PARameter Identification System (PARIS)
for
Automated Finite Element Model Updating

A Thesis
Submitted by

Peeyush Rohela

In Partial Fulfillment of the Requirements
for the Degree of

Master of Science
in
Civil and Environmental Engineering

TUFTS UNIVERSITY
August 2012

Thesis Committee:
Prof. Masoud Sanayei, Chair
Prof. Brian R. Brenner
Prof. Michael A. Zimmerman

Tufts
UNIVERSITY

OFFICE OF GRADUATE STUDIES

CERTIFICATE OF FITNESS

This certifies that the undersigned, appointed to determine the fitness of

Peeyush Rohela

for the degree of Master of Science in Civil and Environmental Engineering

have examined the candidate's thesis/dissertation (or papers) on the subject:

PARameter Identification System (PARIS) for Automated Finite Element Model Updating

and have found it satisfactory.

06/22/2012

Date of Defense

This certifies further that the candidate this day has successfully passed the customary examination.
We recommend, therefore, that the degree be awarded under the usual conditions.

SIGNATURES	NAMES
Masoud Sanayei	Masoud Sanayei
Chairperson, Examining Committee	*Chairperson, Examining Committee*
Brian Brenner	Brian R. Brenner
Michael A. Zimmerman	Michael A. Zimmerman

Medford, Massachusetts 02155
617.627.3395

Abstract

Structural Health Monitoring (SHM) using Non-Destructive Test (NDT) data has become more prevalent due to our deteriorating transport infrastructure and emerging need for its objective condition evaluation. Furthermore, recent progresses in sensor technology, data acquisition systems, and software have significantly enhanced our capabilities for condition monitoring of structures. To address the above need, a SHM program, called PARIS13.0 (PARameter Identification System) has been developed for automated parameter estimation and full-scale Finite Element (FE) model updating. A unique feature of this program is utilization of interactive functionality of existing commercial software packages for Finite Element Analysis (FEA) and numerical computation for condition assessment of full-scale structures. SAP2000 is commercial FEA software with an integrated 3D modeling environment. MATLAB, on the other hand, is a fourth generation programming environment that is optimized for numerical computation. Open Application Programming Interface (OAPI) from SAP2000 developers enables SAP2000 to be used as a slave program for FEA through MATLAB. Additionally, the Graphical User Interface (GUI) of SAP2000 can be used for FE model creation and visualization. This combination facilitated development of a platform for automated full-scale FE model updating in the form of PARIS13.0 computer program. PARIS13.0 calibrates the FE model iteratively and automatically by minimizing the residual between the predicted response of the FE model and the measured response from simulated NDT. The capabilities of PARIS13.0 were validated using three examples for full-scale FE model updating based on simulated NDT data from assumed damage scenarios.

Acknowledgements

"Coming together is a beginning,

Keeping together is a progress,

Working together is a success."

~ Henry Ford

It is with immense gratitude that I acknowledge the support of my thesis adviser, Professor Masoud Sanayei, in sharing his expertise and knowledge through the course of this research. His persistence and patience was instrumental in realization of my master's level research and coursework. I would like to thank other members of my thesis committee, Professor Brian Brenner and Professor Michael Zimmerman, for their recommendations to make this research more concise. I take pleasure in acknowledging the partial funding from the NSF research project "Whatever Happened to the Long Term Bridge Design?" towards this work. Collaborating with other researchers on this project helped me to attain a thorough understanding of various aspects of bridge structural health monitoring.

I extend my gratitude to the graduate students in Room 308A, Anderson Hall. Their presence and support has made the last two years pleasantly memorable. Special thanks to Jesse Sipple for helping with at least one technical question a day for the last one year without judging its level of usefulness, or the lack thereof.

I extend deepest appreciation to my parents for sharing my dream and to my girlfriend Pallavi whose unyielding support and constant encouragement has only brought out the better of me.

Nomenclature

Parameter Estimation

NUP	Number of Unknown Parameters
NPG	Number of Parameter Groups
NLC	Number of Load Cases
SS	Static Stiffness
SF	Static Flexibility
SSTR	Static Strain
MS	Modal Stiffness
MF	Modal Flexibility
p	Vector of Unknown Parameters
p_G	Vector of Grouped Unknown Parameters
S(p)	Analytical Sensitivity Matrix
$S_G(p)$	Analytical Sensitivity Matrix for Grouped Parameters
e(p)	Error Function Vector
NDT	Non-Destructive Test
FE	Finite Element
q	Measured Physical Quantity
q^m	Measured Physical Quantity Contaminated with Measurement Errors
f	Applied Force Vector
ε	Strain Vector
u	Displacement Vector
M	Mass Matrix
K	Stiffness Matrix
D	Dynamic Matrix $D = K^{-1}M$
B	Mapping Matrix from Strain-Displacement Relationship $\varepsilon = B\ q$
ϕ	Mode Shape
J(p)	Scalar Objective Function
Σ_e	Covariance Matrix of Error Function
GT	Group Transformation Matrix
NM	Total Number of Measurements

Subscripts

a	Measured
b	Unmeasured

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

Structural Health Monitoring (SHM) holds the promise of helping to address the problem of deteriorating bridge infrastructure. Implementation of SHM requires development of Finite Element (FE) models that can serve as a baseline for comparing measured structural performance to analytical predictions. A key part of the process is FE model updating. FE model updating is a global SHM technique that is capable of assessing structural damage at elemental level by observing the structure's global static response to known static loads or its global dynamic response to known excitation. The behavior of a FE model is governed by its stiffness, mass, and damping parameters. The underlying principle of FE model updating is to iteratively calibrate these parameters of the FE model of a structure such that its response reconciles with that of an actual structure under known static loads or known dynamic excitation. Alternatively referred to as parameter estimation, FE model updating can be an effective method for condition assessment

of structures. This thesis describes development and implementation of a set of tools for structural parameter estimation and FE model updating.

1.2 Scope of Work

This research is focused on developing an in-house SHM software for structural parameter estimation and automated finite element model updating for full-scale structures. A computer program developed at Tufts University, PARameter Identification System (PARIS13.0), is the centerpiece of this research. PARIS13.0 is designed with a goal of applying stiffness and mass matrices based FE model updating techniques to full-scale structures using simulated NDT data. PARIS13.0 is capable of updating the stiffness and mass parameters of a structural FE model iteratively and automatically based on the following input data:

i) Initial guess of parameters in the form of FE model created in SAP2000

ii) Known static loads applied to the structure at a subset of degrees of freedom (DOFs) of the FE model.

iii) Measured static or dynamic response at a subset of total DOFs of the model from simulated damage scenarios.

PARIS13.0 utilizes the provision for interfacing commercial software packages, SAP2000 and MATLAB, through Application Programming Interface (API) for achieving full-scale FE model updating capability. There are distinctive advantages that arise out of utilizing interactive functionality of SAP2000 and MATLAB. Primarily, interfacing SAP2000 and MATLAB for creating a robust SHM program avoids the need for writing a Finite Element Analysis (FEA) solver and developing a Graphical User Interface (GUI) for FE model creation and visualization.

Moreover, commercial software are developed by a team of software professionals and are optimized for performance thereafter. The familiarity of these software packages in both the structural engineering industry and academia makes it convenient for a larger user group to reap the benefits of this research.

PARIS13.0 replaces the original PARIS computer program developed at Tufts University between 1997 and 2008. PARIS13.0 is a newly developed program designed to work with full-scale structural FE models made in SAP2000. Unlike its predecessor, it relies on SAP2000 API commands to read and update the parameters of the FE model. It also uses SAP2000 API commands for setting analysis options for finite element analysis in SAP2000 and obtaining results post analysis. In addition to the analytical sensitivity based optimization techniques that it inherits from the older PARIS, it employs advanced optimization features available in MATLAB for fast and efficient FE model updating. The compiled PARIS13.0 computer program will be made available at no cost for use of entire SHM research community through the Tufts University Civil and Environmental Engineering Department's SHM research website.

1.3 Thesis Layout

Chapter 2 of the thesis, which will be later submitted for journal publication, is the major contribution of this research. It presents, in addition to the theory of parameter estimation, the capabilities of the PARIS13.0 program to update full-scale structural FE models. That PARIS13.0 has potential for full-scale FE model updating is further demonstrated using three large-scale FE model updating examples using simulated data. The first example is that of the FE

model of Phase-I IASC-ASCE SHM Benchmark Structure whose stiffness and mass parameters are successfully updated using static and dynamic error functions. The second and third examples use the FE models of Girder-3 of the Powder Mill Bridge (PMB) and of the entire Powder Mill Bridge, respectively. For each of these examples, the FE model description is given first. The assumed damage scenarios are then described. Subsequently, relevant NDT loads application and locations for measurement of response are listed. The results of FE model updating using PARIS13.0 are given at the end of each damage case.

Chapter 3 discusses design and development of the program. Chapter 4 contains additional information about the three FE models used as verification examples in Chapter 2. Chapter 5 is the user reference manual for PARIS13.0. It also contains a sample MATLAB based user input file. PARIS13.0 requires input in form of a SAP2000 model and a MATLAB based input file. The detailed instructions for making a SAP2000 model compatible with SAP2000 are provided in the user manual. The specific format for defining model updating problem through the input file can also be found in the user manual.

Chapter 2

Automated Finite Element Model Updating of Full-Scale Structures with PARameter Identification System (PARIS)

2.1 Introduction

The National Bridge Inventory recognizes nearly 11% of the bridges in the United States to be structurally deficient (NBI Database). The criterion for structural deficiency is identification of one or more defects in the structural members which is in need of repair. In absence of more quantified damage assessment, bridge inspections conducted as per the National Bridge Inspection Standards (NBIS) provide a more subjective evaluation of the structural health of the bridge. Inspections are regularly conducted every two years, with special inspections called for more frequently in some cases where deterioration has progressed to warrant more attention. Most inspections rely on visual evaluation and not on direct measurements and analysis.

Structural Health Monitoring (SHM) using parameter estimation has the potential to be used in conjunction with the current methods of inspection to add a layer of confidence in decision making with regards to structural management and maintenance. SHM can provide for a more

objective evaluation of the state of the structure by assessing deterioration or damage levels from observing the response of a structure to known static loads or controlled excitation. Unlike limited visual inspections, SHM can provide a continuous stream of measured data to help benchmark structural condition and evaluate long term trends and performance.

Parameter estimation is the process of updating the parameters of an a-priori mathematical model to correlate its predicted response with the measured data at selected observation points for a given set of excitations. This is called the inverse problem and is unlike the forward analysis problem, which is to determine response of a structure to a known excitation as a function of physical known parameters of the model. An acceptable method for solving input-output inverse problems is by minimizing the residual between the predicted response from a mathematical model and the measured response of the actual system constituting the observed data set. The process thus involves starting with an a-priori mathematical model based on certain assumptions of the initial values of the parameters. Key parameters of the model are considered to be unknown and then iteratively updated for minimizing the residual and updating the unknown parameters until their best fit values are achieved. The updated parameter estimates are reflective of actual state of the structure and can thus be used to assess the level of damage in the structure.

Techniques used for damage assessment are broadly classified as global and local. Global SHM methods are used to determine damage in the structure by observing overall response of the structure. Chang et al. (2003) reviewed the current global SHM methods under research for structural parameter estimation and structural damage assessment. Early SHM researchers like Begg et al. (1976) based their hypotheses fundamentally on identifying damage through change

in dynamic characteristics of structure. Local SHM methods are used to assess severity of the damage in a localized region. Material flaw detection based on distribution of eddy currents is an example of local method but is more popularly used in aerospace industry.

SHM techniques are further categorized as model based and non-model based. There are various methods on the non-model based approach. As an example, Farrar and Worden (2006) discussed recent research that identifies SHM problem primarily as a Statistical Pattern Recognition problem. Deraemaeker et al. (2008) used output-only vibration measurements under variable environmental conditions for damage detection using a numerical example of a bridge. Techniques built on adaptive systems like Artificial Neural Networks (ANN), Wang et al. (2011), have also found their place in the SHM community.

A set of model based methods for structural health monitoring relies on observing changes in the structure's stiffness, mass, and damping matrices to correlate predicted and measured response. Creating a finite element model based on design assumptions is the first step in physical model based damage assessment. The accuracy of a computer model can be evaluated on how well it is able to predict the system's actual response. The initial model is, in most cases, not an accurate representation of true system behavior, which further necessitates model refinement and calibration. Typical stiffness parameters of a FE model are EA (axial rigidity), EI (bending rigidity) and GJ (torsional rigidity) for frame elements, E (modulus of elasticity) for shell and solid elements, k_X (translational stiffness) and k_θ (rotational stiffness) for joint springs. For dynamic systems, mass parameter in a FE model is m (mass) of the finite elements. Structural damage is considered here as a change in any of the structural parameters used for defining the a-

priori model. Assuming the structure is in linear elastic region, measured response of the structure is then used to update stiffness and mass parameters of the FE model until a close match between its analytical and measured response is obtained. An updated model represents 'as is' state of the system and forms the baseline for prediction of operational response.

Kaouk and Zimmerman (1994) adopted the Minimum Rank Perturbation Theory (MRTP) for locating damage and determining its extent from measured eigenvalues and eigenvectors. Using vibrational test data, Doebling et al. (1998) presented the problem of determining local stiffness parameters as a well-determined linear least square problem based on the decomposition of the flexibility matrix. Wang et al. (2001) presented a structural damage identification algorithm for Damage Signature Matching (DMS) using change in static displacement response as a function of perturbation in stiffness matrix and changes in natural frequency. Farrar and Jauregui (1997) performed comparative analysis of various damage based assessment method using modal data from a damaged and undamaged bridge. Teughels and De Roeck (2004) applied an iterative sensitivity based FE model updating technique using discrepancies in natural frequencies and mode shape data to a beam finite element model of a 3 span-57m long highway bridge in Switzerland. This research is based on the parameter estimation formulations developed by researchers at Tufts University for PARIS (PARameter Identification System) and replaces the earlier computer program with an entirely new PARIS13.0 which has extended and advanced capabilities for full-scale FE model updating. This work comprises of Phase-I and Phase-II of the 3-Phase plan to achieve FE model updating capability based on NDT data. Phase-I is the computer program development and forms a major part of this work. Phase-II includes testing the program with simulated NDT data to validate program functionality. Model updating

examples that use simulated NDT data are also a part of this work. Phase-III, which is to utilize real NDT data from three diagnostic NDTs on the PMB to establish a baseline FE model of the bridge, is not included in this study. Phase-III will be the next milestone to be achieved in the future.

Other examples of SHM and parameter estimation software are DIAMOND and FEMtools. DIAMOND (Doebling et al., 1997) from Los Alamos National Laboratory is a software toolbox with several vibration based damage identification algorithms for vibration tests simulation, modal data analysis, finite element correlation, and comparison of both linear and nonlinear damage identification techniques. Its successor, DIAMOND II (Allen et al., 2003) is a collection of algorithms for treatment of SHM as a statistical pattern recognition problem. FEMtools (FEMtools Website), a commercial FE model updating software that interfaces with FEA packages like ABAQUS and NASTRAN is developed by Dynamic Design Solutions. PARIS13.0, the subject of this research, is an in-house research software that can be customized and modified as needed to conduct parameter estimation and model updating studies on full-scale structural models using simulated NDT data.

2.2 PARIS13.0 (PARameter Identification System)

PARIS (PARameter Identification System) is the acronym for a custom MATLAB based computer program for parameter estimation and FE model updating. The initial software was developed by Sanayei (1997) with his graduate students at Tufts University. Using static and modal measurements as input data, it could be used for stiffness and mass parameter estimation

at the element level. However, its use was limited to FE model updating for only planar and spatial truss and frame structures. The in-house FEA solver for PARIS was also written in MATLAB. Text based finite element modeling and restricted FEA capabilities rendered PARIS unsuitable for full-scale model updating. PARIS13.0 is the new program that succeeds PARIS. Figure 2.1 illustrates the FE model updating process in PARIS13.0. The flowchart connectors labeled as API describe the two-way automatic data exchange between MATLAB and SAP2000. This section contains description of the parameter estimation process using PARIS13.0.

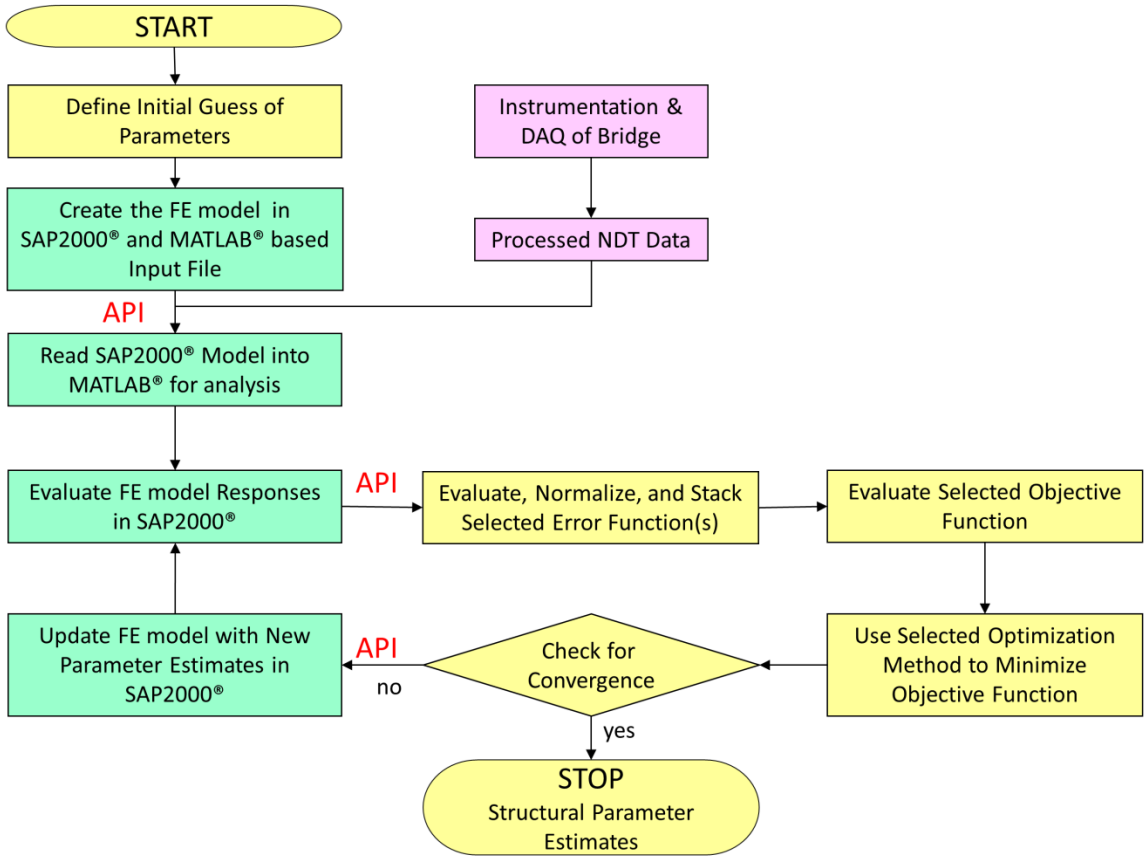


Figure 2.1 PARIS13.0 Flowchart

Parameter estimation process, as shown in Figure 2.1 begins with an initial FE model created in SAP2000 which is based on the preliminary guess of unknown structural parameters. These parameters are rigidities and mass properties of elements in an a-priori model. The MATLAB-

based PARIS13.0 program invokes SAP2000 using Application Programming Interface (API) commands for use in parameter estimation. Residual between the predicted response from the FE model and measured simulated NDT data indicates the level of mismatch between the physical model and the real structure's responses. Each type of residual is termed as an error function. Various error functions are stacked for multi-response FE model updating. A scalar objective function is formed using the error functions and is optimized in MATLAB. Unknown parameter values are updated in an iterative fashion using analytical or numerical sensitivities. The updated model becomes the initial model for the next iteration. In each iteration of the optimization process for parameter estimation, the SAP2000 model is invoked by MATLAB for calculating the FE responses. The iterations continue until the convergence criteria are met. Successful convergence indicates that the physical model closely assimilate the real structure's behavior. The following paragraphs describe various concepts and formulations used in PARIS13.0 in the sequence in which they are encountered in the FE model updating process.

2.2.1 Finite Element Model Creation and Analysis Options

PARIS13.0 is programmed to calibrate a FE model comprised of frame, quadrilateral planar shell, and cuboid solid elements using static and/or modal data. Its usefulness lies in selecting the most appropriate finite element types for modeling a structure. While solid elements may be better suited for modeling problems without geometrical oversimplification, shell elements, on other hand, provide greater convenience in modeling in addition to reduced analysis time. The model updating options have also been extended to update joint spring and 2-node link element stiffnesses in the FE model. It is important however, that the FE model is created in SAP2000

adhering to the specifications provided in PARIS13.0 user manual (Chapter 5). The freedom to choose any selection of active DOFs for analysis also lies with the user.

2.2.2 Error Functions

Error functions are the heart of FE model based parameter estimation. In the parameter estimation process, error functions serve to estimate the unknown parameters of an a-priori model of the structure. Error functions are a measure of residual between the predicted response from the FE model and NDT data either from field measurements or from FE model damage simulation based on input damage indices. Error functions are formulated in terms of unknown parameters. PARIS13.0 has five available error functions, three of which are based on static loads and measurements while the remaining two are based on modal measurements. Generally, an error function is expressed as the difference between analytical and measured physical quantities as 2.1

$$e(p) = q_{predicted} - q_{measured} \tag{2.1}$$

where 'q' denotes a response quantity which can either be strain or translation and rotation under a set of known static loads or modal excitation. The static and modal error functions are further subdivided as stiffness and flexibility based error functions. In essence, the stiffness based error functions measure the residual between applied forces and the flexibility based error functions quantify residual between measured displacements. Static strain error function can also be categorized under flexibility based error function as strains are related directly to displacements in finite element analysis.

Static Stiffness (SS)

Sanayei and Onipede (1990) developed the static stiffness error function in (2.3). They showed that the predicted displacement response from applied static test load at one subset of degrees of freedom (DOF) and measured displacement response at another subset of DOF could be used to detect damage in the structures at the element level. Static stiffness error function is a force based error function that compares predicted and measured forces. Based on the partitioning of the force-displacement relationship as per measured and unmeasured displacements given by equation 2.2, the static stiffness error function is written as 2.3,

$$\begin{Bmatrix} f_a \\ f_b \end{Bmatrix} = \begin{bmatrix} K_{aa} & K_{ab} \\ K_{ba} & K_{bb} \end{bmatrix} \begin{Bmatrix} u_a \\ u_b \end{Bmatrix} \tag{2.2}$$

$$e_{SS}(p) = (K_{aa} - K_{ab}K_{bb}^{-1}K_{ba})\,u_a + K_{ab}K_{bb}^{-1}f_b - f_a \tag{2.3}$$

where vectors u_a and u_b are vectors of measured and unmeasured displacements. Submatrices K_{aa}, K_{ab}, K_{ba}, and K_{bb} in equation 2.2 represent the partitioned terms of the stiffness matrix and f_a and f_b represent the partitioned applied load force vector depending on measured and unmeasured displacements at 'a' and 'b' subsets of degrees of freedom.

Static Flexibility (SF)

Sanayei et al. (1997) developed the static flexibility error function. It is based on the inverse force-displacement relationship and it compares the predicted and measured displacements at a subset of DOFs using equation 2.4,

$$e_{SF}(p) = (K_{aa} - K_{ab}K_{bb}^{-1}K_{ba})^{-1}(f_a - K_{ab}K_{bb}^{-1}f_b) - u_a \tag{2.4}$$

Prior to development of SF error function given by equation 2.4 by Sanayei et al. (1997), Banan and Hjelmstad (1994) had also formulated a similar flexibility based error function in terms of flexibility matrix $A = K^{-1}$. Static flexibility based error function formulation by Sanayei et al. (1997) is used in PARIS13.0.

Static Strain (SSTR)

Sanayei and Saletnik (1996) developed the Static Strain error function. It is defined as the difference between predicted and measured strains at selected observation points as,

$$e_{SSTR}(p) = B_a K^{-1} f_a - \varepsilon_a \tag{2.5}$$

where B_a is the mapping matrix from the relation $\varepsilon = B_a\, q$ and is used to calculate strains from displacements of a frame element. However, for shell elements in PARIS13.0, a more direct approach has been adopted to calculate strains from FE model stresses using Hooke's Law. SAP2000 calculates stresses at the four integration points for shell elements and extrapolates them to the corner nodes for output. Internally in PARIS13.0, strain calculated at a particular node is averaged based on shell elements surrounding that node.

Modal Stiffness (MS)

Sanayei et al. (1998) developed the Modal Stiffness error function. The eigenvalue problem shown in 2.6 forms the basis of modal stiffness error function.

$$K\phi_i = \lambda M \phi_i \tag{2.6}$$

In equation 2.6, *K* and *M* are the stiffness and the mass matrices, respectively. Vector ϕ_i represents the i^{th} natural mode shape and λ_i is the square of the i^{th} natural frequency.

$$e_{MS}(p)_i = (K_{aa} - \lambda_i M_{aa}) - (K_{ab} - \lambda_i M_{ab})(K_{bb} - \lambda_i M_{bb})^{-1}(K_{ba} - \lambda_i M_{ba})\phi_{ai} \tag{2.7}$$

After partitioning equation 2.6 similar to (2.2) and condensing out unmeasured mode shapes, the modal stiffness error function is given by equation 2.7 as a residual of modal forces. Vector $\phi_{a\,i}$ is defined as the measured modal displacements at subset 'a' of degrees of freedom at the i^{th} iteration.

Modal Flexibility (MF)

Sanayei et al. (2001) developed modal flexibility based error function by condensing the characteristic equation written in terms of the flexibility matrix. Hjelmstad (1996) also arrived at a similar modal flexibility based error function formulation by only partitioning the mass matrix at measured and unmeasured DOF without using condensation. Similar to static flexibility, the modal flexibility includes the inverse of stiffness matrix, *K*, in its formulation. Modal flexibility error function, as formulated by Sanayei et al. (2001) for PARIS, is shown in equation 2.8.

$$e_{MF}(p)_i = (\lambda_i^2 D_{ab}(I - \lambda_i D_{bb})^{-1} D_{ba} + \lambda_i D_{aa} - I)\phi_{ai} \tag{2.8}$$

The matrix D in equation 2.8 is the dynamic matrix given by the relationship $D = K^{-1}M$.

2.2.3 Scalar Objective Function

The quadratic scalar objective function *J(p)* defined in equation 2.9 is used to judge convergence or divergence of the iterative model updating process.

$$J(p) = e(p)^T e(p) \tag{2.9}$$

A decreasing value of *J(p)* at successive iterations denotes convergence.

2.2.4 Normalization

Two types of normalization are used: parameter normalization and error function normalization. The difference in the order of magnitudes of various types of parameters and measured quantities gives rise to numerical difficulties by making the inverse problem ill-conditioned. Normalizing parameter values to 1.0 by dividing them by initial estimated values is called parameter normalization. The estimated parameters are thus estimated as a fraction or multiplier of the initial estimate of 1. An estimated value of 0 indicates complete damage. Additionally, a statistical method for adjusting to measurement errors is to normalize an error function based on the observability of each measurement. DiCarlo (2008) implemented this normalization scheme for all three static error functions in PARIS and demonstrated a higher confidence level in estimated parameters especially in presence of measurement errors.

$$J(p) = e(p)^T \Sigma_e^{-1} e(p) \tag{2.10}$$

Using equation 2.10, the static error functions in PARIS13.0 are both weighted and normalized depending on the level of measurement error. The results of this process approximate that of a maximum likelihood estimator.

2.2.5 Objective Function Minimization

Minimizing *J(p)* results in newer estimate of unknown parameters *(p+Δp)* which take the place of initial estimates for the next iteration. PARIS13.0 provides the option of minimizing *J(p)* either by using analytical sensitivity matrix formulated in equation 2.11 or numerical sensitivity by engaging MATLAB Optimization Toolbox.

$$S(p) = \left[\frac{\partial \{e(p)\}}{\partial p} \right] \tag{2.11}$$

$$S(p)\Delta p = -e(p) \tag{2.12}$$

S(p) is the analytical sensitivity matrix obtained by taking the derivative of the error function vector *e(p)* with respect to the unknown parameters. The sensitivity matrix *S(p)* is of size $NM \times NUP$, where *NM* is defined as the total number of measurements and *NUP* is the number of unknown parameters in the problem. The change in parameter values, *Δp*, in the i^{th} *iteration*, is obtained from the iterative solution of equation 2.12 using the Gauss-Newton method leading to

$$p_{i+1} = p_i + \Delta p_i \tag{2.13}$$

An alternative to using analytical sensitivity matrix is successfully implemented by using the algorithm for constrained minimization of the algebraically non-linear multivariable error

function available in the Optimization Toolbox in MATLAB. The minimization function, *fmincon,* uses interior point algorithm which is suited for large and sparse problems as well as for small and dense problems (MATLAB Help Menu). It satisfies upper and lower bound on parameters at all stages and can recover from undefined and infinite values if encountered in the optimization process. This algorithm requires the target function to be defined in terms of unknown parameters that returns a scalar value, which in PARIS13.0 is the scalar objective function *J(p).*

2.2.6 Convergence Criteria

Convergence criteria are required for terminating the optimization routine. The dissimilarity between the updated unknown parameter values and true values is reconciled within reasonable limits at this level. Convergence criteria used in PARIS13.0 are given by equations 2.14 and 2.15.

$$\frac{p_{i+1} - p_i}{p_1} \leq relative\ \Delta p\ tolerance \tag{2.14}$$

$$\frac{J(p)_{i+1} - J(p)_i}{J(p)_1} \leq relative \quad \Delta J(p)\ tolerance \tag{2.15}$$

Relative *Δp* tolerance for convergence is the change in the value of unknown parameter value with respect to its initial value between two successive iterations. Similarly, relative *ΔJ(p)* tolerance is the change in the objective function value with respect to its initial value.

2.2.7 Parameter Grouping

In analytical sensitivity matrix based objective function minimization, the sensitivity matrix *S(p)* is post-multiplied by a Boolean group transformation matrix, *GT*, to map the elements of the sensitivity matrix according to the unknown parameter groups defined by the user. This step reduces the size of *S(p)* from $NM \times NUP$ to $NM \times NPG$, where *NPG* is the number of parameter groups. Newer estimate of parameter is obtained in the same manner as described above except that $S_G(p)$ is now used in equation 2.12 instead of *S(p).* Subsequently, the smaller vector p_G is mapped backwards to the size of total unknown parameters by pre-multiplying it with *GT*.

In the case of constrained function minimization using MATLAB Optimization Toolbox, the objective function *J(p)* takes the input argument as a smaller vector of grouped unknown parameters p_G of size $NPG \times 1$ rather than the original vector of unknown parameters *p* of size $NUP \times 1$. The resulting multivariable optimization problem then has lesser unknown variables compared to *NUP* without grouping.

2.2.8 Multi-response Parameter Estimation

Multi-response parameter estimation refers to the use of more than one type of measured data and corresponding error functions in the model updating process (Bell et al. 2007). In PARIS13.0, stacking is implemented by default whenever a combination of error functions is used. Various error functions and sensitivity matrices can be stacked vertically using equations 2.16 and 2.17.

$$e(p) = \left\{ e(p)_1^T \quad e(p)_2^T \quad \ldots \quad e(p)_n^T \right\}^T \tag{2.16}$$

$$[S(p)] = \left[S(p)_1^T \quad S(p)_2^T \quad \ldots \quad S(p)_n^T \right]^T \tag{2.17}$$

Observing different types of responses simultaneously to estimate a set of unknown parameters is an effective approach as it increases the amount of information in the parameter estimation process. Stacking allows for using different sets of measurements for different load cases and subset of measurements as well as using different types of static and modal data with corresponding error functions. Examples of stacking are using e_{sf}, e_{sstr}, e_{ms} or e_{ss}, e_{sstr}, e_{ms} if such data is available.

2.2.9 Measurement Errors

The accuracy of measurements from a NDT is an important factor in the parameter estimation process. Normally, a bound on accuracy of a measuring device is stated by the manufacturer. This bound on the level of errors, however, can be exceeded due to difficulties in sensor installation and connection losses (Smith, 2009). Although there is no way to accurately model actual measurement errors, basic types of simulated measurement errors are use in PARIS13.0 to better understand the error tolerance of the above error functions. It provides the user with options to introduce uniformly or normally distributed errors in simulated data to study the influence of measurement errors on estimated parameters. While uniform error distribution resembles a banded type of error with equal probabilities of occurrence throughout, the normal error distribution is a non-banded type of error with its probabilities of occurrence being higher

closer to true values (Sanayei et al., 1992). Equation 2.18 and 2.19 are used for contaminating simulated NDT data with proportional and absolute errors, respectively.

$$q^m = q\left(1 + \frac{1}{2}e_q \otimes R_q\right) \tag{2.18}$$

$$q^m = q + \frac{1}{2}e_q \otimes R_q \tag{2.19}$$

In the equations above, q is the vector of measured forces or simulated response. e_q represents the percentage level of measurement error in equation 2.18 where as in equation 2.19 it represents absolute error with units as defined by the user. The vector R_q is fully populated either with uniformly distributed random numbers between 0 and 1 or normally distributed random numbers with 0 mean and standard deviation of 0.5. The symbol $\otimes$ denotes element-wise vector multiplication.

2.3 Verification Examples

The capabilities of PARIS13.0 using SAP2000 for finite element model creation and analysis are demonstrated using three FE model updating examples using simulated NDT data. Each of these examples demonstrates a combination of program features as shown in Table 2.1. This section contains the description of the structures, model updating problem formulation, and parameter estimation results for all three examples.

Table 2.1 PARIS©13.0 Verification Examples

FEATURE	TYPE / DESCRIPTION	IASC-ASCE SCALE MODEL				GIRDER-3 PMB MODEL		FULL-SCALE PMB MODEL	
Finite Elements	Frame		✓						✓	
	Shell		✓				✓		✓	
	Solid						✓			
	Spring						✓		✓	
Error Functions	Damage Cases →	1	2	3	4	1	2	1	2
	SS	✓	✓		✓	✓		✓	
	SF	✓	✓			✓		✓	
	SSTR	✓	✓			✓	✓	✓	✓
	MS			✓	✓				
	MF			✓					
Normalization	Maximum Likelihood	✓							
Measurement Errors	Errors in Applied Forces	✓							
	Errors in Measurements	✓							
Parameter Grouping		✓	✓	✓	✓	✓	✓	✓	✓
Stacking		✓			✓				

The first column presents various features of PARIS13.0 which are further described in column two. Verification examples using these features of the program are shown in columns three to five for the FE model of the IASC-ASCE SHM Benchmark Structure, the Powder Mill Bridge (PMB) Girder-3 model, and the full-scale PMB model, respectively.

2.3.1 IASC-ASCE SHM Benchmark Structure

The first example in this section uses the analytical model based on the Phase-I study of the IASC-ASCE SHM Benchmark Problem. The global geometry and member properties as well as some damage scenarios to simulate damage are taken from Black and Ventura (1998) and Johnson et al. (2004). The FE model used in this research is slightly different than the actual analytical model. Originally, two finite element models were developed by IASC-ASCE SHM Task Group. The first model was a 12 DOF shear-building model in which only two horizontal translations and one rotation is permitted for each floor. The second model had 120 DOFs which

only constrained all nodes at each floor for the same horizontal translations and in-plane rotations (Johnson et al. 2004). The structural model used here has 1944 DOFs because of greater discretization of shell elements at the floor levels and also from the fact that all rotational and translational DOFs are made active in analysis.

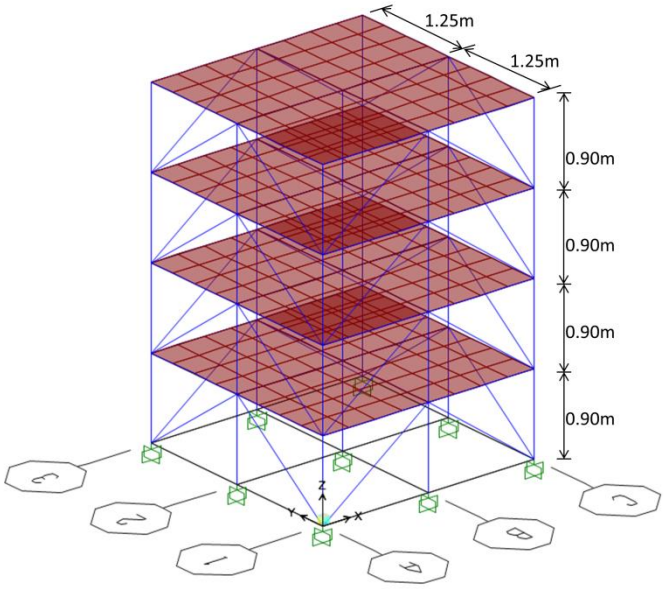


Figure 2.2 IASC-ASCE SHM Benchmark Structure

Structure and Model Description

Figure 2.2 shows the FE model of the experimental steel frame housed in the Earthquake Engineering Research Laboratory at the University of British Columbia. It is a 4-story, 3.6 m high structure with two bays in orthogonal lateral directions with each bay measuring 1.25m. The sections are made out of hot rolled grade 300W steel with 300MPa nominal yield stress. Steel sections that constitute the structure are B100x9 (columns), S75x11 (beams), and L25x25x3 (braces). The floor slab at each story is comprised of 4 square slabs. Mass of each panel is 800kg at the first floor and 600kg at the second and the third floor**.** The fourth floor has 3 panels of 400kg mass and 1 panel that has a mass of 550kg. Each of the 4 slabs measuring 1.25m x 1.25m

is modeled using 16 thin-shell elements. The columns are oriented with their weak axis (y) along the global X axis in the SAP2000 model. All beams connect rigidly to the columns while the braces are pin connected and thus behave as axial members. All columns are fixed at the base.

Damage Scenarios

PARIS13.0 was used to estimate unknown stiffness and mass parameters for four simulated damage scenarios summarized in Table 2.2. Unknown parameters were estimated for the first two damage cases using static error functions. For damage cases 1, the first floor braces with 8 unknown EA parameters were grouped into 4 different unknown parameter groups. For the second damage case, 3 EI_{zz} and 3 EI_{yy} parameters of the base columns at Grid-A were paired as two groups. For the third damage case in which mass of one slab panel is the unknown parameter, modal error functions were used to estimate the reduced mass of 16 shell elements making up that panel as one group. In the fourth damage case, both stiffness and mass parameters were estimated using SS and MS error functions simultaneously. The eight braces on one side of the frame (at Grid A) of the model were paired into 4 unknown stiffness parameter groups based on their story level. The four mass parameter groups comprised of four floor slab panels at each of the 4 story levels.

Table 2.2 Damage Cases for Benchmark Structure

Damage Case	Damage Case Description	Error Function
1	100% axial stiffness loss in all 8 braces at first story	SS, SF, SSTR
2	30% bending rigidity (EI_{zz} and EI_{yy}) in columns at Grid A at first story level	SS, SF, SSTR
3	Reduction in mass of one of the 4 slab panels at roof level of fourth story from 550kg to 400kg	MS, MF
4	Reduction in mass of 4 floor slabs by 40% and 100% axial rigidity loss in 8 braces on story 1 through 4 on one side of the frame at Grid A	SS, MS

Load Cases and Measurement Locations

Diagnostic test loads were applied under four load cases. Table 2.3 shows the location and magnitude of loads for all four load cases. The prefix 'S' refers to a story. Reference is made to grid lines from Figure 2.2 .

Table 2.3 Static Load Application on Benchmark Structure

Load Case	Story - Load Location	Direction	Load (kN)
LC1	S1, S2, S3, S4 - Col. at Grid 1-A	X	1.0
	S1, S2, S3, S4 - Col. at Grid 3-A	X	1.0
LC2	S1, S2, S3, S4 - Col. at Grid 1-A	X	1.0
	S1, S2, S3, S4 - Col. at Grid 3-A	X	-2.0
LC3	S1, S2, S3, S4 - Col. at Grid 1-A	Y	1.0
	S1, S2, S3, S4 - Col. at Grid 1-C	Y	1.0
LC4	S1, S2, S3, S4 - Col. at Grid 1-A	Y	1.0
	S1, S2, S3, S4 - Col. at Grid 1-C	Y	-2.0

Displacement and strain measurement locations for static load cases used with damage case 1, 2, and 4 are listed in Table 2.4 and Table 2.5.

Table 2.4 Displacement Measurement Locations

Damage Case	Load Case	Story - Measurement Location	Disp. Direction		
			X	Y	Z
1	LC1	S1- Col. at Grid 1-C	✓		✓
		S1- Col. at Grid 3-C	✓		✓
	LC2	S1- Col. at Grid 1-C	✓	✓	
		S1- Col. at Grid 3-C	✓	✓	
2	LC1	S1, S4 - Col. at Grid 1-C	✓		✓
	LC2	S1, S4 - Col. at Grid 1-C	✓	✓	
		S1, S4 - Col. at Grid 3-C	✓	✓	
	LC3	S1, S4 - Col. at Grid 1-C		✓	✓
	LC4	S1, S4 - Col. at Grid 1-C	✓	✓	
		S1, S4 - Col. at Grid 3-C	✓	✓	
4	LC1	S2, S4 - Col. at Grid 1-A	✓		✓
		S2, S4 - Col. at Grid 3-A	✓		✓
	LC2	S2, S4 - Col. at Grid 1-A	✓	✓	
		S2, S4 - Col. at Grid 3-A	✓	✓	
	LC3	S1, S4 - Col. at Grid 1-A		✓	✓
		S2, S4 - Col. at Grid 3-A		✓	✓

Table 2.5 Strain Measurement Locations

Damage Case	Load Case	Story - Measurement Location	SG Locations (mm)		
			x	y	z
1	LC1, LC2	S1- Col. at Grid 1-C	450	-50	0
		S1- Col. at Grid 3-C	450	-50	0
2	LC1, LC2	S1- Beam b/w 1-B-C	625	-32	0
		S1- Beam b/w 3-B-C	625	-32	0
		S1- Col. at Grid 1-C	450	-50	0
		S1- Col. at Grid 3-C	450	-50	0
	LC3, LC4	S1- Beam b/w 2-3-A	625	-32	0
		S1- Beam b/w 2-3-C	625	-32	0
		S1- Col. at Grid 3-A	450	-50	0
		S1- Col. at Grid 3-C	450	-50	0

For estimating the unknown mass parameter for damage case 3 and 4, mode shapes 1 to 3 were utilized. Mode 1 and 2 are bending modes in the global Y and the X directions, respectively. Mode 3 is the torsional mode about Z axis. Mode shapes 1 through 3 and associated natural frequencies for undamped vibrations are shown in Figure 2.3

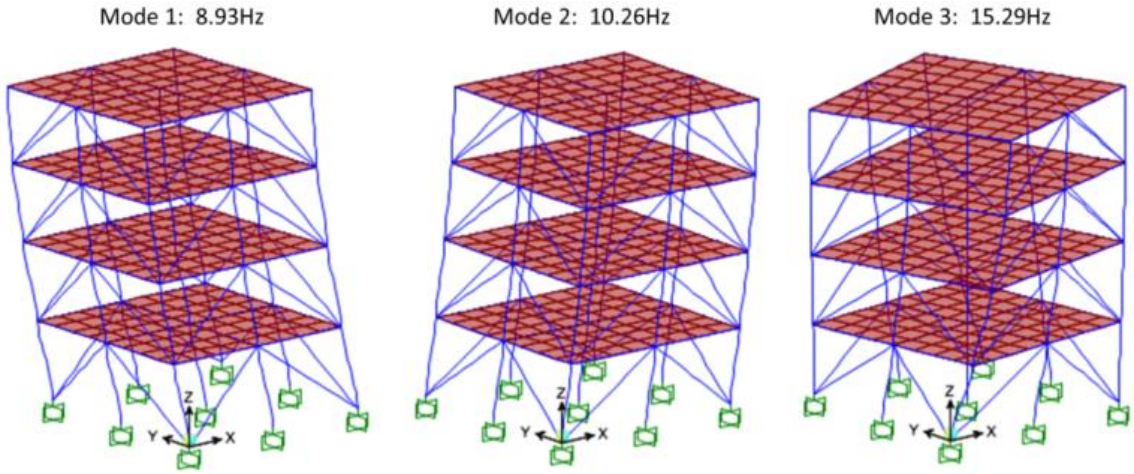


Figure 2.3 Fundamental Vibration Modes of the Benchmark Structure

Displacements were measured for modes 1, 2, and 3 at locations listed in Table 2.6.

Table 2.6 Damage Case 3 and 4 Displacement Measurements

Damage Case	Mode	Location	Disp. Direction		
			X	Y	Z
3	1	S1, S2, S3, S4 - Columns - Grid 1-C		✓	
	2	S1, S2, S3, S4 - Columns - Grid 3-C	✓		
	3	S1, S2, S3, S4 - Columns - Grid 1-C	✓	✓	
4	1	S2, S4 - Columns - Grid 1-A, 3-A		✓	✓
	2	S2, S4 - Columns - Grid 1-A, 3-A	✓		✓
	3	S2, S4 - Columns - Grid 1-A, 3-A	✓	✓	

Parameter Estimation Results and Discussion

The parameter estimation results for the four damage cases described above are presented here. For damage case 1, the parameter estimates are shown in Figure 2.4. The first bar represents the normalized value of initial guess of parameters. Bar number 2 is the true value of the parameter used to simulate NDT data. The last three bars are parameter estimates using simulated measurements for error functions SS, SF, and SSTR, respectively. It can be seen from Figure 2.4 that the EA parameters for damage case 1 were successfully estimated using all three static error functions.

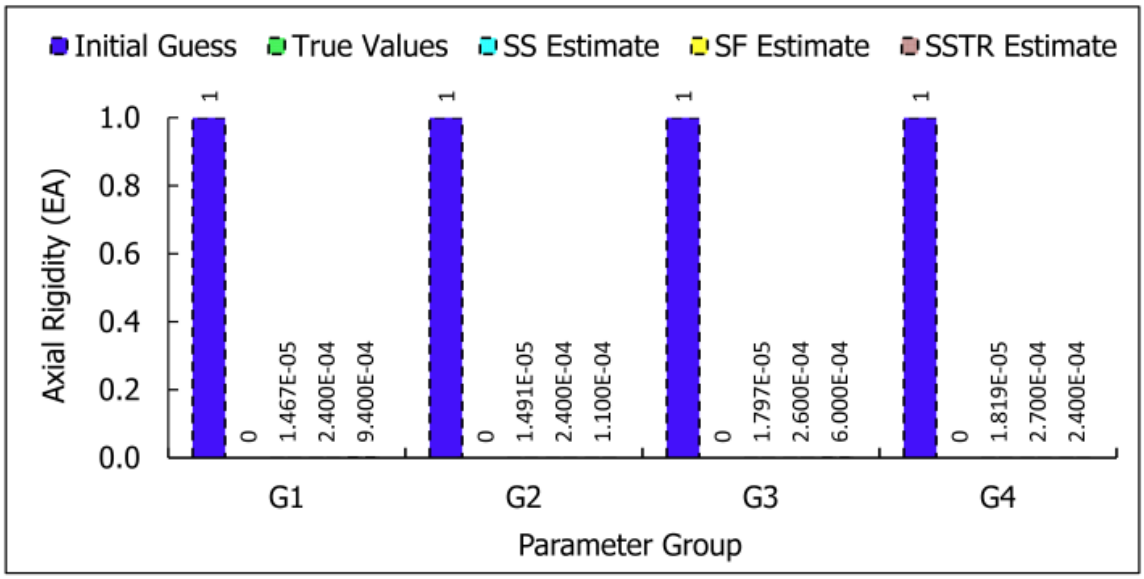


Figure 2.4 Damage Case 1: EA Parameter Estimates

In Figure 2.5 , results of updating EI_{zz} and EI_{yy} parameters for three base columns are presented.

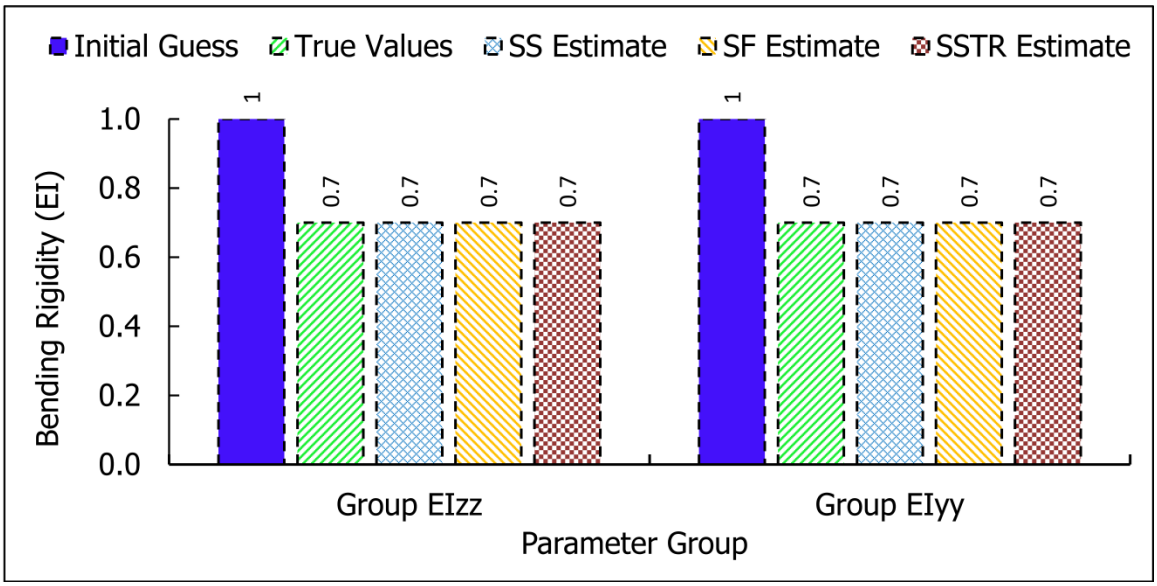


Figure 2.5 Damage Case 2: EI_{zz} and EI_{yy} Estimates

All three static error functions estimated the grouped bending rigidities the major and minor axes of the columns. The convergence criterion for relative ΔJ tolerance was maintained at 10E-10 while that for relative Δp tolerance was 10E-06 for all damage cases.

In damage case 3, the mass parameter of one roof slab panel was updated from an initial guess of 550kg to a final value of 400kg using modal stiffness and modal flexibility error functions. Modal stiffness error function converged faster than the modal flexibility error function.

Damage case 4 represents the combined stiffness and mass parameter updating capability of PARIS13.0 and the resulting parameter estimates are shown in Figure 2.6. The parameter groups G1 through G4 are the unknown EA parameters for braces. Groups G5 through G8 represent unknown floor mass parameters with an estimated value equal to true value of 0.6.

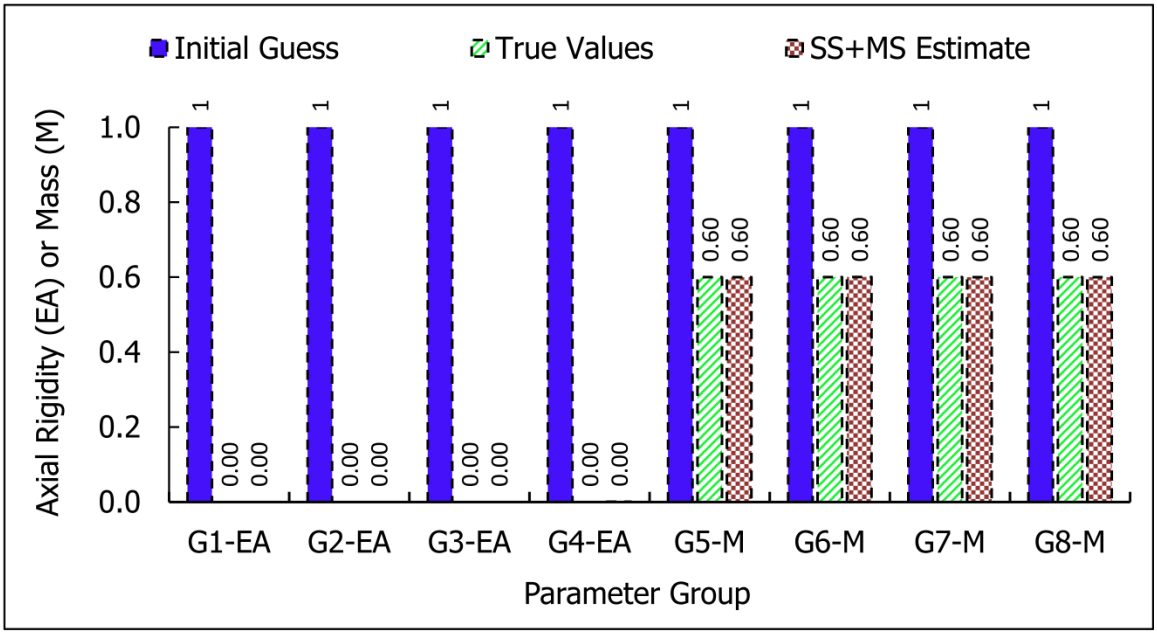


Figure 2.6 Damage Case 4: EA and Mass Estimates

Parameter Estimation in Presence of Measurement Errors

Figures 2.7a and 2.7b represents the parameter estimates for damage case 1 with SS and SF error functions in presence of measurement error. The final EA parameter estimates are shown on a

smaller *y-axis* scale to highlight the difference in magnitude of smaller numbers. A 2% normally distributed proportional error was added to both forces and displacement measurements. The EA parameter estimates using statistical error function normalization were more accurate even in presence of measurement error with a maximum of 0.0506 without normalization and 0.00855 with normalization. This proves the robustness of the statistical normalization method proposed by DiCarlo (2008).

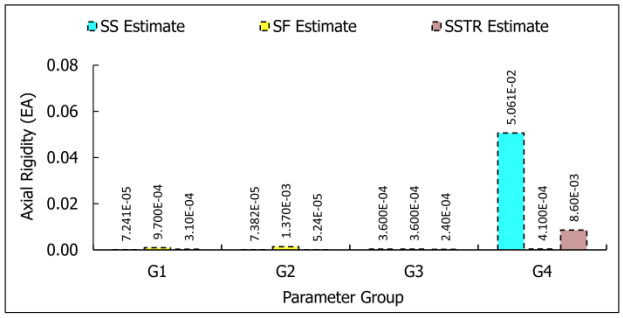


Figure 2.7a Damage Case 1: EA Estimates with Measurement Errors without Normalization

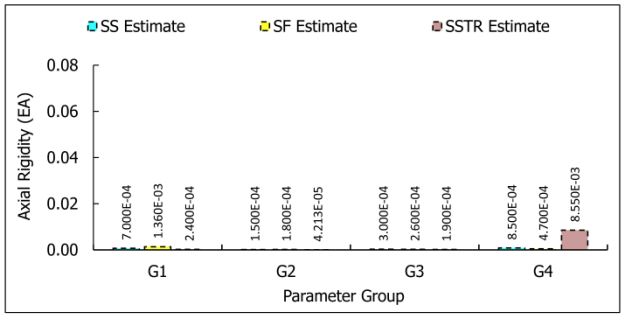


Figure 2.7b Damage Case 1: EA Estimates with Measurement Errors with Statistical Normalization

Table 2.7 compares the EA parameter estimates and number of iterations required for FE model updating in presence of measurement errors. The parameter estimates are shown for damage case 1 using SF and SSTR, first separately and then stacked together.

Table 2.7 Damage Case 1 with Stacked Error Functions

EA Estimates	Error Function(s)		
	SF	SSTR	SF + SSTR
G1	1.36E-03	2.40E-04	9.80E-04
G2	1.80E-04	4.21E-05	1.37E-03
G3	2.60E-04	1.90E-04	3.60E-04
G4	4.70E-04	8.55E-03	4.10E-04
Iterations	29	26	24

The merit of multi-response parameter estimation is seen in reduction of the number of iterations required when SF and SSTR error functions are stacked.

2.3.2 Powder Mill Bridge

The Powder Mill Bridge (PMB), Figure 2.8 , is a three span continuous steel girder bridge with a reinforced concrete deck in composite action. The bridge crosses the Ware River in the town of Barre, Massachusetts. The bridge was instrumented by researchers from Tufts University and the University of New Hampshire during its construction phase in 2009 as a part of the 'Whatever Happened to the Long Term Bridge Design' project sponsored by National Science Foundation Partnership for Education (NSF-PI).

Figure 2.8 Powder Mill Bridge (PMB) in Barre, MA

The six types of sensors installed on the PMB include 100 strain gauges, 36 steel temperature sensors, 30 embedded concrete temperature sensors, 3 ambient temperature sensors, 16 uniaxial accelerometers, 16 biaxial tilters, and 2 pressure plates (Sanayei et al. 2012). Instrumentation of

the PMB facilitates collection of different types of data in real-time. Three non-destructive diagnostic load tests were conducted on the PMB from 2009 through 2011.

2.3.2.1 Powder Mill Bridge Girder-3 Solid-Shell Model

Model Description

The first finite element model used with this example is of a single girder (Girder-3) of the bridge as shown in Figure 2.9 . This model is comprised of cuboid solid elements for bridge deck and haunch and quadrilateral plan thin-shell elements for web and flange of the W920x238 steel girder. The support conditions at the pier cap and abutment are modeled with joint springs based on Phelps (2010). Phelps calculated the elastomeric bearing pad stiffness based on 'Rotation Limits for Elastomeric Bearings' report by the National Cooperative Highway Research Program (NCHRP-596, 2008) and 'Earthquake-Resistant Design with Rubber' by James Kelly (1993). Solid elements used to model deck and haunch have reduced modulus of elasticity for concrete (E_c) in the negative moment region, which is considered to be 20% of the length of each span.

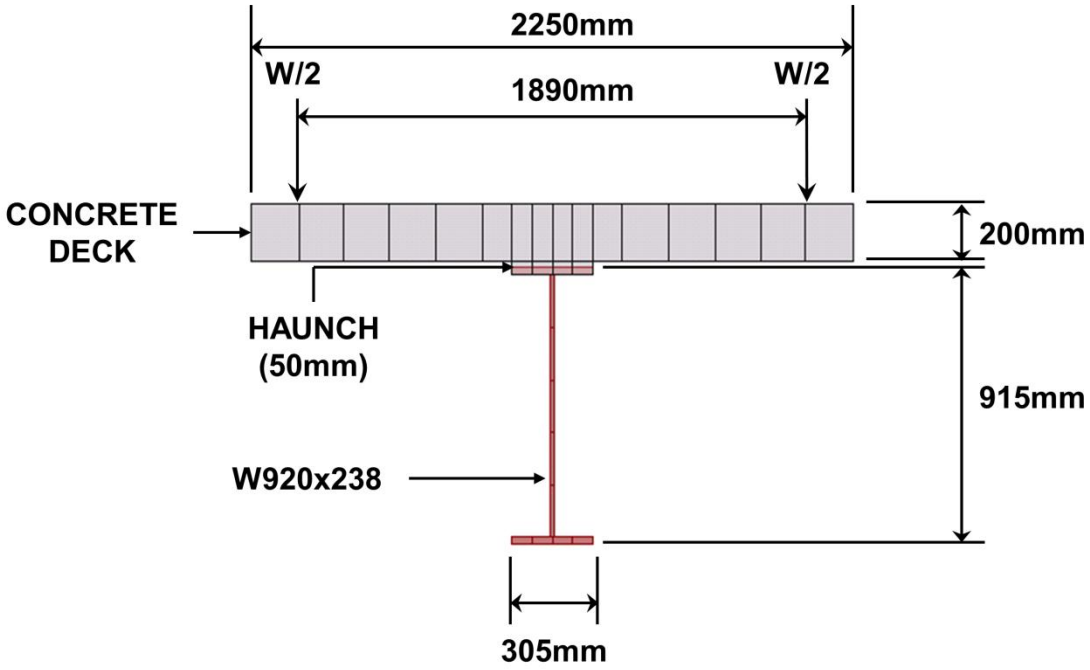


Figure 2.9 PMB Girder-3 Cross-Section with Contributory Deck

Load Cases and Measurement Locations

The most recent diagnostic crawl-speed load test conducted on the bridge used a test truck weighing 353.6kN (79.5kips). Truck path X2 (Iplikcioglu, 2012) required the test truck to cross the entire bridge while being centrally aligned over Girder-3 of the bridge. Truck path X2 was used to simulate NDT data for the PMB Girder-3 Solid-Shell Model.

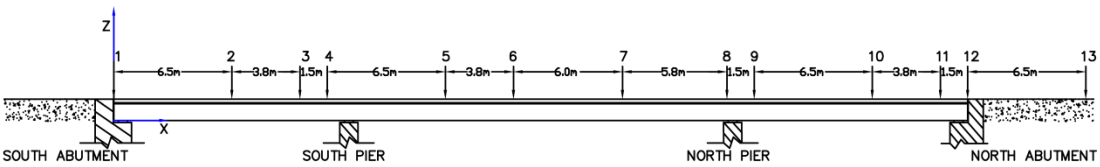


Figure 2.10 Test Truck Front Axle Locations on PMB Girder-3

In order to simulate the crawl speed NDT, the truck location on the bridge was discretized into 13 static load cases as shown in Figure 2.10 . The load test truck dimensions and axle spacings were rounded to the nearest 0.25m dimension so the truck loads could be applied as joint loads in the SAP2000 model. The displacement and strain measurement locations were chosen to be the central node at the bottom flange at the center of the end spans of the bridge and at quarter points at the longer central span. Since PARIS13.0 is designed to accommodate different number of measurements in each load case, only observations from the span on which the truck was located were used with that load case, respectively.

The PMB Girder-3 Solid-Shell model has 9,168 nodes which represent a total of 35,256 DOFs. The model was however analyzed in XZ-plane with two translational and one rotational DOF thus reducing the active DOFs to 27,504. Since the size of the stiffness matrix is dependent on the number of active DOFs, a reduction in the number of active DOFs in a model reduces computational time, thus speeding up the FE model updating process.

Damage Scenarios

This verification example included model updating in two simulated damage scenarios with no measurement errors. The damage scenarios and parameter estimation results are presented next.

Damage Case 1

In the first damage case, loss of 20% horizontal translational stiffness (k_{XX}) and rotational stiffness ($k_{\theta Y}$) was considered to simulate joint deterioration. The damage was assumed to occur similarly at both pier and abutment locations. The bottom flange of the girder has four shell elements across its width thus requiring joint springs at 5 points. All five spring stiffnesses were grouped together as a single parameter at each of the four support locations (two piers and two abutments). PARIS13.0 was used to accurately estimate the unknown stiffness values all three static error functions. In this case the static stiffness and static strain error functions estimated the unknown bending rigidities more accurately than the static flexibility error function.

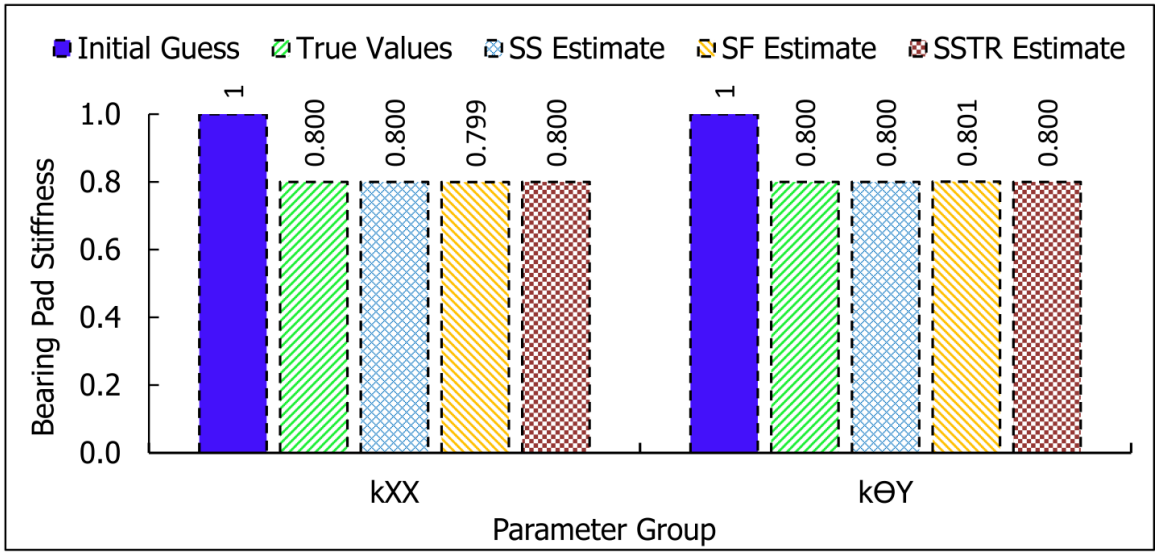


Figure 2.11 Bearing Pad Stiffness (k_{XX} and $k_{\theta Y}$) Estimates

The unknown parameters in the first damage case were estimated using all three static error functions. Mean of translational and rotational stiffness values at four support locations is shown in Figure 2.11 . SS and SSTR functions performed better in terms of accuracy of the parameter estimates. The number of iterations required for model updating for damage case 1 were 27 for the SS error function, 96 for SF error function, and 37 for the SSTR error function.

Damage Case 2

A loss of 20% of the girder's stiffness was assumed to occur at the entire south span of 11.75 m length as a global deterioration case. Since girder is comprised of shell elements, the damage in finite element model would then be reflected in reduction of modulus of elasticity (E_s) of steel. The 624 shell elements making up the south span deck of the FE model were grouped into 8 groups with 78 elements in each group. The elements were grouped along the length of the girder with equal divisions. Static strain error function was used to estimate 20% reduction in E_s.

Unlike translational displacement measurements, strain measurements are reference-independent making it a preferred measurable response quantity. Strain gauges are also a low cost option when compared to both contact and non-contact devices required for measuring displacements. The PMB Girder-3 Model was only updated using SSTR error function for damage case 2. The mean of estimated parameters values of E_s for shell elements was 0.8035 from the expected value of 0.8 with a 0.0148 standard deviation.

2.3.2.2 Powder Mill Bridge Frame-Shell Model

A full-scale FE model of the PMB comprised of frame and shell elements was updated for two damage scenarios. Frame elements were used to model the six main girders of the bridge as well as the diaphragms and outriggers at the north end of the bridge. Offset was applied to frame elements for modeling supports at the bottom flange of the girder. The shell elements constitute the concrete bridge deck. The shell elements are offset upwards by half the shell thickness to model the deck resting on top flange of the girders. The PMB Frame-Shell model is shown in plan in Figure 2.12 with gridlines 1 through 6 representing girder numbers and S1 through S4

representing abutment (S1 and S4) and pier support (S2 and S3) locations. Darker longitudinal lines indicate girders and darker transverse lines indicate diaphragms.

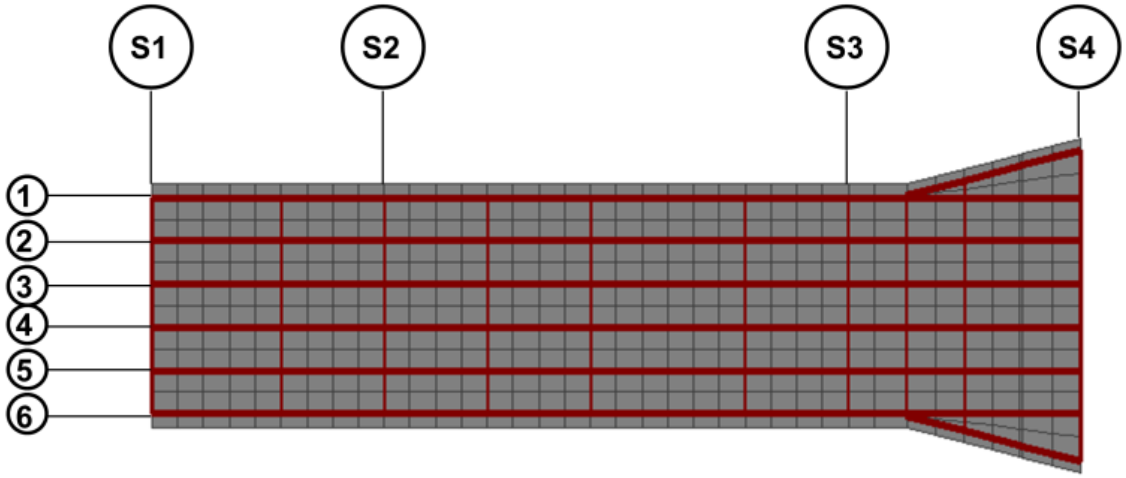


Figure 2.12 PMB Frame-Shell Model Plan

The static load cases were applied in a fashion similar to that for the solid-shell girder model with the addition of four test truck paths across the width of the bridge. Eight sets of truck loads were modeled in each of the five test truck paths on the bridge.

Damage Scenarios

Damage Case 1

A damage case that considers a theoretical 20% reduction in bending capacity of girders 4 and 5 is considered first. The entire length of these two girders is assumed to have a reduced EI_{33} value. Since each of the two girders in the model is comprised of 36 smaller frame elements connected at ends, the elements were grouped together into three groups where each group represents one span length. Starting with the initial guesses of EIs of girders 4 and 5, the

unknown moment of inertias converged to values very close to the true values of EIs that was used for simulating measurements.

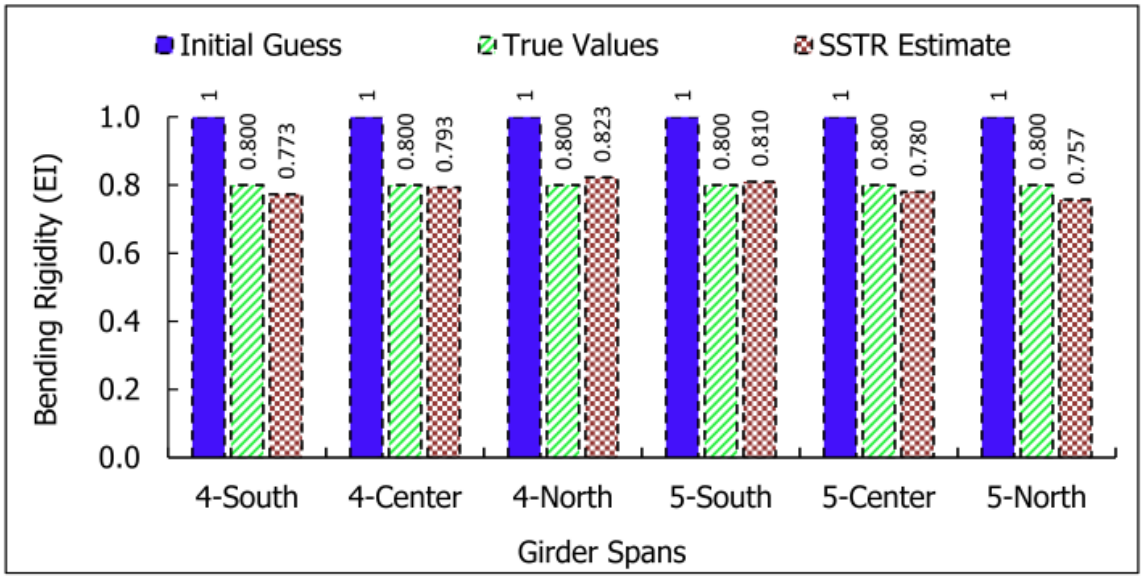


Figure 2.13 EI_{33} Estimates from SSTR Error Function

The results from model updating for frame-shell model using SSTR error function are shown in Figure 2.13 . The bending rigidity of frame elements representing girders 4 and 5 converged with a mean error of -1.33%. However maximum error in Group 3 and Group 6, which represents the north spans of the girders, was higher than for the center and south spans. This could be because of smaller deformations from increased stiffness at the north end of the bridge leading to lower sensitivity in estimating parameter values.

Damage Case 2

Damage to the bridge deck is a prominent type of damage in bridges. This type of damage can be caused by traffic loads, environmental factors including diurnal and seasonal temperature variation, and water infiltration through expansion joints or poor quality control during construction. To simulate this damage scenario, a cluster of 30 shell elements equivalent to deck

surface of 44.04m^2 are grouped together to simulate damage. It represents a patch beginning at length 3.92m from south end till 11.75m length and lies on the right lane looking north. The damage level is taken to be 50% of the initial shell elasticity modulus. The concrete modulus of elasticity for all five deck groups was successfully estimated.

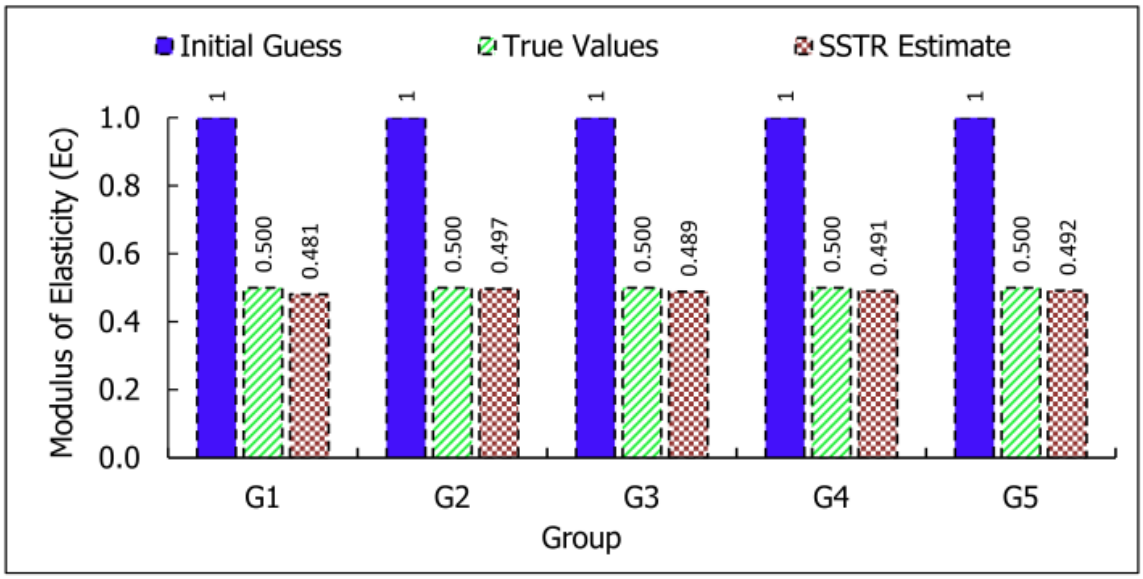


Figure 2.14 E_c Estimates from SSTR Error Function

Parameter estimates for loss of 50% deck stiffness for a cluster of shell elements described under damage case 2 were quite accurate. The model updating process was automatically terminated when relative change in parameter values between two iterations became less than 10E-12. It took 13 iterations for the convergence criteria to be met.

Chapter 3

PARameter Identification System (PARIS13.0) Design and Development

---

3.1 Original PARIS (PARameter Identification System)

PARIS is the acronym for Parameter Identification System. PARIS is the original FE model updating program developed by Professor Masoud Sanayei with his graduate students at Tufts University. It is a MATLAB based program with an elementary Finite Element Analysis (FEA) solver for parameter estimation and FE model updating capability for planar and spatial truss and frame structures. This section provides background of the original PARIS software with both its FE model updating capabilities and limitations and how it served to be a motivation for development of PARIS13.0.

The sequence of steps involved in the model updating process in original PARIS is laid out it in Figure 3.1.

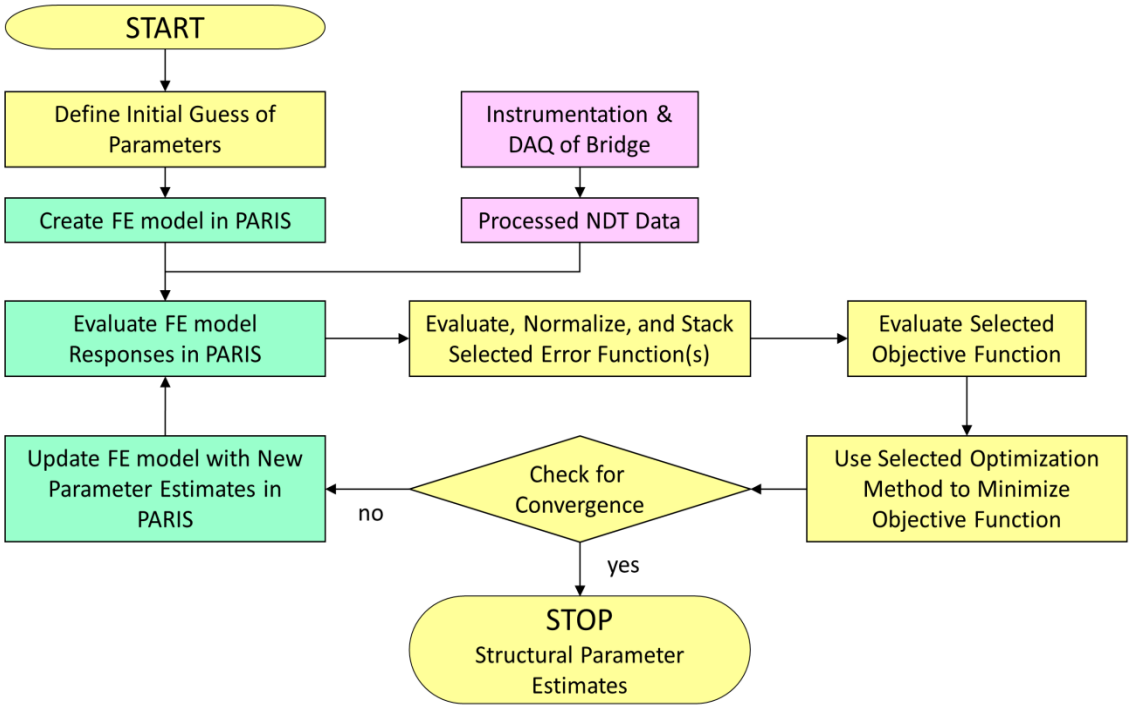


Figure 3.1 PARIS Flowchart

3.1.1 Original PARIS Capabilities

PARIS is a custom built MATLAB based FE model updating program which uses static and modal measurements for estimating stiffness and mass parameters of the finite element model at the element level. The program uses static and modal error functions, *e(p)*, as a measure of residual between predicted response and measured response of the structure. By minimizing the objective function, *J(p)*, a scalar quantity obtained by pre-multiplying the error function vector with its transpose, PARIS is capable of estimating unknown parameters of two and three dimensional trusses and frames. It is based on computation of analytical sensitivity matrix, *S(p)*, for each error function and presents the user with the choice of stacking different error functions for multi-response parameter estimation. PARIS has a wide array of options for parameter estimation. It includes the option of using genetic algorithms for parameter estimation as well. Monte Carlo simulation which is used to study error behavior in parameter estimation process for

algebraically non-linear systems is also available in PARIS. The original PARIS program also features a method for determining optimum sensor placement based on the works of Sanayei and Javdekar (2002) and DiCarlo (2008).

3.1.2 Original PARIS Limitations

PARIS has its own FEA solver for truss and frame elements which is also written in MATLAB. This largely limits the model updating process to the use of simplified equivalent frame model of a structure which in most cases does not accurately capture the response of the structure. Moreover, PARIS requires manual text based entry for FE model creation. Input data required to establish geometric configuration needs user input in the form of joint coordinates and element connectivity, which proves to be rather cumbersome for creating large scale models. Although PARIS generates a line diagram of the structure post finite element analysis, the lack of a graphical user interface inhibits visualization of structure for both model creation and validation. PARIS also requires calculation of degree of freedom number at which response is measured or where a static load is applied. Considering each node has six associated degrees of freedom, it quickly becomes impractical to keep a record of degrees of freedom and their sequence for a full scale model. More difficult is to track any changes made to the model.

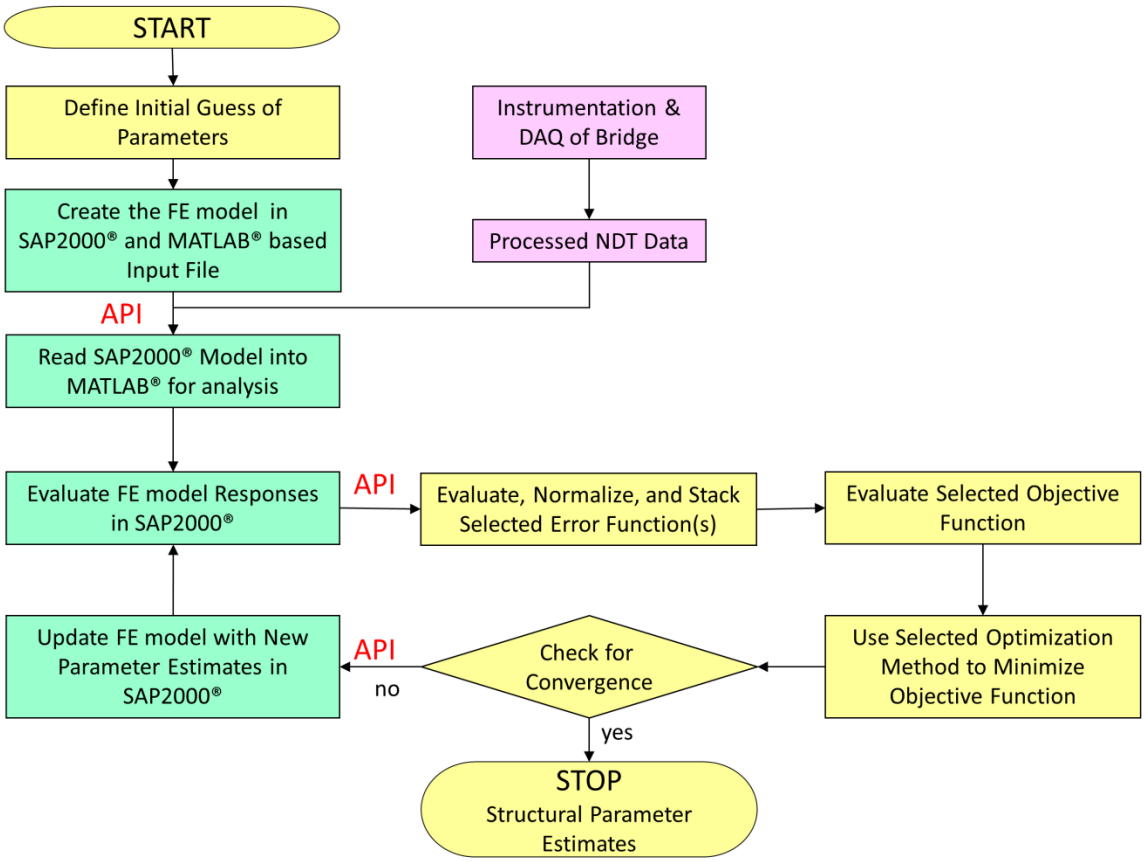
3.2 PARIS13.0 (PARameter Identification System)

3.2.1 Feasibility Study

Zhu (2009) conducted a feasibility study on interfacing SAP2000 with MATLAB and came up with a preliminary module which could be used to update finite element models based on simulated NDT data. His work provided a foundation for systematic development of PARIS13.0.

3.2.2 Program Capabilities

As shown in Figure 2.1 , the flow of processes in PARIS13.0 resemble closely with that of original PARIS.



Duplicate Figure 2.1 PARIS13.0 Flowchart

Table 3.1 summarizes the capabilities of PARIS13.0 in comparison to the original PARIS. PARIS13.0 has the most advantage in the broader range of finite elements available through SAP2000. In PARIS13.0, only five of seven error functions have been kept. The two static and modal stiffness based (SF2 and MF2) error functions which are eliminated produced same results as SF1 and MF1 error functions respectively. Among various normalization methods, only those normalization methods are retained in PARIS13.0 which have proven to be more effective in producing better parameter estimates. In PARIS13.0, the emphasis is on using MATLAB Optimization Toolbox because of its better performance for full-scale FE model updating. Analytical sensitivity based optimization option has also been retained in PARIS13.0 but only with Gauss Newton and Steepest Descent methods for solving inverse problems.

Table 3.1 PARIS and PARIS13.0 Comparison

Feature	Description	PARIS	PARIS13.0
Finite Element Modeling	SAP2000 with API	×	✓
Finite Elements	Truss	✓	✓
	Frame	✓	✓
	Shell (4 Node – Quadrilateral – Planar)	×	✓
	Solid (8 Node – Cuboid)	×	✓
	Link Element (2 Node)	✓	✓
	Joint Spring	✓	✓
Error Functions	Static Stiffness (SS)	✓	✓
	Static Flexibility I (SF1)	✓	✓
	Static Flexibility II (SF2)	✓	×
	Static Strain (SSTR)	✓	✓ (frames, shells)
	Modal Stiffness (MS)	✓	✓
	Modal Flexibility I (MF1)	✓	✓
	Modal Flexibility II (MF2)	✓	×
Normalization	Parameter	✓	✓
	Measured Data	✓	×
	Initial Value Error Function	✓	✓
	Previous Step Error Function	✓	×
	Standard Deviation Error Function	✓	×
	Maximum Likelihood Estimator	✓	✓
Multi-response Parameter Estimation	Error Function Stacking	✓	✓
Optimization Techniques	*Hill Climbing Methods*	✓	✓
	i) Generalized Inverse	✓	×
	ii) Least Squares/ Gauss Newton	✓	✓
	iii) Steepest Descent	✓	✓
	iv) Standard Conjugate Gradient	✓	×
	v) BFGS Quasi Newton	✓	×
	vi) Gauss Newton w/ Line Search	✓	×
	MATLAB Optimization Toolbox	×	✓
	Genetic Algorithms	✓	×
Modeling Error		✓	×
Measurement Error	Absolute and Proportional Errors	✓	✓
	Uniform and Normal Errors	✓	✓
	Monte Carlo Simulation	✓	×
Reading in NDT data		✓	×
Parameter grouping		✓	✓

3.2.3 Planning and Design

Writing a full scale finite element model updating program presented a variety of challenges. It was essential that the program was thought through well in the planning stage to reduce rework at a later stage.

Design Flowcharts

Flowcharts are an excellent method to document complex processes. In the first phase of planning, a set of design flowcharts were developed. The global design of PARIS13.0 is shown in Figure 2.1 . The flowcharts presented here expand the processing boxes in the global flowchart.

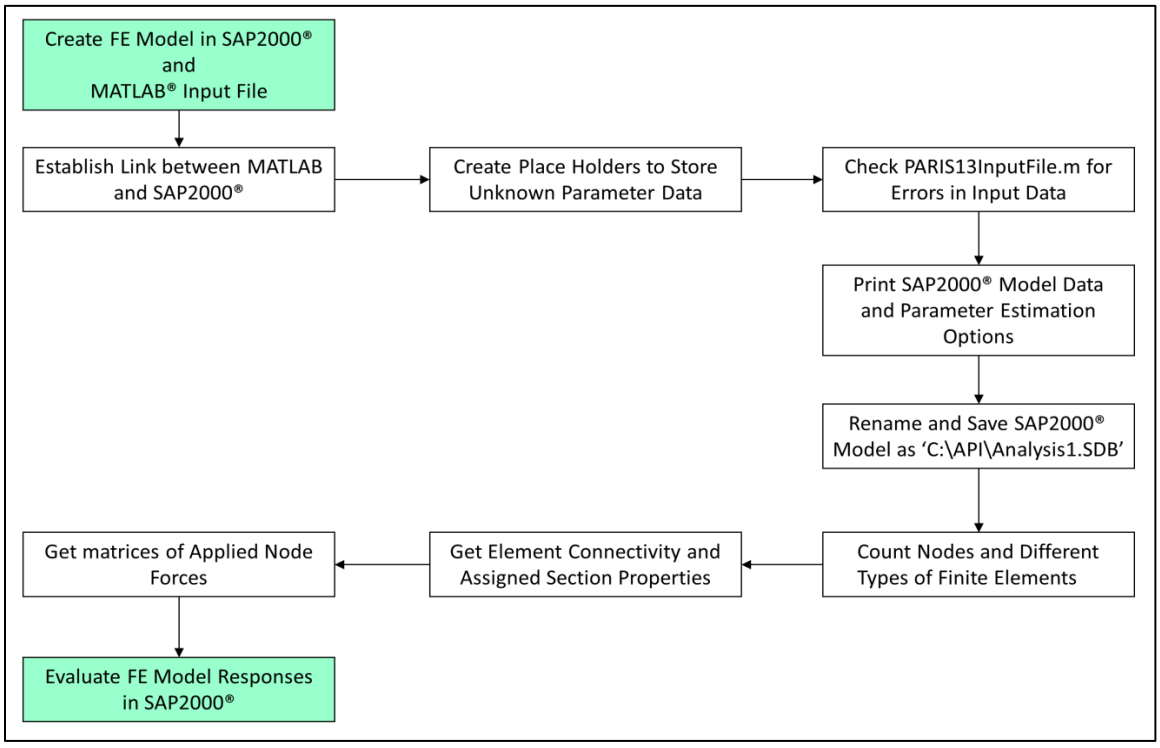


Figure 3.2 Expanded Flowchart Block 1: Read SAP2000 Model into MATLAB

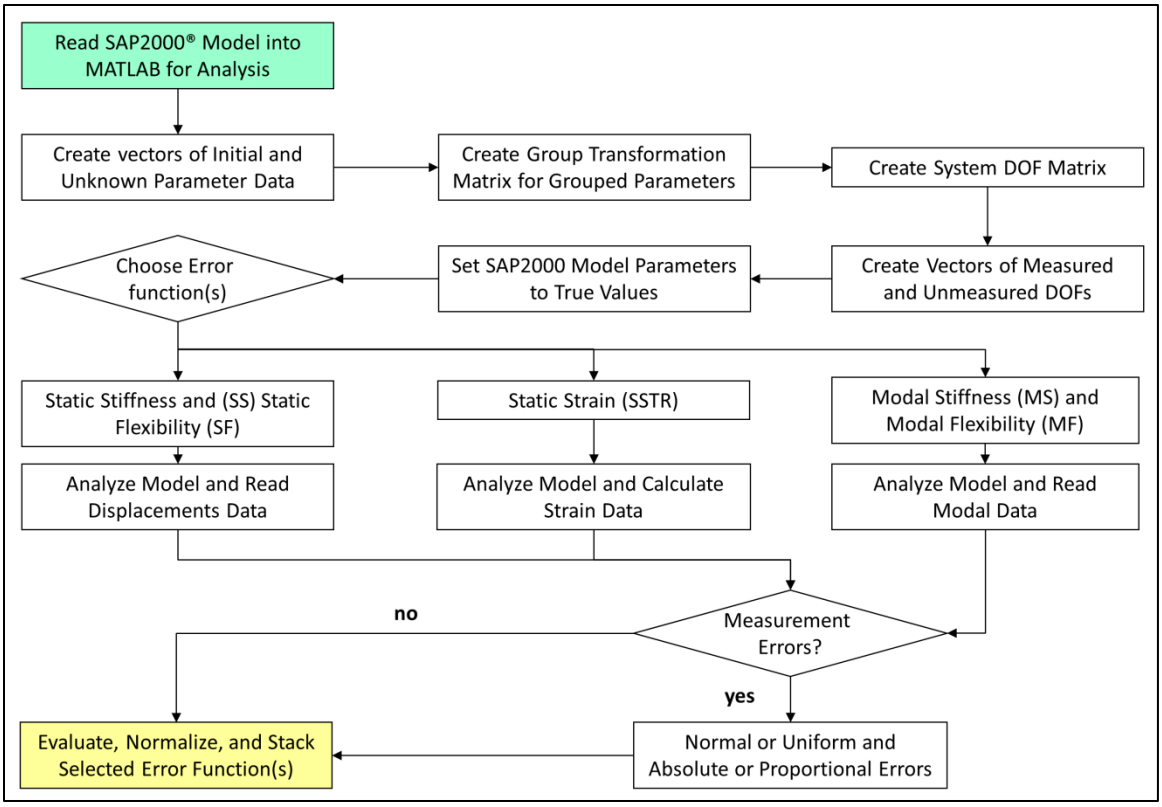


Figure 3.3 Expanded Flowchart Block 2: Evaluate FE Model Responses in SAP2000

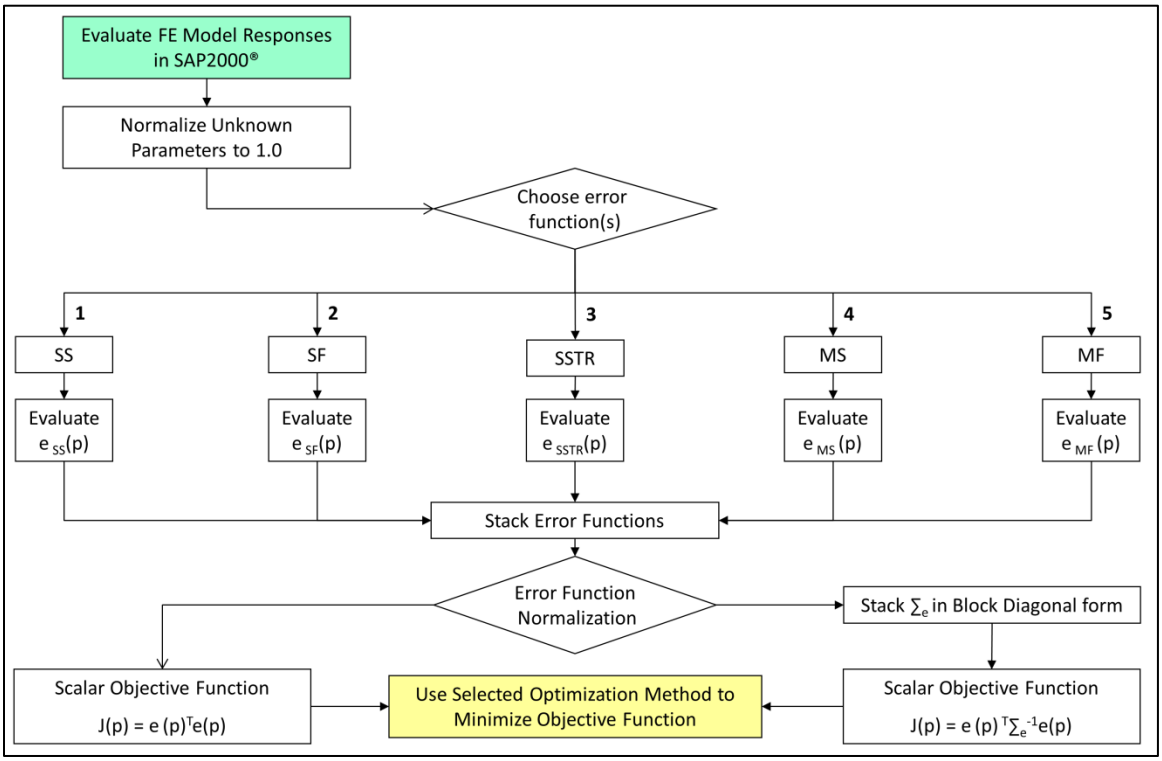


Figure 3.4 Expanded Flowchart Block 3: Evaluate, Normalize, and Stack Error Functions

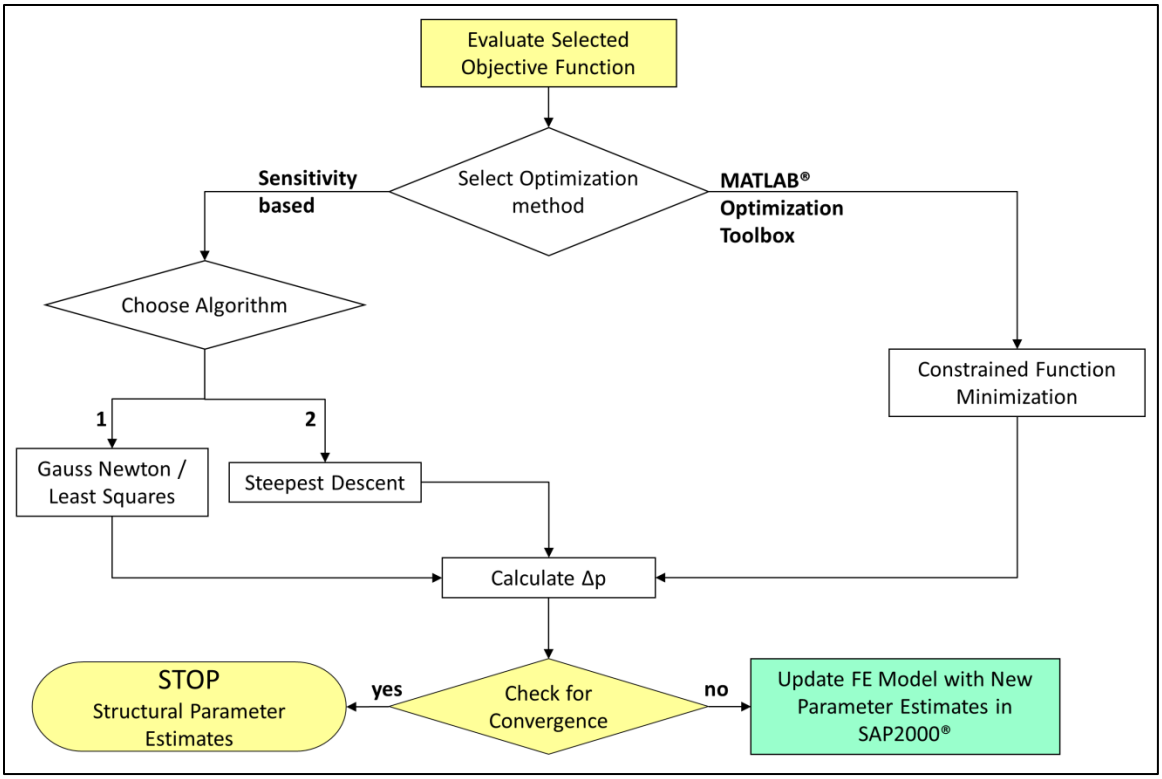


Figure 3.5 Expanded Flowchart Block 4: Minimize Objective Function

Pseudo-code

A pseudo-code was developed next to document the flow of main processes of the program. An important consideration in program design was to make it modular i.e. to design it as a linked set of subroutines instead of a single sequential program. This was done to make editing simpler for smaller subroutines and reduce total number of lines in the program by avoiding repetition of same block of commands. Another reason was to allow for addition of modules in the future versions of the program in an efficient manner. The pseudo-code shown below forms the basis of PARIS13.0.

PARIS13.0 PSEUDOCODE

STANDARDINPUTFILE.M

Input all parameter data and optimization options
Call PARIS 13.0

PARIS13.M

*****Initial data checks and SAP2000 Model*****
Call PRELIMINARYCALCS (create data structures for storing unknown parameter data)
Call DATACHECK (check input data for any errors or inconsistencies)
Call DATAECHO (display input data on screen)
Call READSAPMODEL (read model data from SAP2000 through API)
*****Prepare model for parameter estimation*****
Call NORMPARA (normalize and vectorize unknown parameters, create identifier vector and grouping matrix)
Call PREPMAIN (prepare the system DOF matrix and prepare vectors for measured and unmeasured DOF)
*****Simulate Test Data****
Call SIMNDTDATA (simulate static and dynamic test data)
Call GEN_ME_ERRORS (generate measurement errors)
Call ADD_ME_ERRORS (corrupt simulated data by adding measurement errors)
Call PREPSD (create standard deviation matrices for measurement errors in forces and displacements)
Call PREPMAIN (create error function vectors and save stiffness and mass matrices)
Call SDEPO (create standard deviation and covariance matrices for error functions)
*****Start parameter estimation*****
ANALYTICAL SENSITIVITY ANALYSIS (SensitivitySol)
Call MAKES (create analytical sensitivity matrices)
Call STACKNORMEFN (include measurement errors) or STACKEFN (no measurement errors)
Call SOLVEDELTAP (use hill climbing methods)
$p_{i+1} = p_i + \Delta p_i$
*****Check Convergence*****
If convergence is reached; CLOSE SAP2000 and QUIT
Else; call UPDATEMODEL (update model for next iteration)
NUMERICAL SENSITIVITY ANALYSIS ((MatlabSol))
Define OPTIMSET (define optimization options)
*****Check for parameter grouping*****
If PGFLAG = 1; call OBJFUNGROUP (create function for calculation of objective function J(p))
Elseif PGFLAG = 0; call OBJFUN (create function for calculation of objective function J(p))
Call FMINCON (minimize objective function subject to optimization options)
$p_{i+1} = p_i + \Delta p_i$
*****Check Convergence*****
If convergence is reached; CLOSE SAP2000 and FMINCON and QUIT
Else; update model and prepare for next iteration within OBJFUNGROUP or OBJFUN
*****End parameter estimation*****
Print PARHISTORY (parameter values at each iteration) Note: details only for SensitivitySol
END PARIS13.M

Chapter 4

PARIS13.0 Verification Examples: Additional Information

4.1 Introduction

This chapter provides additional details and relevant information about FE models used as verification examples in Chapter 2.

4.2 IASC-ASCE SHM Benchmark Structure

The IASC-ASCE SHM Benchmark Structure used in this research is based on Phase-I of the study conducted by the IASC-ASCE SHM Task Group. The Phase-I study was based on simulated data from an analytical structural model of a scale-model structure at the Earthquake Engineering Research Laboratory at the University of British Columbia. Since PARIS13.0 is currently only capable of using simulated data for model updating, it was thought that using an analytical model in the likeness of the IASC-ASCE SHM Benchmark Structure would make a relevant verification example.

The model used in this research is slightly different than the analytical model used by the IASC-ASCE SHM Task Group. Originally, two finite element models were developed by the task group. The first model was a 12 DOF shear-building model in which only two horizontal translations and one rotation were permitted for each floor. The second model had 120 DOFs which constrained all nodes at each floor for the same horizontal translation and in-plane rotation (Johnson et al. 2004). The structural model used here has 1944 DOF. Each floor is discretized into 64 square sections which connect to 81 nodes. This adds up to a total of 324 nodes at 4 floors. All translational and rotational DOFs are made active for all nodes except the 9 nodes at base supports which are all fixed. The braces, like in the original IASC-ASCE SHM Benchmark Structure, are assumed to have no bending rigidity and are modeled as axial force members.

4.2.1 Section Properties

The section properties of the one-third scale structural model described in Section 4.2 are given in Table 4.1.

Table 4.1 Section Properties of Structural Members

Section Property	Column	Beam	Brace
Type	B100×9	S75×11	L25×25×3
Cross-sectional Area A (m^2)	1.133×10^{-3}	1.430×10^{-3}	0.141×10^{-3}
Moment of Interia I_{yy} (m^4)	1.970×10^{-6}	1.220×10^{-6}	0
Moment of Interia I_{zz} (m^4)	0.664×10^{-6}	0.249×10^{-6}	0
Torsional Constant J (m^4)	8.010×10^{-9}	3.820×10^{-10}	0
Young's Modulus E (N/m^2)	2.000×10^{11}	2.000×10^{11}	2.000×10^{11}
Shear Modulus (N/m^2)	7.692×10^{10}	7.692×10^{10}	7.692×10^{10}
Mass Density ρ (kg/m^3)	7800	7800	7800

These properties are based on description of the Phase-I IASC-ASCE SHM Benchmark Structure by Black and Ventura (1998) and Johnson et al. (2004).

4.2.2 Damage Scenarios

Damage Case 1

Damage case 1 considers all 8 braces at the first story to have complete loss of axial rigidity. The bracing members assumed to be completely damaged are shown in Figure 4.1.

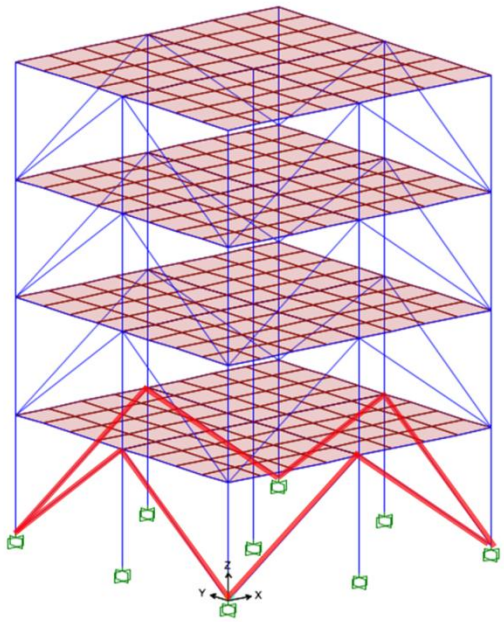


Figure 4.1 First Story Brace Damage

Damage Case 2

The base columns assumed to have 30% reduction in their bending rigidity are shown in Figure 4.2 .

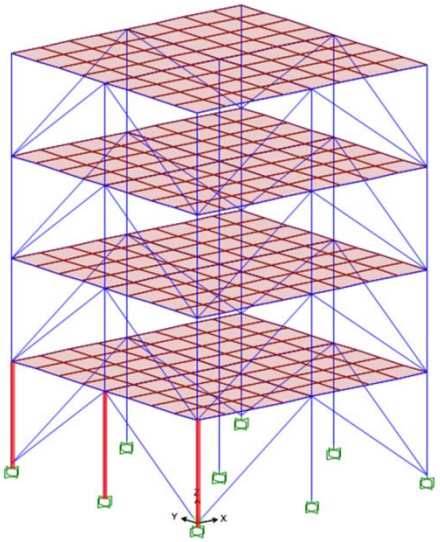


Figure 4.2 Grid-A Base Columns Damage

Damage Case 3

Sixteen shell elements that comprise the 550kg slab panel are highlighted in Figure 4.3 . Damage case 3 considers reduction of mass of this panel from 550kg to 400kg.

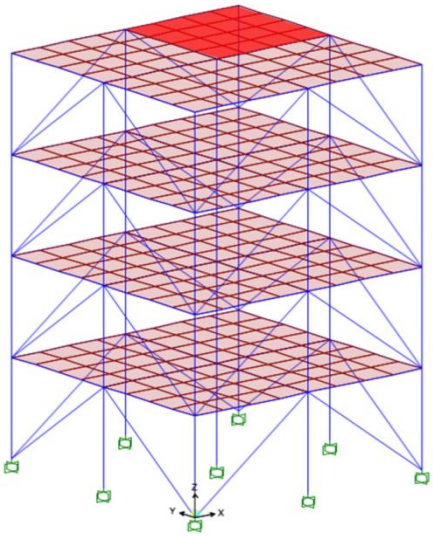


Figure 4.3 Slab Panel Mass Reduction

Damage Case 4

Damage case 4 combines both the stiffness and mass parameters in one parameter estimation problem. The eight braces (highlighted red) shown in Figure 4.4 on Grid-A from story 1 through story 4 are grouped into 4 stiffness parameters based on their story level. The mass of four slab panels at each floor is considered to be an unknown parameter group, adding up to a total of four mass parameter groups (highlighted green).

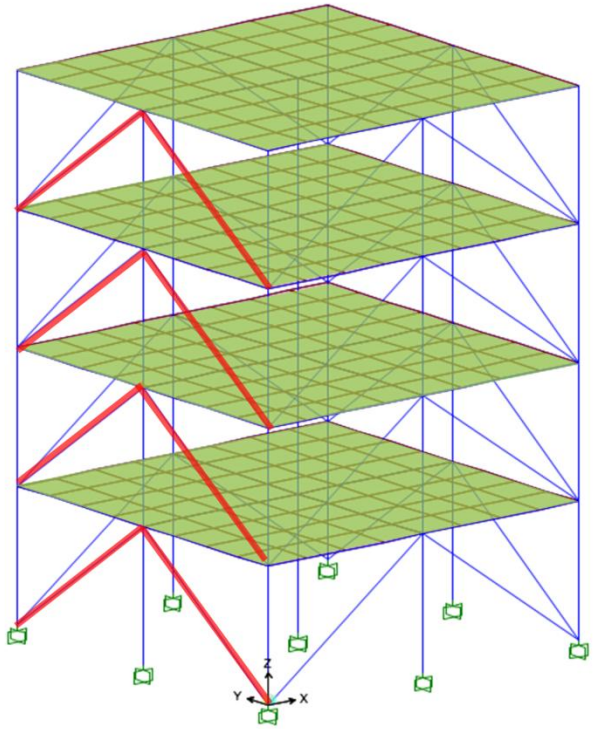


Figure 4.4 Unknown Parameters for Damage Case 4

4.2.3 Static Load Cases

The four static load cases used with static error functions for parameter estimation described in Table 2.3 are shown in Figure 4.5a through Figure 4.5d.

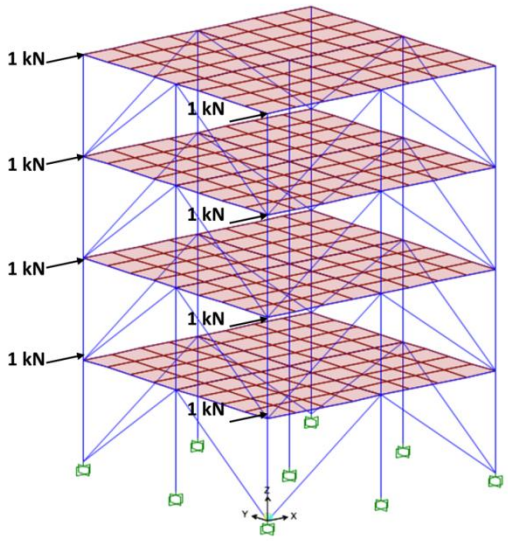


Figure 4.5a SHM Benchmark Structure: Load Case 1

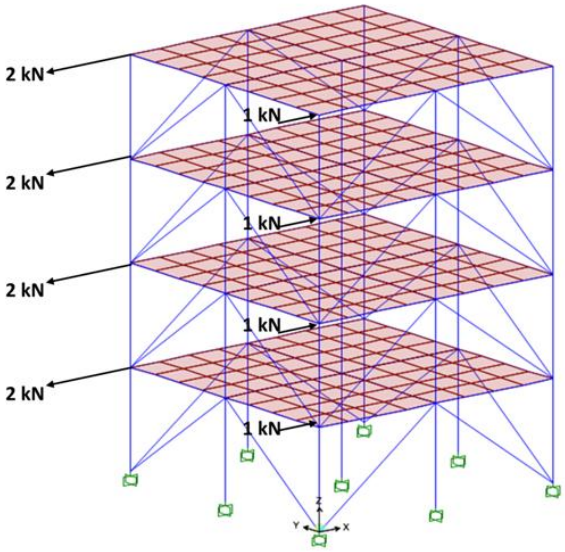


Figure 4.5b SHM Benchmark Structure: Load Case 2

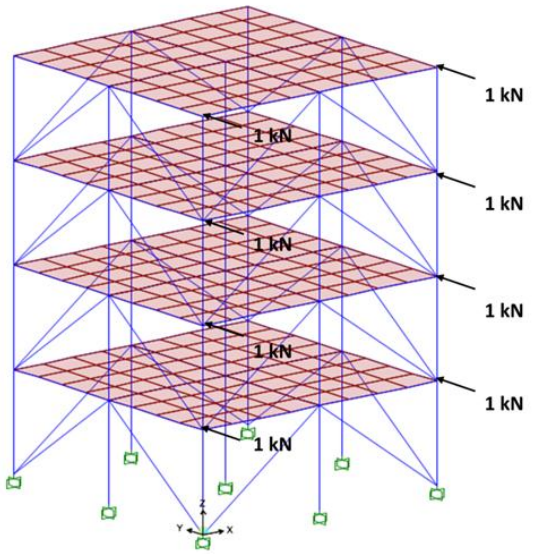


Figure 4.5c SHM Benchmark Structure: Load Case 3

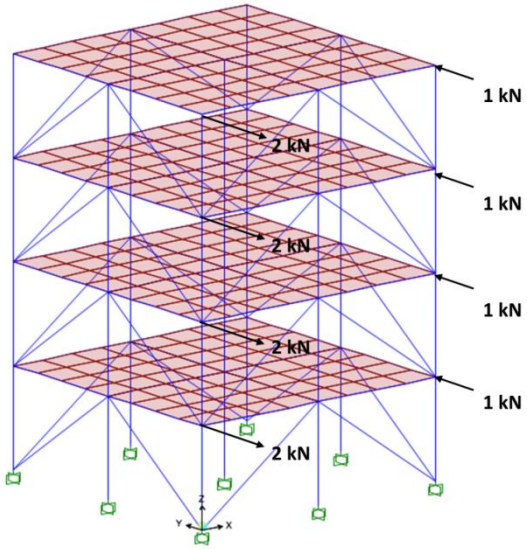


Figure 4.5d SHM Benchmark Structure: Load Case 4

4.2.4 Parameter Estimation Results using Analytical Sensitivity

The unknown parameters for the first damage scenario were estimated using both analytical and numerical sensitivity methods. Section 2.3.1 shows the parameter estimates using numerical sensitivity. Numerical sensitivity analysis using MATLAB Optimization Toolbox was found to

be more effective in both the time required and the accuracy of results. The parameter estimates for damage cases 1 using analytical sensitivity matrix *S(p)* in equation 2.11 are shown in Figure 4.6 Figure 4.6 Figure 4.6 Figure 4.6

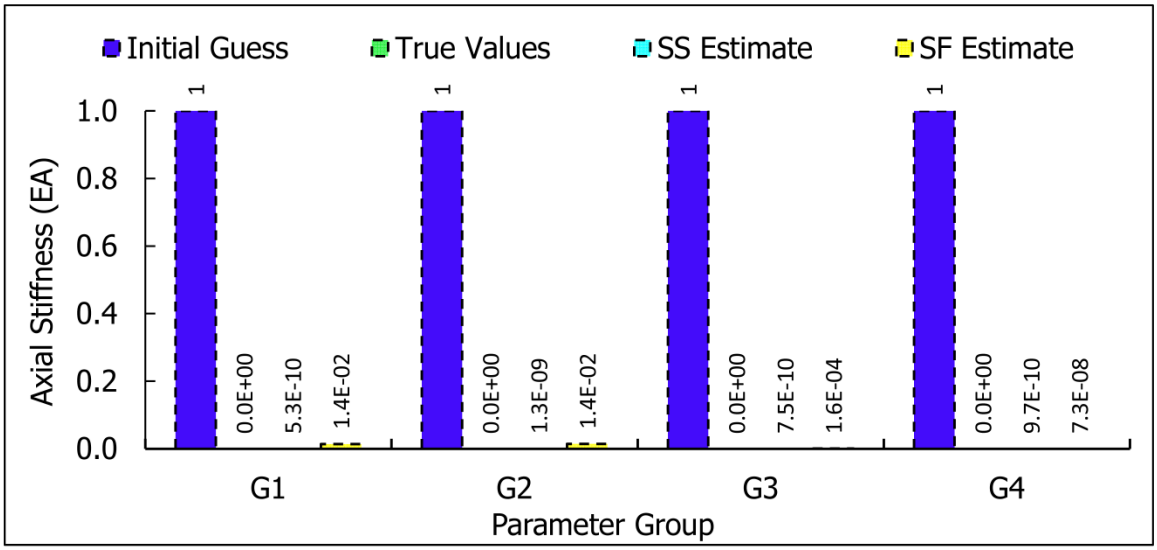


Figure 4.6 Damage Case 1: EA Estimates using Analytical Sensitivity

4.3 PMB Girder-3 Solid-Shell Model

The solid-shell FE model of Girder-3 of the Powder Mill Bridge contains 2470 area elements and 3800 solid elements. Quadrilateral shell (area) elements are used to model the top and bottom flange of the bridge girder. The haunch and the deck are modeled using 8 node cuboid solid elements. Table 4.2 shows the section properties assigned to shell and solid elements.

Table 4.2 Section Properties of elements in PMB Girder-3 Model

Section Property	Flange	Web	Positive Moment Region		Negative Moment Region	
			Deck	Haunch	Deck	Haunch
Material	Steel	Steel	Concrete	Concrete	Concrete	Concrete
Type	W920×238	W920×238	N.A.	N.A.	N.A.	N.A.
Thickness (mm)	25.9	16.5	200	50	200	50
Young's Modulus E (MPa)	200	200	27.4	27.4	18.0	18.0
Shear Modulus (MPa)	76.9	76.9	11.4	11.4	7.5	7.5

4.3.1 Elastomeric Bearing Pads

Based on the calculations by Phelps (2010), the stiffness values listed in Table 4.3 were used to model bearing pads as translational and rotational springs in SAP2000.

Table 4.3 Bearing Pad Stiffness

Stiffness	Notation	Method	Value	Units
Shear	Ux	Strain-Disp	0.992	kN/mm
Shear	Uy	Strain-Disp	0.992	kN/mm
Axial	Uz	Shanton	560.000	kN/mm
Rotational	Rx	Shanton	1782000.000	kN-mm/rad
Rotational	Ry	Shanton	1782000.000	kN-mm/rad
Rotational	Rz	0.01 Rx,y	17820.000	kN-mm/rad

4.3.2 Damage Scenarios

Damage Case 1

Model equivalent of elastomeric bearing pad stiffness in two dimensions is shown in Figure 4.7 k_{XX} and k_{ZZ} are joint spring stiffnesses in the horizontal and vertical direction, respectively. $k_{\theta Y}$ represents the in-plane torsional stiffness. The unknown parameters in damage case 1 are k_{XX} and $k_{\theta Y}$ at all four support locations in the model. A 20% reduction in these stiffnesses is estimated using static error functions using simulated NDT.

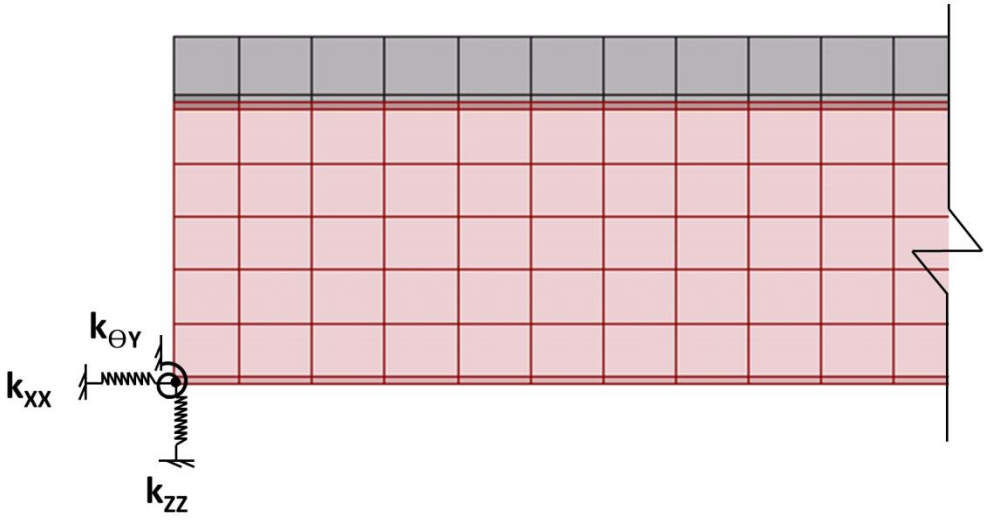


Figure 4.7 Equivalent Spring Stiffnesses for Bearing Pad

Damage Case 2

Highlighted section of the girder in Figure 4.8 shows the 624 shell elements that constitute damaged south span of the girder. A reduction of 20% in the material modulus of elasticity was successfully estimated using static strain error function.

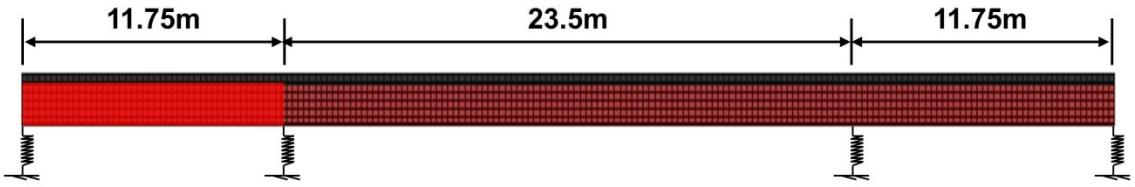


Figure 4.8 South Span Girder Damage

4.3.3 Strain Measurements

Under the actual instrumentation plan, the strain gauges are located at six stations across the length of the bridge. 4 strain gauges are installed at a particular station at each girder. Two strain gauges on the underside of the top flange and two at top of the bottom flange at either side of the

girder web. However, for simplicity, strains are measured at the joints at center line of bottom flange of girder. Along the length, the strains are measured at center of north and south girder spans as well as at quarter points of the longer center span.

4.4 PMB Frame-Shell Model

Figure 4.9 shows the full-scale finite element model of the Powder Mill Bridge. The model is comprised of frame and shell elements. Frame elements constitute the six longitudinal girders and lateral diaphragms in the FE model. The frame elements are shown in plan in Figure 4.10 . The deck is composed of planar quadrilateral shell elements. Figure 4.11 shows these shell elements in plan. The model shown below is the 3D extruded view of the FE model from SAP2000.

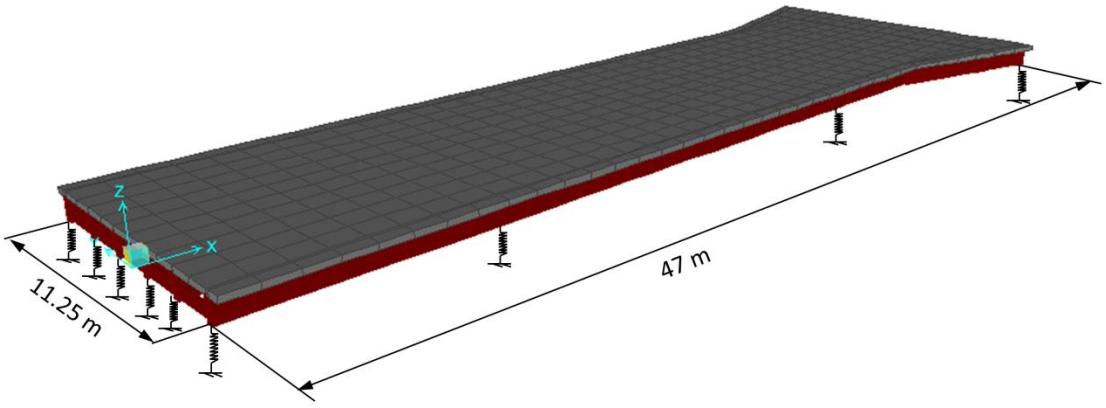


Figure 4.9 Powder Mill Bridge Frame-Shell Model

4.4.1 Damage Scenarios

Damage Case 1

Girders 4 and 5 which are assumed to have a 20% reduced bending capacity are shown in Figure 4.10 . Each girder consists of 48 frame elements. Frame elements are grouped together as north, south, and the center spans. Strain measurements are again taken at the bottom of the girder at the same locations described in Section 2.3.2.

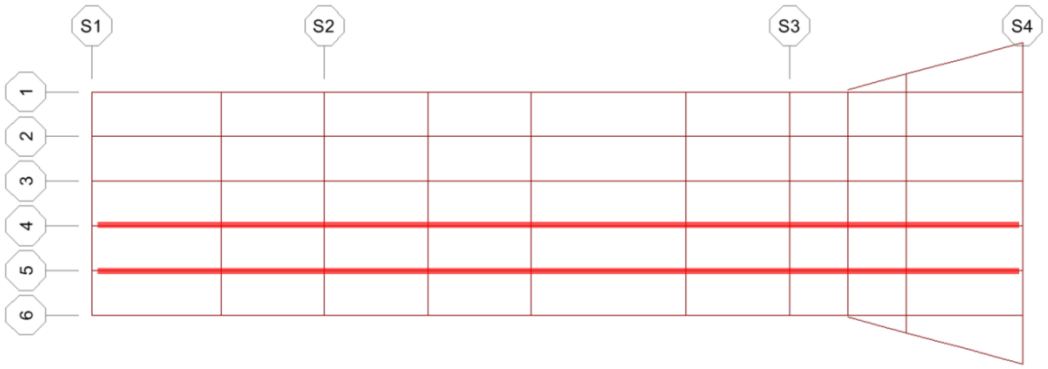


Figure 4.10 Girder 4 and Girder 5 Damage

Damage Case 2

The cluster of shell elements highlighted in red in Figure 4.11 is representative of 50% damage in the deck of the bridge.

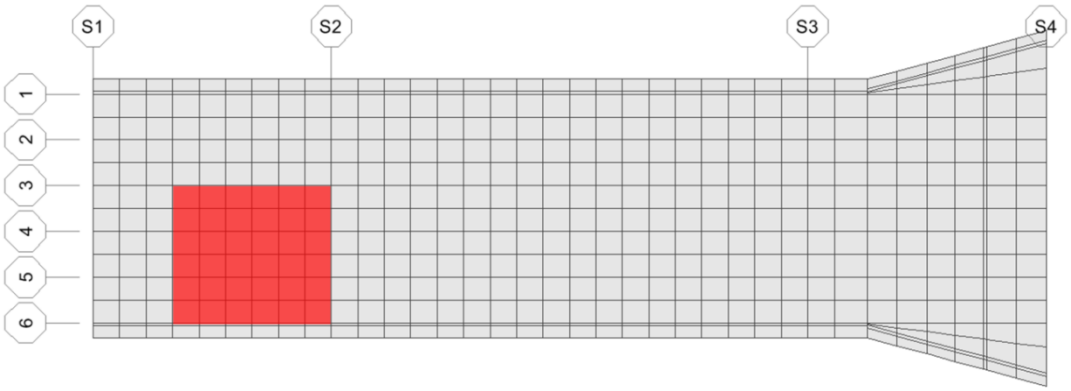


Figure 4.11 Deck Damage

4.4.2 Static Load Cases

Figure 4.12 shows truck locations across bridge width for paths X0 through X4 considered for this example. Truck loads are applied at 8 locations along the length of the bridge in each of the 5 paths to simulate the truck crossing the bridge. Spacing between the front and rearmost axles of the truck is considered to be 6.5m with the center to center spacing between rear axles to be 1.5m. Front axle load is 84.8kN and rear axles weigh 134.8kN each.

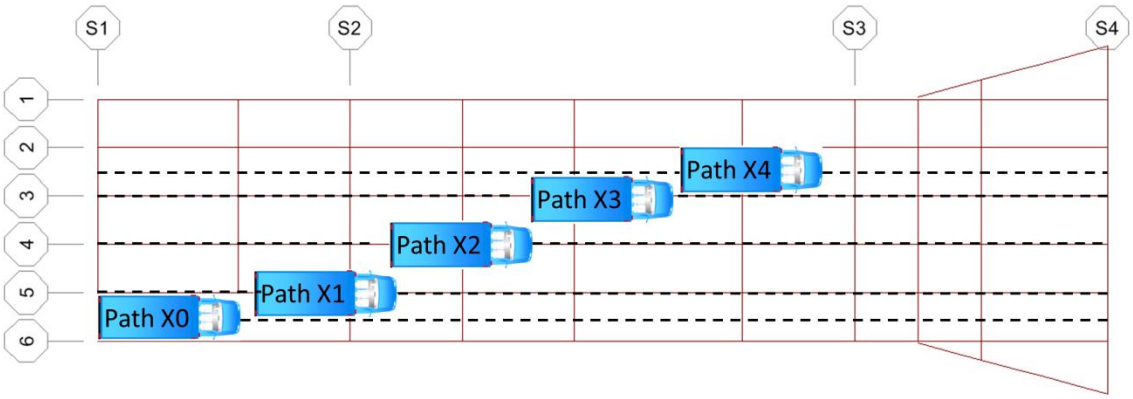


Figure 4.12 Test Truck Load Paths

Chapter 5

PARIS13.0 User Reference Manual

5.1 Introduction

PARIS13.0 program reads the SAP2000 model data into MATLAB using SAP2000's API and then employs SAP2000 program again during the FEA stage of model updating process. It is required that the SAP2000 model is created under specific guidelines to be used with PARIS13.0 program. PARIS13.0 also requires a MATLAB based user input file (PARIS13InputFile.m), which essentially supplies information regarding unknown parameters and parameter estimation options. This chapter is also the reference manual for PARIS13.0 program.

5.2 SAP2000 Model

5.2.1 SAP2000 Model File

The SAP2000 model file (.SDB) should be stored at this location on the computer's hard disk:

`C:\API\`

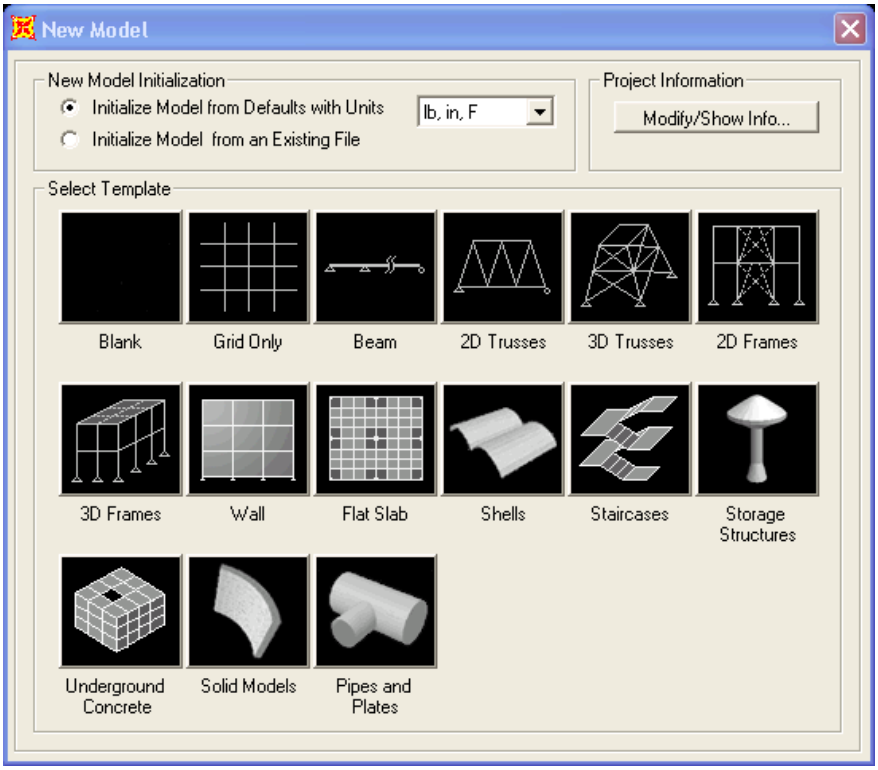
Before running PARIS13.0, it is required that the user creates the API folder in the C:\ drive and save the SAP2000 model at the location mentioned above.

5.2.2 Unit System

The units in SAP2000 can be classified as 'Present Units' and 'Database Units'. While the former can be changed at any time by the user, the latter can only be set before the user starts creating a FE model in SAP2000. The unit system chosen at the beginning cannot be altered by the user and the entire analysis in SAP2000 is carried out in 'Database Units'. However, for result output, any system of units can be chosen as 'Present Units'.

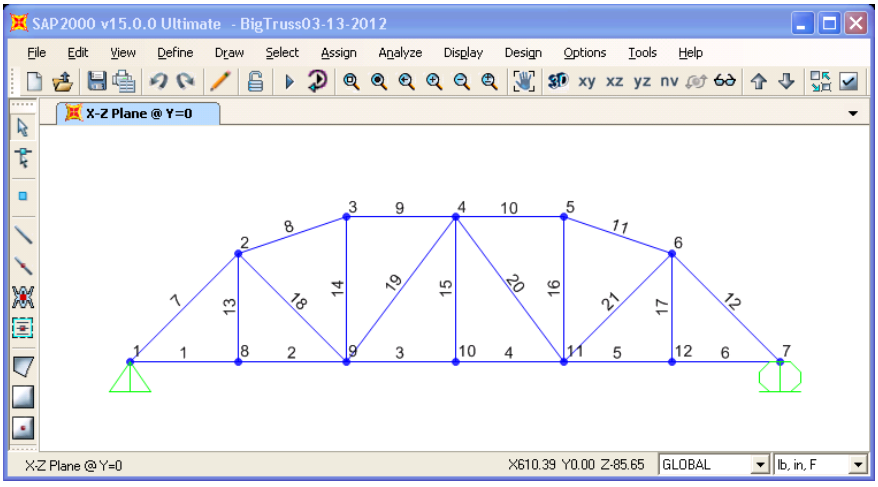
SAP2000 Procedure (Database Units):

i. *File -> New Model*
ii. *Select the system of units from the drop down menu in the dialogue box as shown below.*



SAP2000 Procedure (Present Units):

i. In the right bottom corner of main window in SAP2000, select any system of units from the drop down list to set as present units.



5.2.3 Local and Global Coordinate System

PARIS13.0 performs all internal calculations in the global coordinate system. Therefore, it is important that the SAP2000 model should comply with the following rules:

1. Local axes of all nodes in the model should be oriented similar to global axes.
2. Care should be taken that stiffness values are entered for link elements in local direction that correspond to desired stiffness values in global axes.

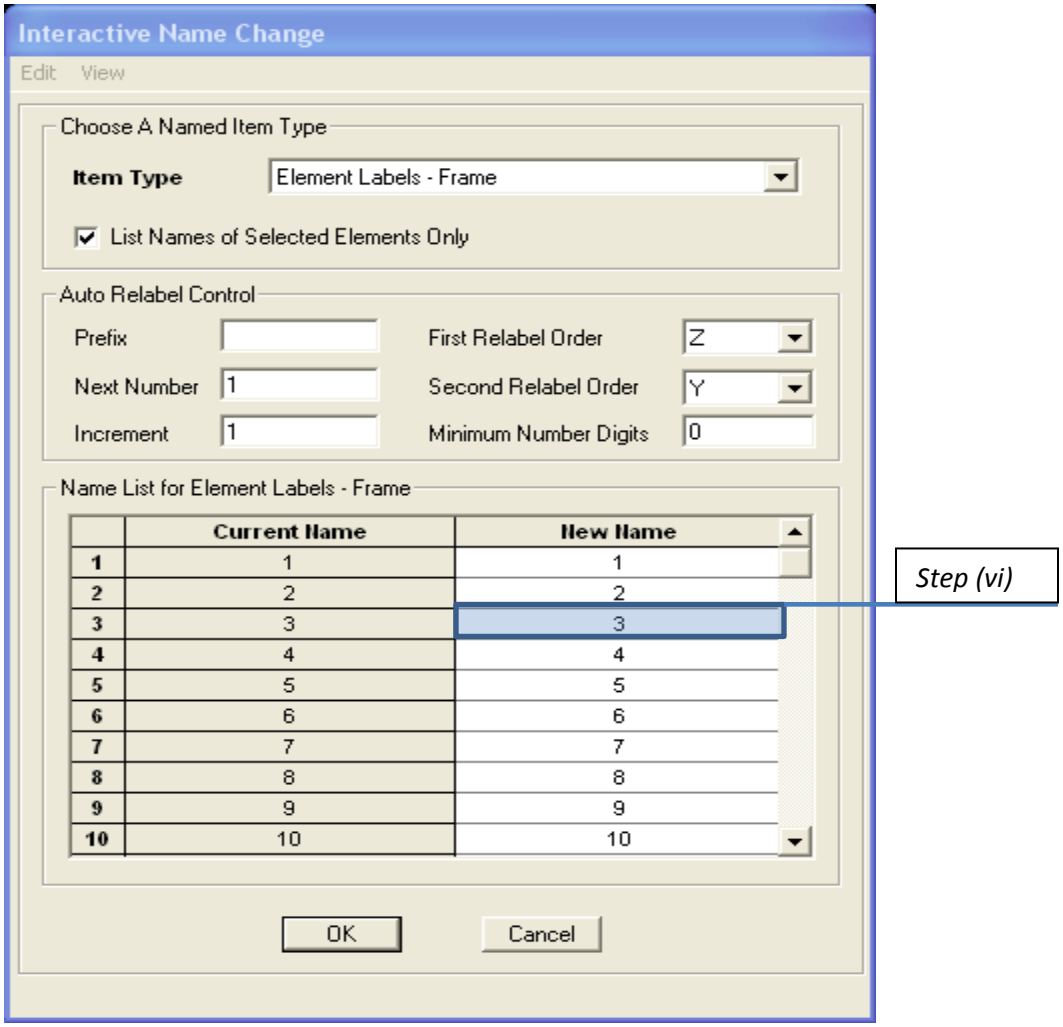
5.2.4 Nodes and Element Labels

1. Nodes / Joints: All nodes in the model should be numbered from 1 through n where n is the total number of nodes/joints elements in the FE model.
2. Frame Elements: All frame elements in the model should be numbered from 1 through n where n is the total number of frame elements in the FE model.
3. Shell Elements: All shell elements in the model should be numbered from 1 through n where n is the total number of shell elements in the FE model.
4. Solid Elements: All solid elements in the model should be numbered from 1 through n where n is the total number of solid elements in the FE model.
5. Link Elements: All link elements in the model should be numbered from 1 through n where n is the total number of link elements in the FE model.

SAP2000 Procedure:

i. Select all objects/elements of a particular type

ii. *Edit -> Change Labels*



iii. *In the 'Interactive Name Change' form: Edit -> Copy All Data in Grid*

iv. *Paste the data on clipboard on to MS Excel and reorder the second columns in ascending order without any gaps. The numbering should be 1 through n for a particular element/object type.*

v. *Copy the data in the second column.*

vi. *In the 'Interactive Name Change' form: Single click on the first cell of the second column 'New Name'.*

Edit -> Paste

vii. *Repeat this procedure for each type of object/element in the FE* model.

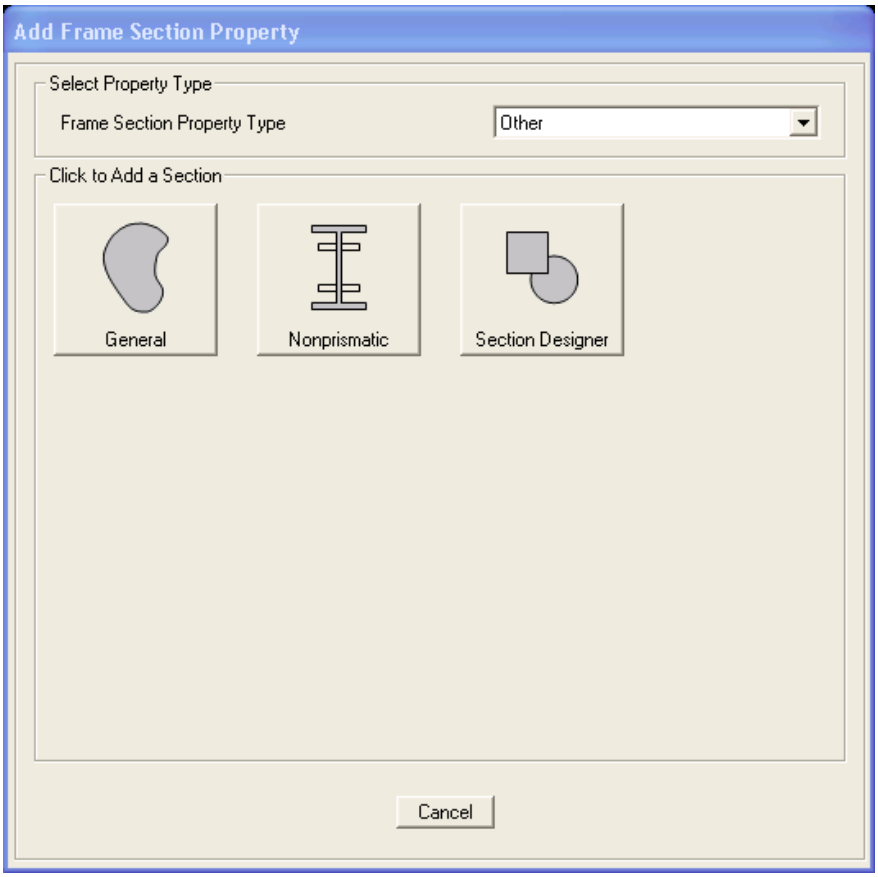
5.2.5 Frame Element Section Properties

All frame section properties must be defined under 'General' category. If a section from SAP2000 library is to be used, the user should first define a general section in SAP2000 with the same section properties as of the desired predefined section.

SAP2000 Procedure:

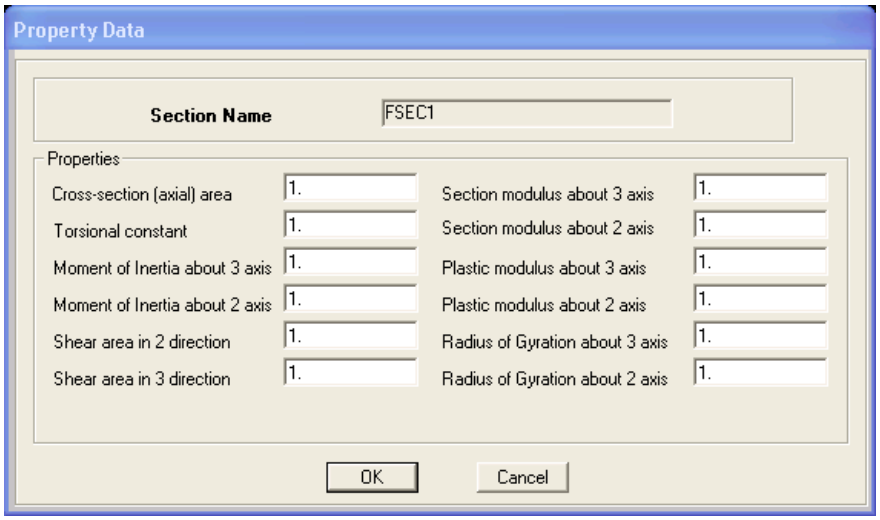
i. *Define -> Section Properties -> Frame Sections*

ii. *Add New Property*



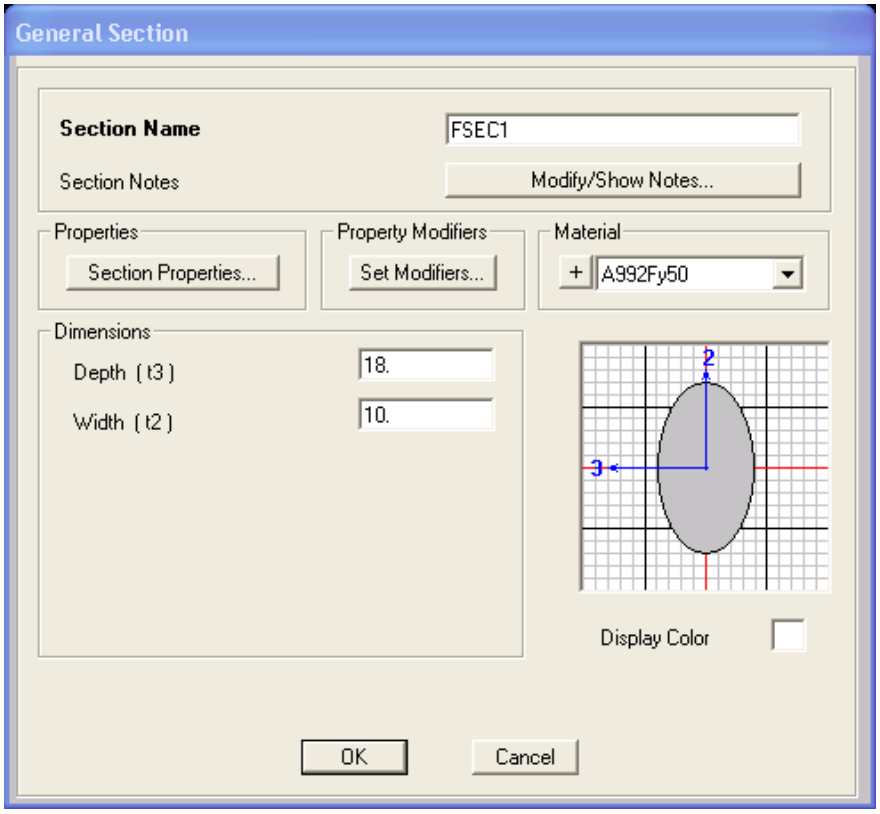
iii. *In the Add Frame Section Property dialogue box, select 'Other' from the drop down menu.*

iv. *Select 'General' and enter section properties manually.*



v. *Click 'Ok'*

vi. *Enter the 'Section Name' in the dialogue box that appears next.*



5.2.6 Analysis option: Active DOF

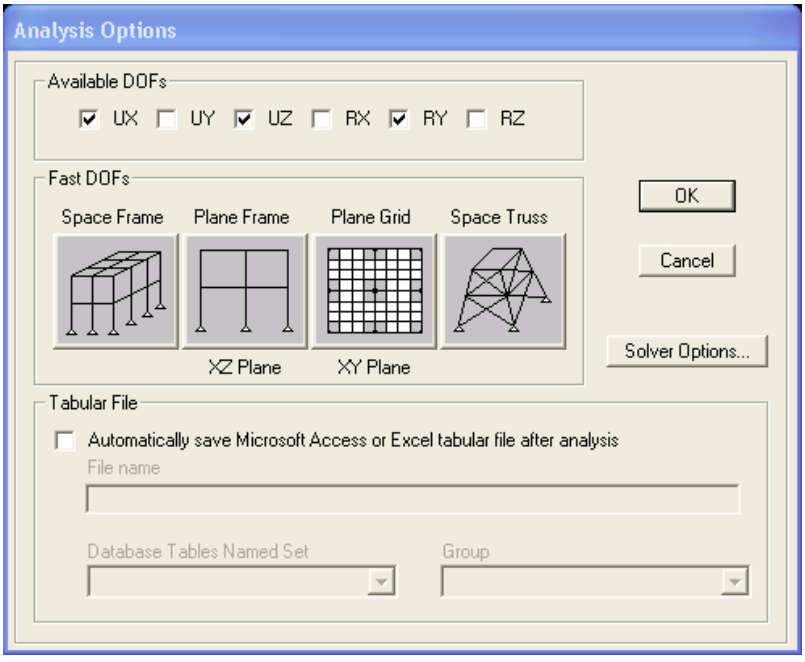
2D FE model can be created in any of the three planes i.e. XY, YZ, or XZ. Gravity loads in SAP2000 can be applied along any axes, thus not restricting the model to be in XZ or YZ plane should gravity loads be considered.

Any combination of DOFs can be chosen for analysis. The inactive DOFs are not considered in PARIS13.0. This option proves to be valuable in use with dynamic error functions where all mode shapes are not always useful like out of plane modes for 2D FE model.

SAP2000 Procedure:

i. *Analysis -> Set Analysis Option*

ii. *Select the DOFs to be made active*



5.2.7 Load Cases

PARIS13.0 provides the user with an option of defining and applying 999 load cases for each of the five available error functions. However, the load cases are required to be defined in a particular format as described in the table below.

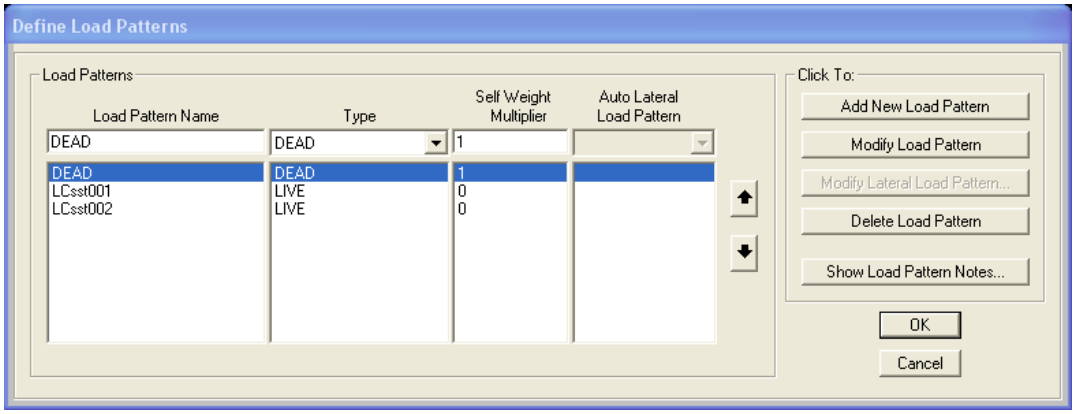
1. At least one load case for Static Stiffness (SS) error function: **LCsst0001** must be defined before running PARIS13.0.
2. Load Pattern and Load Cases should have the same label for corresponding load cases i.e. if a load case LCsst001 is to be defined, the corresponding load pattern should be LCsst001 as well. Nomenclature scheme for Load Cases is shown in Table 5.1.

Table 5.1 Load Cases Nomenclature

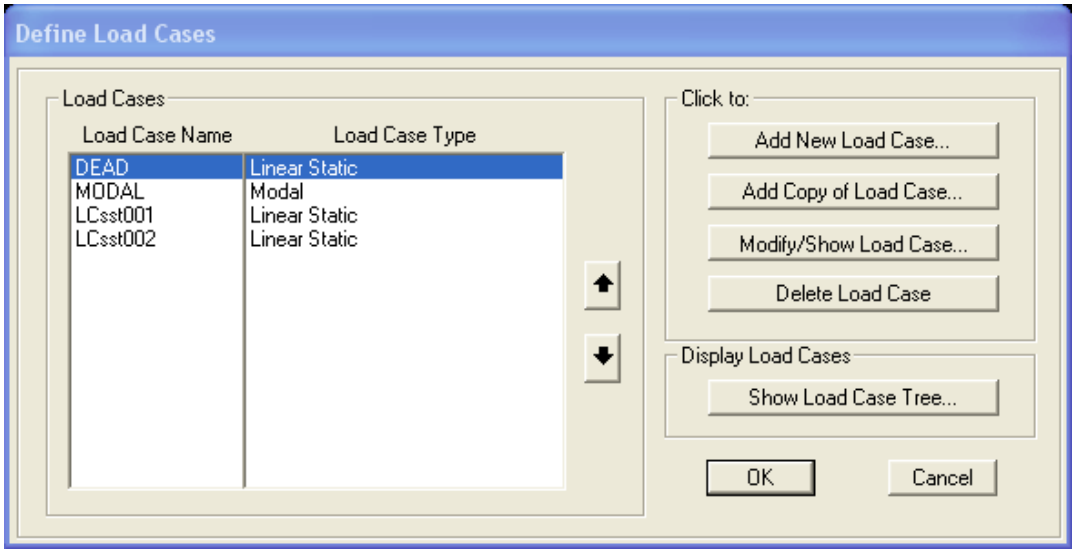
Error Function	Load Case (first)	Load Case (last)
Static Stiffness (SS)	LCsst**001**	LCsst**999**
Static Flexibility (SF)	LCsf1**001**	LCsf1**999**
Static Strain (SSTR)	LCstr**001**	LCstr**999**
Modal Stiffness (MS) Modal Flexibility (MF)	MODAL	

SAP2000 Procedure:

i. Define -> Load Patterns



ii. Define -> Load Cases



5.2.8 Element Grouping

PARIS13.0 is not designed to update models with elements grouped in SAP2000. The user is required to ascertain that element grouping is not done in the SAP2000 model. Instead, parameter grouping in PARIS13.0 is achieved by defining the following in the input file, which is explained in detail under 'Standardized Input File':

1. **`pgflag:`** Flag for existence of parameter grouping
2. **`pg:`** Parameter grouping matrix

5.3 PARIS13.0 Input File Details

Different components of the MATLAB based the input file are described in this section in the order in which they appear in the input data file (PARIS13InputFile.m file). The input file itself contains detailed instructions for entering data.

5.3.1 SAP2000 Model related data

1. SAP2000 File Name

   This is the name of the SAP2000 finite element model file located in the C:\ of computer as described in section 5.2.1

   `filename = 'C:\API\InputFile.SDB';`

2. Finite Elements in SAP2000 Model

   `framemodel`

   Defines whether SAP2000 model consists of frame elements or not.

   0: Model consists of frame elements

   1: Model does not consist of frame elements

   `shellmodel`

   Defines whether SAP2000 model consists of shell elements or not.

   0: Model consists of shell elements

   1: Model does not consist of shell elements

   `solidmodel`

   Defines whether SAP2000 model consists of solid elements or not.

   0: Model consists of solid elements

1: Model does not consist of solid elements

springmodel

Defines whether SAP2000 model consists of nodal spring assignments or not.

0: Model has nodal spring assignments

1: Model does not have spring nodal assignments

5.3.2 Parameter Estimation Related Data

1. Parameter Grouping

   Supplying parameter grouping data to PARIS 13.0 requires the input for two variables:

   pgflag

   Defines whether parameter grouping exists or not.

   0: No grouping

   1: Parameter grouping exists

   pg

   This is the grouping matrix. This matrix should be a rectangular matrix. If the number of elements grouped is different in each row, pad the rows with lesser number of elements with zeros. The values are entered in the format described here.

   Column 1: Group number

   Column 2: Parameter type

There are a total of 20 parameter types that can be updated using PARIS 13.0.

Column 3 onwards: Grouped element numbers

2. Error Function(s)

errorfunction

Error function(s) selected for a particular run of PARIS 13.0 is (are) chosen using this row vector.

0: Do not use corresponding error function

1: Use corresponding error function

3. Unknown Parameters

PARIS13.0 has the capability to update twenty types of parameters in a finite element model. Their description is provided in the Input File. These parameters are further classified under different categories based on to which type of finite elements they belong.

0: Do not consider unknown parameter

1: Consider unknown parameter

upframe

This is the matrix for unknown parameters for frame elements in the SAP2000 model.

Column 1: Frame element number (Entries must be in ascending order in column 1)

Column 2: Area (Area) parameter of frame element

Column 3: Moment of Inertia (Iz) parameter of frame element

Column 4: Moment of Inertia (Iy) parameter of frame element

Column 5: Torsional Constant (J) parameter of frame element

Column 6: Area Mass (AreaMass) mass parameter of frame element

upshell

This is the matrix for unknown parameters for shell elements in the SAP2000 model.

Column 1: Shell element number (Entries must be in ascending order in column 1)

Column 2: Modulus of Elasticity (E) parameter of frame element

Column 3: Shell Mass (ShellMass) parameter of frame element

upolid

This is the row vector containing solid element numbers for which Modulus of Elasticity (E) is an unknown parameter in the SAP2000model.

uplink

This is the matrix for unknown parameters for 2 node link elements in the SAP2000 model.

Column 1: Link element number (Entries must be in ascending order in column 1)

Column 2: Translational Stiffness in local ‘x’ direction (kxx)

Column 3: Translational Stiffness in local ‘y’ direction (kyy)

Column 4: Translational Stiffness in local ‘z’ direction (kzz)

Column 5: Rotational Stiffness about local ‘x’ direction (krx)

Column 6: Rotational Stiffness about local ‘y’ direction (kry)

Column 7: Rotational Stiffness about local ‘z’ direction (krz)

upspring

This is the matrix for unknown parameters for nodal spring assignments in the SAP2000 model.

Column 1: Node number (Entries must be in ascending order in column 1)

Column 2: Translational Stiffness in local ‘x’ direction (ksxx)

Column 3: Translational Stiffness in local ‘y’ direction (ksyy)

Column 4: Translational Stiffness in local ‘z’ direction (kszz)

Column 5: Rotational Stiffness about local ‘x’ direction (ksrx)

Column 6: Rotational Stiffness about local ‘y’ direction (ksry)

Column 7: Rotational Stiffness about local ‘z’ direction (ksrz)

4. Data to Simulate Measurements

PARIS13.0 uses this data to simulate a model to represent the true parameters. The user can input the factor for each parameter type with respect to its initial guess which is representative of the true value of the parameter. The initial guess of parameter value is always 1.

factorforframe

This is a row vector for five parameter types available for frame element.

Column 1: Factor for frame cross-sectional area representing true value

Column 2: Factor for frame Iz representing true value

Column 3: Factor for frame Iy representing true value

Column 4: Factor for frame J representing true value

Column 5: Factor for frame mass representing true value

factorforshell

This is a row vector for two parameter types available for shell element.

Column 1: Factor for material modulus of elasticity (E) area representing true value

Column 2: Factor for shell mass representing true value

factorforshell

This is a single value for factor for material modulus of elasticity (E) for solid element.

factorforlink

This is a row vector for six parameter types available for link element.

Column 1: Factor for link translational stiffness along global X (kxx)

Column 2: Factor for link translational stiffness along global Y (kyy)

Column 3: Factor for link translational stiffness along global Z (kzz)

Column 4: Factor for link rotational stiffness about global X (krx)

Column 5: Factor for link rotational stiffness about global Y (kry)

Column 6: Factor for link rotational stiffness about global Z (krz)

factorforspring

This is a row vector for six parameter types available for spring assignments at supports.

Column 1: Factor for spring translational stiffness along global X (ksxx)

Column 2: Factor for spring translational stiffness along global Y (ksyy)

Column 3: Factor for spring translational stiffness along global Z (kszz)

Column 4: Factor for spring rotational stiffness about global X (ksrx)

Column 5: Factor for spring rotational stiffness about global Y (ksry)

Column 6: Factor for spring rotational stiffness about global Z (ksrz)

5. Observation Points for Measurements

The observations or measurements can be classified into two categories, displacement and strains. The stiffness and flexibility based error functions requires input of measured nodes location as a matrix of nodes where displacement is measured either under the associated load case or for modal analysis.

mnodesss

This is the matrix of measured nodes for SS error function. In columns 3-8, enter:

0: Do not measure translation/rotation along/about this DOF

1: Measure translation/rotation along/about this DOF

Column 1: Load Case corresponding to LCsst000 to LCsst999 load case. (Entries should be in ascending order)

Column 2: Node Number (Entries should be in ascending order for each load case defined in column 1)

Column 3: Measure translation along local ‘x’ direction

Column 4: Measure translation along local ‘y’ direction

Column 5: Measure translation along local ‘z’ direction

Column 6: Measure translation about local ‘x’ direction

Column 7: Measure translation about local ‘y’ direction

Column 8: Measure translation about local ‘z’ direction

mnodessf

This is the matrix of measured nodes for SF error function. Data entry is similar to [mnodesss] described above except that the load cases in column 1 correspond to load case LCsf1000 through LCsf1999 in the SAP2000 model.

Modal error functions, in addition to measurement locations for each mode, also require another input matrix for providing data as to which modes are measured.

fms

This is the matrix of set of modes measured for MS error function

Column 1: Set number / Serial number

Column 2: Measured mode number

mnodesms

This is the matrix of measured nodes for MS error function. Data entry is similar to [mnodesss] described above except that the load cases in column 1 correspond to MODAL case in the SAP2000 model described in column 1 of [fms] matrix.

fmf

This is the matrix of set of modes measured for MF error function

Column 1: Set number / Serial number

Column 2: Measured mode number

mnodesmf

This is the matrix of measured nodes for MS error function. Data entry is similar to [mnodesss] described above except that the load cases in column 1 correspond to MODAL case in the SAP2000 model described in column 1 of [fmf] matrix

6. Error Function Normalization

nepflag

‘0’: Do not normalize

‘1’: Normalize error function with respect to its covariance matrix

‘2’: Normalize error function with respect to its initial value

7. Measurement Errors

fmeflag

‘0’: No measurement error in force measurement

‘1’: Error in force measurement

Absolute error in force measurement

faess: Measurement error level for stiffness and flexibility error functions

faesstr: Measurement error level for static strain error functions

Proportional error in force measurement

fapss: Measurement error level for stiffness and flexibility error functions

fapsstr: Measurement error level for static strain error functions

umeflag

'0': No measurement error in displacement or strain measurement

'1': Error in displacement/strain measurement

Absolute error in displacement/strain measurement

uaess: Measurement error level for stiffness and flexibility error functions

uaesstr: Measurement error level for static strain error functions

Proportional error in displacement/strain measurement

uapss: Measurement error level for stiffness and flexibility error functions

uapsstr: Measurement error level for static strain error functions

8. Solution Techniques

SolveMethod

The variable requires a string to choose between two optimization options available in PARIS13.0. One is based on analytical sensitivities while the other uses constrained function minimization technique available in MATLAB optimization toolbox.

'SensitivitySol': Choose analytical sensitivity solution

'MatlabSol': Constrained function minimization in MATLAB toolbox

9. Under-relaxation

This is a flag to activate under relaxation.

UnderRelaxFlag

'0': Do not under relax change in parameter

'1': Under relax change in parameters

10. Convergence Limits

rdjlim

Convergence limit relative change in the value of quadratic objective function J(p)

rdpnlim

Convergence limit relative change in the normalized value of parameter(s)

11. Output Parameter Estimation Results

OutResults

Write the results of parameter estimation to a text file with the name same as the SAP2000 model filename (.txt file)

‘1’: Create the Output File

‘0’: Do not write results to the Output File

5.4 Sample PARIS13.0 Input File

```
% ************************************************************************
%                           PARIS13.0 INPUT FILE
% ************************************************************************
tic; clc;
clear all; close all

% -------------------Global Variable Declaration---------------------------
% SAP2000 Model file path
global filename unitflag
% SAP2000 Model related variables
global framemodel shellmodel solidmodel linkmodel springmodel
% Parameter estimation data
global pgflag pg errorfunction lsepflag upframe upshell upsolid uplink ...
       upspring factorforframe factorforshell factorforsolid …
       factorforlink factorforspring SolveMethod
% Matrices of measured nodes for static error function
global mnodesss   mnodessf1   mnodessf2   melsstr1  melsstr2
% Matrices of measured frequencies for dynamic error function
global fms fmf1 fmf2
% Matrices of measured nodes for dynamic error function
global mnodesms mnodesmf1 mnodesmf2
% Parameter estimation options
global mni nepflag solflag rdpnlim rdjlim inflag outflag
global OutResults UnderRelaxFlag
global fmeflag umeflag ertflag erdflag
global fpetss fpetsstr faetss faetsstr
global upetss upetsstr uaetss uaetsstr


% ************************************************************************
%                             INPUT DATA FILE
% ************************************************************************

% --------------------SAP2000 MODEL RELATED DATA-------------------------

% Enter the SAP2000 file name. Complete file path should be specified:
% filename = C:\API\3DFrameProblem.SDB'
% NOTE: The file location should be: C:\API\
%       Please create a folder in C:\ on your computer and rename it to API
%       if it does not already exist

filename='C:\API\IASC-ASCE-BenchmarkStructure-PR.SDB';

% Type of Finite Elements Used
framemodel = 1;         %   0  -   Not include frame element
                        %   1  -   Include frame element
shellmodel = 1;         %   0  -   Not include shell element
                        %   1  -   Include shell element
solidmodel = 0;         %   0  -   Not include solid element
                        %   1  -   Include solid element
linkmodel  = 0;         %   0  -   Not include link element
                        %   1  -   Include link element
```

```
springmodel= 0;         %   0  -   Not include node spring
                        %   1  -   Include node spring


% --------------ENTER DATA FOR PARAMETER ESTIMATION-----------------------

% ----------------------Parameter Grouping data---------------------------
pgflag = 1;             %   1  -   Group parameters
                        %   0  -   Do not group parameters
% NOTE: pgflag should be 1 with 'SolveMethod' = 'MatlabSol' below

% Enter grouping matrix
% pg   : Grouping Matrix
% NOTE : A parameter group should contain more than one parameter
%   - [pg] should be a rectangular matrix, empty spaces should be filled
%     with zeros
%   - Column 1 contains the group number
%   - Column 2 represents the parameter type as per the following
%   classification
%     Type   Parameter
%     1 ---> Frame Area
%     2 ---> Frame Iz
%     3 ---> Frame Iy
%     4 ---> Frame J
%     5 ---> Frame AreaMass
%     6 ---> Shell 'E'
%     7 ---> Solid 'E'
%     8 ---> Link 'kxx'
%     9 ---> Link 'kyy'
%    10 ---> Link 'kzz'
%    11 ---> Link 'krx'
%    12 ---> Link 'kry'
%    13 ---> Link 'krz'
%    14 ---> Shell  'mass'
%    15 ---> Spring 'ksxx'
%    16 ---> Spring 'ksyy'
%    17 ---> Spring 'kszz'
%    18 ---> Spring 'ksrx'
%    19 ---> Spring 'ksry'
%    20 ---> Spring 'ksrz'
%   - The elements of each row from column 3 onwards represent contain
%     parameters in that group
%   - Only unknown parameters may be grouped
%   - An unknown parameter can belong to only one group

%     Group #     Parameters Type     Grouped Element Numbers
pg = [  1         1                   1 7
        2         1                   2 8
        3         1                   3 4
        4         1                   5 6];

% ----------------------------Error Function--------------------------------
% Abbreviations for the error functions that are currently available are:
%    SS   : Static Displacement Data - Stiffness Error Function
%    SF1  : Static Displacement Data - Flexibility Error Function (Tufts)
%    SF2  : Static Displacement Data - Flexibility Error Function (Hjemstad)
%    SSTR : Static Strain Data and Error Function
```

```
%    MS   : Modal Displacement Data - Stiffness Error Function
%    MF1  : Modal Displacement Data - Flexibility Error Function (Tufts)
%    MF2  : Modal Displacement Data - Flexibility Error Function (Hjemstad)
           MF2 is not currently available

% Enter matrix for the type of parameter estimation to be performed.
%  Several error functions can be used simultaneously. If the column
corresponding
% to an individual error is '1' then that error function will be used in the
% parameter estimation process, if it is '0', that error function will not be
used.
%                 1     2     3     4     5     6    7
%                 SS    SF1   SF2   SSTR  MS    MF1  MF2
errorfunction = [ 1     0     0     0     0     0    0  ];
% 0 = use the load cases simultaneously, no stacking for each error function
% 1 = use the load cases individually and stack for each error function used
% ----------------------Error Function Stacking--------------------------
% LSEPFLAG is used to turn stacking on. Stacking is 'on' as a default option,
% no user input required. Similar to ERRORFUNCTION, each column represents
% a particular load case.
lsepflag     = errorfunction;
% --------------------------Unknown Parameters---------------------------
% Measured unknown parameters for FRAME ELEMENTS
% NOTE: Enter the unknown parameters in the matrix form
%       Column 1 : Frame element number
%       Column 2 : 1 if 'Area' is unknown parameter, else 0
%       Column 3 : 1 if 'Iz  ' is unknown parameter, else 0
%       Column 4 : 1 if 'Iy  ' is unknown parameter, else 0
%       Column 5 : 1 if 'J   ' is unknown parameter, else 0
%       Column 6 : 1 if 'AreaMass' is unknown parameter, else 0

%         Frame Element #   Area  Iz   Iy   J    AreaMass
upframe = [1                  1    0    0   0      0
           2                  1    0    0   0      0
           3                  1    0    0   0      0
           4                  1    0    0   0      0
           5                  1    0    0   0      0
           6                  1    0    0   0      0
           7                  1    0    0   0      0
           8                  1    0    0   0      0];

% Measured unknown parameters for SHELL ELEMENTS
% NOTE: Enter the unknown shell parameters in the matrix form
%       Column 1 : Shell element number
%       Column 2 : 1 if 'E' is unknown parameter, else 0
%       Column 3 : 1 if 'mass' is unknown parameter, else 0

%         Shell Element #   E    ShellMass
upshell = [];

% Measured unknown parameters for SOLID ELEMENTS
% NOTE: Enter the solid element numbers for which 'E' is unknown in a row
%       vector form
upsolid = [];

% Measured unknown parameters for LINK ELEMENTS
% NOTE: Enter the link element numbers for which stiffness is unknown in
```

```
% along/about different degrees of freedom.
% NOTE: Enter the link parameters in the matrix form
%       Column 1 : Link element number
%       Column 2 : 1 if 'k' in 'X' direction is unknown parameter,else 0
%       Column 3 : 1 if 'k' in 'Y' direction is unknown parameter,else 0
%       Column 4 : 1 if 'k' in 'Z' direction is unknown parameter,else 0
%       Column 5 : 1 if 'k' about 'X' direction is unknown parameter,else 0
%       Column 6 : 1 if 'k' about 'Y' direction is unknown parameter,else 0
%       Column 7 : 1 if 'k' about 'Z' direction is unknown parameter,else 0

%         Link Element #   kXX  kYY  kZZ  krX  krY  krZ
uplink  = [];

% Measured unknown parameters for SPRING NODE ASSIGNMENTS
% NOTE: Enter the spring numbers for which stiffness is unknown in
% along/about different degrees of freedom.
% NOTE: Enter the spring parameters in the matrix form
%       Column 1 : Spring element number
%       Column 2 : 1 if 'k' in 'X' direction is unknown parameter,else 0
%       Column 3 : 1 if 'k' in 'Y' direction is unknown parameter,else 0
%       Column 4 : 1 if 'k' in 'Z' direction is unknown parameter,else 0
%       Column 5 : 1 if 'k' about 'X' direction is unknown parameter,else 0
%       Column 6 : 1 if 'k' about 'Y' direction is unknown parameter,else 0
%       Column 7 : 1 if 'k' about 'Z' direction is unknown parameter,else 0

%         Node #   kXX  kYY  kZZ  krX  krY  krZ
upspring  = [];

% ------------Data related to Measured Static Displacements----------------
unitflag = 6;  % The processed NDT data should be in either of the two
               % of units:
               % lb_in_F  = 1
               % lb_ft_F  = 2
               % kip_in_F = 3
               % kip_ft_F = 4
               % kN_mm_C   = 5
               % kN_m_C   = 6
               % N_m_C   = 10

% If inflag == 0 i.e. simulated data from 'true' section properties are used
% NOTE: This factor is only multiplied with unknown paramters to simulate
% their section properties.

% Enter a factor for each of the 5 properties of frame elements to
% simulate true properties for parameter estimation.
% Frame Elements
%                 Area     Iz       Iy      J       AreaMass
factorforframe  = [0.0    1.00     1.00     1.00     1.00];

% Shell Elements
%                   E       Mass
factorforshell  = [1.00    1.00];

% Solid Elements
factorforsolid  = 1.00;

% Link Elements
```

```
%                   kXX     kYY      kZZ      krX      krY     krZ
factorforlink   = [1.00    1.00     1.00     1.00     1.00    1.00];

% Nodes with Spring Assignments
%                   ksX     ksY      ksZ      ksXX     ksYY    ksZZ
factorforspring = [1.00    1.00     1.00     1.00     1.00    1.00];

% Matrices of measured nodes (mnodesss,mnodessf1,mnodessf2)
%        Column  # 1         load case number. if lsepflag=0 enter 0 for load
case #
%        Column  # 2        measured node number.
%        Columns # 3-8      measured dof code
%                           0 - not measured (or nonexisting in 2d cases)
%                           1 - measured
% NOTE : If in SAP2000 Finite Element Model, a node is associated
% exclusively with a solid element (i.e. no frame or shell element is
% connected to it) then for that node 'rx','ry','rz' should be equal to 0

% Matrix of nodes at which displacements are measured for SS errorfunction
%            Load Case  node #    ux      uy      uz      rx      ry      rz
mnodesss=[   1          32         1       0       1       0       0       0
             1          35         1       0       1       0       0       0
             1          42         1       0       1       0       0       0
             1          45         1       0       1       0       0       0
             2          32         1       1       0       0       0       0
             2          35         1       1       0       0       0       0];

% Matrix of nodes at which displacements are measured for SF1 errorfunction
%            Load Case  node #   ux      uy      uz      rx      ry      rz
mnodessf1 = mnodesss;

%-------------- Static Strain Measurement Related Data --------------------
% NOTE: SSTR (Static Strain) Error Function is only available for FRAME and
% SHELL elements in PARIS13.0.
%-------------------------------------------------------------------------
%
% Matrix of strain measurements for frame elements
%   Column 1     Load case number
%   Column 2     Measured element number
%   Columns 3-5  Define location of strain gauge on element as follows:
%           ---------------------------------------------------------------
%           Column 3     the distance from node i to the gauge in local x
%                        direction
%           Column 4     the distance from the NA to the gauge in the
%                        local y direction (required for 2D or 3D frames)
%           Column 5     the distance from the NA to the gauge in the
%                        local z direction (required for 3D frames only)
%           ---------------------------------------------------------------
% NOTE: Enter element numbers in order
%          Load Case  Element # (xbar)       (ybar)     (zbar)
melsstr1 = [1          3          50          0          0
            1          4          50          0          0
            1          5          50          0          0
            1          8          50          0          0
            2          4          50          0          0
            2          5          50          0          0
            2          8          50          0          0];
```

```
% Matrix of strain measurements of shell elements
%          Load Case  Node #     S11          S22        S12  Top(1)/Bot(0)
melsstr2= [1          5           1            1          1        1
           1          6           1            1          1        1
           1          9           1            1          1        1
           2          5           1            1          1        1];

%------- Selected Modes of Vibration for Modal Error Functions ------------
%      Column #1 - set number (similiar to load case number in static cases)
%      Column #2 - mode of vibration number

% Matrix of set of mode shapes for MS errorfunction
%          Set#   Mode#
fms =   [ 1       3
          2       4];
% Matrix of set of mode shapes for MF1 errorfunction
%          Set#   Mode#
fmf1 =  fms;

% Matrix of measured nodes for MS errorfunction
%         Load Case  node #   ux      uy       uz      rx      ry      rz
mnodesms =[  1        2          1       1       1       1       1      1
             1        5          1       1       1       1       1      1
             1        8          1       1       1       1       1      1
             2        3          1       1       1       1       1      1
             2        6          1       1       1       1       1      1
             2        9          1       1       1       1       1      1];

% Matrix of measured nodes for MF1 errorfunction
%            Load Case  node #   ux      uy      uz      rx      ry      rz
mnodesmf1 =  mnodesms;

% Matrix of measured nodes for MF2 errorfunction
%            Load Case  node #   ux      uy      uz      rx      ry      rz
mnodesmf2 =  mnodesms;

% --------------------NORMALIZATION METHODS--------------------------------
nepflag = 0;       % 0: do not normalize ep,
                   % 1: normalize ep w.r.t the covariance matrix of ep
                   %    in presence of measurement errors:
                   %     -Requires fmeflag, umeflag =1 or manual
                   %      input of standard deviations (see above)
                   %     -Only static error functions supported
                   % 2: normalize ep w.r.t initial value (use only with
                   % SSTR error function when not using nepflag = 1)

% ----------------------MEASUREMENT ERRORS---------------------------------
erdflag  = 2        ;     % error distribution
                          % 1: uniform
                          % 2: normal
ertflag  = 1        ;     % error type
                          % 1: proportional
                          % 2: absolute

% --------------------Measurement Error in Forces-------------------------
fmeflag = 0;     % force measurement error flag
```

```
% Proportional Error - Enter % force error at all dof (unitless)
fpetss     = 0.02;      % For static displacement data
fpetsstr   = 0.02;      % For static strain data
fpetms     = 0.02;      % For modal displacement data

% Absolute Error    - Enter absolute force error at all dof (units)
faetss     = 0.02;      % For static displacement data
faetsstr   = 0.02;      % For static strain data
faetms     = 0.02;      % For modal displacement data

% -----------------Measurement Error in Displacement-----------------------
umeflag = 0;    % displacement error measurement flag

% Proportional Error - Enter % Disp error at all measured nodes (unitless)
upetss    = 0.02;       % For static displacement data
upetsstr  = 0.02;       % For static strain data
upetms    = 0.02;      % For modal displacement data

% Absolute Error    - Enter absolute force error at all dof (units)
uaetss    = 0.02;       % For static displacement data
uaetsstr  = 0.02;       % For static strain    data
uaetms    = 0.02;       % For modal displacement data

% ----------------------SOLUTION TECHNIQUES-------------------------------
% Choose between MATLAB Optimization Method and Sensitivity Analsyses
SolveMethod = 'MatlabSol';  % Enter 'MatlabSol' for MATLAB Optimization
                                % Enter 'SensitivitySol' for Sensitivity
                                % Analysis

% If 'SensitivitySol' is chosen above, choose solution method
solflag = 1;      % Various hill climbing solutions available are:
                  % 1 = use direct inv. / least squares - Gauss-Newton
                  % 2 = use gradient method - Steepest Descent

mni = 50; % max. no. of iterations

% ----------------------UNDER-RELAXATION FACTOR----------------------------
UnderRelaxFlag = 1;    %     0  -   Do not use UnderRelaxation
                       %     1  -   Use UnderRelaxation


% --------------------------CONVERGENCE LIMITS----------------------------
% The following limits all deal with the relative change in parameters
% associated with the parameter estimation procedure.
rdjlim  = 1e-10;     % convergence limit for relative change in 'J(p)'
rdpnlim = 1e-06;     % convergence limit for relative change value of
                     % parameters 'p'

% --------------------Write Parameter Estimation Data to Excel File--------
OutResults = 1;      % Write parameter estimation history to excel file
                     % File Name is the name of SAP2000 file with a xlsx
                     % extension and is 'C:\API\' folder

% paris13; toc;
```

5.5 Table of Main Program Variables

The first column of Table 5.2 contains the main variables in PARIS13.0 program which are required to be defined by the user. Description of each variable is given in the second column.

Table 5.2 User Defined PARIS13.0 Variables

Variable Name	Description
filename	SAP2000 FE model file path and file name
unitflag	Unit system for input/output to SAP2000 model
framemodel	Flag if the FE model contains frame elements
shellmodel	Flag if the FE model contains shell elements
solidmodel	Flag if the FE model contains solid elements
linkmodel	Flag if the FE model contains 2-node link elements
springmodel	Flag if the FE model contains spring node assignments
pgflag	Flag for parameter grouping
pg	Parameter grouping matrix
errorfunction	Vector for choosing error function(s)
upframe	Matrix for unknown frame element parameters
upshell	Matrix for unknown shell element parameters
upsolid	Matrix for unknown solid element parameters
uplink	Matrix for unknown 2-node link element parameters
upspring	Matrix for unknown joint springs parameters
factorforframe	Fraction/Multiplier for frame parameters for simulating NDT data
factorforshell	Fraction/Multiplier for shell parameters for simulating NDT data
factorforsolid	Fraction/Multiplier for solid parameters for simulating NDT data
factorforlink	Fraction/Multiplier for link parameters for simulating NDT data
factorforspring	Fraction/Multiplier for spring parameters for simulating NDT data
SolveMethod	Optimization/Minimization method
solflag	Choice of solution technique for analytical sensitivity method
mnodesss	Matrix of measured nodes for SS error function
mnodessf1	Matrix of measured nodes for SF errorfunction
melsstr1	Matrix of measured frame element strains for SSTR error function
melsstr2	Matrix of measured shell element strains for SSTR error function
fms	Matrix of measured set of mode shapes for MS error function
fmf1	Matrix of measured set of mode shapes for MF1 error function
mni	Maximum number of iterations permitted
nepflag	Normalization method
rdpnlim	Convergence criterion for relative change in parameter value
rdjlim	Convergence criterion for relative change in objective function
fmeflag	Flag for introducing error in simulated force measurements
umeflag	Flag for introducing error in simulated displacement measurements
erdflag	Flag for selecting uniform or normal error distribution
ertflag	Flag for selecting absolute or proportional error type
fpetss	Percentage error in force measurements for SS and SF error functions
fpetsstr	Percentage error in force measurements for SSTR error functions
faetss	Absolute error in force measurements for SS and SF error functions
faetsstr	Absolute error in force measurements for SSTR error functions
upetss	Percentage error in displ. measurements for SS and SF error functions
upetsstr	Percentage error in strain measurements for SSTR error functions
uaetss	Absolute error in displ. measurements for SS and SF error functions
uaetsstr	Absolute error in strain measurements for SSTR error functions

In Table 5.3, the variables which internally get created are listed. The two columns contain variable name and description.

Table 5.3 PARIS13.0 Program Internal Variables

Variable Name	Description
ActiveDOF	Active translational and rotational degrees of freedom (DOFs)
nnod	Total number of nodes in the FE model
nresdof	Total number of restrained degrees of freedom (DOFs)
coord	Matrix of node/joint coordinates
coordsstr	Matrix of node/Joint coordinates matrix for sstr error function
framenode	Nodes to which frame elements connect
shellnode	Nodes to which shell elements connect
solidnode	Nodes to which solid elements connect
linknode	Nodes to which link element connect
nelemframe	Number of frame elements
nelemshell	Number of shell elements
nelemsolid	Number of solid elements
nelemlink	Number of link elements
nspringnode	Number of spring nodes
nlc	Total number of load cases
nlcss	Number of load cases for SS error function
nlcsf1	Number of load cases for SF error function
nlcsstr	Number of load cases for SSTR error function
nlcms	Number of modes measured for MS error function
nlcmf1	Number of modes measured for MF error function
FrameSecProp	Matrix/structure to store frame elements section properties
FrameSecMat	Matrix/structure to store frame elements material properties
FrameModifierInitital	Matrix/structure to store frame elements section properties modifiers
ShellSecProp	Matrix/structure to store shell elements section properties
ShellSecMat	Matrix/structure to store shell elements material properties
ShellModifierInitial	Matrix/structure to store shell elements section properties modifiers
SolidSecProp	Matrix/structure to store solid elements section properties
SolidSecMat	Matrix/structure to store solid elements material properties
LinkSecPropAll	Matrix/structure to store link elements section properties
LinkSecProp	Matrix/structure to store link elements stiffness values
SpringNodeProp	Matrix/structure to store node/joint spring stiffness values
mnfss	Matrix of applied node forces for SS error function
mnfsf1	Matrix of applied node forces for SF error function
mnfsstr	Matrix of applied node forces for SSTR error function
bc	Matrix to store boundary conditions at restrained nodes
connframelement	Matrix to store connectivity of all frame elements
lelemframe	Matrix to store length of all frame elements
p	Column vector of unknown parameters
piNorm	Column vector of normalized initial guess of parameters
pfNorm	Column vector of normalized true parameter values
nup	Total number of unknown parameters
nupframe	Number of unknown parameters for frame elements
nupshell	Number of unknown parameters for shell elements
nupsolid	Number of unknown parameters for solid elements
nuplink	Number of unknown parameters for link elements
nupspring	Number of unknown parameters for joint spring assignments
ndof	Total number of degrees of freedom in the model
frameshelldof	Degrees of freedom common to shell and frame elements

soliddof	Degrees of freedom exclusive to solid elements
vmdofss	Vector of DOFs at which translations/rotations are measured for SS error function
vmdofsf1	Vector of DOFs at which translations/rotations are measured for SF1 error function
vmdofsf2	Vector of DOFs at which translations/rotations are measured for SF2 error function
vmdofsstr1	Vector of measured strain locations for frame elements for SSTR error function
vmdofsstr2	Vector of measured strain locations for shell elements for SSTR error function
vmdofms	Vector of DOFs at which translations/rotations are measured for MS error function
vmdofmf1	Vector of DOFs at which translations/rotations are measured for MF1 error function
vmdofmf2	Vector of DOFs at which translations/rotations are measured for MF2 error function
vudofss	Vector of DOFs at which translations/rotations are not measured for SS error function
vudofsf1	Vector of DOFs at which translations/rotations are not measured for SF1 error function
vudofsf2	Vector of DOFs at which translations/rotations are not measured for SF2 error function
vudofms	Vector of DOFs at which translations/rotations are not measured for MS error function
vudofmf1	Vector of DOFs at which translations/rotations are not measured for MF1 error function
vudofmf2	Vector of DOFs at which translations/rotations are not measured for MF2 error function
nmdofsstr1	Number of strain measurements for frame elements for SSTR error function
nmdofsstr2	Number of strain measurements for shell elements for SSTR error function
lcdofsstr1	Load cases used with strain measurements for frame elements for SSTR error function
lcdofsstr2	Load cases used with strain measurements for shell elements for SSTR error function
sglocs	Strain measurement locations for frame elements
vmdofsstr1	Vector of measured frame element strain locations
vmdofsstr2	Vector of measured shell element strain locations
nsysdof	System degrees of freedom matrix
utdofss	Cell array of measured displacements for SS error function
utdofsf1	Cell array of measured displacements for SF1 error function
utdofsf2	Cell array of measured displacements for SF2 error function
utdofsstr1	Cell array of measured frame element strains for SSTR error function
utdofsstr2	Cell array of measured shell element strains for MS error function
utdofms	Cell array of measured displacements for MS error function
utdofmf1	Cell array of measured displacements for MF1 error function
utdofmf2	Cell array of measured displacements for MF2 error function
Tms	Matrix to store frequency data for mode shapes used in MS error function
Tmf1	Matrix to store frequency data for mode shapes used in MF1 error function
Tmf2	Matrix to store frequency data for mode shapes used in MF2 error function
vfdofss	Matrix of applied forces for all degrees of freedom for SS error function
vfdofsf1	Matrix of applied forces for all degrees of freedom for SF1 error function
vfdofsf2	Matrix of applied forces for all degrees of freedom for SF2 error function
vfdofsstr1	Matrix of applied forces for all degrees of freedom for SSTR error function
vfdofsstr2	Matrix of applied forces for all degrees of freedom for SSTR error function
umess	Cell array of simulated measurement errors for SS error function
umesf1	Cell array of simulated measurement errors for SF1 error function
umesf2	Cell array of simulated measurement errors for SF2 error function
umesstr1	Cell array of simulated measurement errors for frame strains in SSTR error function
umesstr2	Cell array of simulated measurement errors for shell strains in SSTR error function
sdfss	Matrix of standard deviation for applied forces for SS error function
sdfsf1	Matrix of standard deviation for applied forces for SF1 error function
sdfsf2	Matrix of standard deviation for applied forces for SF2 error function
sdfsstr1	Matrix of standard deviation for applied forces for SSTR error function (frame)

sdfsstr2	Matrix of standard deviation for applied forces for SSTR error function (shell)
sduss	Matrix of standard deviation for measured translations/rotations for SS error function
sdusf1	Matrix of standard deviation for measured translations/rotations for SF1 error function
sdusf2	Matrix of standard deviation for measured translations/rotations for SF2 error function
sdusstr1	Matrix of standard deviation for measured translations/rotations for SSTR error function (frame)
sdusstr2	Matrix of standard deviation for measured translations/rotations for SSTR error function (shell)
episs	Initial SS error function vector
episf1	Initial SF1 error function vector
episf2	Initial SF2 error function vector
episstr	Initial SSTR error function vector
epims	Initial MS error function vector
epimf1	Initial MF1 error function vector
epimf2	Initial MF2 error function vector
kinitial	Stiffness matrix
minitial	Mass matrix
epss	SS error function vector
epsf1	SF1 error function vector
epsf2	SF2 error function vector
epsstr	SSTR error function vector
epms	MS error function vector
epmf1	MF1 error function vector
epmf2	MF2 error function vector

Chapter 6

Conclusions and Recommendations for Future Work

6.1 Conclusions

A new computer program, PARIS13.0, has been developed for automated finite element model calibration using simulated NDT data. Its performance has been demonstrated using three examples with simulated test data. The program was used to update the analytical model of the IASC-ASCE SHM Benchmark Structure, the Powder Mill Bridge Girder-3 Solid-Shell model, and the full-scale Powder Mill Bridge Frame-Shell model. This study validates the feasibility of using static and modal error functions for successful full-scale FE model updating. The robustness of statistical error function normalization using the covariance matrix of measured data improved accuracy of the parameter estimates in presence of measurement error.

PARIS13.0 provides a platform for full-scale FE model updating. In addition to the library of finite elements made available by interfacing SAP2000 with MATLAB, the SAP2000 software provides an excellent platform for FE model creation, analysis, and visualization. Also,

MATLAB is a powerful programming platform with access to various mathematical functions, matrix operations, and optimization routines. PARIS13.0 proves to be a major advancement towards using NDT data for baseline FE model development and FE model updating. Integration of commercial FEA packages with custom research software creates opportunities for computing efficacy of parameter estimation and FE model updating techniques. Upon satisfactory refinement, such integration will have immense potential for development of structural management and maintenance systems which are bound to benefit both the users and owners of civil engineering infrastructure.

6.2 Recommendations for Future Work

Development of PARIS13.0 computer program aligns with the following future research goals:

i) Availability of a SHM program for full-scale FE model updating based on NDT data.

ii) Carrying out parameter estimation studies for larger structural systems.

iii) Studying robustness of suggested improvements to parameter estimation process in the areas of sensor placement and accounting for modeling and measurement errors.

iv) Utilizing real NDT data from the instrumented Powder Mill Bridge to establish its baseline FE model.

The goals mentioned above require extensive parameter estimation studies based on simulated NDT data. The purpose of using simulated data will be to carry out systematic study of the FE model updating process in presence of measurement errors. Some other issues that need further attention are listed below:

i) Static strain (SSTR) and dynamic (MS and MF) error functions are currently available for frame and shell elements. Going further into Phase III for use of bridge NDT data, there is a potential to include strain measurements and mass parameters for solid finite elements as well.

ii) Development of statistical normalization scheme for adjusting for measurement errors for modal error functions.

iii) Implementation of Genetic Algorithms for parameter estimation.

The major development of Phase III, however, would be to include real NDT data for FE model updating. An example of such data is the data from the annual non-destructive load tests on the Powder Mill Bridge. The study would be geared towards establishing a baseline FE model for operational response prediction.

References

Allen, D. W., Clough, J. A., Sohn, H., and Farrar, C. R. (2003) "A Software Tool for Graphically Assembling Damage Identification Algorithms" Smart Structures and Materials 2003: Smart Systems and Nondestructive Evaluation for Civil Infrastructures, Proceedings Vol. 5057, pp.138-144

Banan, M. R., and Hjelmstad, K. D. (1993) "Identification of Structural Systems from Measured Response", Report No. SRS 579, UILU-ENG-93-2002, University of Illinois at Urbana-Champaign, Urbana, IL

Begg, R., Mackenzoland, A., Dodds, C., and Loland, O. (1976) "Structural Integrity Monitoring Using Digital Processing of Vibration Signals" 8th Annual Offshore Tech. Conf., pp. 305-311.

Black, C. J. and Ventura, C.E. (1998) "Blind Test on Damage Detection of a Steel Frame Structure", 16th International Modal Analysis Conference (IMAC XVI), Santa Barbara, California, February 2–5, Proceedings, pp. 623–629. 1998.

Chang, P. C., Flatau, A., and Liu, S. C. (2003) "Review Paper: Health Monitoring of Civil Infrastructure."

CSI Analysis Reference Manual for SAP2000, ETABS®, SAFE® and CSiBridge™, Computer and Structures Inc. (2011), "Modeling, Analysis and Design of Bridge Structures" SAP2000v15, Berkeley, California

CSi OAPI Documentation, Computer and Structures Inc. (2011), "Modeling, Analysis and Design of Bridge Structures" SAP2000v15, Berkeley, California

Deraemaeker, A., Reynders, E., DeRoeck, G., and Kullaa, J. (2008) "Vibration-Based Structural Health Monitoring using Output-Only Measurements under Changing Environment", Mechanical Systems and Signal Processing, Volume 22, Issue 1, January 2008, Pages 34-56

DiCarlo, C. J. (2008) "Applications of Statistics to Minimize and Quantify Measurement Error in Finite Element Model Updating" Master of Science Thesis, September 17, 2008, Tufts University

Doebling, S. W., Peterson, L. D., and Alvin, K. F. (1998) “Experimental Determination of Local Structural Stiffness by Disassembly of Measured Flexibility Matrices.” *J. Vibr. Acoust.*, 120(4), 949–957.

Doebling, S.W., Farrar, C.R., and Cornwell, P.J., "DIAMOND: A Graphical User Interface Toolbox for Comparative Modal Analysis and Damage Identification," in Proc. of Sixth International Conference on Recent Advances in Structural Dynamics, Southampton, UK, July 1997, pp. 399-412.

Farrar, C. R. and Jaureguiz, D. A. (1998) “Comparative Study of Damage Identification Algorithms Applied to a Bridge: I. Experiment” Smart Mater. Struct. **7** 704

Farrar, C. R. and Worden, K. (2006) “Introduction to Structural Health Monitoring” Phil. Trans. R. Soc. A 15 February 2007 vol. 365 no. 1851 303-315

FEMtools® Dynamic Design Solutions N.V. (DDS) Interleuvenlaan 64, B-3001, Leuven, Belgium 2012

Gornshteyn, I. (1992). “Parameter Identification of Structures using Selected Modal Measurements” MS thesis, Tufts University, Medford, Massachusetts

Iplikcioglu, M. (2012) “Non-Destructive Testing and Comparison of Approaches for Bridge Load Testing”, Master of Science Thesis, March 17, 2012, Tufts University

Johnson, E. A., Lam, H. F., Katafygiotis, L. S., and Beck J. L. (2004) “Phase I IASC-ASCE Structural Health Monitoring Benchmark Problem Using Simulated Data” Journal of Engineering Mechanics, Vol. 130, No. 1, January 1, 2004.

Kaouk, M. and Zimmerman, D. C. (1994) “Structural Damage Assessment using a Generalized Minimum Rank Perturbation Theory” AIAA Journal (ISSN 0001-1452), vol. 32, no. 4, p. 836-842

MATLAB R2011b Natick, Massachusetts: The MathWorks Inc., 2011.

National Bridges, The National Bridge Inventory Database (NIBS) *http://www.nationalbridges.com/guide-to-ratings*

Phelps, J. E. (2010) “Instrumentation, Non-destructive Testing and Finite Element Model Updating for Bridge Evaluation”, MS Thesis, May 2, 2010, Tufts University

Sanayei, M. and Onipede, O. (1991) "Damage Assessment of Structures Using Static Test Data" AIAA Journal, Vol.29, No.7, July 1991, pp.1174-1179

Sanayei, M. and Saletnik, M.J. (1996) “Parameter Estimation of Structures from Static Strain Measurements I: Formulation”

Sanayei, M., Imbaro, G. R., McClain, J. A. S., and Brown, L. C. (1997), "Structural Model Updating Using Experimental Static Measurements," ASCE, Journal of Structural Engineering, 1997, 123(6), pp. 792-798

Sanayei, M., Wadia-Fascetti, S., McClain, J.A.S., Gornshteyn, I., and Santini, E. M., "Structural Parameter Estimation Using Modal Responses and Incorporating Boundary Conditions," ASCE, Proceedings of the Structural Engineers World Congress, July 18-23, 1998, San Francisco, CA, Elsevier Science, New York, NY T118-2.

Sanayei, M., Onipede, O., and Babu, S. R. (1992), "Selection of Noisy Measurement Locations for Error Reduction in Static Parameter Identification," AIAA Journal, Vol. 30, No. 9, September 1992, pp. 2299-2309.

Sanayei, M., Wadia-Fascetti, S., Arya, B., and Santini, E. M. (2001) "Significance of Modeling Error in Structural Parameter Estimation" Computer-Aided Civil and Infrastructure Engineering, 16(1), 12-26.

Sanayei, M.., Bell E. S., and Rao N. (2007) “Model Updating of UCF Benchmark using PARIS”

Smith, I.F.C. (2009) “Multiple-Model Structural Identification” Encyclopedia of Structural Health Monitoring, p. 2199-2208, Chichester, UK: Wiley, 2009 Reference IMAC-CHAPTER-2009-001

Teughels, A. and De Roeck, G. (2003) “Structural damage identification of the highway bridge Z24 by FE model updating” Journal of Sound and Vibration 278 (2004) 589–610

Wang, B.S., Ni, Y.Q., Ko, J.M. (2011) “Damage detection utilizing the artificial neural network methods to a benchmark structure” International Journal of Structural Engineering, Volume 2, Number 03/2011

Wang, X., Hu, N., Fukunaga, H., Yao, Z.H. (2001) “Structural damage identification using static test data and changes in frequencies” Engineering Structures 23 (2001) 610–621

Zhu W. (2009) “Finite Element Model Updating using Application Programming Interface for PARIS”, MS Thesis, August 25, 2009, Tufts University