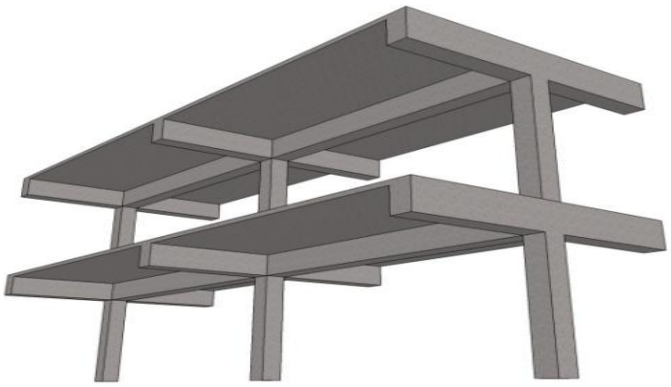
NATIONAL TECHNICAL UNIVERSITY OF ATHENS (NTUA)
SCHOOL OF CIVIL ENGINEERING
INSTITUTE OF STRUCTURAL ANALYSIS AND SEISMIC RESEARCH

STRUCTURAL DAMAGE LOCALIZATION AND QUANTIFICATION USING MODERN OPTIMIZATION TECHNIQUES

Postgraduate thesis

by
Apostolis Batavanis



Advisor:
Professor Manolis Papadrakakis

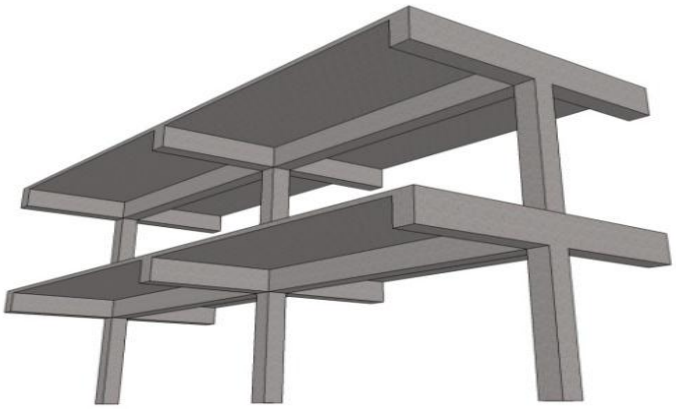
June 2012

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ (ΕΜΠ)
ΣΧΟΛΗ ΠΟΛΙΤΙΚΩΝ ΜΗΧΑΝΙΚΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΣΤΑΤΙΚΗΣ ΚΑΙ ΑΝΤΙΣΕΙΣΜΙΚΩΝ ΕΡΕΥΝΩΝ

ΕΝΤΟΠΙΣΜΟΣ ΚΑΙ ΠΟΣΟΤΙΚΟΠΟΙΗΣΗ ΖΗΜΙΩΝ ΣΕ ΚΑΤΑΣΚΕΥΕΣ ΜΕ ΧΡΗΣΗ ΣΥΓΧΡΟΝΩΝ ΤΕΧΝΙΚΩΝ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗΣ

Μεταπτυχιακή εργασία

Αποστόλης Μπαταβάνης



Επιβλέπων καθηγητής:
Μανόλης Παπαδρακάκης

Ιούνιος 2012

Acknowledgements

I would like to express my deepest appreciation and gratitude to my advisor, Professor Manolis Papadrakakis for entrusting me with the current assignment and for supporting me throughout the course of its completion. His general contribution to the academic years of my life was something extremely valuable for me both as a person and as an engineer.

I would also like to thank Dr. Vagelis Plevris, of the School of Civil Engineering, National Technical University of Athens (NTUA) for his invaluable assistance, his constant guidance and support in spite of the busy schedule during the preparation and completion of this study. I consider him as one of the most impactful persons during my academic years and hopefully in the future as well.

Athens, June 2012

Apostolis Batavanis

Abstract

Structural damage detection is a scientific field that has attracted a lot of interest in the scientific community during the recent years. There have been many studies and researches intending to find a reliable method to identify damage in structural elements both in position and in extend, which is also the main objective of the present study. Most of these methods use the dynamic analysis data of the structures as a diagnostic tool for damage.

In the present thesis the damage identification problem is addressed with two optimization techniques of different philosophy. More specifically, the population-based optimization algorithm PSO (Particle Swarm Optimization) and the mathematical optimization algorithm SQP (Sequential Quadratic Programming) are utilized in order to achieve the optimum results in the damage detection problem. The objective functions used in the optimization process are based on the dynamic analysis data of the structure such as modal flexibilities, natural frequencies and mode shapes. This data was acquired by developing a software that performs the dynamic analysis of structures using the Finite Element Method (FEM). The efficiency of each of the objective functions examined and each of the optimization algorithms is then demonstrated in the test examples section, where a simply supported beam with 10 elements and a fixed frame with 20 elements for different damage cases, are examined.

As far as the layout of this study is concerned, in the first chapter the theoretical background of the dynamic analysis software, developed in the context of the study is analyzed and in the second chapter the optimization techniques used are described in detail. Then, in the third chapter the damage identification problem is defined and the objective functions of the optimization problem are formulated. Finally, the fourth chapter lists the damage identification results of the proposed method in the test applications and the fifth and last chapter sums up the study by presenting the test applications and providing the conclusions of the whole study.

Περίληψη

Η ανίχνευση ζημιών σε κατασκευές αποτελεί ένα θέμα που έχει προσελκύσει μεγάλο ενδιαφέρον στους επιστημονικούς κύκλους κατά τη διάρκεια των τελευταίων ετών. Πραγματοποιήθηκαν πολλές μελέτες και έρευνες ώστε να βρεθεί μια αξιόπιστη μέθοδος για τον εντοπισμό και την ποσοτικοποίση ζημιών σε δομικά στοιχεία, κάτι που αποτελεί τον βασικό σκοπό και της παρούσας εργασίας. Οι περισσότερες από τις μεθόδους αυτές χρησιμοποιούν δεδομένα από τη δυναμική ανάλυση των κατασκευών ως διαγνωστικό εργαλείο για τη ζημία.

Στην παρούσα εργασία το πρόβλημα αναγνώρισης ζημιών αντμετωπίζεται με χρήση σύγχρονων τεχνικών βελτιστοποίησης. Πιο συγκεκριμένα, ο αλγόριθμος βελτιστοποίησης PSO (βελτιστοποίηση σμήνους σωματιδίων) που βασίζεται σε πληθυσμιακά μοντέλα και ο μαθηματικός αλγόριθμος βελτιστοποίησης SQP (διαδοχικός τετραγωνικός προγραμματισμός) χρησιμοποιούνται προκειμένου να επιτευχθούν ικανοποιητικά αποτελέσματα στο πρόβλημα ανίχνευσης βλαβών. Οι αντικειμενικές συναρτήσεις που χρησιμοποιούνται στη διαδικασία βελτιστοποίησης βασίζονται στα δεδομένα της δυναμικής ανάλυσης της κατασκευής όπως οι ιδιομορφικές ευελιξίες, οι φυσικές συχνότητες και οι ιδιομορφές. Τα στοιχεία αυτά υπογίστηκαν με την ανάπτυξη ενός λογισμικού που πραγματοποιεί την δυναμική ανάλυση κατασκευών με τη μέθοδο των πεπερασμένων στοιχείων (FEM). Η αποτελεσματικότητα της κάθε μιας από τις αντικειμενικές συναρτήσεις που προτείνονται και του κάθε ενός από τους αλγορίθμους βελτιστοποίησης αποδεικνύεται στη συνέχεια στις δοκιμαστικές εφαρμογές της αμφιέρειστης δοκού με 10 στοιχεία και του αμφίπακτου πλαισίου με 20 στοιχεία για διαφορετικές περιπτώσεις ζημιών.

Όσον αφορά στη διάρθρωση της παρούσας μελέτης, στο πρώτο κεφάλαιο αναλύεται το θεωρητικό υπόβαθρο του λογισμικου δυναμικής ανάλυσης, που αναπτύχθηκε στα πλαίσια της μελέτης και στο δεύτερο κεφάλαιο περιγράφονται λεπτομερώς οι τεχνικές βελτιστοποίησης που χρησιμοποιούνται. Στη συνέχεια, στο τρίτο κεφάλαιο ορίζεται το πρόβλημα αναγνώρισης ζημιών και διατυπώνονται οι αντικειμενικές συναρτήσεις του προβλήματος βελτιστοποίησης. Τέλος, στο τέταρτο κεφάλαιο παρατίθενται τα αποτελέσματα της προτεινόμενης μεθόδου για αναγνώριση ζημιών στις δοκιμαστικές εφαρμογές και το πέμπτο και τελευταίο κεφάλαιο συνοψίζει τη μελέτη, συμπεριλαμβάνοντας τα τελικά συμπεράσματα που προήλθαν από τις δοκιμαστικές εφαρμογές και τη συνολική μελέτη γενικά.

Table of Contents

List of Figures

List of Tables

Chapter 1

1 Dynamic analysis of structures

1.1 Equations of motion

1.1.1 Single degree of freedom system

A simple system with one degree of freedom (DOF) is considered as seen in Figure 1.1. In the idealized case (a), it consists of a mass *m* concentrated at the roof level, a massless spring that provides stiffness to the system, and a viscous damper that dissipates vibrational energy of the system. Each of these properties is concentrated in three separate, pure components (mass, stiffness and damping component). In the equivalent structural system (b) there is a massless column that contributes to the stiffness and damping of the idealized structure. In both cases the equation of motion is the following:

$$m \cdot \ddot{u}(t) + c \cdot \dot{u}(t) + k \cdot u(t) = p(t) \tag{1.1}$$

where:

m is the mass,

c is the viscous damping coefficient,

k is the stiffness,

p(t) is an external dynamic force, and

$\ddot{u}(t), \dot{u}(t), u(t)$ are the acceleration, velocity and displacement of the system, respectively.

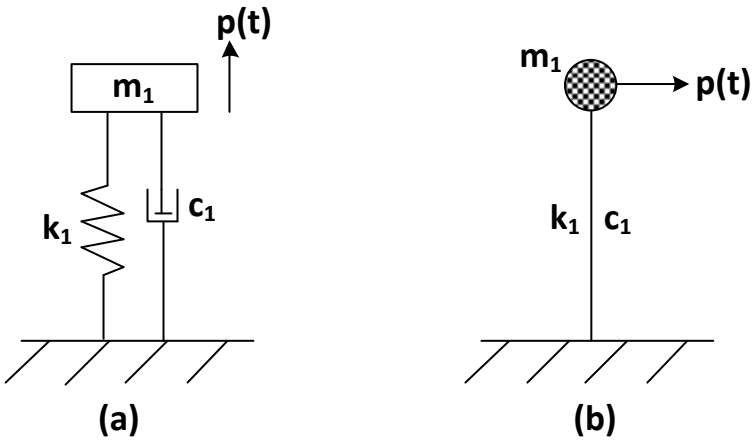


Figure 1.1 Example of a single DOF system: (a) idealized, (b) equivalent structural.

1.1.2 Systems with multiple degrees of freedom

A system with multiple DOFs and more specifically a three-story system is considered as seen in Figure 1.2. In the idealized case (a), the system consists of three masses connected by linear springs and linear viscous dampers subjected to external forces $p_1(t)$, $p_2(t)$ and $p_3(t)$, whereas in the equivalent structural system (b) there are three massless columns contributing to its stiffness and damping properties. The mass is in reality distributed throughout the structure, but in an idealized case it is concentrated at the floor levels. This assumption is generally appropriate for multistory buildings because most of the building mass is indeed at the floor levels. The equilibrium of motion for this idealized multi-story structure is the following:

$$[M]\cdot\{\ddot{u}(t)\}+[C]\cdot\{\dot{u}(t)\}+[K]\cdot\{u(t)\}=\{P(t)\} \tag{1.2}$$

where:

$[M]$ is the mass matrix,

$[C]$ is the viscous damping coefficient matrix,

$[K]$ is the stiffness matrix,

$\{P(t)\}$ is the external dynamic forces vector, and

$\{\ddot{u}(t)\}, \{\dot{u}(t)\}, \{u(t)\}$ are the acceleration, velocity and displacement vectors of the system, respectively.

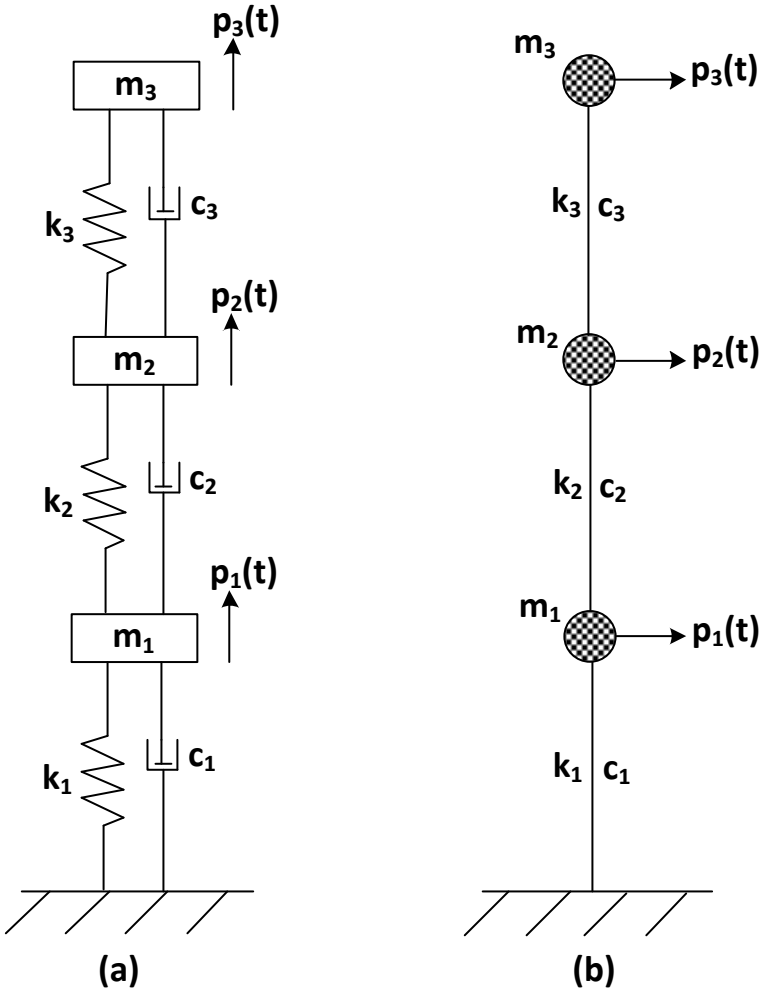


Figure 1.2 Example of a multiple DOFs system: (a) idealized, (b) equivalent structural.

More specifically, for the case of the threestory structure the matrices in Eq. (1.2) are calculated as follows:

Mass matrix (diagonal form): $$[M]=\begin{bmatrix} m_1 & 0 & 0 \\ 0 & m_2 & 0 \\ 0 & 0 & m_3 \end{bmatrix}$$

Stiffness matrix: $$[K]=\begin{bmatrix} k_1+k_2 & -k_2 & 0 \\ -k_2 & k_2+k_3 & -k_3 \\ 0 & -k_3 & k_3 \end{bmatrix}$$

Damping matrix: $$[C]=\begin{bmatrix} c_1+c_2 & -c_2 & 0 \\ -c_2 & c_2+c_3 & -c_3 \\ 0 & -c_3 & c_3 \end{bmatrix}$$

External forces vector: $$\{P(t)\} = \begin{Bmatrix} p_1(t) \\ p_2(t) \\ p_3(t) \end{Bmatrix}$$

It should be noted that in this simplified system where the masses are concentrated in each story, the axial and rotational DOFs are not taken into account for the calculation of the previous matrices. Specifically, in the idealized three-story structure of Figure 1.3 the three DOFs considered, are the *v*-translational ones (v_1, v_2 and v_3).

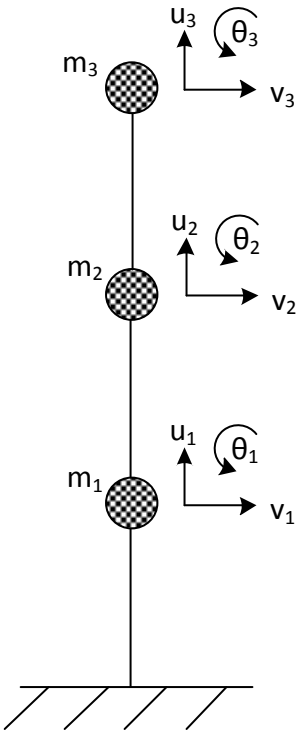


Figure 1.3 DOFs in a three-story idealized planar structure.

1.1.3 Free vibration

By free vibration we denote the motion of a structure without any dynamic excitation (external forces or support motion). Free vibration can be initiated by disturbing the structure from its equilibrium position by some initial displacements or velocities (Chopra, 2007).

For systems of multiple degrees of freedom without damping (*c*=0), the equation of motion becomes:

$$[M]\cdot\{\ddot{u}(t)\} + [K]\cdot\{u(t)\} = \{0\} \tag{1.3}$$

Eq. (1.3) represents N homogeneous differential equations that are coupled through the mass matrix, the stiffness matrix or both matrices, where N is the number of degrees of freedom.

A natural period of vibration T_n of this system is the time required for one cycle of the simple harmonic motion for the n-th natural modes. The corresponding natural circular frequency of vibration is ω_n and is associated with the natural period with the following equation:

$$T_n = \frac{2\pi}{\omega_n} \tag{1.4}$$

The natural frequency of this system represents the frequency at which it vibrates when excited and left to itself.

1.2 Dynamic characteristics of the structure

1.2.1 Natural frequencies and modes

In this paragraph the eigenvalue problem, whose solution gives the natural frequencies and modes of a system, is analyzed. The free vibration of an undamped system in one of its natural vibration modes, graphically displayed in Figure 1.2 for a three-DOF system, can be described mathematically by:

$$\{u(t)\} = q_n(t) \cdot \{\varphi_n\} \tag{1.5}$$

where:

$\{u(t)\}$ is the system displacements vector,

$\{\varphi_n\}$ is the deflected mode shape vector which does not vary with time and

$q_n(t)$ is the time variation of the displacements, which can be described by the following harmonic function:

$$q_n(t) = A_n \cos(\omega_n t) + B_n \sin(\omega_n t) \tag{1.6}$$

where A_n and B_n are constants of integration that can be determined from the initial conditions that initiate the motion.

Combining Eqs (1.5) and (1.6) we take:

$$\{u(t)\} = \{\varphi_n\} \cdot \left(A_n \cos(\omega_n t) + B_n \sin(\omega_n t)\right) \tag{1.7}$$

where the natural frequency ω_n and the mode shape vector $\{\varphi_n\}$ are unknown.

Also the second derivative of the displacement vector can be easily calculated as follows:

$$\{\ddot{u}(t)\} = \{\varphi_n\} \cdot \left(-\omega_n^2 \cdot A_n \cos(\omega_n t) - \omega_n^2 \cdot B_n \sin(\omega_n t)\right) \tag{1.8}$$

Eq. (1.8) can also be written as follows:

$$\{\ddot{u}(t)\} = -\omega_n^2 \cdot q_n(t) \cdot \{\varphi_n\} \tag{1.9}$$

Substituting then the displacement vector from Eq. (1.5) and its second derivative from Eq. (1.9) to the equation of motion for a free vibrated system (Eq. (1.3)), the result is:

$$\left(-\omega_n^2 \cdot [M] \cdot \{\varphi_n\} + [K] \cdot \{\varphi_n\}\right) \cdot q_n(t) = \{0\} \tag{1.10}$$

Ignoring the trivial solution $q_n(t) = 0$ that implies no motion of the structure, Eq. (1.10) results in the following useful condition:

$$[K] \cdot \{\varphi_n\} = \omega_n^2 \cdot [M] \cdot \{\varphi_n\} \tag{1.11}$$

which can also be written as follows:

$$\left([K] - \omega_n^2 \cdot [M]\right) \cdot \{\varphi_n\} = \{0\} \tag{1.12}$$

This algebraic problem is called the matrix eigenvalue problem. When necessary it is called the real eigenvalue problem to distinguish it from the complex eigenvalue for systems with damping. This problem intends to find the natural frequencies ω_n and the mode shapes $\{\varphi_n\}$ of the structure, given its stiffness and mass matrix.

Eq. (1.12) consists of a set of N homogeneous algebraic equations for the N elements φ_{jn} ($j = 1, 2, \dots, N$). It can be easily observed that this set has again a trivial solution $\{\varphi_n\} = \{0\}$, which implies immobility. Its nontrivial solution is the following:

$$\det\left([K] - \omega_n^2 \cdot [M]\right) = 0 \tag{1.13}$$

In dynamic analysis Eq. (1.13) is known as the characteristic equation. This equation has N real and positive roots for ω_n, as the structural mass and stiffness matrices $[M]$ and $[K]$, are symmetric and positive definite. The positive definite property of the stiffness matrix is assured for all civil engineering structures supported in a way that rigid-body motion is prevented. As far as the mass matrix is concerned, its positive definite property is also assured because the lumped masses are nonzero in all DOFs retained in dynamic analysis. This fact will be analyzed more thoroughly in the following paragraph 1.3.1, p. 10, where the DOFs with zero lumped masses are eliminated with the procedure of static condensation.

The N roots of Eq. (1.13) determine the N natural frequencies ω_n ($n = 1, 2, \dots, N$) of vibration. These roots of the characteristic equation are also known as eigenvalues. With the natural frequencies ω_n known, Eq. (1.12) can be solved for the corresponding vectors $\{\varphi_n\}$ to within a multiplicative constant. These vectors are also known as eigenvectors and represent the natural mode shapes of vibration of the structure. The calculation of mode shapes in a structure is very useful, as they represent the shape that the building will take when vibrating in free motion.

So for a vibrating system with N DOFs there are N natural vibration frequencies ω_n, N corresponding natural periods T_n and N natural mode shape vectors $\{\varphi_n\}$ with $n = 1, 2, \dots, N$. The term *natural* is used to emphasize the fact that these vibration properties are natural properties of the structure in free vibration, and they depend only on its mass and stiffness properties.

1.2.2 Calculation procedure

In bibliography there can be found many methods for calculating the dynamic characteristics of a structure by solving the eigenvalue problem described in the previous paragraph. In Eq. (1.12) it can be observed that there are two matrices (mass and stiffness matrix), used to calculate the natural frequencies and mode shapes of the structure. However, the classic eigenvalue problem is solved by using a single matrix as shown in the following equation:

$$[A]\cdot[x]=[\Lambda]\cdot[x] \tag{1.14}$$

where:

$[A]$ is a given matrix,

$[\Lambda]$ is the eigenvalues matrix and

$[x]$ is the eigenvectors matrix.

So, having calculated the mass and stiffness matrices of the system as shown in paragraph 1.1.2, the dynamic characteristics of a structure with N degrees of freedom can be obtained with the suggested procedure consisting of the following steps (Mouzakis & Dertimanis, 2011), (Inman, 2006):

1. Calculation of the normalized stiffness matrix (symmetric):

$$\left[\tilde{K}\right]=[M]^{-1/2}\cdot[K]\cdot[M]^{-1/2} \tag{1.15}$$

2. Calculation of the symmetric eigenvalue problem for the matrix $\tilde{K}$:

$$\left[\tilde{K}\right]\cdot[R]=[R]\cdot[\Lambda] \tag{1.16}$$

   The columns of matrix $[R]$ contain the eigenvectors of $[\tilde{K}]$, whereas matrix $[\Lambda]$ is diagonal and its elements are the squares of the natural frequencies:

$$[R]=[r_1\ r_2\ r_3\dots r_N] \quad \text{and} \quad [\Lambda]=diag\left(\omega_1^2,\omega_2^2,\omega_3^2\dots\omega_N^2\right) \tag{1.17}$$

3. Calculation of the mode shape matrix as follows:

$$[\Phi]=[M]^{-1/2}\cdot[R] \tag{1.18}$$

   The columns of $[\Phi]$ matrix are the mode shapes of the structure:

$$[\Phi]=[\varphi_1\ \varphi_2\ \varphi_3\dots\varphi_N] \tag{1.19}$$

4. Arrangement of natural frequencies of the structure in sequence from smallest to largest $\left(\omega_1 < \omega_2 < \omega_3 < \ldots < \omega_N\right)$. This classification should be also applied to the corresponding natural mode shapes and natural periods.

   This classification intends to bring the natural frequencies and mode shapes of the structure with the biggest contribution in its vibration in the first place.

5. Normalization of mode shapes.

1.2.3 Normalization of modes

Following the procedure described in the previous paragraph, the mode shapes have been calculated in a matrix form of $N \times N$ dimensions. The first column of this matrix represents the first mode shape of the structure, the second column represents the second mode shape of the structure and so on.

$$\left[\Phi\right] = \begin{bmatrix} \varphi_{11} & \varphi_{12} & \varphi_{13} & \cdots & \varphi_{1N} \\ \varphi_{21} & \varphi_{22} & \varphi_{23} & \cdots & \varphi_{2N} \\ \varphi_{31} & \varphi_{32} & \varphi_{33} & \cdots & \varphi_{3N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \varphi_{N1} & \varphi_{N2} & \varphi_{N3} & \cdots & \varphi_{NN} \end{bmatrix} \tag{1.20}$$

The mode shapes corresponding to a specific natural frequency of a system are unique only to a multiplicative constant. Although the scaling of normal modes is arbitrary, for practical considerations mode shapes should be scaled (i.e., normalized) by a chosen convention. Some of the most common methods for mode shape normalization are analysed using the following random mode shape vectors of a simple three-story structure with masses m in each story, as an example:

$$\left\{\varphi_1\right\} = A_1 \cdot \begin{Bmatrix} 0.42 \\ 0.82 \\ 1.17 \end{Bmatrix}, \quad \left\{\varphi_2\right\} = A_2 \cdot \begin{Bmatrix} -0.91 \\ -0.79 \\ 0.88 \end{Bmatrix}, \quad \left\{\varphi_3\right\} = A_3 \cdot \begin{Bmatrix} -1.11 \\ 0.96 \\ -0.27 \end{Bmatrix}$$

where A_1, A_2, A_3 are arbitrary constants.

A. Mass normalization

The mode shapes normalized with this method must satisfy the following equation (Kelly, 2011):

$$\left\{\varphi_n\right\}^T \cdot \left[M\right] \cdot \left\{\varphi_n\right\} = 1 \tag{1.21}$$

Using Equation (1.21) for the first mode shape φ_1 of the example and assuming the same mass m for all DOFs the constant A_1 can be calculated:

$$A_1^2 \cdot \begin{bmatrix} 0.42 & 0.82 & 1.17 \end{bmatrix} \cdot \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & m \end{bmatrix} \cdot \begin{bmatrix} 0.42 \\ 0.82 \\ 1.17 \end{bmatrix} = 1 \Leftrightarrow \tag{1.22}$$

$$A_1 = \frac{0.672}{\sqrt{m}} \tag{1.23}$$

So the mass normalized first mode shape becomes:

$$\varphi_1 = \frac{1}{\sqrt{m}} \cdot \begin{Bmatrix} 0.28 \\ 0.55 \\ 0.79 \end{Bmatrix} \tag{1.24}$$

In the same way the other mass normalized mode shapes will be:

$$\varphi_2 = \frac{1}{\sqrt{m}} \cdot \begin{Bmatrix} -0.61 \\ -0.53 \\ 0.59 \end{Bmatrix}, \quad \varphi_3 = \frac{1}{\sqrt{m}} \cdot \begin{Bmatrix} -0.74 \\ 0.64 \\ -0.18 \end{Bmatrix} \tag{1.25}$$

Numerically this method results in a modal mass matrix that is an identity matrix. This normalization approach is appropriate for modal dynamic response calculations because it simplifies both computational and data storage requirements. When mass normalization is used with a model of a heavy, massive structure, the magnitude of each of the terms of the eigenvectors is very small.

B. Unity to largest element normalization

Another method of mode shape normalization is to normalize each mode, so that its largest element is equal to unity. For the example described before, the constants multiplying the mode shapes will take the following values:

$$\begin{aligned} A_1 &= 0.8547 \rightarrow \varphi_1 = \begin{Bmatrix} 0.36 \\ 0.70 \\ 1.00 \end{Bmatrix} \\ A_2 &= -1.0989 \rightarrow \varphi_2 = \begin{Bmatrix} 1.00 \\ 0.87 \\ -0.97 \end{Bmatrix} \\ A_3 &= -0.9009 \rightarrow \varphi_3 = \begin{Bmatrix} 1.00 \\ -0.86 \\ 0.24 \end{Bmatrix} \end{aligned} \tag{1.26}$$

C. Unity to top story normalization

Other times it may be advantageous to normalize each mode so that the element corresponding to a particular degree of freedom such as the top floor of a structure is unity. In this case the constants that multiply the mode shapes of the example would take the following values:

$$A_1 = 0.8547 \rightarrow \varphi_1 = \begin{Bmatrix} 0.36 \\ 0.70 \\ 1.00 \end{Bmatrix}$$
$$A_2 = 1.1364 \rightarrow \varphi_2 = \begin{Bmatrix} -1.03 \\ -0.90 \\ 1.00 \end{Bmatrix} \quad (1.27)$$
$$A_3 = -3.7037 \rightarrow \varphi_3 = \begin{Bmatrix} 4.11 \\ -3.56 \\ 1.00 \end{Bmatrix}$$

1.3 Effective stiffness matrix

In the previous paragraphs it became clear that in order to perform the dynamic analysis of any structure the top priority is to calculate its mass and stiffness matrix. In dynamic analysis the stiffness matrix should not contain the degrees of freedom to which zero masses are assigned and the ones which have zero displacements. This is achieved by applying the following procedures:

- Static condensation of the rotational degrees of freedom of the structure.
- Rearrangement of the stiffness matrix by subtracting the constrained degrees of freedom of the structure.

Following these procedures the effective stiffness matrix, which can be used in the dynamic analysis of the structure, is obtained.

1.3.1 Static condensation

In static analysis all the degrees of freedom are included i.e. 3 DOFs for each node for a two-dimensional beam or frame. The static condensation method is used to eliminate the rotational degrees of freedom of a structure from dynamic analysis on to which zero masses are assigned. This method is based on the assumption that the rotational inertia terms in the rotational DOFs of the mass matrix can be neglected.

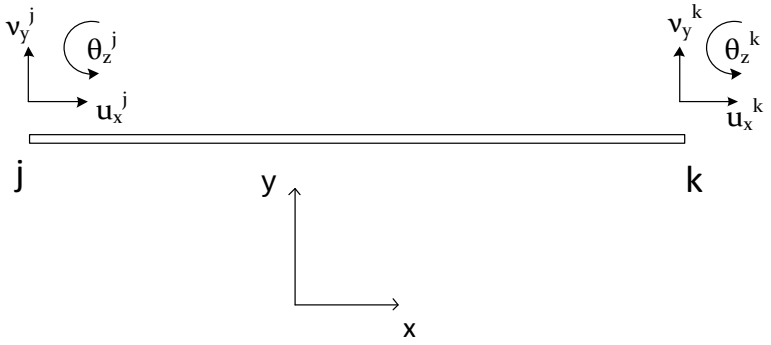


Figure 1.4 DOFs in a single beam element in 2D.

For the sake of convenience, a simple beam element in the two-dimensional space (Figure 1.4) with 3 DOFs for each node (2 translational and 1 rotational) is considered and its stiffness matrix is calculated as follows (Papadrakakis, 1996):

$$[K] = \begin{array}{c} \\ u_x^{\,j} \\ v_y^{\,j} \\ \theta_z^{\,j} \\ u_x^{\,k} \\ v_y^{\,k} \\ \theta_z^{\,k} \end{array} \begin{array}{c} \begin{array}{cccccc} u_x^{\,j} & v_y^{\,j} & \theta_z^{\,j} & u_x^{\,k} & v_y^{\,k} & \theta_z^{\,k} \end{array} \\ \left[\begin{array}{cccccc} \frac{EA}{L} & 0 & 0 & -\frac{EA}{L} & 0 & 0 \\ 0 & \frac{12EI}{L^3} & \frac{6EI}{L^2} & 0 & -\frac{12EI}{L^3} & \frac{6EI}{L^2} \\ 0 & \frac{6EI}{L^2} & \frac{4EI}{L} & 0 & -\frac{6EI}{L^2} & \frac{2EI}{L} \\ -\frac{EA}{L} & 0 & 0 & \frac{EA}{L} & 0 & 0 \\ 0 & -\frac{12EI}{L^3} & -\frac{6EI}{L^2} & 0 & \frac{12EI}{L^3} & -\frac{6EI}{L^2} \\ 0 & \frac{6EI}{L^2} & \frac{2EI}{L} & 0 & -\frac{6EI}{L^2} & \frac{4EI}{L} \end{array}\right] \end{array} \tag{1.28}$$

where:

E is the modulus of elasticity,

A is the section area

I is the moment of inertia of the section and

L is the length of the element.

In dynamic analysis of the beam element the axial DOFs ($u_x^{\,j}$ and $u_x^{\,k}$) can be easily subtracted from the stiffness matrix of the structure, as they are uncoupled from the other DOFs of the element. From the remaining DOFs it should be kept in mind that the masses are ideally assigned to the y-translational ones. This assumption, however, is not valid in frame elements, where the masses are assigned to both x- and y-translational DOFs. As a result the rotational degrees of freedom are eliminated from the final

stiffness matrix of the structure with the procedure of static condensation. It should be noted that the DOFs that are condensated, are still part of the rest calculations in the analysis of the structure. Also, this procedure should be applied after the calculation of the stiffness matrix of the whole structure and not at the stiffness matrix of each element, as it would lead to mechanisms.

In static condensation the final stiffness matrix of the structure is divided into 4 sub matrices:

K_{cc} - includes the rows and columns of the non-condensated DOFs.

K_{ce} - includes the rows of the non-condensated and the columns of the condensated DOFs.

K_{ec} - includes the rows of the condensated and the columns of the non-condensated DOFs.

K_{ee} - includes the rows and columns of the condensated DOFs.

The mathematical equation that calculates the condensated stiffness matrix $[K_c]$ is the following:

$$[K_c] = [K_{cc}] - [K_{ce}][K_{ee}]^{-1}[K_{ec}] \tag{1.29}$$

and its dimensions are equal to the non-condensated degrees of freedom.

Example: We consider a random stiffness matrix of a two-point supported beam with 3 elements (Figure 1.5) that includes only the *y*-translational and rotational DOFs as previously explained.

Given data:

$E = 21 \cdot 10^7$ kN/m^3,

$I = 0.25 \cdot 10^{-3}$ m^4,

$L = 2$ m (element length), for each elememt.

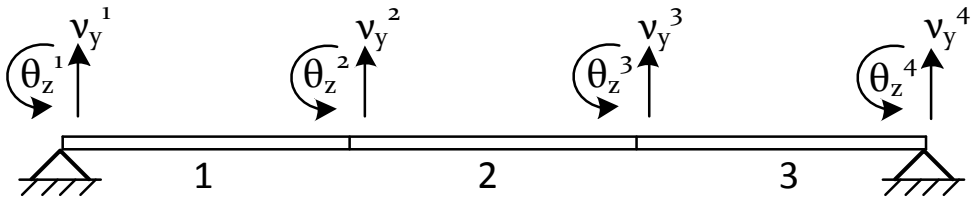


Figure 1.5 Degrees of freedom in a beam with 3 elements.

The stiffness matrix of the 3-element beam is then the following:

$$[K] = \begin{array}{c} \\ v_y^1 \\ \theta_z^1 \\ v_y^2 \\ \theta_z^2 \\ v_y^3 \\ \theta_z^3 \\ v_y^4 \\ \theta_z^4 \end{array} \begin{array}{c} \begin{array}{cccccccc} v_y^1 & \theta_z^1 & v_y^2 & \theta_z^2 & v_y^3 & \theta_z^3 & v_y^4 & \theta_z^4 \end{array} \\ \begin{bmatrix} 78750 & 78750 & -78750 & 78750 & 0 & 0 & 0 & 0 \\ 78750 & 105000 & -78750 & 52500 & 0 & 0 & 0 & 0 \\ -78750 & -78750 & 157500 & 0 & -78750 & 78750 & 0 & 0 \\ 78750 & 52500 & 0 & 210000 & -78750 & 52500 & 0 & 0 \\ 0 & 0 & -78750 & -78750 & 157500 & 0 & -78750 & 78750 \\ 0 & 0 & 78750 & 52500 & 0 & 210000 & -78750 & 52500 \\ 0 & 0 & 0 & 0 & -78750 & -78750 & 78750 & -78750 \\ 0 & 0 & 0 & 0 & 78750 & 52500 & -78750 & 105000 \end{bmatrix} \end{array} \quad (1.30)$$

Then the matrices of equation (1.29) are formed:

$$[K_{cc}] = \begin{array}{c} \\ v_y^1 \\ v_y^2 \\ v_y^3 \\ v_y^4 \end{array} \begin{array}{c} \begin{array}{cccc} v_y^1 & v_y^2 & v_y^3 & v_y^4 \end{array} \\ \begin{bmatrix} 78750 & -78750 & 0 & 0 \\ -78750 & 157500 & -78750 & 0 \\ 0 & -78750 & 157500 & -78750 \\ 0 & 0 & -78750 & 78750 \end{bmatrix} \end{array}, \quad [K_{ee}] = \begin{array}{c} \\ \theta_z^1 \\ \theta_z^2 \\ \theta_z^3 \\ \theta_z^4 \end{array} \begin{array}{c} \begin{array}{cccc} \theta_z^1 & \theta_z^2 & \theta_z^3 & \theta_z^4 \end{array} \\ \begin{bmatrix} 105000 & 52500 & 0 & 0 \\ 52500 & 210000 & 52500 & 0 \\ 0 & 52500 & 210000 & 52500 \\ 0 & 0 & 52500 & 105000 \end{bmatrix} \end{array}$$

$$[K_{ce}] = \begin{array}{c} \\ v_y^1 \\ v_y^2 \\ v_y^3 \\ v_y^4 \end{array} \begin{array}{c} \begin{array}{cccc} \theta_z^1 & \theta_z^2 & \theta_z^3 & \theta_z^4 \end{array} \\ \begin{bmatrix} 78750 & 78750 & 0 & 0 \\ -78750 & 0 & 78750 & 0 \\ 0 & -78750 & 0 & 78750 \\ 0 & 0 & -78750 & -78750 \end{bmatrix} \end{array}, \quad [K_{ec}] = \begin{array}{c} \\ \theta_z^1 \\ \theta_z^2 \\ \theta_z^3 \\ \theta_z^4 \end{array} \begin{array}{c} \begin{array}{cccc} v_y^1 & v_y^2 & v_y^3 & v_y^4 \end{array} \\ \begin{bmatrix} 78750 & -78750 & 0 & 0 \\ 78750 & 0 & -78750 & 0 \\ 0 & 78750 & 0 & -78750 \\ 0 & 0 & 78750 & -78750 \end{bmatrix} \end{array}$$

Finally the condensated matrix $[K_c]$ is calculated by using equation (1.29):

$$[K_c] = \begin{bmatrix} 10500 & -23625 & 15750 & -2625 \\ -23625 & 63000 & -55125 & 15750 \\ 15750 & -55125 & 63000 & -23625 \\ -2625 & 15750 & -23625 & 10500 \end{bmatrix}$$

This matrix can be used in the calculation of the dynamic characteristics of the beam, as it does not contain the rotational DOFs at which zero mass is assigned.

Observation: The procedure of static condensation can be also applied to frame elements without any significant changes in its implementation. Main difference is the fact that the masses are assigned to both *x*- and *y*-translational DOFs, as it was previously mentioned.

1.3.2 Rearrangement of stiffness matrix

In case there are constrained DOFs in a structure which limit the relative displacements, the rearrangement of these DOFs. This procedure is carried out through the calculation of the rearrangement matrix [V]. This matrix intends to separate the constrained DOFs in the stiffness matrix from the free ones. After the formation of the rearrangement matrix, the rearranged stiffness matrix [K_m] is calculated as follows (Papadrakakis, 1996):

$$[K_m] = [V][K][V]^T \tag{1.31}$$

where $[V]^T$ is the transpose of the rearrangement matrix [V].

The rearranged stiffness matrix [K_m] is divided then into the following 4 sub matrices:

K_{ff} - includes the rows and columns of the free DOFs.

K_{fs} - includes the rows of the free and the columns of the constrained DOFs.

K_{sf} - includes the rows of the constrained and the columns of the free DOFs.

K_{ss} - includes the rows and columns of the constrained DOFs.

Example:
In the example of the two-point supported beam with 3 elements (Figure 1.5), the y-translational DOFs of nodes 1 and 4 are constrained (v_y^1 and v_y^4). The rearrangement matrix [V], which includes binary values 0 and 1, is formed then as follows:

$$[V] = \begin{array}{c} \\ v_y^1 \\ \theta_z^1 \\ v_y^2 \\ \theta_z^2 \\ v_y^3 \\ \theta_z^3 \\ v_y^4 \\ \theta_z^4 \end{array} \begin{array}{c} \begin{array}{cccccccc} v_y^1 & \theta_z^1 & v_y^2 & \theta_z^2 & v_y^3 & \theta_z^3 & v_y^4 & \theta_z^4 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

Thereafter by applying equation (1.31) the rearranged matrix [K_m] is calculated. This matrix is then divided into 4 sub matrices as follows:

$$[K_m] = \begin{array}{c} \\ \theta_z^1 \\ v_y^2 \\ \theta_z^2 \\ v_y^3 \\ \theta_z^3 \\ \theta_z^4 \\ \hline v_y^1 \\ v_y^4 \end{array} \begin{array}{c} \begin{array}{cccccc} \theta_z^1 & v_y^2 & \theta_z^1 & v_y^3 & \theta_z^3 & \theta_z^4 \end{array} \quad \begin{array}{cc} v_y^1 & v_y^4 \end{array} \\ \left[\begin{array}{cc|c} \multicolumn{2}{c|}{\multirow{2}{*}{K_{ff} \textit{(active DOFs)} 6×6}} & \multirow{2}{*}{K_{fs} 6×2} \\ & & \\ \hline \multicolumn{2}{c|}{\multirow{2}{*}{K_{sf} 2×6}} & \multirow{2}{*}{K_{ss} 2×2} \\ & & \end{array}\right] \end{array}$$

In the dynamic analysis of the beam the stiffness matrix $[K_{ff}]$ will be used, as it does not contain the constrained DOFs with zero displacements.

Generally the effective stiffness matrix for the dynamic analysis will come as a result of the combination of the procedures of static condensation (paragraph 1.3.1) and rearrangement of stiffness matrix (paragraph 1.3.2).

Observation: The equation (1.31) is a multiplication of matrices, which for large-scale structures could be very time-consuming for every algorithm. So in order to reduce this computational cost, the following algorithm of the stiffness matrix rearrangement is suggested (in Matlab programming language):

```
%% Form Rearrangement matrix V
ZeroDisplacements = 0; %Set known (zero) DOF displacements to zero
UnknownDisplacements = 0; %Set unknown DOF displacements to zero
for n=1:(NumDOFs) %Scan all DOFs
    node = fix((n-1)/2)+1; %Node where the DOF belongs to
    DOF_ID = n-2*(node-1); %Assign 1, 2, 3 for x-displacement, y-
displacement or z-rotation, respectively
    if BoundaryConditions(n,1)==1 %In case of constrained DOF
        ZeroDisplacements = ZeroDisplacements + 1; %Count known DOFs
        V1order(ZeroDisplacements)=n; %order known DOFs assending
    else UnknownDisplacements =UnknownDisplacements + 1; %In case of
unconstrained DOF, Count unknown DOFs
        V2order(UnknownDisplacements)=n; %order unknown DOFs ascending
    end
end

if UnknownDisplacements==0 Vorder = V1order;
else Vorder = [V2order V1order]; %Put together to form Vorder
end

Km = KGlobal(Vorder,Vorder);
Keff = Km(1:UnknownDisplacements,1:UnknownDisplacements)
```

Chapter 2

2 Optimization

2.1 Introduction

The term optimization refers to solving a problem in the best possible way. In most cases the goal is to minimize or maximize an objective function by choosing values for its variables within an acceptable range, given that some constraints are satisfied.

Optimization techniques are an integral part of modern structural engineering. This is due to the fact that designing a structure with the best combination of safety and cost factors, is of high priority nowadays. The aim of optimization methods is to design a structure in the best possible way, so that the accomplishment of safety requirements as well as the minimization of the weight of the structure, and therefore the cost of the structure, are both achieved.

In the general case the solution of an optimization problem consists of the three following steps:

i) Mathematical formulation of the optimization problem (objective function definitions, design variables and possible constraints of the problem).

ii) Selection of the suitable optimization algorithm.

iii) Application of the selected optimization algorithm to solve the problem.

As far as the selection of a suitable algorithm is concerned, the rapid development of programming technology has contributed to the formulation of a wide variety of optimization algorithms that can solve even large-scale structural problems. Each optimization algorithm presents, of course, some advantages and drawbacks in its operation. In the present thesis the optimization problems are solved using the following two methods:

- Particle Swarm Optimization (PSO)
- Sequential Quadratic Programming (SQP).

2.2 Particle Swarm Optimization (PSO)

2.2.1 Introduction

The Particle Swarm Optimization (PSO) is a method inspired by the behavior of different kinds of flocks (birds, bees, fishes, etc.), which is characterized by distinct social and psychological principles. These principles lead the flock to adapt its physical movement towards food seeking in a particular way, which ensures both the speed of the quest and the avoidance of potential adversities such as hostile predators. This method has been given considerable attention in recent years among the optimization research community.

More specifically, a swarm of birds or insects or a school of fish searches for food, resources or protection in a very typical manner. If a member of the swarm discovers a desirable path to go, the rest of the swarm will follow quickly. Every member searches for the best in its locality, learns from its own experience as well as from the others typically from the best performer among them. Even human beings show a tendency to behave in this way as they learn from their own experience, their immediate neighbors and the ideal performers in the society. The PSO method mimics the behavior described above in an iterative way that leads the swarm to the best possible path for food.

It is pretty clear that PSO is a population-based optimization method built on the premise that social sharing of information among the individuals can provide an evolutionary advantage. The fact that, as an optimization method based on population data, PSO requires a relatively small number of parameters, as it will be shown in the next subchapters, reduces the computational cost and facilitates the implementation of the algorithm. Due to its simple implementation, PSO can be used in both simple and large-scale structural optimization problems. In this way PSO has been a rather attractive optimization method in scientific circles.

The algorithm was first proposed by Kennedy and Eberhart (Kennedy & Eberhart, 1995). PSO has been used widely in the recent years and has been modified in a variety of versions that can handle the majority of optimization problems with or without the presence of constraints.

2.2.2 PSO for unconstrained optimization problems

According to the PSO method, a random population of candidate solutions is considered to be a particle moving through the multi-dimensional design space in search of the position of the global optimum. The particles coexist and cooperate with each other to achieve this position. Every particle can be characterized by its physical position in the design space and its speed of movement. Also, each particle has the ability to remember the best position it has passed so far or personal best (*Pbest*) and the best position that any other particle of the swarm has passed so far or global best (*Gbest*). In every iteration

the speed of the particle is updated in a stochastical way. Finally the old and new speed vectors are used in order to update the position of the particle iteratively.

2.2.2.1 Mathematical formulation of PSO

The update equations for the speed and the position of the particles are the following (Plevris V. , 2009):

$$\left\{v^j\left(t+1\right)\right\} = w\left\{v^j\left(t\right)\right\} + c_1\left\{r_1\right\} \circ \left(\left\{x^{Pb,j}\right\} - \left\{x^j\left(t\right)\right\}\right) + c_2\left\{r_2\right\} \circ \left(\left\{x^{Gb}\right\} - \left\{x^j(t)\right\}\right) \quad (2.1)$$

$$\left\{x^j\left(t+1\right)\right\} = \left\{x^j\left(t\right)\right\} + \left\{v^j\left(t+1\right)\right\} \quad (2.2)$$

where:

w : Inertia weight parameter.

$\left\{v^j\left(t\right)\right\}$: The velocity vector of particle *j* at time *t*.

$\left\{x^j\left(t\right)\right\}$: The position vector of particle *j* at time *t*.

$\left\{x^{Pb,j}\right\}$: Vector of the personal best location found by the particle *j* until current iteration.

$\left\{x^{Gb}\right\}$: Vector of the global best location found by the entire swarm up to the current iteration.

c_1, c_2 : The acceleration coefficients of PSO. They represent the degree of "confidence" in the best solution found by the individual particle (c_1 – cognitive parameter) and by the entire swarm (c_2 – social parameter).

$\left\{r_1\right\}, \left\{r_2\right\}$: Vectors containing random numbers with uniform distribution in the interval [0, 1].

Observation:

The symbol "$\circ$" of Eq. (2.1) denotes the Hadamard product. In mathematics, the Hadamard product is a binary operation that takes two matrices of the same dimensions, and produces another matrix where each element *ij* is the product of elements *ij* of the original two matrices. Here it is used to ensure that the various random numbers of vectors $\{r_1\}$ and $\{r_2\}$ are applied in each dimension of the particle for every iteration.

For example, the Hadamard product of two matrices A and B with the same dimensions (2 × 2) is the following:

$$\mathrm{A} \circ \mathrm{B} = \begin{bmatrix} a_{11}b_{11} & a_{12}b_{12} \\ a_{21}b_{21} & a_{22}b_{22} \end{bmatrix} \quad (2.3)$$

The following figure (Figure 2.1) shows the movement of a particle in the two-dimensional design space according to Eq. (2.1) and (2.2).

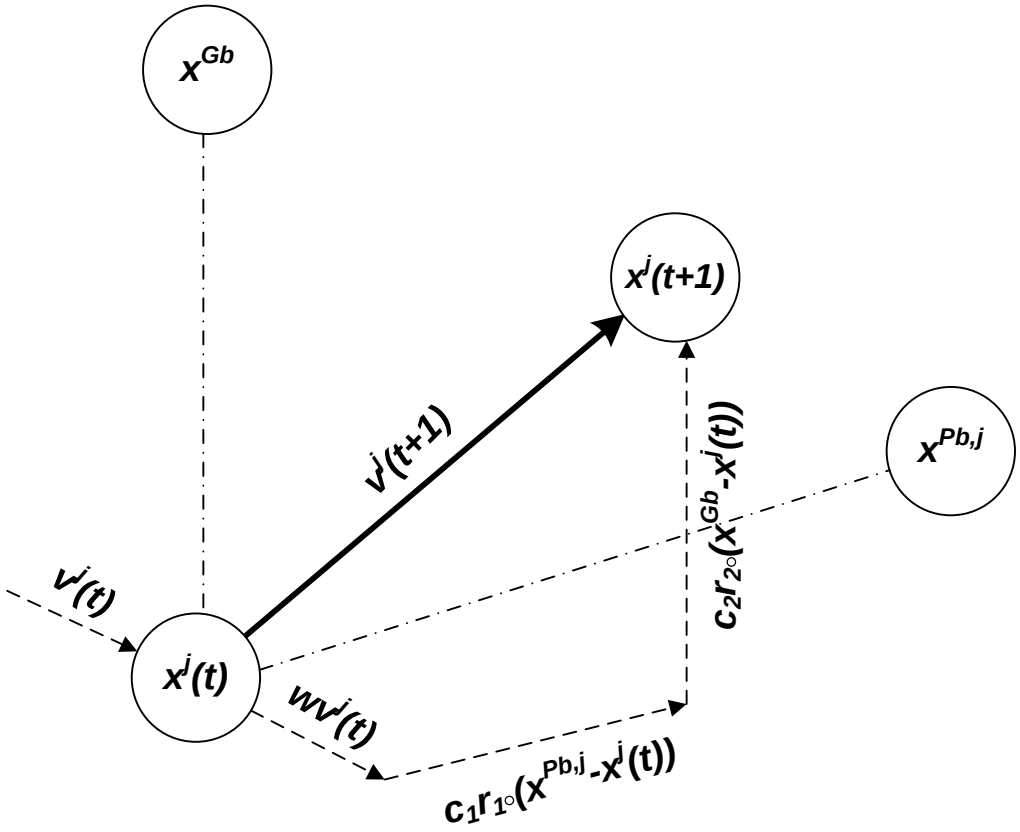


Figure 2.1 Movement of a particle in 2D design space.

The position of the particle j is $x^j(t)$ at time t and $x^j(t+1)$ at time $t+1$. It can be easily observed that the movement of the particle j is affected by:

i) The former speed of the particle, $v^j(t)$.

ii) The best position found by the particle or personal best, $x^{Pb,j}$.

iii) The best position found by the entire swarm or global best, x^{Gb}.

2.2.2.2 PSO parameters

Table 2.1 presents the main parameters used in the PSO algorithms and their common values. The weight inertia parameter *w,* which is included in Eq.(2.1), is a kind of scale in the range of which the design space is explored by the swarm. This parameter can improve significantly the convergence of the algorithm. In every iteration of the algorithm inertia weight is multiplied to the value of the current speed of the particles

and consequently affects the update equations. It can be observed that for large values of inertia weight the swarm explores a large part of the available design space.

Table 2.1 Main PSO parameters.

Symbol	Description	Details
NP	Number of particles	A typical range of values is 10-40. Depending on the difficulty and complexity of the problem a certain number is required to achieve satisfactory results.
n	Dimension of particles	It is determined by the problem to be optimized.
w	Inertia weight	Usually is set to a value less than 1, i.e. 0.95. It can also be updated during iterations.
$\{x\}^L$, $\{x\}^U$	Vectors containing the lower and upper bounds of the n design variables, respectively	They are determined by the problem to be optimized. Different ranges for different dimensions of particles can be applied in general.
$\{v_{max}\}$	Vector containing the maximum allowable velocity for each dimension during one iteration	Usually is set half the length of the allowable interval for the given dimension, but different values for different dimensions can be applied as well. $\left(v^{\max} = \frac{x^U - x^L}{2} \right)$
c_1, c_2	Cognitive and social parameters	Usually $c_1=c_2=2$. Other values can also be used provided that $0 < c_1+c_2 < 4$. (Perez and Behdinan 2007a).

Observation:

In case, during an iteration, the updated value of the speed of a particle results in updating the position x_i of a design variable in i dimension out of the lower and upper bounds x^L, x^U (specifically $x_i \le x^L$ or $x_i \ge x^U$), the value of this variable x_i is set to the value of the nearest bound (x^L or x^U). Also in order to avoid any position updates out of the specified design space by the next iteration of the algorithm, the corresponding coefficient v_i of the speed vector v is set equal to zero for this iteration.

2.2.2.3 PSO algorithm for unconstrained optimization problems

The procedure followed by the PSO algorithm to deal with optimization problems without constraints is briefly described in Table 2.2:

Table 2.2 Algorithmic steps of a basic PSO for unconstrained problems.

1) *Initialization* of the algorithm by randomly generating the position vectors of the particles within the design space $\left(\vec{x}_1, \vec{x}_2, ..., \vec{x}_{NP}\right)$ and calculating their corresponding speed vectors $\left(\vec{v}_1, \vec{v}_2, ..., \vec{v}_{NP}\right)$, where *NP* the number of particles.
2) *Analysis and evaluation* of the fitness value for current positions of the particles $\left(F(\vec{x}_1), F(\vec{x}_2), ..., F(\vec{x}_{NP})\right)$, where *NP* the number of particles.
3) *Personal bes*t: If the current fitness value is better than the best fitness value (*Pbest*) in the particle's history then this value is set as the new *Pbest* and the current position as the new $x^{Pb,j}$ for each *j* particle.
4) *Global best*: The best fitness value of all the particles' *Pbest* is set as *Gbest* and the corresponding position as the new x^{Gb}.
5) *Update*: Calculate particle velocity from Eq. (2.1) and update particle position from Eq. (2.2).
6) *Feasibility check*: If for any dimension i $x_i \le x^L$ or $x_i \ge x^U$ then $x_i = x^L$ or $x_i = x^U$, respectively and $v_i = 0$.
7) *Termination:* For the number of iterations of the problem the steps 2-6 are repeated and then the results are reported.

2.3 Mathematical optimization

2.3.1 Description of the SQP method

Mathematical (gradient-based) optimization methods are generally considered as local methods. They exhibit fast convergence by exploiting gradient information but they cannot guarantee the estimation of the global optimum, as they can be easily trapped in local minima. These methods require user-defined initial estimates of the solution (Plevris & Papadrakakis, 2011).

The mathematical optimizer used in this study is a SQP method. SQP methods are the standard general purpose mathematical programming algorithms for solving nonlinear programming (NLP) optimization problems. They are also considered to be the most suitable methods for solving structural optimization problems with the mathematical programming approach. Such methods make use of local curvature information derived from linearization of the original functions, by using their derivatives with respect to the design variables at points obtained in the process of optimization.

The mathematical formulation of the general structural optimization problem can be stated as follows:

$$\begin{aligned} &\min_{x\in\Re^n} f(x) \qquad \{x\} = \{x_1, x_2, \ldots, x_n\}^T \\ &\text{Subject to} \\ &g_i(x) \le 0, \qquad i = 1, \ldots, m \;\rightarrow\; \text{Inequality constraints} \\ &\{x\}^L \le \{x\} \le \{x\}^U \;\rightarrow\; \text{Parameter bounds} \end{aligned} \tag{2.4}$$

Where:

$\{x\}$ is a vector of length n containing the design variables,

$f(\{x\})$: $\Re^n \rightarrow \Re$ is the objective function, whose value should be minimized,

$g(\{x\})$: returns a vector of length m containing the values of inequality constraints.

$\{x\}^L$, $\{x\}^U$ are two vectors of length n containing the lower and upper bounds of the design variables, respectively.

Observation: The above mathematical formulation does not contain any equality constraints, as they are usually not the case in most structural optimization problems.

The general aim of the SQP is to transform the problem into an easier sub problem that can then be solved and used as the basis of an iterative process. More specifically, given the problem description of Eq. (2.4), the SQP method proceeds with the conversion of the NLP problem into a sequence of Quadratic Programming (QP) sub problems based on a quadratic approximation of the Lagrangian function:

$$L(x,\lambda) = f(x) + \sum_{i=1}^{m} \lambda_i g_i(x) \tag{2.5}$$

where λ_i are the Lagrange multipliers under the nonnegativity restriction for the inequality constraints. By linearizing the nonlinear constraints the QP sub problem can be obtained, which has the following form:

$$\begin{aligned} &\min_{d\in\Re^n} q(d) = \frac{1}{2}\{d\}^T \{H_k\}\{d\} + \nabla f\left(x_k\right)^T \{d\} \\ &\text{Subject to} \\ &\nabla g_i\left(x_k\right)^T \{d\} + g_i\left(x_k\right) \le 0, \; i = 1, \ldots, m \end{aligned} \tag{2.6}$$

Where:

$\{d\}$ is the search direction and

$\{H_k\}$ is a positive definite approximation of the Hessian matrix of the Lagrangian function of Eq. (2.5).

The SQP implementation consists of the two following main stages:

- Updating the Hessian matrix
- Quadratic Programming solution

Updating the Hessian Matrix: At each major iteration a positive definite quasi-Newton approximation of the Hessian of the Lagrangian function, H, is calculated using the Broyden-Fletcher-Goldfarb-Shanno (BFGS) method, where λ_k, $k = 1,\ldots,m$, is an estimate of the Lagrange multipliers.

$$H_{k+1} = H_k + \frac{q_k q_k^T}{q_k^T s_k} - \frac{H_k^T s_k^T s_k H_k}{s_k^T H_k s_k} \tag{2.7}$$

where:

$$s_k = x_{k+1} - x_k \tag{2.8}$$

$$q_k = \left(\nabla f\left(x_{k+1}\right) + \sum_{i=1}^{m} \lambda_i \cdot \nabla g_i\left(x_{k+1}\right) \right) - \left(\nabla f\left(x_k\right) + \sum_{i=1}^{m} \lambda_i \cdot \nabla g_i\left(x_k\right) \right) \tag{2.9}$$

Quadratic Programming Solution: At each major iteration of the SQP method, a QP problem in the form of Eq. (2.6) is solved using an active set strategy (Gill, Murray, & Wright, 1991).

The solution procedure involves two phases. The first phase involves the calculation of a feasible point (if one exists). The second phase involves the generation of an iterative sequence of feasible points that converge to the solution. In this method an active set $\overline{A}_k$, is maintained that is an estimate of the active constraints (i.e., those that are on the constraint boundaries) at the solution point.

$\overline{A}_k$ is updated at each iteration k, and this is used to form a basis for a search direction $\hat{d}_k$. The search direction $\hat{d}_k$ is calculated and minimizes the objective function while remaining on any active constraint boundaries. The feasible subspace for $\hat{d}_k$ is formed from a basis Z_k whose columns are orthogonal to the estimate of the active set $\overline{A}_k$, so that the following equation is valid:

$$\overline{A}_k \cdot Z_k = 0 \tag{2.10}$$

Thus a search direction, which is formed from a linear summation of any combination of the columns of Z_k, is guaranteed to remain on the boundaries of the active constraints.

The matrix Z_k is formed from the last $m - l$ columns of the QR decomposition of the matrix $\overline{A}_k^T$, where l is the number of active constraints and $l < m$. That is, Z_k is given by

$$\left[Z_k\right] = Q\left[:, l+1:m\right] \tag{2.11}$$

Where:

$$Q^T \bar{A}_k^T = \begin{bmatrix} R \\ 0 \end{bmatrix} \tag{2.12}$$

Once Z_k is found, a new search direction $\hat{d}_k$ is sought that minimizes $q(d)$ where $\hat{d}_k$ is in the null space of the active constraints. In this way, $\hat{d}_k$ is a linear combination of the columns of Z_k:

$$\hat{d}_k = [Z_k] \cdot \{p\} \tag{2.13}$$

where p is a random vector.

Then if the quadratic is viewed as a function of p, by substituting for $\hat{d}_k$ in Eq. (2.6):

$$q(p) = \frac{1}{2}\{p\}^T [Z_k]^T \{H\}[Z_k]\{p\} + \nabla f(x_k)^T [Z_k]\{p\} \tag{2.14}$$

Differentiating this with respect to p yields:

$$\nabla q(p) = [Z_k]^T \{H\}[Z_k]\{p\} + [Z_k]^T \nabla f(x_k) \tag{2.15}$$

Where:

$\nabla q(p)$ is the projected gradient of the quadratic function because it is the gradient projected in the subspace defined by Z_k, and

$Z_k{}^T H Z_k$ is the projected Hessian.

Because the Hessian matrix H is positive definite, due to the implementation of SQP, the minimum of the function $q(p)$ in the subspace defined by Z_k occurs when the projected gradient $\nabla q(p)$ is equal to zero, which is the solution of the system of linear equations. Using Eq. (2.15) we get:

$$[Z_k]^T \{H\}[Z_k]\{p\} = -[Z_k]^T \nabla f(x_k) \tag{2.16}$$

A step is then taken of the following form:

$$\{x\}_{k+1} = \{x\}_k + a_k \cdot \hat{d}_k \tag{2.17}$$

Where

α_k is the step length parameter for every k iteration of the SQP algorithm, and

$\hat{d}_k = [Z_k] \cdot \{p\}$.

At each iteration, to determine the step length parameter a line search is performed using a merit function and the new design point is then calculated using Eq. (2.17).

Summing up the previous description, the main algorithmic steps of the SQP method are presented in the following table:

Table 2.3 Algorithmic steps of mathematical optimization using SQP method.

1) Initialization of the algorithm by finding a feasible point to start, so that the bound and inequality constraints are not violated (k=1).
2) Solution of the QP sub problem described in the previous paragraph to determine the search direction $\hat{d}_k$ and set of λ_{k+1} as the vector of the Lagrange multiplier of the linear constraints obtained from the QP.
3) Calculation of the step length parameter α_k and of the new design point: $\{x\}_{k+1} = \{x\}_k + a_k \cdot \hat{d}_k$
4) Update of the Hessian matrix H_{k+1} from H_k using a quasi-Newton formula, Eq. (2.7).
5) Iteration of the algorithm by incrementing k.
6) Termination: For the number of iterations of the problem the steps 2-5 are repeated until one of the termination criteria is met.

2.3.2 Mathematical optimization in Matlab

In the present thesis the mathematical optimization is applied using Matlab programming language. The general constrained optimization problem can be stated as follows:

$$\min_x f(x) \quad \text{Subject to} \quad \left\{ \begin{array}{l} c(x) \le 0 \\ ceq(x) = 0 \\ A \cdot x \le b \\ Aeq \cdot x \le beq \\ lb \le x \le ub \end{array} \right\} \tag{2.18}$$

The main function to solve the optimization problem of Eq. (2.18) is fmincon, which attempts to find a constrained minimum of a scalar function of several variables starting at an initial estimate. The syntax of this function is the following:

$$[x,\ fval] = f\text{mincon}(\ fun, x0, A, b, Aeq, beq, lb, ub, nonlcon, options) \tag{2.19}$$

The main parameters and options of the previous function are presented in the following tables (Table 2.4 and Table 2.5):

Table 2.4 Main SQP algorithm parameters in Matlab.

Symbol	Description
x	The vector that minimizes the objective function of the problem.
fval	The value of the objective function at the solution *x*.
fun	The function to be minimized that accepts a vector *x* and returns a scalar *f*.
x0	Vector containing an initial guess of the solution x of the problem.
A, Aeq	Matrices for linear inequality and equality constraints, respectively.
b, beq	Vectors for linear inequality and equality constraints, respectively.
lb, ub	Vectors containing the lower and upper bounds of the design variables, respectively.
nonlcon	The function that computes the nonlinear inequality constraints $c(x) \leq 0$ and the nonlinear equality constraints $ceq(x) = 0$. `nonlcon` accepts a vector `x` and returns the two vectors c and ceq, which contain the nonlinear inequalities and equalites evaluated at x, respectively.

Table 2.5 Main SQP algorithm options in Matlab.

Symbol	Description	Details
Algorithm	Optimization algorithm used.	Trust-region, Active-set, SQP, Interior-point. In the present thesis SQP algorithm is used.
MaxFunEvals	Maximum number of function evaluations allowed.	It is a bound on the number of function evaluations. A typical value is 100 × number of variables.
MaxIter	Maximum number of iterations allowed.	It is a bound on the number of solver iterations. A typical value is 400-1000.
TolFun	Termination tolerance on the function value.	It is a lower bound on the change in the value of the objective function during a step. If $\|f(x_i) - f(x_i+1)\| <$ TolFun, the iterations end. TolFun is sometimes used as a relative bound, meaning iterations end when $\|f(x_i) - f(x_i+1)\| <$ TolFun$(1 + \|f(x_i)\|)$, or a similar relative measure. A typical value is 10^{-6}.
TolX	Termination tolerance on *x*.	It is a lower bound on the size of a step, meaning the norm of $(x_i - x_i+1)$. If the solver attempts to take a step that is smaller than TolX, the iterations end. TolX is sometimes used as a relative bound, meaning iterations end when $\|(xi - xi+1)\| <$ TolX$*(1 + \|xi\|)$, or a similar relative measure. A typical value is 10^{-6}.

Depending on the optimization problem the main parameters and options which are presented in the previous tables can take different values which help the optimizer to converge more accurately and faster. Also, in the present thesis the optimization problem is unconstrained so the parameters containing constraint values are left blank.

Finally the algorithm stops running when one of the following termination criteria is satisfied:

i. The maximum number of iterations or function evaluations is reached (*MaxFunEvals*, *MaxIter*).

ii. The termination tolerance criterion on the objective function (*TolFun*) or the design variable (*TolX*) is met.

Chapter 3

3 Damage identification

3.1 Introduction

The problem of the parametric identification of structural models using measured dynamic data has received significant attention in the recent years as a reliable method for detecting the extent and the location of damage. It is proven that changes in the dynamic characteristics of a structure are a symptom of damage occurrence. Specifically changes in the modal parameters, namely natural frequencies and mode shapes, can provide an accurate indication of damage in a structure. Since modal parameters are dependent on the physical properties of the structure, specifically stiffness and mass, the Finite Element Method (FEM) may be used as a tool for detecting and locating damaged elements in a structure through the update of modal parameters. This method can be used for both the extend and the location of damage even in large-scale structures.

Modal parameters can be easily and cheaply obtained from measured vibration responses. In the current thesis, however, it is assumed that both the value and the location of damage in a structure are known in advance, so that the modal data is obtained through FEM. This data is simulated to the experimental one by adding noise at the modal parameters in a suitable manner.

The problem of damage detection in structures is often converted into a single-objective optimization problem with or without constraints. The main focus in this optimization problem is to find a suitable objective function that represents the update of certain modal parameters in the best possible way. This function should contain the modal parameters that are sensitive to detect even minor structural changes due to damage and also its estimation should be relatively easy. In bibliography a wide variety of different objective functions for damage identification problems can be found and will be presented in the next paragraphs.

3.2 Damage identification model

If a structure is properly modeled using the FEM, structural damage mathematically affects the stiffness and mass matrices and physically its dynamic properties, such as natural frequencies and mode shapes (Yu & Chen, 2010). It can be easily observed that the global mass matrix remains the same in both the undamaged and damaged structure. This assumption is considered to be pretty accurate for the majority of real applications.
It is generally known that the eigenvalue equation of an undamaged structure is the following:

$$\left(\left[K\right]-\omega_j^{\,2}\left[M\right]\right)\left\{\varphi_j\right\}=0, \qquad j=1,2,\ldots\ldots,N_m \tag{3.1}$$

where:
$[K]$ – The global stiffness matrix.
$[M]$ – The global mass matrix.
$\{\varphi_j\}$ – Vibration mode shape vector.
ω_j – Natural frequency corresponding to j-th vibration mode shape $\{\varphi_j\}$.
N_m – The total number of vibration mode shapes obtained.

For a damaged structure Eq. (3.1) becomes as follows:

$$\left(\left[K_d\right]-\left(\omega_j^{\,d}\right)^2\left[M\right]\right)\left\{\varphi_j^{\,d}\right\}=0, \qquad j=1,2,\ldots\ldots,N_m \tag{3.2}$$

where:
$[K_d]$ – The damaged global stiffness matrix.
$[M]$ – The global mass matrix.
$\{\varphi_j^{\,d}\}$ – Vibration mode shape vector in the damaged structure.
$\omega_j^{\,d}$ – Natural frequency corresponding to j-th vibration mode shape $\{\varphi_j^{\,d}\}$.
N_m – The total number of vibration mode shapes obtained.

Eq. (3.2) forms the basis of the damage identification method used in the current thesis through an inverse procedure that makes use of the measured natural frequencies and mode shapes that are known in advance for the damaged structure. The main idea of the proposed method is that the model stiffness is updated using optimization techniques so that the calculated modal parameters match the measured ones.
In order to form the optimization problem, damage will be quantified using a scalar variable or index d whose values are between 0 and 1. The value zero refers to no damage at all whereas values close to one imply devastating damage in the corresponding elements of the structure.

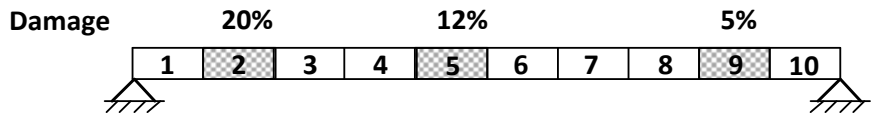


Figure 3.1 Simply supported beam divided in 10 elements with damage.

For example in the simply supported beam of Figure 3.1, where there is damage in some of the beam elements, the damage index will be the following:

$$d = \{0 \quad 0.20 \quad 0 \quad 0 \quad 0.12 \quad 0 \quad 0 \quad 0 \quad 0.05 \quad 0\},$$

with the values corresponding to each one of the 10 beam elements.
It becomes clear that damage index d provides all the data needed about damage in structures. As it was mentioned, the presence of damage in elements affects the value of stiffness in the corresponding elements. Since damage detection is carried out at the level of elements, the stiffness value for each element is assumed to change as follows:

$$\left[k_n{}^d\right] = \left[k_n\right] \cdot \left(1 - d_n\right) \tag{3.3}$$

where:
n – The number of element.
$[k_n{}^d], [k_n]$ – The element n stiffness matrix for the damaged and undamaged state, respectively.
d_n – The damage index value of the element.

The stiffness matrix of the damaged structure $[K_d]$ is then obtained through the assemblage of the damaged element stiffness matrices.

The formulation of the damage identification problem is based on the knowledge of the dynamic characteristics of the damaged structure, as they can be obtained through experimental processes. These experiments can give information about the natural frequencies, natural periods of vibration and the mode shapes of the structure. Of course only a limited data of these characteristics can be measured experimentally, but it will become clear that this data is adequate for these kinds of problems. The purpose of the damage identification problem is to find the location and the extent of damage in structures by using only this experimental data. In other words the damage index d is the solution of the problem, which can be solved with optimization techniques as it will be analyzed in the next paragraphs.

3.3 Single objective function for damage identification

3.3.1 Types of models in modal parameters calculation

The objective function for damage detection in the present thesis consists of the modal parameters calculated from the following models:

a) Experimental model
b) Numerical model

The experimental model consists of the dynamic characteristics data acquired by specific experiments. In the present thesis, however, this data is not available and for this reason the experimental values of natural frequencies and mode shapes of structures are obtained with the following procedure:
First of all, the damage index values are considered to be known in advance. The dynamic analysis of the structure, using equation (3.2), calculates then the modal parameters of the structure. These modal parameters represent the real accurate values and cannot be obtained precisely in real applications through experiments. To be more compatible with real experimental values of modal parameters, noise must be added to the calculated modal parameters.
During modal testing it is customary to assume that the frequencies of vibration are accurately determined and that is in the determination of the amplitudes of the mode shapes that the experimental errors occur. This assumption is usually valid since the frequency of shakers, even at resonance, can be quite accurately controlled. So the calculated data is simulated to the experimental one by adding noise to each mode shape using the following equation (Udwadia, 2005):

$$\varphi_{j,\exp}^{i} = \varphi_{j,cal}^{i} \cdot \left(1 + NoiseRatio \cdot \xi\right) \tag{3.4}$$

where:
$\varphi_{j,\exp}^{i}$ – The *i*-th value of the *j*-th experimental mode shape vector.
$\varphi_{j,cal}^{i}$ – The *i*-th value of the *j*-th calculated mode shape vector.
NoiseRatio – The percentage of noise added to the calculated data (usually 0-10%).
ξ – Uniformly distributed number between -1 and +1.

In the numerical model the damage index values are now the design variables of the optimization problem and are therefore not known in advance. Starting from random values for the damage index, the numerical stiffness matrix and the modal parameters are calculated for every iteration of the optimization algorithm. The main idea of the objective function to be minimized is the difference between modal parameter values between the numerical and the experimental model, as shown in the following equation:

The formulated unconstrained optimization problem is solved by using any optimization method (Genetic Algorithms, Particle Swarm Optimization, Mathematical Optimization etc.). The solution of the problem provides a pretty good estimate of the damage extent and location as shown in the chapter of applications.
In the next paragraph a wide variety of different objective functions for the optimization problem of damage detection will be presented.

3.3.2 The Modal Assurance Criteria (MAC, CoMAC)

The main idea of the objective function in damage identification problems is the minimization of the deviation between the numerical modal parameter values and the experimental ones. In order to compare the two sets of values for the numerical and experimental model, the use of simple mathematical formulas is imperative. More specifically the Modal Assurance Criterion (MAC) and the Co-ordinate Modal Assurance Criterion (CoMAC) are useful mathematical tools providing a measure of consistency and correlation between estimates of a modal vector (Allemang, 2003).

The Modal Assurance Criterion

One of the most popular tools for the quantitative comparison of modal vectors is the Modal Assurance Criterion (MAC). The purpose of this criterion is to indicate the correlation between two sets of mode shapes. Common application of the MAC criterion is the assessment between analytical and experimental modal vectors, as the complicated experimental results are often inconsistent to the ones obtained through FEM.
The MAC is calculated as the normalized scalar product of the two sets of vectors $\{\varphi_A\}$ and $\{\varphi_B\}$. Its formulation is the following:

$$MAC_{ij} = \frac{\left|\{\varphi_A\}_i^T \{\varphi_B\}_j\right|^2}{\left(\{\varphi_A\}_i^T \{\varphi_A\}_i\right)\left(\{\varphi_B\}_j^T \{\varphi_B\}_j\right)} \tag{3.5}$$

The MAC constant takes on values from zero, representing no consistent correspondence, to one, representing a consistent correspondence. In this manner, if the modal vectors under consideration truly exhibit a consistent relationship, the modal assurance criterion should approach unity.

Considering all modal vectors *i* and *j* of the two sets of mode shapes, the MAC matrix is calculated. The dimension of this matrix depends on the number of the mode shapes considered. For example if two mode shapes of an experimental and an analytical set are correlated, the dimension of the MAC matrix would be 2×2. If the diagonal terms of the MAC matrix approach unity and the other terms approach zero, a satisfactory degree of consistency between the two sets is achieved. Otherwise the deviation of diagonal MAC values from 1 could be interpreted as a damage indicator in structures.

As explained, if the modal assurance criterion has a value near zero, this is an indication that the modal vectors are not consistent. This can be due to the following reasons:

- The system is non-stationary. This can occur whenever the system is undergoing a change in mass or stiffness during the testing period.
- The system is nonlinear. System nonlinearities will appear differently in frequency response functions generated from different exciter positions or excitation signals. The modal parameter estimation algorithms will also not handle the different nonlinear characteristics in a consistent manner.

If the modal assurance criterion has a value near unity, this is an indication that the modal vectors are consistent. This does not necessarily mean that they are correct. If the same errors, random or bias, exist in all modal vector estimates, this is not delineated by the modal assurance criterion.

Example: We assume a simple three-story structure with the stiffness and mass matrices given as input data:

Undamaged state

$$[M]_{(kg)} = \begin{bmatrix} 100000 & 0 & 0 \\ 0 & 90000 & 0 \\ 0 & 0 & 80000 \end{bmatrix} \qquad [K]_{(N/m)} = \begin{bmatrix} 4\cdot10^8 & -2\cdot10^8 & 0 \\ -2\cdot10^8 & 4\cdot10^8 & -2\cdot10^8 \\ 0 & -2\cdot10^8 & 2\cdot10^8 \end{bmatrix}$$

The dynamic analysis of this undamaged state of the structure using Eq. (3.1) calculates its natural frequencies and mode shapes. Taking into account only the first two mode shapes the results of the dynamic analysis are the following:

$$[\varphi_A] = \begin{bmatrix} 0.4608 & -1.1689 \\ 0.8152 & -0.3562 \\ 1 & 1 \end{bmatrix} \quad \text{and} \quad [\omega_A{}^2] = \begin{bmatrix} 462.0958 & 0 \\ 0 & 3390.5343 \end{bmatrix}$$

The mode shapes have been normalized so that the value is unity in the top story of the structure.

Damaged state

In the damaged state we consider a damage of 40% and 20% in the first and second element of the structure, respectively. This damage is implemented in the algorithm as the following vector:

$$\{d\} = \begin{Bmatrix} 0.40 \\ 0.20 \\ 0 \end{Bmatrix}$$

As Eq. (3.3) suggests, the damage affects the stiffness of the elements and in this way the stiffness matrix becomes:

$$[K]^{d}_{(N/m)} = \begin{bmatrix} 2.8\cdot10^{8} & -1.6\cdot10^{8} & 0 \\ -1.6\cdot10^{8} & 3.6\cdot10^{8} & -2\cdot10^{8} \\ 0 & -2\cdot10^{8} & 2\cdot10^{8} \end{bmatrix}$$

As explained previously, the mass matrix remains the same for both the damaged and undamaged state.

Again taking into account only the first two mode shapes of the damaged structure the results of the dynamic analysis using Eq. (3.2) are now the following:

$$[\varphi_B] = \begin{bmatrix} 0.5628 & -1.3089 \\ 0.8743 & -0.0806 \\ 1 & 1 \end{bmatrix} \quad \text{and} \quad [\omega_B{}^2] = \begin{bmatrix} 314.1485 & 0 \\ 0 & 2701.4812 \end{bmatrix}$$

The following figure (Figure 3.2) depicts the mode shapes of the matrices $[\varphi_A]$ and $[\varphi_B]$ in the form of the three-story structure, so that the unity mode shape is at the top floor:

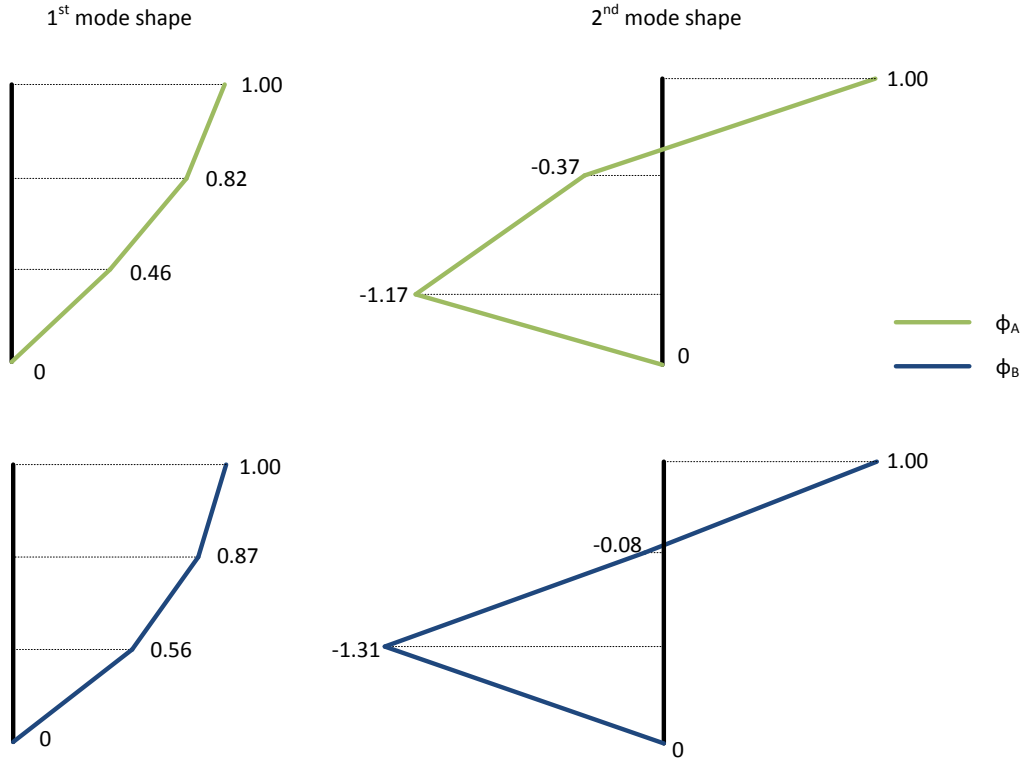


Figure 3.2 Drawing of two mode shapes of the example.

Using Equation (3.5) for every vector of mode shape matrices $[\varphi_A]$ and $[\varphi_B]$ the MAC matrix is the following:

$$mac\left([\varphi_A],[\varphi_B]\right)=\begin{bmatrix}0.9956 & 0.0215\\ 0.0002 & 0.9655\end{bmatrix}$$

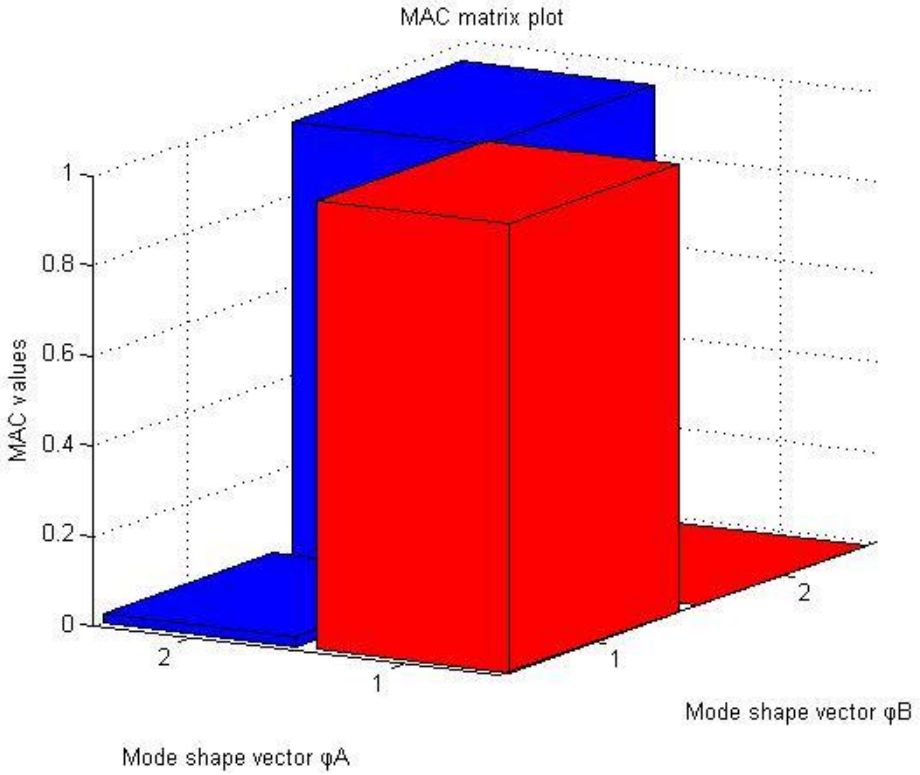


Figure 3.3 MAC matrix for the mode shapes of example 1.

As seen in the results of Figure 3.3, by collecting the diagonal terms of the MAC matrix the corresponding values are close to one. That means that the modal vectors of $[\varphi_A]$ and $[\varphi_B]$ mode shape matrices are consistent, although they have different values.

Alternative case

If we switch the modal vectors of mode shape matrix $[\varphi_B]$ and leave mode shape matrix $[\varphi_A]$ the same as before:

$$[\varphi_A]=\begin{bmatrix}0.4608 & -1.1689\\ 0.8152 & -0.3562\\ 1 & 1\end{bmatrix} \quad \text{and} \quad [\varphi_B]=\begin{bmatrix}-1.3089 & 0.5628\\ -0.0806 & 0.8743\\ 1 & 1\end{bmatrix}$$

Again, using Equation (3.5) for every vector of mode shape matrices $[\varphi_A]$ and $[\varphi_B]$ the MAC matrix is now the following:

$$mac\left([\varphi_A],[\varphi_B]\right)=\begin{bmatrix}0.0215 & 0.9956\\ 0.9655 & 0.0002\end{bmatrix}$$

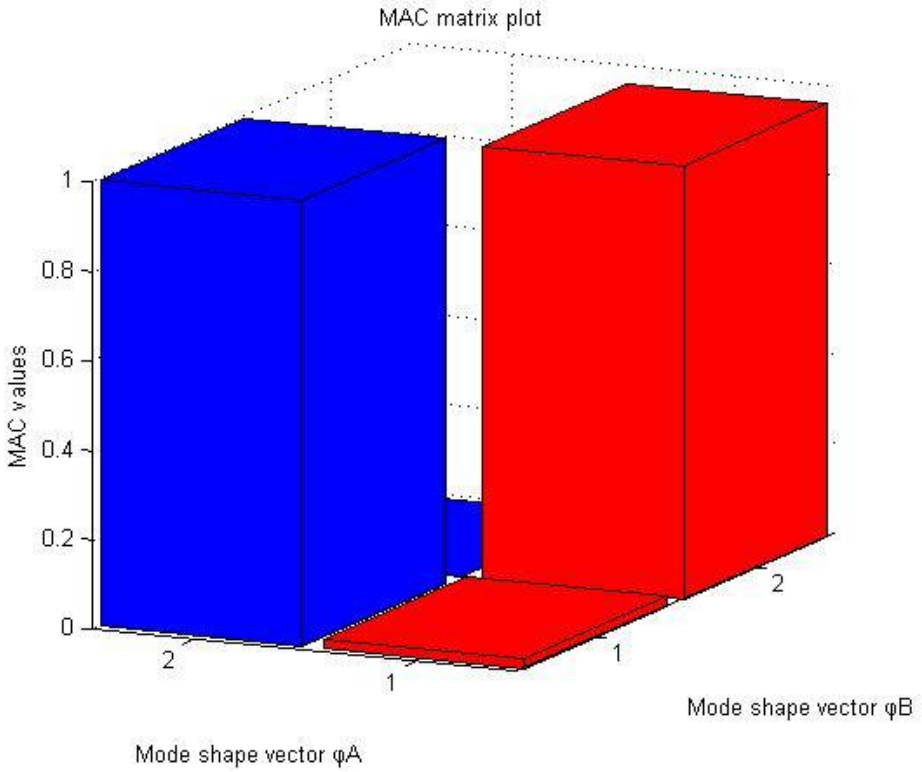


Figure 3.4 MAC matrix for the mode shapes of an alternative case.

In this simple case the inconsistency between the mode shape matrices $[\varphi_A]$ and $[\varphi_B]$, and more specifically between the two modal vectors of the matrices, is fairly obvious, as the diagonal terms are close to zero.

Observations:

1. Using the MAC matrix any modal sets can be evaluated in a rather efficient way and any inconsistencies between them can be accurately discovered.
2. If we switch the mode shape matrices in the calculation of the MAC matrix, the result is the transpose of the initial MAC matrix:

$$mac\left([\varphi_A],[\varphi_B]\right)=mac\left([\varphi_B],[\varphi_A]\right)^T \tag{3.6}$$

3. Because of the fact that the mode shapes represent the shape of the structure vibration and are not specific numbers, any linear combination of the mode shape values will produce the same MAC matrix. More specifically the following equation is valid:

$$mac\left([\varphi_A],[\varphi_B]\right)=mac\left([k\cdot\varphi_A],[l\cdot\varphi_B]\right) \tag{3.7}$$

Where k, l are random multiplicative constants.

The Co-ordinate Modal Assurance Criterion

In the comparison of two sets of modal vectors, one of the issues of interests is the influence of individual DOFs on the vector resemblance. The spatial dependence of the previously-presented correlation criterion (MAC) can be misleading. So, an extension of the modal assurance criterion is the coordinate modal assurance criterion (CoMAC). The CoMAC attempts to identify locations where mode shapes from two sets of test data do not agree, potentially indicating damage locations. The CoMAC is calculated over a set of mode pairs, analytical versus analytical, experimental versus experimental, or experimental versus analytical. For two sets of modes that are to be compared, there is a value of the CoMAC computed for each degree of freedom. For the coordinate *k* of a structure and using *n* mode shapes, the CoMAC factor is defined as follows (Rades, 2010):

$$COMAC_{(k)} = \frac{\left(\sum_{c=1}^{n} \left| \{\varphi_A\}_k^{(c)} \{\varphi_B\}_k^{(c)} \right| \right)^2}{\sum_{c=1}^{n} \left(\{\varphi_A\}_k^{(c)} \right)^2 \sum_{c=1}^{n} \left(\{\varphi_B\}_k^{(c)} \right)^2} \tag{3.8}$$

It can be observed that the CoMAC is basically a row-wise correlation of two sets of compatible vectors, which in MAC is done column-wise.

As it applies with MAC, the deviation of CoMAC values from 1 could also provide an indication of damage in structures. Unlike the MAC, the CoMAC does not have any difficulty in comparing modes that are close in frequency.

The only thing the CoMAC does, is to detect local differences between two sets of modal vectors. It does not identify modeling errors, because their location can be different from the areas where their consequences are felt. Another limitation is the fact that CoMAC weights all DOFs equally, irrespective of their magnitude in the modal vector.

Example: Using the previous example we have the two mode shape matrices $[\varphi_A]$ and $[\varphi_B]$:

$$[\varphi_A] = \begin{bmatrix} 0.4608 & -1.1689 \\ 0.8152 & -0.3562 \\ 1 & 1 \end{bmatrix} \quad \text{and} \quad [\varphi_B] = \begin{bmatrix} 0.5628 & -1.3089 \\ 0.8743 & -0.0806 \\ 1 & 1 \end{bmatrix}$$

Using Equation (3.8) for the matrices $[\varphi_A]$ and $[\varphi_B]$ the CoMAC values are the following:

$$comac([\varphi_A],[\varphi_B]) = \begin{Bmatrix} 0.9991 \\ 0.9010 \\ 1.0000 \end{Bmatrix}$$

Each value of the CoMAC vector corresponds to a certain DOF of the structure. For the simple three-story structure of the example the CoMAC vector consists of three values. From the results it can be said that all DOFs of the modal vectors of mode shape matrices $[\varphi_A]$ and $[\varphi_B]$ are highly correlated. Also from the type of normalization used for the mode shapes of the example, the CoMAC value of the top story DOF of the structure is expected to be unity.

Observations:

1. If we switch the mode shape matrices in the calculation of the CoMAC vector, the result is equal to the initial CoMAC vector:

$$comac\left([\varphi_A],[\varphi_B]\right) = comac\left([\varphi_B],[\varphi_A]\right) \tag{3.9}$$

2. As it applies with the MAC matrix, any linear combination of the mode shape values will produce the same CoMAC vector as well. More specifically the following equation is valid:

$$comac\left([\varphi_A],[\varphi_B]\right) = comac\left([k\cdot\varphi_A],[l\cdot\varphi_B]\right) \tag{3.10}$$

Where k, l are random multiplicative constants.

3.3.3 Flexibility matrix

As is well known, damage affects the stiffness matrix of the structure and more specifically it reduces the stiffness of the individual damaged elements. A reduction in stiffness corresponds to an increased instructural flexibility. In structural health-monitoring techniques it is advantageous to use changes in flexibility as an indicator of damage rather than using stiffness perturbations. This is due to the following reasons (Bernal, 2000 a):

- The flexibility matrix is dominated by the lower modes and so good approximations can be obtained even when only a few lower modes are employed.
- The flexibility matrices are directly attainable through the modes and mode shapes, determined by the system identification process
- Iterative algorithms usually converge the fastest to high eigenvalues.
- In flexibility-based methods, these eigenvalues correspond to the dominant low-frequency components in structural vibrations.

Therefore, the dynamically measured flexibility matrix which is easily calculated from the identified modal parameters, can be used as a damage identification method (Perera, Ruiz, & Manzano, 2007).
The flexibility matrix is defined as follows:

$$[F] = \sum_{j=1}^{N_m} \frac{1}{\omega_j^{\,2}} \cdot \{\varphi_j\} \cdot \{\varphi_j\}^T \tag{3.11}$$

where:
ω_j – Natural frequency corresponding to j-th vibration mode shape.
$\{\varphi_j\}$ – The j-th mode shape vector.
$\{\varphi_j\}^T$ – The transpose j-th mode shape vector.
N_m – The total number of vibration mode shapes obtained.

Or in matrix form:

$$[F]=[\Phi]\cdot[\Lambda]^{-1}\cdot[\Phi]^T \tag{3.12}$$

where:
$[\Phi]$, $[\Phi]^T$ – The mode shape matrix $[\{\Phi_1\}, \{\Phi_2\}, ..., \{\Phi_{Nm}\}]$ and its transpose, respectively.
$[\Lambda]$ – The spectral matrix containing the eigenfrequencies of N_m vibrating modes, diag(ω_j^2).

Each column of the flexibility matrix represents the displacement pattern of a structure associated with a unit force applied at the associated degree of freedom. As shown in equation (3.11), as the value of frequency increases the modal contribution to the flexibility matrix increases. As a result a good estimate of the flexibility matrix can be calculated with a small number of the first low-frequency modes.
As it was stressed previously, the objective function of the damage detection problem has to be formulated in terms of the discrepancy between the numerical (Finite Element Method) and the experimental quantities. In order to compare the experimental and the numerical values of the flexibility matrix, the modal flexibility assurance criterion (*MACFLEX*) is applied as follows:

$$MACFLEX_j = \frac{\left(\left|\left\{F_{numj}\right\}^T\left\{F_{\exp j}\right\}\right|\right)^2}{\left(\left\{F_{numj}\right\}^T\left\{F_{numj}\right\}\right)\left(\left\{F_{\exp j}\right\}^T\left\{F_{\exp j}\right\}\right)} \tag{3.13}$$

where $\{F_{num}\}_j$, $\{F_{exp}]_j$ the vectors of the flexibility matrix, corresponding for the j-th mode, for the numerical and experimental model, respectively. This criterion uses only the diagonal terms of the *MAC* matrix defined in Eq. (3.5), as it compares the same vectors of each set of flexibilities. In this way *MACFLEX* is a vector with as many values as the number of vectors in the flexibility matrices.
To consider the total number of flexibility vectors (N_m), a total co-ordinate modal flexibility assurance criterion is defined in the following way:

$$MACFLEX = \prod_{j=1}^{N_m} MACFLEX_j \tag{3.14}$$

As mentioned earlier an index equal to zero means no correlation between the two sets of flexibilities, whereas a value close to one means almost no change in flexibilities and

consequently a perfect correlation between the experimental and the updated numerical results. Therefore, the objective function to be minimized must be expressed as:

$$F_1 = 1 - MACFLEX \tag{3.15}$$

Example: Continuing the previous example we have the following sets of mode shapes and natural frequencies:

$$[\varphi_A] = \begin{bmatrix} 0.4608 & -1.1689 \\ 0.8152 & -0.3562 \\ 1 & 1 \end{bmatrix} \quad \text{and} \quad [\varphi_B] = \begin{bmatrix} 0.5628 & -1.3089 \\ 0.8743 & -0.0806 \\ 1 & 1 \end{bmatrix}$$

$$[\omega_A^2] = \begin{bmatrix} 462.0958 & 0 \\ 0 & 3390.5343 \end{bmatrix} \quad \text{and} \quad [\omega_B^2] = \begin{bmatrix} 314.1485 & 0 \\ 0 & 2701.4812 \end{bmatrix}$$

Using Eq. (3.12) the flexibility matrices for the undamaged and damaged state of the three-story structure of the example are the following:

$$[F_A] = \begin{bmatrix} 0.0009 & 0.0009 & 0.0007 \\ 0.0009 & 0.0015 & 0.0017 \\ 0.0007 & 0.0017 & 0.0025 \end{bmatrix} \quad \text{and} \quad [F_B] = \begin{bmatrix} 0.0016 & 0.0016 & 0.0013 \\ 0.0016 & 0.0024 & 0.0028 \\ 0.0013 & 0.0028 & 0.0036 \end{bmatrix}$$

For the two sets of flexibilities the *MACFLEX* criterion (Eq. (3.13)) is then applied:

$$macflex_j([F_A],[F_B]) = \begin{Bmatrix} 0.9960 \\ 0.9998 \\ 0.9916 \end{Bmatrix}$$

Finally using Eqs. (3.14) and (3.15) the total modal flexibility assurance criterion and the value of the objective function to be minimized is calculated:

$$macflex([F_A],[F_B]) = 0.9960 \times 0.9998 \times 0.9916 = 0.9874$$

$$F_1 = 1 - 0.9874 = 0.0126$$

3.3.4 Natural frequencies and mode shapes

On the other hand, the natural frequencies provide global information of the structure and they can be accurately identified through the dynamic measurements. Hence, the eigenfrequencies should be indispensable quantities to be used in the updating process. Furthermore, although very sensitive to local damage, modal flexibility provides a certain dispersion in the estimation of the damaged elements. To correct, to some extent, this deficiency as a second objective function, the modified total modal assurance criterion *(MTMAC)* dependent on mode shapes and frequencies has been adopted which is defined as follows:

$$MTMAC_j = \frac{MAC\Phi_j}{1+\left|\frac{\omega_{\exp j}^2 - \omega_{numj}^2}{\omega_{\exp j}^2 + \omega_{numj}^2}\right|} \tag{3.16}$$

Where:

$$MAC\Phi_j = \frac{\left(\left|\left\{\Phi_{numj}\right\}^T \left\{\Phi_{\exp j}\right\}\right|\right)^2}{\left(\left\{\Phi_{numj}\right\}^T \left\{\Phi_{numj}\right\}\right)\left(\left\{\Phi_{\exp j}\right\}^T \left\{\Phi_{\exp j}\right\}\right)} \tag{3.17}$$

where $\{\Phi_{num}\}_j$, $\{\Phi_{exp}]_j$ the vectors of the mode shape matrix, corresponding for the *j*-th mode, for the numerical and experimental model, respectively. This criterion uses only the diagonal terms of the MAC matrix defined in Eq. (3.5), as it compares the same modal vectors of each set of mode shapes. In the denominator of Eq. (3.16) ω_{numj} and ω_{expj} are the natural frequencies corresponding to the *j*-th mode. In this way the MTMAC is a vector with as many values as the number of mode shapes considered. To consider the total number of mode shapes (N_m) a total criterion is defined in the following way:

$$MTMAC = \prod_{j=1}^{N_m} MTMAC_{(j)} \tag{3.18}$$

As mentioned earlier an index equal to zero means no correlation between the two sets of mode shapes and natural frequencies, whereas a value close to one denotes almost a perfect correlation between the experimental and the updated numerical results. Therefore, the objective function to be minimized is expressed as follows:

$$F_2 = 1 - MTMAC \tag{3.19}$$

Example: For the two sets of natural frequencies and mode shapes of the previous example the MTMAC criterion (Eq. (3.16)) is then applied:

$$mtmac_j\left([\varphi_A],[\varphi_B],[\omega_A],[\omega_B],\right) = \begin{Bmatrix} 0.8363 \\ 0.8674 \end{Bmatrix}$$

Finally using Eqs. (3.18) and (3.19) the total MTMAC criterion and the value of the second objective function to be minimized is calculated:

$$mtmac\left([\varphi_A],[\varphi_B],[\omega_A],[\omega_B],\right) = 0.8363 \times 0.8674 = 0.7254$$

$$F_2 = 1 - 0.7254 = 0.2746$$

Observation: From the values of the two objective functions of the example it can be assumed that the *MTMAC* criterion is more sensitive to damage compared to the *MACFLEX* criterion. This assumption will be justified with more examples and applications in the next paragraphs.

3.3.5 Objective function using both *MACFLEX* and *MTMAC*

One of the main purposes of the current thesis was to find a way to use both of the objective functions described in the previous paragraphs, so that their advantages in the damage identification problems could be utilized in an optimization problem simultaneously.

Multiobjective approach

In bibliography there can be found many articles and references that describe the solution of a multiobjective optimization problem using two objective functions (Eq. (3.15) and Eq. (3.19)) (Perera, Ruiz, & Manzano, 2007).

However by definition a multiobjective optimization problem consists of objectives which have conflicting values. That means that when the one objective maximizes its value the other objective minimizes it. For example in engineering a classic multi-objective optimization problem intends to maximize the safety of a structure while at the same time to minimize its cost.

As opposed to single-objective optimization problems which accept one single-optimum solution, multiobjective optimization problems do not have a single-optimal solution, but rather a set of alternative solutions, named the Pareto-front curve, which are optimal in the sense that no other solutions in the search space are superior to them when all objectives are considered. A typical example of a Pareto curve for two objective functions can be seen in the following figure:

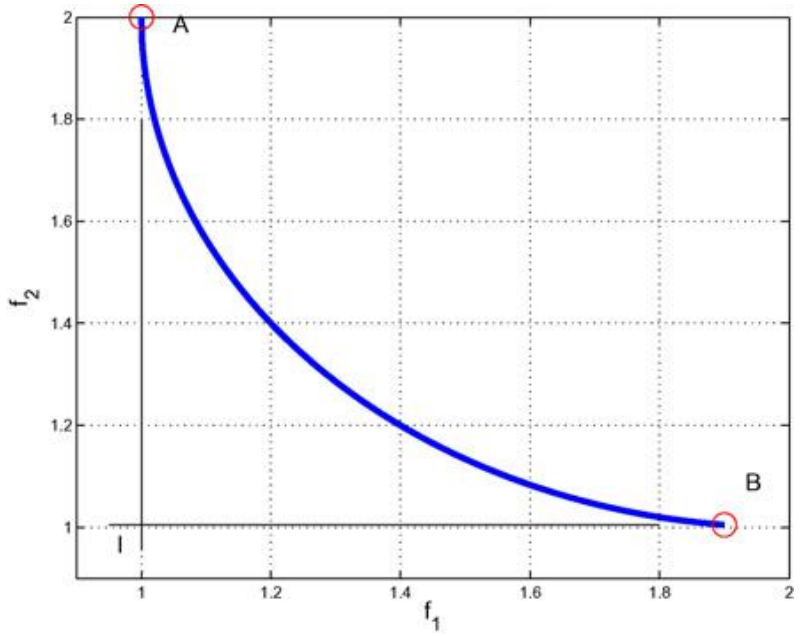


Figure 3.5 Typical Pareto front for multiobjective problems.

In this way to test if the optimization problem consisting of the two objectives *MACFLEX* and *MTMAC* is multiobjective, we have to prove that their values are conflicting.

Example: We take the following damaged state of the simple three-story structure:

$$\{d\} = \begin{Bmatrix} 0 \\ 0.025 \cdot i \\ 0 \end{Bmatrix}$$

Where i is the number of values we want to take for the criteria *MACFLEX* and *MTMAC*. In our example i is set to 20 and the diagram containing the values of *MACFLEX* and *MTMAC* for the different damage cases is the following:

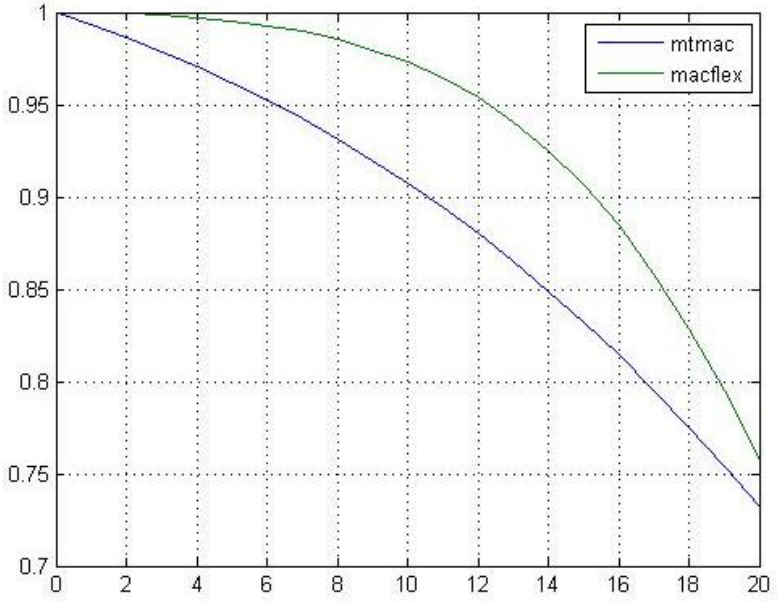


Figure 3.6 Values of MACFLEX and MTMAC for different damage cases.

As it can be seen in Figure 3.6 the values of both objectives decrease in the different damage cases. This is a clear indication that the objectives are not conflicting and the problem cannot be solved as a multiobjective one. Additionally it is a priori known that the solution of the optimization problem containing these two objectives is the single point where each one of the objective functions F_1 and F_2 have values equal to zero. So, by definition the optimization problem using the objectives *MACFLEX* and *MTMAC* is not multiobjective but single-objective.

Nevertheless the multiobjective approach is widely used due to the lack of a clear objective function in the context of damage detection problems. Most times, from multiple objectives a single-objective function is constructed using some weighting functions and the resultant optimization problem is solved in the usual way. The weights have a strong influence on the results and depend essentially on the accuracy of the measured data; more uncertain data should be affected by smaller weights. However, as the modeling errors and the uncertainty in the measured data are not usually known a priori, it is difficult to perform an optimal choice of the weight values.

Single objective function using both *MACFLEX* and *MTMAC*

In the present thesis a single objective function using both *MACFLEX* and *MTMAC* objectives was sought, instead of using the multiobjective approach. This objective function was formed by the knowledge that the single point which minimizes both of the objective functions F_1 and F_2 (Eq. (3.15) and Eq. (3.19)) is the origin of the diagram containing the values of both objectives. In this way, the objective function F_3 used in the current thesis is the radius of this diagram containing the values of the two objectives:

$$F_3 = \sqrt{(1-MACFLEX)^2 + (1-MTMAC)^2} \tag{3.20}$$

By minimizing F_3 the optimizer converges to a solution as close to the a priori known single point solution of the optimization problem as possible.

But before using the objective function F_3 in the damage identification problem, we have to acknowledge the issue with the difference in the range of values of the *MACFLEX* and *MTMAC* objectives. This is because using the radius of a diagram as an objective function requires the same range of values in the diagram axes.

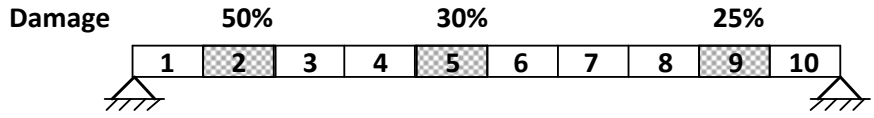
In the previous example for a random damage case of a simple three story structure the values of the objective functions F_1 and F_2 were the following:

$$F_1 = 1 - 0.9874 = 0.0126$$

$$F_2 = 1 - 0.7254 = 0.2746$$

It is clearly visible that the values of the objective functions are not in the same range although their domains of definition are identical [0:1]. So by using F_3 as the objective function for this optimization problem, the results might be misleading. In order to avoid this problem, the objective function F_1 must be multiplied with a suitable factor, so that the values of both objectives are in the same range during the optimization process.

Example: We take a simply supported beam structure with 10 elements and a random damage case, as seen in the following figure:



For this damage identification problem, an optimization algorithm is used and the values of the objectives functions F_1 and F_2 are recorded during the iterations of the algorithm. In order to demonstrate the difference in the range of the values of these two objectives, Figure 3.7 presents the ratio F_2/F_1 during the course of the algorithm iterations.

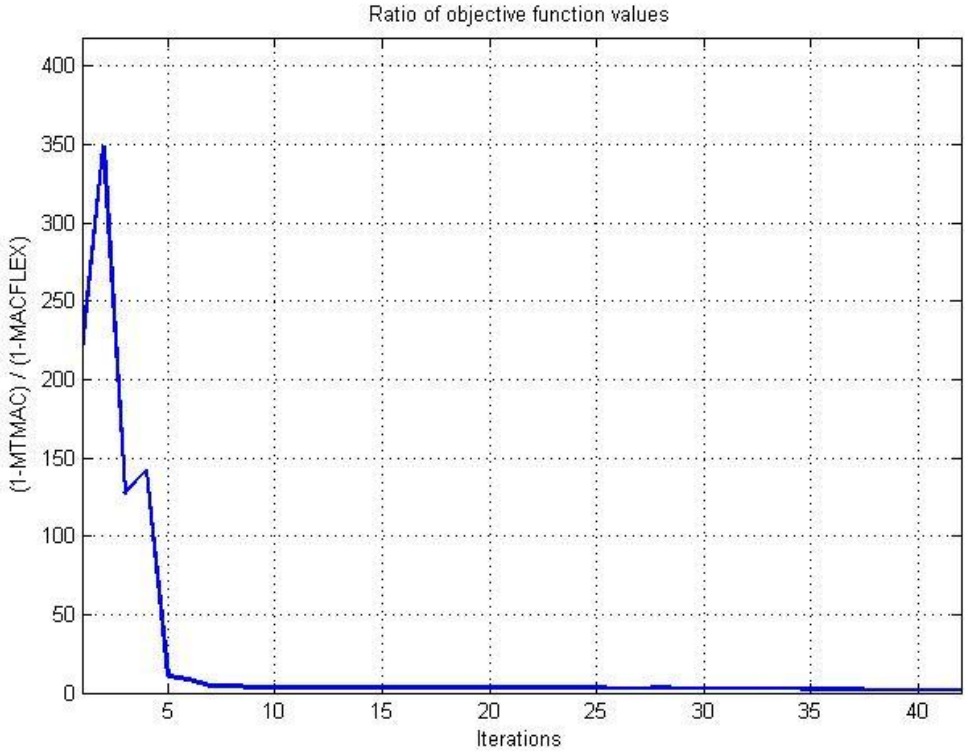


Figure 3.7 Ratio of objective functions during the algorithm iterations.

It can be observed that during the first iterations of the algorithm, the difference in the range of the objective functions values is significant. In this way using the objective function F_3 keeps the algorithm from converging towards the right solution in the first optimization runs.

To avoid this problem it is suggested that the *MACFLEX* objective function is multiplied with a factor that equals the range of its values with the *MTMAC* objective function. In our example the multiplication factor is set to 1000 and the objective function F_3 is formed as follows:

$$F_1 = 1000 \cdot (1 - MACFLEX)$$

$$F_2 = 1 - MTMAC$$

$$F_3 = \sqrt{F_1^2 + F_2^2}$$

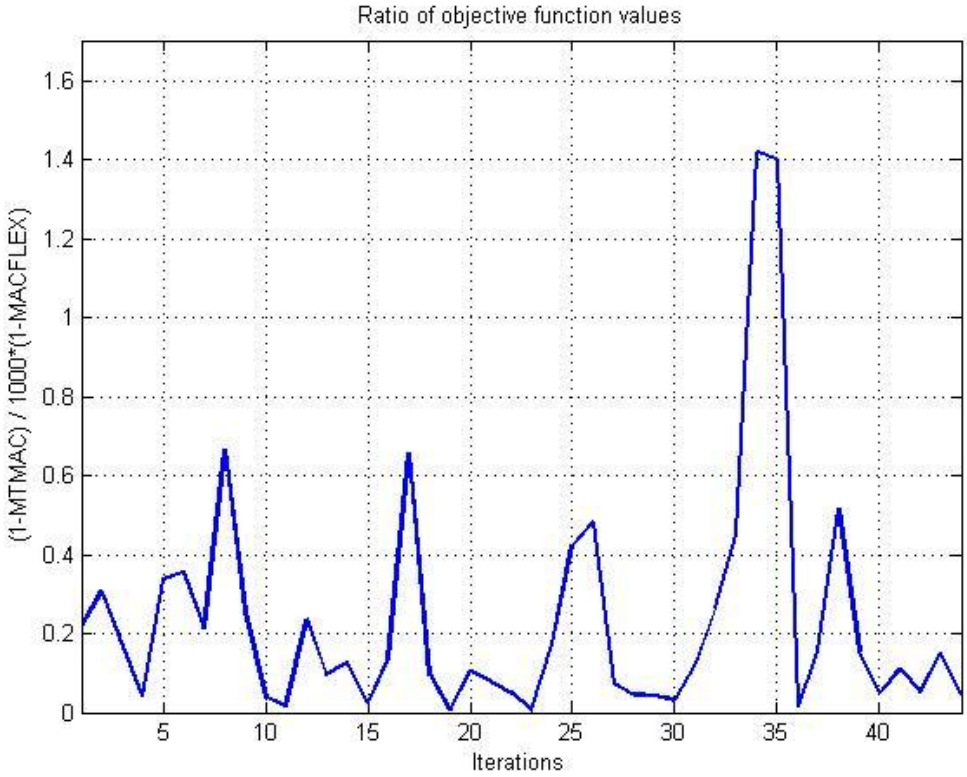


Figure 3.8 Ratio of objective functions during the algorithm iterations using a multiplication factor.

It can be observed now that with the use of a multiplication factor the values of the two criteria are generally within the same range from the first optimization runs and thus the optimization algorithm can converge to the optimal solution in a proper and quicker way. The validity of this assumption will be tested in the following paragraphs with more examples and applications.

Chapter 4

4 Applications

4.1 Simply supported beam with 10 elements

4.1.1 Input data

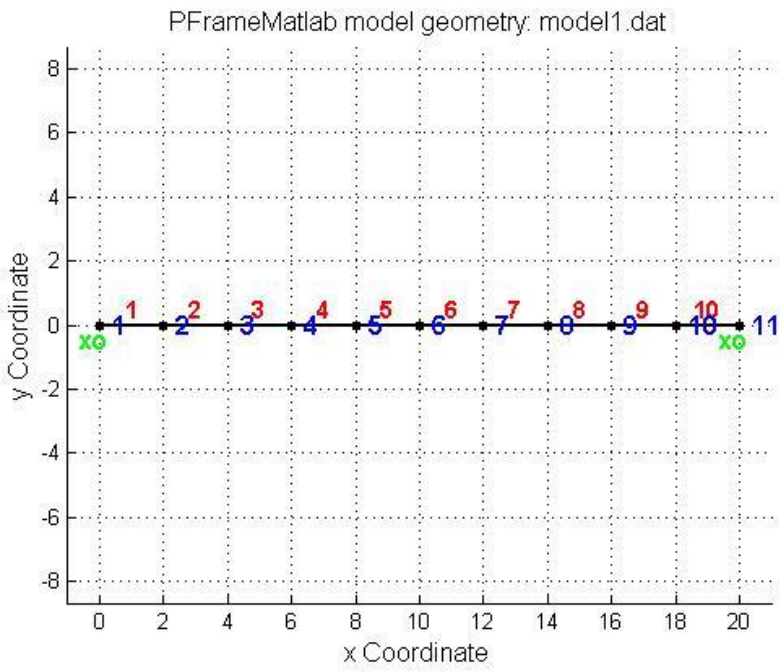


Figure 4.1 Model of simply supported beam with 10 elements.

Material properties:

$E = 2.1 \cdot 10^{8}\,\mathrm{kN/m^3}$

Section properties:

$A = 5 \cdot 10^{-3}\,\mathrm{m^2}, I_{yy} = 8 \cdot 10^{-5}\,\mathrm{m^4}$

Mass properties:

Each non-constrained node has mass:
m = 0.5 kgr.

Table 4.1 General optimization parameter values for simply supported beam.

Design variable	d (damage index)
Number of design variables	10
Lower bound (lb)	0
Upper bound (ub)	0.99

Table 4.2 PSO parameter values for simply supported beam.

Number of runs	5
Swarm size	20
Iterations	500
v_{max}	0.1
w_{max}	0.95
w_{min}	0.8
c_1, c_2	2

Table 4.3 SQP parameter values for simply supported beam.

x0	(lb + ub) / 2
MaxFunEvals	5000
MaxIter	500
TolFun	10^{-10}
TolX	10^{-10}

The damage cases that were tested in this example were the following:

1) Single damaged element
2) Three damaged elements
3) Uniform damage

Objective functions:

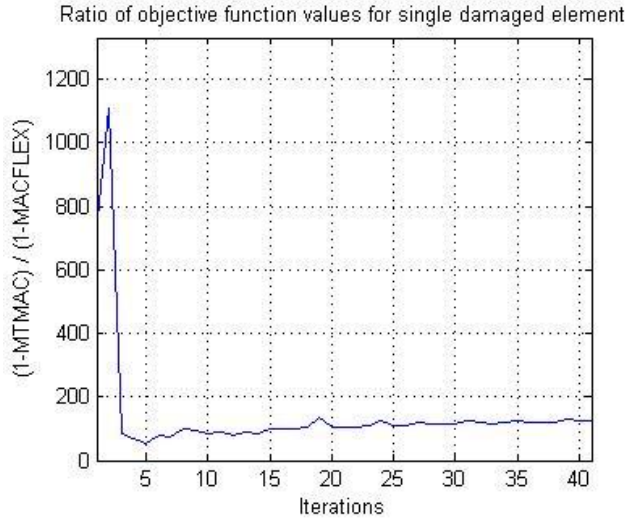
$$F_1 = 1 - MACFLEX$$

$$F_2 = 1 - MTMAC$$

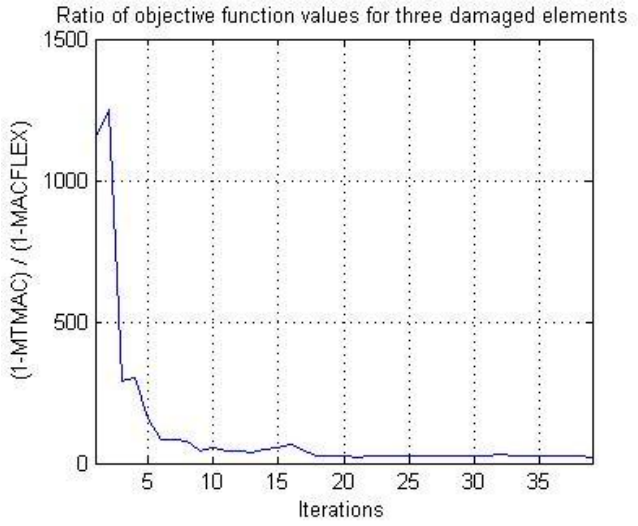
$$F_3 = \sqrt{\left(10000 \cdot \left(1 - MACFLEX\right)\right)^2 + \left(1 - MTMAC\right)^2}$$

The multiplication factor 10000 was chosen after some test runs of the optimization algorithms and the calculation of the ratio of the objective functions F_1 and F_2 for the different damage cases. In these test runs F_3 was used as the objective function without a multiplication factor and the ratio for the three damage cases can be seen in the following figures:

1)



2)



3)

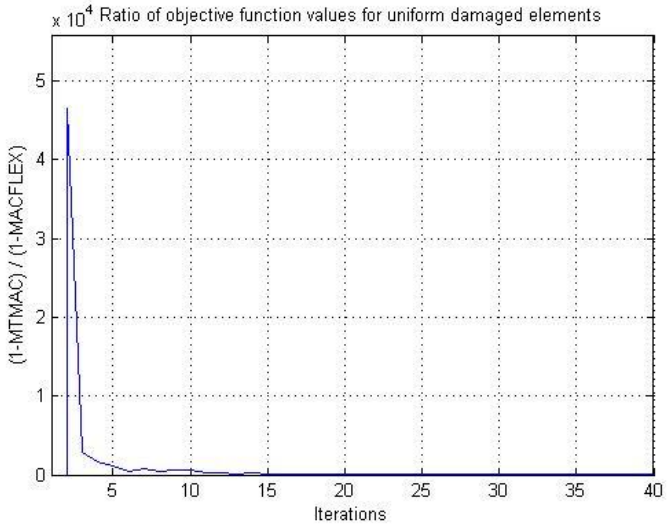


Figure 4.2 Ratio of objective functions F_1,F_2 for the different damage cases of the beam.

The first 3 eigenmodes are used to calculate the values of *MACFLEX* and *MTMAC* criteria.

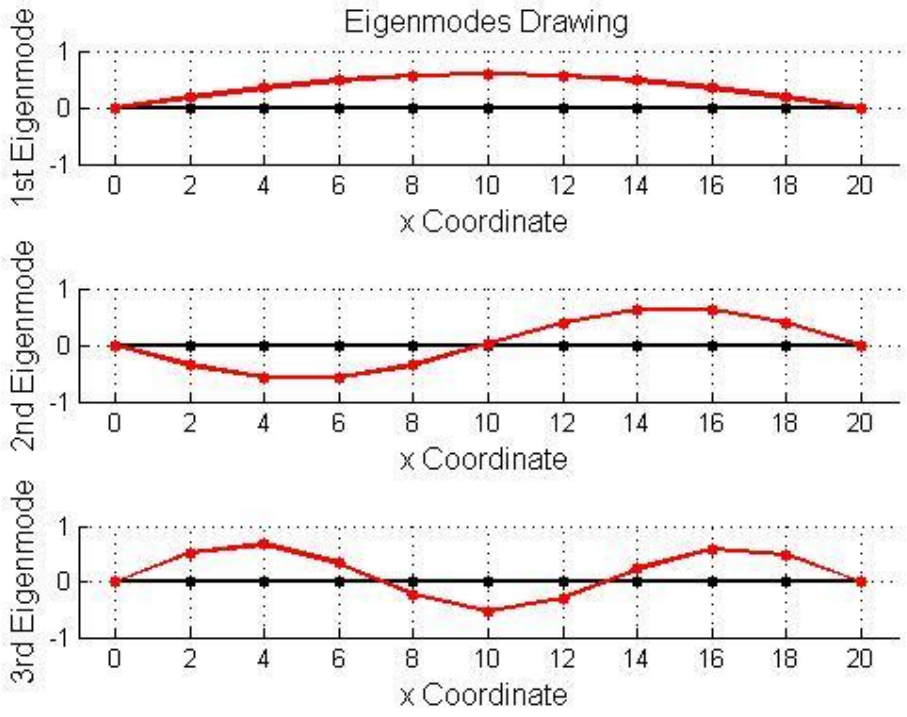


Figure 4.3 First three eigenmodes for simply supported beam with 10 elements in the undamaged state.

In optimization with PSO the algorithm runs 5 times and the best result is obtained by the run with the minimum objective function value. In the mathematical optimization, however the result is always the same, so there is no need for extra runs.

4.1.2 Single damaged element

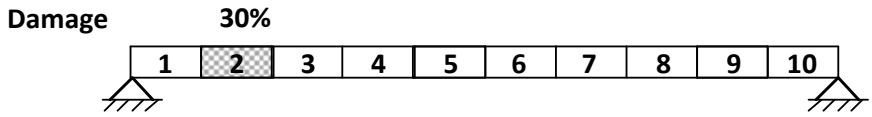


Figure 4.4 Single element damage case for simply supported beam with 10 elements.

a) No noise

i) Optimization only with *MACFLEX* criterion: $F_1 = 1 - MACFLEX$

Table 4.4 Optimization results for single element damaged beam using objective function F_1.

	Element	1	2	3	4	5	6	7	8	9	10
PSO	**Damage (%)**	48.25	56.97	47.83	34.70	47.29	39.47	41.55	46.01	34.86	48.28
SQP	**Damage (%)**	48.01	63.55	47.99	47.91	48.00	47.90	48.01	47.89	48.02	47.84

Minimum objective function value	
PSO	$1.20 \cdot 10^{-6}$
SQP	$8.68 \cdot 10^{-12}$

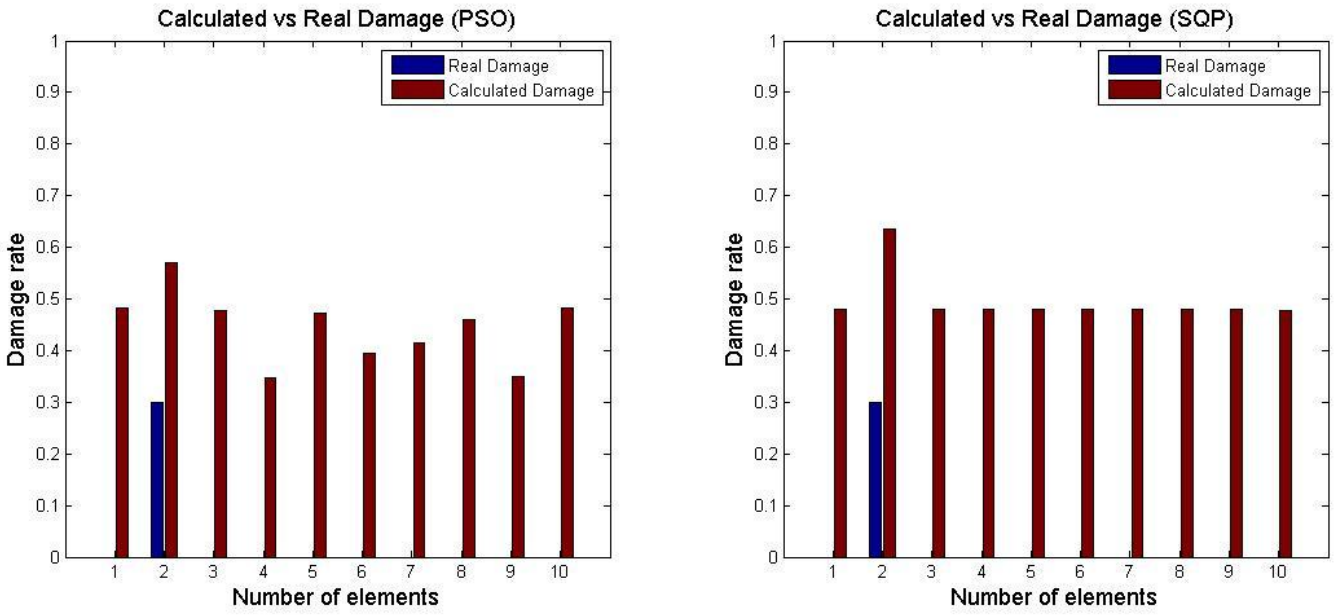


Figure 4.5 Damage results for single element damaged beam using objective function F_1.

It can be observed that using the objective function with *MACFLEX* gives us a clear indication of the position of the damaged element in the beam, although there is not a valid damage rate for the specific element. In this case, the beam element with damage is clearly the second one.

ii) Optimization only with *MTMAC* criterion: $F_2 = 1 - MTMAC$

Table 4.5 Optimization results for single element damaged beam using objective function F_2.

	Element	1	2	3	4	5	6	7	8	9	10
PSO	**Damage (%)**	0.00	30.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
SQP	**Damage (%)**	0.00	30.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Minimum objective function value	
PSO	$4.57 \cdot 10^{-8}$
SQP	$2.25 \cdot 10^{-7}$

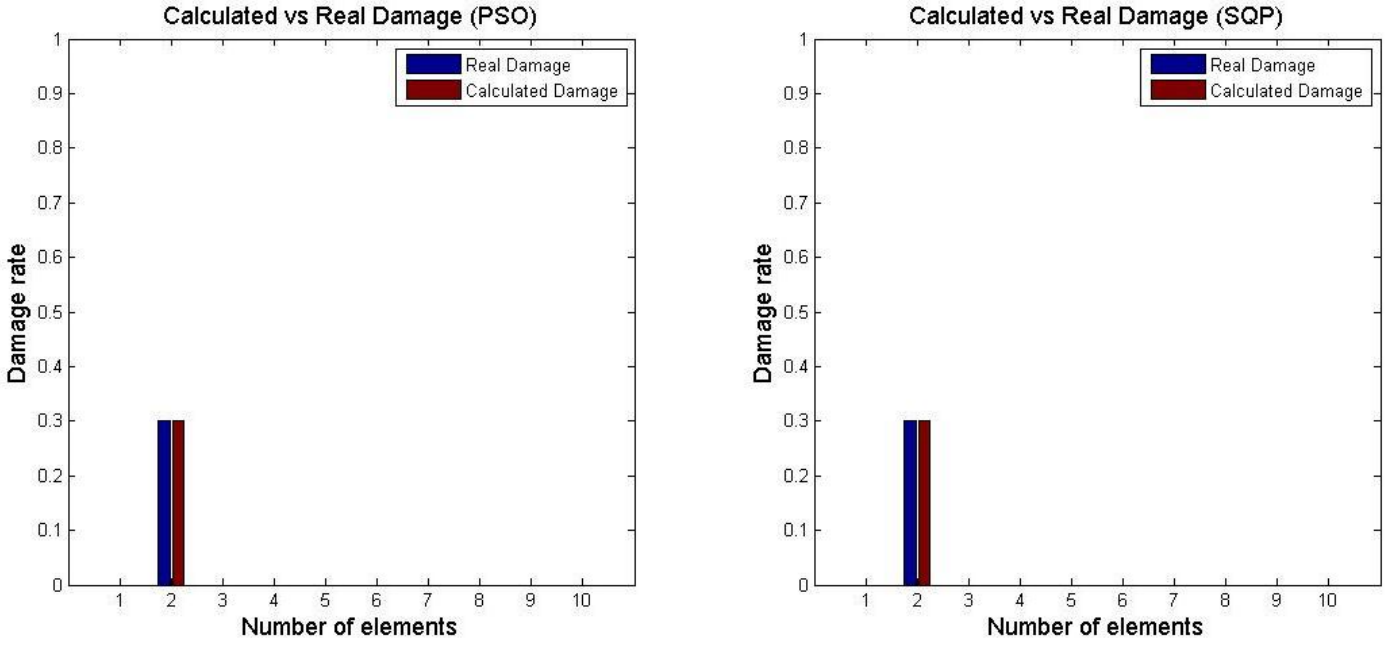


Figure 4.6 Damage results for single element damaged beam using objective function F_2.

The *MTMAC* objective function gives a more clear indication about both the position and the rate of the damage in the elements of the beam. In this case *MTMAC* leads to a complete match between the real and the calculated damage. However, in more complicated damage cases the results are not so accurate, as it will be proved in the next examples.

iii) Optimization with both *MACFLEX* and *MTMAC* criteria:

$$F_3 = \sqrt{\left(10000 \cdot \left(1 - MACFLEX\right)\right)^2 + \left(1 - MTMAC\right)^2}$$

Table 4.6 Optimization results for single element damaged beam using objective function F_3.

	Element	1	2	3	4	5	6	7	8	9	10
PSO	**Damage (%)**	0.00	30.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
SQP	**Damage (%)**	0.00	30.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Minimum objective function value	
PSO	$6.81 \cdot 10^{-8}$
SQP	$6.53 \cdot 10^{-9}$

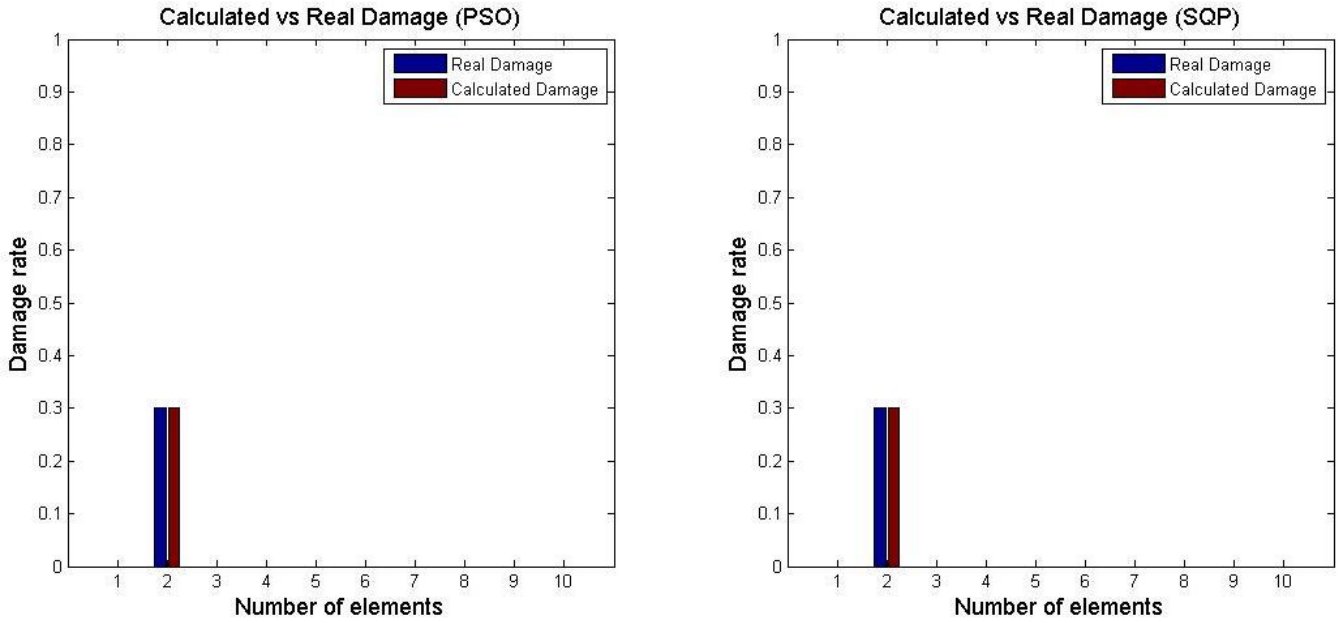


Figure 4.7 Damage results for single element damaged beam using objective function F_3.

As before the single objective function using both *MACFLEX* and *MTMAC* criteria leads to a complete match between the real and the calculated damage of the beam elements.

b) Calculated data with noise 5% and 10%

In order to simulate the calculation data to the experimental one, so that our application is more compatible to a real experiment, Eq. (3.4) is used to apply noise to the mode shapes of the beam. The value of *NoiseRatio* is set to 5% and 10% and the optimization algorithm used in this case is SQP, as it provided better results in minimizing the objective function F_3 of this example.

Table 4.7 Optimization results for single element damaged beam with noise using objective function F_3.

Noise	Element	1	2	3	4	5	6	7	8	9	10
5%	**Damage (%)**	0.00	29.98	0.00	0.00	0.00	0.00	0.00	0.00	0.02	0.00
10%	**Damage (%)**	0.00	29.88	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.64

Noise	Minimum objective function value
5%	$1.76 \cdot 10^{-3}$
10%	$6.63 \cdot 10^{-3}$

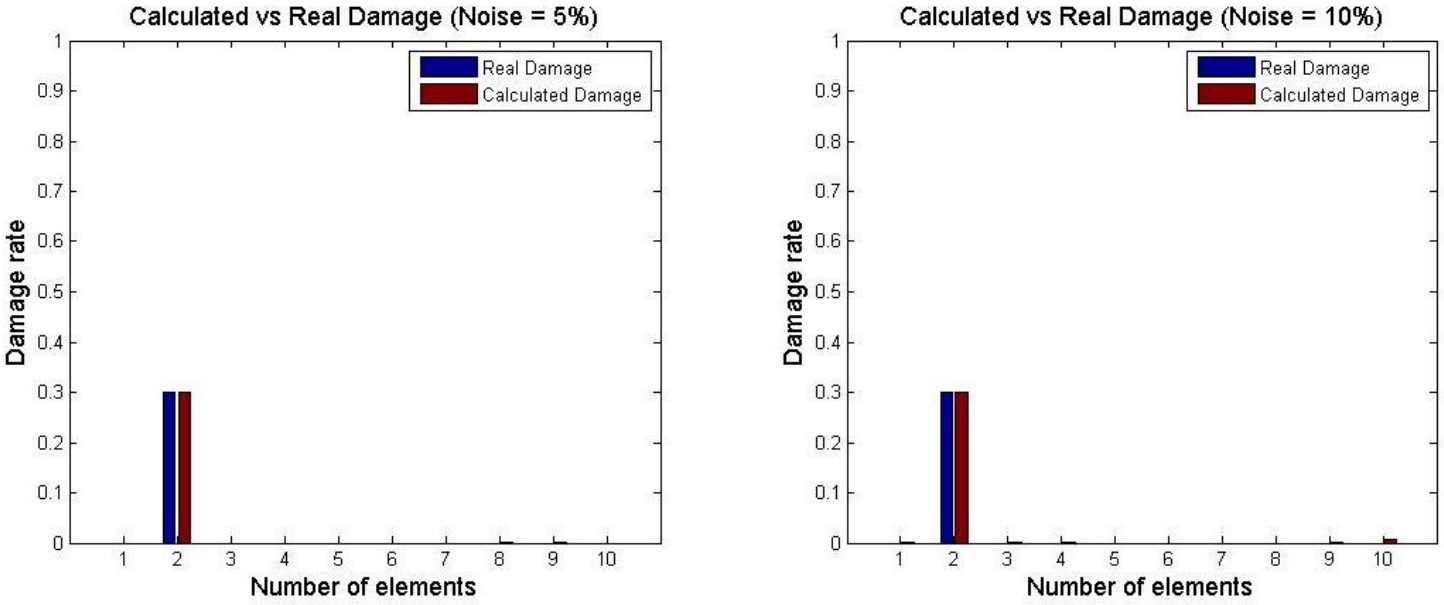


Figure 4.8 Damage results for single element damaged beam with noise using objective function F_3.

It can be seen in Figure 4.8 that even with noise 10% the algorithm converges to the right solution by minimizing the objective function using both *MACFLEX* and *MTMAC*.

4.1.3 Three damaged elements

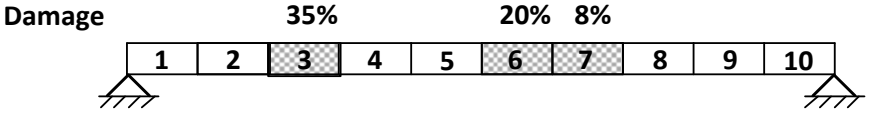


Figure 4.9 Three elements damage case for simply supported beam with 10 elements.

a) No noise

i) Optimization only with *MACFLEX* criterion: $F_1 = 1 - MACFLEX$

Table 4.8 Optimization results for three element damaged beam using objective function F_1.

	Element	1	2	3	4	5	6	7	8	9	10
PSO	**Damage (%)**	34.02	36.40	56.90	38.58	30.16	50.55	38.67	35.15	37.09	25.87
SQP	**Damage (%)**	45.44	46.29	64.71	46.40	45.44	57.05	50.03	46.09	45.49	48.25

Minimum objective function value	
PSO	$2.67 \cdot 10^{-7}$
SQP	$4.64 \cdot 10^{-9}$

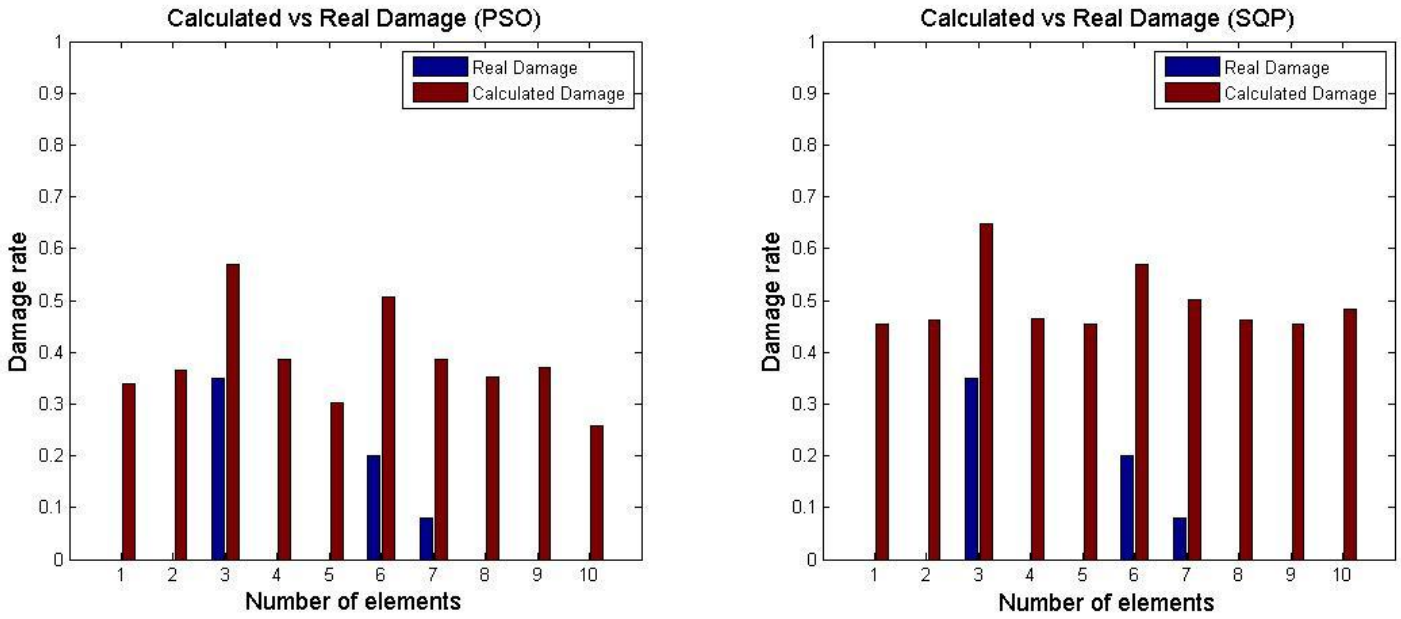


Figure 4.10 Damage results for three element damaged beam using objective function F_1.

Again the objective function with *MACFLEX* gives us a clear indication of the position of the damaged elements in the beam and also in this case of the extent of the damage. In this way it can be observed that the third element is more damaged than the sixth element and the seventh one.

ii) Optimization only with *MTMAC* criterion: $F_2 = 1 - MTMAC$

Table 4.9 Optimization results for three element damaged beam using objective function F_2.

	Element	1	2	3	4	5	6	7	8	9	10
PSO	**Damage (%)**	0.00	0.00	34.15	0.00	5.75	14.97	10.48	0.00	0.00	2.52
SQP	**Damage (%)**	0.00	1.21	33.96	1.64	0.00	19.38	8.54	0.00	0.00	1.47

Minimum objective function value	
PSO	$5.18 \cdot 10^{-4}$
SQP	$9.13 \cdot 10^{-6}$

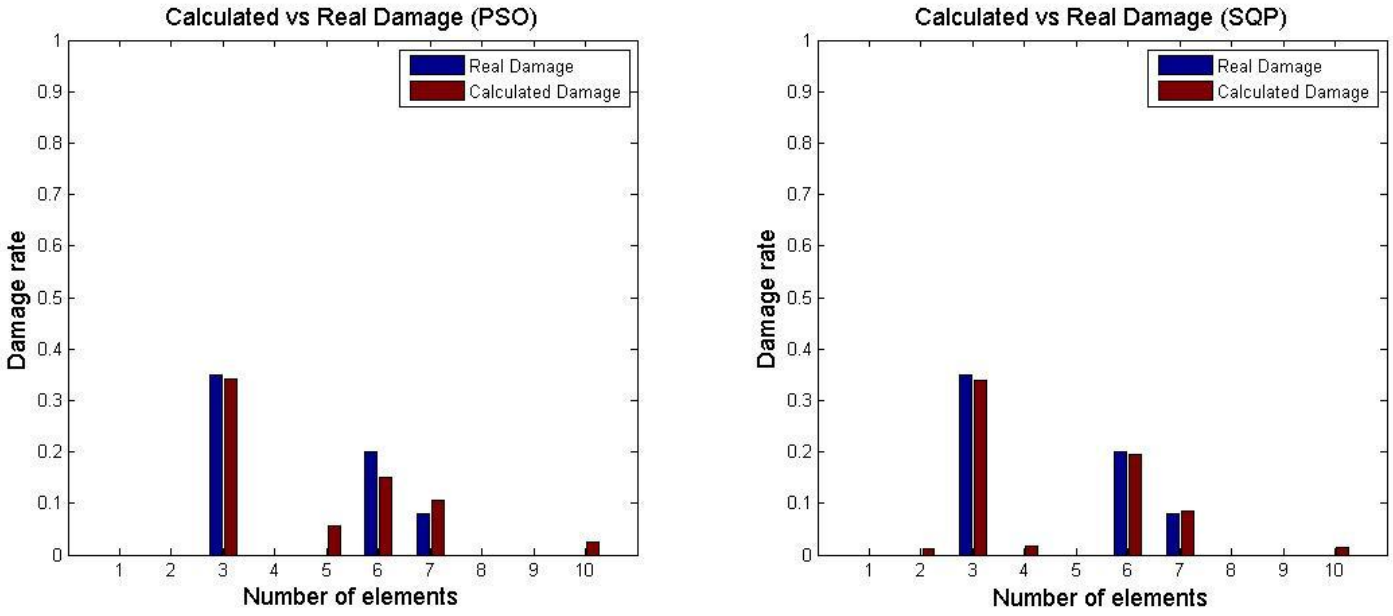


Figure 4.11 Damage results for three element damaged beam using objective function F_2.

In this example the objective function using *MTMAC* provides some good results of the position and the extent of damage in the beam elements, although there some errors between the real and the calculated damage.

iii) Optimization with both *MACFLEX* and *MTMAC* criteria:

$$F_3 = \sqrt{\left(10000 \cdot \left(1 - MACFLEX\right)\right)^2 + \left(1 - MTMAC\right)^2}$$

Table 4.10 Optimization results for three element damaged beam using objective function F_3.

	Element	1	2	3	4	5	6	7	8	9	10
PSO	**Damage (%)**	0.00	0.00	35.01	0.00	0.00	19.92	7.87	0.00	0.16	0.00
SQP	**Damage (%)**	0.00	0.00	35.00	0.00	0.00	19.99	8.00	0.00	0.00	0.00

Minimum objective function value	
PSO	$3.07 \cdot 10^{-4}$
SQP	$4.08 \cdot 10^{-8}$

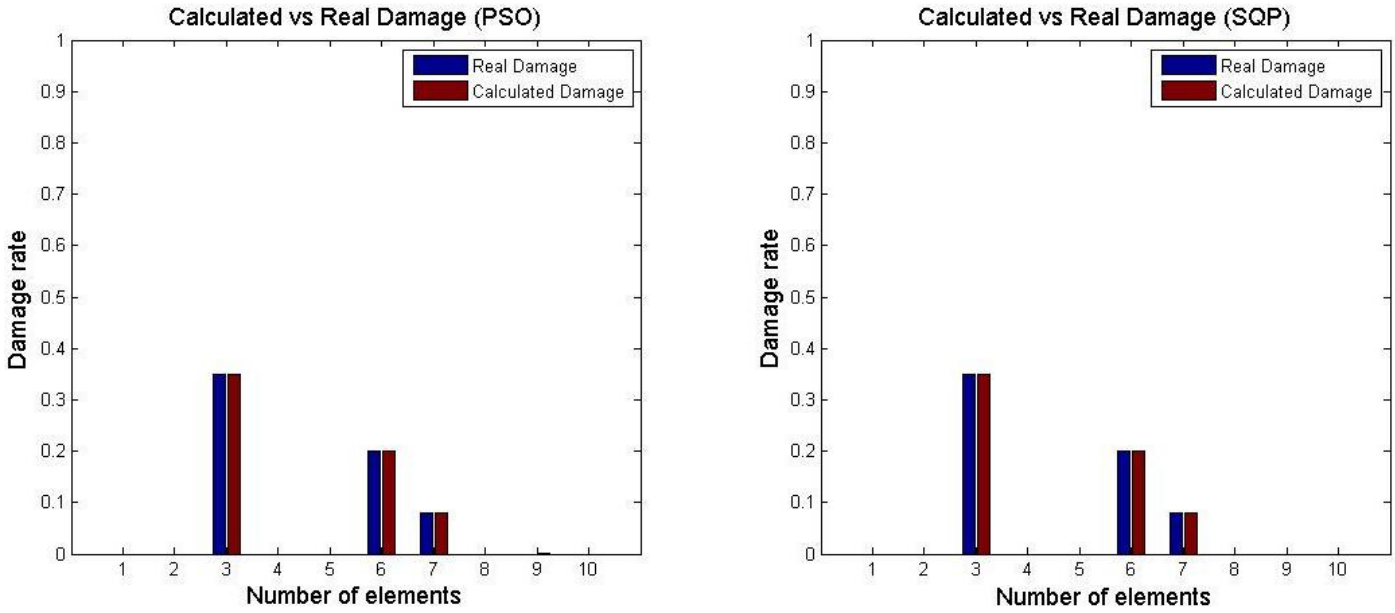


Figure 4.12 Damage results for three element damaged beam using objective function F_3.

In this case the single objective function using both *MACFLEX* and *MTMAC* criteria leads again to a complete match between the real and the calculated damage of the beam elements.

b) Calculated data with noise 5% and 10%

Again we simulate the calculated data by setting the value of *NoiseRatio* in Eq. (3.4) to 5% and 10% and the optimization algorithm used is SQP, as it provided better results in minimizing the objective function F_3 of this example, as well.

Table 4.11 Optimization results for three element damaged beam with noise using objective function F_3.

Noise	Element	1	2	3	4	5	6	7	8	9	10
5%	**Damage (%)**	0.00	0.00	34.91	0.26	0.00	19.92	8.01	0.00	0.00	0.50
10%	**Damage (%)**	0.42	0.27	34.74	0.00	1.36	18.90	8.58	0.00	0.00	0.00

Noise	Minimum objective function value
5%	$2.32 \cdot 10^{-3}$
10%	$1.14 \cdot 10^{-2}$

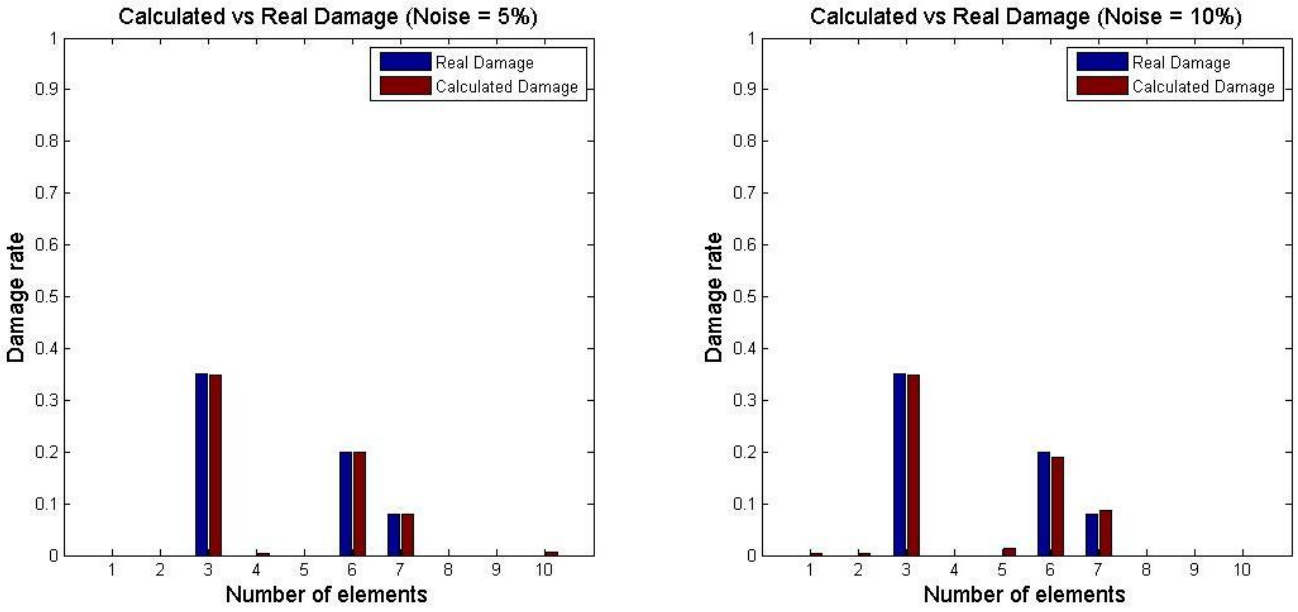


Figure 4.13 Damage results for three element damaged beam with noise using objective function F_3.

It can be seen in Figure 4.13 that even though the application of noise in the calculated data influences the value of the objective function, the comparison of the calculated and real damage is still pretty accurate.

4.1.4 Uniform damage

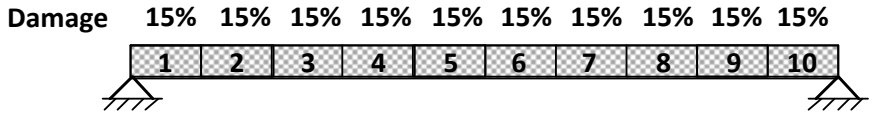


Figure 4.14 Uniform damage case for simply supported beam with 10 elements.

a) No noise

i) Optimization only with *MACFLEX* criterion: $F_1 = 1 - MACFLEX$

Table 4.12 Optimization results for uniform damaged beam using objective function F_1.

	Element	1	2	3	4	5	6	7	8	9	10
PSO	**Damage (%)**	38.58	34.58	37.99	31.81	40.95	28.87	41.87	30.84	37.13	35.61
SQP	**Damage (%)**	35.85	35.32	35.72	35.30	35.80	35.18	35.96	34.92	36.38	33.69

Minimum objective function value	
PSO	$5.26 \cdot 10^{-7}$
SQP	$1.82 \cdot 10^{-10}$

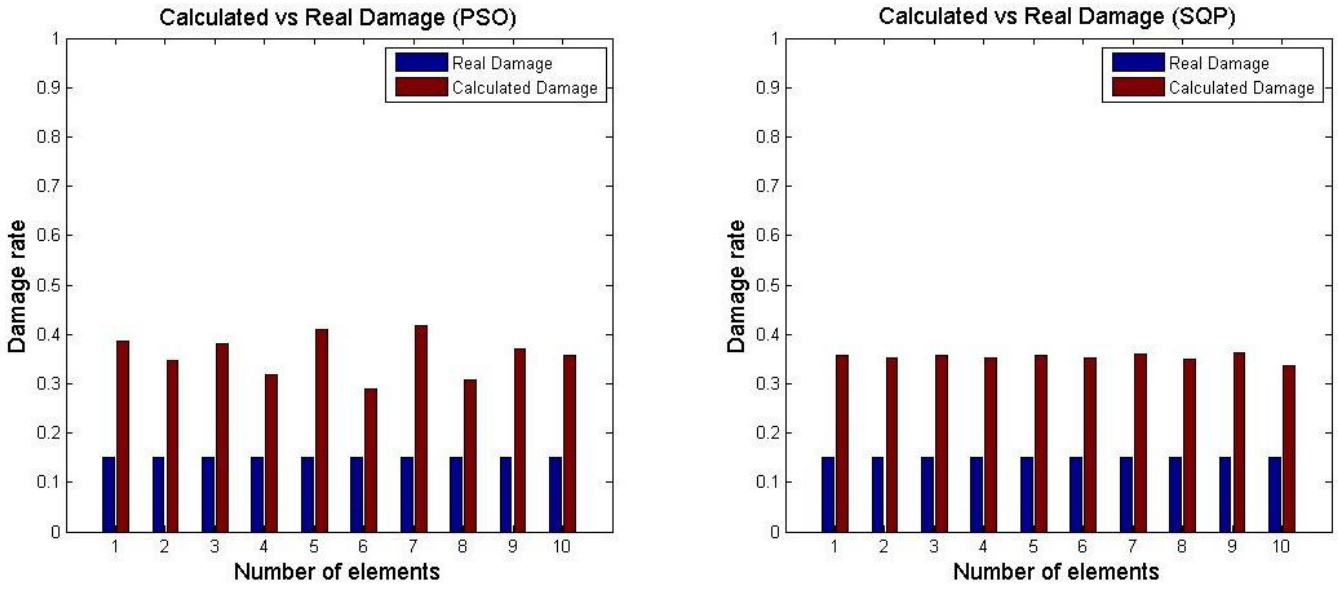


Figure 4.15 Damage results for uniform damaged beam using objective function F_1.

In the uniform damage case, the objective function using *MACFLEX* indicates that there is a uniform pattern in damage, although it fails to identify its value.

ii) Optimization only with *MTMAC* criterion: $F_2 = 1 - MTMAC$

Table 4.13 Optimization results for uniform damaged beam using objective function F_2.

	Element	1	2	3	4	5	6	7	8	9	10
PSO	**Damage (%)**	29.40	9.61	16.39	16.15	19.91	5.34	22.18	3.32	19.75	21.24
SQP	**Damage (%)**	25.41	9.62	17.54	13.72	15.40	15.44	13.47	17.76	9.53	25.65

Minimum objective function value	
PSO	$8.03 \cdot 10^{-4}$
SQP	$1.82 \cdot 10^{-5}$

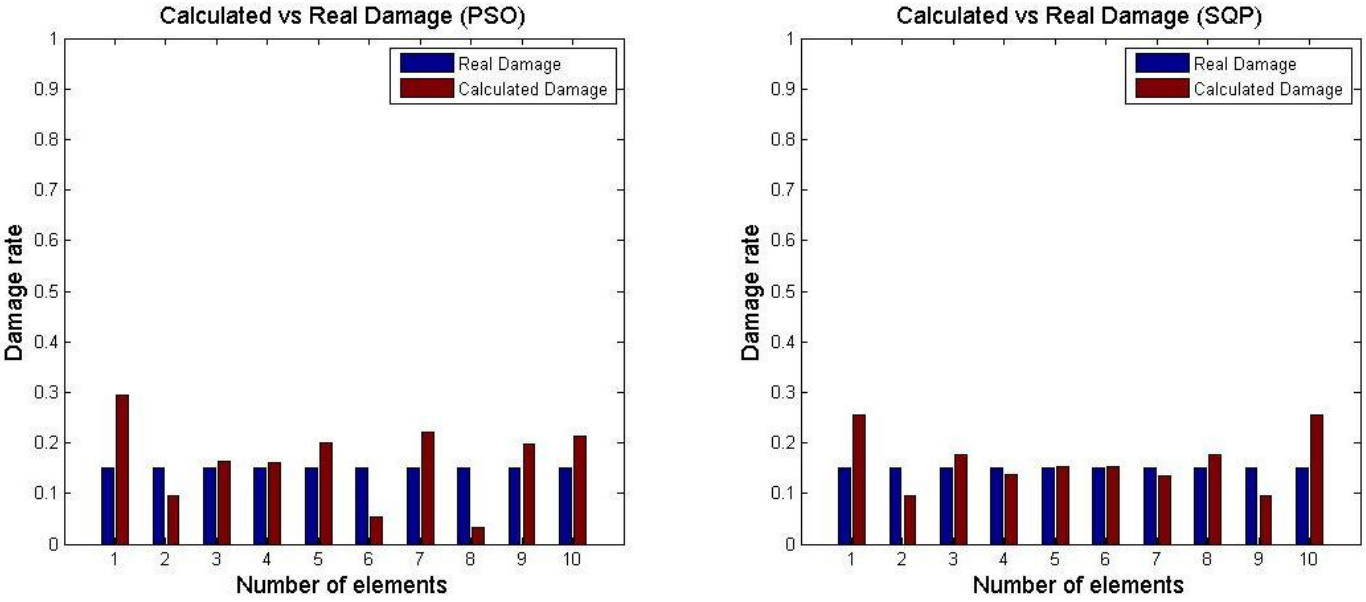


Figure 4.16 Damage results for uniform damaged beam using objective function F_2.

In the uniform damage case the objective function using *MTMAC* provides some indication of the value and position of the damage in the beam elements. Its accuracy, however is not great in this damage case.

iii) Optimization with both *MACFLEX* and *MTMAC* criteria:

$$F_3 = \sqrt{\left(10000 \cdot \left(1 - MACFLEX\right)\right)^2 + \left(1 - MTMAC\right)^2}$$

Table 4.14 Optimization results for uniform damaged beam using objective function F_2.

	Element	1	2	3	4	5	6	7	8	9	10
PSO	**Damage (%)**	18.71	12.71	16.66	13.85	14.71	18.87	6.04	22.22	9.71	17.40
SQP	**Damage (%)**	15.37	14.78	15.21	14.76	15.28	14.66	15.44	14.38	15.96	12.83

Minimum objective function value	
PSO	$6.25 \cdot 10^{-3}$
SQP	$7.14 \cdot 10^{-5}$

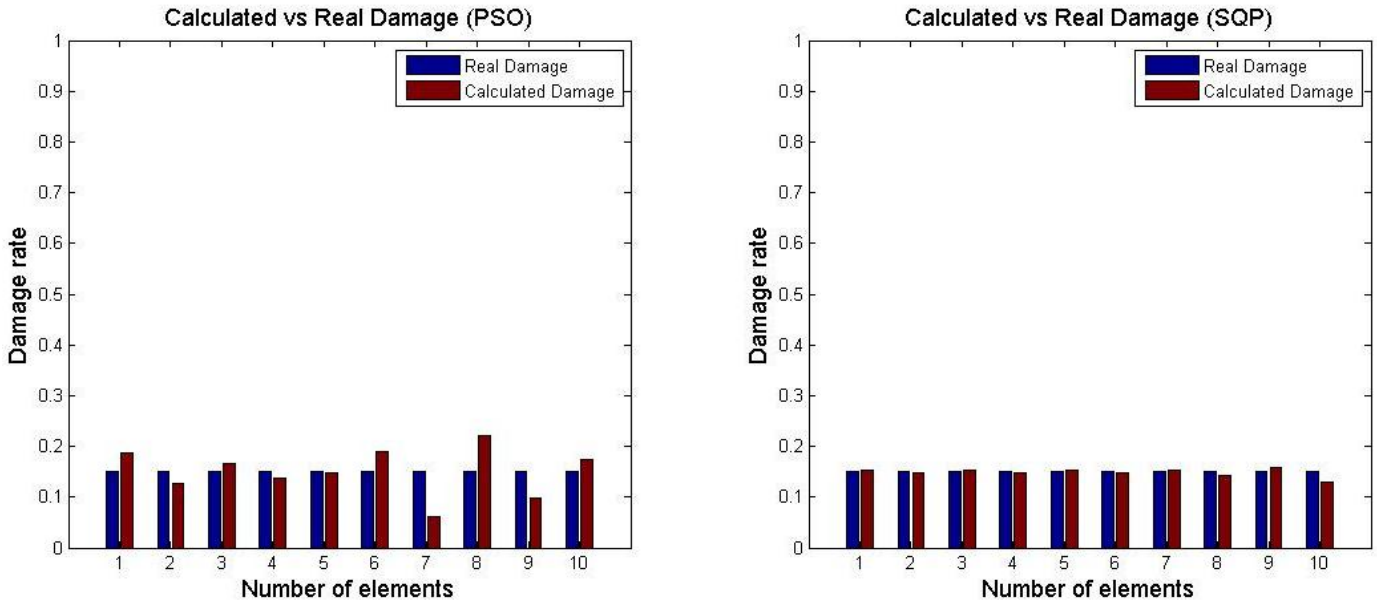


Figure 4.17 Damage results for uniform damaged beam using objective function F_3.

In the uniform damage case the single objective function using both *MACFLEX* and *MTMAC* criteria identifies again the real damage of the beam elements. More specifically, the match is perfect in the case of the SQP mathematical optimization algorithm.

Observation: After this set of examples it can be said with accuracy that the mathematical optimization algorithm SQP is clearly better than PSO, dealing with these types of damage identification problems. So, in the next applications of damage identification only the mathematical optimization algorithm SQP will be used.

b) Calculated data with noise 5% and 10%

Table 4.15 Optimization results for uniform damaged beam with noise using objective function F_3.

Noise	Element	1	2	3	4	5	6	7	8	9	10
5%	**Damage (%)**	14.97	15.06	15.02	14.77	15.39	14.85	14.21	17.52	8.54	28.20
10%	**Damage (%)**	12.39	16.48	13.50	16.17	14.81	14.12	16.17	15.01	10.32	29.82

Noise	Minimum objective function value
5%	$1.98 \cdot 10^{-3}$
10%	$9.40 \cdot 10^{-3}$

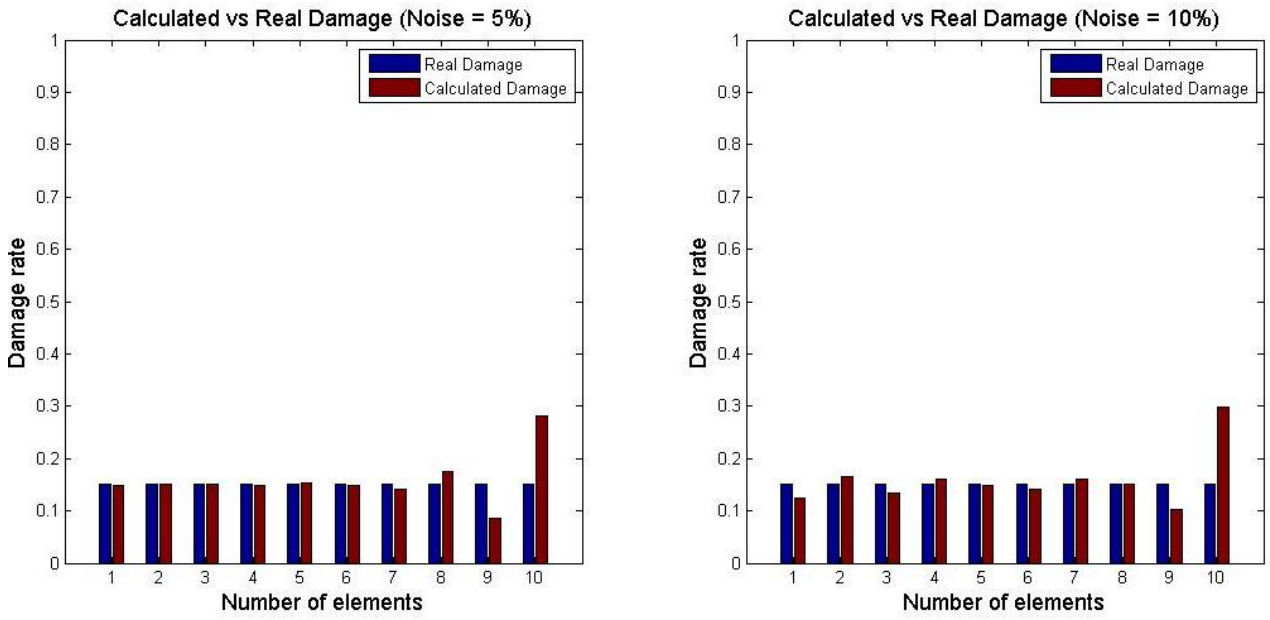


Figure 4.18 Damage results for uniform element damaged beam with noise using objective function F_3.

It can be seen in Figure 4.18 that now the noise partly influences the results of the optimization using F_3 objective function. However, this is a pretty complicated damage case and the match of the calculated and real damage remains satisfactory with noise application.

4.2 Fixed frame with 20 elements

4.2.1 Input data

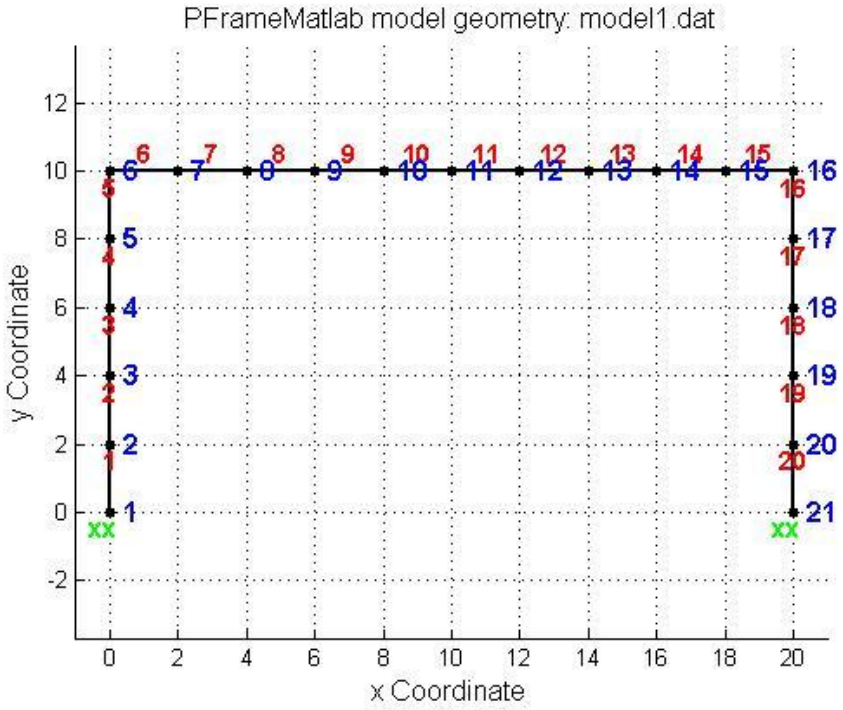


Figure 4.19 Model of fixed frame with 20 elements.

Material properties:

$E = 2.1 \cdot 10^{8} \text{kN/m}^3$

Section properties:

$A = 5 \cdot 10^{-3} \text{m}^2, I_{yy} = 8 \cdot 10^{-5} \text{m}^4$

Mass properties:
Each non-constrained node has mass:
m = 0.5 kgr.

Table 4.16 General optimization parameter values for fixed frame.

Design variable	*d* (damage index)
Number of design variables	20
Lower bound (lb)	0
Upper bound (ub)	0.99

Table 4.17 SQP parameter values for fixed frame.

x0	(lb + ub) / 2
MaxFunEvals	10000
MaxIter	1000
TolFun	10^{-10}
TolX	10^{-10}

Objective functions:

$$F_1 = 1 - MACFLEX$$

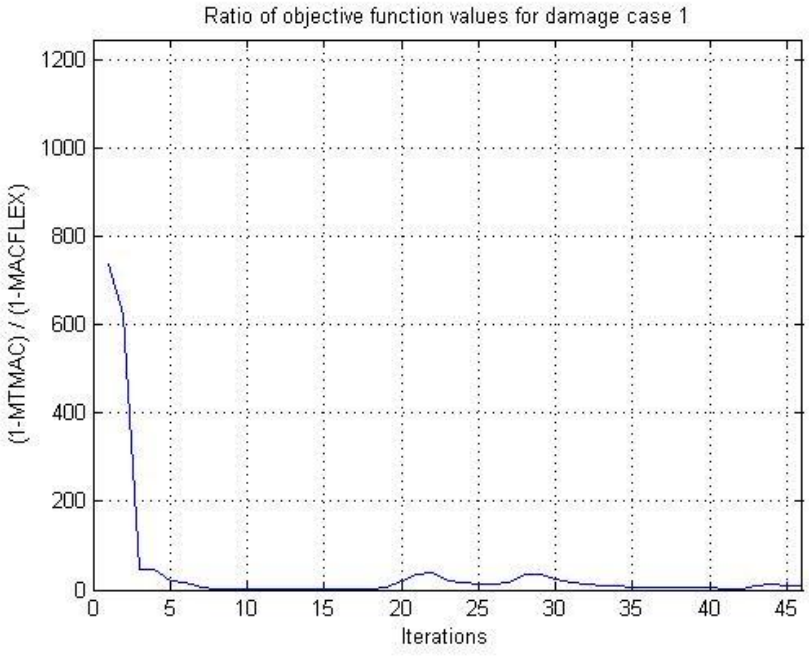
$$F_2 = 1 - MTMAC$$

Damage case 1: $F_3 = \sqrt{\left(250 \cdot \left(1 - MACFLEX\right)\right)^2 + \left(1 - MTMAC\right)^2}$

Damage case 2: $F_3 = \sqrt{\left(415 \cdot \left(1 - MACFLEX\right)\right)^2 + \left(1 - MTMAC\right)^2}$

The multiplication factors 250 and 415 were chosen after some test runs of the optimization algorithms and the calculation of the ratio of the objective functions F_1 and F_2 for the different damage cases. . In these test runs F_3 was used as the objective function without a multiplication factor and the ratio for the two damage cases described in the next paragraph, can be seen in the following figures:

1) :



2) :

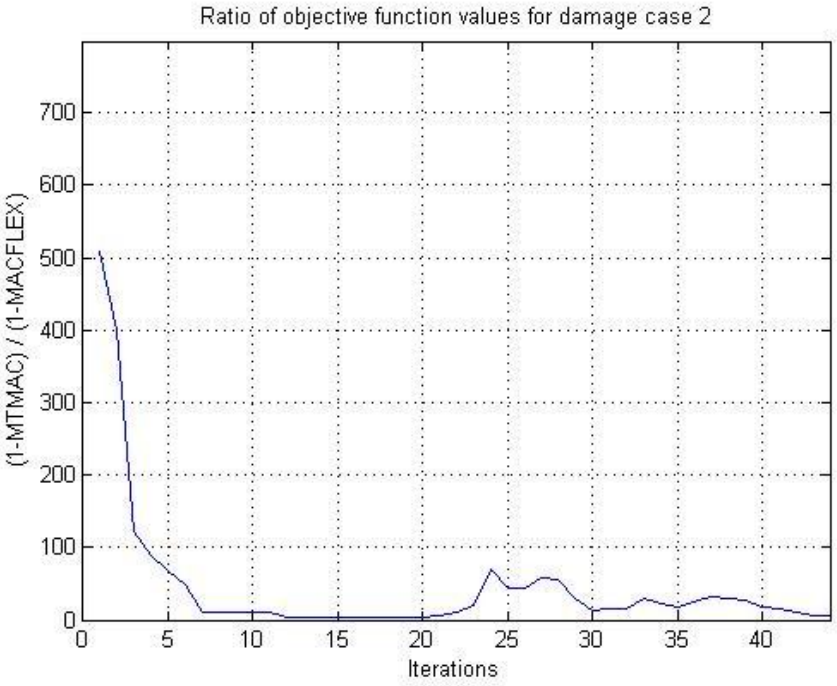
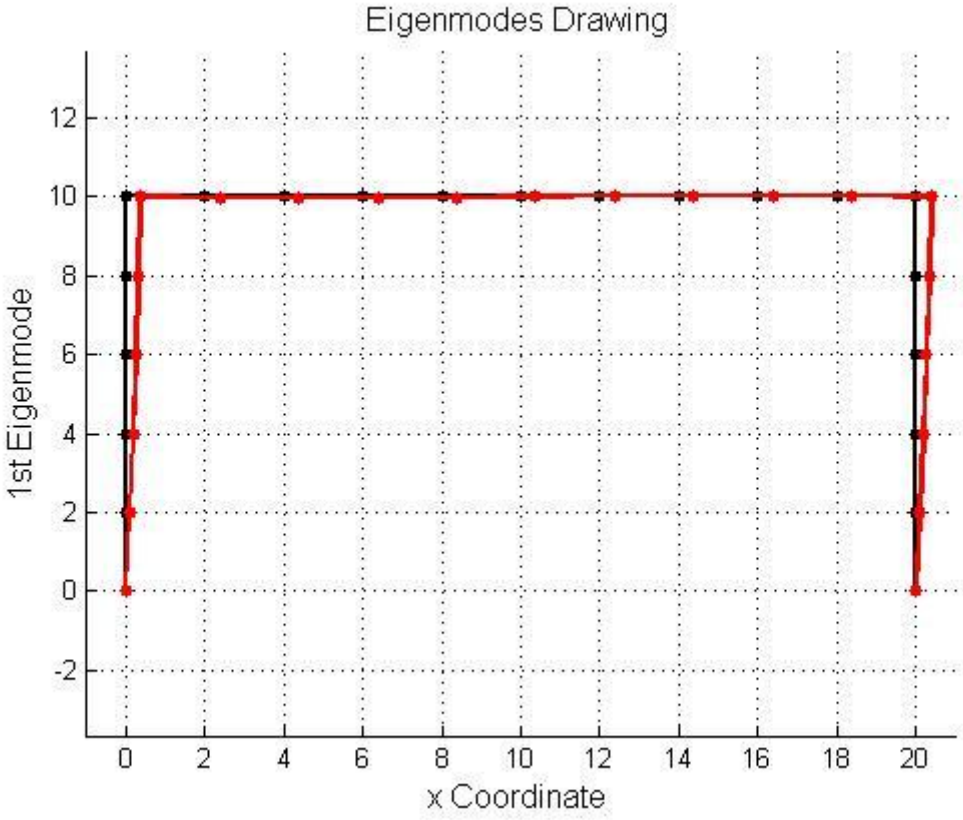


Figure 4.20 Ratio of objective functions F_1,F_2 for different damage cases of the frame.

The first 3 eigenmodes are used to calculate the values of *MACFLEX* and *MTMAC* criteria.



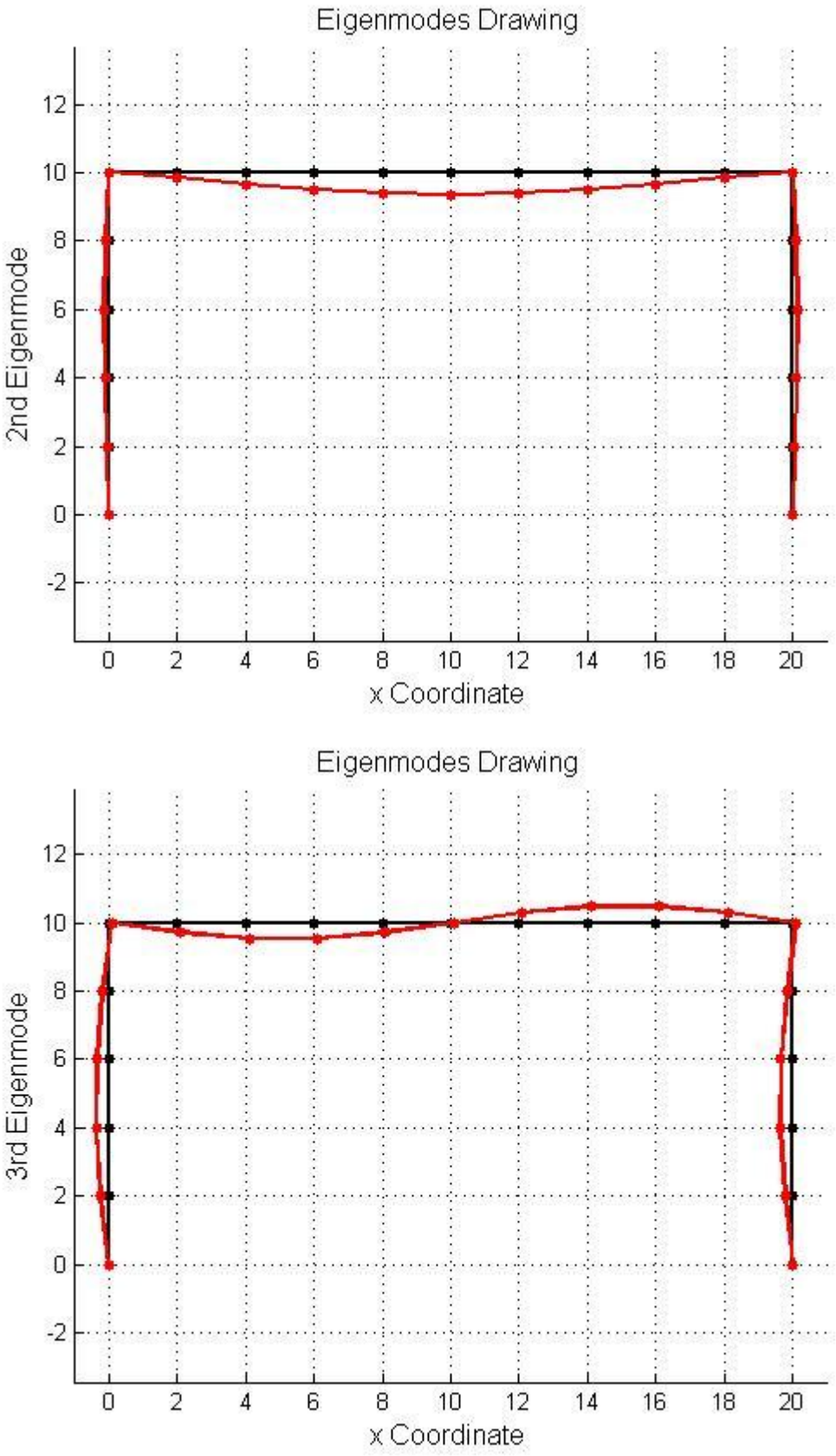


Figure 4.21 First three eigenmodes of fixed frame with 20 elements.

4.2.2 Damage case 1

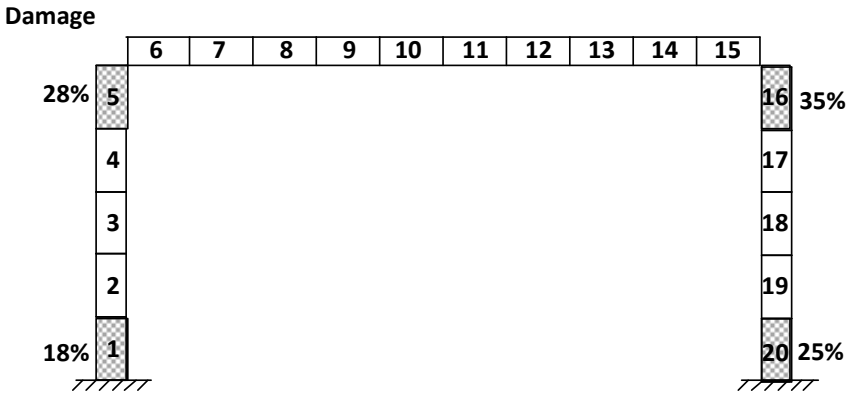


Figure 4.22 Damage case 1 of fixed frame with 20 elements.

a) No noise

ii) Optimization only with *MACFLEX* criterion: $F_1 = 1 - MACFLEX$

Table 4.18 Optimization results for damage case 1 of fixed frame using objective function F_1.

Element	1	2	3	4	5	6	7	8	9	10
Damage (%)	55.67	45.94	45.94	45.94	61.08	47.22	45.92	47.62	45.21	50.04

Element	11	12	13	14	15	16	17	18	19	20
Damage (%)	50.15	44.05	48.22	45.03	47.78	64.09	50.47	49.61	49.56	49.33

Minimum objective function value
$4.30 \cdot 10^{-10}$

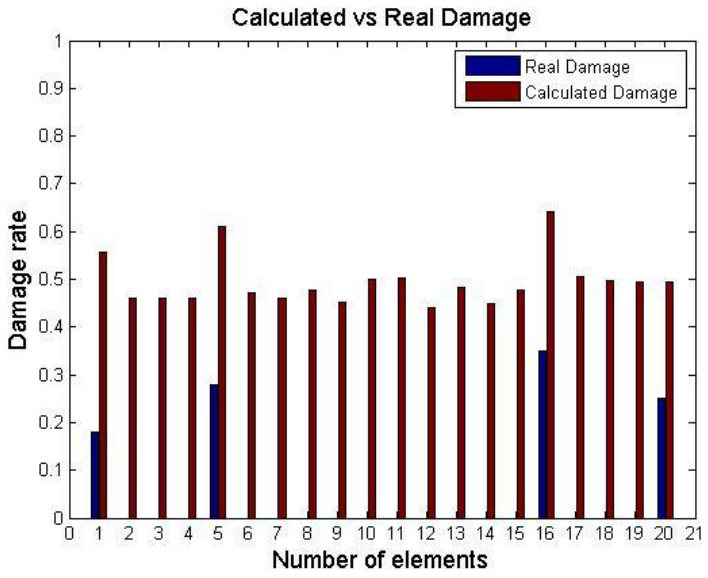


Figure 4.23 Damage results for damage case 1 of fixed frame using objective function F_1.

In this damage case for the fixed frame, the objective function using *MACFLEX* indicates the elements with damage and the range of the damage rate for these elements. For example it can be seen that the 16th element is more damaged than the 5th one. However this objective function fails to identify the damage in the 20th element.

ii) Optimization only with *MTMAC* criterion: $F_2 = 1 - MTMAC$

Table 4.19 Optimization results for damage case 1 of fixed frame using objective function F_1.

Element	1	2	3	4	5	6	7	8	9	10
Damage (%)	14.11	1.35	0.00	5.97	27.51	0.18	0.00	0.03	0.00	0.00

Element	11	12	13	14	15	16	17	18	19	20
Damage (%)	0.00	0.00	0.02	0.00	0.00	34.66	6.43	0.00	3.18	20.68

Minimum objective function value
$6.21 \cdot 10^{-6}$

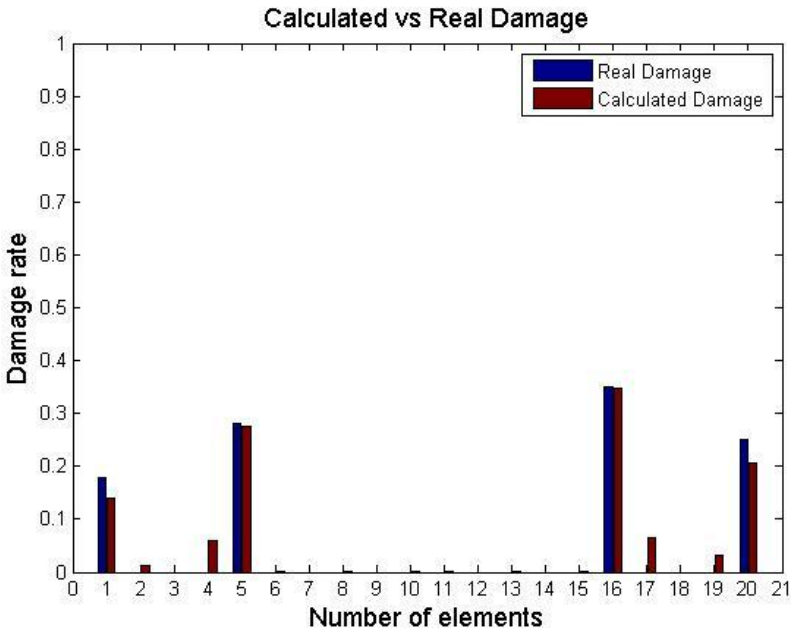


Figure 4.24 Damage results for damage case 1 of fixed frame using objective function F_2.

In this case, the objective function using *MTMAC* has rather satisfactory results in both the damage localization and quantification in the elements of the fixed frame. Nevertheless, some minor errors can be found in some of the frame elements.

iii) Optimization with both *MACFLEX* and *MTMAC* criteria:

$$F_3 = \sqrt{\left(250 \cdot \left(1 - MACFLEX\right)\right)^2 + \left(1 - MTMAC\right)^2}$$

Table 4.20 Optimization results for damage case 1 of fixed frame using objective function F_3.

Element	1	2	3	4	5	6	7	8	9	10
Damage (%)	17.99	0.00	0.00	0.00	27.98	0.02	0.00	0.02	0.00	0.00

Element	11	12	13	14	15	16	17	18	19	20
Damage (%)	0.00	0.00	0.02	0.00	0.09	34.57	6.96	0.00	2.41	20.78

Minimum objective function value
$3.99 \cdot 10^{-6}$

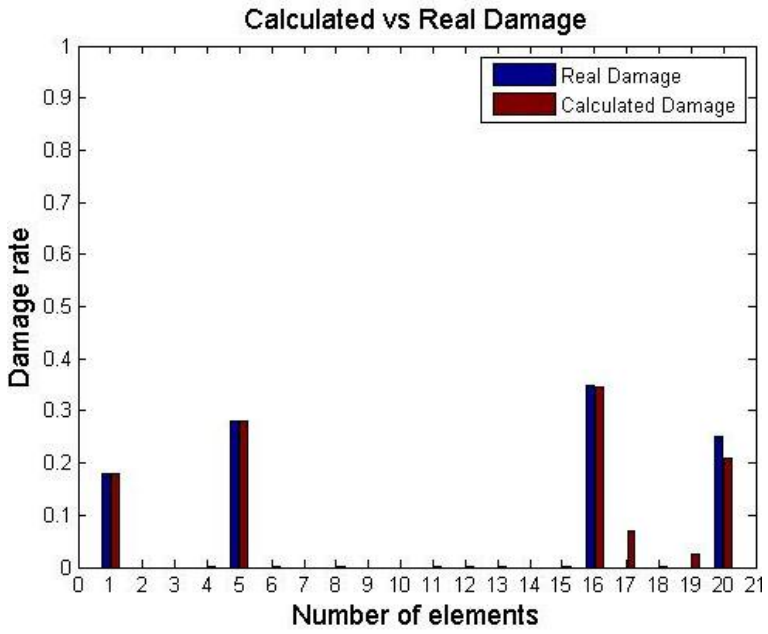


Figure 4.25 Damage results for damage case 1 of fixed frame using objective function F_3.

Again, the objective function using both *MACFLEX* and *MTMAC* with the multiplication factor clearly provides better results in identifying the damage rate and position of the frame elements.

b) Calculated data with noise 5% and 10%

Table 4.21 Optimization results for damage case 1 of fixed frame with noise using objective function F_3.

Noise	Element	1	2	3	4	5	6	7	8	9	10
5%	**Damage (%)**	17.87	0.00	0.00	0.00	27.71	0.00	0.04	0.20	0.00	0.00
10%	**Damage (%)**	17.55	0.00	0.00	0.00	28.06	0.00	0.00	0.00	0.00	0.00

Noise	Element	11	12	13	14	15	16	17	18	19	20
5%	**Damage (%)**	0.00	0.03	0.00	0.10	0.60	34.09	11.25	0.00	4.81	15.01
10%	**Damage (%)**	0.00	0.14	0.77	0.79	2.32	31.93	10.25	0.00	0.00	9.08

Noise	Minimum objective function value
5%	$3.78 \cdot 10^{-3}$
10%	$1.12 \cdot 10^{-2}$

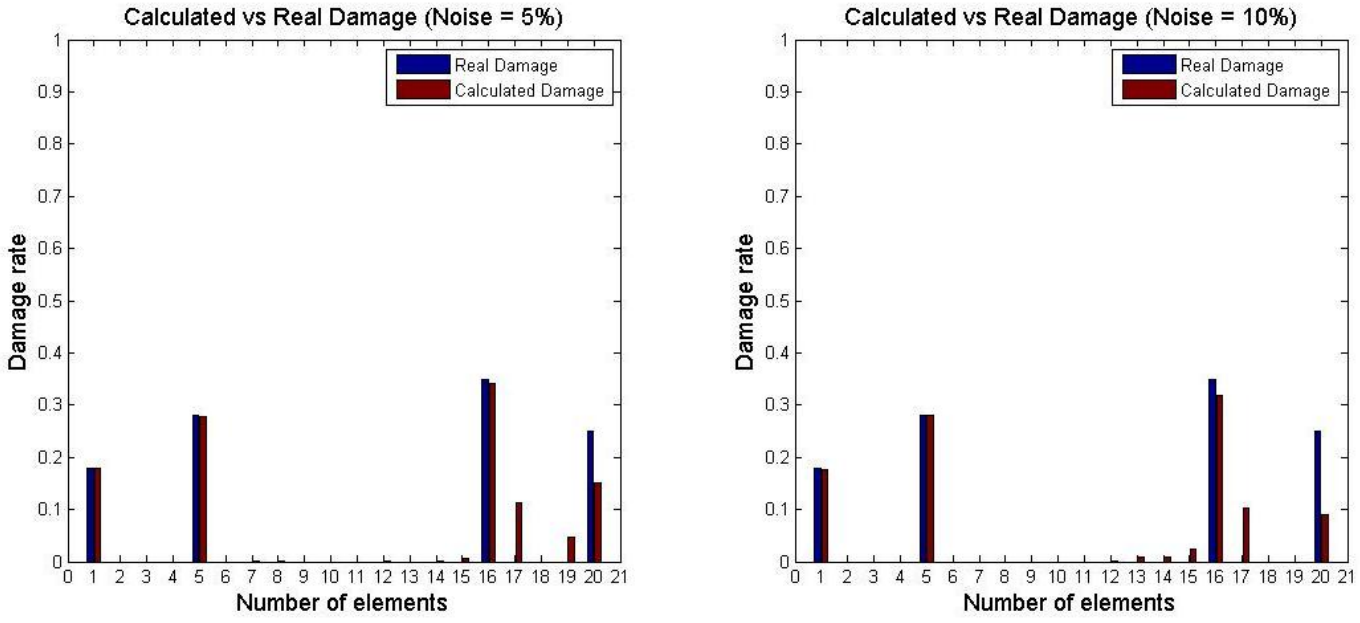


Figure 4.26 Damage results for damage case 1 of fixed frame with noise using objective function F_3.

It can be seen in Figure 4.31 that even though the noise partly influences the results of the optimization the match of the calculated and real damage is still acceptable.

4.2.3 Damage case 2

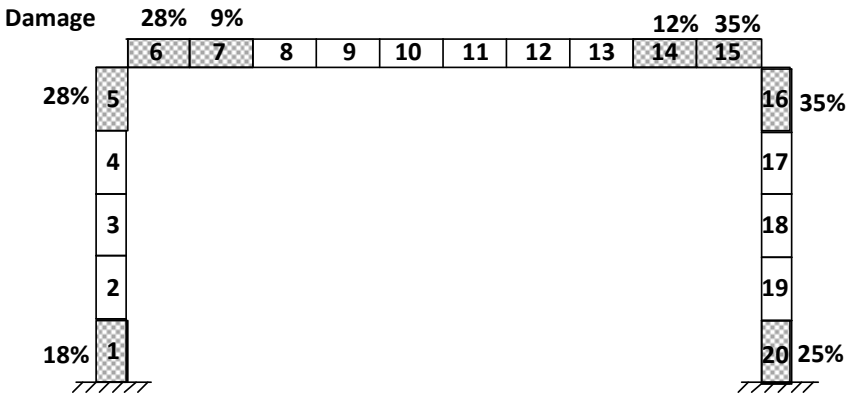


Figure 4.27 Damage case 2 of fixed frame with 20 elements.

a) No noise

iii) Optimization only with *MACFLEX* criterion: $F_1 = 1 - MACFLEX$

Table 4.22 Optimization results for damage case 2 of fixed frame using objective function F_1.

Element	1	2	3	4	5	6	7	8	9	10
Damage (%)	54.73	44.80	44.80	44.80	60.26	58.13	48.92	42.82	41.97	46.58

Element	11	12	13	14	15	16	17	18	19	20
Damage (%)	46.74	39.90	44.55	48.28	63.38	61.24	50.37	49.57	49.66	49.59

Minimum objective function value
$9.63 \cdot 10^{-6}$

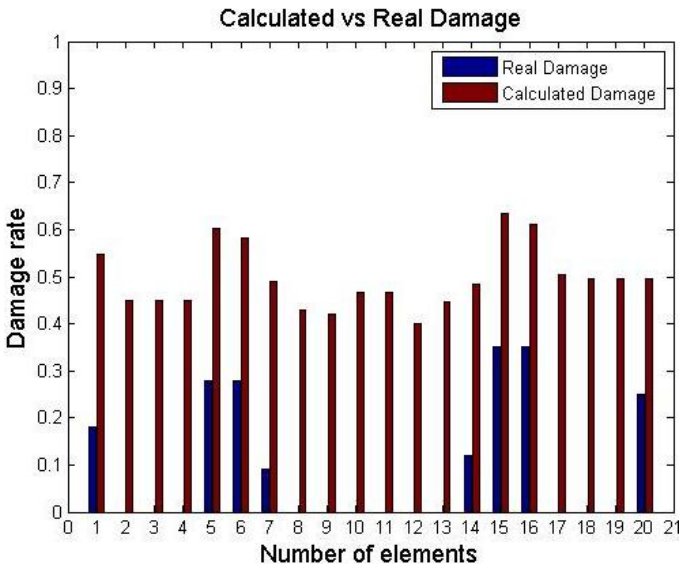


Figure 4.28 Damage results for damage case 2 of fixed frame using objective function F_1.

ii) Optimization only with *MTMAC* criterion: $F_2 = 1 - MTMAC$

In the second damage case for the fixed frame, the objective function using *MACFLEX* clearly indicates the elements with damage and the range of the damage rate for these elements. For example it can be seen that the 15th element is more damaged than the 16th one, although the real damage value of the elements is not determined.

Table 4.23 Optimization results for damage case 2 of fixed frame using objective function F_2.

Element	1	2	3	4	5	6	7	8	9	10
Damage (%)	12.06	0.74	0.01	11.32	27.23	28.17	8.84	0.00	0.00	0.10

Element	11	12	13	14	15	16	17	18	19	20
Damage (%)	0.00	0.04	0.00	12.00	34.79	34.54	12.23	0.00	3.20	18.61

Minimum objective function value
$1.25 \cdot 10^{-5}$

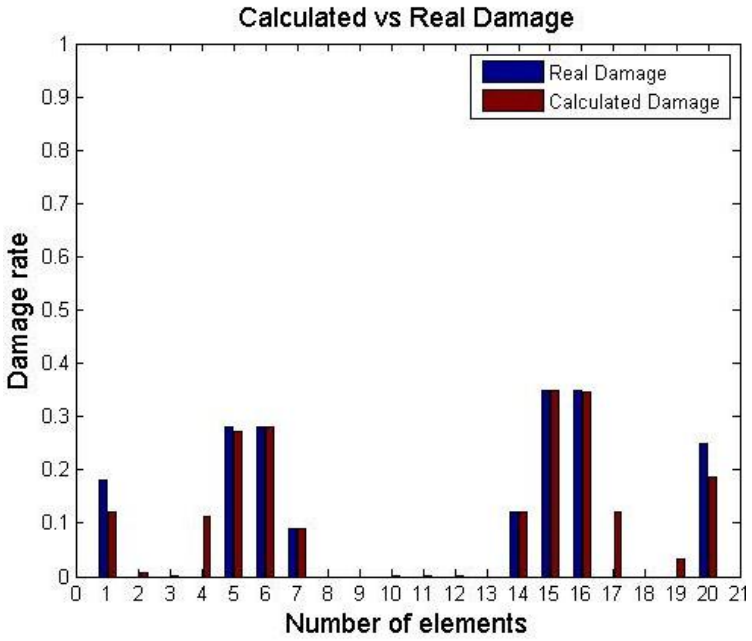


Figure 4.29 Damage results for damage case 2 of fixed frame using objective function F_2.

In this case, the objective function using *MTMAC* finds accurately the damage rate in some elements (5,6.7,14,15,16) but fails to identify the exact damage value in some others (1,20). Also, damage is discovered in healthy elements (4,17). These errors can be eliminated using the objective function F_3, which takes advantage of both *MACFLEX* and *MTMAC*.

iii) Optimization with both *MACFLEX* and *MTMAC* criteria:

$$F_3 = \sqrt{\left(415 \cdot \left(1 - MACFLEX\right)\right)^2 + \left(1 - MTMAC\right)^2}$$

Table 4.24 Optimization results for damage case 2 of fixed frame using objective function F_3.

Element	1	2	3	4	5	6	7	8	9	10
Damage (%)	18.00	0.00	0.00	0.00	28.01	27.95	9.01	0.00	0.00	0.10

Element	11	12	13	14	15	16	17	18	19	20
Damage (%)	0.00	0.04	0.00	11.87	34.96	34.60	9.49	0.00	2.52	20.16

Minimum objective function value
$4.12 \cdot 10^{-6}$

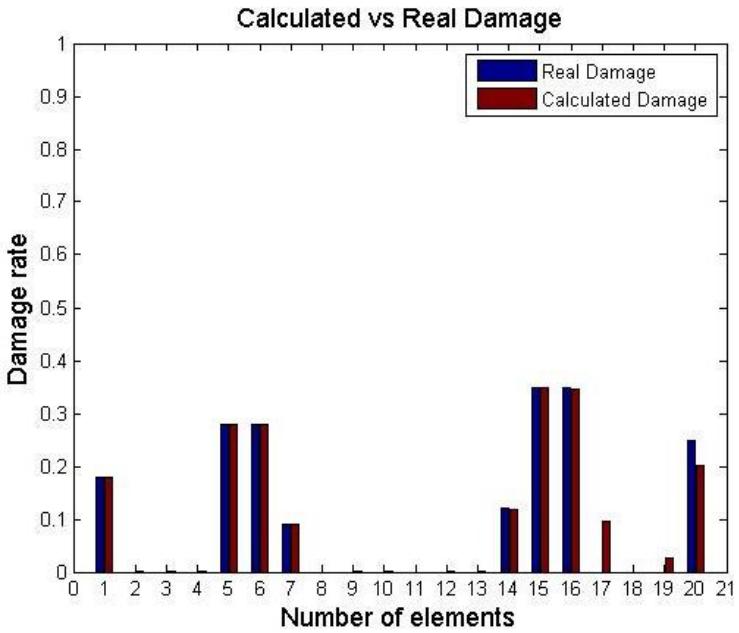


Figure 4.30 Damage results for damage case 2 of fixed frame using objective function F_2.

In this case the objective function using both *MACFLEX* and *MTMAC* with the multiplication factor improves again the results of damage detection in the frame elements compared to the ones from the objective functions using each criterion itself.

b) Calculated data with noise 5% and 10%

Table 4.25 Optimization results for damage case 2 of fixed frame with noise using objective function F_3.

Noise	Element	1	2	3	4	5	6	7	8	9	10
5%	**Damage (%)**	18.10	0.01	0.00	0.01	28.08	26.75	10.37	0.00	1.26	2.18
10%	**Damage (%)**	18.96	1.13	1.14	0.88	29.37	27.18	8.43	0.01	0.00	0.03

Noise	Element	11	12	13	14	15	16	17	18	19	20
5%	**Damage (%)**	0.00	0.02	0.00	8.90	34.29	35.15	10.94	0.83	5.17	17.11
10%	**Damage (%)**	0.00	0.00	0.00	12.45	35.77	32.85	13.86	7.59	0.64	10.83

Noise	Minimum objective function value
5%	$3.12 \cdot 10^{-3}$
10%	$1.26 \cdot 10^{-2}$

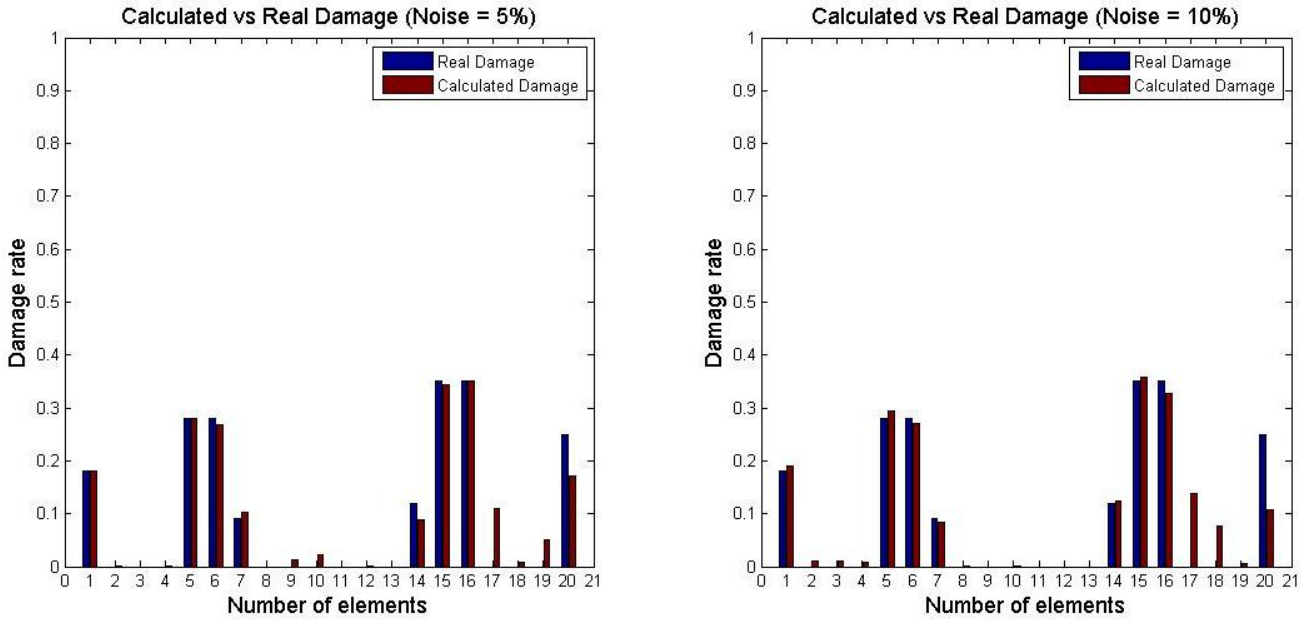


Figure 4.31 Damage results for damage case 2 of fixed frame with noise using objective function F_3.

Figure 4.31 indicates that even if noise is applied to the dynamic analysis data so that it is simulated to the experimental one, the optimization results provided with the objective

function F_3 lead to a satisfactory match between the real and the calculated damage in the elements of the fixed frame.

Chapter 5

5 Conclusions

5.1 Introduction

The main purpose of the current thesis was to identify the damage in structures, both in position and in extend, by using some distinct and easily measured dynamic characteristics of the structure. To solve this damage identification problem, the following two optimization algorithms were used:

i. Particle Swarm Optimization (PSO), which is a population based optimization method.

ii. Sequential Quadratic Programming (SQP), which is a mathematical optimization method.

The objective functions to be minimized in this unconstrained optimization problem compare the values of certain dynamic characteristics between the experimental and the numerical model of the structure.
The experimental model should contain the dynamic analysis data acquired by real experiments. In the current work real experimental data was not available and so this data was calculated by knowing the real damage (design variable) a priori and by implementing noise afterwards for the simulation with real experiments. The numerical model contains the dynamic analysis data calculated during the optimization procedure.
The objective functions used were the following:

Objective functions:

$$F_1 = 1 - MACFLEX$$

$$F_2 = 1 - MTMAC$$

$$F_3 = \sqrt{\left(mult \cdot \left(1 - MACFLEX\right)\right)^2 + \left(1 - MTMAC\right)^2}$$

Where:

MACFLEX is a criterion using the dynamic flexibility of the structure, which can be very sensitive at damage and has numerous advantages compared to stiffness,

MTMAC is a criterion using the natural frequencies and mode shapes of the structure which can indicate damage in structure with efficiency, and

mult is a multiplication factor that balances the range of values between the two objectives F_1 and F_2.

The third objective function (F_3) was implemented in the current thesis so that the advantages of the first two objective functions could be both utilized.

As it is a priori known that both objectives F_1 and F_2 have a single point solution at the point zero, the objective function F_3 was defined as the radius of the diagram containing the values of the objectives F_1 and F_2. In this way by minimizing F_3 the optimizer converges to a solution as close to the a priori known single point solution of the optimization problem as possible.

Also because of the fact that the objectives F_1 and F_2 are not in the same value range throughout the optimization runs, a multiplication factor (*mult*) was used in order to help the convergence of the optimization algorithm.

The test applications of the current work were to identify the damage position and rate for different damage cases in the following test structures:

- Simply supported beam with 10 elements using the first three eigenmodes.
- Fixed frame with 20 elements using the first three eigenmodes.

5.2 General conclusions

First of all, it became clear that the mathematical optimization methods and more specifically SQP was proven to be more suitable to handle these types of damage identification problems in comparison with PSO. In the test application of the simply supported beam with 10 elements the SQP algorithm converged to a better solution by using each one of the three objective functions. Additionally, the time needed for the algorithm convergence was significantly lower in SQP than in PSO. In order to exploit the advantages of better convergence and lower computational cost, SQP was the only optimization algorithm used in the more complex test application of the fixed frame.

The first objective function using the *MACFLEX* criterion can indicate the position of the damage in structures and the range of damage in structural elements, although it cannot lead to accurate damage values. More specifically, this objective function can be a useful tool to identify both if there is damage to a structural element and the extent of this damage compared to other damaged elements. However, using the *MACFLEX* criterion as the single objective function for the damage identification problem cannot provide a final solution due to its inaccuracy of the real damage values.

The second objective function using the *MTMAC* criterion can provide far more better results than the first in identifying the damage position and extend of structural elements. However some times it can lead to some significant errors as it is shown in the test applications of the simply supported beam and the fixed frame.

The assumption that this type of problem could be handled with multiobjective approaches was not taken into account, as neither of the following definitions of a multiobjective problem were satisfied:

- The objectives F_1 and F_2 have not conflicting values.
- There is a single solution that minimizes both objectives F_1 and F_2 at the same time (0,0).

The third objective function using both the *MACFLEX* and *MTMAC* criteria can indicate with the best possible accuracy, the position and the exact value of damage in structures. The reason for its efficiency is the fact that it can utilize the advantages of both objectives F_1 and F_2 at the same time. In every damage case of both test structures, the objective function F_3 was proven to lead to the best match between the real and the calculated damage of the structural elements.

The efficiency of the objective function F_3 depends greatly on a suitable choice of a multiplication factor for the *MACFLEX* criterion. This factor is applied due to the fact that during the optimization process the objective function using *MACFLEX* takes significantly lower values than the one using *MTMAC*. In order to ensure the best algorithm convergence a multiplication factor is selected so that both objectives have an equal contribution to the objective function. It should be noted that the choice of this factor is made after some test runs of the optimization algortihms.

To make the case of the success of the objective function F_3 in the damage identification process even stronger, some random noise was applied to the initial mode shape data, so that the experimental measurements errors are taken into account. Even with noise 10% the optimization algorithm using the objective function F_3 provided a great match of the real and calculated damage data, although it had some minor impact in its efficiency.

The major advantages of the optimization methods used to solve these types of problems were also illustrated. More specifically, the efficiency in finding the rightful solution and the low computational cost are of major importance in these engineering problems. Nevertheless, the damage identification procedure presented in the current thesis should not replace the actual work and judgement of a designer engineer but provide a supplement of useful ideas and conclusions.

In general a simple yet quite useful tool was presented to solve damage identification problems. Even in large-scale structures with the measurement of a small number of their dynamic characteristics, the damage in its structural elements can be revealed both in position and in actual value within a limited amount of time. The simulated and real case studies have shown that the proposed methodology is robust, fast and easy to implement.

5.3 Future research

The current thesis presents a method that can detect the damage position and rate in structural elements. This method was tested to a simply supported beam and a fixed frame with a limited amount of elements. So, an expansion of this study is to test this method on more complex and large-scale structures with many structural elements, where the optimization methods may have a much harder time to achieve the damage localization and quantification.

The objective functions that were used in this case study were proven to be indicative of damage in the structural elements in a rather efficient way. Nevertheless, the possibility that the use of different objective functions in the damage identification problems can lead to even better results should not be ruled out. This is due to the fact that the data calculated from the dynamic analysis of a structure can indicate damage in a lot of different ways.

Also, the use of a multiplicative factor for the objective function using *MACFLEX* was an assumption that was taken in order to balance the range of values of the objective functions. The selection of a suitable factor depends on the problem and should be a subject of further study and research.

Finally, as it was stated previously real experimental data was not available in the context of the current thesis and the dynamic analysis data was calculated using the damage index as a priori known and was simulated to the experimental one using random noise. It becomes clear that if the dynamic analysis data of the structure such as mode shapes, natural frequencies etc., was measured with real experiments the differentiation of the results might be notable. In this way, the efficiency of the method proposed in the current thesis for damage identification should be also demonstrated with the use of real experimental data.

Bibliography

Allemang, R. J. (2003, August). The Modal Assurance Criterion (MAC):Twenty Years of Use and Abuse. *Sound and Vibration*, pp. 14-21.

Bernal, D. (2000 a). Extracting Flexibility Matrices from State-Space Realizations. *European COST F3 Conference on System Identification and Structural HealthMonitoring*, (pp. 127-135). Madrid, Spain.

Chopra, A. K. (2007). Dynamics of Structures. In *Theory and Applications to Earthquake Engineering* (3rd ed., pp. 401-445). Berkeley: Pearson, Prentice Hall.

Gill, P. E., Murray, W., & Wright, M. E. (1991). *Numerical Linear Algebra and Optimization* (Vol. I). Addison-Wesley.

Inman, D. J. (2006). *Vibration with control.* Chichester, UK: John Wiley & Sons Ltd.

Kelly, G. S. (2011). Free vibrations of MDOF systems. In *Mechanical Vibrations Theory and Applications* (pp. 533-593). Akron: Thomson-Engineering.

Kennedy, J., & Eberhart, R. (1995). Particle Swarm Optimization. *IEEE International Conference on Neural Networks.* Piscataway, NJ, USA.

Mouzakis, H. P., & Dertimanis, V. K. (2011). Experimental modal analysis. In *Notes on Experimental Earthquake Engineering* (pp. 85-110). Athens.

Papadrakakis, M. (1996). *Μαθήματα Στατικής ΙΙΙ - Σύγχρονες μέθοδοι αναλύσεως φορέων.* Athens: NTUA.

Perera, R., Ruiz, A., & Manzano, C. (2007, October). An evolutionary multiobjective framework for structural damage localization and quantification. *Engineering Structures, 29*(10), pp. 2540-2550.

Plevris, V. (2009). PhD Dissertation in Innovative Computational Techniques for the Optimum Structural Design Considering Uncertainties. Athens: National Technical University of Athens.

Plevris, V., & Papadrakakis, M. (2011, January). A Hybrid Particle Swarm - Gradient Algorithm for Global Structural Optimization. *Computer-Aided Civil and Infrastructure Engineering*(26), pp. 48-68.

Rades, M. (2010). Test-analysis correlation. In M. Rades, *Mechanical Vibrations II Structural Dynamic Modeling* (pp. 287-314). Printech.

Udwadia, F. E. (2005, July). Structural Identification and Damage Detection from Noisy Modal Data. *Journal of Aerospace Engineering, 18*(3), pp. 179-187.

Yu, L., & Chen, X. (2010). Bridge Damage Identification by Combining Modal Flexibility and PSO Methods. *Prognostics and Health Management Conference, 2010. PHM '10.* Macau.