \mathbb{M}^B & & \\ \mathbb{N}^B\,\mathbb{M}^B & & \\ \ddots\ \ddots & & \\ \mathbb{N}^B\,\mathbb{M}^B & & \\ \hline & \mathbb{M}^A & \\ \hline & \mathbb{J}_1^A & \mathbb{I}^A \\ & \mathbb{J}_2^A & \mathbb{I}^A \\ & \vdots & \ddots \\ & \mathbb{J}_{m-1}^A & \mathbb{I}^A \end{array}\right] \left[\begin{array}{c} \mathbb{Y}_1^B \\ \mathbb{Y}_2^B\ \ \mathbb{Y}_1^B \\ \vdots\ \ \vdots\ \ \ddots \\ \mathbb{Y}_m^B\,\mathbb{Y}_{m-1}^B \cdots\ \mathbb{Y}_1^B \\ \hline \mathbb{Y}_m^A \\ \hline \mathbb{Y}_1^A \\ \mathbb{Y}_2^A \\ \vdots \\ \mathbb{Y}_{m-1}^A \end{array}\right] = \left[\begin{array}{c} \mathbb{C}^B \\ \mathbb{C}^B \\ \ddots \\ \mathbb{C}^B \\ \hline \mathbb{C}^A \\ \hline 0 \\ 0 \\ \vdots \\ 0 \end{array}\right] \tag{4.56}$$

Note that the structure of Y shown above is obtained when one solves for it columnwise taking one column at a time from the right hand side. For example the first row yields:

$$\begin{gathered} \mathbb{M}^B\mathbb{Y}_1^B = \mathbb{C}^B \\ \left[\begin{array}{ccc} \boldsymbol{M}^B & 0 & \boldsymbol{K}^B \\ -\gamma_B\Delta t\boldsymbol{I}^B & \boldsymbol{I}^B & 0 \\ -\beta_B\Delta t^2\boldsymbol{I}^B & 0 & \boldsymbol{I}^B \end{array}\right] \left[\begin{array}{c} \ddot{\boldsymbol{Y}}_1^B \\ \dot{\boldsymbol{Y}}_1^B \\ \boldsymbol{Y}_1^B \end{array}\right] = \left[\begin{array}{c} \boldsymbol{C}^{B^T} \\ 0 \\ 0 \end{array}\right] \end{gathered} \tag{4.57}$$

Let $\tilde{\boldsymbol{M}}^B = \boldsymbol{M}^B + \beta_B\Delta t^2\boldsymbol{K}^B$. Thus

$$\begin{aligned} \ddot{\boldsymbol{Y}}_1^B &= \tilde{\boldsymbol{M}}^{B^{-1}}\boldsymbol{C}^{B^T} \\ \dot{\boldsymbol{Y}}_1^B &= \gamma_B\Delta t\tilde{\boldsymbol{M}}^{B^{-1}}\boldsymbol{C}^{B^T} \\ \boldsymbol{Y}_1^B &= \beta_B\Delta t^2\tilde{\boldsymbol{M}}^{B^{-1}}\boldsymbol{C}^{B^T} \end{aligned} \tag{4.58}$$

For the subsequent rows in the first column:

$$\mathbb{N}^B\mathbb{Y}^B_{j-1} + \mathbb{M}^B\mathbb{Y}^B_j = \mathbf{0} \quad \Longrightarrow \; \mathbb{M}^B\mathbb{Y}^B_j = -\mathbb{N}^B\mathbb{Y}^B_{j-1}$$

$$\begin{bmatrix} \boldsymbol{M}^B & 0 & \boldsymbol{K}^B \\ -\gamma_B\Delta t\boldsymbol{I}^B & \boldsymbol{I}^B & 0 \\ -\beta_B\Delta t^2\boldsymbol{I}^B & 0 & \boldsymbol{I}^B \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{Y}}^B_j \\ \dot{\boldsymbol{Y}}^B_j \\ \boldsymbol{Y}^B_j \end{bmatrix} = - \begin{bmatrix} 0 & 0 & 0 \\ -\Delta t(1-\gamma_B)\boldsymbol{I}^B & -\boldsymbol{I}^B & 0 \\ -\Delta t^2(\frac{1}{2}-\beta_B)\boldsymbol{I}^B & -\Delta t\boldsymbol{I}^B & -\boldsymbol{I}^B \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{Y}}^B_{j-1} \\ \dot{\boldsymbol{Y}}^B_{j-1} \\ \boldsymbol{Y}^B_{j-1} \end{bmatrix} \tag{4.59}$$

Thus

$$\begin{bmatrix} \boldsymbol{M}^B & 0 & \boldsymbol{K}^B \\ -\gamma_B\Delta t\boldsymbol{I}^B & \boldsymbol{I}^B & 0 \\ -\beta_B\Delta t^2\boldsymbol{I}^B & 0 & \boldsymbol{I}^B \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{Y}}^B_j \\ \dot{\boldsymbol{Y}}^B_j \\ \boldsymbol{Y}^B_j \end{bmatrix} = \begin{bmatrix} 0 \\ \hat{\dot{\boldsymbol{Y}}}^B_j \\ \hat{\boldsymbol{Y}}^B_j \end{bmatrix} \tag{4.60}$$

where $\hat{\dot{\boldsymbol{Y}}}^B_j$ and $\hat{\boldsymbol{Y}}^B_j$ are obtained from Newmark relations as:

$$\begin{aligned} \hat{\dot{\boldsymbol{Y}}}^B_j &= \dot{\boldsymbol{Y}}^B_{j-1} + \Delta t\,(1-\gamma_B)\,\ddot{\boldsymbol{Y}}^B_{j-1} \\ \hat{\boldsymbol{Y}}^B_j &= \boldsymbol{Y}^B_{j-1} + \Delta t\,\dot{\boldsymbol{Y}}^B_{j-1} + \Delta t^2\left(\tfrac{1}{2}-\beta_B\right)\ddot{\boldsymbol{Y}}^B_{j-1} \end{aligned} \tag{4.61}$$

This amounts to solving for the subsequent values of $\mathbb{Y}^B_j \; \forall j \in [2, m]$ by time stepping the initial $\mathbb{Y}^B_1$ under *zero* load. For subdomain A, $\mathbb{Y}^A_m$ can also be solved in a similar way:

$$\mathbb{M}^A\mathbb{Y}^A_m = \mathbb{C}^A$$

$$\begin{bmatrix} \boldsymbol{M}^A & 0 & \boldsymbol{K}^A \\ -\gamma_A\Delta T\boldsymbol{I}^A & \boldsymbol{I}^A & 0 \\ -\beta_A\Delta T^2\boldsymbol{I}^A & 0 & \boldsymbol{I}^A \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{Y}}^A_m \\ \dot{\boldsymbol{Y}}^A_m \\ \boldsymbol{Y}^A_m \end{bmatrix} = \begin{bmatrix} \boldsymbol{C}^{A^T} \\ 0 \\ 0 \end{bmatrix} \tag{4.62}$$

Let $\tilde{\boldsymbol{M}}^A = \boldsymbol{M}^A + \beta_A\Delta T^2\boldsymbol{K}^A$. Thus

$$\begin{aligned} \ddot{\boldsymbol{Y}}^A_m &= \tilde{\boldsymbol{M}}^{A^{-1}}\boldsymbol{C}^{A^T} \\ \dot{\boldsymbol{Y}}^A_m &= \gamma_A\Delta T\tilde{\boldsymbol{M}}^{A^{-1}}\boldsymbol{C}^{A^T} \\ \boldsymbol{Y}^A_m &= \beta_A\Delta T^2\tilde{\boldsymbol{M}}^{A^{-1}}\boldsymbol{C}^{A^T} \end{aligned} \tag{4.63}$$

The intermediate values $\mathbb{Y}_j^A \; \forall j \in [1,(m-1)]$ are obtained by linearly interpolating between $\mathbf{0}$ and $\mathbb{Y}_m^A$.

Computing the Interface Matrix $[-\mathsf{CY}+\mathsf{G}]$:

$$-\mathsf{CY} = -\left[\begin{array}{ccc|c} & & \mathbb{R}^A & \\ & \mathbb{R}^A & & \\ \ddots & & & \\ & & \mathbb{R}^A & \\ \hline \mathbb{B}^B & \mathbb{B}^A & & \end{array}\right] \left[\begin{array}{c} \mathbb{Y}_1^B \\ \mathbb{Y}_2^B \quad \mathbb{Y}_1^B \\ \vdots \quad \vdots \quad \ddots \\ \mathbb{Y}_m^B \; \mathbb{Y}_{m-1}^B \cdots \; \mathbb{Y}_1^B \\ \hline \mathbb{Y}_m^A \\ \hline \mathbb{Y}_1^A \\ \mathbb{Y}_2^A \\ \vdots \\ \mathbb{Y}_{m-1}^A \end{array}\right] \tag{4.64}$$

$$-\mathsf{CY} = -\left[\begin{array}{ccc|c} & & & \mathbb{R}^A\mathbb{Y}_1^A \\ & & & \mathbb{R}^A\mathbb{Y}_2^A \\ & & & \vdots \\ & & & \mathbb{R}^A\mathbb{Y}_{m-1}^A \\ \hline \boldsymbol{C}^B\dot{\boldsymbol{Y}}_m^B & \boldsymbol{C}^B\dot{\boldsymbol{Y}}_{m-1}^B & \cdots & \boldsymbol{C}^B\dot{\boldsymbol{Y}}_2^B \end{array}\right] \left[\begin{array}{c} \\ \\ \\ \\ \hline \boldsymbol{C}^B\dot{\boldsymbol{Y}}_1^B + \boldsymbol{C}^A\dot{\boldsymbol{Y}}_m^A \end{array}\right] \tag{4.65}$$

where

$$\begin{aligned} \mathbb{R}^A\mathbb{Y}_j^A &= \left[\begin{array}{ccc} \boldsymbol{C}^A\boldsymbol{M}^A & 0 & \boldsymbol{C}^A\boldsymbol{K}^A \end{array}\right] \left[\begin{array}{c} \ddot{\boldsymbol{Y}}_j^A \\ \dot{\boldsymbol{Y}}_j^A \\ \boldsymbol{Y}_j^A \end{array}\right] \\ &= \boldsymbol{C}^A\left(\boldsymbol{M}^A\ddot{\boldsymbol{Y}}_j^A + \boldsymbol{K}^A\boldsymbol{Y}_j^A\right) \\ &= \left(\frac{j}{m}\right)\boldsymbol{C}^A\left(\boldsymbol{M}^A\left(\tilde{\boldsymbol{M}}^{A^{-1}}\boldsymbol{C}^{A^T}\right) + \boldsymbol{K}^A\left(\beta_A\Delta T^2\tilde{\boldsymbol{M}}^{A^{-1}}\boldsymbol{C}^{A^T}\right)\right) \\ &= \left(\frac{j}{m}\right)\boldsymbol{C}^A\left(\tilde{\boldsymbol{M}}^A\tilde{\boldsymbol{M}}^{A^{-1}}\right)\boldsymbol{C}^{A^T} \\ \implies \mathbb{R}^A\mathbb{Y}_j^A &= \left(\frac{j}{m}\right)\boldsymbol{I}^R \end{aligned} \tag{4.66}$$

Note that the term $\boldsymbol{C}^B\dot{\boldsymbol{Y}}_1^B + \boldsymbol{C}^A\dot{\boldsymbol{Y}}_m^A$ obtained here is exactly the same as that obtained by Gravouil and Combescure [93]:

$$\boldsymbol{H}_{GC} = \gamma_B \Delta t \boldsymbol{C}^B \tilde{\boldsymbol{M}}^{B^{-1}} \boldsymbol{C}^{B^T} + \gamma_A \Delta T \boldsymbol{C}^A \tilde{\boldsymbol{M}}^{A^{-1}} \boldsymbol{C}^{A^T} \tag{4.67}$$

Thus $[-\mathsf{CY} + \mathsf{G}]$:

$$\left[\begin{array}{cccc|c} & \boldsymbol{I}^R & & & -\left(\frac{1}{m}\right)\boldsymbol{I}^R \\ & & \boldsymbol{I}^R & & -\left(\frac{2}{m}\right)\boldsymbol{I}^R \\ & & & \ddots & \vdots \\ & & & \boldsymbol{I}^R & -\left(\frac{m-1}{m}\right)\boldsymbol{I}^R \\ \hline -\boldsymbol{C}^B\dot{\boldsymbol{Y}}_m^B & -\boldsymbol{C}^B\dot{\boldsymbol{Y}}_{m-1}^B & \cdots & -\boldsymbol{C}^B\dot{\boldsymbol{Y}}_2^B & -\boldsymbol{H}_{GC} \end{array}\right] \tag{4.68}$$

Computing the Interface Right Hand Side Vector $\{\mathsf{Q} - \mathsf{CV}\}$:

$$\left[\begin{array}{c} \boldsymbol{C^A P}_1^A \\ \boldsymbol{C^A P}_2^A \\ \vdots \\ \boldsymbol{C^A P}_{m-1}^A \\ \hline 0 \end{array}\right] - \left[\begin{array}{ccc|c} & & & \mathbb{R}^A & \\ & & & & \mathbb{R}^A & \\ & & & & & \ddots & \\ & & & & & & \mathbb{R}^A \\ \hline & \mathbb{B}^B & & \mathbb{B}^A & & & \end{array}\right] \left[\begin{array}{c} \mathbb{V}_1^B \\ \mathbb{V}_2^B \\ \vdots \\ \mathbb{V}_m^B \\ \hline \mathbb{V}_m^A \\ \hline \mathbb{V}_1^A \\ \mathbb{V}_2^A \\ \vdots \\ \mathbb{V}_{m-1}^A \end{array}\right] \tag{4.69}$$

Thus

$$\mathsf{Q}-\mathsf{CV}=\left[\begin{array}{c}\boldsymbol{C^A}\left(\boldsymbol{P}_1^A-\boldsymbol{M}^A\ddot{\boldsymbol{V}}_1^A-\boldsymbol{K}^A\boldsymbol{V}_1^A\right)\\ \boldsymbol{C^A}\left(\boldsymbol{P}_2^A-\boldsymbol{M}^A\ddot{\boldsymbol{V}}_2^A-\boldsymbol{K}^A\boldsymbol{V}_2^A\right)\\ \vdots\\ \boldsymbol{C^A}\left(\boldsymbol{P}_{m-1}^A-\boldsymbol{M}^A\ddot{\boldsymbol{V}}_{m-1}^A-\boldsymbol{K}^A\boldsymbol{V}_{m-1}^A\right)\\ \hline -\left(\boldsymbol{C}^B\dot{\boldsymbol{V}}_m^B+\boldsymbol{C}^A\dot{\boldsymbol{V}}_m^A\right)\end{array}\right] \tag{4.70}$$

Computing the Interface Reaction Forces Λ:

Now the interface equation $[\mathsf{G}-\mathsf{CY}]\{\Lambda\}=\{\mathsf{Q}-\mathsf{CV}\}$ can be solved for Λ using equations (4.68) and (4.70). Let $\boldsymbol{S}_j=\boldsymbol{C}^A\left(\boldsymbol{P}_j^A-\boldsymbol{M}^A\ddot{\boldsymbol{V}}_j^A-\boldsymbol{K}^A\boldsymbol{V}_j^A\right)\quad\forall j\in[1,(m-1)]$. Multiplying each row j (where $j\in[1,(m-1)]$) of the interface equation by $\boldsymbol{C}^B\dot{\boldsymbol{Y}}_{m-j+1}^B$ and adding it to the last row one obtains:

$$\left[-\boldsymbol{H}_{GC}-\sum_{j=1}^{m-1}\left(\frac{j}{m}\right)\boldsymbol{C}^B\dot{\boldsymbol{Y}}_{m-j+1}^B\right]\boldsymbol{\Lambda}_m=-\left(\boldsymbol{C}^B\dot{\boldsymbol{V}}_m^B+\boldsymbol{C}^A\dot{\boldsymbol{V}}_m^A\right)+\sum_{j=1}^{m-1}\boldsymbol{C}^B\dot{\boldsymbol{Y}}_{m-j+1}^B\boldsymbol{S}_j \tag{4.71}$$

which can be written as:

$$\boldsymbol{H}\boldsymbol{\Lambda}_m=-\left(\boldsymbol{C}^B\dot{\boldsymbol{V}}_m^B+\boldsymbol{C}^A\dot{\boldsymbol{V}}_m^A\right)+\sum_{j=1}^{m-1}\boldsymbol{C}^B\dot{\boldsymbol{Y}}_{m-j+1}^B\boldsymbol{S}_j \tag{4.72}$$

where

$$\boldsymbol{H}=-\boldsymbol{C}^A\dot{\boldsymbol{Y}}_m^A-\sum_{j=1}^{m}\left(\frac{j}{m}\right)\boldsymbol{C}^B\dot{\boldsymbol{Y}}_{m-j+1}^B \tag{4.73}$$

Using the values of $\boldsymbol{\Lambda}_m$ computed from equation (4.72) one can compute the interface reaction forces at the intermediate time steps as:

$$\boldsymbol{\Lambda}_j=\boldsymbol{S}_j+\left(\frac{j}{m}\right)\boldsymbol{\Lambda}_m\quad\forall j\in[1,(m-1)] \tag{4.74}$$

Computing the Updated Solution $\mathsf{W} = -\mathsf{Y}\Lambda$:

$$\mathsf{W} = \begin{bmatrix} \mathbb{W}_1^B \\ \mathbb{W}_2^B \\ \vdots \\ \mathbb{W}_m^B \\ \hline \mathbb{W}_m^A \\ \hline \mathbb{W}_1^A \\ \mathbb{W}_2^A \\ \vdots \\ \mathbb{W}_{m-1}^A \end{bmatrix} = -\mathsf{Y}\Lambda = - \begin{bmatrix} \mathbb{Y}_1^B & & & \\ \mathbb{Y}_2^B & \mathbb{Y}_1^B & & \\ \vdots & \vdots & \ddots & \\ \mathbb{Y}_m^B & \mathbb{Y}_{m-1}^B & \cdots & \mathbb{Y}_1^B \\ \hline & & & \mathbb{Y}_m^A \\ \hline & & & \mathbb{Y}_1^A \\ & & & \mathbb{Y}_2^A \\ & & & \vdots \\ & & & \mathbb{Y}_{m-1}^A \end{bmatrix} \begin{bmatrix} \boldsymbol{\Lambda}_1 \\ \boldsymbol{\Lambda}_2 \\ \vdots \\ \boldsymbol{\Lambda}_{m-1} \\ \hline \boldsymbol{\Lambda}_m \end{bmatrix} \tag{4.75}$$

where

$$\begin{aligned} \mathbb{W}_j^B &= \sum_{i=1}^{j} \mathbb{Y}_{j-k+1}^B \boldsymbol{\Lambda}_k \quad \forall j \in [1, m] \\ \mathbb{W}_m^A &= \mathbb{Y}_m^A \boldsymbol{\Lambda}_m \\ \mathbb{W}_j^A &= \mathbb{Y}_j^A \boldsymbol{\Lambda}_m \quad \forall j \in [1, (m-1)] \end{aligned} \tag{4.76}$$

4.4.2 Interface Decoupling and Physical Interpretation

The interface matrix (4.73) takes a complicated form since the intermediate time steps are coupled at the interface. However, it is possible to decouple the interface matrix for a simpler implementation using the final result in equation (4.74) as follows.

Henceforth we will follow the notation of Section 4.1 and the quantities defined in expressions (4.6) and (4.7) will be associated with a particular subdomain denoted by its superscript. The fully discretized equations for subdomain A at time t_m can be written as:

$$\mathbb{M}^A \mathbb{U}_m^A + \mathbb{C}^A \boldsymbol{\Lambda}_m = \mathbb{P}_m^A - \mathbb{N}^A \mathbb{U}_0^A \tag{4.77}$$

where $\mathbb{C}^A$ can be obtained from the definition:

$$\mathbb{C}^k = \begin{bmatrix} \boldsymbol{C}^{k^T} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \tag{4.78}$$

Similarly, the fully discretized equations for subdomain B at time t_j can be written as:

$$\mathbb{M}^B \mathbb{U}_j^B + \mathbb{C}^B \boldsymbol{\Lambda}_j = \mathbb{P}_j^B - \mathbb{N}^B \mathbb{U}_{j-1}^B \qquad \forall j \in [1, 2, \cdots, m] \tag{4.79}$$

The equation of continuity of velocities is written only at the final time step m as:

$$\boldsymbol{C}^A \dot{\boldsymbol{U}}_m^A + \boldsymbol{C}^B \dot{\boldsymbol{U}}_m^B = \mathbf{0} \tag{4.80}$$

This allows each subdomain to be integrated accurately with its own time step rather than constraining the velocity from subdomain B to match the linearly interpolated velocity from subdomain A at the intermediate steps as done by Gravouil and Combescure [93].

In order to solve the system of equations (4.77), (4.79) and (4.80) we split the kinematic quantities from subdomain A into two parts:

$$\mathbb{U}_m^A = \mathbb{V}_m^A + \mathbb{W}_m^A \tag{4.81}$$

where $\mathbb{V}_m^A = [\ddot{\boldsymbol{V}}_m^A, \dot{\boldsymbol{V}}_m^A, \boldsymbol{V}_m^A]^T$ and $\mathbb{W}_m^A = [\ddot{\boldsymbol{W}}_m^A, \dot{\boldsymbol{W}}_m^A, \boldsymbol{W}_m^A]^T$. The quantities in $\mathbb{V}_m^A$ are computed from external forces only (*free* problem):

$$\mathbb{M}^A \mathbb{V}_m^A = \mathbb{P}_m^A - \mathbb{N}^A \mathbb{U}_0^A \tag{4.82}$$

and the quantities in $\mathbb{W}_m^A$ are computed from interface reactions only (*link* problem):

$$\mathbb{M}^A \mathbb{W}_m^A = -\mathbb{C}^A \mathbf{\Lambda}_m \tag{4.83}$$

Note that by summing the above contributions, the original equation (4.77) for subdomain A is recovered:

$$\mathbb{M}^A \left(\mathbb{V}_m^A + \mathbb{W}_m^A \right) = \mathbb{M}^A \mathbb{U}_m^A = \mathbb{P}_m^A - \mathbb{N}^A \mathbb{U}_0^A - \mathbb{C}^A \mathbf{\Lambda}_m \tag{4.84}$$

Once the *free* problem given by eqn. (4.82) has been solved, one can compute the state variables for subdomain A at the intermediate time steps t_j by linearly interpolating between t_0 and t_m as:

$$\mathbb{V}_j^A = \left(1 - \frac{j}{m}\right) \mathbb{U}_0^A + \left(\frac{j}{m}\right) \mathbb{V}_m^A \tag{4.85}$$

Similarly, one can linearly interpolate for $\mathbb{W}_j^A$ between $\mathbf{0}$ and $\mathbb{W}_m^A$:

$$\mathbb{W}_j^A = \left(\frac{j}{m}\right) \mathbb{W}_m^A \tag{4.86}$$

Note again that by summing the above two contributions we obtain the state for subdomain A at t_j as linearly interpolated between t_0 and t_m:

$$\mathbb{V}_j^A + \mathbb{W}_j^A = \mathbb{U}_j^A = \left(1 - \frac{j}{m}\right) \mathbb{U}_0^A + \left(\frac{j}{m}\right) \mathbb{U}_m^A \tag{4.87}$$

Knowing the *free* state at t_j one can compute the *unbalanced free* interface reaction at t_j defined as:

$$\boldsymbol{S}_j \equiv \boldsymbol{C}^A \left(\boldsymbol{P}_j^A - \boldsymbol{M}^A \ddot{\boldsymbol{V}}_j^A - \boldsymbol{K}^A \boldsymbol{V}_j^A \right) \qquad \forall j \in [1, 2, \cdots, m] \tag{4.88}$$

which is the amount by which subdomain A is out of force equilibrium at t_j under external

forces only. Note that $\boldsymbol{S}_m = \mathbf{0}$ from (4.82). Substituting expression (4.85) above we get:

$$\begin{aligned}\boldsymbol{S}_j = & \\ & \boldsymbol{C}^A \left[\left(\frac{j}{m} \right) \left(\boldsymbol{P}_m^A - \boldsymbol{M}^A \ddot{\boldsymbol{V}}_m^A - \boldsymbol{K}^A \boldsymbol{V}_m^A \right) + \left(1 - \frac{j}{m} \right) \left(\boldsymbol{P}_0^A - \boldsymbol{M}^A \ddot{\boldsymbol{U}}_0^A - \boldsymbol{K}^A \boldsymbol{U}_0^A \right) \right] \\ & + \boldsymbol{C}^A \left[\boldsymbol{P}_j^A - \left(1 - \frac{j}{m} \right) \boldsymbol{P}_0^A - \left(\frac{j}{m} \right) \boldsymbol{P}_m^A \right]\end{aligned} \tag{4.89}$$

Since $\boldsymbol{P}_0^A - \boldsymbol{M}^A \ddot{\boldsymbol{U}}_0^A - \boldsymbol{K}^A \boldsymbol{U}_0^A = \boldsymbol{C}^{A^T} \boldsymbol{\Lambda_0}$ this simplifies to:

$$\boldsymbol{S}_j = \left(1 - \frac{j}{m} \right) \boldsymbol{\Lambda_0} + \boldsymbol{C}^A \left[\boldsymbol{P}_j^A - \left(1 - \frac{j}{m} \right) \boldsymbol{P}_0^A - \left(\frac{j}{m} \right) \boldsymbol{P}_m^A \right] \tag{4.90}$$

Similar to the *free* problem, we can define the *unbalanced link* interface reaction as:

$$\boldsymbol{T}_j \equiv -\boldsymbol{C}^A \left(\boldsymbol{M}^A \ddot{\boldsymbol{W}}_j^A + \boldsymbol{K}^A \boldsymbol{W}_j^A \right) - \boldsymbol{\Lambda}_j \qquad \forall j \in [1, 2, \cdots, m] \tag{4.91}$$

which is the amount by which subdomain A is out of force equilibrium at t_j under interface reactions only. Note that for equilibrium at t_j we must have $\boldsymbol{S}_j + \boldsymbol{T}_j = \mathbf{0}$:

$$\boldsymbol{S}_j + \boldsymbol{T}_j = \boldsymbol{C}^A \left(\boldsymbol{P}_j^A - \boldsymbol{M}^A \ddot{\boldsymbol{U}}_j^A - \boldsymbol{K}^A \boldsymbol{U}_j^A \right) - \boldsymbol{\Lambda}_j = \mathbf{0} \qquad \forall j \in [1, 2, \cdots, m] \tag{4.92}$$

which is obtained by premultiplying the equilibrium equation for subdomain A at t_j by $\boldsymbol{C}^A$. Here we use the identity:

$$\boldsymbol{C}^k \boldsymbol{C}^{k^T} = \boldsymbol{I}^\Lambda \tag{4.93}$$

where $\boldsymbol{I}^\Lambda$ is an identity matrix of dimension L, the size of the interface. Substituting equation (4.86) in (4.91), we obtain:

$$\boldsymbol{T}_j = -\left(\frac{j}{m} \right) \boldsymbol{C}^A \left(\boldsymbol{M}^A \ddot{\boldsymbol{W}}_m^A + \boldsymbol{K}^A \boldsymbol{W}_m^A \right) - \boldsymbol{\Lambda}_j \qquad \forall j \in [1, 2, \cdots, m] \tag{4.94}$$

and from (4.83), we obtain:

$$\boldsymbol{M}^A \ddot{\boldsymbol{W}}_m^A + \boldsymbol{K}^A \boldsymbol{W}_m^A = -\boldsymbol{C}^{A^T} \boldsymbol{\Lambda}_m \tag{4.95}$$

Substituting (4.95) in (4.94), we get:

$$\boldsymbol{T}_j = \left(\frac{j}{m}\right) \boldsymbol{\Lambda}_m - \boldsymbol{\Lambda}_j \qquad \forall j \in [1, 2, \cdots, m] \tag{4.96}$$

Using $\boldsymbol{S}_j = -\boldsymbol{T}_j$, one can compute $\boldsymbol{\Lambda}_j$ as:

$$\boldsymbol{\Lambda}_j = \boldsymbol{S}_j + \left(\frac{j}{m}\right) \boldsymbol{\Lambda}_m \qquad \forall j \in [1, 2, \cdots, m] \tag{4.97}$$

Substituting expression (4.90) above one obtains:

$$\boldsymbol{\Lambda}_j = \left[\left(1 - \frac{j}{m}\right) \boldsymbol{\Lambda_0} + \left(\frac{j}{m}\right) \boldsymbol{\Lambda_m}\right] + \boldsymbol{C}^A \left[\boldsymbol{P}_j^A - \left(1 - \frac{j}{m}\right) \boldsymbol{P}_0^A - \left(\frac{j}{m}\right) \boldsymbol{P}_m^A\right] \tag{4.98}$$

Using (4.97), we can now write the equation of motion for subdomain B (4.79) as:

$$\mathbb{M}^B \mathbb{U}_j^B + \left(\frac{j}{m}\right) \mathbb{C}^B \boldsymbol{\Lambda}_m = \mathbb{P}_j^B - \mathbb{N}^B \mathbb{U}_{j-1}^B - \mathbb{C}^B \boldsymbol{S}_j \qquad \forall j \in [1, 2, \cdots, m] \tag{4.99}$$

Following the notation of Section 4.1, the above equation can be written in matrix form as:

$$\left[\begin{array}{ccccc|c} \mathbb{M}^B & & & & & \frac{1}{m}\mathbb{C}^B \\ \mathbb{N}^B & \mathbb{M}^B & & & & \frac{2}{m}\mathbb{C}^B \\ & \ddots & \ddots & & & \vdots \\ & & \mathbb{N}^B & \mathbb{M}^B & & \mathbb{C}^B \\ \hline & & & & & \end{array}\right] \left[\begin{array}{c} \mathbb{U}_1^B \\ \mathbb{U}_2^B \\ \vdots \\ \mathbb{U}_m^B \\ \hline \boldsymbol{\Lambda}_m \end{array}\right] = \left[\begin{array}{c} \mathbb{P}_1^B - \mathbb{N}^B\ \mathbb{U}_0^B - \mathbb{C}^B \boldsymbol{S}_1 \\ \mathbb{P}_2^B - \mathbb{C}^B \boldsymbol{S}_2 \\ \vdots \\ \mathbb{P}_m^B - \mathbb{C}^B \boldsymbol{S}_m \end{array}\right] \tag{4.100}$$

We can write the final system of equations to be solved by appending equations (4.77) and

(4.80) to the above as:

$$\left[\begin{array}{ccc|c|c} \mathbb{M}^B & & & & \frac{1}{m}\mathbb{C}^B \\ \mathbb{N}^B & \mathbb{M}^B & & & \frac{2}{m}\mathbb{C}^B \\ & \ddots & \ddots & & \vdots \\ & & \mathbb{N}^B \;\; \mathbb{M}^B & & \mathbb{C}^B \\ \hline & & & \mathbb{M}^A & \mathbb{C}^A \\ \hline\hline & & \mathbb{B}^B & \mathbb{B}^A & \mathbf{0} \end{array}\right] \left[\begin{array}{c} \mathbb{U}_1^B \\ \mathbb{U}_2^B \\ \vdots \\ \mathbb{U}_m^B \\ \hline \mathbb{U}_m^A \\ \hline\hline \boldsymbol{\Lambda}_m \end{array}\right] = \left[\begin{array}{c} \mathbb{P}_1^B - \mathbb{N}^B\,\mathbb{U}_0^B - \mathbb{C}^B\boldsymbol{S}_1 \\ \mathbb{P}_2^B - \mathbb{C}^B\boldsymbol{S}_2 \\ \vdots \\ \mathbb{P}_m^B - \mathbb{C}^B\boldsymbol{S}_m \\ \hline \mathbb{P}_m^A - \mathbb{N}^A\,\mathbb{U}_0^A \\ \hline\hline \mathbf{0} \end{array}\right] \tag{4.101}$$

where $\mathbb{B}^k = \left[\begin{array}{ccc} \mathbf{0} & \boldsymbol{C}^k & \mathbf{0} \end{array}\right]$. Note that the first block represents the system of equations for subdomain B, the second block for subdomain A and the last row imposes the continuity of velocity constraint at the final time step.

4.4.3 Solution Procedure

Equation (4.101) can be solved using a bordered system approach. The system is divided along the bold lines shown in eqn. (4.101) as:

$$\left[\begin{array}{cc} \mathsf{M} & \mathsf{C} \\ \mathsf{B} & 0 \end{array}\right] \left[\begin{array}{c} \mathsf{U} \\ \boldsymbol{\Lambda}_m \end{array}\right] = \left[\begin{array}{c} \mathsf{P} \\ 0 \end{array}\right] \tag{4.102}$$

Let

$$\mathsf{U} = \mathsf{V} + \mathsf{W} \quad \text{where} \quad \mathsf{V} = \mathsf{M}^{-1}\mathsf{P} \quad ; \quad \mathsf{W} = -\mathsf{Y}\boldsymbol{\Lambda}_m \quad \text{and} \quad \mathsf{Y} = \mathsf{M}^{-1}\mathsf{C} \tag{4.103}$$

Note that the first row in equation (4.102) is satisfied automatically, which can be demonstrated as follows:

$$\mathsf{MU} + \mathsf{C}\boldsymbol{\Lambda}_m = \mathsf{P} \quad \Rightarrow \mathsf{MV} + \mathsf{MW} + \mathsf{C}\boldsymbol{\Lambda}_m = \mathsf{P} \quad \Rightarrow (\mathsf{MV} - \mathsf{P}) = (\mathsf{MY} - \mathsf{C})\boldsymbol{\Lambda}_m \tag{4.104}$$

The second row is used for computing the interface reactions:

$$\mathsf{BU} = 0 \quad \Rightarrow \mathsf{BV} + \mathsf{BW} = 0 \quad \Rightarrow [\mathsf{BY}]\boldsymbol{\Lambda}_m = \mathsf{BV} \tag{4.105}$$

Thus, to solve the system, one must first compute V from eqn. (4.103) then solve for $\boldsymbol{\Lambda}_m$ from eqn. (4.105) and finally update the solution with W from eqn. (4.103).

Computing V.

Let us first compute V from the equation $\mathsf{MV} = \mathsf{P}$, which can be written as:

$$\left[\begin{array}{ccccc|c} \mathbb{M}^B & & & & & \\ \mathbb{N}^B & \mathbb{M}^B & & & & \\ & \ddots & \ddots & & & \\ & & \mathbb{N}^B & \mathbb{M}^B & & \\ \hline & & & & & \mathbb{M}^A \end{array}\right] \left[\begin{array}{c} \mathbb{V}_1^B \\ \mathbb{V}_2^B \\ \vdots \\ \mathbb{V}_m^B \\ \hline \mathbb{V}_m^A \end{array}\right] = \left[\begin{array}{c} \mathbb{P}_1^B - \mathbb{N}^B \boldsymbol{U}_0^B - \mathbb{C}^B \boldsymbol{S}_1 \\ \mathbb{P}_1^B - \mathbb{C}^B \boldsymbol{S}_2 \\ \vdots \\ \mathbb{P}_m^B - \mathbb{C}^B \boldsymbol{S}_m \\ \hline \mathbb{P}_m^A - \mathbb{N}^A \boldsymbol{U}_0^A \end{array}\right] \tag{4.106}$$

This system can be solved by first advancing the solution for subdomain A by ΔT under external loads only (*free* problem). Then, the unbalanced *free* interface reactions $\boldsymbol{S}_j$ can be computed at the intermediate time steps from eqn. (4.88) using the solution from subdomain A. Finally, one can solve the *free* problem in subdomain B by time stepping it m times through steps of Δt each under external forces and $\boldsymbol{S}_j$, j varying from 1 to m.

Computing Y.

Next we compute Y from the equation $\mathsf{MY} = \mathsf{C}$, which has the form:

$$\left[\begin{array}{ccccc|c} \mathbb{M}^B & & & & & \\ \mathbb{N}^B & \mathbb{M}^B & & & & \\ & \ddots & \ddots & & & \\ & & \mathbb{N}^B & \mathbb{M}^B & & \\ \hline & & & & & \mathbb{M}^A \end{array}\right] \left[\begin{array}{c} \mathbb{Y}_1^B \\ \mathbb{Y}_2^B \\ \vdots \\ \mathbb{Y}_m^B \\ \hline \mathbb{Y}_m^A \end{array}\right] = \left[\begin{array}{c} \frac{1}{m}\mathbb{C}^B \\ \frac{2}{m}\mathbb{C}^B \\ \vdots \\ \mathbb{C}^B \\ \hline \mathbb{C}^A \end{array}\right] \tag{4.107}$$

This system can be solved for $\mathbb{Y}_m^A$ by applying a unit load to one shared degree of freedom at a time and advancing by a single time step ΔT from complete zero initial conditions. Similarly, for subdomain B, $\mathbb{Y}_j^B$s can be computed by applying a linearly varying load (ramped up to unity in m time steps of Δt) at the shared degrees of freedom, one at a time.

Computing interface reactions $\mathbf{\Lambda}_m$.

The interface system matrix $[\mathsf{BY}]$ in eqn. (4.105) can be computed as:

$$\left[\begin{array}{ccc|c} 0 & \cdots & 0 & \mathbb{B}^B \end{array}\right] \left[\begin{array}{c|c} & \mathbb{B}^A \end{array}\right] \left[\begin{array}{c} \mathbb{Y}_1^B \\ \mathbb{Y}_2^B \\ \vdots \\ \mathbb{Y}_m^B \\ \hline \mathbb{Y}_m^A \end{array}\right] = \mathbb{B}^B\mathbb{Y}_m^B + \mathbb{B}^A\mathbb{Y}_m^A = \boldsymbol{C}^A\dot{\boldsymbol{Y}}_m^A + \boldsymbol{C}^B\dot{\boldsymbol{Y}}_m^B \tag{4.108}$$

Note that this system matrix is exactly the same as that obtained in the GC method for the case $\Delta T = \Delta t$. For multi-time-step cases, the contribution of subdomain A is the same but subdomain B is advanced further in time. The right hand side vector $\{\mathsf{BV}\}$ for the interface

equation (4.105) is also computed similarly:

$$\left[\begin{array}{cccc|c} 0 & \cdots & 0 & \mathbb{B}^B & \mathbb{B}^A \end{array}\right] \left[\begin{array}{c} \mathbb{V}_1^B \\ \mathbb{V}_2^B \\ \vdots \\ \mathbb{V}_m^B \\ \hline \mathbb{V}_m^A \end{array}\right] = \mathbb{B}^B\mathbb{V}_m^B + \mathbb{B}^A\mathbb{V}_m^A = \boldsymbol{C}^A\dot{\boldsymbol{V}}_m^A + \boldsymbol{C}^B\dot{\boldsymbol{V}}_m^B \tag{4.109}$$

Thus the final interface equation to be solved takes the form:

$$[\boldsymbol{H}]\ \{\boldsymbol{\Lambda}_m\} = \{\boldsymbol{C}^A\dot{\boldsymbol{V}}_m^A + \boldsymbol{C}^B\dot{\boldsymbol{V}}_m^B\} \tag{4.110}$$

where $\boldsymbol{H} = \boldsymbol{C}^A\dot{\boldsymbol{Y}}_m^A + \boldsymbol{C}^B\dot{\boldsymbol{Y}}_m^B$. Having computed the correct interface reaction forces from eqn. (4.110), one can now update the *free* solution computed previously.

Computing W.

Finally we can evaluate W as:

$$\mathsf{W} = -\mathsf{Y}\boldsymbol{\Lambda}_m = -\left[\begin{array}{c} \mathbb{Y}_1^B \\ \mathbb{Y}_2^B \\ \vdots \\ \mathbb{Y}_m^B \\ \hline \mathbb{Y}_m^A \end{array}\right] \boldsymbol{\Lambda}_m \tag{4.111}$$

Using this W the solution is updated as $\mathsf{U} = \mathsf{V} + \mathsf{W}$ and this process is repeated over for the next cycle of time step ΔT.

One may choose to update only the final solution and move to the next time step. Updating the intermediate solutions does not have any effect on the coupling method and is necessary only for post-processing.

4.4.4 Final Coupling Algorithm

The final multi-time-step coupling algorithm, shown in Figure 4.7, can be presented in three stages:

Predictor Stage.

Solve the *free* problem in subdomain A by timestepping once through ΔT under external loads only (Equations (4.82),(4.106)). Solve the *free* problem in subdomain B by timestepping m times through time steps of Δt each under external loads and unbalanced free interface reactions from subdomain A (Equation (4.106)).

Interface Solve.

Solve for the interface reaction $\mathbf{\Lambda}_m$ at the final time step from equation (4.110). Compute the interface reaction $\mathbf{\Lambda}_j$ at the intermediate time steps using eqn. (4.97).

Corrector Stage.

Update the solution for subdomain A using eqn. (4.83) and for subdomain B using eqn. (4.111)

Note that, unlike the GC method, one does not need to solve for interface reactions at every intermediate time step and this leads to much greater computational efficiency. The present multi-time-step coupling method can be extended for multiple subdomain cases also following the derivation presented above.

4.4.5 Starting Procedure

Although one does not need any special starting procedure for the present algorithm, it must be mentioned that the initial acceleration calculations for the Newmark time stepping

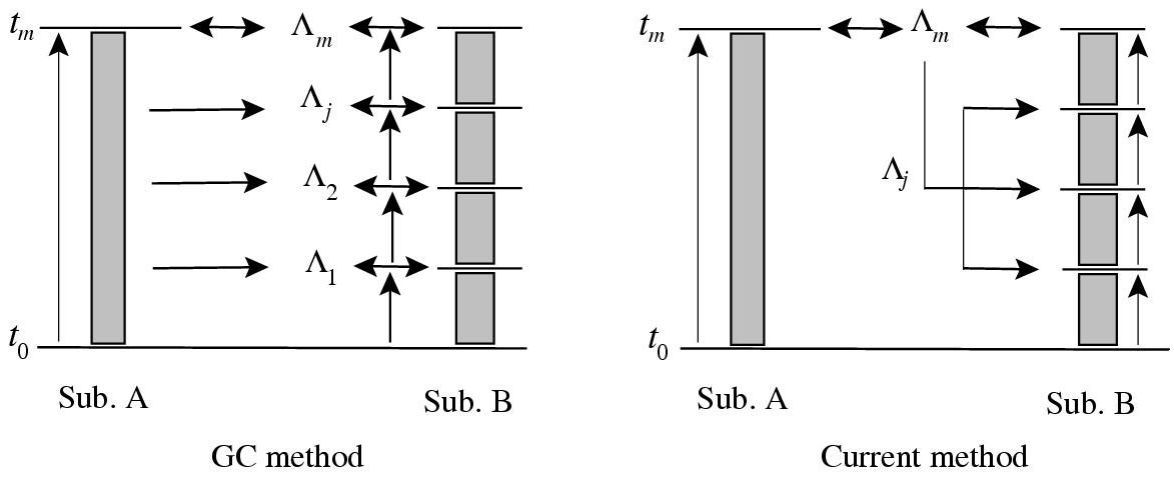

Figure 4.7: Comparison of the GC method with the current multi-time-step coupling algorithm.

schemes must be done on the *undecomposed* system as:

$$\ddot{\boldsymbol{U}}_0 = \boldsymbol{M}^{-1}\left(\boldsymbol{P}_0 - \boldsymbol{K}\boldsymbol{U}_0\right) \tag{4.112}$$

Once the state variables for all the degrees of freedom in the original structure have been computed, one can partition the mesh accordingly and start the multi-time-step coupling algorithm as detailed above.

4.5 Stability Analysis

The stability analysis is carried out using the energy method (see Chapter 9, [11]). We will prove that the change in energy due to our coupling algorithm over the time step t_0 to t_m is identically *zero*. First, we define the undivided forward difference and average operators as:

$$\begin{aligned} [\boldsymbol{x}_{j-1}] &\equiv \boldsymbol{x}_j - \boldsymbol{x}_{j-1} \qquad & \langle \boldsymbol{x}_{j-1} \rangle &\equiv \tfrac{1}{2}\left(\boldsymbol{x}_j + \boldsymbol{x}_{j-1}\right) \\ [\![\boldsymbol{x}_0]\!] &\equiv \boldsymbol{x}_m - \boldsymbol{x}_0 \qquad & \langle\!\langle \boldsymbol{x}_0 \rangle\!\rangle &\equiv \tfrac{1}{2}\left(\boldsymbol{x}_m + \boldsymbol{x}_0\right) \end{aligned} \tag{4.113}$$

for Δt and ΔT time steps respectively. Following the derivation presented in [11], we can define the change in energy for subdomains A and B over timesteps ΔT and Δt, respectively, as:

$$\mathscr{E}_0^A \equiv [\![T^A(\ddot{\boldsymbol{U}}_0^A)]\!] + [\![U^A(\dot{\boldsymbol{U}}_0^A)]\!] = -D([\![\ddot{\boldsymbol{U}}_0^A]\!]) + E_\Lambda^A([\![\dot{\boldsymbol{U}}_0^A]\!], [\![\boldsymbol{\Lambda}_0]\!]) \tag{4.114}$$

and

$$\mathscr{E}_{j-1}^B \equiv [T^B(\ddot{\boldsymbol{U}}_{j-1}^B)] + [U^B(\dot{\boldsymbol{U}}_{j-1}^B)] = -D([\ddot{\boldsymbol{U}}_{j-1}^B]) + E_\Lambda^B([\dot{\boldsymbol{U}}_{j-1}^B], [\boldsymbol{\Lambda}_{j-1}]) \tag{4.115}$$

where

$$\begin{aligned} T^k(\boldsymbol{x}) &\equiv \tfrac{1}{2}\boldsymbol{x}^T\boldsymbol{A}^k\boldsymbol{x} \\ U^k(\boldsymbol{x}) &\equiv \tfrac{1}{2}\boldsymbol{x}^T\boldsymbol{K}^k\boldsymbol{x} \\ D^k(\boldsymbol{x}) &\equiv (\gamma_k - \tfrac{1}{2})\boldsymbol{x}^T\boldsymbol{A}^k\boldsymbol{x} \\ E_\Lambda^k(\boldsymbol{x}, \boldsymbol{\lambda}) &\equiv \frac{1}{\Delta t_k}\boldsymbol{x}^T\boldsymbol{C}^{k^T}\boldsymbol{\lambda} \end{aligned} \tag{4.116}$$

and

$$\boldsymbol{A}^k \equiv \boldsymbol{M}^k + \Delta t_k^2\left(\beta_k - \frac{\gamma_k}{2}\right)\boldsymbol{K}^k \tag{4.117}$$

The change in energy for the total system from t_0 to t_m can then be computed as:

$$\mathscr{E} = \mathscr{E}_0^A + \sum_{j=1}^{m}\mathscr{E}_{j-1}^B = -D([\![\ddot{\boldsymbol{U}}_0^A]\!]) - \sum_{j=1}^{m}D([\ddot{\boldsymbol{U}}_{j-1}^B]) + E_\Lambda \tag{4.118}$$

where

$$E_\Lambda = E_\Lambda^A([\![\dot{\boldsymbol{U}}_0^A]\!], [\![\boldsymbol{\Lambda}_0]\!]) + \sum_{j=1}^{m}E_\Lambda^B([\dot{\boldsymbol{U}}_{j-1}^B], [\boldsymbol{\Lambda}_{j-1}]) \tag{4.119}$$

A method is considered to be numerically stable if the total energy change $\mathscr{E}$ between time steps t_0 and t_m under zero *external* loads is less than or equal to zero.

In our case, it suffices to show that for $\mathscr{E} \leq 0$ the energy change due to the coupling at

the interface E_Λ must be less than or equal to zero. It is evident from equation (4.118) that the change in energy $\mathscr{E}$, would then become a sum of negative terms. Using eqn. (4.97), we can write:

$$[\boldsymbol{\Lambda}_j] = \boldsymbol{\Lambda}_j - \boldsymbol{\Lambda}_{j-1} = (\boldsymbol{S}_j - \boldsymbol{S}_{j-1}) + \left(\frac{1}{m}\right)\boldsymbol{\Lambda}_m \tag{4.120}$$

Thus, from (4.119) we get:

$$\begin{aligned} E_\Lambda =& \frac{1}{m\Delta t}\left(\dot{\boldsymbol{U}}_m^A - \dot{\boldsymbol{U}}_0^A\right)^T \boldsymbol{C}^{A^T}\left(\boldsymbol{\Lambda}_m - \boldsymbol{\Lambda}_0\right) + \frac{1}{m\Delta t}\left(\sum_{j=0}^{m-1}[\dot{\boldsymbol{U}}_j^B]^T\right)\boldsymbol{C}^{B^T}\boldsymbol{\Lambda}_m \\ &+ \frac{1}{\Delta t}\sum_{j=0}^{m-1}\left([\dot{\boldsymbol{U}}_j^B]^T\boldsymbol{C}^{B^T}[\boldsymbol{S}_j]\right) \end{aligned} \tag{4.121}$$

Note that:

$$\sum_{j=0}^{m-1}[\dot{\boldsymbol{U}}_j^B] = [\![\dot{\boldsymbol{U}}_0^B]\!] = \dot{\boldsymbol{U}}_m^B - \dot{\boldsymbol{U}}_0^B \tag{4.122}$$

Thus, E_Λ can be further simplified to:

$$\begin{aligned} E_\Lambda =& \frac{1}{m\Delta t}\left(\left(\boldsymbol{C}^A\dot{\boldsymbol{U}}_m^A + \boldsymbol{C}^B\dot{\boldsymbol{U}}_m^B\right) - \left(\boldsymbol{C}^A\dot{\boldsymbol{U}}_0^A + \boldsymbol{C}^B\dot{\boldsymbol{U}}_0^B\right)\right)\boldsymbol{\Lambda}_m \\ &- \frac{1}{m\Delta t}\left(\dot{\boldsymbol{U}}_m^A - \dot{\boldsymbol{U}}_0^A\right)^T\boldsymbol{C}^{A^T}\boldsymbol{\Lambda}_0 + \frac{1}{\Delta t}\sum_{j=0}^{m-1}\left([\dot{\boldsymbol{U}}_j^B]^T\boldsymbol{C}^{B^T}[\boldsymbol{S}_j]\right) \end{aligned} \tag{4.123}$$

The first term is zero because of the continuity of velocities constraint at t_0 and t_m. The second term can be simplified using equations (4.88) for $\boldsymbol{S}_j$ (noting that $\boldsymbol{P}_j^A = 0$) as:

$$[\boldsymbol{S}_j] = -\boldsymbol{C}^A\left(\boldsymbol{M}^A[\ddot{\boldsymbol{V}}_j^A] + \boldsymbol{K}^A[\boldsymbol{V}_j^A]\right) \tag{4.124}$$

Since $\ddot{\boldsymbol{V}}_j^A$ and $\boldsymbol{V}_j^A$ are interpolated linearly between t_0 and t_m, we have:

$$[\boldsymbol{V}_j^A] = \frac{1}{m}\left(\boldsymbol{V}_m^A - \boldsymbol{U}_0^A\right)\ ; \qquad [\ddot{\boldsymbol{V}}_j^A] = \frac{1}{m}\left(\ddot{\boldsymbol{V}}_m^A - \ddot{\boldsymbol{U}}_0^A\right) \tag{4.125}$$

Substituting (4.125) into (4.124), we get:

$$\begin{aligned}
[\boldsymbol{S}_j] =& -\boldsymbol{C}^A\left(\frac{1}{m}\boldsymbol{M}^A\left(\ddot{\boldsymbol{V}}_m^A-\ddot{\boldsymbol{U}}_0^A\right)+\frac{1}{m}\boldsymbol{K}^A\left(\boldsymbol{V}_m^A-\boldsymbol{U}_0^A\right)\right)\\
=& -\frac{1}{m}\boldsymbol{C}^A\left(\boldsymbol{M}^A\ddot{\boldsymbol{V}}_m^A+\boldsymbol{K}^A\boldsymbol{V}_m^A-\boldsymbol{M}^A\ddot{\boldsymbol{U}}_0^A-\boldsymbol{K}^A\boldsymbol{U}_0^A\right)\\
=&\frac{1}{m}\boldsymbol{C}^A\left(\boldsymbol{M}^A\ddot{\boldsymbol{U}}_0^A+\boldsymbol{K}^A\boldsymbol{U}_0^A\right)\\
=& -\frac{1}{m}\boldsymbol{C}^A\left(\boldsymbol{C}^{A^T}\boldsymbol{\Lambda}_0\right)=-\frac{1}{m}\boldsymbol{\Lambda}_0
\end{aligned} \tag{4.126}$$

since $\boldsymbol{S}_m=-\boldsymbol{M}^A\ddot{\boldsymbol{V}}_m^A-\boldsymbol{K}^A\boldsymbol{V}_m^A=\boldsymbol{0}$ and $\boldsymbol{C}^A\boldsymbol{C}^{A^T}=\boldsymbol{I}^\Lambda$.

Substituting (4.126) into (4.123) we get:

$$\begin{aligned}
E_\Lambda =&\frac{1}{m\Delta t}\left(\left(\boldsymbol{C}^A\dot{\boldsymbol{U}}_m^A+\boldsymbol{C}^B\dot{\boldsymbol{U}}_m^B\right)-\left(\boldsymbol{C}^A\dot{\boldsymbol{U}}_0^A+\boldsymbol{C}^B\dot{\boldsymbol{U}}_0^B\right)\right)\boldsymbol{\Lambda}_m\\
&-\frac{1}{m\Delta t}\left(\dot{\boldsymbol{U}}_m^A-\dot{\boldsymbol{U}}_0^A\right)^T\boldsymbol{C}^{A^T}\boldsymbol{\Lambda}_0-\frac{1}{m\Delta t}\left(\sum_{j=0}^{m-1}[\dot{\boldsymbol{U}}_j^B]^T\right)\boldsymbol{C}^{B^T}\boldsymbol{\Lambda}_0
\end{aligned} \tag{4.127}$$

Again, using (4.122), we get:

$$E_\Lambda=\frac{1}{m\Delta t}\left[\left(\boldsymbol{C}^A\dot{\boldsymbol{U}}_m^A+\boldsymbol{C}^B\dot{\boldsymbol{U}}_m^B\right)-\left(\boldsymbol{C}^A\dot{\boldsymbol{U}}_0^A+\boldsymbol{C}^B\dot{\boldsymbol{U}}_0^B\right)\right](\boldsymbol{\Lambda}_m-\boldsymbol{\Lambda}_0)=0 \tag{4.128}$$

This shows that the coupling algorithm neither adds nor removes energy from the coupled system, hence is unconditionally stable and energy preserving. Thus, one can couple arbitrary Newmark schemes with different time steps using the present method and expect the stability characteristics of individual subdomains to be preserved. We also note that since E_Λ is exactly zero the current method does not exhibit any dissipation unlike the GC method.

4.6 Numerical Results

Test problems that were solved to validate the present coupling method include a *split* single degree of freedom (SDOF) system, a 1-D fixed-free bar problem with a step end load and a 2-D cantilever beam problem.

4.6.1 The Split SDOF Problem

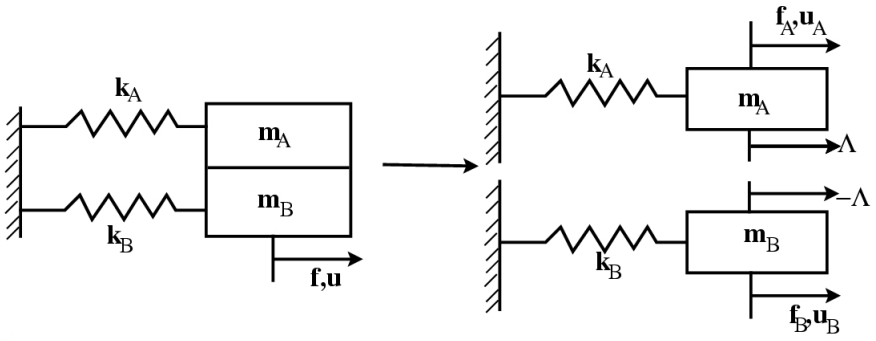

Figure 4.8: A split SDOF problem.

The simplest multi-time-step coupling problem is the split SDOF system. A SDOF mass and spring system is split into two SDOF mass and spring systems A and B linked by an interface reaction Λ as shown in Figure 4.8. The systems A and B can be integrated with different time steps and different Newmark schemes. In the present problem the following values for the system variables were chosen: $\mathsf{m}_a = 1.0 \times 10^{-6}$, $\mathsf{m}_b = 2.0 \times 10^{-6}$, $\mathsf{k}_a = 2.0 \times 10^4$ and $\mathsf{k}_b = 3.0 \times 10^4$. The system was excited by a step loading function going from 0 to $\mathsf{f}_a = 3.0$ and $\mathsf{f}_b = 1.0$ at $t = 0$.

Figures 4.9 and 4.10 show the results for a case of time step ratio $m = 2$ between A and B. The time steps chosen were $\Delta T = 8.0 \times 10^{-6}$ and $\Delta t = 4.0 \times 10^{-6}$ with Newmark parameters $(\gamma_A, \beta_A) = (\gamma_B, \beta_B) = (0.5, 0.25)$. Thus we couple the implicit constant average acceleration method with itself here. The critical time step (Δt_{cr}) for the central difference scheme in system B is 1.63×10^{-5}. Note that the time steps for systems A and B were

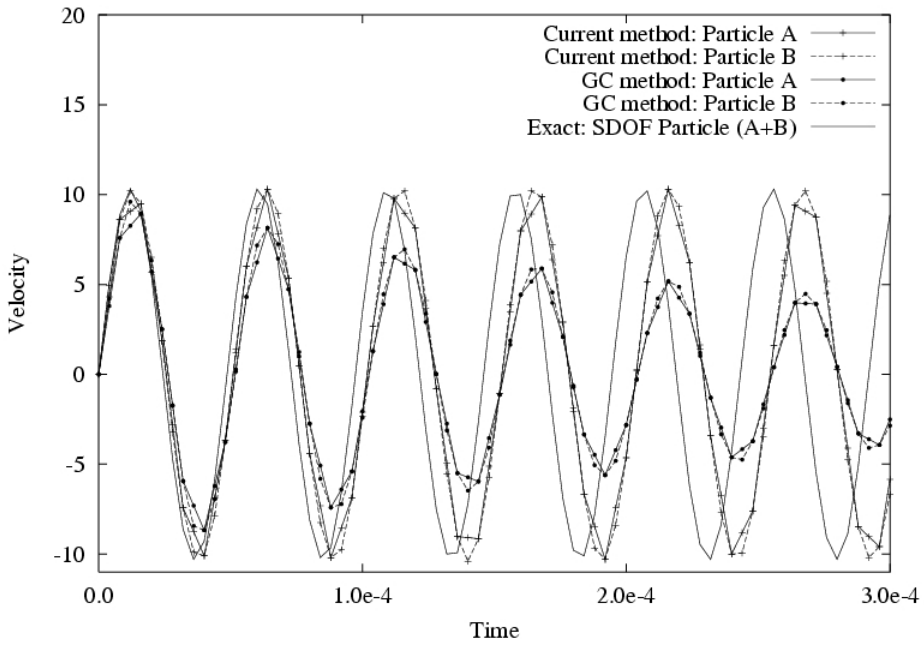

Figure 4.9: Split SDOF problem: Velocity response for $m = 2$; $\Delta T/\Delta t_{cr} = 0.5$; $\Delta t/\Delta t_{cr} = 0.25$.

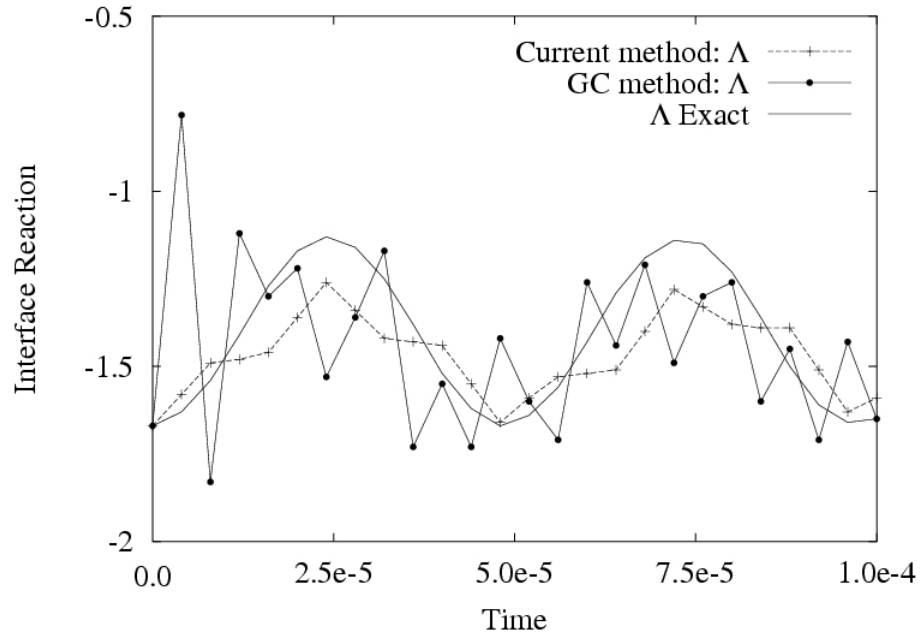

Figure 4.10: Split SDOF problem: Interface reactions for $m = 2$; $\Delta T/\Delta t_{cr} = 0.5$; $\Delta t/\Delta t_{cr} = 0.25$.

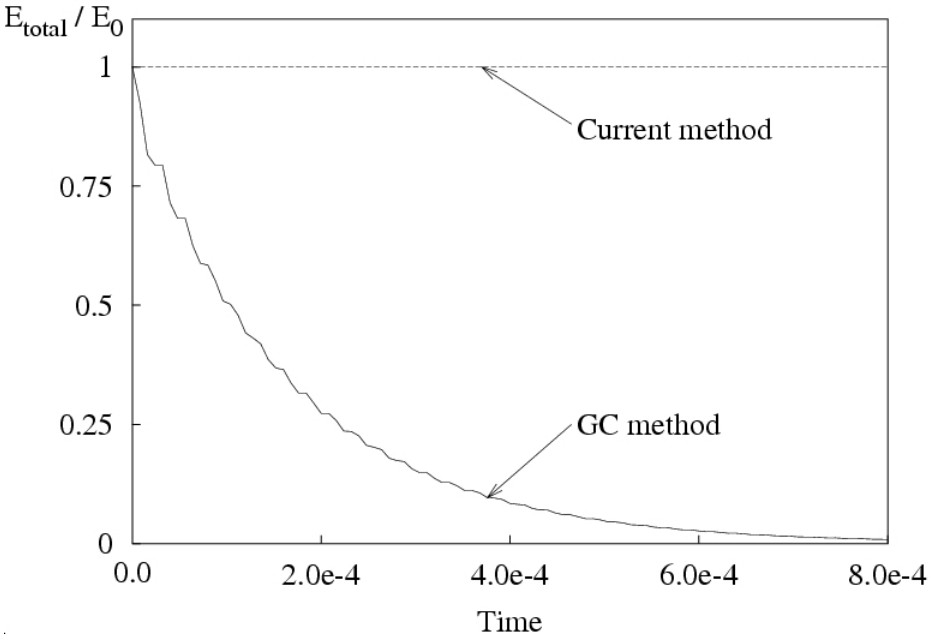

Figure 4.11: Split SDOF problem: Total energy for $m = 2$;
$\Delta T/\Delta t_{cr} = 0.5$; $\Delta t/\Delta t_{cr} = 0.25$; $\mathrm{E}_0 = 2.67 \times 10^6$.

chosen so coarse in order to distinguish the different curves. The velocity response in Figure 4.9 shows that there is significant amplitude decay in the GC method while the present method does not exhibit any dissipation. One may also note that the velocities from the two subdomains match exactly at intervals of ΔT with the computed response from subdomain A being linearly interpolated between the interval. Figure 4.10 compares the interface reactions computed by the present method with those from the GC method.

It is clear that the present method computes interface reactions that are much closer to their exact value whereas the interface reactions from the GC method show large oscillations around the exact value. A comparison of total energy defined using the norms given in equation (4.116) as

$$\mathrm{E}_{\mathrm{total}} \equiv \left(T^A(\ddot{\boldsymbol{U}}_n^A) + U^A(\dot{\boldsymbol{U}}_n^A)\right) + \left(T^B(\ddot{\boldsymbol{U}}_n^B) + U^B(\dot{\boldsymbol{U}}_n^B)\right) \tag{4.129}$$

is presented in Figure 4.11 for the above problem. This confirms the result from the stability

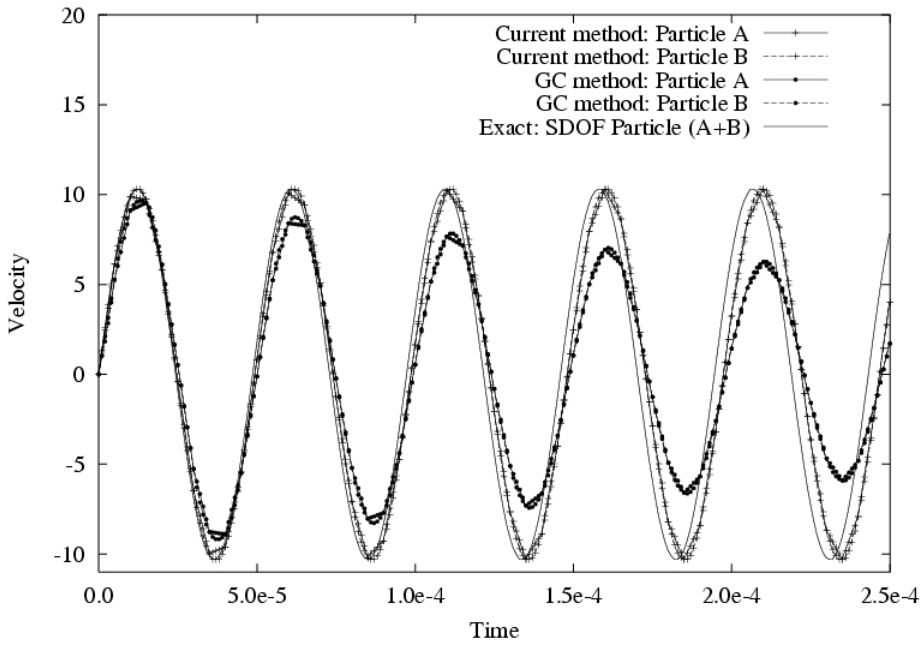

Figure 4.12: Split SDOF problem: Velocity response for $m = 5$; $\Delta T/\Delta t_{cr} = 0.3$; $\Delta t/\Delta t_{cr} = 0.06$.

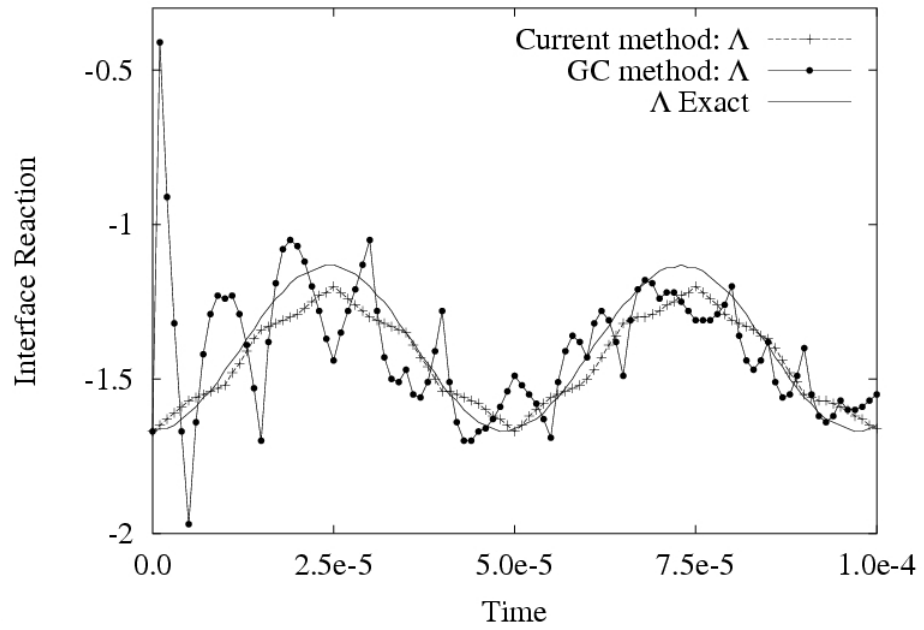

Figure 4.13: Split SDOF problem: Interface reactions for $m = 5$; $\Delta T/\Delta t_{cr} = 0.3$; $\Delta t/\Delta t_{cr} = 0.06$.

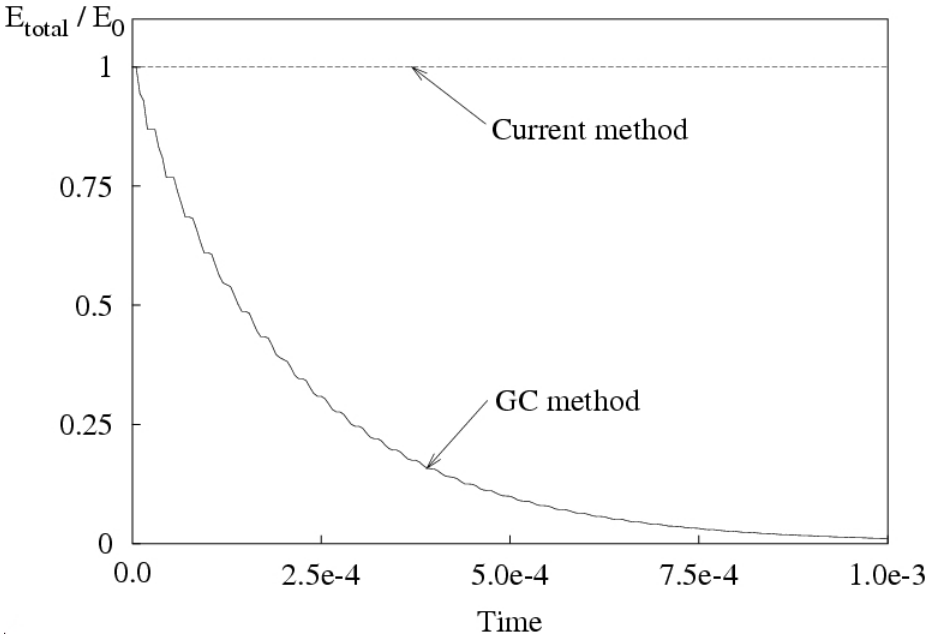

Figure 4.14: Split SDOF problem: Total energy for $m = 5$;
$\Delta T/\Delta t_{cr} = 0.3$; $\Delta t/\Delta t_{cr} = 0.06$; $E_0 = 2.67 \times 10^6$

analysis that the present coupling method exactly preserves the total energy of the system.

Similar results for a time step ratio $m = 5$ are presented in Figures 4.12 - 4.14. The time steps used here are $\Delta T = 5.0 \times 10^{-6}$ and $\Delta t = 1.0 \times 10^{-6}$. One can again see that the velocity response from the GC method decays and the interface tractions oscillate around the exact value while the present method performs much better. One may also note that subdomain B is integrated with a much finer time step here and shows smooth behavior at the crests of the waves while subdomain A with a coarse time step shows distinct steps in its response. This shows that using the present method one can integrate the two subdomains with the desired accuracy for each of them.

4.6.2 1-D Fixed-free Bar Problem

In this problem, a 1-D bar fixed at one end and free at the other was meshed uniformly with 2 node linear bar elements and partitioned at the mid point as shown in Figure 4.15.

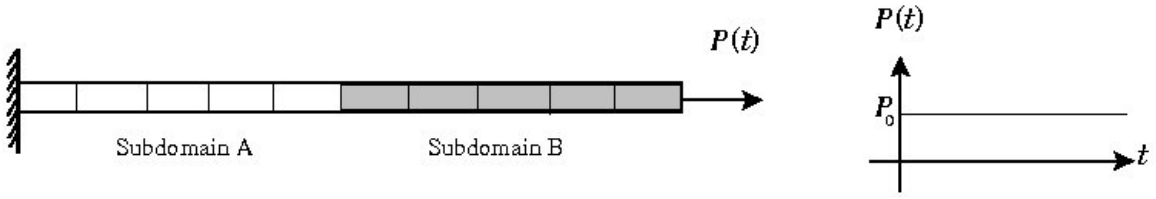

Figure 4.15: 1-D fixed free bar with a step end load.

The parameters chosen for the problem were: mass density $\rho = 1.0 \times 10^{-4}$, Young's modulus $E = 1.0 \times 10^4$, total length $L = 1.0$, sectional area $A = 0.2$, end load $\boldsymbol{P}_0 = 10.0$, $\Delta t_A = \Delta T = 2.0 \times 10^{-5}$, $\Delta t_B = \Delta t = 2.0 \times 10^{-6}$, $(\gamma_A, \beta_A) = (0.5, 0.25)$ and $(\gamma_B, \beta_B) = (0.5, 0.0)$. Thus, we choose to solve the above problem by treating half the mesh implicitly using the constant average acceleration method and half explicitly using the central difference method. Note that the critical time step for the central difference method Δt_{cr} is 1.0×10^{-5} and the time step ratio between the subdomains m is 10 with $\Delta T/\Delta t_{cr} = 2.0$ and $\Delta t/\Delta t_{cr} = 0.2$.

Figures 4.16 and 4.17 show the end displacement and interface reactions respectively

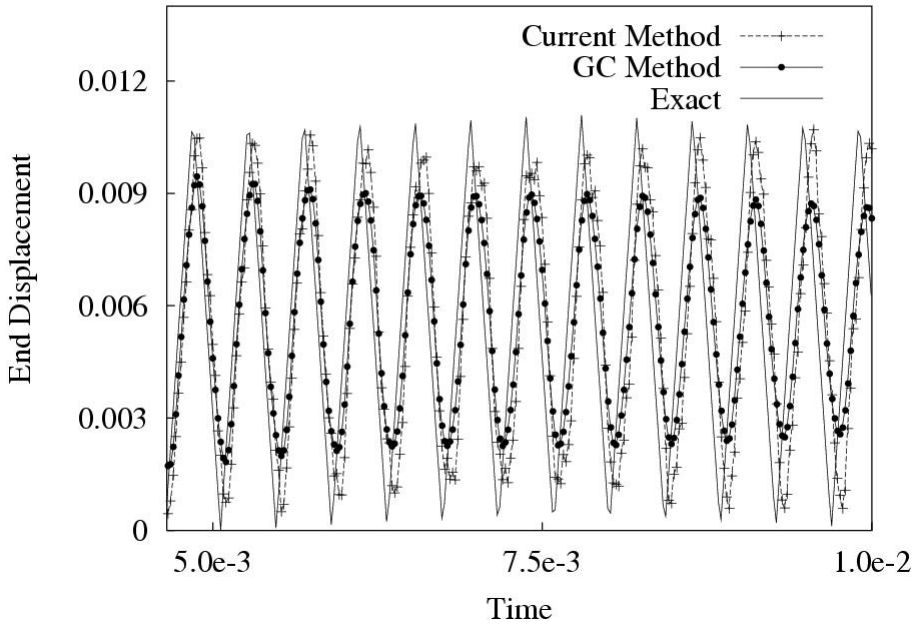

Figure 4.16: End displacement response for 1-D fixed-free bar under step load; $m = 10$

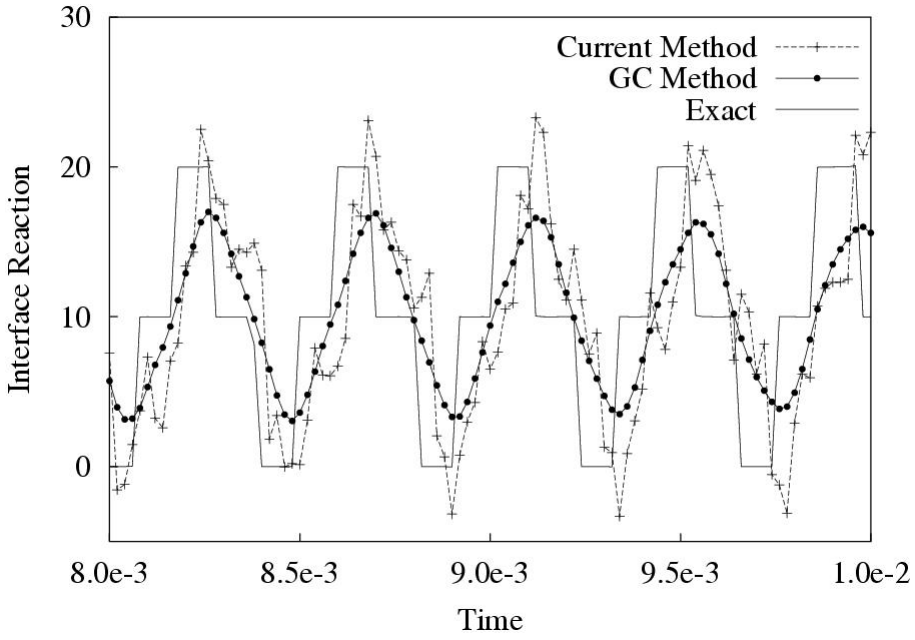

Figure 4.17: Interface reactions for 1-D fixed-free bar under step load; $m = 10$

for the fixed-free bar under a step end load. We have used coarse time steps and plotted the displacement response starting at $t = 0.00465$ to accentuate the dissipation due to the GC method. The dissipation is also visible in the interface reactions as its sharp features are smoothened out by the GC method while the present method preserves the average magnitude of the response. Note that the overshoot and undershoot around the exact value are obtained in the Newmark schemes too and are not an artifact of the present coupling method.

4.6.3 2-D Cantilever Beam Problem

In this section, we consider a 2-D cantilever beam problem with a step end load as shown in Figure 4.18. The beam is meshed uniformly with 4 node quadrilateral elements and partitioned as shown. Subdomain A is integrated with a time step $\Delta T = 8.0 \times 10^{-5}$ and Newmark parameters $(\gamma_A, \beta_A) = (0.5, 0.25)$ while subdomain B is integrated with $\Delta t = 8.0 \times 10^{-6}$ and $(\gamma_B, \beta_B) = (0.5, 0.25)$. Thus, we couple the constant average accelaration method

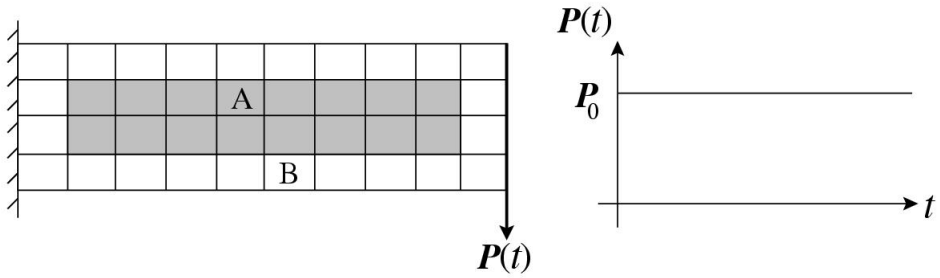

Figure 4.18: 2-D cantilever beam with a step end load.

with itself with a time step ratio $m = 10$. This particular partition was chosen so that the ratio of the number of degrees of freedom along the interface to the total number of degrees of freedom in the problem be sufficiently large so as to bring out the dissipation exhibited by

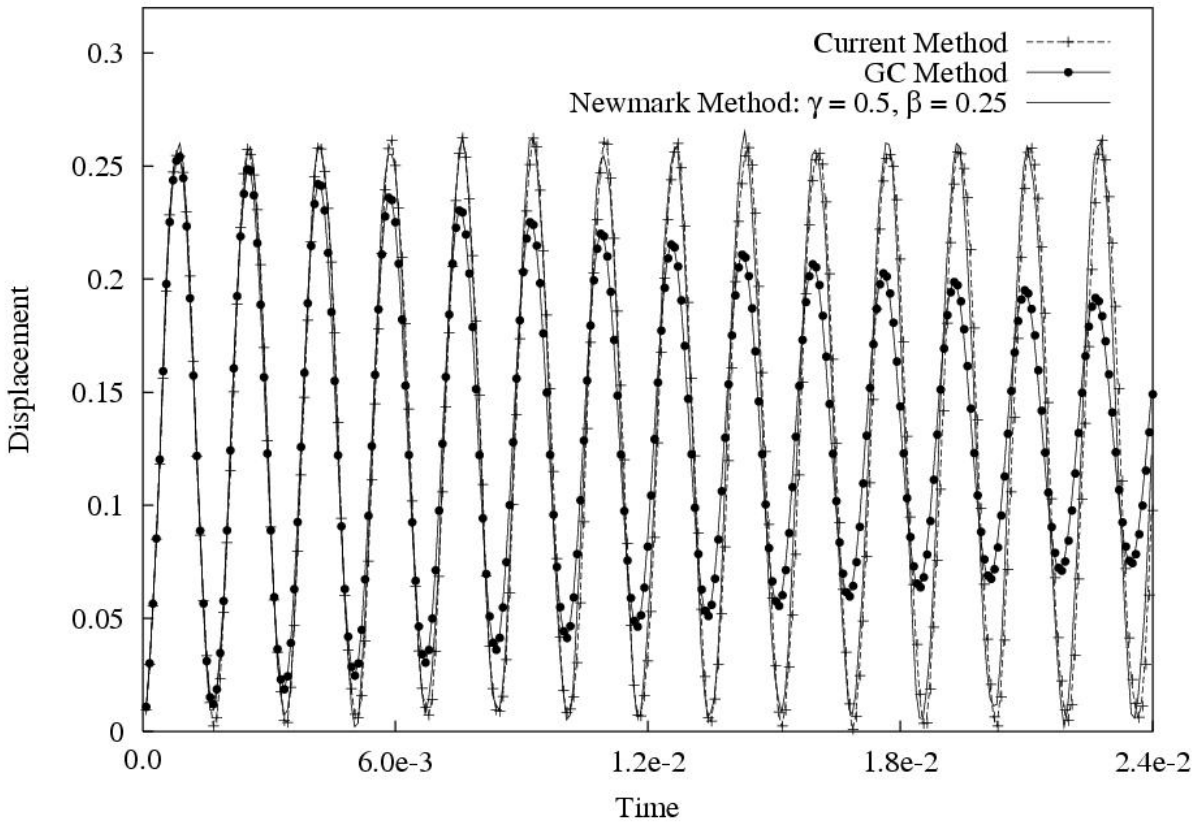

Figure 4.19: Vertical displacement of the mid point on the free edge for the 2-D beam problem, $m = 10$.

the GC method. The parameter values used for this problem are: beam dimensions $L = 1.0$; $d = 0.4$, Young's modulus $E = 1.0 \times 10^4$, density $\rho = 1.0 \times 10^{-4}$, end load $\boldsymbol{P}_0 = 20.0$ distributed over the free edge. For comparison, we also solve the same problem without partitioning using the constant average acceleration method with a time step of 8.0×10^{-5}. Figure 4.19 shows that the GC method is dissipative even for larger problems when the size of the interface is comparable to the total problem size while the present method preserves the total energy of the component subdomains.

4.6.4 Rocket Case with Cracks

A rocket case with one longitudinal and one tangential crack is shown in Figure 4.20. The mesh is decomposed into 2 subdomains such that the 4 crack tip zones together form a single disconnected subdomain and the rest of the mesh form the other subdomain. The case is modeled with a linear elastic material model representing steel and a time step ratio of 10 was used between the subdomains.

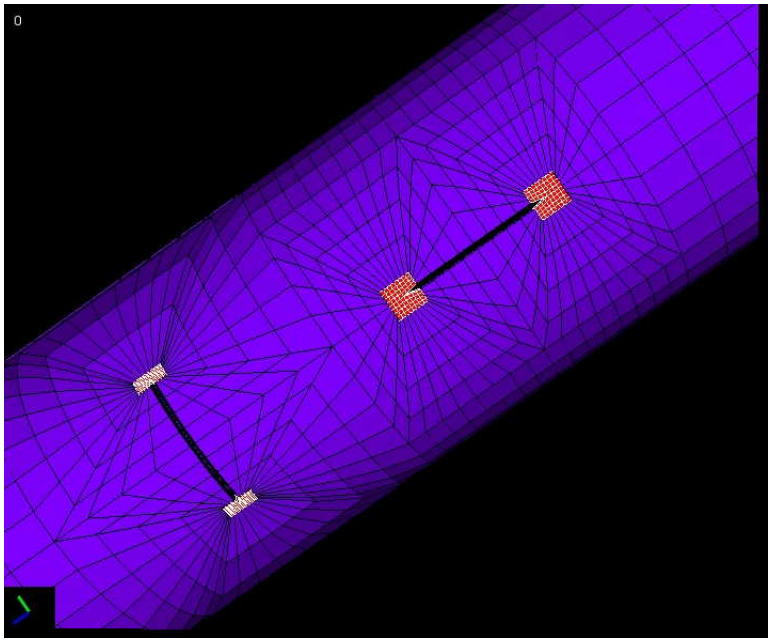

Figure 4.20: Mesh decomposition of the rocket case.

The rocket case is loaded with a sudden head end pressure simulating the start of combustion and the response of the stress wave passing the cracks is shown in Figure 4.21. Note that the coupling is seamless and there is no artifact of the interface around the crack tips.

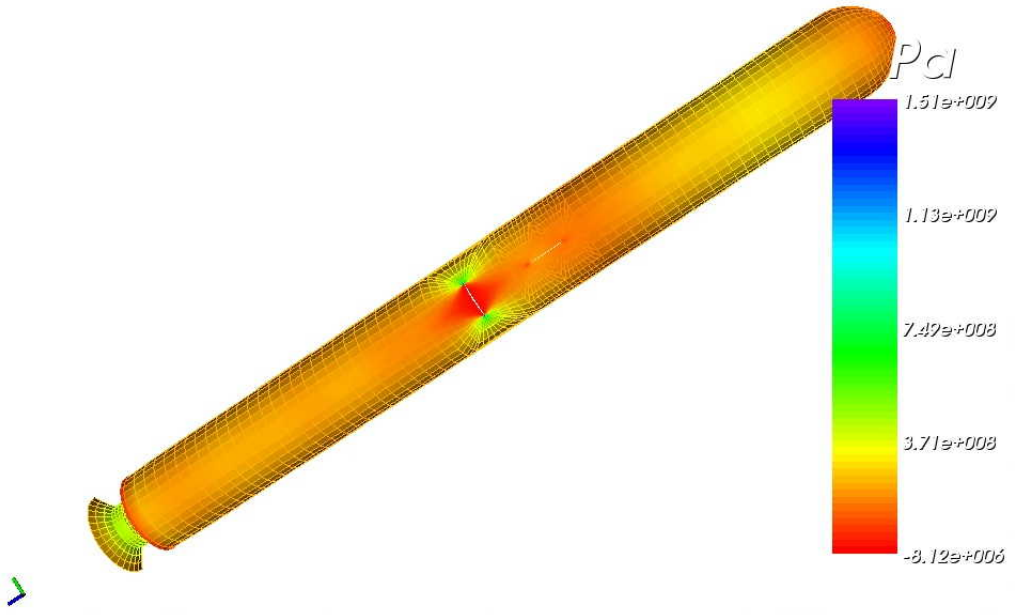

Figure 4.21: Linear elastic response of cracked rocket case to sudden head end pressure.

4.7 Conclusion

The present multi-time-step coupling method provides a powerful approach to solving large-scale structural dynamics problems very efficiently and accurately. Using this coupling method, one can divide a large mesh into smaller subdomains and integrate them independently with different numerical schemes and time steps. This method also affords the possibility of solving different subdomains in parallel with minimal communication needed between them. If the subdomains are chosen carefully, one can solve large-scale problems much faster in time compared to traditional time integration schemes with a uniform time step for the entire mesh.

The present coupling method has been shown to be unconditionally stable as long as the stability requirements of the individual subdomains are met. This means that one should be able to couple the subdomains with as large a time step ratio between them as one desires. It has also been shown that the present method preserves the total energy from its component subdomains while the GC method exhibits significant dissipation when the timestep ratio between the subdomains is high or the size of the interface is comparable to the problem size. Since the energy contribution due to the coupling is zero, one can also integrate individual subdomains to any degree of accuracy one desires.

The most important feature of the present method is that it is computationally far more efficient than the GC method. This is made possible by enforcing the continuity of velocities constraint only at the maximum time step in the mesh. Thus, one is required to solve for the interface reactions at the maximum time step rather than the minimum time step as in the case of the GC method. This makes the present coupling method m (time step ratio) times faster than the GC method.

Chapter 5 discusses the formulation of the present multi-time-step coupling method for non-linear problems.

Chapter 5

Extension to Non-linear Systems

A multi-time-step domain decomposition and coupling method is formulated for non-linear structural dynamics using Newmark schemes. In this method, a large global mesh is partitioned into sub-meshes, each consisting of elements with similar time-step requirements. The different sub-meshes are solved independently and then coupled together in a consistent manner to obtain the global solution. This allows one to integrate each subdomain of the global mesh with an appropriate time-step and/or time stepping scheme. The present method is an extension of the method proposed earlier for linear systems with two subdomains (Ref. [95]) to analyze non-linear systems with multiple subdomains and multiple time-steps. The coupling is achieved through Lagrange multipliers by imposing continuity of velocity across the interfaces between subdomains. We show that the coupling method exactly preserves the total system energy and inherits the stability and accuracy characteristics of its component subdomains. It also exhibits seamless integration between the subdomains with minimal interface computation. Results from several benchmark problems verify these properties and are used to compare its performance with respect to other coupling methods.

5.1 Introduction

Physical problems that involve multiple fields or span over a wide range of length and/or time scales are ideal candidates for coupled simulations. Non-linear structural dynamics of a large and complex system is one such problem. It involves capturing not only the global response of the structure but also the localized fine scale phenomena such as crack prop-

agation and plastic yielding. This requires a very fine spatial and temporal discretization in certain regions of interest within the domain whereas a relatively coarser discretization suffices elsewhere. Most commonly, a finite element mesh is used to discretize the spatial domain and a finite difference time stepping scheme is used for numerical time integration. The time-step of such schemes is usually limited by a stability condition or an accuracy requirement of individual elements. In large meshes, the differences in element sizes and associated time-steps can span over multiple orders of magnitude. Traditionally, however, the time step for the entire mesh is limited by the smallest time-step in the domain. This is severely restrictive and computationally very inefficient. In order to overcome these limitations, one must solve these regions or subdomains separately with different time steps and/or integration schemes and then couple the solutions together.

Domain decomposition methods [96] such as the finite element tearing and interconnecting (FETI) method [81] are commonly used to partition large finite element meshes into subdomains. FETI methods employ element partitioning and use Lagrange multipliers to impose continuity conditions at the interface of shared nodes. Gravouil and Combescure [93] proved that imposing continuity of velocities at the interface led to a stable algorithm but their method was shown to be dissipative. Prakash and Hjelmstad [95] developed a multi-time-step coupling method which is stable, energy preserving and computationally more efficient than other methods currently available in the literature. The idea of *unbalanced interface reactions* proposed in [95] is extended to *unbalanced interface residuals* for non-linear subdomains. The non-linear multi-time-step coupling method exhibits the properties of stability, energy preservation and computational efficiency similar to its linear counterpart. In addition, an iterative PCG method is also presented to solve the interface problem. This is especially suitable for solving non-linear problems.

5.2 Non-linear Multi-time-step Coupling Method

The semi-discrete equations governing the dynamics of a structural domain Ω decomposed into 2 subdomains Ω^A and Ω^B are given by:

$$\boldsymbol{M}^A\ddot{\boldsymbol{U}}^A + \boldsymbol{R}^A(\dot{\boldsymbol{U}}^A, \boldsymbol{U}^A) + \boldsymbol{C}^{A^{\mathrm{T}}}\boldsymbol{\Lambda} = \boldsymbol{P}^A \tag{5.1}$$

$$\boldsymbol{M}^B\ddot{\boldsymbol{U}}^B + \boldsymbol{R}^B(\dot{\boldsymbol{U}}^B, \boldsymbol{U}^B) + \boldsymbol{C}^{B^{\mathrm{T}}}\boldsymbol{\Lambda} = \boldsymbol{P}^B \tag{5.2}$$

$$\boldsymbol{C}^A\dot{\boldsymbol{U}}^A + \boldsymbol{C}^B\dot{\boldsymbol{U}}^B = \boldsymbol{0} \tag{5.3}$$

where $\boldsymbol{M}^k$, $\boldsymbol{R}^k$, $\boldsymbol{U}^k$ and $\boldsymbol{P}^k$ denote the mass matrix, internal force vector, displacement vector and the load vector respectively associated with subdomain Ω^k, $k = A, B$. $\boldsymbol{C}^k$ is the boolean connectivity matrix and $\boldsymbol{\Lambda}$ is the vector of Lagrange multipliers associated with the FETI decomposition. Similar to the linear formulation, we choose to impose continuity of velocities between the subdomains through equation (5.3). In order to simplify the formulation of the multi-time-step coupling method for non-linear systems, only two subdomains are considered here. The present method can be extended for multiple subdomains using a recursive domain decomposition approach along the lines of [97].

Subdomains A and B are integrated with time steps ΔT and Δt respectively where $\Delta T = m\Delta t$ for some integer multiple m. The Newmark parameters for the two subdomains are (γ_A, β_A) and (γ_B, β_B). We will consider stepping the solution from a known state t_0 to $t_m = t_0 + \Delta T$. This procedure can then be repeated to advance the solution for as many time steps as desired. We will follow the *block matrix* notation for multi-time-step quantities described in section §4.1.

The residual equations for stepping subdomain A from time step 0 to m can be written

as:

$$\mathcal{G}_m^{Aa}(\mathbb{U}_m^A, \mathbf{\Lambda}_m) \equiv \boldsymbol{M}^A \ddot{\boldsymbol{U}}_m^A + \boldsymbol{R}^A(\dot{\boldsymbol{U}}_m^A, \boldsymbol{U}_m^A) + \boldsymbol{C}^{A^T} \mathbf{\Lambda}_m - \boldsymbol{P}_m^A \tag{5.4}$$

$$\mathcal{G}_m^{Av}(\mathbb{U}_m^A) \equiv \dot{\boldsymbol{U}}_m^A - \left[\dot{\boldsymbol{U}}_0^A + \Delta T\,(1-\gamma_A)\,\ddot{\boldsymbol{U}}_0^A + \gamma_A \Delta T \ddot{\boldsymbol{U}}_m^A\right] \tag{5.5}$$

$$\mathcal{G}_m^{Ad}(\mathbb{U}_m^A) \equiv \boldsymbol{U}_m^A - \left[\boldsymbol{U}_0^A + \Delta T\ \dot{\boldsymbol{U}}_0^A + \Delta T^2\left(\tfrac{1}{2} - \beta_A\right)\ddot{\boldsymbol{U}}_0^A + \beta_A \Delta T^2 \ddot{\boldsymbol{U}}_m^A\right] \tag{5.6}$$

Note that only equation (5.4) contains a non-linear term but since the solution is coupled through the Newmark relations and the continuity of velocities constraint, it is better to write all the equations in residual form. The residual equations to advance subdomain B from time step $j-1$ to j can be written as:

$$\mathcal{G}_j^{Ba}(\mathbb{U}_j^B, \mathbf{\Lambda}_j) \equiv \boldsymbol{M}^B \ddot{\boldsymbol{U}}_j^B + \boldsymbol{R}^B(\dot{\boldsymbol{U}}_j^B, \boldsymbol{U}_j^B) + \boldsymbol{C}^{B^T} \mathbf{\Lambda}_j - \boldsymbol{P}_j^B \tag{5.7}$$

$$\mathcal{G}_j^{Bv}(\mathbb{U}_j^B, \mathbb{U}_{j-1}^B) \equiv \dot{\boldsymbol{U}}_j^B - \left[\dot{\boldsymbol{U}}_{j-1}^B + \Delta t\,(1-\gamma_B)\,\ddot{\boldsymbol{U}}_{j-1}^B + \gamma_B \Delta t \ddot{\boldsymbol{U}}_j^B\right] \tag{5.8}$$

$$\mathcal{G}_j^{Bd}(\mathbb{U}_j^B, \mathbb{U}_{j-1}^B) \equiv \boldsymbol{U}_j^B - \left[\boldsymbol{U}_{j-1}^B + \Delta t\ \dot{\boldsymbol{U}}_{j-1}^B + \Delta t^2\left(\tfrac{1}{2} - \beta_B\right)\ddot{\boldsymbol{U}}_{j-1}^B + \beta_B \Delta t^2 \ddot{\boldsymbol{U}}_j^B\right] \tag{5.9}$$

where $j \in \{1, 2, \cdots m\}$. As in the linear formulation, we will enforce continuity of velocities at the final time step m through the residual equation:

$$\mathcal{G}_m^{\Lambda}(\mathbb{U}_m^A, \mathbb{U}_m^B) \equiv \boldsymbol{C}^A \dot{\boldsymbol{U}}_m^A + \boldsymbol{C}^B \dot{\boldsymbol{U}}_m^B \tag{5.10}$$

Residual equations for the interface reactions at the intermediate time steps are obtained from the equilibrium equation of subdomain A at the interface:

$$\mathcal{G}_j^{\Lambda}(\mathbb{U}_j^A, \mathbf{\Lambda}_j) \equiv \quad \boldsymbol{C}^A \left[\boldsymbol{M}^A \ddot{\boldsymbol{U}}_j^A + \boldsymbol{R}^A(\dot{\boldsymbol{U}}_j^A, \boldsymbol{U}_j^A) + \boldsymbol{C}^{A^T} \mathbf{\Lambda}_j - \boldsymbol{P}_j^A\right] \tag{5.11}$$

where $j \in \{1, 2, \cdots (m-1)\}$. In addition, we need to interpolate the state variables of

subdomain A between t_0 and t_m for every intermediate time step t_j:

$$\mathcal{G}_j^{Aa}(\mathbb{U}_j^A, \mathbb{U}_m^A) \equiv \ddot{\boldsymbol{U}}_j^A - \left(1 - \frac{j}{m}\right)\ddot{\boldsymbol{U}}_0^A - \left(\frac{j}{m}\right)\ddot{\boldsymbol{U}}_m^A \tag{5.12}$$

$$\mathcal{G}_j^{Av}(\mathbb{U}_j^A, \mathbb{U}_m^A) \equiv \dot{\boldsymbol{U}}_j^A - \left(1 - \frac{j}{m}\right)\dot{\boldsymbol{U}}_0^A - \left(\frac{j}{m}\right)\dot{\boldsymbol{U}}_m^A \tag{5.13}$$

$$\mathcal{G}_j^{Ad}(\mathbb{U}_j^A, \mathbb{U}_m^A) \equiv \boldsymbol{U}_j^A - \left(1 - \frac{j}{m}\right)\boldsymbol{U}_0^A - \left(\frac{j}{m}\right)\boldsymbol{U}_m^A \tag{5.14}$$

5.2.1 Linearization

Assuming that the state at some Newton iteration i is known, we can linearize each of the above residual equations about that state using the Newton's method:

$$\mathcal{G}(\boldsymbol{x}^i + \Delta\boldsymbol{x}^i) \approx \mathcal{G}(\boldsymbol{x}^i) + \nabla\mathcal{G}(\boldsymbol{x}^i)\,\Delta\boldsymbol{x}^i = \boldsymbol{0} \tag{5.15}$$

which can be solved for $\Delta\boldsymbol{x}^i$ to update the solution as:

$$\boldsymbol{x}^{i+1} = \boldsymbol{x}^i + \Delta\boldsymbol{x}^i \tag{5.16}$$

The process is repeated by incrementing the iteration number $i \leftarrow i+1$ until the residual is less than some desired tolerance.

Equations of motion for subdomains A and B can be linearized (see section §2.8.2) for iteration i around an assumed state $\ddot{\boldsymbol{U}}_j^{A^i}$, $\ddot{\boldsymbol{U}}_j^{B^i}$ and $\boldsymbol{\Lambda}_j^i$ for $1 \le j \le m$ as:

$$\boldsymbol{M}^k\Delta\ddot{\boldsymbol{U}}_j^{k^i} + \left.\frac{\partial\boldsymbol{R}_j^k}{\partial\ddot{\boldsymbol{U}}_j^k}\right|_{\ddot{\boldsymbol{U}}_j^{k^i}}\Delta\ddot{\boldsymbol{U}}_j^{k^i} + \boldsymbol{C}^{k^T}\Delta\boldsymbol{\Lambda}_j^i = -\mathcal{G}^k(\ddot{\boldsymbol{U}}_j^{k^i}, \boldsymbol{\Lambda}_j^i) \tag{5.17}$$

The tangent term can be computed as:

$$\frac{\partial\boldsymbol{R}_j^k}{\partial\ddot{\boldsymbol{U}}_j^k} = \frac{\partial\boldsymbol{R}_j^k}{\partial\dot{\boldsymbol{U}}_j^k}\frac{\partial\dot{\boldsymbol{U}}_j^k}{\partial\ddot{\boldsymbol{U}}_j^k} + \frac{\partial\boldsymbol{R}_j^k}{\partial\boldsymbol{U}_j^k}\frac{\partial\boldsymbol{U}_j^k}{\partial\ddot{\boldsymbol{U}}_j^k} \tag{5.18}$$

$$= \gamma^k\Delta t^k\boldsymbol{D}_j^k + \beta^k\Delta t^{k^2}\boldsymbol{K}_j^k \tag{5.19}$$

Defining the tangent damping matrix as:

$$\boldsymbol{D}_j^{k^i} \equiv \left.\frac{\partial \boldsymbol{R}_j^k}{\partial \dot{\boldsymbol{U}}_j^k}\right|_{\boldsymbol{U}_j^{k^i}} \tag{5.20}$$

and the tangent stiffness matrix as:

$$\boldsymbol{K}_j^{k^i} \equiv \left.\frac{\partial \boldsymbol{R}_j^k}{\partial \boldsymbol{U}_j^k}\right|_{\boldsymbol{U}_j^{k^i}} \tag{5.21}$$

The linearized equations (5.4)-(5.9) can be expressed compactly as:

$$\mathbb{M}_m^{A^i}\ \Delta\mathbb{U}_m^{A^i} + \mathbb{C}^A \Delta\boldsymbol{\Lambda}_m^i = -\mathbb{G}_m^{A^i} \tag{5.22}$$

$$\mathbb{M}_j^{B^i}\ \Delta\mathbb{U}_j^{B^i} + \mathbb{N}^B\ \Delta\mathbb{U}_{j-1}^{B^i} + \mathbb{C}^B \Delta\boldsymbol{\Lambda}_j^i = -\mathbb{G}_j^{B^i} \tag{5.23}$$

where $j \in \{1, 2, \cdots m\}$ and

$$\begin{aligned}
\mathbb{M}_j^{k^i} &= \begin{bmatrix} \boldsymbol{M}^k & \boldsymbol{D}_j^{k^i} & \boldsymbol{K}_j^{k^i} \\ -\gamma_k \Delta t_k \boldsymbol{I}^k & \boldsymbol{I}^k & 0 \\ -\beta_k \Delta t_k^2 \boldsymbol{I}^k & 0 & \boldsymbol{I}^k \end{bmatrix};\ \Delta\mathbb{U}_j^{k^i} = \begin{bmatrix} \Delta\ddot{\boldsymbol{U}}_j^{k^i} \\ \Delta\dot{\boldsymbol{U}}_j^{k^i} \\ \Delta\boldsymbol{U}_j^{k^i} \end{bmatrix} \\
\mathbb{N}^k &= \begin{bmatrix} 0 & 0 & 0 \\ -\Delta t_k(1-\gamma_k)\boldsymbol{I}^k & -\boldsymbol{I}^k & 0 \\ -\Delta t_k^2(\frac{1}{2}-\beta_k)\boldsymbol{I}^k & -\Delta t_k \boldsymbol{I}^k & -\boldsymbol{I}^k \end{bmatrix};\ \mathbb{C}^k = \begin{bmatrix} \boldsymbol{C}^{k^T} \\ 0 \\ 0 \end{bmatrix} \\
\mathbb{G}_m^{A^i} &= \begin{bmatrix} \mathcal{G}_m^{Aa}(\mathbb{U}_m^{A^i}, \boldsymbol{\Lambda}_m^i) \\ \mathcal{G}_m^{Av}(\mathbb{U}_m^{A^i}) \\ \mathcal{G}_m^{Ad}(\mathbb{U}_m^{A^i}) \end{bmatrix};\ \mathbb{G}_j^{B^i} = \begin{bmatrix} \mathcal{G}_j^{Ba}(\mathbb{U}_j^{B^i}, \boldsymbol{\Lambda}_j^i) \\ \mathcal{G}_j^{Bv}(\mathbb{U}_j^{B^i}, \mathbb{U}_{j-1}^{B^i}) \\ \mathcal{G}_j^{Bd}(\mathbb{U}_j^{B^i}, \mathbb{U}_{j-1}^{B^i}) \end{bmatrix}
\end{aligned} \tag{5.24}$$

Note that $\Delta\mathbb{U}_0^{A^i} = 0$ and $\Delta\mathbb{U}_0^{B^i} = 0$ since they are known from the previous time step. The

equations for linear interpolation of the state for subdomain A can be written compactly as:

$$\mathbb{I}^A \, \Delta\mathbb{U}_j^{A^i} - \left(\frac{j}{m}\right) \mathbb{I}^A \, \Delta\mathbb{U}_m^{A^i} = -\mathbb{G}_j^{A^i} \tag{5.25}$$

The interface equations at the intermediate time steps can also be written in the compact notation as:

$$\mathbb{R}_j^{A^i} \Delta\mathbb{U}_j^{A^i} + \boldsymbol{I}^\Lambda \Delta\boldsymbol{\Lambda}_j^i = -\mathbb{G}_j^{\Lambda^i} \tag{5.26}$$

where $\mathbb{R}_j^{A^i} \equiv \boldsymbol{C}^A \begin{bmatrix} \boldsymbol{M}^A & \boldsymbol{D}_j^{A^i} & \boldsymbol{K}_j^{A^i} \end{bmatrix}$ and $\mathbb{G}_j^{\Lambda^i} \equiv \mathcal{G}_j^\Lambda(\mathbb{U}_j^{A^i}, \boldsymbol{\Lambda}_j^i)$ for $j \in \{1, 2, \cdots (m-1)\}$. Note that $\boldsymbol{I}^\Lambda$ is an identity matrix with dimension equal to the interface size. Lastly the interface equation at the final time step is:

$$\mathbb{B}^A \Delta\mathbb{U}_m^{A^i} + \mathbb{B}^B \Delta\mathbb{U}_m^{B^i} = -\mathbb{G}_m^{\Lambda^i} \tag{5.27}$$

where $\mathbb{B}^k \equiv \begin{bmatrix} \mathbf{0} & \boldsymbol{C}^k & \mathbf{0} \end{bmatrix}$ and $\mathbb{G}_m^{\Lambda^i} \equiv \mathcal{G}_m^\Lambda(\mathbb{U}_m^{A^i}, \mathbb{U}_m^{B^i})$.

5.2.2 Solution

In order to solve the above system of equations efficiently in a time-stepping manner, we will derive a relationship between the incremental interface reactions at the intermediate time steps $\Delta\boldsymbol{\Lambda}_j^i$ and the incremental interface reaction at the final time step $\Delta\boldsymbol{\Lambda}_m^i$. For this purpose, we assemble the linearized equations of the previous section into a big matrix to solve them using a bordered approach:

$$\left[\begin{array}{cc|c} \mathsf{M}^{B^i} & & \mathsf{C}^B \\ & \mathsf{M}^{A^i} & \mathsf{C}^A \\ \hline \mathsf{B}^{B^i} & \mathsf{B}^{A^i} & \mathsf{H} \end{array}\right] \left[\begin{array}{c} \Delta\mathsf{U}^{B^i} \\ \Delta\mathsf{U}^{A^i} \\ \hline \Delta\Lambda^i \end{array}\right] = \left[\begin{array}{c} -\mathsf{G}^{B^i} \\ -\mathsf{G}^{A^i} \\ \hline -\mathsf{G}^{\Lambda^i} \end{array}\right] \tag{5.28}$$

where

$$\mathsf{M}^{B^i} = \begin{bmatrix} \mathbb{M}_1^{B^i} & & & \\ \mathbb{N}^B & \mathbb{M}_2^{B^i} & & \\ & \ddots & \ddots & \\ & & \mathbb{N}^B & \mathbb{M}_m^{B^i} \end{bmatrix};\ \mathsf{C}^B = \begin{bmatrix} \mathbb{C}^B & & & \\ & \mathbb{C}^B & & \\ & & \ddots & \\ & & & \mathbb{C}^B \end{bmatrix} \tag{5.29}$$

$$\mathsf{M}^{A^i} = \left[\begin{array}{cccc|c} \mathbb{I}^A & & & & -(\frac{1}{m})\mathbb{I}^A \\ & \mathbb{I}^A & & & -(\frac{2}{m})\mathbb{I}^A \\ & & \ddots & & \vdots \\ & & & \mathbb{I}^A & -(\frac{m-1}{m})\mathbb{I}^A \\ \hline & & & & \mathbb{M}_m^{A^i} \end{array}\right];\ \mathsf{C}^A = \left[\begin{array}{ccc|c} 0 & \cdots & 0 & 0 \\ \vdots & \cdots & \vdots & \vdots \\ \vdots & \cdots & \vdots & 0 \\ \hline 0 & \cdots & 0 & \mathbb{C}^A \end{array}\right] \tag{5.30}$$

$$\mathsf{B}^{B^i} = \begin{bmatrix} 0 & \cdots & 0 & 0 \\ \vdots & \cdots & \vdots & \vdots \\ \vdots & \cdots & \vdots & 0 \\ 0 & \cdots & 0 & \mathbb{B}^B \end{bmatrix};\ \mathsf{B}^{A^i} = \left[\begin{array}{cccc|c} \mathbb{R}_1^{A^i} & & & & \\ & \mathbb{R}_2^{A^i} & & & \\ & & \ddots & & \\ & & & \mathbb{R}_{m-1}^{A^i} & \\ \hline & & & & \mathbb{B}^A \end{array}\right] \tag{5.31}$$

$$\mathsf{H} = \left[\begin{array}{cccc|c} \boldsymbol{I}^\Lambda & & & & \\ & \boldsymbol{I}^\Lambda & & & \\ & & \ddots & & \\ & & & \boldsymbol{I}^\Lambda & \\ \hline & & & & 0 \end{array}\right];\ \Delta\mathsf{\Lambda}^i = \left[\begin{array}{c} \Delta\boldsymbol{\Lambda}_1^i \\ \Delta\boldsymbol{\Lambda}_2^i \\ \vdots \\ \Delta\boldsymbol{\Lambda}_{m-1}^i \\ \hline \Delta\boldsymbol{\Lambda}_m^i \end{array}\right] \tag{5.32}$$

$$\Delta\mathsf{U}^{k^i} = \begin{bmatrix} \Delta\mathbb{U}_1^{k^i} & \Delta\mathbb{U}_2^{k^i} & \cdots & \Delta\mathbb{U}_m^{k^i} \end{bmatrix}^T;\ \mathsf{G}^{k^i} = \begin{bmatrix} \mathbb{G}_1^{k^i} & \mathbb{G}_2^{k^i} & \cdots & \mathbb{G}_m^{k^i} \end{bmatrix}^T \tag{5.33}$$

Let

$$\begin{aligned} &\Delta \mathsf{U}^{k^i} = \Delta \mathsf{V}^{k^i} + \Delta \mathsf{W}^{k^i} \text{ where } \quad \Delta \mathsf{V}^{k^i} = -\left[\mathsf{M}^{k^i}\right]^{-1} \mathsf{G}^{k^i} \\ &\Delta \mathsf{W}^{k^i} = -\mathsf{Y}^{k^i} \Delta \Lambda^i \; ; \quad \mathsf{Y}^{k^i} = \left[\mathsf{M}^{k^i}\right]^{-1} \mathsf{C} \end{aligned} \tag{5.34}$$

Similar to $\Delta \mathsf{U}^{k^i}$, $\Delta \mathsf{V}^{k^i}$ and $\Delta \mathsf{W}^{k^i}$ can also be expressed as:

$$\Delta \mathsf{V}^{k^i} = \begin{bmatrix} \Delta \mathbb{V}_1^{k^i} \\ \Delta \mathbb{V}_2^{k^i} \\ \cdots \\ \Delta \mathbb{V}_m^{k^i} \end{bmatrix} ; \quad \Delta \mathsf{W}^{k^i} = \begin{bmatrix} \Delta \mathbb{W}_1^{k^i} \\ \Delta \mathbb{W}_2^{k^i} \\ \cdots \\ \Delta \mathbb{W}_m^{k^i} \end{bmatrix} \tag{5.35}$$

For subdomain B, Y^{B^i} is given by:

$$\begin{aligned} \mathsf{Y}^{B^i} &= [\mathsf{M}^{B^i}]^{-1} \mathsf{C}^B \\ &= \begin{bmatrix} \mathbb{M}_1^{B^i} & & & \\ \mathbb{N}^B & \mathbb{M}_2^{B^i} & & \\ & \ddots & \ddots & \\ & & \mathbb{N}^B & \mathbb{M}_m^{B^i} \end{bmatrix} \begin{bmatrix} \mathbb{C}^B & & & \\ & \mathbb{C}^B & & \\ & & \ddots & \\ & & & \mathbb{C}^B \end{bmatrix} \\ &= \begin{bmatrix} \mathbb{Y}_{11}^{B^i} & & & \\ \mathbb{Y}_{21}^{B^i} & \mathbb{Y}_{22}^{B^i} & & \\ \vdots & \vdots & \ddots & \\ \mathbb{Y}_{m1}^{B^i} & \mathbb{Y}_{m2}^{B^i} & \cdots & \mathbb{Y}_{mm}^{B^i} \end{bmatrix} \quad \text{(say)} \end{aligned} \tag{5.36}$$

Similarly, for subdomain A, Y^{A^i} is given by:

$$\mathsf{Y}^{A^i} = [\mathsf{M}^{A^i}]^{-1}\mathsf{C}^A$$

$$= \left[\begin{array}{ccc|c} \mathbb{I}^A & & & -(\frac{1}{m})\mathbb{I}^A \\ & \mathbb{I}^A & & -(\frac{2}{m})\mathbb{I}^A \\ & & \ddots & \vdots \\ & & \mathbb{I}^A & -(\frac{m-1}{m})\mathbb{I}^A \\ \hline & & & \mathbb{M}_m^{A^i} \end{array}\right] \left[\begin{array}{ccc|c} 0 & \cdots & 0 & 0 \\ \vdots & \cdots & \vdots & \vdots \\ \vdots & \cdots & \vdots & 0 \\ \hline 0 & \cdots & 0 & \mathbb{C}^A \end{array}\right] = \left[\begin{array}{ccc|c} 0 & \cdots & 0 & \mathbb{Y}_1^{A^i} \\ \vdots & \cdots & \vdots & \vdots \\ \vdots & \cdots & \vdots & \mathbb{Y}_{m-1}^{A^i} \\ \hline 0 & \cdots & 0 & \mathbb{Y}_m^{A^i} \end{array}\right] \tag{5.37}$$

From the last row of equation (5.28), $\Delta\Lambda^i$ can be computed as:

$$\left[\mathsf{H} - \sum_{k=A,B} \mathsf{B}^{k^i}\mathsf{Y}^{k^i}\right] \Delta\Lambda^i = \{-\mathsf{G}^{\Lambda^i} - \sum_{k=A,B} \mathsf{B}^{k^i}\Delta\mathsf{V}^{k^i}\} \tag{5.38}$$

One may verify that the interface matrix on the left of the above equation is given by:

$$- \left[\begin{array}{ccccc|c} -\boldsymbol{I}^\Lambda & & & & & \mathbb{R}_1^{A^i}\mathbb{Y}_1^{A^i} \\ & -\boldsymbol{I}^\Lambda & & & & \mathbb{R}_2^{A^i}\mathbb{Y}_2^{A^i} \\ & & \ddots & & & \vdots \\ & & & & -\boldsymbol{I}^\Lambda & \mathbb{R}_{m-1}^{A^i}\mathbb{Y}_{m-1}^{A^i} \\ \hline \boldsymbol{C}^B\dot{\boldsymbol{Y}}_{m1}^{B^i} & \boldsymbol{C}^B\dot{\boldsymbol{Y}}_{m2}^{B^i} & \cdots & & \boldsymbol{C}^B\dot{\boldsymbol{Y}}_{m(m-1)}^{B^i} & \boldsymbol{C}^B\dot{\boldsymbol{Y}}_{mm}^{B^i} + \boldsymbol{C}^A\dot{\boldsymbol{Y}}_m^{A^i} \end{array}\right] \tag{5.39}$$

Note that $\mathbb{M}_m^{A^i}\mathbb{Y}_m^{A^i} = \mathbb{C}^A$ and thus $\mathbb{Y}_m^{A^i}$ takes the form $\{\ddot{\boldsymbol{Y}}_m^{A^i}, \gamma_A\Delta T\ddot{\boldsymbol{Y}}_m^{A^i}, \beta_A\Delta T^2\ddot{\boldsymbol{Y}}_m^{A^i}\}^T$, where

$\ddot{\boldsymbol{Y}}_m^{A^i} = [\tilde{\boldsymbol{M}}_m^{A^i}]^{-1}\boldsymbol{C}^{A^T}$ and $\tilde{\boldsymbol{M}}_m^{A^i} = [\boldsymbol{M}^A + \gamma_A \Delta T \boldsymbol{D}^A + \beta_A \Delta T^2 \boldsymbol{K}_m^{A^i}]$. Thus:

$$
\begin{aligned}
&\mathbb{R}_j^{A^i}\mathbb{Y}_j^{A^i} = (\frac{j}{m})\mathbb{R}_j^{A^i}\mathbb{Y}_m^{A^i} \\
&= (\frac{j}{m})\boldsymbol{C}^A \begin{bmatrix} \boldsymbol{M}^A & \boldsymbol{D}^A & \boldsymbol{K}_j^{A^i} \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{Y}}_m^{A^i} \\ \gamma_A \Delta T \ddot{\boldsymbol{Y}}_m^{A^i} \\ \beta_A \Delta T^2 \ddot{\boldsymbol{Y}}_m^{A^i} \end{bmatrix} \\
&= (\frac{j}{m})\boldsymbol{C}^A \left[\boldsymbol{M}^A + \gamma_A \Delta T \boldsymbol{D}^A + \beta_A \Delta T^2 \boldsymbol{K}_j^{A^i}\right] \left[\tilde{\boldsymbol{M}}_m^{A^i}\right]^{-1} \boldsymbol{C}^{A^T} \\
&= (\frac{j}{m})\boldsymbol{C}^A \left[\tilde{\boldsymbol{M}}_j^{A^i}\right] \left[\tilde{\boldsymbol{M}}_m^{A^i}\right]^{-1} \boldsymbol{C}^{A^T}
\end{aligned}
\tag{5.40}
$$

Let

$$
\boldsymbol{J}_j^{\Lambda} \equiv \boldsymbol{C}^A \left[\tilde{\boldsymbol{M}}_j^{A^i}\right] \left[\tilde{\boldsymbol{M}}_m^{A^i}\right]^{-1} \boldsymbol{C}^{A^T} \tag{5.41}
$$

Note that if subdomain A is linear over the interval t_0 to t_m or if explicit integration is used, then $\boldsymbol{J}_j^{\Lambda} = \boldsymbol{I}^{\Lambda}$. The right hand side of equation (5.38) is given by:

$$
\{-\mathsf{G}^{\Lambda^i} - \sum_{k=A,B} \mathsf{B}^{k^i} \Delta \mathsf{V}^{k^i}\} = \left[\begin{array}{c} -\mathbb{G}_1^{\Lambda^i} - \mathbb{R}_1^{A^i}\ \Delta\mathbb{V}_1^{A^i} \\ -\mathbb{G}_2^{\Lambda^i} - \mathbb{R}_2^{A^i}\ \Delta\mathbb{V}_2^{A^i} \\ \vdots \\ -\mathbb{G}_{m-1}^{\Lambda^i} - \mathbb{R}_{m-1}^{A^i}\ \Delta\mathbb{V}_{m-1}^{A^i} \\ \hline -\mathbb{G}_m^{\Lambda^i} - \mathbb{B}^A\ \Delta\mathbb{V}_m^{A^i} - \mathbb{B}^B\ \Delta\mathbb{V}_m^{B^i} \end{array}\right] \tag{5.42}
$$

Define

$$
\begin{aligned}
\boldsymbol{S}_j^i &\equiv -\mathbb{G}_j^{\Lambda^i} - \mathbb{R}_j^{A^i} \Delta\mathbb{V}_j^{A^i} \\
\boldsymbol{S}_j^i &= -\boldsymbol{C}^A \left[\boldsymbol{M}^A \ddot{\boldsymbol{U}}_j^{A^i} + \boldsymbol{D}^A \dot{\boldsymbol{U}}_j^{A^i} + \boldsymbol{R}^A(\boldsymbol{U}_j^{A^i}) + \boldsymbol{C}^{A^T} \boldsymbol{\Lambda}_j^i - \boldsymbol{P}_j^A\right] \\
&\quad - \boldsymbol{C}^A \left[\boldsymbol{M}^A\ \Delta\ddot{\boldsymbol{V}}_j^{A^i} + \boldsymbol{D}^A\ \Delta\dot{\boldsymbol{V}}_j^{A^i} + \boldsymbol{K}_j^{A^i}\ \Delta\boldsymbol{V}_j^{A^i}\right]
\end{aligned}
\tag{5.43}
$$

As in the linear case, here also $\boldsymbol{S}_m^i = \boldsymbol{0}$. Note that $\Delta\mathbb{V}_j^{A^i} = \frac{j}{m}\Delta\mathbb{V}_m^{A^i}$, if $\mathbb{G}_j^{A^i}$ are ensured to be *zero* apriori which can be done by choosing an initial guess for the Newton-Raphson

iterations that satisfies equations (5.12)-(5.14).

Now, we can solve any row j of the interface problem for $\Delta\boldsymbol{\Lambda}_j^i$ in terms of $\Delta\boldsymbol{\Lambda}_m^i$ from the expressions (5.39) and (5.42) as:

$$\Delta\boldsymbol{\Lambda}_j^i = \boldsymbol{S}_j^i + \left(\frac{j}{m}\right) \boldsymbol{J}_j^\Lambda \Delta\boldsymbol{\Lambda}_m^i \tag{5.44}$$

Using the above relation, equation (5.28) can be simplified and solved in a time-stepping manner.

5.3 Implementation

Using the relation (5.44), equation (5.23) can be expressed as:

$$\mathbb{M}_j^{B^i}\ \Delta\mathbb{U}_j^{B^i} + \mathbb{N}^B\ \Delta\mathbb{U}_{j-1}^{B^i} + \left(\frac{j}{m}\right) \mathbb{C}^B \boldsymbol{J}_j^\Lambda \Delta\boldsymbol{\Lambda}_m^i = -\mathbb{G}_j^{B^i} - \mathbb{C}^B \boldsymbol{S}_j^i \tag{5.45}$$

The final simplified system to be solved takes the form:

$$\left[\begin{array}{cc|c} \mathbb{M}_1^{B^i} & & \frac{1}{m}\mathbb{C}^B\boldsymbol{J}_1^\Lambda \\ \mathbb{N}^B\ \mathbb{M}_2^{B^i} & & \frac{2}{m}\mathbb{C}^B\boldsymbol{J}_2^\Lambda \\ \ddots\ \ddots & & \vdots \\ \mathbb{N}^B\ \mathbb{M}_m^{B^i} & & \mathbb{C}^B\boldsymbol{J}_m^\Lambda \\ \hline & \mathbb{M}_m^{A^i} & \mathbb{C}^A \\ \hline \mathbb{B}^B & \mathbb{B}^A & \mathbf{0} \end{array}\right] \left[\begin{array}{c} \Delta\mathbb{U}_1^{B^i} \\ \Delta\mathbb{U}_2^{B^i} \\ \vdots \\ \Delta\mathbb{U}_m^{B^i} \\ \hline \Delta\mathbb{U}_m^{A^i} \\ \hline \Delta\boldsymbol{\Lambda}_m^i \end{array}\right] = \left[\begin{array}{c} -\mathbb{G}_1^{B^i} - \mathbb{C}^B\boldsymbol{S}_1^i \\ -\mathbb{G}_2^{B^i} - \mathbb{C}^B\boldsymbol{S}_2^i \\ \vdots \\ -\mathbb{G}_m^{B^i} - \mathbb{C}^B\boldsymbol{S}_m^i \\ \hline -\mathbb{G}_m^{A^i} \\ \hline -\mathbb{G}_m^{\Lambda^i} \end{array}\right] \tag{5.46}$$

Note that $\boldsymbol{J}_m^\Lambda = \boldsymbol{I}^\Lambda$.

Equation (5.46) can be represented again by a bordered system similar to equation (5.28)

with the following quantities redefined:

$$\mathsf{H} = \mathbf{0};\ \Delta\Lambda^i = \Delta\boldsymbol{\Lambda}^i_m;\ \mathsf{G}^{\Lambda^i} = \mathbb{G}^{\Lambda^i}_m$$

$$\mathsf{M}^{A^i} = \mathbb{M}^{A^i}_m;\ \mathsf{C}^A = \mathbb{C}^A;\ \mathsf{B}^{A^i} = \mathbb{B}^A;\ \mathsf{G}^{A^i} = \mathbb{G}^{A^i}_m;\ \Delta\mathsf{U}^{A^i} = \Delta\mathbb{U}^{A^i}_m$$

$$\mathsf{B}^{B^i} = \begin{bmatrix} 0 & \cdots & 0 & \mathbb{B}^B \end{bmatrix};\ \mathsf{C}^B = \begin{bmatrix} \frac{1}{m}\mathbb{C}^B\boldsymbol{J}^\Lambda_1 \\ \frac{2}{m}\mathbb{C}^B\boldsymbol{J}^\Lambda_2 \\ \vdots \\ \mathbb{C}^B\boldsymbol{J}^\Lambda_m \end{bmatrix}; \mathsf{G}^{B^i} = \begin{bmatrix} \mathbb{G}^{B^i}_1 + \mathbb{C}^B\boldsymbol{S}^i_1 \\ \mathbb{G}^{B^i}_2 + \mathbb{C}^B\boldsymbol{S}^i_2 \\ \vdots \\ \mathbb{G}^{B^i}_m + \mathbb{C}^B\boldsymbol{S}^i_m \end{bmatrix}$$

M^{B^i} and $\Delta\mathsf{U}^{B^i}$ remain unchanged.

Following the steps in eqn. (5.34), we can compute $\Delta\mathsf{V}^{A^i}$ by solving:

$$\mathbb{M}^{A^i}_m \Delta\mathbb{V}^{A^i}_m = -\mathbb{G}^{A^i}_m \tag{5.47}$$

$\Delta\mathbb{V}^{A^i}_j$ are obtained by linear interpolation. The unbalanced residuals $\boldsymbol{S}^i_j$ can then be computed from eqn. (5.43) and transferred to subdomain B which is solved as:

$$\begin{bmatrix} \mathbb{M}^{B^i}_1 & & & \\ \mathbb{N}^B & \mathbb{M}^{B^i}_2 & & \\ & \ddots & \ddots & \\ & & \mathbb{N}^B & \mathbb{M}^{B^i}_m \end{bmatrix} \begin{bmatrix} \Delta\mathbb{V}^{B^i}_1 \\ \Delta\mathbb{V}^{B^i}_2 \\ \vdots \\ \Delta\mathbb{V}^{B^i}_m \end{bmatrix} = \begin{bmatrix} -\mathbb{G}^{B^i}_1 - \mathbb{C}^B\boldsymbol{S}^i_1 \\ -\mathbb{G}^{B^i}_2 - \mathbb{C}^B\boldsymbol{S}^i_2 \\ \vdots \\ -\mathbb{G}^{B^i}_m - \mathbb{C}^B\boldsymbol{S}^i_m \end{bmatrix} \tag{5.48}$$

Let $\mathbb{V}^{k^i}_j \equiv \mathbb{U}^{k^i}_j + \Delta\mathbb{V}^{k^i}_j$. Thus, $\Delta\mathbb{U}^{k^i}_j = \Delta\mathbb{V}^{k^i}_j + \Delta\mathbb{W}^{k^i}_j$ and $\mathbb{U}^{k^{(i+1)}}_j = \mathbb{V}^{k^i}_j + \Delta\mathbb{W}^{k^i}_j$.

The Y^{k^i} matrices can also be computed similarly:

$$[\mathbb{M}_m^{A^i}]\mathbb{Y}_m^{A^i} = \mathbb{C}^A \tag{5.49}$$

$$\begin{bmatrix} \mathbb{M}_1^{B^i} & & & \\ \mathbb{N}^B & \mathbb{M}_2^{B^i} & & \\ & \ddots & \ddots & \\ & & \mathbb{N}^B & \mathbb{M}_m^{B^i} \end{bmatrix} \begin{bmatrix} \mathbb{Y}_1^{B^i} \\ \mathbb{Y}_2^{B^i} \\ \vdots \\ \mathbb{Y}_m^{B^i} \end{bmatrix} = \begin{bmatrix} \frac{1}{m}\mathbb{C}^B \boldsymbol{J}_1^\Lambda \\ \frac{2}{m}\mathbb{C}^B \boldsymbol{J}_2^\Lambda \\ \vdots \\ \mathbb{C}^B \boldsymbol{J}_m^\Lambda \end{bmatrix} \tag{5.50}$$

It is usually not feasible to compute Y^{k^i} from the above equation. An iterative approach will be presented in the following section to overcome this difficulty. First, let us outline the entire solution procedure.

The interface problem is:

$$\begin{aligned} \left[\mathbb{B}^B\mathbb{Y}_m^{B^i} + \mathbb{B}^A\mathbb{Y}_m^{A^i}\right]\{\Delta\boldsymbol{\Lambda}_m^i\} &= \{\mathbb{B}^B\Delta\mathbb{V}_m^{B^i} + \mathbb{B}^A\Delta\mathbb{V}_m^{A^i} + \mathbb{G}_m^{\Lambda v^i}\} \\ &= \boldsymbol{C}^A\Delta\dot{\boldsymbol{V}}_m^{A^i} + \boldsymbol{C}^B\Delta\dot{\boldsymbol{V}}_m^{B^i} + \mathcal{G}_m^\Lambda(\mathbb{U}_m^{A^i}, \mathbb{U}_m^{B^i}) \\ &= \boldsymbol{C}^A\left(\dot{\boldsymbol{U}}_m^{A^i} + \Delta\dot{\boldsymbol{V}}_m^{A^i}\right) + \boldsymbol{C}^B\left(\dot{\boldsymbol{U}}_m^{B^i} + \Delta\dot{\boldsymbol{V}}_m^{B^i}\right) \\ \Rightarrow [\boldsymbol{H}]\{\Delta\boldsymbol{\Lambda}_m^i\} &= \{\boldsymbol{C}^A\dot{\boldsymbol{V}}_m^{A^i} + \boldsymbol{C}^B\dot{\boldsymbol{V}}_m^{B^i}\} \end{aligned} \tag{5.51}$$

where $\boldsymbol{H} \equiv \boldsymbol{C}^A\dot{\boldsymbol{Y}}_m^{A^i} + \boldsymbol{C}^B\dot{\boldsymbol{Y}}_m^{B^i}$.

Finally, the solution can be updated by computing $\Delta\mathsf{W}^{k^i}$:

$$\Delta\mathbb{W}_m^{A^i} = -\mathbb{Y}_m^{A^i}\Delta\boldsymbol{\Lambda}_m^i \tag{5.52}$$

$$\begin{bmatrix} \Delta\mathbb{W}_1^{B^i} \\ \Delta\mathbb{W}_2^{B^i} \\ \vdots \\ \Delta\mathbb{W}_m^{B^i} \end{bmatrix} = - \begin{bmatrix} \mathbb{Y}_1^{B^i} \\ \mathbb{Y}_2^{B^i} \\ \vdots \\ \mathbb{Y}_m^{B^i} \end{bmatrix} \Delta\boldsymbol{\Lambda}_m^i \tag{5.53}$$

5.3.1 Iterative Solution of Interface Problem

Since the size of the interface problem (5.51) can be quite large, it is usually not feasible to compute and factorize the interface matrix $\boldsymbol{H}$ especially for non-linear systems when it needs to be done at every Newton-Raphson iteration. Thus, the familiar PCG method for FETI is employed to solve the multi-time-step interface problem iteratively. The PCG iteration number is denoted by ν to distinguish it from the non-linear Newton-Raphson iteration number i.

1. Initialize PCG iteration number $\nu = 0$

$$\begin{aligned} \text{Assume} \quad & \Delta\boldsymbol{\Lambda}_m^{i\ 0} = \boldsymbol{0} \\ & \boldsymbol{r}^{i0} = \{\boldsymbol{C}^A \dot{\boldsymbol{V}}_m^{A^i} + \boldsymbol{C}^B \dot{\boldsymbol{V}}_m^{B^i}\} - \boldsymbol{H}\Delta\boldsymbol{\Lambda}_m^{i\ 0} \\ & \boldsymbol{d}^{i0} = \tilde{\boldsymbol{H}}^{-1}\boldsymbol{r}^{i0} \\ & \boldsymbol{p}^{i0} = \boldsymbol{d}^{i0} \end{aligned} \tag{5.54}$$

2. Iterate for $\nu = 0, 1, 2 \cdots$ until convergence

$$\begin{aligned} \zeta^{i\nu} &= \frac{\boldsymbol{r}^{i\nu T}\boldsymbol{d}^{i\nu}}{\boldsymbol{p}^{i\nu T}\boldsymbol{H}\boldsymbol{p}^{i\nu}} \\ \Delta\boldsymbol{\Lambda}_m^{i\ \nu+1} &= \Delta\boldsymbol{\Lambda}_m^{i\ \nu} + \zeta^{i\nu}\boldsymbol{p}^{i\nu} \\ \boldsymbol{r}^{i\nu+1} &= \boldsymbol{r}^{i\nu} - \zeta^{i\nu}\boldsymbol{H}\boldsymbol{p}^{i\nu} \\ \boldsymbol{d}^{i\nu+1} &= \tilde{\boldsymbol{H}}^{-1}\boldsymbol{r}^{i\nu+1} \\ \alpha^{i\nu+1} &= \frac{\boldsymbol{r}^{i\nu+1T}\boldsymbol{d}^{i\nu+1}}{\boldsymbol{r}^{i\nu T}\boldsymbol{d}^{i\nu}} \\ \boldsymbol{p}^{i\nu+1} &= \boldsymbol{d}^{i\nu+1} + \alpha^{i\nu+1}\boldsymbol{p}^{i\nu} \end{aligned} \tag{5.55}$$

where $\tilde{\boldsymbol{H}}^{-1}$ is a suitable preconditioner. An approximate lumped preconditioner can be defined as:

$$\tilde{\boldsymbol{H}}^{-1} \equiv \sum_{k=A,B} \frac{1}{\gamma^k \Delta t_k} \boldsymbol{C}^k \tilde{\boldsymbol{M}}^k \boldsymbol{C}^{k^T} \tag{5.56}$$

Note that this is not the same as the conventional FETI lumped preconditioner because of the different time steps used for subdomains A and B. The matrix-vector products $\boldsymbol{H}\boldsymbol{p}^{i^\nu} = \boldsymbol{C}^A \dot{\boldsymbol{Y}}_m^{A^i} \boldsymbol{p}^{i^\nu} + \boldsymbol{C}^B \dot{\boldsymbol{Y}}_m^{B^i} \boldsymbol{p}^{i^\nu}$ can be computed subdomain-wise. Noting that the matrices $\dot{\boldsymbol{Y}}_m^{A^i}$ and $\dot{\boldsymbol{Y}}_m^{B^i}$ are computed from eqns. (5.49) and (5.50), we have:

$$\boldsymbol{C}^A \dot{\boldsymbol{Y}}_m^{A^i} \boldsymbol{p}^{i^\nu} = \mathbb{B}^A \left[\mathbb{M}_m^{A^i}\right]^{-1} \{\mathbb{C}^A \boldsymbol{p}^{i^\nu}\} \tag{5.57}$$

which amounts to solving subdomain A for one time step under a *load* $\boldsymbol{C}^{A^T} \boldsymbol{p}^{i^\nu}$. Similarly $\boldsymbol{C}^B \dot{\boldsymbol{Y}}_m^{B^i} \boldsymbol{p}^{i^\nu}$ can be obtained from the *last row* of the expression:

$$\mathbb{B}^B \begin{bmatrix} \mathbb{Y}_1^{B^i} \\ \mathbb{Y}_2^{B^i} \\ \vdots \\ \mathbb{Y}_m^{B^i} \end{bmatrix} \boldsymbol{p}^{i^\nu} = \mathbb{B}^B \begin{bmatrix} \mathbb{M}_1^{B^i} & & & \\ \mathbb{N}^B & \mathbb{M}_2^{B^i} & & \\ & \ddots & \ddots & \\ & & \mathbb{N}^B & \mathbb{M}_m^{B^i} \end{bmatrix}^{-1} \begin{bmatrix} \frac{1}{m}\mathbb{C}^B \boldsymbol{J}_1^\Lambda \boldsymbol{p}^{i^\nu} \\ \frac{2}{m}\mathbb{C}^B \boldsymbol{J}_2^\Lambda \boldsymbol{p}^{i^\nu} \\ \vdots \\ \mathbb{C}^B \boldsymbol{J}_m^\Lambda \boldsymbol{p}^{i^\nu} \end{bmatrix} \tag{5.58}$$

which amounts to solving subdomain B for m time steps under a *load* $\boldsymbol{C}^{B^T} \boldsymbol{J}_j^\Lambda \boldsymbol{p}^{i^\nu}$ at sub-time-step j for all $j \in [1, 2, \cdots, m]$. From (5.41) one may note that:

$$\boldsymbol{C}^{B^T} \boldsymbol{J}_j^\Lambda \boldsymbol{p}^{i^\nu} = \boldsymbol{C}^{B^T} \boldsymbol{C}^A \left[\tilde{\boldsymbol{M}}_j^{A^i}\right] \left[\tilde{\boldsymbol{M}}_m^{A^i}\right]^{-1} \boldsymbol{C}^{A^T} \boldsymbol{p}^{i^\nu} \tag{5.59}$$

Since the term $\left[\tilde{\boldsymbol{M}}_m^{A^i}\right]^{-1} \boldsymbol{C}^{A^T} \boldsymbol{p}^{i^\nu}$ in the above expression has already been computed for subdomain A in equation (5.57), this modified residual calculation is not expensive. Also note that one can save the quantities $\zeta^{i^\nu} \mathbb{Y}_m^{A^i} \boldsymbol{p}^{i^\nu}$ and $\zeta^{i^\nu} \mathbb{Y}_j^{B^i} \boldsymbol{p}^{i^\nu}$ computed above in order to update the subdomain solutions upon convergence using equations (5.52) and (5.53). Since

$\Delta\boldsymbol{\Lambda}_m^i = \Delta\boldsymbol{\Lambda}_m^{i\ 0} + \sum_{\nu=0}^{\nu^*} \zeta^{i\nu} \boldsymbol{p}_m^{i\ \nu}$ therefore

$$\Delta\mathbb{W}_m^{A^i} = -\mathbb{Y}_m^{A^i} \Delta\boldsymbol{\Lambda}_m^{i\ 0} - \sum_{\nu=0}^{\nu^*} \zeta^{i\nu} \mathbb{Y}_m^{A^i} \boldsymbol{p}_m^{i\ \nu} \tag{5.60}$$

$$\begin{bmatrix} \Delta\mathbb{W}_1^{B^i} \\ \Delta\mathbb{W}_2^{B^i} \\ \vdots \\ \Delta\mathbb{W}_m^{B^i} \end{bmatrix} = - \begin{bmatrix} \mathbb{Y}_1^{B^i} \\ \mathbb{Y}_2^{B^i} \\ \vdots \\ \mathbb{Y}_m^{B^i} \end{bmatrix} \Delta\boldsymbol{\Lambda}_m^{i\ 0} - \sum_{\nu=0}^{\nu^*} \begin{bmatrix} \zeta^{i\nu} \mathbb{Y}_1^{B^i} \boldsymbol{p}_m^{i\ \nu} \\ \zeta^{i\nu} \mathbb{Y}_2^{B^i} \boldsymbol{p}_m^{i\ \nu} \\ \vdots \\ \zeta^{i\nu} \mathbb{Y}_m^{B^i} \boldsymbol{p}_m^{i\ \nu} \end{bmatrix} \tag{5.61}$$

5.3.2 Final Algorithm

The final algorithm for advancing the solution from time t_0 to t_m is outlined below.

1. Initialize the state variables for subdomains A and B from the converged state for the previous step. Let the iteration number at convergence of previous step be i^*.

   (a) Assume $\ddot{\boldsymbol{U}}_m^{A^0} = \ddot{\boldsymbol{U}}_0^{A^{i^*}}$ and compute $\dot{\boldsymbol{U}}_m^{A^0}$, $\boldsymbol{U}_m^{A^0}$ from (5.5) and (5.6) such that $\mathcal{G}_m^{Av} = 0$ and $\mathcal{G}_m^{Ad} = 0$.

   (b) Assume $\ddot{\boldsymbol{U}}_j^{B^0} = \ddot{\boldsymbol{U}}_0^{B^{i^*}}$ and compute $\dot{\boldsymbol{U}}_j^{B^0}$, $\boldsymbol{U}_j^{B^0}$ from (5.8) and (5.9) such that $\mathcal{G}_j^{Bv} = 0$ and $\mathcal{G}_j^{Bd} = 0$ for all $j \in \{1, 2, \cdots, m\}$.

   (c) Assume $\boldsymbol{\Lambda}_m^0 = \boldsymbol{\Lambda}_0^{i^*}$.

2. Begin Newton-Raphson iterations $i = 0$:

   (a) Compute the residuals in equation (5.46): $\mathbb{G}_m^{A^i}(\mathbb{U}_m^{A^i}, \boldsymbol{\Lambda}_m^i)$, $\mathbb{G}_j^{B^i}(\mathbb{U}_j^{B^i}, \boldsymbol{\Lambda}_j^i)$ and $\mathbb{G}_m^{\Lambda^i}(\mathbb{U}_m^{A^i}, \mathbb{U}_m^{B^i})$.

   (b) Check for convergence $\|\mathbb{G}_m^{A^i}\| < tol$, $\|\mathbb{G}_j^{B^i}\| < tol$ and $\|\mathbb{G}_m^{\Lambda^i}\| < tol$. If converged go to step 3.

   (c) Compute the tangent stiffness matrices $\boldsymbol{K}_j^{A^i}$ and $\boldsymbol{K}_j^{B^i}$.

   (d) Solve for $\Delta\mathbb{V}_m^{A^i}$ from (5.47) and linearly interpolate to obtain $\Delta\mathbb{V}_j^{A^i}$. Get $\mathbb{V}_j^{A^i}$.

(e) Compute $\boldsymbol{S}_j^i$ from (5.43) and transfer unbalanced residuals to subdomain B: $-\mathbb{G}_j^{B^i} - \mathbb{C}^B \boldsymbol{S}_j^i$.

(f) Solve for $\Delta\mathbb{V}_j^{B^i}$ from (5.48) under the modified residuals. Get $\mathbb{V}_j^{B^i}$.

(g) Initialize PCG solution of the interface problem from equations (5.54).

(h) Begin PCG iterations $\nu = 0$:

i. Convergence check: If $\|\boldsymbol{r}^{i^\nu}\| < tol$ go to step (2i).

ii. Solve for $\boldsymbol{C}^A \dot{\boldsymbol{Y}}_m^{A^i} \boldsymbol{p}^{i^\nu}$ from equation (5.57). Save the increments to $\Delta\mathbb{W}_m^{A^i}$ from (5.60).

iii. Solve for $\boldsymbol{C}^B \dot{\boldsymbol{Y}}_m^{B^i} \boldsymbol{p}^{i^\nu}$ from the last row of (5.58). Save the increments to $\Delta\mathbb{W}_j^{B^i}$ from (5.61).

iv. Form $\boldsymbol{H}\boldsymbol{p}^{i^\nu}$. Compute quantities in equation (5.55).

v. Increment PCG iteration counter $\nu \leftarrow \nu + 1$ and check for maximum iteration limit. If $\nu < MAX$ then go to 2(h)i. else display error "PCG did not Converge" and stop.

(i) Update the solution vectors:
$\mathbb{U}_j^{k^{i+1}} = \mathbb{U}_j^{k^i} + \Delta\mathbb{U}_j^{k^i}$, $\boldsymbol{\Lambda}_j^{i+1} = \boldsymbol{\Lambda}_j^i + \Delta\boldsymbol{\Lambda}_j^i$.

(j) Increment the iteration count $i \leftarrow i + 1$ and check for maximum iteration limit. If $i < MAX$ then go to (2a) else display error "Newton Loop did not Converge" and stop.

3. Output and Update quantities for the next time step. Return to calling subroutine.

5.4 Modified Newton Coupling Approach

The non-linear multi-time-step coupling method described above is based on a consistent linearization of the governing equations. This consistent linearization poses some restrictions

on the sequence of computation for the coupling method. For instance, if subdomain A is non-linear and is integrated implicitly then the unbalanced interface residuals cannot be computed in advance unlike the linear coupling method. Some modifications similar to a modified Newton approach lead to significant simplification of the non-linear coupling method as discussed below.

One may choose to replace the interface constraint for the intermediate time steps (5.11) with the equation:

$$\begin{aligned}&\mathcal{G}_j^{\Lambda}(\mathbb{U}_j^A, \boldsymbol{\Lambda}_j) \equiv \\ &\quad \boldsymbol{C}^A \left[\boldsymbol{M}^A \ddot{\boldsymbol{U}}_j^A + \left(1 - \frac{j}{m}\right) \boldsymbol{R}^A(\dot{\boldsymbol{U}}_0^A, \boldsymbol{U}_0^A) + \left(\frac{j}{m}\right) \boldsymbol{R}^A(\dot{\boldsymbol{U}}_m^A, \boldsymbol{U}_m^A) + \boldsymbol{C}^{A^T} \boldsymbol{\Lambda}_j - \boldsymbol{P}_j^A \right]\end{aligned} \tag{5.62}$$

Linearizing:

$$\boldsymbol{C}^A \left[\left(\frac{j}{m}\right) \left(\boldsymbol{M}^A \Delta \ddot{\boldsymbol{U}}_m^{A^i} + \boldsymbol{D}_m^{A^i} \Delta \dot{\boldsymbol{U}}_m^{A^i} + \boldsymbol{K}_m^{A^i} \Delta \boldsymbol{U}_m^{A^i} \right) + \boldsymbol{C}^{A^T} \Delta \boldsymbol{\Lambda}_j^i \right] = -\mathcal{G}_j^{\Lambda^i}(\mathbb{U}_j^{A^i}, \boldsymbol{\Lambda}_j^i) \tag{5.63}$$

$$\Rightarrow \qquad \left(\frac{j}{m}\right) \mathbb{R}_m^{A^i} \Delta \mathbb{U}_m^{A^i} + \boldsymbol{I}^{\Lambda} \Delta \boldsymbol{\Lambda}_j^i = -\mathbb{G}_j^{\Lambda^i} \tag{5.64}$$

Using this relation one can replace B^{A^i} in expression (5.31) with:

$$\mathsf{B}^{A^i} = \left[\begin{array}{ccc|c} 0 & & & \frac{1}{m}\mathbb{R}_m^{A^i} \\ & 0 & & \frac{2}{m}\mathbb{R}_m^{A^i} \\ & & \ddots & \vdots \\ & & 0 & \frac{m-1}{m}\mathbb{R}_m^{A^i} \\ \hline & & & \mathbb{B}^A \end{array}\right] \tag{5.65}$$

The interface matrix (5.39) takes the form:

$$-\left[\begin{array}{ccccc|c} -\boldsymbol{I}^{\Lambda} & & & & & \frac{1}{m}\mathbb{R}_m^{A^i}\mathbb{Y}_m^{A^i} \\ & -\boldsymbol{I}^{\Lambda} & & & & \frac{2}{m}\mathbb{R}_m^{A^i}\mathbb{Y}_m^{A^i} \\ & & \ddots & & & \vdots \\ & & & -\boldsymbol{I}^{\Lambda} & & \frac{m-1}{m}\mathbb{R}_m^{A^i}\mathbb{Y}_m^{A^i} \\ \hline \boldsymbol{C}^B\dot{\boldsymbol{Y}}_{m1}^{B^i} & \boldsymbol{C}^B\dot{\boldsymbol{Y}}_{m2}^{B^i} & \cdots & \boldsymbol{C}^B\dot{\boldsymbol{Y}}_{m(m-1)}^{B^i} & & \boldsymbol{C}^B\dot{\boldsymbol{Y}}_{mm}^{B^i} + \boldsymbol{C}^A\dot{\boldsymbol{Y}}_m^{A^i} \end{array}\right] \tag{5.66}$$

where $\mathbb{R}_m^{A^i}\mathbb{Y}_m^{A^i} = \boldsymbol{I}^{\Lambda}$. Similarly, the interface right hand side (5.42):

$$\{-\mathsf{G}^{\Lambda^i} - \sum_{k=A,B}\mathsf{B}^{k^i}\Delta\mathsf{V}^{k^i}\} = \left[\begin{array}{c} -\mathbb{G}_1^{\Lambda^i} - \frac{1}{m}\mathbb{R}_m^{A^i}\ \Delta\mathbb{V}_m^{A^i} \\ -\mathbb{G}_2^{\Lambda^i} - \frac{2}{m}\mathbb{R}_m^{A^i}\ \Delta\mathbb{V}_m^{A^i} \\ \vdots \\ -\mathbb{G}_{m-1}^{\Lambda^i} - \frac{m-1}{m}\mathbb{R}_m^{A^i}\ \Delta\mathbb{V}_m^{A^i} \\ \hline -\mathbb{G}_m^{\Lambda^i} - \mathbb{B}^A\ \Delta\mathbb{V}_m^{A^i} - \mathbb{B}^B\ \Delta\mathbb{V}_m^{B^i} \end{array}\right] \tag{5.67}$$

This results in a simplified expression for (5.44) similar to the linear coupling method:

$$\Delta\boldsymbol{\Lambda}_j^i = \boldsymbol{S}_j^i + \left(\frac{j}{m}\right)\Delta\boldsymbol{\Lambda}_m^i \tag{5.68}$$

where

$$\begin{aligned} \boldsymbol{S}_j^i =& -\mathbb{G}_j^{\Lambda^i} - \left(\frac{j}{m}\right)\mathbb{R}_m^{A^i}\Delta\mathbb{V}_m^{A^i} \\ =& \boldsymbol{C}^A\left[\boldsymbol{P}_j^A - \left(1-\frac{j}{m}\right)\boldsymbol{P}_0^A - \left(\frac{j}{m}\right)\boldsymbol{P}_m^A\right] - \left[\boldsymbol{\Lambda}_j^i - \left(1-\frac{j}{m}\right)\boldsymbol{\Lambda}_0 - \left(\frac{j}{m}\right)\boldsymbol{\Lambda}_m^i\right] \\ & - \boldsymbol{C}^A\left(1-\frac{j}{m}\right)\left(\boldsymbol{M}^A\ddot{\boldsymbol{U}}_0^A + \boldsymbol{R}^A(\dot{\boldsymbol{U}}_0^A, \boldsymbol{U}_0^A) + \boldsymbol{C}^{A^T}\boldsymbol{\Lambda}_0^i - \boldsymbol{P}_0^A\right) \\ & - \boldsymbol{C}^A\left(\frac{j}{m}\right)\left(\boldsymbol{M}^A\ddot{\boldsymbol{U}}_m^{A^i} + \boldsymbol{R}^A(\dot{\boldsymbol{U}}_m^{A^i}, \boldsymbol{U}_m^{A^i}) + \boldsymbol{C}^{A^T}\boldsymbol{\Lambda}_m^i - \boldsymbol{P}_m^A\right) \\ & - \boldsymbol{C}^A\left(\frac{j}{m}\right)\left(\boldsymbol{M}^A\ \Delta\ddot{\boldsymbol{V}}_m^{A^i} + \boldsymbol{D}^A\ \Delta\dot{\boldsymbol{V}}_m^{A^i} + \boldsymbol{K}_j^{A^i}\ \Delta\boldsymbol{V}_m^{A^i}\right) \end{aligned} \tag{5.69}$$

The third term is identically zero and the last two terms cancel each other because of the relation (5.47). Thus one can express $\mathbf{\Lambda}_j^{i+1}$ as:

$$\begin{aligned}\mathbf{\Lambda}_j^{i+1} &= \mathbf{\Lambda}_j^i + \Delta\mathbf{\Lambda}_j^i = \mathbf{\Lambda}_j^i + \boldsymbol{S}_j^i + \left(\frac{j}{m}\right)\Delta\mathbf{\Lambda}_m^i \\ &= \boldsymbol{C}^A\left[\boldsymbol{P}_j^A - \left(1-\frac{j}{m}\right)\boldsymbol{P}_0^A - \left(\frac{j}{m}\right)\boldsymbol{P}_m^A\right] + \left[\left(1-\frac{j}{m}\right)\mathbf{\Lambda}_0 - \left(\frac{j}{m}\right)\mathbf{\Lambda}_m^{i+1}\right]\end{aligned} \tag{5.70}$$

If one assumes that the load $\boldsymbol{P}^A$ varies linearly with the time step ΔT then the first term in the expression above and expression (5.69) is zero. In addition, $\boldsymbol{S}_j^i = 0$ because the second term in (5.69) can be insured to be zero by appropriately choosing the assumed solution at iteration i for $\mathbf{\Lambda}_j^i$.

Note that for a fully consistent non-linear coupling method, if $\boldsymbol{S}_j^i$ needs to be transferred at a cross point between multiple subdomains, then it should be transferred to the subdomain with the smallest time step.

5.5 Results

As a first step, the current non-linear multi-time-step coupling method was verified by solving the linear problems presented in [95]. The linear problems needed just one Newton-Raphson iteration and the results were identical to those of the linear coupling method.

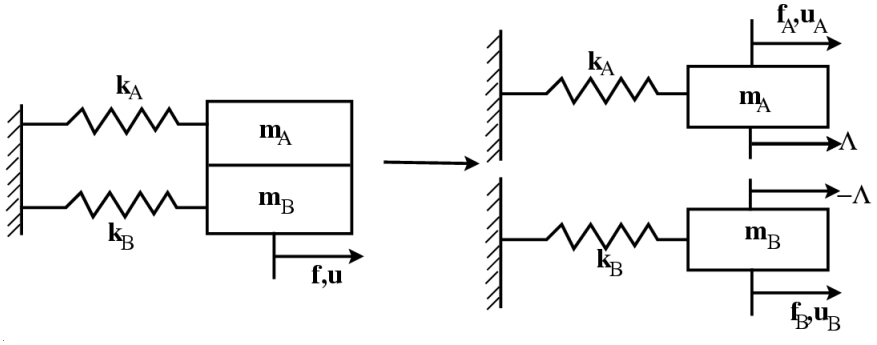

Figure 5.1: A split SDOF problem with non-linear springs.

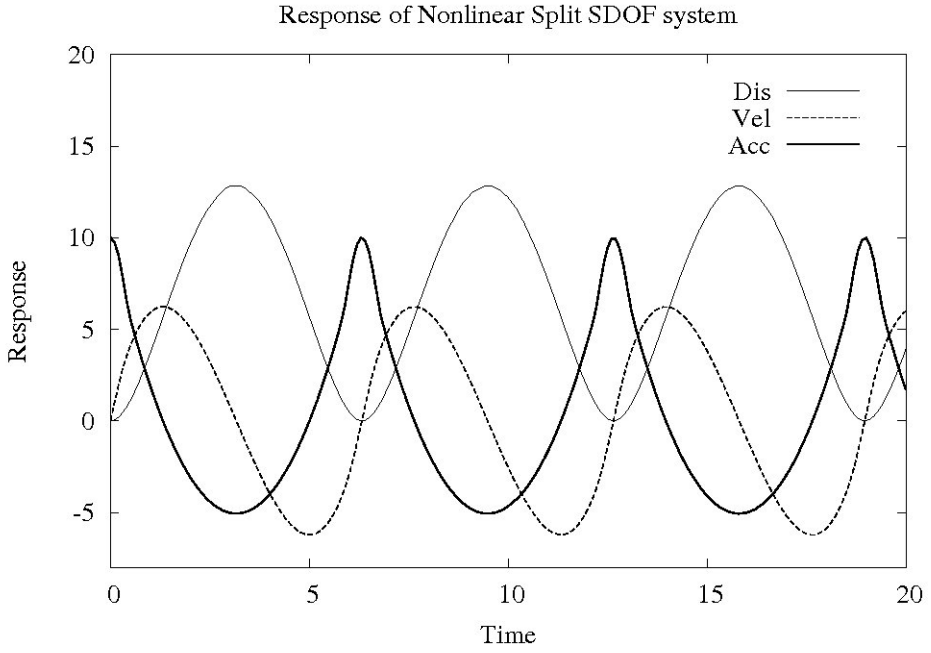

Figure 5.2: Non-linear split SDOF system under step load.

The simplest non-linear coupling problem is a split single degree of freedom (SDOF) system with non-linear springs as shown in figure 5.1. A number of non-linear elastic and inelastic springs were considered. Results from a particular system are shown in figures 5.2 and 5.3.

The two subsystem masses used were 1.5 and 0.5. Both masses were excited with step loads of 12 and 8 respectively. The non-linear function chosen to represent the spring force for both the springs can be expressed by a composite curve:

$$\begin{aligned} (\boldsymbol{U}^k \leq -u_y): \qquad & \boldsymbol{R}^k(\boldsymbol{U}^k) = -k_0 \sin(u_y) \sqrt{\frac{-\boldsymbol{U}^k}{u_y}} \\ \text{if} \quad (-u_y \leq \boldsymbol{U}^k \leq u_y): \qquad & \boldsymbol{R}^k(\boldsymbol{U}^k) = k_0 \sin(\boldsymbol{U}^k) \\ (\boldsymbol{U}^k \geq u_y): \qquad & \boldsymbol{R}^k(\boldsymbol{U}^k) = k_0 \sin(u_y) \sqrt{\frac{\boldsymbol{U}^k}{u_y}} \\ \text{where} \quad u_y < \frac{\pi}{2} & \end{aligned} \tag{5.71}$$

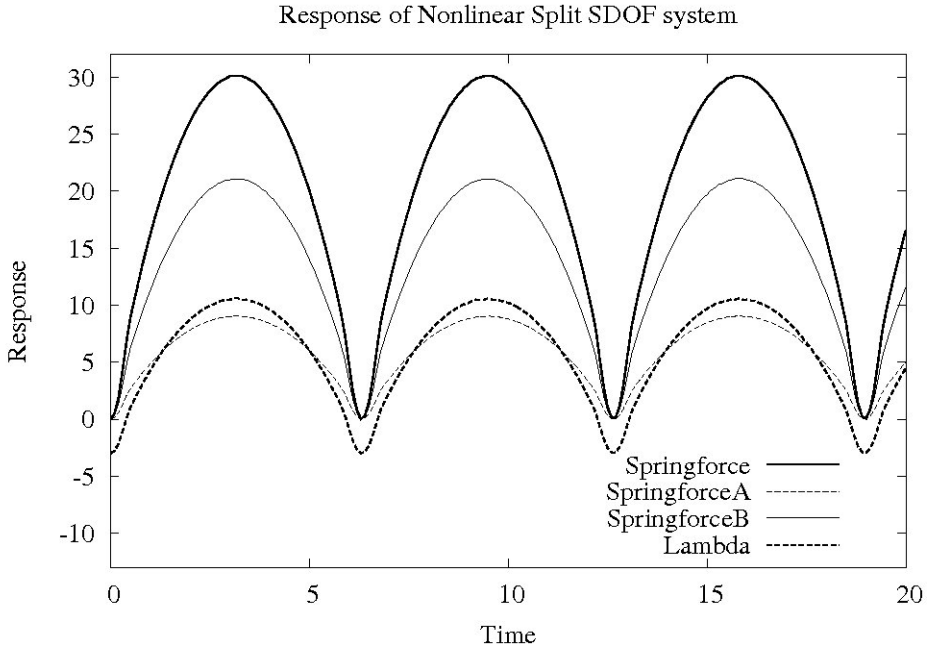

Figure 5.3: Spring forces and interface reaction for the non-linear split SDOF system under step load.

The values chosen for the material parameters (k_0; u_y) were (3; 1) for system A and (7; 1) for system B. The time steps for systems A and B were chosen as 0.5 and 0.1 respectively. Figure 5.2 shows the response of the corresponding undecomposed system. The response of the split system almost exactly matches that of the undecomposed system. The total and individual spring forces and the interface reactions between A and B are shown in figure 5.3. A variety of coupling cases such as implicit-explicit, implicit-implicit and explicit-explicit were considered and the results were found to be identical. These results suggest that the current method can also be expected to preserve the subsystem energies without dissipation or instability.

The response of a block of granular pavement material under shear is presented in figure 5.4. The specimen is a cube of size 0.4 m fixed at the bottom and loaded with an asymmetric shear force applied instantaneously on the top face. A non-linear elastic model presented in [98] was used to model the resilient elastic behavior of the granular material. The bottom

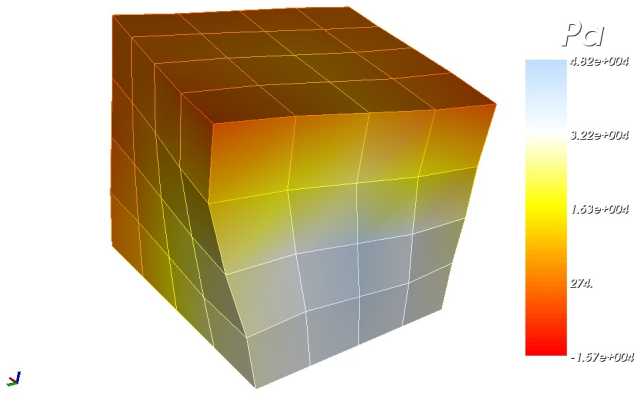

Figure 5.4: A block of pavement material under shear.

two layers are integrated implicitly and the top two layers are integrated explicitly with the same time step of 2×10^{-4} seconds. The results show seamless integration between the subdomains.

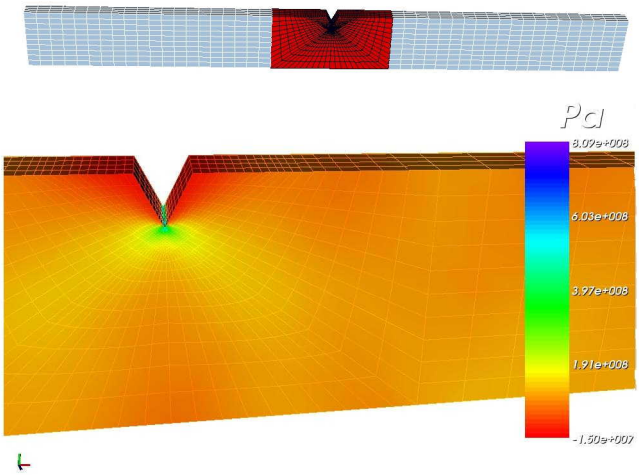

Figure 5.5: Response of a notched beam.

Next, a 3-D notched cantilever beam shown in figure 5.5 was considered. The beam was fixed on the left end and excited with a step axial surface pressure of 200 MPa on the right end. The dimensions of the beam were 2m $\times$ 0.2m $\times$ 0.05m. It was discretized with 5955 nodes and 4400 elements and the mesh was refined around the notch to capture the stress

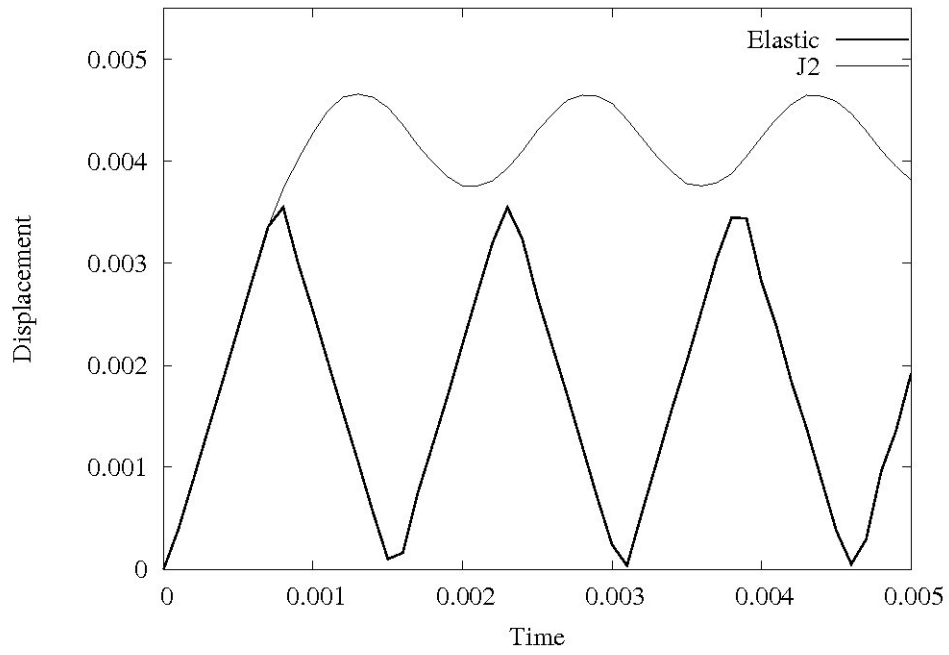

Figure 5.6: Horizontal displacement response of the free end of the notched beam.

singularity. The mesh was decomposed as shown in figure 5.5. The darkly shaded region was integrated explicitly with a time step of 1×10^{-6} while the rest of the mesh was integrated implicitly with a time step 1×10^{-5}. The problem was first solved using a linear elastic material with density $\rho = 7.8 \times 10^3$ kg/m^3, Young's modulus $E = 210$ GPa and Poisson's ratio $\nu = 0.33$ and then using a non-linear inelastic material model based on 'J2' Plasticity [7] with kinematic and isotropic hardening. The yield stress was chosen to be 260 MPa and plastic hardening modulus used was $H = 21$ GPa. The horizontal displacement and velocity response of the free end of the beam are plotted in figures 5.6 and 5.7. One may note that the inelastic beam deforms more because of the accumulated plastic strain. The initial amplitude of the vibration is also damped out by the inelastic beam after which it continues to oscillate within its modified elastic region. There is no discernible difference between the results obtained using the current multi-time-step approach and the traditional uniform time step approach.

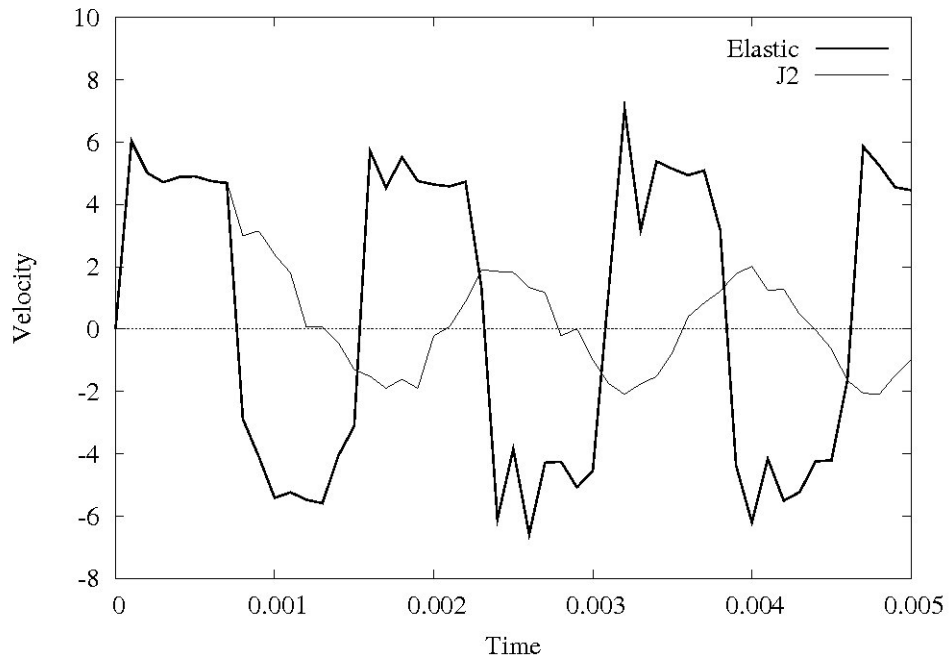

Figure 5.7: Horizontal velocity response of the free end of the notched beam.

5.6 Conclusion

An extension of the multi-time-step coupling method to non-linear problems has been presented. This method helps one to partition a large finite element mesh into smaller subdomains and solve them separately with different time integration schemes and/or time steps. In addition to being computationally faster than a uniform time-step approach, it is stable and energy preserving. The iterative PCG approach for solving the interface problem adds to the efficiency of this method.

The present method offers the greatest benefits when the problem under consideration is largely linear with possibly small regions exhibiting non-linear response. In such cases, one should decompose the mesh in such a manner that elements with similar time step requirements are grouped together. Subdomains that behave linearly should be integrated implicitly with large time steps and the non-linear subdomains should be integrated explicitly to maximize efficiency.

The present approach can also be extended to multiple subdomains following the recursive coupling approach presented in [97]. It is found that when there are more than two levels of time steps, only a recursive coupling approach is feasible and the traditional FETI approach using global interfaces is not practical.

Chapter 6

Recursive Coupling for Multiple Subdomains

A consistent coupling method for a finite element domain decomposed recursively into a hierarchy of subdomains is presented. The decomposition is achieved by dividing a given problem domain successively into two subdomains at a time until a hierarchy of subdomains is obtained. These subdomains are solved independently and the individual solutions are coupled together in accordance with the hierarchy to obtain the global solution. This enables one to integrate each subdomain of the global mesh with an appropriate time-step and/or time stepping scheme. The present approach is an extension of a method developed for linear systems with two subdomains (Ref. [95]) to systems with multiple subdomains. The coupling is achieved through Lagrange multipliers by imposing continuity of velocity across the interfaces between subdomains. The method exactly preserves the total system energy and inherits the stability and accuracy characteristics of its component subdomains. It also exhibits seamless integration between the subdomains with minimal interface computation. Results from several benchmark problems verify these properties and are used to compare its performance with respect to other coupling methods.

6.1 Introduction

Over the last decade, domain decomposition methods [96] such as the finite element tearing and interconnecting (FETI) method [81] have been successfully employed to solve a wide variety of problems in statics and dynamics. For instance, 2^{nd} order PDEs arising from 2-D and 3-D continuum elements [89] and 4^{th} order PDEs arising from structural

plate and shell elements [86] have been investigated. Traditionally the FETI method is implemented using an iterative preconditioned conjugate projected gradient (PCPG) method [81] to compute the Lagrange multipliers on the interface. Variations of the FETI method have been proposed to effectively handle *cross points* on the interface. Cross points are nodes that are shared by three or more subdomains and their presence can lead to a singular interface problem even though it is still consistently solvable. The basic FETI method using PCPG incorporates this redundancy for better global convergence of the solution. An advancement of the basic FETI method, the FETI-DP (dual-primal) method [88] equates the degrees of freedom (dofs) at the cross points to global dofs as opposed to equating them to each other. Hence, the resulting interface problem relates the dual Lagrange multipliers to the primal global dofs. One can then eliminate the primal variables at the cross points through a *coarse* problem and solve for the Lagrange multipliers as usual with the regular PCPG method.

We present a recursive domain decomposition based implementation of the FETI method that circumvents the problematic cross points. The current method for is formulated for structural dynamics problems where multiple subdomains are integrated with the same time-step but with possibly different Newmark schemes. The structural dynamics problem is chosen so that the individual subdomain problems are non-singular and this choice does not restrict the applicability of the current method in any way. It can also be applied to problems in statics using the numerical techniques developed for the conventional FETI method for *floating* subdomains. The current method is also a precursor to a more general approach for solving problems where different subdomains can be integrated with different time-steps.

The recursive domain decomposition approach has not been used as commonly in the computational mechanics community as in the computer science community. However, some relevant works which address issues similar to those that arise in the present method are cited.

Hackbusch [99] presented one of the first approaches using nested multi-level grids for

elliptic Eigen-problems. Bennighof, Lehoucq and co-workers [100, 101] have developed an automated multi-level substructuring methods for Eigen-problems in elastodynamics. Papalukopoulos and Natsiavas [102] have presented a multi-level substructuring method for non-linear dynamic analysis of very-large scale mechanical systems.

Saad and co-workers [103, 104] have developed algebraic multi-level solvers and preconditioners based on recursive domain-based partitioning for incomplete LU factorization of sparse matrix systems. Le Borne [105] has studied approximate matrix inversion methods using a hierarchical decomposition of the global system matrix.

Dubinski [106] presented a parallel tree implementation of an N-body problem using recursive bisection. Marzouk and Ghoniem [107] have discussed a parallel implementation of a hierarchical clustering method for studying N-body problems with applications to fluid dynamics, gravitational astrophysics and molecular dynamics.

Yang and Hsieh [108] have developed an iterative approach for parallel implementation of non-linear dynamic finite element analysis using direct substructuring. Griebel and Zumbusch [109] have surveyed current methods in the literature for parallelization, multi-level iterative solvers and domain decomposition.

6.2 FETI for Dynamics

The semi-discrete equations governing the dynamics of a structural domain Ω decomposed into S subdomains $\Omega^k, 1 \leq k \leq S$, are given by:

$$\boldsymbol{M}^k \ddot{\boldsymbol{U}}^k + \boldsymbol{K}^k \boldsymbol{U}^k + \boldsymbol{C}^{k^{\mathrm{T}}} \boldsymbol{\Lambda} = \boldsymbol{P}^k \qquad \forall k : 1 \leq k \leq S \tag{6.1}$$

$$\sum_{k=1}^{S} \boldsymbol{C}^k \dot{\boldsymbol{U}}^k = \boldsymbol{0} \tag{6.2}$$

where $\boldsymbol{M}^k$, $\boldsymbol{K}^k$, $\boldsymbol{U}^k$ and $\boldsymbol{P}^k$ denote the mass matrix, stiffness matrix, displacement vector and the load vector respectively. $\boldsymbol{C}^k$ is the boolean connectivity matrix and $\boldsymbol{\Lambda}$ is the vector

of Lagrange multipliers associated with the FETI decomposition. Note that equation (4.21) represents the constraint of continuity of velocities between the subdomains. These equations can be integrated in time using the Newmark time stepping schemes. The fully discretized equations for advancing all the subdomains from time step t_n to t_{n+1} can be expressed in *block matrix form* [95] as:

$$\mathbb{M}^k\ \mathbb{U}^k_{n+1} + \mathbb{C}^k\ \mathbf{\Lambda}_{n+1} = \mathbb{P}^k_{n+1} - \mathbb{N}^k\ \mathbb{U}^k_n \qquad \forall k : 1 \le k \le S \tag{6.3}$$

$$\sum_{k=1}^{S} \mathbb{B}^k \mathbb{U}^k_{n+1} = \mathbf{0} \tag{6.4}$$

where

$$\begin{gathered}
\mathbb{U}^k_n = \begin{bmatrix} \ddot{\boldsymbol{U}}^k_n \\ \dot{\boldsymbol{U}}^k_n \\ \boldsymbol{U}^k_n \end{bmatrix} \quad ; \quad \mathbb{P}^k_{n+1} = \begin{bmatrix} \boldsymbol{P}^k_{n+1} \\ 0 \\ 0 \end{bmatrix} \quad ; \quad \mathbb{C}^k = \begin{bmatrix} \boldsymbol{C}^{k^T} \\ 0 \\ 0 \end{bmatrix} \quad ; \quad \mathbb{B}^k = \begin{bmatrix} 0 & \boldsymbol{C}^k & 0 \end{bmatrix} \\
\mathbb{M}^k = \begin{bmatrix} \boldsymbol{M}^k & 0 & \boldsymbol{K}^k \\ -\gamma^k \Delta t \boldsymbol{I}^k & \boldsymbol{I}^k & 0 \\ -\beta^k \Delta t^2 \boldsymbol{I}^k & 0 & \boldsymbol{I}^k \end{bmatrix} \quad ; \quad \mathbb{N}^k = \begin{bmatrix} 0 & 0 & 0 \\ -\Delta t(1-\gamma^k)\boldsymbol{I}^k & -\boldsymbol{I}^k & 0 \\ -\Delta t^2(\frac{1}{2}-\beta^k)\boldsymbol{I}^k & -\Delta t \boldsymbol{I}^k & -\boldsymbol{I}^k \end{bmatrix}
\end{gathered} \tag{6.5}$$

γ^k and β^k are the Newmark parameters, $\Delta t = t_{n+1} - t_n$ is the time step and $\boldsymbol{I}^k$ is the subdomain identity matrix. Equations (6.3) and (6.4) can be combined and expressed in a global matrix as:

$$\left[\begin{array}{cccc|c} \mathbb{M}^1 & & & & \mathbb{C}^1 \\ & \mathbb{M}^2 & & & \mathbb{C}^2 \\ & & \ddots & & \vdots \\ & & & \mathbb{M}^S & \mathbb{C}^S \\ \hline \mathbb{B}^1 & \mathbb{B}^2 & \cdots & \mathbb{B}^S & \mathbf{0} \end{array}\right] \left[\begin{array}{c} \mathbb{U}^1_{n+1} \\ \mathbb{U}^2_{n+1} \\ \vdots \\ \mathbb{U}^S_{n+1} \\ \hline \mathbf{\Lambda}_{n+1} \end{array}\right] = \left[\begin{array}{c} \mathbb{P}^1_{n+1} - \mathbb{N}^1\ \mathbb{U}^1_n \\ \mathbb{P}^2_{n+1} - \mathbb{N}^2\ \mathbb{U}^2_n \\ \vdots \\ \mathbb{P}^S_{n+1} - \mathbb{N}^S\ \mathbb{U}^S_n \\ \hline \mathbf{0} \end{array}\right] \tag{6.6}$$

The above system can be solved efficiently using the bordered system solution procedure [95]. One may verify that the solution is given by:

$$\mathbb{U}^k_{n+1} = \mathbb{V}^k_{n+1} + \mathbb{W}^k_{n+1} \tag{6.7}$$

where
$$\mathbb{M}^k \mathbb{V}^k_{n+1} = \mathbb{P}^k_{n+1} - \mathbb{N}^k \mathbb{U}^k_{n+1} \tag{6.8}$$

$$\mathbb{M}^k \mathbb{Y}^k = \mathbb{C}^k \quad \text{and} \quad \mathbb{W}^k_{n+1} = -\mathbb{Y}^k \mathbf{\Lambda}_{n+1} \tag{6.9}$$

The interface reactions $\mathbf{\Lambda}_{n+1}$ can be computed from:

$$\boldsymbol{H}\, \mathbf{\Lambda}_{n+1} = \boldsymbol{f} \tag{6.10}$$

where
$$\boldsymbol{H} \equiv \sum_{k=1}^{S} \mathbb{B}^k \mathbb{Y}^k = \sum_{k=1}^{S} \gamma^k \Delta t\, \boldsymbol{C}^k \left[\tilde{\boldsymbol{M}}^k\right]^{-1} \boldsymbol{C}^{k^T} \tag{6.11}$$

$$\boldsymbol{f} \equiv \sum_{k=1}^{S} \mathbb{B}^k \mathbb{V}^k_{n+1} = \sum_{k=1}^{S} \boldsymbol{C}^k \dot{\boldsymbol{V}}^k_{n+1} \tag{6.12}$$

and $\tilde{\boldsymbol{M}}^k \equiv \boldsymbol{M}^k + \beta^k \Delta t^2 \boldsymbol{K}^k$ is the effective system matrix usually obtained in dynamics. Note that the computation of $\mathbb{V}^k_{n+1}$ from equation (6.8) is done separately and independently for each subdomain by simply advancing the previous solution $\mathbb{U}^k_n$ by one time step. Similarly, $\mathbb{Y}^k$ matrices can also be computed independently for each subdomain by solving for them one column at a time against the columns of $\mathbb{C}^k$. Also note that solving against the columns of $\mathbb{C}^k$ simply amounts to applying a unit (positive or negative) load, one at a time, to each degree of freedom on the interface of Ω^k and computing its response for one time step from 'at rest' initial conditions. If $\mathbb{C}^k$ matrices include redundancies from the cross points, then the interface matrix $\boldsymbol{H}$ becomes singular, though still consistently solvable with $\boldsymbol{f}$.

The solution at each time step is obtained through a three step process:

1. Solve all the subdomains independently for $\mathbb{V}^k_{n+1}$ from (6.8).

2. Solve for the Lagrange multipliers from $\boldsymbol{H}\mathbf{\Lambda}_{n+1} = \boldsymbol{f}$.

3. Update all the subdomains: $\mathbb{U}^k_{n+1} = \mathbb{V}^k_{n+1} + \mathbb{W}^k_{n+1}$ where $\mathbb{W}^k_{n+1} = -\mathbb{Y}^k \boldsymbol{\Lambda}_{n+1}$.

FETI, implemented with direct solvers, uses the $\mathbb{Y}^k$ matrices which depend upon the subdomain matrices $\tilde{\boldsymbol{M}}^k$. For linear problems, these need to be computed only once initially. One should also form and factorize the interface matrix $\boldsymbol{H}$ as an initial computation to increase efficiency of interface solve in step 2. However, for large problems, this interface problem itself can be quite large, making direct solvers unsuitable.

FETI with the PCPG method, in contrast, solves the interface problem iteratively and does not need to compute the $\mathbb{Y}^k$ matrices. Matrix-vector products of the type $\boldsymbol{H}\boldsymbol{\Lambda}_{n+1}$ are computed subdomain-wise for better efficiency. The disadvantage in this case is that, in the presence of cross points, there is a rapid loss of orthogonality between the search vectors of the PCPG method.

6.3 Recursive Implementation of FETI

In this section we will illustrate a hierarchical implementation of the FETI method and compare its computational cost with the traditional FETI method implemented using direct solvers. Given a finite element mesh partitioned into S subdomains, a hierarchy of subdomains can be built by combining two subdomains at a time until the original undecomposed mesh is recreated. Such a hierarchy can be effectively represented by the *tree* structure as shown in figure 6.1. The *leaf* nodes in the *tree* represent the subdomains that are not further subdivided. The original undecomposed structure Ω is called the *root* node. Let $\Omega_{(A,B)}$ be a subdomain created by coupling two subdomains Ω_A and Ω_B. In general, A and B can either be subdomain numbers of leaf nodes or *nested* pairs of subdomain numbers representing coupled nodes. The node $\Omega_{(A,B)}$ is called the *parent* node of its left and right *daughter* nodes Ω_A and Ω_B, respectively.

Note that we always couple two subdomains at a time and so the *cross points* are never created within a single coupling level. In addition, the size of the interface between any two

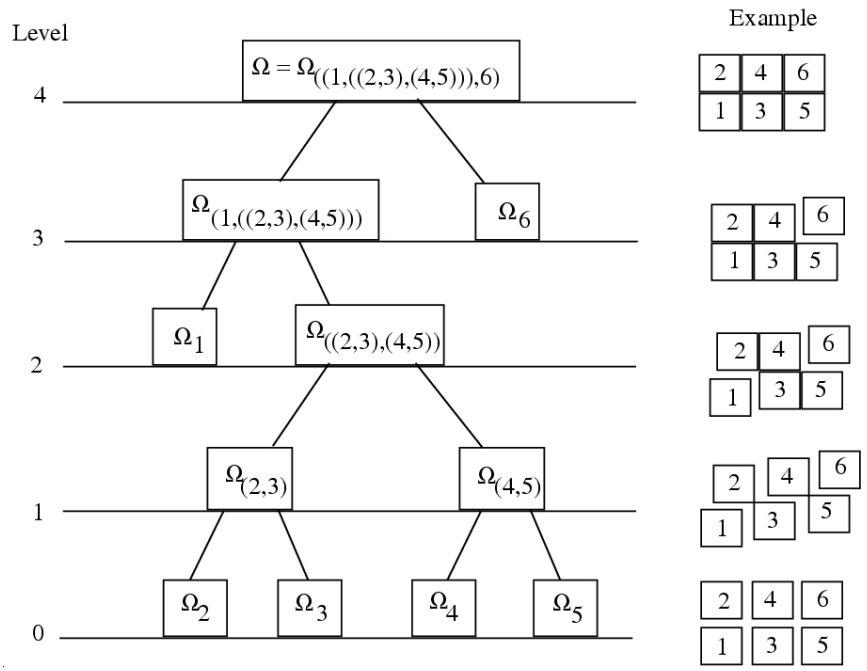

Figure 6.1: A hierarchy of subdomains.

subdomains is always much smaller compared to the entire interface Γ_b. This makes the FETI interface problem (6.10) more suitable for direct solvers.

The computations involved with solving a structure in this hierarchical manner can be laid out as follows. At any time step, the global problem to be solved is:

$$\mathbb{M}\,\mathbb{U}_{n+1} = \mathbb{P}_{n+1} - \mathbb{N}\,\mathbb{U}_n \tag{6.13}$$

where all the matrices are associated with the global undecomposed mesh. Since the coupling described in the previous section preserves the total system energy exactly [95], solution to any subdomain in the tree can be replaced by the coupled solution of its daughter subdomains. Thus, the global problem (6.13) can be replaced by a multi-level coupled problem by successively decomposing it into two sub-problems according to the hierarchy shown in

figure 6.1. In general, for any node $\Omega_{(A,B)}$ in the tree, the problem:

$$\mathbb{M}^{(A,B)}\,\mathbb{U}_{n+1}^{(A,B)} = \mathbb{P}_{n+1}^{(A,B)} - \mathbb{N}^{(A,B)}\,\mathbb{U}_{n}^{(A,B)} \tag{6.14}$$

is substituted with two coupled sub-problems for Ω_A and Ω_B:

$$\left[\begin{array}{cc|c} \mathbb{M}^A & & \mathbb{C}_A^{(A,B)} \\ & \mathbb{M}^B & \mathbb{C}_B^{(A,B)} \\ \hline \mathbb{B}_A^{(A,B)} & \mathbb{B}_B^{(A,B)} & \mathbf{0} \end{array}\right] \left[\begin{array}{c} \mathbb{U}_{n+1}^A \\ \mathbb{U}_{n+1}^B \\ \hline \mathbf{\Lambda}_{n+1}^{(A,B)} \end{array}\right] = \left[\begin{array}{c} \mathbb{P}_{n+1}^A - \mathbb{N}^A\mathbb{U}_n^A \\ \mathbb{P}_{n+1}^B - \mathbb{N}^B\mathbb{U}_n^B \\ \hline \mathbf{0} \end{array}\right] \tag{6.15}$$

similar to equation (6.6). Note that the matrices $[\mathbb{C}_A^{(A,B)}, \mathbb{B}_A^{(A,B)}]$ and $[\mathbb{C}_B^{(A,B)}, \mathbb{B}_B^{(A,B)}]$ are interface matrices associated with the subdomains Ω_A and Ω_B respectively for the interface $\Gamma_b^{(A,B)}$ between them exclusively. The Lagrange multiplier $\mathbf{\Lambda}_{n+1}^{(A,B)}$ is also associated only with the interface $\Gamma_b^{(A,B)}$. In addition to the hierarchy of subdomains, the procedure described above also creates a hierarchy of interfaces such that the global interface Γ_b is the union of all the individual interfaces between subdomains. The system (6.15) can be solved using the procedure described in the previous section. One must, however, order the computations carefully to account for the possibility that each of the subproblems above may be further subdivided.

Computation of Y Matrices

The solution of $\mathsf{Y} = \mathsf{M}^{-1}\mathsf{C}$ can be carried out subdomain-wise. Further we note that one does not explicitly need to compute the inverse of the M matrix. The matrix Y can be built one column at a time by solving it against the columns of C:

$$\mathbb{M}^k\mathbb{Y}_k^{(A,B)} = \mathbb{C}_k^{(A,B)} \qquad \text{for} \quad k = A, B \tag{6.16}$$

$$\left.\begin{aligned} \tilde{\boldsymbol{M}}^k\,\ddot{\boldsymbol{Y}}_k^{(A,B)} &= \boldsymbol{C}_k^{(A,B)^T} \\ \Rightarrow \quad \dot{\boldsymbol{Y}}_k^{(A,B)} &= \gamma^k \Delta t\,\ddot{\boldsymbol{Y}}_k^{(A,B)} \\ \boldsymbol{Y}_k^{(A,B)} &= \beta^k \Delta t^2\,\ddot{\boldsymbol{Y}}_k^{(A,B)} \end{aligned}\right| \quad \text{for} \quad k = A, B \tag{6.17}$$

Also note that solving against the columns of $\boldsymbol{C}_k^{(A,B)^T}$ simply amounts to applying a unit load, one at a time, to each dof on the interface $\Gamma_b^{(A,B)}$ and computing the response of subdomain Ω_k for a single time step from at rest initial conditions. This can easily be done with conventional dynamics codes since the subdomain matrices would already be formed and factorized.

Once the Y matrices have been obtained, the interface matrix $\boldsymbol{H}^{(A,B)}$ can be formed as:

$$\begin{aligned} \boldsymbol{H}^{(A,B)} = \mathsf{B}\,\mathsf{Y} &= \begin{bmatrix} \mathbb{B}_A^{(A,B)} & \mathbb{B}_A^{(A,B)} \end{bmatrix} \begin{bmatrix} \mathbb{Y}_A^{(A,B)} \\ \mathbb{Y}_B^{(A,B)} \end{bmatrix} \\ \Rightarrow \quad \boldsymbol{H}^{(A,B)} &= \boldsymbol{C}_A^{(A,B)}\,\dot{\boldsymbol{Y}}_A^{(A,B)} + \boldsymbol{C}_B^{(A,B)}\,\dot{\boldsymbol{Y}}_B^{(A,B)} \end{aligned} \tag{6.18}$$

which is exactly the same result as in (6.11) if only two subdomains Ω_A and Ω_B are to be coupled.

Solution and Coupling of Subdomains

The independent responses $\mathsf{V} = \mathsf{M}^{-1}\mathsf{P}$ for both the subdomains can also be computed subdomain-wise:

$$\mathbb{M}^k \mathbb{V}_{n+1}^k = \mathbb{P}_{n+1}^k - \mathbb{N}^k \mathbb{U}_n^k \qquad \text{for} \quad k = A, B \tag{6.19}$$

$$\left.\begin{aligned} \tilde{\boldsymbol{M}}^k\,\ddot{\boldsymbol{V}}_{n+1}^k &= \boldsymbol{P}_{n+1}^k - \boldsymbol{K}^k \hat{\boldsymbol{U}}_{n+1}^k \\ \Rightarrow \quad \dot{\boldsymbol{V}}_{n+1}^k &= \hat{\dot{\boldsymbol{V}}}_{n+1}^k + \gamma^k \Delta t\,\ddot{\boldsymbol{V}}_{n+1}^k \\ \boldsymbol{V}_{n+1}^k &= \hat{\boldsymbol{V}}_{n+1}^k + \beta^k \Delta t^2\,\ddot{\boldsymbol{V}}_{n+1}^k \end{aligned}\right| \quad \text{for } k = A, B \tag{6.20}$$

The Lagrange multipliers are then solved from:

$$\boldsymbol{H}^{(A,B)}\,\boldsymbol{\Lambda}_{n+1}^{(A,B)} = \mathsf{B}\,\mathsf{V} = \begin{bmatrix} \mathbb{B}_A^{(A,B)} & \mathbb{B}_A^{(A,B)} \end{bmatrix} \begin{bmatrix} \mathbb{V}_{n+1}^{A} \\ \mathbb{V}_{n+1}^{B} \end{bmatrix} \tag{6.21}$$

$$\Rightarrow \quad \boldsymbol{H}^{(A,B)}\,\boldsymbol{\Lambda}_{n+1}^{(A,B)} = \{\boldsymbol{C}_A^{(A,B)}\,\dot{\boldsymbol{V}}_{n+1}^{A} + \boldsymbol{C}_B^{(A,B)}\,\dot{\boldsymbol{V}}_{n+1}^{B}\}$$

and the subdomain solutions updated using:

$$\begin{bmatrix} \ddot{\boldsymbol{U}}_{n+1}^{k} \\ \dot{\boldsymbol{U}}_{n+1}^{k} \\ \boldsymbol{U}_{n+1}^{k} \end{bmatrix} = \begin{bmatrix} \ddot{\boldsymbol{V}}_{n+1}^{k} \\ \dot{\boldsymbol{V}}_{n+1}^{k} \\ \boldsymbol{V}_{n+1}^{k} \end{bmatrix} - \begin{bmatrix} \ddot{\boldsymbol{Y}}_{k}^{(A,B)} \\ \dot{\boldsymbol{Y}}_{k}^{(A,B)} \\ \boldsymbol{Y}_{k}^{(A,B)} \end{bmatrix} \boldsymbol{\Lambda}_{n+1}^{(A,B)} \qquad \text{for} \quad k = A, B \tag{6.22}$$

Starting from the root node in the tree and moving down through various levels of coupling in the tree, the solution is advanced through one time step once the entire tree has been traversed.

6.3.1 Solving a General Subdomain in the Hierarchy

A non-iterative direct solution procedure for the interface problems at various levels of the tree are considered here. The procedure for solving a general node $\Omega_{(A,B)}$ in the tree is given by the *self-recursive* subroutine TreeSolve presented in figure 6.2. $\Omega_{(A,B)}$, $\mathbb{P}^{(A,B)}$ and $\mathbb{U}_n^{(A,B)}$ are inputs and $\mathbb{U}_{n+1}^{(A,B)}$ is the output.

Remarks

1. The entire tree can be solved with one call: TreeSolve(Ω, $\mathbb{P}$, $\mathbb{U}_n$, $\mathbb{U}_{n+1}$) for a given load $\mathbb{P}$ and initial conditions $\mathbb{U}_n$.

2. Only the leaf nodes are actually solved for their response to external forces using conventional dynamics codes and it is these responses that are coupled in a hierarchical fashion.

```
SUB TreeSolve(Ω_(A,B), ℙ^(A,B), 𝕌_n^(A,B), 𝕌_{n+1}^(A,B))
   If Ω_(A,B) is a Leaf Node then
      Solve Ω_(A,B)
   Else
      Call TreeSolve(Ω_A, ℙ^A, 𝕌_n^A, 𝕍_{n+1}^A)
      Call TreeSolve(Ω_B, ℙ^B, 𝕌_n^B, 𝕍_{n+1}^B)
      Compute Λ_{n+1}^(A,B) from H^(A,B) Λ_{n+1}^(A,B) = f^(A,B)
      Update 𝕌_{n+1}^A = 𝕍_{n+1}^A − 𝕐_A^(A,B) Λ_{n+1}^(A,B)
      Update 𝕌_{n+1}^B = 𝕍_{n+1}^B − 𝕐_B^(A,B) Λ_{n+1}^(A,B)
   End if
END SUB TreeSolve
```

Figure 6.2: Subroutine to solve a general node $\Omega_{(A,B)}$ in the tree.

3. The coupling is not computationally expensive since $\mathbf{\Lambda}_{n+1}^{(A,B)}$ is computed by forward-reduction and back-substitution with a pre-factorized $\boldsymbol{H}^{(A,B)}$ and the update is a simple matrix-vector multiplication operation.

4. The solutions for any node in the tree can be stored by overwriting the solutions of its component subdomains. No extra storage is required for storing the response of subdomains higher in the tree.

6.3.2 Initial Computation of Subdomain Matrices

Initial computation of the $\mathbb{Y}_A^{(A,B)}$ and $\mathbb{Y}_B^{(A,B)}$ matrices can also be done using a self-recursive subroutine BuildY presented in figure 6.3. Input to the subroutine is the node $\Omega_{(A,B)}$ and output consists of the matrices $\mathbb{Y}_A^{(A,B)}$ and $\mathbb{Y}_B^{(A,B)}$ for coupling its two daughter nodes Ω_A and Ω_B.

Remarks

1. The initial call to build the $\mathbb{Y}$ matrices for all the subdomains is: BuildY(Ω, $\mathbb{Y}^{\Omega}_{\mathsf{LeftChild}(\Omega)}$, $\mathbb{Y}^{\Omega}_{\mathsf{RightChild}(\Omega)}$).

2. Note that the subroutine BuildY makes calls to TreeSolve which in turn needs the $\mathbb{Y}$

```
SUB BuildY(Ω_(A,B), 𝕐_A^(A,B), 𝕐_B^(A,B))
  Do for K = A, B
    If Ω_K is NOT a Leaf Node :
      Call BuildY(Ω_K, 𝕐^K_LeftChild(K), 𝕐^K_RightChild(K))
    Do for each dof J in Γ_b^(A,B)
      If K = A : ℙ^K = +1 load on dof J in Ω_K
      If K = B : ℙ^K = −1 load on dof J in Ω_K
      Call TreeSolve(Ω_K, ℙ^K, 0, column J of 𝕐_K^(A,B))
    End do
  End Do
END SUB BuildY
```

Figure 6.3: Subroutine to build the $\mathbb{Y}$ matrices for a general node.

matrices to have already been built. This does not pose a problem because $\mathbb{Y}$ matrices are built starting from the lowest level in the tree proceeding up. The recursive BuildY subroutine ensures that all the $\mathbb{Y}$ matrices below a particular level are available before calling TreeSolve at that level.

3. The convention adopted for $\mathbb{C}_A^{(A,B)}$ and $\mathbb{C}_B^{(A,B)}$ matrices is that the former contains only zeros and positive ones and the latter contains only zeros and negative ones.

4. The $\mathbb{Y}$ matrices for the entire tree can be stored subdomain-wise. One needs to allocate space for these matrices only at the leaf subdomain level. After all the matrices have been built, every leaf subdomain will contain $\mathbb{Y}$ matrices for as many levels of coupling as its depth in the tree.

5. The $\boldsymbol{H}^{(A,B)}$ matrix at any coupling level can be computed as $\boldsymbol{H}^{(A,B)} = \boldsymbol{C}_A^{(A,B)}\,\dot{\boldsymbol{Y}}_A^{(A,B)} + \boldsymbol{C}_B^{(A,B)}\,\dot{\boldsymbol{Y}}_B^{(A,B)}$.

The final algorithm for solving a linear dynamics problem is summarized in figure 6.4. Note that the initial acceleration calculation is also coupled between subdomains. Thus, it has to be carried out either on the original undecomposed mesh or on the decomposed mesh by imposing continuity of initial accelerations.

1. Compute initial acceleration on the global mesh.
2. Construct the hierarchy of subdomains and build their interface matrices.
3. Form the individual subdomain system matrices.
4. Build the Y matrices for all subdomains.
5. Form and factorize the $\boldsymbol{H}$ matrices for each coupling level.
6. Do for $n = 1$ to number of time steps
   (a) Call $\mathsf{TreeSolve}(\Omega, \mathbb{P}, \mathbb{U}_n, \mathbb{U}_{n+1})$.
   (b) Output results.

   End do
7. Finish

Figure 6.4: Algorithm for the hierarchical FETI implementation.

6.4 Comparison of Computational Cost

We will consider the six-subdomain example presented in figure 6.1 for the present comparison. Note that the global interface Γ_b is a union of all the interfaces at each coupling level. The interfaces and their sizes are summarized as follows:

Coupling Level	Interface Names	Interface size
0	-	$n_0 = 0$
1	$\Gamma_b^{(2,3)}$ & $\Gamma_b^{(4,5)}$	$n_1 = 1 + 1$
2	$\Gamma_b^{((2,3),(4,5))}$	$n_2 = 3n$
3	$\Gamma_b^{(1,((2,3),(4,5)))}$	$n_3 = 2n$
4	$\Gamma_b^{((1,((2,3),(4,5))),6)}$	$n_4 = 2n$

It has been assumed that the subdomains are almost equal in size and that the interface size between two adjacent subdomains is n. The total interface size is $7n$ which is the same as that for the conventional FETI method. In order to compare computational costs, one may

note that the present hierarchical implementation differs with the conventional approach on precisely two counts: (a) the multi-level interface solves and (b) the generation of multi-level $\mathbb{Y}$ matrices.

The cost of an interface solve at each time step for the conventional FETI method implemented using direct solvers is $\propto (7n)^2$: which is the cost of forward-reduction and back-substitution on the entire interface problem. The total cost of building all the $\mathbb{Y}$ matrices is the same as the cost for solving all the subdomains once for each degree of freedom on the interface: $2 \times 7n$ subdomain solves.

For the hierarchical implementation, the interfaces are divided between different coupling levels. In this case, the cost of an interface solve is the sum of the costs for all the individual interfaces: $\propto (n_4)^2 + (n_3)^2 + (n_2)^2 + (n_1)^2$. The cost of computing the $\mathbb{Y}$ matrix for particular subdomain in the tree is equal to the cost of solving its corresponding subtree once for each dof on the interface at the coupling level in question. For instance, in the present case, the total cost of computing $\mathbb{Y}$ matrices at all coupling levels, in addition to the $2 \times 7n$ subdomain solves, is:

- Level 4 : $\propto [(n_3)^2 + (n_2)^2 + (n_1)^2] + 5$ subdomain updates
- Level 3 : $\propto [(n_2)^2 + (n_1)^2] + 4$ subdomain updates
- Level 2 : $\propto [(n_1)^2] + 4$ subdomain updates
- Level 1 : $\propto [(n_0)^2] = 0$

The costs in the square brackets are for solving interface problems lower in the hierarchy and the subdomain updates are simple matrix-vector multiplications of the form $-\mathbb{Y}\mathbf{\Lambda}$.

The above discussion shows that although the initial computational cost of $\mathbb{Y}$ matrices is slightly greater for the hierarchical method than that for the conventional approach, the per time step cost of interface solves is smaller because $(n_4)^2 + (n_3)^2 + (n_2)^2 + (n_1)^2 \leq (7n)^2$ since $n_4 + n_3 + n_2 + n_1 = 7n$. This would translate into a big advantage for long time dynamic simulations where the problem needs to be run for a large number of time steps.

6.5 Effect of Tree Topology

The computational costs associated with this approach may actually depend upon the hierarchy of decomposition of the global structure i.e. topology of the tree. We will consider a simple square domain shown in figure 6.5 for comparing computational costs of the Hierarchical FETI method with the traditional PCG method. The domain is decomposed into $n \times n = n^2$ subdomains of equal sizes and each subdomain side contains l nodes. The total interface contains $(n-1)nl$ nodes and each subdomain contains l^2 nodes. Let $N \equiv n^2$ and $L \equiv l^2$.

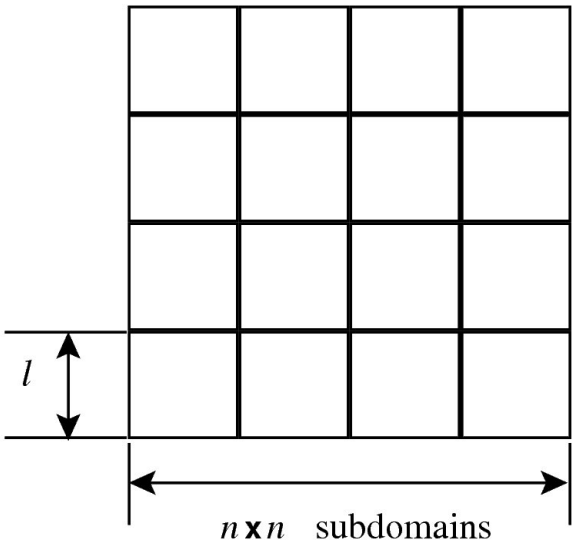

Figure 6.5: An example problem domain.

Case (i) Full Tree

The subdomains are coupled as shown in figure 6.6 where the thickness of the interface lines represents their coupling sequence. The thinnest interfaces are coupled first and the thickest interfaces are coupled last. The associated tree is fully populated and has the minimum number of levels possible. The total number of levels in the tree is D where $D = \log_2 N$ and $N = 2^D$.

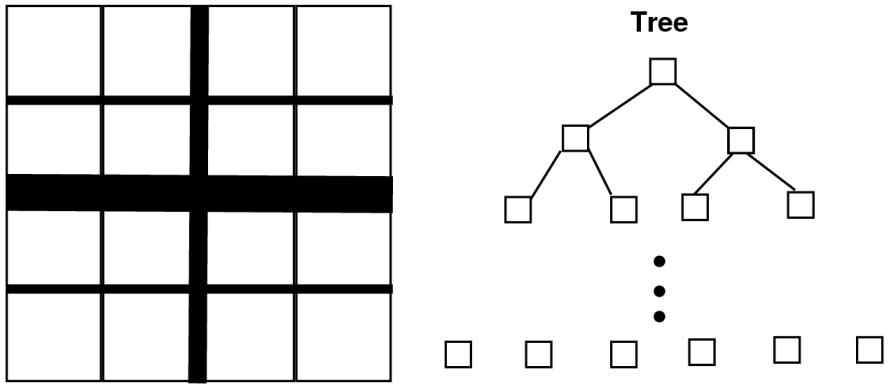

Figure 6.6: Full tree.

Case (ii) Sparse Tree

The subdomains are coupled by adding them successively one at a time to a growing subdomain as shown in figure 6.7. The associated tree is minimally populated and has the maximum number of levels possible. In this case we shall assume that the size of the interface is $2l$ at every level of coupling. The total depth D of the tree is $n^2 - 1 = N - 1$.

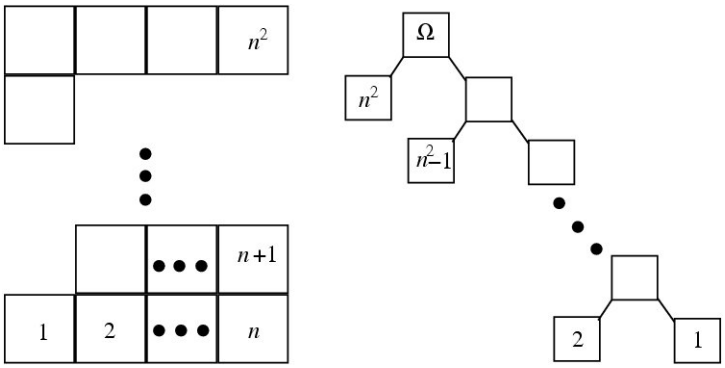

Figure 6.7: Sparse tree.

Depending upon the actual discretization within each subdomain, certain tree topology may be more suitable than others. Finding the optimal tree topology for a given problem is quite a challenging problem.

6.6 Multiple Subdomains with Multiple Time-steps

The conventional FETI implementation for multiple subdomains with a uniform time step uses a single global interface between all the subdomains and couples them all at once using the PCPG approach. The GC method uses a similar global interface approach for coupling multiple subdomains even for multiple time steps.

In the present case, it turns out, that this global interface coupling is feasible only for multiple subdomains using the same time step. When more than two subdomains using two or more time steps are coupled, the global interface problem involves the simultaneous solution of all the different time step levels which is computationally impractical. Consider the simplest case with three-way split SDOF problem with subdomains A, B and C. Let subdomain A be integrated with time step ΔT and subdomains B and C integrated with Δt where $\Delta T = 2\Delta t$. Subdomain equations for advancing the state of the system by ΔT are:

$$\begin{aligned}
&\mathbb{M}^A\mathbb{U}^A_{n+2} + \mathbb{N}^A\mathbb{U}^A_n + \mathbb{C}^A\mathbf{\Lambda}_{n+2} = \mathbb{P}^A_{n+2} \\
&\mathbb{U}^A_{n+1} = \frac{1}{2}\left(\mathbb{U}^A_n + \mathbb{U}^A_{n+2}\right) \\
&\mathbb{M}^B\mathbb{U}^B_{n+1} + \mathbb{N}^B\mathbb{U}^B_n + \mathbb{C}^B\mathbf{\Lambda}_{n+1} = \mathbb{P}^B_{n+1} \\
&\mathbb{M}^B\mathbb{U}^B_{n+2} + \mathbb{N}^B\mathbb{U}^B_{n+1} + \mathbb{C}^B\mathbf{\Lambda}_{n+2} = \mathbb{P}^B_{n+2} \\
&\mathbb{M}^C\mathbb{U}^C_{n+1} + \mathbb{N}^C\mathbb{U}^C_n + \mathbb{C}^C\mathbf{\Lambda}_{n+1} = \mathbb{P}^C_{n+1} \\
&\mathbb{M}^C\mathbb{U}^C_{n+2} + \mathbb{N}^C\mathbb{U}^C_{n+1} + \mathbb{C}^C\mathbf{\Lambda}_{n+2} = \mathbb{P}^C_{n+2}
\end{aligned} \tag{6.23}$$

Interface equations for conventional FETI with dofs $\mathbf{\Lambda} = (\mathbf{\Lambda}^{AB}, \mathbf{\Lambda}^{BC}, \mathbf{\Lambda}^{CA})$ are:

$$\begin{aligned}
\mathbf{\Lambda}^{BC}_{n+1}: &\quad \boldsymbol{C}^{B_{BC}}\dot{\boldsymbol{U}}^B_{n+1} + \boldsymbol{C}^{C_{BC}}\dot{\boldsymbol{U}}^C_{n+1} = 0 \\
\mathbf{\Lambda}^{AB}_{n+1}: &\quad \boldsymbol{C}^{A_{AB}}(\boldsymbol{M}^A\ddot{\boldsymbol{U}}^A_{n+1} + \boldsymbol{K}^A\boldsymbol{U}^A_{n+1} + \boldsymbol{C}^{A^T}\mathbf{\Lambda}_{n+1} - \boldsymbol{P}^A_{n+1}) = 0 \\
\mathbf{\Lambda}^{CA}_{n+1}: &\quad \boldsymbol{C}^{A_{CA}}(\boldsymbol{M}^A\ddot{\boldsymbol{U}}^A_{n+1} + \boldsymbol{K}^A\boldsymbol{U}^A_{n+1} + \boldsymbol{C}^{A^T}\mathbf{\Lambda}_{n+1} - \boldsymbol{P}^A_{n+1}) = 0 \\
\mathbf{\Lambda}_{n+2}: &\quad \mathbb{B}^A\mathbb{U}^A_{n+2} + \mathbb{B}^B\mathbb{U}^B_{n+2} + \mathbb{B}^C\mathbb{U}^C_{n+2} = 0
\end{aligned} \tag{6.24}$$

Note that the equations for both time steps t_{n+1} and t_{n+2} are singular because of the presence of redundant interface dofs. However, they are still solvable.

Alternatively, for the algebraically partitioned interface with dofs $\boldsymbol{\Lambda}^A$, $\boldsymbol{\Lambda}^B$, $\boldsymbol{\Lambda}^C$ and interface velocity $\dot{\boldsymbol{U}}^I$, the equations are:

$$
\begin{aligned}
\boldsymbol{\Lambda}^A_{n+1}: \quad & \boldsymbol{C}^A(\boldsymbol{M}^A\ddot{\boldsymbol{U}}^A_{n+1} + \boldsymbol{K}^A\boldsymbol{U}^A_{n+1} + \boldsymbol{C}^{A^T}\boldsymbol{\Lambda}^A_{n+1} - \boldsymbol{P}^A_{n+1}) = 0 \\
\boldsymbol{\Lambda}^B_{n+1}: \quad & \boldsymbol{C}^B\dot{\boldsymbol{U}}^B_{n+1} = \boldsymbol{B}^B\dot{\boldsymbol{U}}^I_{n+1} \\
\boldsymbol{\Lambda}^C_{n+1}: \quad & \boldsymbol{C}^C\dot{\boldsymbol{U}}^C_{n+1} = \boldsymbol{B}^C\dot{\boldsymbol{U}}^I_{n+1} \\
\boldsymbol{U}^I_{n+1}: \quad & \boldsymbol{B}^{A^T}\boldsymbol{\Lambda}^A_{n+1} + \boldsymbol{B}^{B^T}\boldsymbol{\Lambda}^B_{n+1} + \boldsymbol{B}^{C^T}\boldsymbol{\Lambda}^C_{n+1} = 0 \\
\boldsymbol{\Lambda}^A_{n+2}: \quad & \boldsymbol{C}^A\dot{\boldsymbol{U}}^B_{n+2} = \boldsymbol{B}^A\dot{\boldsymbol{U}}^I_{n+2} \\
\boldsymbol{\Lambda}^B_{n+2}: \quad & \boldsymbol{C}^B\dot{\boldsymbol{U}}^B_{n+2} = \boldsymbol{B}^B\dot{\boldsymbol{U}}^I_{n+2} \\
\boldsymbol{\Lambda}^C_{n+2}: \quad & \boldsymbol{C}^C\dot{\boldsymbol{U}}^C_{n+2} = \boldsymbol{B}^C\dot{\boldsymbol{U}}^I_{n+2} \\
\boldsymbol{U}^I_{n+2}: \quad & \boldsymbol{B}^{A^T}\boldsymbol{\Lambda}^A_{n+2} + \boldsymbol{B}^{B^T}\boldsymbol{\Lambda}^B_{n+2} + \boldsymbol{B}^{C^T}\boldsymbol{\Lambda}^C_{n+2} = 0
\end{aligned}
\tag{6.25}
$$

which are non-singular.

In either case, the interface equations are coupled across both the time steps and cannot be solved in a time stepping manner t_{n+1} followed by t_{n+2} using a global interface between the three subdomains A, B and C. Through some algebra, trying to decouple the two time steps, it turns out that if one couples two subdomains and treats them as a single new subdomain then one can solve the interface equations in a time stepping manner *recursively*.

The idea of recursive, hierarchical domain decomposition naturally generalizes to cases with multiple time steps where different nodes in the tree structure can now be integrated with different time steps. However, if a particular node in the tree has more than two daughter nodes, then all the daughter nodes must all have the same time step. This allows one to couple multiple levels of time steps by coupling two subdomains at a time in a recursive manner.

6.7 Results

We first consider a *split* single degree of freedom (SDOF) problem where a mass and spring system excited with a step load is split into 4 subdomains (with masses = (2, 1, 4, 3); spring stiffnesses = (3, 2, 3, 1) ; step loads = (1, 3, 2, 2)) and integrated with the same time step of 0.75 but with different combinations of Newmark schemes. The

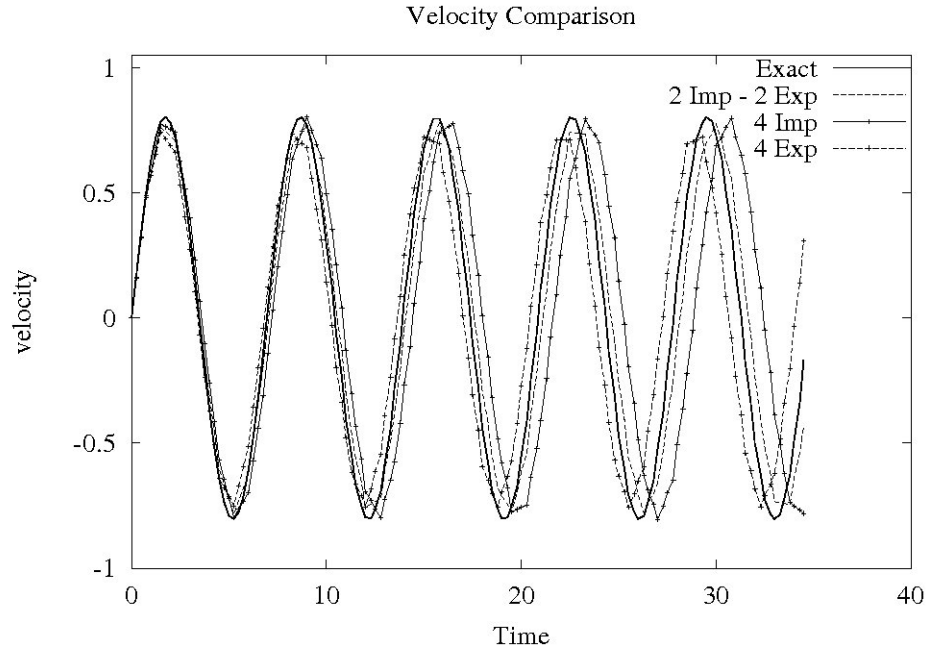

Figure 6.8: Split SDOF response.

velocity response of the system depicted in figure 6.8 shows a good agreement between the coupled solutions and the exact solution. Note that, when all the four subdomains are integrated with the same scheme (explicit/implicit), they exactly reproduce the response of a single undecomposed system integrated with that scheme. The same period shortening and elongation is observed for explicit and implicit integration respectively. When 2 subdomains are integrated explicitly and 2 implicitly, these effects tend to cancel out and produce an average response.

An example of a 3D cantilever beam with a rectangular cross-section, decomposed into

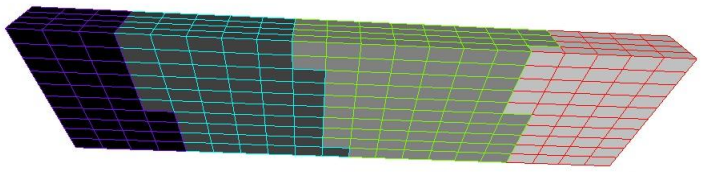

Figure 6.9: Decomposition of a 3D beam.

4 irregular subdomains is considered in figure 6.9. The beam is fixed on the left end and excited with a step load in the axial direction at the free end. All the subdomains are integrated explicitly with the same time step, just below the Courant stability limit. The response consists of a stress wave bouncing back and forth through the length of the beam and matches exactly with the response of the corresponding undecomposed system. This decomposition is chosen only to verify that the coupling is seamless and is not ideal for practical computations since it creates large interfaces.

The response of a fan excited with impact loads of varying duration applied on the tips of its blades is shown in figure 6.10. The problem was decomposed into 6 subdomains, one for each blade as shown in figure 6.11. An implicit time step 10 times the Courant stability limit of the corresponding explicit scheme was used for all the 6 subdomains. The results were found to be identical for the decomposed and the undecomposed fan.

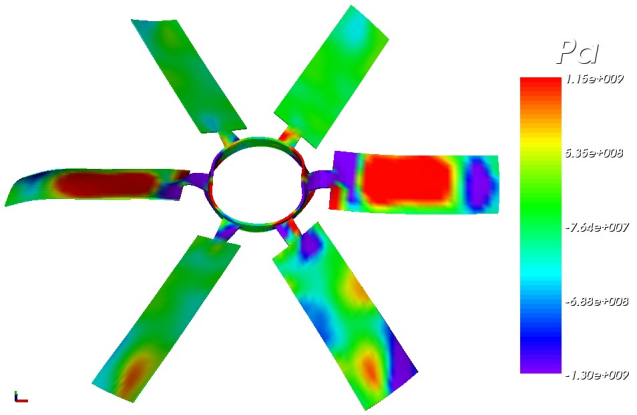

Figure 6.10: Response of fan blades to impact loading.

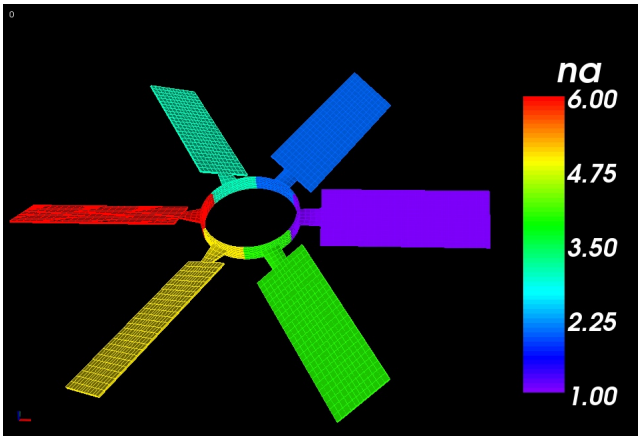

Figure 6.11: Domain decomposition of a fan with 6 blades.

Figure 6.12 shows a notched beam decomposed into 3 subdomains as shown. The center subdomain containing the notch has a very fine mesh while the two end subdomains are relatively coarse. The two end subdomains are integrated implicitly while the center subdomain is integrated explicitly with time step ratio 10:1 between them. There was no discernible difference between its response and that of an undecomposed system integrated explicitly with the smaller time step. The computational times for the two cases were also comparable. This example shows that one can integrate different subdomains with multiple time-steps and preserve the accuracy of the solution within each subdomain.

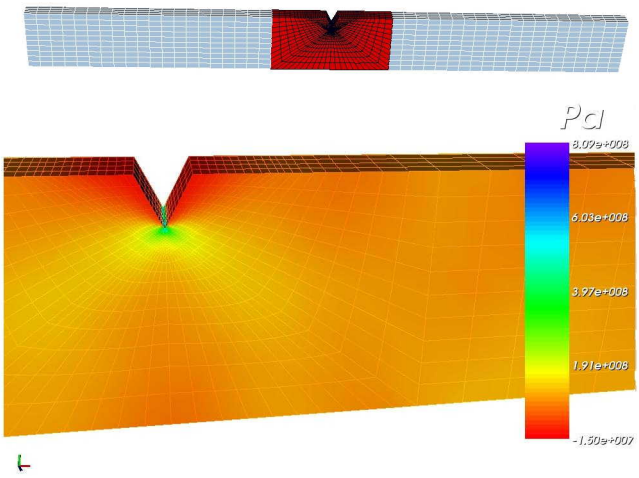

Figure 6.12: Response of a notched beam.

6.8 Parallel Implementation

In order to avail all the benefits of domain decomposition in structural dynamics, it is important to develop an efficient approach for parallel implementation of the recursive multi-time-step method. A simple *loop-level* parallelization can be implemented rather quickly by computing all the tasks of the multi-time-step in serial as before but using multiple processors within each task such as the subdomain and interface solves. This approach was found to stagnate in terms of performance beyond 4 processors and is not very attractive.

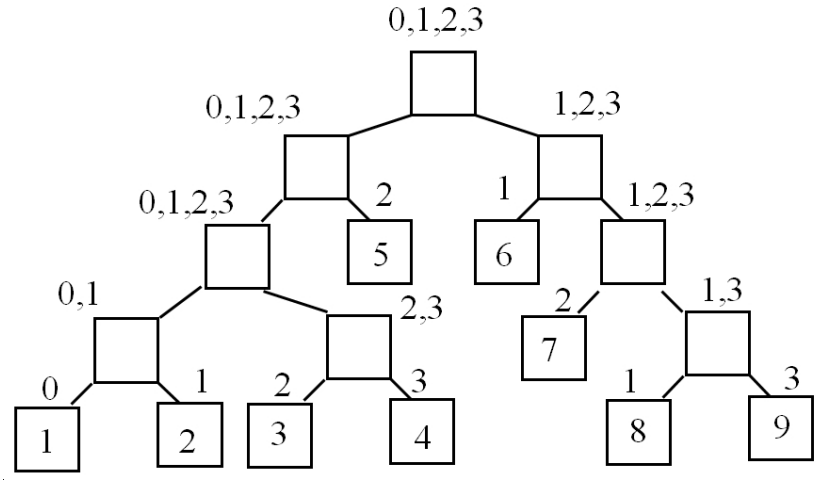

Figure 6.13: Parallel implementation with 9 subdomains on 4 processors.

A complete parallelization of all the tasks associated with recursive multi-time-step method can be implemented by considering the topology of the tree to be coupled. Let the number of processors available be p and the total number subdomains (leaf nodes) in the tree be S. It is assumed that mesh can always be decomposed in such a way that $S \geq p$ so that each leaf node can be assigned a unique processor. All the data associated with a particular subdomain is stored only on the processor it is assigned. Now starting at the lowest level in the tree, one can move up the tree by associating each parent node in the tree with all the processors of its daughter nodes. The interface data associated with coupling two or more daughter subdomains is common to all processors associated with the parent and is

stored on all of them. In this way, each processor has knowledge of the entire tree structure but stores only the subdomain and interface data that is relevant to itself. Similarly every node in the tree has associated with itself some subset of the available processors.

An efficient parallel implementation of the present method is being developed to benchmark its performance with respect to conventional methods in the literature. Synchronization of tasks and correct communication ordering between processors and subdomains is a challenging problem in this parallel tree coding approach. The current parallelization strategy is to have all the processors run the same recursive code and traverse the full tree in order to achieve coupling. Each processor traverses down the levels in the tree and at every tree node it checks whether it is involved at that particular node or not. If it is involved, then it traverses down further that path, else returns to the parent tree node and explores other branches as shown in Figure 6.14.

```
SUB RecursiveSolve(Ω_(A,B))
   if (myid) ∈ PROCSET(Ω_(A,B)) then
      if Ω_(A,B) is a Leaf Node then
         Solve Ω_(A,B)
      else
         Call RecursiveSolve(Ω_A, ℙ^A, 𝕌_n^A, 𝕍_{n+1}^A)
         Call RecursiveSolve(Ω_B, ℙ^B, 𝕌_n^B, 𝕍_{n+1}^B)
         Couple Ω_A and Ω_B      ⇒ SYNCHRONIZE with PROCSET(Ω_(A,B))
      end if
   end if
END SUB RecursiveSolve
```

Figure 6.14: Parallelization of recursive subroutines.

The variable myid denotes the processor identifier and PROCSET($\Omega_{(A,B)}$) is the set of all processors associated with the tree node $\Omega_{(A,B)}$. This implementation avoids deadlock between processors. Note that if a certain processor is not in PROCSET($\Omega_{(A,B)}$) then it does not traverse any daughter nodes of $\Omega_{(A,B)}$ further down in the tree which helps in load balancing if one chooses the tree and processor topology carefully.

6.9 Conclusion

A method for coupling multiple subdomains in a hierarchical manner is presented. The method is shown to be more economical in terms of computational cost for long time dynamic simulations with respect to the conventional FETI method implemented with direct solvers. An iterative implementation of the same is comparable to the PCPG implementation of FETI and FETI-DP methods.

It was found that the present method is more suitable for problems where one can identify subdomains with small interfaces between them. It was shown that the coupling is seamless and preserves the total system energy exactly. This approach achieves the greatest efficiency when used in conjunction with multiple time-steps. Problems which are largely linear and have small zones of non-linear behavior are also good candidates for the present method since the non-linearity can be isolated locally within a subdomain rather than iterating for convergence over the entire mesh.

A parallel implementation of the present hierarchical method for non-linear problems with multiple levels of time steps is in progress. The present code is capable of analyzing linear problems using recursive domain decomposition for the same time step across all subdomains. Parallelization of the non-linear multi-time-step coupling method is yet to be verified.

Among other issues that will arise when using this approach is the choice of subdomains. One may develop strategies for adaptive subdivision of the global mesh into subdomains as the mesh is refined. This can complement adaptive mesh refinement strategies being used currently.

Chapter 7

Conclusions and Future Directions

The present multi-time-step coupling method provides an efficient way for solving large-scale problems in structural dynamics. It is the only method in the literature that allows domain decomposition with multiple time steps and time stepping schemes and perfectly preserves the computational characteristics such as stability, accuracy and total energy of individual subdomains and of the global problem.

The present method uses dual Schur domain decompostion method to divide the mesh into two or more subdomains coupled to each other with Lagrange multipliers. These Lagrange multipliers ensure the continuity of the coupled solution across the interfaces between the subdomains. The method is implemented in three steps: solve the subdomains independently, solve for the Lagrange multipliers on the interface and update the individual subdomains. For multiple time step problems, the computation of Lagrange multipliers is required only at the large time step and this makes it possible for one to efficiently couple subdomains with very large time step ratios between them. A stability analysis for linear problems shows that the present method exactly preserves the total energy of the system which is crucial in ensuring that the coupling is stable and accurate. It is observed that as long as the stability requirements for the time steps are met within individual subdomains, the coupling itself is unconditionally stable.

The multi-time-step method is also extended to incorporate non-linear structural behavior. This is achieved by linearizing the system equations and then using the linear coupling method incrementally. For large non-linear problems, a preconditioned conjugate gradient (PCG) approach is used to compute the Lagrange multipliers on the interface. The present

method is most efficient for problems with localized non-linearities as the non-linear behavior can be restricted to a small subdomain and the rest of the mesh can be treated linearly.

The PCG approach for solving the Lagrange multipliers on the interface becomes expensive when more than two subdomains share the same interface. A recursive coupling method which successively couples two subdomains at a time until the original structure is recovered circumvents this problem. It is found that this approach is not only beneficial for the traditional FETI implementation but is also the only feasible method for coupling three or more subdomains with different time steps.

A parallel implementation of the present method using the message passing interface (MPI) is also presented. Different subdomains are assigned to different processors and the same recursive code is run on all processors to avoid deadlock. MPI is used to communicate data between processors for computing the Lagrange multipliers on the interface.

7.1 Future Directions

Innovative ideas are continually being developed in domain decomposition and coupling methods for large-scale problems. Application of the present multi-time-step coupling method to other time integrators such as the generalized-α method is being explored. Another enhancement of the current method is to allow non-matching mesh interfaces between subdomains. A variety of methods such as the mortar method are already available in the literature for the same.

Large-scale problems usually have physical phenomena that span multiple orders of magnitude in space and time. In such cases it is more efficient to have multiple levels of time-steps that serve as a gradation for coupling the various time scales of the problem. For instance, if one is studying dynamic crack propagation in a large structure using a *multi-scale* model based perhaps on *atomistic* calculations at the crack tip, then one may choose subdomains that form multiple concentric zones of elements around the crack tip and use varying levels

of time steps for each of them.

Time integration of other physical systems such as first order transient heat conduction with multiple time steps using similar concepts is also being considered. Perhaps a more general extension of this idea could lead to the development of a multi-scale method in time or in space-time. Application areas for problems that involve mutliple scales in space and time such as fluid-structure interaction problems, contact, fracture, multi-body problems, multi-field coupled problems, quantum-continuum coupling to extract bulk properties from atomic or nanoscale computations would benefit immensely from such a multi-scale method in time.

References

[1] P. Chadwick. *Continuum Mechanics, Concise Theory and Problems*. Dover Publications, 1976.

[2] K. D. Hjelmstad. *Fundamentals of Structural Mechanics*. Springer, 2005.

[3] J. Bonet and R. D. Wood. *Nonlinear continuum mechanics for finite element analysis*. Cambridge University Press, Cambridge, UK, 1997.

[4] J. E. Marsden and T. J. R. Hughes. *Mathematical Foundations of Elasticity*. Dover Publications Inc., New York, 1994.

[5] C. Truesdell and W. Noll. *The Non-Linear Field Theories of Mechanics*. Springer, 2004.

[6] M. E. Gurtin. *An Introduction to Continuum Mechanics*. Academic Press, 1981.

[7] J. C. Simo and T. J. R. Hughes. *Computational Inelasticity*. Springer, Berlin, 1998.

[8] P. Haupt. *Continuum Mechanics and Theory of Materials*. Springer, 2000.

[9] S. C. Brenner and L. R. Scott. *The Mathematical Theory of Finite Element Methods*. Springer, 1994.

[10] J. E. Marsden and T. S. Ratiu. *Introduction to Mechanics and Symmetry*. Springer, 2002.

[11] T. J. R. Hughes. *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*. Dover Publications Inc., Mineola, New York, 2000.

[12] O. C. Zienkiewicz and R. L. Taylor. *The Finite Element Method. Volume 1, The Basis*. Butterworth-Heinemann, Oxford, UK, fifth edition, 2000.

[13] N. M. Newmark. A method of computation for structural dynamics. *Journal of Engineering Mechanics, ASCE*, 85:67–94, 1959.

[14] M. T. Heath. *Scientific Computing, An Introductory Survey*. WCB/McGraw-Hill, 1997.

[15] Klaus-Jürgen Bathe. *Finite Element Procedures*. Prentice Hall, New Jersey, 1996.

[16] J. Chung and G. M. Hulbert. A time integration algorithm for structural dynamics with improved numerical dissipation: the generalized-α method. *Journal of Applied Mechanics, ASME*, 60:371–375, 1993. Transactions of the ASME.

[17] H. M. Hilber, T. J. R. Hughes, and R. L. Taylor. Improved numerical dissipation for time integration algorithms in structural dynamics. *Earthquake Engineering & Structural Dynamics*, 5:283–292, 1977.

[18] W.L. Wood, M. Bossak, and O.C. Zienkiewicz. An alpha modification of newmark's method. *International Journal for Numerical Methods in Engineering*, 15:1562–1566, 1981.

[19] J. C. Simo and K. K. Wong. Unconditionally stable algorithms for rigid body dynamics that exactly preserve energy and momentum. *International Journal for Numerical Methods in Engineering*, 31:19–52, 1991.

[20] J. C. Simo, N. Tarnow, and K. K. Wong. Exact energy-momentum conserving algorithms and symplectic schemes for non-linear dynamics. *Computer Methods in Applied Mechanics and Engineering*, 100:63–116, 1992.

[21] O. Gonzales and J. C. Simo. On the stability of symplectic and energy-momentum algorithms for non-linear hamiltonian sytems with symmetry. *Computer Methods in Applied Mechanics and Engineering*, 134:197–222, 1996.

[22] F. Armero and I. Romero. On the formulation of high-frequency dissipative time-stepping algorithms for non-linear dynamics. Part I: low-order methods for model problems and nonlinear elastodynamics. *Computer Methods in Applied Mechanics and Engineering*, 190:2603–2649, 2001.

[23] F. Armero and I. Romero. On the formulation of high-frequency dissipative time-stepping algorithms for nonlinear dynamics. Part II: second-order methods. *Computer Methods in Applied Mechanics and Engineering*, 190:6783–6824, 2001.

[24] A. P. Veselov. Integrable discrete-time systems and difference operators. *Functional Analysis and Its Applications*, 22:83–93, 1988.

[25] J. E. Marsden and M. West. Discrete mechanics and variational integrators. *Acta Numerica*, 10:357–514, 2001.

[26] E. Hairer, C. Lubich, and G. Wanner. *Geometric Numerical Integration: Structure Preserving Algorithms for Ordinary Differential Equations*. Springer-Verlag, New York, NY, second edition, 2006.

[27] C. Kane, J. E. Marsden, M. Ortiz, and M. West. Variational integrators and the newmark algorithm for conservative and dissipative mechanical systems. *International Journal for Numerical Methods in Engineering*, 49:1295–1325, 2000.

[28] A. Lew, J. E. Marsden, M. Ortiz, and M. West. Variational time integrators. *International Journal for Numerical Methods in Engineering*, 60:153–212, 2004.

[29] C. Johnson, U. Nävert, and J. Pitkäranta. Finite element methods for linear hyperbolic problems. *Computer Methods in Applied Mechanics and Engineering*, 45:285–312, 1984.

[30] T. J. R. Hughes and G. M. Hulbert. Space-time finite element methods for elastodynamics: formulation and error estimates. *Computer Methods in Applied Mechanics and Engineering*, 66:339–363, 1987.

[31] G. M. Hulbert and T. J. R. Hughes. Space-time finite element methods for second-order hyperbolic equations. *Computer Methods in Applied Mechanics and Engineering*, 84:327–348, 1990.

[32] X. D. Li and N. E. Wiberg. Structural dynamic analysis by a time-discontinuous galerkin finite element method. *International Journal for Numerical Methods in Engineering*, 39:2131–2152, 1996.

[33] M. Mancuso and F. Ubertini. An efficient time discontinuous galerkin procedure for non-linear structural dynamics. *Computer Methods in Applied Mechanics and Engineering*, 195(44-47):6391–6406, 2006.

[34] R. Abedi, B. Petracovici, and R.B. Haber. A space-time discontinuous galerkin method for linearized elastodynamics with element-wise momentum balance. *Computer Methods in Applied Mechanics and Engineering*, 195:3247–3273, 2006.

[35] P. Betsch and P. Steinmann. Conservation properties of a time FE method - Part II: Time-stepping schemes for non-linear elastodynamics. *International Journal for Numerical Methods in Engineering*, 50:1931–1955, 2001.

[36] X. Zhou and K. K. Tamma. Algorithms by design with illustrations to solid and structural mechanics/dynamics. *International Journal for Numerical Methods in Engineering*, 66(11):1738–1790, 2006.

[37] A. Toselli and O. Widlund. *Domain Decomposition Methods - Algorithms and Theory*. Springer, 2005.

[38] Y. Fragakis and M. Papadrakakis. The mosaic of high performance domain decomposition methods for structural mechanics: Formulation, interrelation and numerical efficiency of primal and dual methods. *Computer Methods in Applied Mechanics and Engineering*, 192:3799–3830, 2003.

[39] *Domain Decomposition Methods in Science and Engineering, Proceedings of the Seventeenth Domain Decomposition meeting in St. Wolfgang-Strobl, Austria*, Lecture Notes in Computational Science and Engineering. Springer-Verlag, 2006.

[40] P.L. Lions. On schwarz alternating method. I. In *Proceedings of The First International Symposium on Domain Decomposition methods for Partial Differential Equations, Paris, France*, pages 1–42. SIAM, Philadelphia, 1987.

[41] R. H. Dodds Jr. and L.A. Lopez. Substructuring in linear and nonlinear analysis. *International Journal for Numerical Methods in Engineering*, 15:583–597, 1980.

[42] F.-X. Roux. Domain decomposition methods for static problems. *Recherche Aerospatiale*, 1:37–48, 1990.

[43] Jan Mandel. Balancing domain decomposition. *Communications in Numerical Methods in Engineering*, 9:233–241, 1993.

[44] T. Belytschko and R. Mullen. Mesh partitions of explicit-implicit time integration. In *US-Germany Symposium on Formulations and Computational Algorithms in Finite Element Analysis*, pages 673–690, MIT, Cambridge, MA, 1976.

[45] T. J. R. Hughes and W.K. Liu. Implicit-explicit finite elements in transient analysis: Stability theory. *Journal of Applied Mechanics, ASME*, 45:371–374, 1978. Transactions of the ASME.

[46] T. J. R. Hughes and W.K. Liu. Implicit-explicit finite elements in transient analysis: Implementation and numerical examples. *Journal of Applied Mechanics, ASME*, 45:375–378, 1978. Transactions of the ASME.

[47] T. J. R. Hughes, K. S. Pister, and R. L. Taylor. Implicit-explicit finite elements in nonlinear transient analysis. *Computer Methods in Applied Mechanics and Engineering*, 17/18:159–182, 1979.

[48] T. J. R. Hughes and R. A. Stephenson. Convergence of implicit-explicit algorithms in nonlinear transient analysis. *International Journal of Engineering Science*, 19:295–302, 1981.

[49] I. Miranda, R. M. Ferencz, and T. J. R. Hughes. An improved implicit-explicit time integration method for structural dynamics. *Earthquake Engineering & Structural Dynamics*, 18:643–653, 1989.

[50] T. Belytschko and Y. Y. Lu. Stability analysis of elemental explicit-implicit partitions by fourier methods. *Computer Methods in Applied Mechanics and Engineering*, 95:87–96, 1992.

[51] W. K. Liu and T. Belytschko. Mixed-time implicit-explicit finite elements for transient analysis. *Computers and Structures*, 15:445–450, 1982.

[52] T. Belytschko and R. Mullen. Stability of explicit-implicit mesh partitions in time integration. *International Journal for Numerical Methods in Engineering*, 12:1575–1586, 1978.

[53] T. Belytschko, H.-J. Yen, and R. Mullen. Mixed methods for time integration. *Computer Methods in Applied Mechanics and Engineering*, 17/18:259–275, 1979.

[54] K. C. Park. Partitioned transient analysis procedures for coupled-field problems: Stability analysis. *Journal of Applied Mechanics, ASME*, 47:370–376, 1980. Transactions of the ASME.

[55] K. C. Park and C. A. Felippa. Partitioned transient analysis procedures for coupled-field problems: Accuracy analysis. *Journal of Applied Mechanics, ASME*, 47:919–926, 1980. Transactions of the ASME.

[56] T. Belytschko, W. K. Liu, and P. Smolinski. Multi-stepping implicit-explicit procedures in transient analysis. In *Proceedings of the International Conference on Innovative Methods for Nonlinear Problems*, pages 135–153. Pineridge Press International Ltd., Swansea, U.K., 1984.

[57] T. Belytschko and Y. Y. Lu. Explicit multi-time step integration for first and second order finite element semidiscretizations. *Computer Methods in Applied Mechanics and Engineering*, 108:353–383, 1993.

[58] M. O. Neal and T. Belytschko. Explicit-explicit subcycling with non-integer time step ratios for structural dynamic systems. *Computers and Structures*, 31:871–880, 1989.

[59] P. Smolinski. An explicit multi-time step integration method for second order equations. *Computer Methods in Applied Mechanics and Engineering*, 94:25–34, 1992.

[60] P. Smolinski. Stability analysis of multi-time step explicit integration method. *Computer Methods in Applied Mechanics and Engineering*, 95:291–300, 1992.

[61] P. Smolinski, S. Sleith, and T. Belytschko. Stability of an explicit multi-time step integration algorithm for linear structural dynamics equations. *Computational Mechanics*, 18:236–244, 1996.

[62] P. Smolinski. Subcycling integration with non-integer time steps for structural dynamics problems. *Computers and Structures*, 59:273–281, 1996.

[63] P. Smolinski and Y.-S. Wu. An implicit multi-time step integration method for structural dynamics problems. *Computational Mechanics*, 22:337–343, 1998.

[64] Y. S. Wu and P. Smolinski. A multi-time step integration algorithm for structural dynamics based on the modified trapezoidal rule. *Computer Methods in Applied Mechanics and Engineering*, 187:641–660, 2000.

[65] W. J. T. Daniel. The subcycled Newmark algorithm. *Computational Mechanics*, 20:272–281, 1997.

[66] W. J. T. Daniel. A study of the stability of subcycling algorithms in structural dynamics. *Computer Methods in Applied Mechanics and Engineering*, 156:1–13, 1998.

[67] W. J. T. Daniel. Explicit/implicit partitioning and a new explicit form of the generalized alpha method. *Communications in Numerical Methods in Engineering*, 19:909–920, 2003.

[68] M. Ortiz and B. Nour-Omid. Unconditionally stable concurrent procedures for transient finite element analysis. *Computer Methods in Applied Mechanics and Engineering*, 58:151–174, 1986.

[69] M. Ortiz, B. Nour-Omid, and E. D. Sotelino. Accuracy of a class of concurrent algorithms for transient finite element analysis. *International Journal for Numerical Methods in Engineering*, 26:379–391, 1988.

[70] M. Ortiz, E. D. Sotelino, and B. Nour-Omid. Efficiency of group implicit concurrent algorithms for transient finite element analysis. *International Journal for Numerical Methods in Engineering*, 28:2761–2776, 1989.

[71] E. D. Sotelino. A concurrent explicit-implicit algorithm in structural dynamics. *Computers & Structures*, 51:181–190, 1994.

[72] S. Modak and E. D. Sotelino. The iterative group implicit algorithm for parallel transient finite element analysis. *International Journal for Numerical Methods in Engineering*, 47:869–885, 2000.

[73] Y. Dere and E. D. Sotelino. Modified iterative group-implicit algorithm for the dynamic analysis of structures. *Journal of Structural Engineering, ASCE*, 130:1436–1444, 2004.

[74] A. Lew, J. E. Marsden, M. Ortiz, and M. West. Asynchronous variational integrators. *Archive for Rational Mechanics and Analysis*, 167:85 – 146, 2003.

[75] K. C. Park and C. A. Felippa. A variational principle for the formulation of partitioned structural systems. *International Journal for Numerical Methods in Engineering*, 47:395–418, 2000.

[76] C. A. Felippa et al. Partitioned analysis of coupled mechanical systems. *Computer Methods in Applied Mechanics and Engineering*, 190:3247–3270, 2001.

[77] M. A. Puso. A 3d mortar method for solid mechanics. *International Journal for Numerical Methods in Engineering*, 59:315–336, 2004.

[78] B. Flemisch, M. A. Puso, and B. I. Wohlmuth. A new dual mortar method for curved interfaces: 2d elasticity. *International Journal for Numerical Methods in Engineering*, 63:813–832, 2005.

[79] P. Hansbo, C. Lovadina, I. Perugia, and G. Sangalli. A lagrange multiplier method for the finite element solution of elliptic interface problems using non-matching meshes. *Numerische Mathematik*, 100:91–115, 2005.

[80] C. Farhat and F. X. Roux. A method for finite element tearing and interconnecting and its parallel solution algorithm. *International Journal for Numerical Methods in Engineering*, 32:1205–1227, 1991.

[81] C. Farhat and F. X. Roux. Implicit parallel processing in structural mechanics. *Computational Mechanics Advances*, 2:1–124, 1994.

[82] C. Farhat, J. Mandel, and F. X. Roux. Optimal convergence properties of the FETI domain decomposition method. *Computer Methods in Applied Mechanics and Engineering*, 115:365–385, 1994.

[83] C. Farhat, L. Crivelli, and F. X. Roux. Extending substructure based iterative solvers to multiple load and repeated analyses. *Computer Methods in Applied Mechanics and Engineering*, 117:195–209, 1994.

[84] K. C. Park, M. R. Justino Jr., and C. A. Felippa. An algebraically partitioned FETI method for parallel structural analysis: algorithm description. *International Journal for Numerical Methods in Engineering*, 40:2717–2737, 1997.

[85] D. J. Rixen, C. Farhat, R. Tezaur, and J. Mandel. Theoretical comparison of the FETI and algebraically partitioned FETI methods, and performance comparisons with a direct sparse solver. *International Journal for Numerical Methods in Engineering*, 46:501–533, 1999.

[86] C. Farhat and J. Mandel. The two-level FETI method for static and dynamic plate problems Part I: An optimal iterative solver for biharmonic systems. *Computer Methods in Applied Mechanics and Engineering*, 155:129–151, 1998.

[87] C. Farhat, K. Pierson, and M. Lesoinne. The second generation FETI methods and their applications to the parallel solution of large-scale linear and geometrically nonlinear structural analysis probelms. *Computer Methods in Applied Mechanics and Engineering*, 184:333–374, 2000.

[88] C. Farhat, M. Lesoinne, P. LeTallec, K. Pierson, and D. Rixen. FETI-DP: a dual primal unified FETI method-part I: A faster alternative to the two-level FETI method. *International Journal for Numerical Methods in Engineering*, 50:1523–1544, 2001.

[89] C. Farhat, L. Crivelli, and F. X. Roux. Transient FETI methodology for large-scale parallel implicit computations in structural mechanics. *International Journal for Numerical Methods in Engineering*, 37:1945–1975, 1994.

[90] C. Farhat, L. Crivelli, and M. Geradin. Implicit time integration of a class of constrained hybrid formulations - Part I: Spectral stability theory. *Computer Methods in Applied Mechanics and Engineering*, 125:71–107, 1995.

[91] C. Farhat and M. Chandesris. Time-decomposed parallel time-integrators: theory and feasibility studies for fluid, structure, and fluid-structure applications. *International Journal for Numerical Methods in Engineering*, 58:1397–1434, 2003.

[92] C. Farhat, J. Cortial, C. Dastillung, and H. Bavestrello. Time-parallel implicit integrators for the near-real-time prediction of linear structural dynamic responses. *International Journal for Numerical Methods in Engineering*, 67(5):697–724, 2006.

[93] A. Gravouil and A. Combescure. Multi-time-step explicit-implicit method for nonlinear structural dynamics. *International Journal for Numerical Methods in Engineering*, 50:199–225, 2001.

[94] A. Combescure and A. Gravouil. A numerical scheme to couple subdomains with different time-steps for predominantly linear transient analysis. *Computer Methods in Applied Mechanics and Engineering*, 191:1129–1157, 2002.

[95] A. Prakash and K. D. Hjelmstad. A FETI based multi-time-step coupling method for newmark schemes in structural dynamics. *International Journal for Numerical Methods in Engineering*, 61:2183–2204, 2004.

[96] Y. Fragakis and M. Papadrakakis. The mosaic of high-performance domain decomposition methods for structural mechanics - Part II: Formulation enhancements, multiple right-hand sides and implicit dynamics. *Computer Methods in Applied Mechanics and Engineering*, 193:4611–4662, 2004.

[97] A. Prakash and K. D. Hjelmstad. A multi-time step coupling method for newmark schemes in structural dynamics. In *Proceedings of McMat2005: Joint ASME / ASCE / SES Conference on Mechanics and Materials*, 2005.

[98] E. Taciroglu and K. D. Hjelmstad. Simple nonlinear model for elastic response of cohesionless granular materials. *Journal of Engineering Mechanics, ASCE*, 128:969–978, 2002.

[99] W. Hackbush. On the computation of approximate eigenvalues and eigenfunctions of elliptic operators by means of a multigrid method. *SIAM Journal on Numerical Analysis*, 16:201–215, 1979.

[100] J. K. Bennighof and R. B. Lehoucq. An automated multilevel substructuring method for eigenspace computation in linear elastodynamics. *SIAM Journal on Scientific Computing*, 25:2084–2106, 2004.

[101] P. Arbenz, U. L. Hetmaniuk, R. B. Lehoucq, and R. S. Tuminaro. A comparison of eigensolvers for large-scale 3d modal analysis using amg-preconditioned iterative methods. *International Journal for Numerical Methods in Engineering*, 64:204–236, 2005.

[102] C. Papalukopoulos and S. Natsiavas. Dynamics of large scale mechanical models using multilevel substructuring. *Journal of Computational and Nonlinear Dynamics, ASME*, 2:40–51, 2007. Transactions of the ASME.

[103] Y. Saad and J. Zhang. BILUTM: A domain-based multilevel block ILUT preconditioner for general sparse matrices. *SIAM Journal on Matrix Analysis and Applications*, 21:279–299, 1999.

[104] Z. Li, Y. Saad, and M. Sosonkina. pARMS: a parallel version of the algebraic recursive multilevel solver. *Numerical Linear Algebra with Applications*, 10:485–509, 2003.

[105] S. Le Borne. Multilevel hierarchical matrices. *SIAM Journal on Matrix Analysis and Applications*, 28:871–889, 2006.

[106] J. Dubinski. A parallel tree code. *New Astronomy*, 1:133–147, 1996.

[107] Y. M. Marzouk and A. F. Ghoniem. K-means clustering for optimal partitioning and dynamic load balancing of parallel hierarchical n-body simulations. *Journal of Computational Physics*, 207:493–528, 2006.

[108] Y.-S. Yang and S.-H. Hsieh. Iterative mesh partitioning optimization for parallel nonlinear dynamic finite element analysis with direct substructuring. *Computational Mechanics*, 28:456–468, 2002.

[109] M. Griebel and G. Zumbusch. Parallel adaptive subspace correction schemes with applications to elasticity. *Computer Methods in Applied Mechanics and Engineering*, 184:303–332, 2000.

Author's Biography

Arun Prakash was born January 17, 1977 in Ghazipur, Uttar Pradesh, India. He received his bachelor's degree (B.Tech.) in Civil Engineering from Indian Institute of Technology (IIT), Delhi in 1999. He then obtained his master's degree (M.S.) in Civil Engineering from University of Illinois at Urbana-Champaign in 2001 following which he has been pursuing a doctorate in the same department. During his graduate studies, he has been employed as research assistant by the Center for Simulation of Advanced Rockets (CSAR) under the guidance of Professor Keith D. Hjelmstad. His research interests fall in the wide area of computational methods for coupled multi-physics problems.