ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΠΟΛΙΤΙΚΩΝ ΜΗΧΑΝΙΚΩΝ
ΤΟΜΕΑΣ ΔΟΜΟΣΤΑΤΙΚΗΣ
ΕΡΓΑΣΤΗΡΙΟ ΣΤΑΤΙΚΗΣ ΚΑΙ ΑΝΤΙΣΕΙΣΜΙΚΩΝ ΕΡΕΥΝΩΝ

ΒΕΛΤΙΣΤΟΠΟΙΗΣΗ ΤΟΠΟΛΟΓΙΑΣ ΜΕ ΧΡΗΣΗ ΜΟΝΑΔΩΝ ΕΠΕΞΕΡΓΑΣΙΑΣ ΓΡΑΦΙΚΩΝ ΣΕ ΠΡΟΒΛΗΜΑΤΑ ΠΟΛΙΤΙΚΟΥ ΜΗΧΑΝΙΚΟΥ: ΘΕΩΡΗΣΗ ΜΑΖΙΚΩΝ ΚΑΙ ΘΕΡΜΙΚΩΝ ΔΡΑΣΕΩΝ ΣΕ ΔΥΟ ΚΑΙ ΤΡΕΙΣ ΔΙΑΣΤΑΣΕΙΣ

Διπλωματική Εργασία

ΚΑΖΑΚΗΣ ΓΕΩΡΓΙΟΣ

Επιβλέπων
Λαγαρός Νικόλαος, Επίκουρος Καθηγητής ΕΜΠ
Συνεπιβλέπωντες
Μιχαηλίδης Γεώργιος, SIMaP, INP Grenoble
Καλλιώρας Νικόλαος, ΥΔ ΕΜΠ

ΜΑΡΤΙΟΣ 2016

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον κ. Νικόλαο Λαγαρό, επιβλέποντα της διπλωματικής εργασίας, για την ανάθεση του συγκεκριμένου θέματος και τη στήριξη καθ' όλη τη διάρκεια της εκπόνησής του. Επίσης θα ήθελα να ευχαριστήσω τον κ. Γεώργιο Μιχαηλίδη ο οποίος συνέβαλε τα μέγιστα στην υλοποίηση της διπλωματικής εργασίας. Επιπλέον ευχαριστώ τον κ. Νικόλαο Καλλιώρα για τη πολύτιμη επιστημονική βοήθεια που μου παρείχε. Τέλος πάνω από όλα θέλω να ευχαριστήσω τους δικούς μου ανθρώπους για την υποστήριξή τους.

ΠΕΡΙΛΗΨΗ

Η παρούσα διπλωματική εργασία αποτελείται από δύο μέρη. Στο πρώτο μέρος εξετάζεται η επίδραση μαζικών και θερμικών δυνάμεων στο πρόβλημα βελτιστοποίησης τοπολογίας με στόχο τη δημιουργία κατασκευών με δενδροειδή μορφή. Στο δεύτερο μέρος διερευνώνται μέθοδοι και τεχνικές με τις οποίες μπορεί να γίνει χρήση της κάρτας γραφικής επεξεργασίας για την ελαχιστοποίηση του χρόνου υλοποίησης του αλγόριθμου βελτιστοποίησης τοπολογίας.

ABSTRACT

This thesis consists of two parts. The first part examines the influence of mass and thermal forces in topology optimization problems in order to create tree-like structures. The second part explores methods and techniques which use the Graphic Processor Unit in order to minimize the implementation time of topology optimization algorithms.

Περιεχόμενα

ΚΕΦΑΛΑΙΟ 1

ΕΙΣΑΓΩΓΗ

1.1 Βελτιστοποίηση Τοπολογίας

Η βελτιστοποίηση τοπολογίας είναι μια μαθηματική διαδικασία η οποία έχει ως στόχο την εύρεση του βέλτιστου σχήματος μιας κατασκευής της οποίας έχουν προσδιοριστεί οι φορτίσεις και οι στηρίξεις. Η εφαρμογή αυτής της μεθόδου βοηθάει στη δημιουργία ενός φορέα που προσεγγίζει σε μεγάλο βαθμό την τελική κατασκευή έχοντας ως αποτέλεσμα την εξοικονόμηση χρόνου σύλληψης του φορέα.

1.2 Στόχος Διπλωματικής Εργασίας

Οι στόχοι της διπλωματικής εργασίας είναι δύο. Αρχικά η μόρφωση δενδροειδών κατασκευών μέσο του αλγορίθμου βελτιστοποίησης τοπολογίας και στη συνέχεια η ελαχιστοποίηση του χρόνου υλοποίησης του αλγορίθμου αυτού. Για να επιτεθούν οι στόχοι αυτοί επιλέγηκαν οι αλγόριθμοι top88 του Ole Sigmund[1] για την εφαρμογή σε δύο διαστάσει και top3D των Allaire και Kelly[2] για την εφαρμογή σε τρεις διαστάσεις. Η πλατφόρμα εφαρμογής είναι η γλώσσα προγραμματισμού Matlab, το πλεονέκτημα που δίνει η Matlab είναι η εύκολη γραφική απεικόνιση που προσφέρει σε σχέση με μια πιο γρήγορη γλώσσα προγραμματισμού όπως είναι η C++. Η μέθοδος που υιοθετήθηκε για τον υπολογισμό του μέτρου ελαστικότητας E στη βελτιστοποίηση τοπολογίας είναι η SIMP(Solid Isotropic Material with Penalization). Η SIMP είναι ευρέος διαδεδομένη μέθοδος και αρκετά αποτελεσματική για απλά προβλήματα βελτιστοποίησης τοπολογίας.

Για την μόρφωση δενδροειδών κατασκευών υιοθετήθηκε η εφαρμογή θερμικών φορτίσεων. Όπως και τα δέντρα έτσι και μια κατασκευή θα πάρει δενδροειδή μορφή προκειμένου να μεταφέρει με βέλτιστο τρόπο τη θερμότητα που δέχεται από το περιβάλλον στο έδαφος. Επομένως ο συνδυασμός θερμικών και στατικών δυνάμεων μπορεί να δώσει ως αποτέλεσμα μια κατασκευή της οποίας η μορφή είναι δενδροειδής και έχει τη δυνατότητα να πάρει τα στατικά φορτία.

Για την βελτιστοποίηση του χρόνου έγινε η χρήση της υπολογιστικής ικανότητας της μονάδας επεξεργασίας γραφικών GPU. Η GPU, έχοντας περισσότερους πυρήνες από τον επεξεργαστή, έχει αρκετά μεγαλύτερη υπολογιστική δυνατότητα. Το μειονέκτημα στη χρήση της είναι η μνήμη της. Σε αντίθεση με τον επεξεργαστή, ο οποίος αξιοποιεί τη RAM, η GPU μπορεί να αξιοποιήσει μόνο τη δική της μνήμη. Αυτό περιορίζει αρκετά τη χρήση της στις μεθόδους βελτιστοποίησης τοπολογίας, οι

οποίες απαιτούν την επίλυση της εξίσωσης ισορροπίας. Για την αντιμετώπιση αυτού του προβλήματος έχουν εφαρμοστεί κάποιες μέθοδοι. Η αποδοτικότητα τους στη Matlab καθώς και άλλες λύσεις εφαρμόζονται σε αυτή τη διπλωματική.

1.3 Δομή της Διπλωματικής

Στο κομμάτι αυτό αναγράφεται η δομή της διπλωματικής εργασίας, καθώς και περιληπτικά το περιεχόμενο του κάθε κεφαλαίου.

- ΚΕΦΑΛΑΙΟ 2
  ΒΙΒΛΙΟΓΡΑΦΙΚΗ ΕΠΙΣΚΟΠΗΣΗ
  Στο κεφάλαιο αυτό γίνεται αναφορά στα συγγράμματα και τις δημοσιεύσεις που χρειάστηκε να μελετηθούν για να πραγματοποιηθεί η διπλωματική εργασία.
- ΚΕΦΑΛΑΙΟ 3
  ΘΕΩΡΙΤΙΚΟ ΥΠΟΒΑΘΡΟ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗΣ ΤΟΠΟΛΟΓΙΑΣ
  Στο κεφάλαιο αυτό αναλύονται οι βασικές έννοιες που είναι απαραίτητες για την κατανόηση των προβλημάτων της βελτιστοποίησης τοπολογίας.
- ΚΕΦΑΛΑΙΟ 4
  ΚΩΔΙΚΕΣ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗΣ ΤΟΠΟΛΟΓΙΑΣ ΜΕ ΧΡΗΣΗ ΤΗΣ ΜΕΘΟΔΟΥ SIMP
  Στο κεφάλαιο αυτό γίνεται επεξήγηση της λειτουργίας των αλγορίθμων που αποτέλεσαν τη βάση για τις εφαρμογές που πραγματοποιήθηκαν.
- ΚΕΦΑΛΑΙΟ 5
  ΕΠΕΜΒΑΣΕΙΣ ΚΑΙ ΜΟΡΦΟΠΟΙΗΣΕΙΣ ΣΤΟΥΣ ΚΩΔΙΚΕΣ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗΣ ΤΟΠΟΛΟΓΙΑΣ
  Στο κεφάλαιο αυτό αναλύονται οι εφαρμογές που έγιναν προκειμένου να επιτευθεί η μόρφωση δενδροειδών κατασκευών, καθώς και ελαχιστοποίηση του χρόνου με χρήση GPU.
- ΚΕΦΑΛΑΙΟ 6
  ΣΥΜΠΕΡΑΣΜΑΤΑ
  Στο κεφάλαιο αυτό αναγράφονται τα συμπεράσματα που προέκυψαν από τη παρούσα διπλωματική.

ΚΕΦΑΛΑΙΟ 2

ΒΙΒΛΙΟΓΡΑΦΙΚΗ ΕΠΙΣΚΟΠΗΣΗ

2.1 Εισαγωγή

Στο κεφάλαιο αυτό αναφέρεται η βιβλιογραφική έρευνα η οποία συντελέστηκε κατά τη διάρκεια εκπόνησης της διπλωματικής εργασίας. Αυτή είχε ως στόχο να υπάρξει επαρκής κατανόηση του θεωρητικού υποβάθρου της βελτιστοποίησης τοπολογίας και των εφαρμογών της καθώς και τη χρήση της κάρτας γραφικής επεξεργασίας GPU. Γίνεται αναφορά σε κείμενα και βιβλία τα οποία χρησιμοποιήθηκαν καθώς και μερικά λόγια για το περιεχόμενο του κάθε ενός. Τα αποτελέσματα της βιβλιογραφικής έρευνας χωρίζονται στα βιβλία, τις δημοσιεύσεις καθώς και του κώδικες προγραμματισμού που έχουν γραφεί πάνω στη βελτιστοποίηση τοπολογίας.

Η πρώτη επαφή με τον τομέα της βελτιστοποίησης τοπολογίας έγινε μέσω μιας δημοσίευσης των Ole Sigmund και Kurt Maute του 2013, που ονομάζεται Topology optimization approaches , A comparative review[3]. Η δημοσίευση αρχικά κάνει μια ιστορική επισκόπηση της βελτιστοποίησης, αναφέροντας πως οι πρώτες πρακτικές χρήσεις της ήταν στους τομείς των ηλεκτρολόγων και μηχανολόγων μηχανικών και τα τελευταία χρόνια έχει ξεκινήσει να βρίσκει εφαρμογή στις κατασκευές. Έπειτα συνεχίζει με μια γενική επισκόπηση της εξέλιξης της βελτιστοποίησης τοπολογίας, των διαφόρων προσεγγίσεων που έχουν γίνει στο πρόβλημα και των μεθόδων επίλυσης που έχουν αναπτυχθεί.

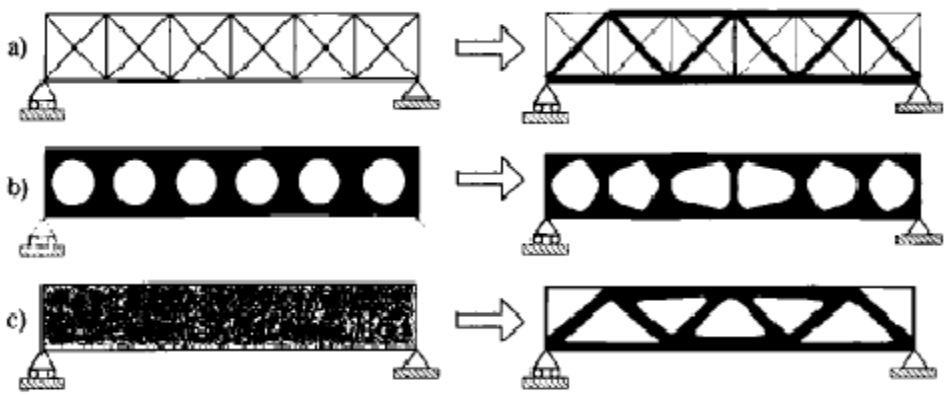
2.2 Συγκράματα που αναφέρονται στη βελτιστοποίηση τοπολογίας

Συνεχίζοντας αναφέρονται τα βιβλία Topology Optimization Theory, Methods and Applications και An Introduction to Structural Optimization και Solid Mechanics and its Applications, τα οποία αποτελούν βασικά συγκράματα για τον τομέα της βελτιστοποίησης τοπολογίας.

Οι M.P.Bendsoe και O.Sigmund εξέδωσαν το 2004 το βιβλίο Topology Optimization Theory, Methods and Applications[4]. Το βιβλίο αυτό είναι μια ανανεωμένη έκδοση του βιβλίου Optimization of Structural Topology, Shape and Material(1995) του M.P.Bendsoe, η οποία εκδόθηκε για να συμπεριλάβει όλες τις θεωρίες και εξελίξεις που υπήρξαν στον τομέα από τότε που γράφτηκε το αρχικό βιβλίο. Στο βιβλίο αυτό γίνεται η μαθηματική μορφοποίηση του προβλήματος της βελτιστοποίησης τοπολογίας και συνεχίζει αναφέροντας μεθόδους και φόρμουλες που μπορούν να

αξιοποιηθούν για να επιλυθεί. Αναφέρονται κάποιες επιπλοκές που δημιουργούνται με τη χρήση διαφόρων μεθόδων καθώς και τρόπους να αποφευχθούν. Τα παραπάνω αναπτύσσονται για κάθε μια από τις τρεις κύριες κατηγορίες στις οποίες είναι χωρισμένο το βιβλίο. Τη βελτιστοποίηση τοπολογίας σε ισότροπα υλικά, τη βελτιστοποίηση τοπολογίας σε ανισότροπα υλικά και τη βελτιστοποίηση τοπολογίας σε κατασκευές δικτυωμάτων. Στο ένθετο στο τέλος του βιβλίου δίνονται και παραδείγματα γραμμένα σε κώδικα της γλώσσας προγραμματισμού matlab καθώς και περισσότερα στοιχεία για προβλήματα και μεθόδους τα οποία παρουσιάζονται εντός του βιβλίου. Σε αυτό το βιβλίο γίνεται αναφορά στη βελτιστοποίηση τοπολογίας με με βάση την ελαχιστοποίηση του έργου καθώς και την αυξομείωση των πυκνοτήτων των στοιχείων προκειμένου να επιτευχθεί η βελτιστοποίηση. Όπως επίσης και το μοντέλο SIMP και η Optimality Criteria method. Αυτές οι έννοιες θα αποτελέσουν τη βάση για την κατανόηση του τρόπου λειτουργίας της βελτιστοποίησης τοπολογίας και των προγραμμάτων τα οποία αναφέρονται στα επόμενα κεφάλαια.

Επόμενο βιβλίο είναι το An Introduction to Structural Optimization, Solid Mechanics and its Applications[5] που εξέδωσαν οι Peter W. Christensen και Andres Klarbring το 2009. Το βιβλίο αυτό είναι συνδυασμός διαλέξεων που δόθηκαν στο Linkoping University στη Σουηδία σε μια περίοδο δεκαπέντε χρόνων. Στο βιβλίο αυτό γίνεται διαχωρισμός της βελτιστοποίησης κατασκευών σε τρεις κατηγορίες. Τη βελτιστοποίση του μεγέθους (size optimization), τη βελτιστοποίηση της μορφής (shape optimization) και τη βελτιστοποίηση της τοπολογίας (topology optimization). Αυτές οι τρεις κατηγορίες και η σημασία τους θα αναλυθούν στα επόμενα κεφάλαια.



Εικόνα 1. a) Βελτιστοποίηση μεγέθους διατομής δικτυωμάτων. b) Βελτιστοποίηση μορφής. c) Βελτιστοποίηση τοπολογίας.

Αρχικά το βιβλίο κάνει μια εισαγωγή στις βασικές έννοιες της βελτιστοποίησης κατασκευών. Έπειτα εισάγει την έννοια του "κυρτού προγραμματισμού" (convex

programming) η οποία έχει ιδιαίτερη σημασία γιατί εξασφαλίζει την ύπαρξη ελαχίστου σε μια συνάρτηση βελτιστοποίησης. Αναφέρεται στην έννοια της Lagrangian Duality, που είναι ένας βασικός τρόπος επίλυσης προβλημάτων ελαχιστοποίησης με περιορισμούς. Αναλύει κάποιες μεθόδους που χρησιμοποιούνται για να μετατρέψουν ένα μη κυρτό πρόβλημα σε κυρτό. Ένα παράδειγμα τέτοιας μεθόδου είναι η MMA. Έχοντας ως βάση τα παραπάνω στο τελευταίο κομμάτι του βιβλίου αναλύεται η βελτιστοποίηση τοπολογίας σε κατασκευές "distributed parameter systems" με δείκτη τη δυσκαμψία.

2.3 Δημοσιεύσεις για τη βελτιστοποίηση τοπολογίας

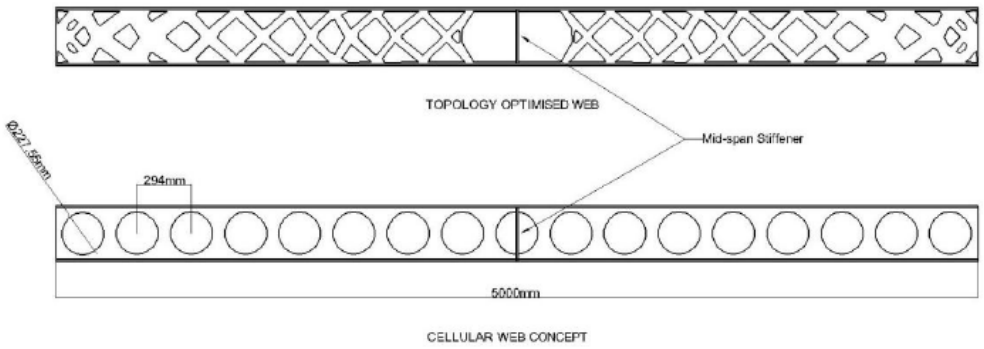
Τώρα προχωράμε σε δημοσιεύσεις που έχουν γίνει και αναφέρονται στη βελτιστοποίηση τοπολογίας.

Οι Per Dombernowsky και Asbjorn Sondergaard δημοσίευσαν το 2009 το Three-dimensional topology optimization in architectural and structural design of concrete structures[6]. Η δημοσίευση αυτή αναφέρεται στην αρχιτεκτονική και τη βελτιστοποίηση τοπολογίας σε κατασκευές από σκυρόδεμα αλλά και στα ίδια τα δομικά στοιχεία. Υποστήριξαν ότι η βελτιστοποίηση τοπολογίας μπορεί να χρησιμοποιηθεί στα αρχικά στάδια της σύλληψης του φορέα δίνοντας ένα αρχικό βέλτιστο σχήμα πάνω στο οποίο θα είναι πιο εύκολη η σύλληψη του τελικού φορέα. Επίσης, με χρήση μεθόδων βελτιστοποίησης σε δομικά στοιχεία, μπορούν να προκύψουν καλύτερες μορφές και σχήματα τα οποία εξοικονομούν υλικό χωρίς όμως να μειώνουν την αποτελεσματικότητα και την απόδοση την οποία έχει το δομικό στοιχείο. Ένα παράδειγμα αποτελεί η πλάκα η οποία απεικονίζεται στη δημοσίευσή τους.

Εικόνα 2. Μορφή πλάκας μετά από βελτιστοποίηση

Οι Konstantinos Daniel Tsavdaridis, James J. Kingman και Vassili V. Toropov στη δημοσίευσή τους με όνομα Structural Topology Optimization Study And Novel

Perforated Beams[7] ασχολήθηκαν με την εφαρμογή της βελτιστοποίησης τοπολογίας σε μεταλλικές δοκούς με οπές. Σκοπός τους ήταν να προσδιορίσουν με χρήση της βελτιστοποίησης τη μορφή και τον αριθμό των οπών σε μια μεταλλική δοκό μορφής I. Η βελτιστοποίηση έγινε με βάση την ελάχιστη ενέργεια και το ελάχιστο βάρος. Μέσα από τις αναλύσεις και προσομοιώσεις που έκαναν είδαν ότι η βελτιστοποιημένη δοκός είχε καλύτερες ιδιότητες, όπως η δυσκαμψία της, από την τυπική δοκό. Κάποια από τα αποτελέσματα φαίνονται στην εικόνα. Η βελτιστοποίηση τοπολογίας στις μεταλλικές δοκούς έχει αρκετό ενδιαφέρον επειδή είναι μαζικά παραγόμενες και επομένως είναι πιο εύκολο να εφαρμοστεί σε αυτές.



Εικόνα 3. a) Βελτιστοποιημένη δοκός b) τυπική δοκός

Οι Oded Amir και Ole Sigmund το 2010 δημοσίευσαν το έγγραφο On reducing computational effort in topology optimization: how far can we go[8]? Στο οποίο αναφέρονται στους διαφόρους τρόπους θα μπορούσε να μειωθεί ο χρόνος που χρειάζεται για να ολοκληρωθεί ένας αλγόριθμος βελτιστοποίησης τοπολογίας. Ένας αρκετά αποτελεσματικός τρόπος που αναφέρεται είναι η υιοθέτηση της μεθόδου PCG(Preconditioned Conjugated Gradient) για την επίλυση της εξίσωσης ισορροπίας. Επίσης και η εξάρτηση της ακρίβειας των μετακινήσεων που προκύπτουν από τη PCG στη βελτιστοποίηση των κατασκευών.

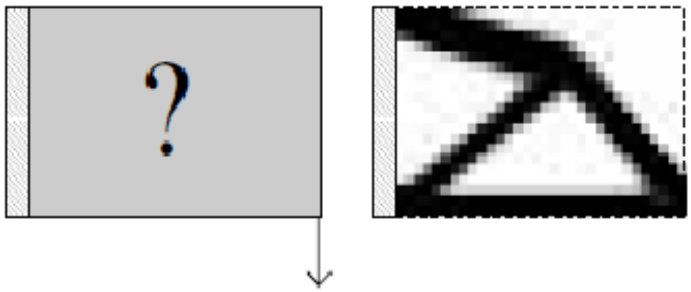
Οι Michael Bruyneel και Pierre Duysinx δημοσίευσαν το Topology Optimization with self-weight loading: unexpected problems and solutions[9]. Στη δημοσίευση αυτή γίνεται αναφορά στο πρόβλημα της δημιουργίας μεγάλης επιφάνειας γρι υλικού όταν η βελτιστοποίηση τοπολογίας λαμβάνει υπόψη της και της βαρύτιμες δυνάμεις τις κατασκευής. Επίσης παρουσιάζεται ο λόγος και προτείνεται μια τροποποίηση της μεθόδου SIMP ως λύση του προβλήματος.

Ο Jonathan Richard Shewchuk δημοσίευσε το 1994 το Introduction to the Conjugate Gradient Method Without the Agonizing Pain[10]. Αντικείμενο της δημοσίευσης είναι η δομή και η λειτουργία και η εφαρμογή της μεθόδου Conjugated Gradient καθώς και των μεθόδων Gradient Descent και Conjugated Directions. Προκειμένου να γίνει κατανοητές στην αρχή της δημοσίευσης υπάρχουν και αρκετές αναφορές σε προκαταρκτικές γνώσεις που είναι απαραίτητες.

Η δημοσίευση High-Performance GPU computing for density-based topology optimization[11] η οποία εκδόθηκε από το περιοδικό International Journal for Numerical Methods in Engineering αναφέρεται στην αξιοποίηση της κάρτας γραφικής επεξεργασίας GPU στα προβλήματα βελτιστοποίησης τοπολογίας. Ποιό συγκεκριμένα στις εφαρμογές που αξιοποιούνται για τη χρήση της GPU τε τέτοια προβλήματα στο περιβάλλον της γλώσσας προγραμματισμού C.

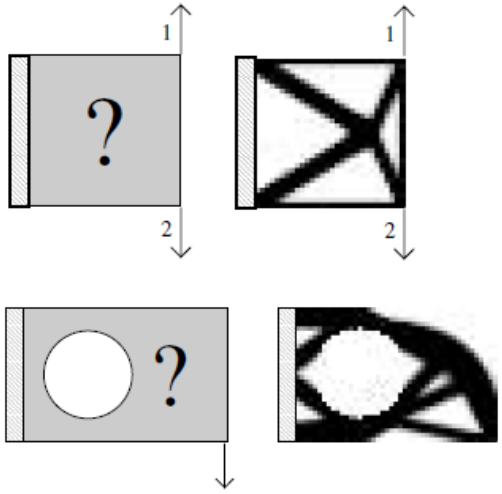
2.4 Κώδικες σχετικά με τη βελτιστοποίηση τοπολογίας

Το 2001, ο O.Sigmund εξέδωσε ένα άρθρο σχετικά με τη βελτιστοποίηση τοπολογίας σε δυο διαστάσεις, στο οποίο ανέπτυξε έναν κώδικα 99 γραμμών[12] γραμμένο σε matlab. Στο άρθρο αυτό αρχικά δίνεται το θεωρητικό υπόβαθρο του προβλήματος και της λύσης την οποία ακολουθεί ο κώδικας. Έπειτα επεξηγείται ο τρόπος με τον οποίο η λύση αυτή αποτυπώθηκε μέσα στον κώδικα έχοντας ως βάση το πρόβλημα βελτιστοποίησης μιας δοκού που της ασκείται ένα φορτίο και είναι πακτωμένη από τη μια πλευρά, όπως στην εικόνα. Επίσης στο τέλος της.



Εικόνα 4. Πρόβλημα βελτιστοποίησης κώδικα 88 γραμμών

εργασίας αναπτύσσονται και οι τρόποι με τους οποίους μπορούν να συμπεριληφθούν στον κώδικα κάποιες ειδικές περιπτώσεις, όπως οι πολλαπλές δυνάμεις και η ύπαρξη περιοχής που δεν πρέπει να καλυφθεί με υλικό (οπής).



Εικόνα 5. a) Δυο δυνάμεις b) οπή

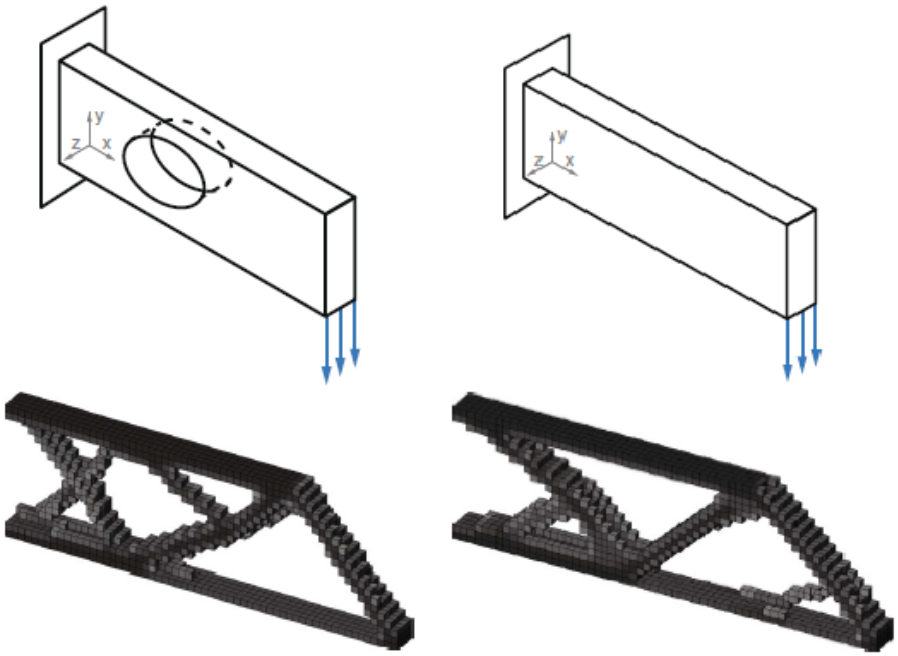
Τέλος αναφέρεται και η χρήση άλλου κριτηρίου βελτιστοποίησης, σε περίπτωση πολλαπλών δεσμεύσεων, εκτός της Optimality Criteria, την οποία θα αναπτύξουμε στο επόμενο κεφάλαιο.

Μετά από εννιά χρόνια, το 2010, οι Erik Andreassen, Anders Clausen, Mattias Schevenels, Boyan S. Lazarov και O. Sigmund δημοσίευσαν ένα άρθρο, στο οποίο παρουσίαζαν έναν κώδικα 88 σειρών[1] σε matlab, βασισμένο στον κώδικα 99 σειρών του O. Sigmund. Για να κάνουν τον κώδικα μικρότερο και γρηγορότερο αξιοποίησαν τις δυνατότητες που δίνονται από τη matlab για τη δημιουργία πινάκων. Επίσης άλλαξαν τον τρόπο που το πρόγραμμα λαμβάνει υπόψη του την πυκνότητα της κατασκευής. Το παράδειγμα το οποίο επιλύει ο καινούργιος κώδικας παρέμεινε το ίδιο και στο τέλος της δημοσίευσης δίνονται και πάλι οι τρόποι με τους οποίους μπορούν να συμπεριληφθούν οι πολλαπλές δυνάμεις και οι οπές.

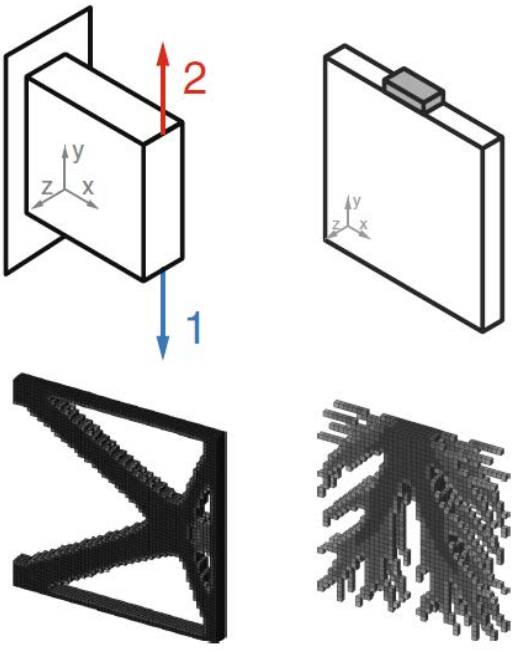
Την ίδια χρονιά, οι G. Allaire και A. Kelly, από την Ecole Polytechnique (X), δημοσίευσαν έναν κώδικα βελτιστοποίησης τοπολογίας σε τρισδιάστατο χώρο με χρήση του λογισμικού πεπερασμένων στοιχείων Free Fem[2]. Το πρόβλημα που επιλύουν είναι η εύρεση του βέλτιστου σχήματος μιας κατασκευής στην οποία ασκείται κατακόρυφη δύναμη προς τα κάτω στο δεξιό της άκρο και είναι πακτωμένη από τα αριστερά.

Το 2013 οι Kai Liu και Andres Tovar δημοσίευσαν ένα άρθρο, στο οποίο παρουσίασαν έναν κώδικα στη matlab για την επίλυση προβλημάτων βελτιστοποίησης τοπολογίας σε τρεις διαστάσεις[13]. Στο άρθρο αρχικά αναλύεται το θεωρητικό υπόβαθρο και οι αλλαγές που θεωρήθηκαν απαραίτητες για την ομαλή λειτουργία του κώδικα σε σχέση με τους κώδικες που έτρεχαν σε δυο διαστάσεις. Έπειτα εξηγείται, όπως και στους παραπάνω κώδικες, ο τρόπος που γράφτηκε ο κώδικας και πώς λειτουργεί. Στη συνέχεια αναλύονται και άλλες

μέθοδοι βελτιστοποίησης, όπως η SQP και η MMA. Τέλος, η δημοσίευση περιέχει τη θεωρία και την εφαρμογή στον κώδικα για την περίπτωση που η φόρτιση είναι θερμική και επιβάλλεται σε όλη την επιφάνεια του φορέα καθώς και την περίπτωση πολλαπλής δύναμης και οπής.



Εικόνα 6. a) Βελτιστοποίηση με οπή στην δοκό b) Αρχικό πρόβλημα βελτιστοποίησης σε τρεις διαστάσεις



Εικόνα 7. a) Περίπτωση δύο δυνάμεων b) Περίπτωση θερμικού φορτίου

2.5 Αναφορές σε κατασκευές που για την κατασκευή τους χρησιμοποιήθηκε η βελτιστοποίηση τοπολογίας

Μια από τις εφαρμογές της βελτιστοποίησης τοπολογίας και ίσως η πιο γνωστή είναι το στέγαστρο που κρατάει ένα κομμάτι της οροφής του National Convention Center στη Doha στο Qatar. Το στέγαστρο σχεδιάστηκε από τον Ιάπωνα αρχιτέκτονα Arata Isozaki. Η βασική ιδέα είναι τα μέλη που στηρίζεται η οροφή να έχουν μια δενδροειδή μορφή, εμπνευσμένη από το ιερό ισλαμικό δένδρο, το Sidrat Al-Muntaha. Για αυτό το στέγαστρο ο Isozaki βραβεύτηκε με το RIBA gold Metal το 1986.

Εικόνα 8. Στέγαστρο του National Convention Center

ΚΕΦΑΛΑΙΟ 3

ΘΕΩΡΗΤΙΚΟ ΥΠΟΒΑΘΡΟ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗΣ ΤΟΠΟΛΟΓΙΑΣ

3.1 Εισαγωγή

Σε αυτό το κεφάλαιο γίνεται αναφορά στο θεωρητικό κομμάτι που απαιτείται για την κατανόηση του προβλήματος της βελτιστοποίησης κατασκευών. Αρχικά αναφέρονται οι γενικές έννοιες που συναντώνται στον τομέα της βελτιστοποίησης καθώς και οι πιο διαδεδομένες μέθοδοι βελτιστοποίησης. Στη συνέχεια διατυπώνεται το πρόβλημα της βελτιστοποίησης των κατασκευών και αναφέρονται οι κατηγορίες στις οποίες αυτά τα προβλήματα χωρίζονται καθώς και το βασικό θεωρητικό υπόβαθρο που απαιτείται για την κατανόηση των μεθόδων που χρησιμοποιούνται από τους αλγορίθμους με τους οποίους ασχολείται αυτή η διπλωματική. Τέλος αναπτύσσονται μέθοδοι βελτιστοποίησης συνεχούς μέσου οι οποίες είναι η Optimality Criteria και η Gradient Descent.

3.2 Βασικές έννοιες του τομέα της Βελτιστοποίησης

3.2.1 Βασικές συναρτήσεις

Προκειμένου να μελετήσουμε το φαινόμενο της βελτιστοποίησης είναι σκόπιμο να αναφέρουμε ένα απλό παράδειγμα το οποίο θα βοηθήσει στην κατανόηση της μορφής του προβλήματος και των βασικών συναρτήσεων που συναντώνται στα προβλήματα βελτιστοποίησης[14].

Ας θεωρήσουμε ένα κυλινδρικό δοχείο με ύψος h και διάμετρο d. Το ύψος δεν πρέπει να ξεπερνά τα 10m και η διάμετρος τα 4m. Επίσης θέλουμε ο λόγος του ύψους προς διάμετρο να είναι μεταξύ 1-3 και ο όγκος του κυλίνδρου να είναι 50m^3. Εμείς επιθυμούμε να υπολογίσουμε τις διαστάσεις του κυλίνδρου έτσι ώστε να ελαχιστοποιείται το κόστος αγοράς c το οποίο θεωρούμε ανάλογο με τον όγκο του κυλίνδρου μέσω της σχέσης

$$c = kV \qquad (0.0.a)$$

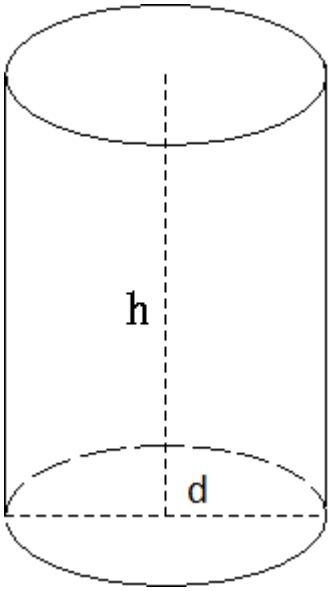
όπου k το κόστος του υλικού κατασκευής ανά μονάδα όγκου και V ο όγκος.

Ο όγκος του κυλίνδρου δίνεται από τη σχέση

$$V = \left(\frac{\pi d^2}{4} \right) h \qquad (0.1.a)$$

επομένως το κόστος αγοράς c του κυλίνδρου συναρτήσει των διαστάσεών του δίνεται, αν αντικαταστήσουμε την (0.1.a) στην (0.0.a), από

$$c = \left(\frac{\pi d^2}{4} \right) kh \qquad (0.2.a)$$



Εικόνα 9. κύλινδρος με διάμετρο d και ύψος h.

Επομένως το πρόβλημα τώρα μπορεί να γραφεί ως

$$\min_{d,h} c(x) = \left(\frac{\pi d^2}{4} \right) kh \qquad (0.3.a)$$

που υπόκεινται στους παρακάτω περιορισμούς

$$1 \le \frac{h}{d} \le 3 \qquad (0.3.b)$$

$$0 \le h \qquad (0.3.c)$$

$$h \leq 10 \tag{0.3.d}$$

$$0 \leq d \tag{0.3.e}$$

$$d \leq 4 \tag{0.3.f}$$

$$\left(\frac{\pi d^2}{4}\right) h = 50 \tag{0.3.g}$$

Παίρνοντας τις μεταβλητές από τη μια πλευρά των ισοτήτων και ανισοτήτων στις συναρτήσεις περιορισμού παίρνουμε την τελική μορφή

$$\min_{d,h}\left(\frac{\pi d^2}{4}\right) kh \tag{0.4.a}$$

s.t. (subject to)

$$\left(\frac{\pi d^2}{4}\right) h - 50 = 0 \tag{0.4.b}$$

$$h - 3d \leq 0 \tag{0.4.c}$$

$$d - h \leq 0 \tag{0.4.d}$$

$$h - 10 \leq 0 \tag{0.4.e}$$

$$-h \leq 0 \tag{0.4.f}$$

$$d - 4 \leq 0 \tag{0.4.g}$$

$$-d \leq 0 \tag{0.4.h}$$

Αυτή είναι η μορφή ενός προβλήματος βελτιστοποίησης. Οι μεταβλητές του ύψους h και της διαμέτρου d καλούνται μεταβλητές βελτιστοποίησης. Οι μεταβλητές βελτιστοποίησης είναι τα μεγέθη τα οποία, ανάλογα με την τιμή που παίρνουν, καθορίζουν - αλλάζουν τα υπόλοιπα μεγέθη βελτιστοποίησης. Ένα πρόβλημα βελτιστοποίησης χαρακτηρίζεται από την ύπαρξη αρκετών μεταβλητών οι οποίες συνήθως τοποθετούνται σε πίνακα και συμβολίζονται με x.

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \tag{0.5.a}$$

Στην περίπτωσή μας ο πίνακας παίρνει την μορφή

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} h \\ d \end{bmatrix} \tag{0.6.a}$$

Η συνάρτηση (0.4.a) καλείται συνάρτηση βελτιστοποίησης ή αντικειμενική συνάρτηση (objective function) και είναι η συνάρτηση που θέλουμε να βελτιστοποιήσουμε. Η αντικειμενική συνάρτηση δημιουργείται έτσι ώστε να εξαρτάται πλήρως από τις μεταβλητές βελτιστοποίησης και συμβολίζεται συνήθως με f.

$$f(x) = \left(\frac{\pi x_2^2}{4} \right) k x_1 \tag{0.6.b}$$

Οι περιορισμοί της μορφής της σχέσης (0.4.b) καλούνται ισοτικοί περιορισμοί (equality constraints). Οι ισοτικοί περιορισμοί συνήθως συμβολίζονται με h ή g.

$$h(x) = \left(\frac{\pi d^2}{4} \right) h - 50 = 0 \tag{0.6.c}$$

Οι περιορισμοί της μορφής των σχέσεων (0.4.c) έως (0.4.h) ονομάζονται ανισοτικοί περιορισμοί (inequality constraints) και συνήθως συμβολίζονται με g. Αν τους περιορισμούς που έχουν μόνο μια από τις μεταβλητές d και h τους γράψουμε στη μορφή:

$$x_k^L \le x_k \le x_k^U, k = 1, 2, \ldots, n$$

όπου L είναι το lower bound και U είναι το Upper bound, και τους υπόλοιπους περιορισμούς τους γράψουμε στην μορφή:

$$h_i(\mathrm{x}) = 0, i = 1, 2, \ldots, m$$
$$g_j(x) \le 0, \mathrm{j} = 1, 2, \ldots, \mathrm{p}$$

έχουμε την τελική μορφή ενός προβλήματος βελτιστοποίησης:

$$\begin{aligned} &\min_x f(x) \\ &\text{s.t.} \\ &\mathrm{h}_i(\mathrm{x}) = 0, i = 1,2,...,\mathrm{m} \\ &g_j(x) \le 0,\ j = 1,2,...,\mathrm{p} \\ &\mathrm{x}_k^L \le x_k \le x_k^U, k = 1,2,...,\mathrm{n} \end{aligned} \qquad (0.7.a)$$

Έτσι, με τη χρήση του παραδείγματος, καταλήξαμε στη γενική μορφή ενός προβλήματος βελτιστοποίησης. Ανάλογα με την ύπαρξη ή μη κάποιων περιορισμών το πρόβλημα βελτιστοποίησης μπορεί να χαρακτηριστεί:

- Πρόβλημα βελτιστοποίησης χωρίς περιορισμούς

Όταν δεν έχουμε περιορισμούς στην αντικειμενική συνάρτηση

$$\begin{aligned} &\min_x f(x) \\ &x \in R^n \end{aligned}$$

- Πρόβλημα βελτιστοποίησης με ισοτικούς περιορισμούς

$$\begin{aligned} &\min_x f(x) \\ &s.t. \\ &h(x) = 0 \end{aligned}$$

- Πρόβλημα βελτιστοποίησης με ισοτικούς και ανισοτικούς περιορισμούς

$$\begin{aligned} &\min_x f(x) \\ &s.t. \\ &h(x) = 0 \\ &g(x) \le 0 \end{aligned}$$

3.2.2 Μέθοδοι επίλυσης

Για την επίλυση των παραπάνω προβλημάτων βελτιστοποίησης έχουν αναπτυχθεί διάφοροι αλγόριθμοι βελτιστοποίησης. Οι αλγόριθμοι αυτοί μπορούν να χωριστούν σε δύο μεγάλες κατηγορίες:

- Μαθηματικές ή αιτιοκρατικές μέθοδοι βελτιστοποίησης:

- Γραμμικός Προγραμματισμός

Αντιμετωπίζει προβλήματα βελτιστοποίησης στα οποία η αντικειμενική συνάρτηση καθώς και οι συναρτήσεις περιορισμών είναι γραμμικές συναρτήσεις.

- Μη Γραμμικός Προγραμματισμός
  Αντιμετωπίζει προβλήματα βελτιστοποίησης στα οποία η αντικειμενική συνάρτηση ή οι συναρτήσεις περιορισμών ή και οι δύο δεν αποτελούν γραμμικές συναρτήσεις. Τέτοια προβλήματα αποτελούν την πλειοψηφία των προβλημάτων βελτιστοποίησης κατασκευών.
- Ακέραιος Προγραμματισμός
  Εφαρμόζεται στις περιπτώσεις που κάποια ή και όλες οι μεταβλητές βελτιστοποίησης αποτελούνται από διακριτούς ακεραίους αριθμούς.
- Γεωμετρικός Προγραμματισμός
  Εφαρμόζεται όταν η αντικειμενική συνάρτηση και οι συναρτήσεις περιορισμού είναι πολυωνυμικές συναρτήσεις.
- Δυναμικός Προγραμματισμός
  Εφαρμόζεται στην περίπτωση μεγάλων και περίπλοκων προβλημάτων βελτιστοποίησης και έχει ως στόχο την διάσπασή τους σε μικρότερα για να γίνει ευκολότερη η επίλυση τους.

- Εξελικτικοί αλγόριθμοι (Evolutionary algorithms) βελτιστοποίησης:

Οι μέθοδοι αυτές χρησιμοποιούν μηχανισμούς εμπνευσμένους από εξελικτικές διαδικασίες της φύσης όπως είναι η αναπαραγωγή, η μετάλλαξη και η φυσική επιλογή. Όλες οι λύσεις του προβλήματος βελτιστοποίησης παίζουν το ρόλο μελών ενός πληθυσμού και η αντικειμενική συνάρτηση αντιπροσωπεύει για κάθε μια από αυτές τις λύσεις πόσο κοντά βρίσκονται στο ζητούμενο αποτέλεσμα του προβλήματος. Οι διάφοροι τύποι των εξελικτικών αλγορίθμων είναι:

- Γενετικοί Αλγόριθμοι (Genetic Algorithms)
- Γενετικός προγραμματισμός (Genetic Programming)
- Εξελικτικός Προγραμματισμός (Evolutionary Programming)
- Προγραμματισμός της Γονιδιακής Έκφρασης (Gene Expression Programming)
- Εξέλιξη της Στρατηγικής (Evolution Strategy)
- Διαφορική Εξέλιξη (Differential Evolution)
- Νευροεξέλιξη (Neuroevolution)
- (Lerning Classifier System)

3.3 Μαθηματικό υπόβαθρο βελτιστοποίησης κατασκευών

Ως κατασκευή μπορούμε να σκεφτούμε ένα σύνολο υλικού το οποίο θέλουμε να μπορεί να παραλάβει ορισμένα φορτία. Η βελτιστοποίηση μιας κατασκευής είναι η μόρφωση της κατασκευής έτσι ώστε να μπορεί να παραλάβει τα φορτία όσο το δυνατόν καλύτερα. Όμως, ανάλογα με το πρόβλημα που θέλουμε να μπορεί να αντιμετωπίσει η κατασκευή, η ερμηνεία του "καλύτερα" αλλάζει. Ορισμένες κατασκευές πρέπει να είναι ελαφριές, οπότε στόχος της βελτιστοποίησης είναι η ελαχιστοποίηση του βάρους. Σε άλλες κατασκευές επιδιώκουμε να μεγιστοποιήσουμε τη στιβαρότητα ή να ελαχιστοποιήσουμε το κόστος ή τη συνολική μετακίνηση. Ανάλογα με το πρόβλημα οποιοδήποτε από τα παραπάνω μεγέθη μπορεί να θεωρηθεί ως αντικειμενική συνάρτηση της κατασκευής που βελτιστοποιούμε.

3.3.1 Μαθηματική αποτύπωση προβλήματος βελτιστοποίησης κατασκευών

Η μαθηματική μορφή ενός προβλήματος βελτιστοποίησης είναι η παρακάτω

Minimize f(x,y)

Subject to

περιορισμός σχεδιασμού στο x

περιορισμός συμπεριφοράς στο y

Εξίσωση ισορροπίας

- Αντικειμενική συνάρτηση (objective function) f(x,y)
  Η αντικειμενική συνάρτηση παίρνει την τιμή του μεγέθους που θέλουμε να βελτιστοποιήσουμε στην κατασκευή. Μπορεί να μετράει μετακίνηση, ενέργεια, βάρος, δυσκαμψία, τάση ή και κόστος κατασκευής.
- Μεταβλητές σχεδιασμού (design variables) x
  Οι μεταβλητές σχεδιασμού είναι τα μεγέθη τα οποία θα αλλάζουν κατά τη διάρκεια της βελτιστοποίησης. Ως μεταβλητές σχεδιασμού μπορούμε να θεωρήσουμε γεωμετρικά χαρακτηριστικά της κατασκευής, το πάχος ενός στοιχείου στην κατασκευή, τις ράβδους ενός δικτυώματος ή και τη διάμετρο των ράβδων αυτών.
- Μεταβλητή κατάστασης (state variable) y
  Η μεταβλητή κατάστασης μιας κατασκευής είναι μια συνάρτηση που εκφράζει την απόκριση της κατασκευής στις μεταβολές των μεταβλητών

σχεδιασμού. Η πλέον συνήθης μορφή της μεταβλητής κατάστασης είναι η μετακίνηση που προκύπτει από την εξίσωση ισορροπίας.
Οι περιορισμοί χωρίζονται σε τρεις κατηγορίες:

- Περιορισμοί σχεδιασμού (design constraints)
  Οι περιορισμοί αυτοί αφορούν τις μεταβλητές σχεδιασμού x.
- Περιορισμοί συμπεριφοράς (behavioral constraints)
  Οι περιορισμοί αυτοί αφορούν τις μεταβλητές κατάστασης y.
- Εξίσωση ισορροπίας (equilibrium constraint)
  Αυτό ο περιορισμός παίρνει τη μορφή της εξίσωσης ισορροπίας
  K(x)*u=F(x),
  όπου K(x) είναι το μητρώο δυσκαμψίας της κατασκευής που ανάλογα με την μέθοδο συσχετίζεται με τις μεταβλητές σχεδιασμού, u είναι οι μετακινήσεις της κατασκευής και, αφού αυτές δείχνουν την απόκριση της κατασκευής, συνήθως y=u.
  Τέλος F(x) είναι οι δράσεις οι οποίες μπορούν να εξαρτώνται από τις μεταβλητές σχεδιασμού αν το πρόβλημα μας το επιβάλλει.

Αν θεωρήσουμε το y=u και $u(x)=K(x)^{-1}*F(x)$ τότε το πρόβλημα βελτιστοποίησης μπορεί να πάρει την μορφή

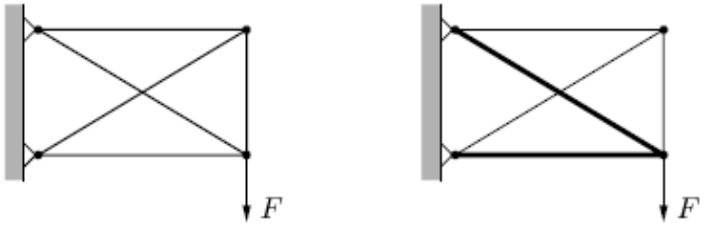
$$\min F(x, u(x))$$
$$s.t.$$
$$g(x, u(x)) < 0$$

Στην οποία το u(x) αποκτάτε από την εξίσωση ισορροπίας και στο g(x,u(x)) συμπεριλαμβάνονται όλοι οι περιορισμοί. Αυτή η μορφή καλείται nested formulation.

3.3.2 Κατηγοριοποίηση Προβλημάτων Βελτιστοποίησης Κατασκευών

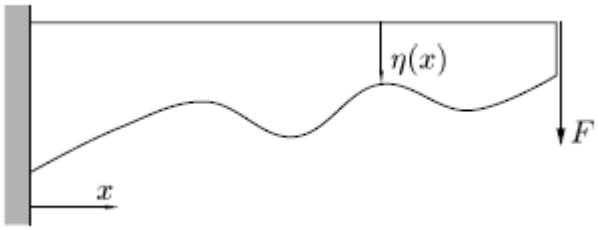
Λαμβάνοντας την μεταβλητή x ως γεωμετρικό παράγοντα της κατασκευής μπορούμε να χωρίσουμε τα προβλήματα σε τρεις κατηγορίες.

- Βελτιστοποίηση Μεγέθους (Sizing Optimization)
  Σε αυτή την κατηγορία οι μεταβλητές x παίρνουν τις τιμές των διαστάσεων των διατομών των δικτυωμάτων ή το πάχος της διατομής πλακών.



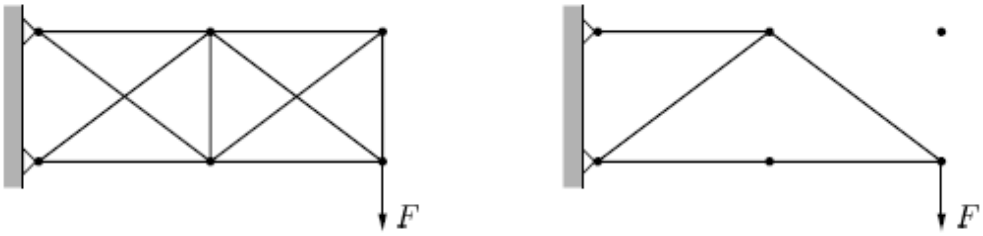
Εικόνα 10. Παράδειγμα βελτιστοποίησης μεγέθους σε πρόβλημα δικτυώματος

- Βελτιστοποίηση Σχήματος (Shape Optimization)
  Εδώ οι μεταβλητές x εκπροσωπούν το μορφή του σχήματος μιας συνεχόμενης κατασκευής μέσα σε ένα δεδομένο χώρο.



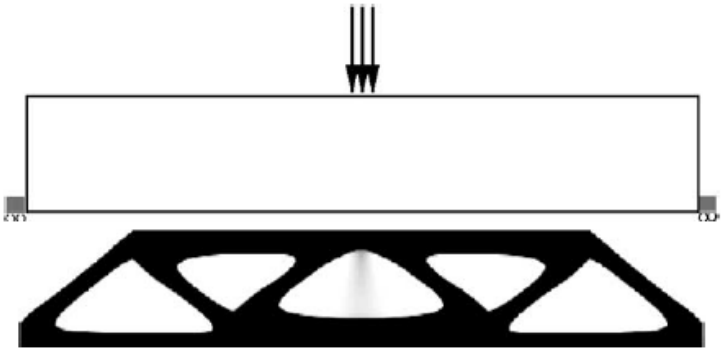
Εικόνα 11. Παράδειγμα βελτιστοποίηση σχήματος. Εδώ πρέπει να βρεθεί η μορφή της συνάρτησης η που θα καθορίσει το σχήμα του προβόλου.

- Βελτιστοποίηση Τοπολογίας (Topology Optimization)
  Είναι η πλέον συνήθης μορφή βελτιστοποίησης κατασκευών. Σε αυτή η μεταβλητές σχεδιασμού μπορούν να πάρουν την τιμή 0 και έτσι να αφαιρεθεί υλικό από τον καθορισμένο χώρο προκειμένου να ικανοποιηθούν οι απαιτήσεις του προβλήματος.
  Στην περίπτωση δικτυώματος οι μεταβλητές σχεδιασμού που αντιπροσωπεύουν τις διατομές των ράβδων μπορούν να πάρουν την τιμή 0 και να αφαιρεθούν ράβδοι που δεν χρειάζονται.



Εικόνα 12. Το εμβαδόν των διατομών των ράβδων που δεν χρειάζονται παίρνει την τιμή 0.

Στην περίπτωση κατασκευής συνεχόμενου υλικού το πάχος ορισμένων σημείων του υλικού μπορεί να πάρει την τιμή 0 οπότε ουσιαστικά να αφαιρεθεί από τον αρχικό χώρο.



Εικόνα 13. Μηδενίζοντας το πάχος ορισμένων περιοχών παίρνουμε το βέλτιστο σχήμα μιας τέτοιας κατασκευής της οποίας το τελικό εμβαδόν είναι το μισό του αρχικού.

Ανάλογα με τη μορφή των μεταβλητών x το πρόβλημα βελτιστοποίησης μπορεί να χωριστεί σε δύο κατηγορίες.

- Διακριτά παραμετρικά συστήματα (discrete parameter systems)
  Τα διακριτά συστήματα έχουν έναν πεπερασμένο αριθμό μεταβλητών. Ένα παράδειγμα διακριτών συστημάτων είναι τα δικτυώματα όπως στην εικόνα 4.
- Κατανεμημένα παραμετρικά συστήματα (distributed parameter systems)
  Αυτά τα συστήματα έχουν τη μορφή ενός συνεχόμενου μέσου όπως είναι το σχήμα της εικόνας 3. Τέτοια προβλήματα δύσκολα λύνονται με αναλυτικές μεθόδους για αυτό έχουν αναπτυχθεί διάφορες μέθοδοι διακριτοποίησης τους. Μια τέτοια μέθοδος είναι η εφαρμογή των πεπερασμένων στοιχείων. Τα προβλήματα βελτιστοποίησης σχήματος αποτελούν τέτοια συστήματα καθώς και κάποια προβλήματα βελτιστοποίησης τοπολογίας.

3.3.3 Κυρτός προγραμματισμός και βασικοί ορισμοί

Συνεχίζοντας αναφέρουμε τις βασικές αρχές του κυρτού προγραμματισμού.

3.3.3.1 Τοπικό και ολικό ελάχιστο

Ας θεωρήσουμε το πρόβλημα βελτιστοποίησης τοπολογίας της μορφής

$$\begin{aligned} &\min_{x} f(x) \\ &s.t. \\ &g_i(x) \le 0 \\ &x_j^{\min} \le x_j \le x_j^{\max} \end{aligned} \qquad \text{(0.8.a)}$$

Τότε μπορούμε να ορίσουμε ως feasible point οποιοδήποτε x το οποίο ικανοποιεί τους περιορισμούς της βελτιστοποίησης. Επομένως για να λύσουμε το πρόβλημα πρέπει να βρούμε x^* τέτοιο ώστε:

$$f(x^*) \le f(x)$$

για κάθε feasible point x. Το x^* καλείται ολικό ελάχιστο (global minimum) της f(x). Αν τώρα η συνάρτηση f παίρνει μεγαλύτερες τιμές από το $f(x^*)$ μόνο σε μια περιοχή γύρω από τον x^* τότε το σημείο x^* καλείται τοπικό ελάχιστο (local minimum).

Διάνυσμα κλίσης συνάρτησης (gradient)

Για μια συνάρτηση f(x), όπου το x μπορεί να είναι ένας πίνακας με μεταβλητές, το διάνυσμα κλίσης της είναι:

$$\nabla f(x) = \begin{pmatrix} \frac{\partial f(x)}{dx_1} \\ \frac{\partial f(x)}{dx_2} \\ \vdots \\ \frac{\partial f(x)}{dx_n} \end{pmatrix} = \begin{bmatrix} \frac{\partial f(x)}{dx_1} & \frac{\partial f(x)}{dx_2} & \cdots & \frac{\partial f(x)}{dx_n} \end{bmatrix}^T$$

Τα σημεία στα οποία η παράγωγος της συνάρτησης μηδενίζεται ονομάζονται στάσιμα σημεία (stationary points) και αποτελούν σημεία τοπικών ελαχίστων ή μεγίστων. Σε ένα πρόβλημα με περιορισμούς όπως το (0.8.a) τα στάσιμα σημεία μπορεί να βρίσκονται εκτός των περιορισμών οπότε το ελάχιστο να είναι στο όριο των περιορισμών.

Ορισμός κυρτού συνόλου και κυρτής συνάρτησης

Ένα σύνολο Α είναι κυρτό αν $\forall x_1, x_2 \in A$ και $\forall \lambda \in (0,1)$ ισχύει

$$\lambda x_1 + (1-\lambda) x_2 \in A$$

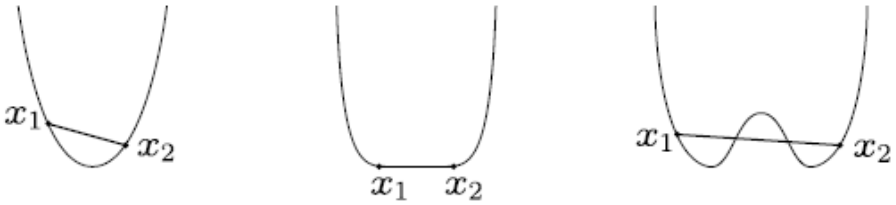
Δηλαδή ένα σύνολο είναι κυρτό όταν όλα τα σημεία της γραμμής που ενώνει δυο τυχαία σημεία x_1,x_2 βρίσκονται εντός του συνόλου.

Εικόνα 14. a) Κυρτό σύνολο b) Μη κυρτό σύνολο

Μια συνάρτηση f:A→B είναι κυρτή (convex) συνάρτηση αν $\forall x_1, x_2 \in A$ με $x_1 \neq x_2$ και $\forall \lambda \in (0,1)$ ισχύει

$$f(\lambda x_1 + (1-\lambda) x_2) \leq \lambda f(x_1) + (1-\lambda) f(x_2)$$

Αν αντί για ≤ έχουμε < τότε η συνάρτηση καλείται αυστηρά κυρτή (strictly convex). Στο σχήμα παρακάτω παρουσιάζονται μια αυστηρά κυρτή, μια κυρτή και μια μη κυρτή συνάρτηση.



Εικόνα 15. a) Αυστηρά κυρτή συνάρτηση b) Κυρτή συνάρτηση c) Μη κυρτή συνάρτηση

3.3.3.2 Ελάχιστα στην περίπτωση κυρτών συναρτήσεων

Σε ένα πρόβλημα βελτιστοποίησης συχνά δημιουργείται το πρόβλημα να εγκλωβιστεί η μέθοδος εύρεσης του ελαχίστου σε κάποιο τοπικό ελάχιστο το οποίο διαφέρει πολύ από το ολικό. Όμως στην περίπτωση που η αντικειμενική συνάρτηση είναι κυρτή και ορίζεται πάνω σε κυρτό σύνολο τότε το τοπικό ελάχιστο είναι ολικό ελάχιστο.

3.3.3.3 Υπολογισμός ελαχίστων κυρτών συναρτήσεων με περιορισμούς

Ας θεωρήσουμε το πρόβλημα

$$\min_x f(x)$$
$$s.t.$$
$$g_i(x) \le 0$$
$$x_j^{\min} \le x_j \le x_j^{\max}$$

Για να υπολογίσουμε το ελάχιστο της συνάρτησης f μέσα στους περιορισμούς, ορίζουμε την συνάρτηση του Λαγκράνζ (Lagrangian fuction).

$$L(x,\lambda) = f(x) + \sum_i \lambda_i g_i(x) \qquad (0.9.a)$$

όπου το λ ονομάζεται πολλαπλασιαστής Λαγκράνζ (Lagrange multiplier).

Το ελάχιστο της συνάρτησης του Λαγκρλανζ είναι το ελάχιστο της συνάρτησης f μέσα στους περιορισμούς.

Παρακάτω θα αναπτυχθούν δύο τρόποι υπολογισμού του ελαχίστου της συνάρτησης του Λαγράνζ.

KKT Conditions

Το ελάχιστο της συνάρτησης του Λαγκράνζ είναι ο συνδυασμός (x,λ) εκείνος που ικανοποιεί τις συνθήκες των Karush-Kuhn-Tucker (KKT conditions). Οι KKT συνθήκες είναι οι εξής:

$$\frac{\partial L(\mathrm{x},\lambda)}{\partial x_j} \le 0 \quad \text{if} \qquad x_j = x_j^{\max}$$

$$\frac{\partial L(\mathrm{x},\lambda)}{\partial x_j} = 0 \quad \text{if} \qquad x_j^{\min} \le x_j \le x_j^{\max}$$

$$\frac{\partial L(\mathrm{x},\lambda)}{\partial x_j} \ge 0 \quad \text{if} \qquad x_j = x_j^{\min}$$

$$\lambda_i g_i(x) = 0$$

$$g_i(x) \le 0$$

$$\lambda_i \ge 0$$

$$x_j^{\min} \le x_j \le x_j^{\max}$$

Όπου

$$\frac{\partial L(\mathrm{x},\lambda)}{\partial x_j} = \frac{\partial g(x)}{\partial x_j} + \sum_i \lambda_i \frac{\partial g_i(x)}{\partial x_j}$$

Lagrangian Duality

Για να βρούμε το ελάχιστο της συνάρτησης L αρκεί αρχικά να βρούμε το μέγιστο της L ως προς λ για λ>0 και μετά για το λ που θα προκύψει να βρούμε το ελάχιστο ως προς x.

$$\min_x \max_\lambda L(x,\lambda)$$

3.3.3.4 Υπολογισμός κυρτών προσεγγίσεων για μη κυρτές συναρτήσεις

Οι μέθοδοι που αναφέρθηκαν μας δίνουν το ελάχιστο μιας συνάρτησης f που υπόκειται σε περιορισμούς. Η παραπάνω ανάλυση, όμως, μπορεί να εφαρμοστεί μόνο σε κυρτά προβλήματα. Στην πράξη τα περισσότερα προβλήματα βελτιστοποίησης τοπολογίας είναι μη κυρτά. Για αντιμετωπιστεί αυτό ένας τρόπος είναι να προσεγγιστεί η μη κυρτή συνάρτηση με μια κυρτή. Υπάρχουν διάφορες μέθοδοι που επιτυγχάνουν αυτό το σκοπό. Οι πιο γνωστές είναι η SLP (Sequential Linear Programming), η SQP (Sequential Quadratic Programming), CONLIN (Convex Linearization) και η MMA (Method of Moving Asymptotes).

Παρακάτω θα αναλυθεί η SLP ως παράδειγμα.

Sequential Linear Programming (SLP)

Σκοπός της SLP είναι να δώσει μια κυρτή προσέγγιση σε μια μη κυρτή αντικειμενική συνάρτηση. Η SLP κάνει χρήση της σειράς Taylor που η γενική μορφή της είναι η παρακάτω

$$f(x) = f(x_0) + \frac{\partial f(x_0)}{dx}(\mathrm{x} - x_0) + \frac{1}{2}(\mathrm{x} - x_0)^2 \frac{\partial^2 f(x_0)}{dx^2} + R$$

Όπου R είναι μια μικρή ποσότητα η οποία δίνει την απόκλιση της προσέγγισης. Για την προσέγγιση της SLP χρησιμοποιούνται οι δύο πρώτοι όροι της προσεγγιστικής σειράς. Επομένως η αντικειμενική συνάρτηση και οι συναρτήσεις περιορισμών παίρνουν την μορφή

$$\min_x f(x) \cong \min_x f(x_0) + \frac{\partial f(x_0)}{dx}(\mathrm{x} - x_0)$$

s.t.

$$\mathrm{g}_i(x) = g_i(x_0) + \frac{\partial g_i(x_0)}{dx}(\mathrm{x} - x_0) \le 0$$

$$x_j^{\min} \le x_j \le x_j^{\max}$$

3.3.4 Επίλυση προβλημάτων βελτιστοποίησης κατασκευών με μετατροπή σε κυρτά προβλήματα βελτιστοποίησης.

Όταν έχουμε ένα πρόβλημα βελτιστοποίησης κατασκευών όπως αυτό που αναπτύξαμε παραπάνω

$$\min_x f(x, \mathrm{u(x)})$$
$$s.t.$$
$$g_i(x, u(x)) \le 0$$
$$x_j^{\min} \le x_j \le x_j^{\max}$$

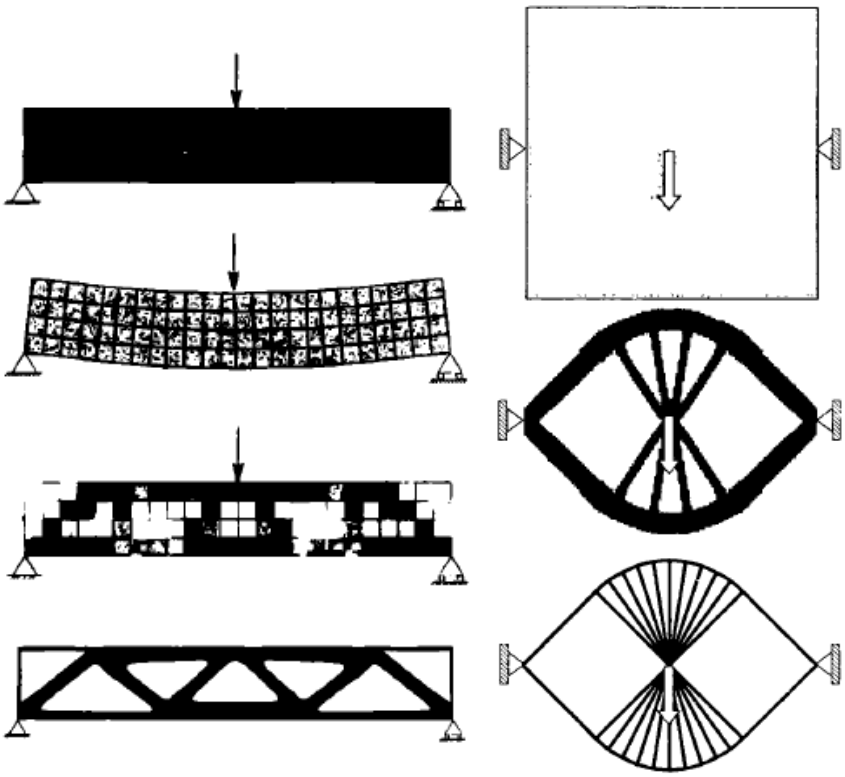
Τότε η γενική μεθοδολογία επίλυσής του ακολουθεί τα παρακάτω βήματα

1. Αρχικά ξεκινάμε με ένα αρχικό σχεδιασμό των μεταβλητών x_0 για να ξεκινήσουμε την πρώτη επανάληψη.
2. Στη συνέχεια με επίλυση της σχέσης $K(x_0)\ u(x_0)=F(x_0)$ βρίσκουμε την τιμή της μεταβλητής $u(x_0)$.
3. Για την x_0 υπολογίζουμε την αντικειμενική συνάρτηση και τις συναρτήσεις περιορισμών καθώς και τις παραγώγους τους.

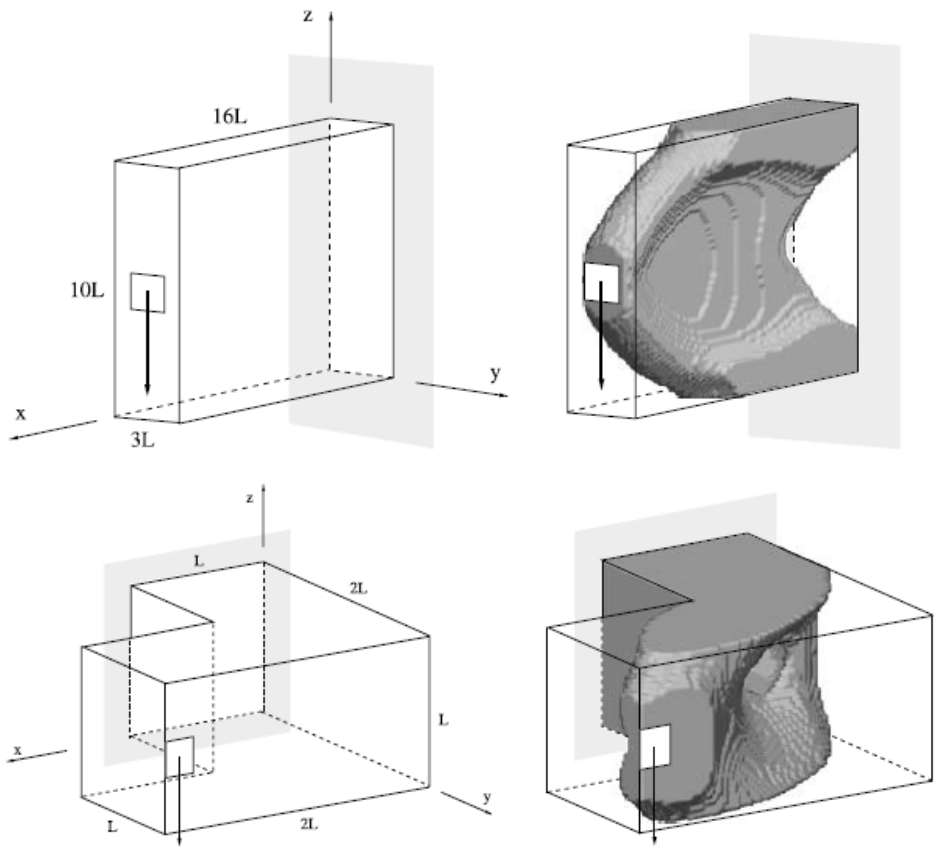
4. Χρησιμοποιούμε μια από τις μεθόδους (SLP,SQP,CONLIN,MMA) μετατροπής των συναρτήσεων σε κυρτές προσεγγίσεις προκειμένου να υπολογίσουμε την $f(x_1)$ αν αυτές δεν είναι κυρτές.
5. Επιλύουμε το πρόβλημα βελτιστοποίησης με χρήση των KKT conditions ή της Lagrangian Duality προκειμένου να καταλήξουμε σε ένα νέο x_1.
6. Συνεχίζουμε τη διαδικασία αυτή μέχρι να ικανοποιηθεί κάποιο κριτήριο το οποίο έχουμε επιλέξει.

3.3.5 Προβλήματα βελτιστοποίησης κατασκευών σε συνεχόμενα συστήματα

Τα προβλήματα τα οποία αποτέλεσαν το αντικείμενο αυτής της διπλωματικής είναι τα συνεχόμενα συστήματα (Distributed Parameter Systems).Σε ένα τέτοιο πρόβλημα βελτιστοποίησης δίνεται μια επιφάνεια (αν αναφερόμαστε σε δύο διαστάσεις) ή ένας χώρος (αν αναφερόμαστε σε τρεις διαστάσεις) στον οποίο ορίζονται τα σημεία στήριξης και οι ασκούμενες δυνάμεις. Για αυτά τα δεδομένα ζητείται να καθοριστεί το σχήμα που πρέπει να λάβει η κατασκευή εντός της επιφάνειας ή του χώρου που της δίνεται για να μπορέσει να παραλάβει τα φορτία με τον καλύτερο τρόπο, χωρίς όμως να ξεπερνά ένα ποσοστό της επιφάνειας ή χώρου που της δίνεται.



Εικόνα 16. Παραδείγματα προβλήματος βελτιστοποίησης κατασκευών συνεχόμενου συστήματος σε δύο διαστάσεις



Εικόνα 17. Παραδείγματα προβλήματος βελτιστοποίησης κατασκευών συνεχόμενου συστήματος σε τρεις διαστάσεις

Για την αντιμετώπιση αυτών των προβλημάτων πρέπει πρώτα να γίνει διακριτοποίησή τους. Αυτό επιτυγχάνεται με την χρήση των πεπερασμένων στοιχείων.

Έχοντας διακριτοποιήσει το σύστημα μπορούμε τώρα να ορίσουμε τι μέγεθος θα παριστάνουν οι μεταβλητές βελτιστοποίησης x. Εφόσον σε ένα τέτοιο πρόβλημα αυτό που μας ενδιαφέρει είναι η ύπαρξη ή μη των διαφόρων πεπερασμένων στοιχείων από τα οποία αποτελείται η επιφάνεια ή ο χώρος μας, ορίζουμε ως x το ποσοστό του πάχους του κάθε ενός πεπερασμένου στοιχείου και το ονομάζουμε πυκνότητα του στοιχείου αυτού. Οι μεταβλητές βελτιστοποίησης x μπορούν να λάβουν τιμές από μηδέν μέχρι ένα.

$$0 \le x_e \le 1$$

Μηδέν σημαίνει ότι το πεπερασμένο στοιχείο δεν έχει πάχος και ένα ότι έχει το πλήρες πάχος του το οποίο είναι ίδιο για όλα.

Η αντικειμενική συνάρτηση f είναι το έργο το οποίο παράγεται από τις δυνάμεις που ασκούνται στην κατασκευή και συμβολίζεται με

$$C = F^T * u$$

όπου το u είναι η μετακίνηση της κατασκευής λόγω της επιβολής των δυνάμεων F. Όμως επειδή F=K*u η σχέση μπορεί να γραφεί και ως:

$$C=u^T * K * u$$

Οι συναρτήσεις περιορισμού παίρνουν την μορφή

$$\int_{\Omega} x d\Omega = \sum_{1}^{n} x_e \alpha = V$$

όπου Ω είναι ο χώρος που μας δίνεται, α είναι ο όγκος ενός πεπερασμένου στοιχείου και V είναι το συνολικό επιτρεπόμενο ποσοστό του όγκου που μπορούμε να έχουμε (με αυτό τον τρόπο καθορίζεται και η επιφάνεια την οποία μπορεί να καλύπτει η τελική κατασκευή).

Επομένως το πρόβλημα παίρνει την μορφή

$$\min_x C(x) = F * u(x) \qquad (0.10.a)$$

s.t.

$$\sum_{1}^{n} x_e \alpha = V$$

$$0 \le x_e \le 1$$

Για να λάβουμε υπόψη μας το πάχος του κάθε ενός στοιχείου στον υπολογισμό του μητρώου δυσκαμψίας του χρησιμοποιούμε την μέθοδο SIMP.

Solid Isotropic Material with Penalization (SIMP)

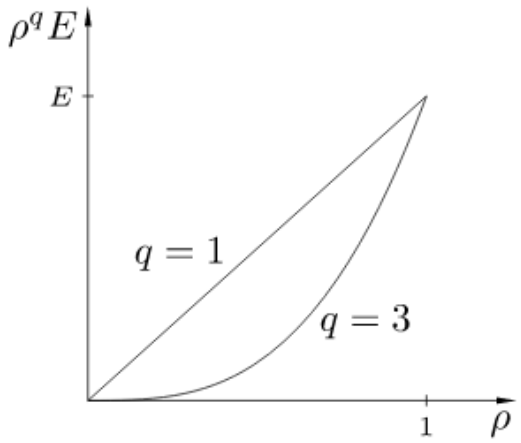
Η μέθοδος SIMP θεωρεί την παρακάτω σχέση μεταξύ του μέτρου ελαστικότητας E και των πυκνοτήτων x ενός πεπερασμένου στοιχείου.

$$E_e(\mathrm{x}_e) = x_e^P E_e$$

Επομένως η σχέση μεταξύ του κλασικού μητρώου δυσκαμψίας (δηλαδή του μητρώου δυσκαμψίας για πυκνότητα ίση με 1) και του τρέχοντος είναι η ακόλουθη

$$K_e = x_e^p K_e^0$$

Όπου K_e^0 είναι το κλασικό μητρώο δυσκαμψίας και p είναι μια δύναμη της οποίας η τιμή εξαρτάται από το πρόβλημα. Η δύναμη αυτή χρησιμοποιείται επειδή στο πρόβλημα βελτιστοποίησης θέλουμε να αποφύγουμε τα x να παίρνουν τιμές ανάμεσα στα μηδέν και ένα. Θέλουμε είτε να έχουμε πλήρες υλικό (1), είτε καθόλου (0). Επομένως με χρήση της δύναμης p αν το x πάρει τιμή ανάμεσα στο μηδέν και ένα τότε ο αριθμός που προκύπτει είναι πολύ μικρός επομένως κάνει το υλικό σε εκείνο το στοιχείο πολύ εύκαμπτο (μικρή δυσκαμψία) οπότε δεν συμφέρει να τοποθετηθεί τμήμα από τον επιτρεπόμενό του όγκο σε τέτοιες τιμές x που δεν συμβάλουν ιδιαίτερα στην συνολική δυσκαμψία.



Εικόνα 18. Τιμές της πυκνότητας υψωμένης στη τρίτη σε σχέση με τη πυκνότητα.

Επιλύουμε το πρόβλημα (0.10.a) χρησιμοποιώντας την μέθοδο OC.

Optimality Criteria Method (OC)

Η OC υπολογίζει τις καινούργιες πυκνότητες γραμμικοποιόντας την αντικειμενική συνάρτηση και ελαχιστοποιώντας στη συνέχεια τον όρο που εξαρτάτε από αυτές.

Αρχικά υπολογίζουμε τη παράγωγο της αντικειμενικής συνάρτησης. Για τον υπολογισμό της θα πρέπει να αφαιρέσουμε από την αντικειμενική συνάρτηση τον όρο

$$C(x) = F * u(x) - \lambda(K(x)u(\mathrm{x}) - \mathrm{F})$$

Αυτό γίνεται ώστε με την παραγώγιση να αποκτήσουμε τη δυνατότητα, επιλέγοντας το κατάλληλο λ, να απαλείψουμε τους όρους που περιέχουν τη παράγωγο των μετακινήσεων ως προς το x. Η παράγωγος du/dx είναι αδύνατον να υπολογιστεί. Συνεπώς με αυτό τον τρόπο έχουμε τη δυνατότητα να υπολογίσουμε το dc/dx χωρίς να ανησυχούμε για το du/dx. Η παράγωγος της συνάρτησης C(x) ως προς το x_e του κάθε στοιχείου είναι ίση με:

$$\frac{\partial C(x)}{\partial x_e} = F\frac{\partial u_e(x\ )}{\partial x_e} - \lambda\frac{\partial K_e(x)}{\partial x_e}u_e(x) - \lambda K_e(x)\frac{\partial u_e(x)}{\partial x_e}$$

Επομένως για να μηδενίσουμε τον όρο της παραγώγου της μετακίνησης φέρνουμε την εξίσωση στη μορφή

$$\frac{\partial C(x)}{\partial x_e} = -\lambda\frac{\partial K_e(x)}{\partial x_e}u_e(x) + (F - \lambda K_e(x))\frac{\partial u_e(x\ )}{\partial x_e} \qquad (0.10.b)$$

επιλέγουμε το λ= u(x). Έτσι η παράσταση μηδενίζεται.

$$F - \lambda K_e(x) = 0$$

Επομένως η εξίσωση (0.10.b) παίρνει την μορφή:

$$\frac{\partial C(x)}{\partial x_e} = -u_e(x)^T\frac{\partial K_e(x)}{\partial x_e}u_e(x) \qquad (0.12.a)$$

Από τη σχέση 12α φαίνεται ότι η παράγωγος έχει πάντα τιμή θετική.

Στη συνέχεια γίνεται η γραμμικοποίηση της αντικειμενικής συνάρτησης. Για τον όρο με τη παράγωγο χρησιμοποιείτε η αλλαγή μεταβλητής:

$$y_e = x_e^{-a}$$

όπου α παίρνει τιμές μεγαλύτερες του μηδενός.

Επομένως εφαρμόζοντας τη γραμμικοποίηση προκύπτει

$$\mathrm{C}(x) = C(x^k) + (\mathrm{y}_e - \mathrm{y}_e^k)\sum_{e=1}^{N}\frac{\partial C(x^k)}{\partial y_e}\bigg|_{x=x^k} = C(x^k) + \mathrm{y}_e\sum_{e=1}^{N}\frac{\partial C(x^k)}{\partial y_e}\bigg|_{x=x^k} - \mathrm{y}_e^k\sum_{e=1}^{N}\frac{\partial C(x^k)}{\partial y_e}\bigg|_{x=x^k}$$

όπου x^k είναι ο πίνακας των προηγούμενων πυκνοτήτων.

Σύμφωνα με τον κανόνα της αλυσίδας

$$\frac{\partial C}{\partial y_e} = \frac{\partial C}{\partial x_e}\frac{\partial x_e}{\partial y_e} = \frac{\partial C}{\partial x_e}\frac{1}{\dfrac{\partial x_e^{-a}}{\partial x_e}} = -\frac{x_e^{1+a}}{a}\frac{\partial C}{\partial x_e}$$

Η συνάρτηση C(x) παίρνει την μορφή

$$\mathrm{C}(x) = C(x^k) + \mathrm{y}_e^k(\frac{(x_e^k)^{1+a}}{a}\frac{\partial C}{\partial x_e}) + \sum_{e=1}^{N} b_e^k x_e^{-a}$$

Όπου

$$b_e^k = \frac{(x_e^k)^{1+a}}{a}\frac{\partial J}{\partial x_e}\bigg|_{x=x^k}$$

Γνωρίζοντας ότι η παράγωγος της αντικειμενικής συνάρτησης παίρνει τιμές μόνο θετικές μπορούμε να συμπεράνουμε ότι:

$$\mathrm{y}_e^k(\frac{(x_e^k)^{1+a}}{a}\frac{\partial C}{\partial x_e}) < 0 \quad \text{και} \quad \sum_{e=1}^{N} b_e^k x_e^{-a} > 0$$

Επομένως για να μεγιστοποιήσουμε τη μείωση της αντικειμενικής συνάρτησης θα πρέπει να ελαχιστοποιήσουμε τον θετικό όρο.

$$\sum_{e=1}^{N} b_e^k x_e^{-a} > 0$$

Με αυτό τον τρόπο θεωρούμε το υποπρόβλημα:

$$\begin{cases} \min\limits_{x} \sum\limits_{e=1}^{n} b_e^k x_e^{-a} \\ s.t. \begin{cases} x * a = V \\ 0 \le x_e \le 1 \end{cases} \end{cases}$$

Για να επιλύσουμε αυτό το πρόβλημα εφαρμόζουμε την Lagrangian duality θεωρώντας την συνάρτηση

$$L(\mathrm{x}, \lambda) = \sum_{e=1}^{n} b_e^k x_e^{-a} + \lambda(\mathrm{x} * \alpha - \mathrm{V})$$

Αρχικά θέλουμε να βρούμε το ελάχιστο ως προς x και μετά το μέγιστο ως προς λ. Για να βρούμε το ελάχιστο ως προς x υπολογίζουμε την παράγωγο

$$\frac{\partial L}{\partial x_e} = -a b_e^k x_e^{-a-1} + \lambda\alpha$$

Στο σημείο όπου η παράγωγος μηδενίζεται έχουμε το ελάχιστο.

Άρα

$$\frac{\partial L}{\partial x_e} = 0 \Leftrightarrow x_e = (\frac{ab_e^k}{\lambda \alpha_e})^{\frac{1}{1+a}}$$

Αν η λύση ξεπερνά τις τιμές των άκρων τότε τοποθετούμε την ακριανή τιμή.

$$x_e = \begin{cases} 0 & \alpha\nu \quad (\frac{ab_e^k}{\lambda \alpha_e})^{\frac{1}{1+a}} < 0 \\ (\frac{ab_e^k}{\lambda \alpha_e})^{\frac{1}{1+a}} & \alpha\nu \quad (\frac{ab_e^k}{\lambda \alpha_e})^{\frac{1}{1+a}} = 0 \\ 1 & \alpha\nu \quad (\frac{ab_e^k}{\lambda \alpha_e})^{\frac{1}{1+a}} > 1 \end{cases}$$

Για να υπολογίσουμε τώρα το μέγιστο ως προς λ πρέπει να παραλογίσουμε την L(x,λ) ως προς λ για το x που βρήκαμε παραπάνω.

$$\frac{\partial L}{\partial \lambda} = \sum_{e=1}^{n} \alpha_e x_e - V \qquad (0.13.a)$$

Η παράγωγος πρέπει να εξισωθεί με μηδέν και αυτό μας δίνει τον αρχικό περιορισμό για τον όγκο. Επομένως για να υπολογιστεί το λ που μεγιστοποιεί την L πρέπει να γίνουν επαναλήψεις στις οποίες θα υπολογίζεται το x για κάθε λ και μετά θα ελέγχεται ο περιορισμός (0.13.a) και αν δεν ικανοποιείτε το λ θα αλλάζει και η διαδικασία θα επαναλαμβάνεται.

Η διαδικασία περιγράφεται περιληπτικά παρακάτω

1. Ξεκινάμε την επαναληπτική διαδικασία με ένα αρχικό x_0 ή x^K από την προηγούμενη επανάληψη.

2. Για αυτό το x υπολογίζουμε το μητρώο των μετακινήσεων u^k από τον τύπο $K(x^K)\, u^k = F$. Για να λάβουμε το μητρώο x υπόψη μας στο μητρώο δυσκαμψία χρησιμοποιούμε την μέθοδο SIMP.

3. Υπολογίζουμε τη παράγωγο της αντικειμενικής συνάρτησης dc/dx.

4. Για να βρομούμε το καινούριο x^{k+1} εφαρμόζουμε τον τύπο

$$x_e^{k+1} = \min\left\{\max\left((\frac{ab_e^k}{\lambda \alpha_e})^{\frac{1}{1+a}}, 0\right), 1\right\}$$

Για λ που ικανοποιεί την σχέση

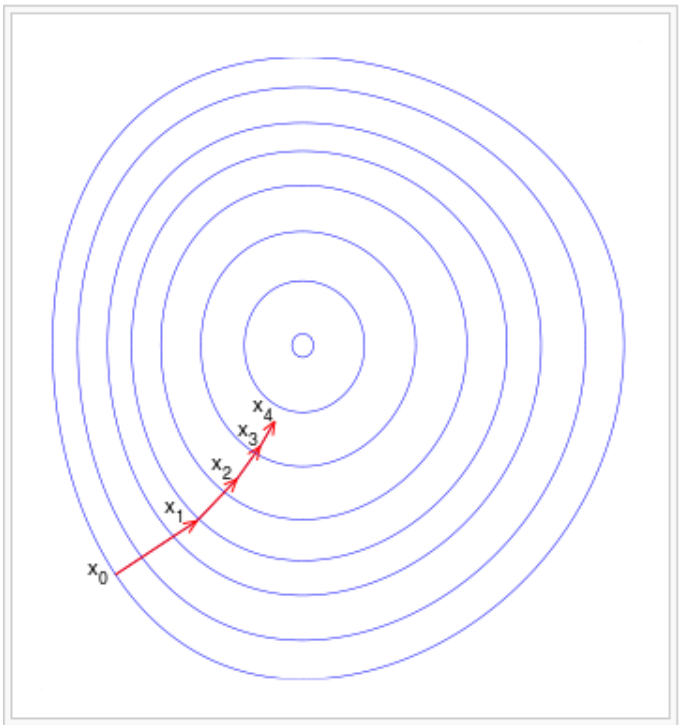
$$\sum_{e=1}^{n} \alpha_e x_e(\lambda) - V = 0$$

Το βήμα 4 είναι μια επαναληπτική διαδικασία για τον προσδιορισμό του κατάλληλου λ.

Παρακάτω αναλύεται η επίλυση του προβλήματος (0.10.a) χρησιμοποιώντας τη μέθοδο Gradient descent.

Gradient Descent Method

Η μέθοδος αυτή βασίζεται στη παρατήρηση ότι όταν βρισκόμαστε σε ένα σημείο a του f(a) η αντικειμενική συνάρτηση μειώνεται προς την κατεύθυνση της αρνητικής παραγώγου της συνάρτησης f.



Εικόνα 19. Μέσο της επαναληπτικής διαδικασίας περνάμε από τα x1¬,x2... και καταλήγουμε στο βέλτιστο.

Επομένως με αυτή την μέθοδο η ελαχιστοποίηση της αντικειμενικής συνάρτησης γίνεται με μια επαναληπτική διαδικασία στην οποία οι μεταβλητές σχεδιασμού κάθε φορά προκύπτουν από τον τύπο:

$$x^{k+1} = x^k - m\frac{\partial J(\mathrm{x}_e)}{\partial x_e}$$

όπου το m είναι βήμα που ορίζεται από τον χρήστη.

Η διαδικασία βελτιστοποίησης του προβλήματος είναι παρόμοια με της Optimality Criteria.

Τα βήματα 1 και 2 και 3 είναι τα ίδια.

Στο βήμα 4 για να βρούμε το καινούριο x^{k+1} εφαρμόζουμε τον τύπο της Gradient Descent.

$$x^{k+1} = x^k - m\frac{\partial J(\mathrm{x}_e)}{\partial x_e}$$

Στα προβλήματα βελτιστοποίησης τοπολογίας ο ορισμός του βήματος m κάθε φορά δεν είναι εύκολος. Επομένως το βήμα ορίζεται στην αρχή αυθαίρετα και μετά από κάθε επανάληψη γίνεται ένας παραπάνω έλεγχος. Συγκρίνουμε τα $C(x^k)$ και $C(x^{k+1})$. Αν το $jC(x^{k+1})$ είναι μεγαλύτερο από το $C(x^k)$ τότε ξανά γυρνάμε πίσω στο x^k απορρίπτοντας το x^{k+1} μικραίνουμε το βήμα και πάμε πάλη στο βήμα 1. Η διαδικασία αυτή συνεχίζεται μέχρι τα x^{k+1} και x^k να έχουν πολύ μικρή διαφορά.

ΚΕΦΑΛΑΙΟ 4

ΚΩΔΙΚΕΣ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗΣ ΤΟΠΟΛΟΓΙΑΣ ΜΕ ΧΡΗΣΗ ΤΗΣ ΜΕΘΟΔΟΥ SIMP

4.1 Εισαγωγή

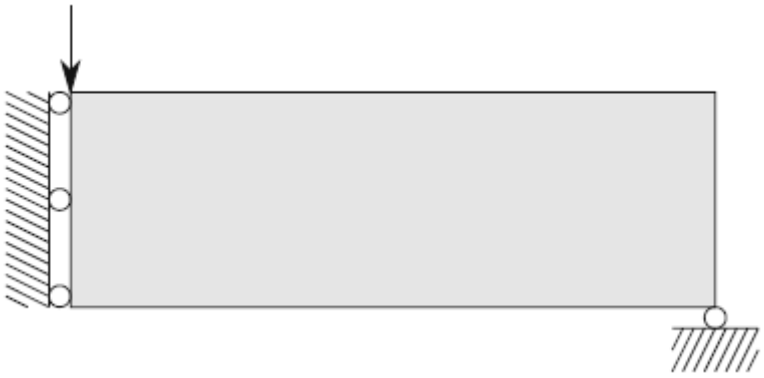
Σε αυτό το κεφάλαιο θα γίνει αναφορά στου προγραμματιστικούς κώδικες που αποτέλεσαν τη βάση για τη διπλωματική εργασία. Ο πρώτος κώδικας αναφέρεται σε προβλήματα βελτιστοποίησης τοπολογίας δύο διαστάσεων. Είναι γραμμένος σε 88 σειρές και αποτελεί βελτιστοποιημένη χρονικά έκδοση του προηγούμενου κώδικα βελτιστοποίησης τοπολογίας που ήταν γραμμένος σε 99 σειρές. Ο δεύτερος κώδικας αναφέρεται σε προβλήματα τριών διαστάσεων και χρησιμοποιεί τις ίδιες τεχνικές βελτιστοποίησης όπως και ο κώδικας 88 γραμμών για να επιλύσει τα προβλήματα αυτά.

4.2 Κώδικας Βελτιστοποίησης Τοπολογίας 88 γραμμών

Ο κώδικας αυτός επιλύει ένα πρόβλημα βελτιστοποίησης τοπολογίας σε 88 γραμμές στον δισδιάστατο χώρο με χρήση matlab. Η βάση του είναι παρμένη από έναν παλαιότερο κώδικα μεγέθους 99 γραμμών. Ο καινούριος κώδικας κάνει χρήση των ιδιοτήτων και εργαλείων της matlab για τη χρήση πινάκων προκειμένου να μειώσει το απαιτούμενο μήκος αλλά και να αυξήσει την ταχύτητα του.

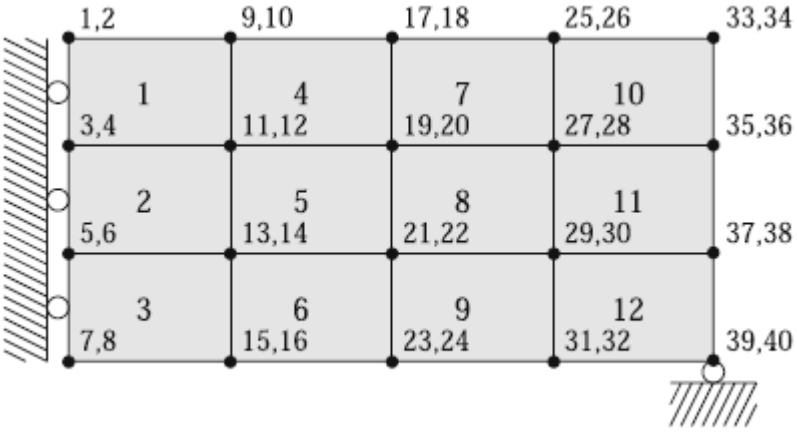
4.2.1 Διαδικασία Επίλυσης Προβλήματος Βελτιστοποίησης

Το πρόβλημα που επιλύεται στον κώδικα είναι μιας δοκού που ασκείται φορτίο στην άνω αριστερά άκρη της όπως στην εικόνα.



Εικόνα 20. Παράδειγμα που βελτιστοποιείτε από τον top88

Το σχήμα χωρίζεται σε τετραγωνικά πεπερασμένα στοιχεία όπου το κάθε ένα έχει πυκνότητα x_e.



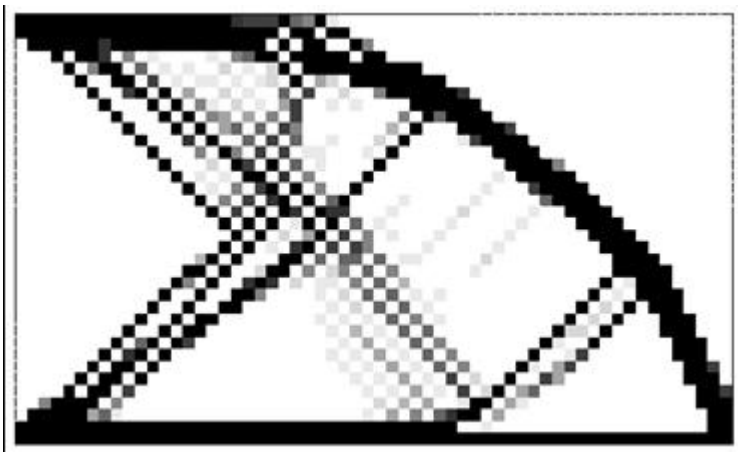
Εικόνα 21. Διακριτοποίηση παραδείγματος

Η μέθοδος που χρησιμοποιείται για να ληφθούν υπόψη οι πυκνότητες x στο μητρώο δυσκαμψίας του κάθε στοιχείου είναι μια τροποποίηση της μεθόδου SIMP. Σύμφωνα με την αλλαγή αυτή οι πυκνότητες x λαμβάνονται υπόψη στο μητρώο δυσκαμψίας του κάθε στοιχείου μέσο του μέτρου ελαστικότητας E από τη σχέση:

$$E_e(\mathrm{x}_e) = \mathrm{E}_{\min} + x_e^P(\mathrm{E}_0 - \mathrm{E}_{\min}) \tag{4.1}$$

όπου E_0 είναι το μέτρο ελαστικότητα του υλικού και E_{min} είναι η ελάχιστη τιμή την οποία μπορεί να πάρει το μέτρο ελαστικότητας σε περιοχές όπου η πυκνότητα γίνεται μηδέν.

Η μέθοδος βελτιστοποίησης που χρησιμοποιείται είναι η Optimality Criteria. Για να αποφευχθεί η κατανομή του υλικού στην μορφή σκακιέρας (checker board problem) όπως στην εικόνα, γίνεται χρήση ενός φίλτρου, το οποίο εξαρτά την κάθε μια πυκνότητα με τις πυκνότητες των στοιχείων γύρω της, τα οποία βρίσκονται εντός ενός κύκλου του οποίου η ακτίνα ορίζεται από τον χρήστη.



Εικόνα 22. Αποτέλεσμα βελτιστοποίησης χωρίς την εφαρμογή του φίλτρου.

Το πρόγραμμα δίνει δύο δυνατότητες. Η πρώτη είναι το φίλτρο να εφαρμοστεί μόνο στην παράγωγο του έργου. Επομένως η φιλτραρισμένη παράγωγος προκύπτει από τον τύπο

$$\frac{\partial C}{\partial x_e} = \frac{\sum_i H_{ei} x_i \frac{\partial C}{\partial x_i}}{\max(\gamma, \mathrm{x}_e) \sum_i H_{ei}} \tag{4.2}$$

Όπου

i είναι το κάθε στοιχείο που βρίσκεται γύρο από το στοιχείο e σε απόσταση μικρότερη από την ακτίνα.

γ είναι μια μικρή τιμή για να αποφύγουμε την διαίρεση με το μηδέν.

H_{ei} δίνεται από τον τύπο $H_{ei} = \max(0, \mathrm{r}_{\min} - \Delta(e,i))$

r_{min} είναι η ακτίνα γύρω από το στοιχείο την οποία ορίζει ο χρήστης και Δ(e,i) είναι η απόσταση από κέντρο σε κέντρο του στοιχείου με το κάθε άλλο στοιχείο γύρω του που βρίσκεται εντός της ακτίνας r_{min}.

Η δεύτερη είναι το φίλτρο να εφαρμοστεί και στις πυκνότητες x μέσω του τύπου

$$x_e = \frac{\sum_i H_{ei} x_i}{\sum_i H_{ei}} \tag{4.3}$$

4.2.2 Αλγόριθμος Επίλυσης Προβλήματος Βελτιστοποίησης σε δύο διαστάσεις

Ο κώδικας είναι γραμμένος μέσα σε μια συνάρτηση που ονομάζεται top88.

```
function top88(nelx,nely,volfrac,penal,rmin,ft)
```

Τα δεδομένα εισόδου της συνάρτησης είναι

- nelx,nely που είναι ο αριθμός των τετραγωνικών πεπερασμένων στοιχείων κατά την x και y διεύθυνση αντίστοιχα.
- volfrac που αποτελεί την αρχική τιμή για τις πυκνότητες x.
- penal που είναι η δύναμη στην οποία υψώνουμε το x στον τύπο
- r_{min} είναι η ακτίνα στην οποία εφαρμόζεται το φίλτρο γύρο από το κάθε στοιχείο.
- ft παίρνει τιμή 1 ή 2 και είναι ο δείκτης για το ποιο φίλτρο θα εφαρμοστεί.

```
%% MATERIAL PROPERTIES
E0 = 1;
Emin = 1e-9;
nu = 0.3;
```

Έπειτα συνεχίζουμε προσδιορίζοντας τις ακόλουθες ιδιότητες του υλικού.

E_0 είναι το μέτρο ελαστικότητας

E_{min} είναι η τιμή του μέτρου ελαστικότητας σε περιοχές όπου η πυκνότητα μηδενίζεται.

nu είναι ο λόγος του Poisson.

Για να αποφευχθεί η δημιουργία του μητρώου δυσκαμψίας από την αρχή σε κάθε επανάληψη γίνεται η χρήση εργαλείων της matlab για πίνακες.

Αρχικά δημιουργούνται οι πίνακες A11,A12,B11,B12

```
A11 = [12  3 -6 -3;  3 12  3  0; -6  3 12 -3; -3  0 -3
12];
A12 = [-6 -3  0  3; -3 -6 -3 -6;  0 -3 -6  3;  3 -6  3 -
6];
B11 = [-4  3 -2  9;  3 -4 -9  4; -2 -9 -4 -3;  9  4 -3 -
4];
B12 = [ 2 -3  4 -9; -3  2  9 -2;  4  9  2  3; -9 -2  3
2];
```

οι οποίοι θα χρησιμοποιηθούν ως συντελεστές για την μόρφωση του μητρώου δυσκαμψίας τετραγωνικού στοιχείου μέσω της σχέσης

```
KE = 1/(1-nu^2)/24*([A11 A12;A12' A11]+nu*[B11 B12;B12'
B11]);
```

Το μητρώο δυσκαμψίας τετραγωνικού στοιχείου με πάχος t=1 και διαστάσεις dx=dy έχει τη μορφή[15]

$$k=\frac{1}{24(1-\nu^2)}\begin{bmatrix} 12-4\nu & & & & & & & \\ 3+3\nu & 12-4\nu & & & & & & \\ -6-2\nu & 3-9\nu & 12-4\nu & & & & & \\ -3+9\nu & 4\nu & -3-3\nu & 12-4\nu & & & & \\ -6+2\nu & -3-3\nu & 4\nu & 3-9\nu & & & & \\ -3-3\nu & -6+2\nu & -3+9\nu & -10-2\nu & & & & \\ 4\nu & -3+9\nu & -2+2\nu & 3+3\nu & & & & \\ 3-9\nu & 2-2\nu & 3+3\nu & -6+2\nu & & & & \end{bmatrix} \cdots$$

$$\cdots \begin{bmatrix} 12-4\nu & & & \\ 3+3\nu & 12-4\nu & & \\ -6-2\nu & 3-9\nu & 12-4\nu & \\ -3+9\nu & 4\nu & -3-3\nu & 12-4\nu \end{bmatrix}$$

Στην συνέχεια υπάρχουν τρία στάδια τα οποία καταλήγουν στη μόρφωση του μητρώου edofMat.

```
nodenrs = reshape(1:(1+nelx)*(1+nely),1+nely,1+nelx);

edofVec = reshape(2*nodenrs(1:end-1,1:end-1)+1,nelx*nely,1);

edofMat = repmat(edofVec,1,8)+repmat([0 1 2*nely+[2 3 0 1] -2 -1] ,nelx*nely,1);
```

Το μητρώο edofMat περιέχει σε κάθε σειρά του τους βαθμούς ελευθερίας που αντιστιχούν σε κάθε στοιχείο.

$$edofMat=\begin{bmatrix} 3 & 4 & 11 & 12 & 9 & 10 & 1 & 2 \\ 5 & 6 & 13 & 14 & 11 & 12 & 3 & 4 \\ 7 & 8 & 15 & 16 & 13 & 14 & 5 & 6 \\ 11 & 12 & 19 & 20 & 17 & 18 & 9 & 10 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 31 & 32 & 39 & 40 & 37 & 38 & 29 & 30 \end{bmatrix}$$

Ο πρώτος βαθμός ελευθερίας του στοιχείου είναι ο οριζόντιος βαθμός ελευθερίας της κάτω αριστερά γωνίας του τετραγώνου στο τοπικό σύστημα.

Για να επιτευχθεί η μόρφωσή του, πρώτα μορφώνεται το μητρώο nodenrs. Το μητρώο αυτό έχει διαστάσεις του αριθμούς των κόμβων κατά x επί τους αριθμούς των κόμβων κατά y ((nelx+1)(nely+1)). Το περιεχόμενό του είναι οι κόμβοι της δοκού.

$$noders = \begin{bmatrix} 1 & 5 & 9 & 13 & 17 \\ 2 & 6 & 10 & 14 & 18 \\ 3 & 7 & 11 & 15 & 19 \\ 4 & 8 & 12 & 16 & 20 \end{bmatrix}$$

Αυτό επιτυγχάνεται με την εντολή reshape της matlab.

Έπειτα μορφώνεται το μητρώο edofVec το οποίο είναι ένα μητρώο στήλη που περιέχει τον πάνω αριστερά οριζόντιο βαθμό ελευθερίας για κάθε ένα στοιχείο. Το μέγεθος του μητρώου είναι ο αριθμός των στοιχείων.

$$edofVec = \begin{bmatrix} 1 \\ 3 \\ 5 \\ 7 \\ \vdots \\ 39 \end{bmatrix}$$

Τέλος χρησιμοποιείται η εντολή repmat(edofVec,1,8) η οποία αντιγράφει το μητρώο edofVec οχτώ φορές οριζόντια.

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 \\ 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 \\ 7 & 7 & 7 & 7 & 7 & 7 & 7 & 7 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 39 & 39 & 39 & 39 & 39 & 39 & 39 & 39 \end{bmatrix}$$

και μετά προστίθεται το μητρώο repmat([0 1 2*nely+[2 3 0 1] -2 -1] ,nelx*nely,1), το οποίο μετατρέπει την κάθε σειρά στους βαθμούς ελευθερίας του κάθε στοιχείου. Έτσι δημιουργείται το μητρώο edofMat.

Στην συνέχεια υπολογίζονται τα μητρώα iK και jK με χρήση του μητρώου edofMat για την σύνθεση του μητρώου δυσκαμψίας της κατασκευής εντός των επαναλήψεων βελτιστοποίησης. Για τη σύνθεση των μητρώων iK, jK γίνεται χρήση της εντολής kron. Η εντολή kron(A,B) πολλαπλασιάζει το κάθε στοιχείο του μητρώου A (n,m) με τον πίνακα B (b,q)

$$kron(A,B)=\begin{bmatrix} a_{1,1}*B & \cdots & a_{1,\mathrm{m}}*B \\ \vdots & \ddots & \vdots \\ a_{n,1}*B & \cdots & a_{n,\mathrm{m}}*B \end{bmatrix}$$

επομένως επιστρέφει ένα μητρώο διαστάσεων n*b,m*q.

iK = reshape(kron(edofMat,ones(8,1))',64*nelx*nely,1);

jK = reshape(kron(edofMat,ones(1,8))',64*nelx*nely,1);

Με την χρήση των παραπάνω εντολών τα μητρώα iK και jK παίρνουν την μορφή

$$iK=\begin{bmatrix} dofselement1vertically \\ \vdots \\ 8times \\ \vdots \\ dofselement2vertically \\ \vdots \end{bmatrix}_{64*nelx*nely,1}$$

$$jK = \begin{bmatrix} 1 \\ \vdots \\ 8times \\ \vdots \\ 2 \\ \vdots \\ 8times \\ \vdots \\ 3 \\ \vdots \end{bmatrix}_{64*nelx*nely,1}$$

Ο ορισμός των φορτίων και των δεσμευμένων από στηρίξεις βαθμών ελευθερίας επιτυγχάνεται με τις εντολές

F = sparse(2,1,-1,2*(nely+1)*(nelx+1),1);

U = zeros(2*(nely+1)*(nelx+1),1);

fixeddofs = union([1:2:2*(nely+1)],[2*(nelx+1)*(nely+1)]);

alldofs = [1:2*(nely+1)*(nelx+1)];

freedofs = setdiff(alldofs,fixeddofs);

όπου

Η δύναμη παίρνει την τιμή -1 γιατί ασκείται προς τα κάτω.

Οι δεσμευμένοι βαθμοί ελευθερίας είναι οι οριζόντιοι βαθμοί που βρίσκονται στην αριστερή πλευρά της κατασκευής. Για να είναι εύκολο να υπολογιστεί το φίλτρο μέσα στην επανάληψη βελτιστοποίησης, χωρίς να είναι απαραίτητη η χρήση εσωτερικής επανάληψης, δημιουργούνται οι πίνακες H και Hs. Ο πίνακας H είναι τετραγωνικός πίνακας με διαστάσεις τον αριθμό των στοιχείων της κατασκευής ((nelx*nely)* (nelx*nely)). Η κάθε σειρά και κάθε στήλη του πίνακα H αντιπροσωπεύει το κάθε στοιχείο της κατασκευής. Οι τιμές που λαμβάνει ο πίνακας σε κάθε σειρά είναι οι τιμές του συντελεστή H_{ei} για τα στοιχεία εντός της ακτίνας εφαρμογής του φίλτρου και 0 για τα υπόλοιπα.

$$H = \begin{bmatrix} H_{11} & \cdots & H_{1(\text{nelxnely})} \\ \vdots & \ddots & \vdots \\ H_{(\text{nelxnely})1} & \cdots & H_{(\text{nelxnely})(\text{nelxnely})} \end{bmatrix} \begin{matrix} \sigma\tau o\iota\chi\varepsilon\acute{\iota}o & 1 \\ \vdots & \\ \sigma\tau o\iota\chi\varepsilon\acute{\iota}o & (\text{nelxnely}) \end{matrix}$$

$$\begin{matrix} \sigma\tau o\iota\chi\varepsilon\acute{\iota}o & \sigma\tau\varepsilon\iota\chi\varepsilon\acute{\iota}o \\ 1 & (\text{nelxnely}) \end{matrix}$$

Αυτό επιτυγχάνεται με την δημιουργία των πινάκων iH,jH και sH, οι οποίοι λαμβάνουν τιμές έτσι ώστε H(iH(i),jH(i))=sH(i). Επομένως ο πίνακας iH παίρνει τις τιμές των στοιχείων των σειρών του πίνακα H ενώ ο πίνακας jH παίρνει τις τιμές των στοιχείων των στηλών του πίνακα H έτσι ώστε να τοποθετούνται μέσο της εντολής H = sparse(iH,jH,sH) οι τιμές του πίνακα sH στο μητρώο H.

Το μητρώο Hs είναι ένα μητρώο στήλη που προκύπτει από το άθροισμα των σειρών του μητρώου H. Στην συνέχεια προετοιμάζεται η επανάληψη βελτιστοποίησης τοπολογίας. Δημιουργείται το μητρώο x που είναι τετραγωνικό και έχει διαστάσεις nelx*nely. Το μητρώο x περιέχει τις πυκνότητες για το κάθε στοιχείο της κατασκευής και στην αρχή ξεκινάει τοποθετώντας παντού την τιμή volfrac.

Δημιουργείται το μητρώο xPhys το οποίο ξεκινάει ίσο με το μητρώο x και μετά σε κάθε επανάληψή του τοποθετούνται οι τιμές των φιλτραρισμένων πυκνοτήτων μέσω της σχέσης

xPhys(:) = (H*xnew(:))./Hs;

Δίνονται οι αρχικές τιμές στις μεταβλητές loop και change

Loop = 0;
Change =1;

Η μεταβλητή loop μετράει τον αριθμό των επαναλήψεων και η μεταβλητή change μετράει την διαφορά ανάμεσα στο καινούργιο και το παλιό μητρώο x και αποτελεί την συνθήκη τερματισμού της επανάληψης βελτιστοποίησης.

Η επανάληψη βελτιστοποίησης ξεκινάει με την συνθήκη για το change. Όσο είναι μεγαλύτερο από 0,01 η βελτιστοποίηση θα συνεχίζεται.

Αρχικά γίνεται η επίλυση των πεπερασμένων στοιχείων για τον υπολογισμό του μητρώου των μετατοπίσεων U με την χρήση των πινάκων που δημιουργήθηκαν παραπάνω.

%% FE-ANALYSIS

```
sK = reshape(KE(:)*(Emin+xPhys(:)'.^penal*(E0-Emin)),64*nelx*nely,1);
K = sparse(iK,jK,sK); K = (K+K')/2;
U(freedofs) = K(freedofs,freedofs)\F(freedofs);
```

Το μητρώο sK είναι ένα μητρώο στήλη, διαστάσεων 64*nelx*nely το οποίο περιέχει για κάθε ένα στοιχείο το γινόμενο του μητρώου δυσκαμψίας του στοιχείου με την σχέση $E_{min}+x_e^p(E_0-E_{min})$.

Μέσω της εντολής sparse μορφώνεται το ολικό μητρώο δυσκαμψίας της κατασκευής. Το μητρώο iK λειτουργεί ως δείκτης, προσδιορίζοντας σε ποια οριζόντια στήλη θα τοποθετηθεί το κάθε sK ενώ το μητρώο jK περνάει με την σειρά από όλους του βαθμούς ελευθερίας των στοιχείων της κατασκευής για να δείξει την στήλη στην οποία θα τοποθετηθεί το sK. Μετά την μόρφωση του μητρώου K λύνεται η εξίσωση ισορροπίας $U=K^{-1}F$.

Έπειτα υπολογίζονται η αντικειμενική συνάρτηση και η παράγωγός της καθώς και η παράγωγος του όγκου ως προς τις πυκνότητες x. Η αντικειμενική συνάρτηση υπολογίζεται μέσω του τύπου

$$C(x)=U^T KU=\sum_{e=1}^{N} E_e(x_e)u_e{}^T k u_e=\sum_{e=1}^{N}(\mathrm{E}_{\min}+x_e^P(\mathrm{E}_0-\mathrm{E}_{\min}))u_e{}^T k u_e \qquad (4.4)$$

Για τον υπολογισμό της χρησιμοποιείται το βοηθητικό μητρώο ce.

```
ce = reshape(sum((U(edofMat)*KE).*U(edofMat),2),nely,nelx);
```

Το οποίο κάνει τον υπολογισμό $u_e^T k u_e$ αξιοποιώντας το μητρώο edofMat μέσω του μητρώου U(edofMat), το οποίο είναι ένα μητρώο του οποίου οι σειρές αντιπροσωπεύουν τα στοιχεία και οι στήλες τις μετακινήσεις στους βαθμούς ελευθερίας των στοιχείων αυτών. Με χρήση της εντολής reshape το μητρώο ce παίρνει την μορφή ορθογωνίου μητρώου με διαστάσεις nelx*nely και στοιχεία το αποτέλεσμα της σχέσης $u_e^T k u_e$ για κάθε ένα στοιχείο x. Επομένως η αντικειμενική συνάρτηση υπολογίζεται

```
c = sum(sum((Emin+xPhys.^penal*(E0-Emin)).*ce));
```

Η παράγωγος της αντικειμενικής συνάρτησης δίνεται από την σχέση

$$\frac{\partial C(x)}{\partial x_e}=-u_e(x)^T\frac{\partial K_e(x)}{\partial x_e}u_e(x)=-px_e^{p-1}(\mathrm{E}_0-\mathrm{E}_{\min})\,\mathrm{u}_e^T k u_e \qquad (4.5)$$

Με χρήση του βοηθητικού μητρώου ce που περιέχει τις τιμές των $u_e^T k u_e$ υπολογίζεται ως

```
dc = -penal*(E0-Emin)*xPhys.^(penal-1).*ce;
```

Η παράγωγος του όγκου ως προς τις πυκνότητες x είναι ένας μοναδιαίος πίνακας με διαστάσεις nelx*nely.

Στη συνέχεια γίνεται το φιλτράρισμα των παραγώγων της αντικειμενικής συνάρτησης και του όγκου με χρήση των σχέσεων εφαρμογής του φίλτρου που περιγράφηκαν παραπάνω. Ανάλογα με το ft που έχουμε επιλέξει γίνεται φιλτράρισμα μόνο της παραγώγου της αντικειμενικής συνάρτησης ή και της παραγώγου του όγκου.

```
%% FILTERING/MODIFICATION OF SENSITIVITIES
if ft == 1
  dc(:) = H*(x(:).*dc(:))./Hs./max(1e-3,x(:));
elseif ft == 2
  dc(:) = H*(dc(:)./Hs);
  dv(:) = H*(dv(:)./Hs);
end
```

Στο επόμενο κομμάτι γίνεται ο υπολογισμός του συντελεστή του Laplace λ έτσι ώστε να ικανοποιείτε η σχέση

$$\sum_{e=1}^{n} \alpha_e x_e - V = 0 \tag{4.6}$$

Όπου το x_e προκύπτει από την σχέση

$$x_e = \begin{cases} 0 & if \quad (\frac{ab_e^k}{\lambda \alpha_e})^{\frac{1}{1+a}} < 0 \\ (\frac{ab_e^k}{\lambda \alpha_e})^{\frac{1}{1+a}} & if \quad (\frac{ab_e^k}{\lambda \alpha_e})^{\frac{1}{1+a}} = 0 \\ 1 & if \quad (\frac{ab_e^k}{\lambda \alpha_e})^{\frac{1}{1+a}} > 1 \end{cases}$$

Όπου

$$b_e^k = \frac{(x_e^k)^{1+a}}{a} \frac{\partial J}{\partial x_e}\bigg|_{x=x^k} \tag{4.7}$$

και α=1.

Για τον υπολογισμό του λ ορίζουμε ένα μέγιστο και ελάχιστο λ και ξεκινάμε με το μέσον τους. Υπολογίζοντας το x ελέγχουμε αν η συνθήκη για τον όγκο ικανοποιείται και αν όχι, τότε αν ο όγκος προκύπτει μεγαλύτερος αυξάνουμε το λ ενώ αν προκύπτει μικρότερος το μειώνουμε μέχρι να γίνει ίσος. Για το x_e ισχύει ο επιπλέον περιορισμός ότι σε κάθε επανάληψη υπολογισμού του λ δεν μπορεί να αλλάζει τιμή περισσότερο από μια μεταβλητή που στον κώδικα παίρνει την τιμή 0,2.

Επίσης κάθε φορά γίνεται και εφαρμογή του φίλτρου για τις πυκνότητες x αν έχει γίνει επιλογή ft=2.

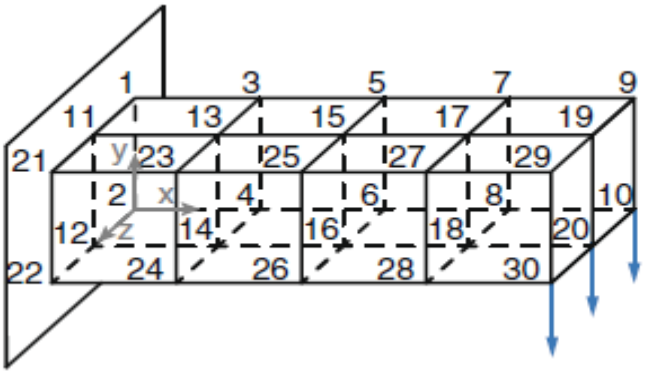
```
%% OPTIMALITY CRITERIA UPDATE OF DESIGN VARIABLES AND PHYSICAL DENSITIES
 l1 = 0; l2 = 1e9; move = 0.2;
 while (l2-l1)/(l1+l2) > 1e-3
   lmid = 0.5*(l2+l1);
   xnew = max(0,max(x-move,min(1,min(x+move,x.*sqrt(-dc./dv/lmid)))));
   if ft == 1
     xPhys = xnew;
   elseif ft == 2
     xPhys(:) = (H*xnew(:))./Hs;
   end
   if sum(xPhys(:)) > volfrac*nelx*nely, l1 = lmid; else l2 = lmid; end
 end
```

Τέλος υπολογίζεται η διαφορά των νέων πυκνοτήτων σε σχέση με τις προηγούμενες, εκτυπώνεται στην επιφάνεια της matlab ο αριθμός της επανάληψης, η αντικειμενική συνάρτηση και η διαφορά change και δημιουργείται ένα γράφημα που απεικονίζει την κατασκευή όπως έχει μορφωθεί σε εκείνη την επανάληψη.

4.3 Κώδικας Βελτιστοποίησης Τοπολογίας 3D

Ο κώδικας αυτός επιλύει ένα πρόβλημα βελτιστοποίησης τοπολογίας σε τρεις διαστάσεις με χρήση του λογισμικού Matlab. Η δομή του είναι παρόμοια με τον κώδικα 88 γραμμών που αναλύθηκε παραπάνω.

Το παράδειγμα του επιλύεται σε αυτό τον κώδικα είναι μιας δοκού η οποία στηρίζεται πακτωμένα από την μια πλευρά και έχει τρεις κατακόρυφες δυνάμεις από την άλλη.



Οι αλλαγές που υπάρχουν είναι οι εξής:

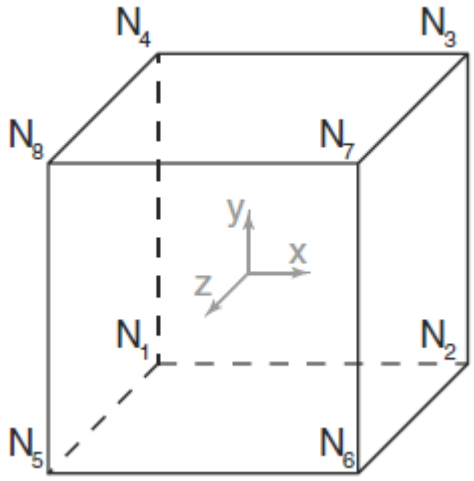
- Στην αρχή του προγράμματος ορίζεται ένας μέγιστος αριθμός επαναλήψεων βελτιστοποίησης, μετά τον οποίο η βελτιστοποίηση σταματάει ακόμα και αν δεν έχει φτάσει στο βέλτιστο.

  maxloop = 200;   % Maximum number of iterations

- Ο υπολογισμός του τοπικού μητρώου δυσκαμψίας γίνεται μέσω της συνάρτησης KE = lk_H8(nu), όπου nu ο λόγος του Poisson. Στην συνάρτηση αυτή δημιουργούνται έξι υπομητρώα και μετά το ολικό τοπικό μητρώο δυσκαμψίας.
- Για την δημιουργία του μητρώου edofMat απαιτείται η δημιουργία των μητρώων nodegrd, nodeids, nodeidz, nodeids και edofVec. Το μητρώο nodegrd περιέχει τους κόμβους της πρώτης επιφάνειας στον άξονα x,y. Το μητρώο nodeids περιέχει τους πάνω αριστερά κόμβους του κάθε στοιχείου της ίδιας επιφάνειας. Το nodeidz περιέχει αθροιστικά τους βαθμούς ελευθερίας της κάθε επιφάνειας μέχρι την επιφάνεια z-1. Το nodeids με χρήση του nodeidz τελικά αποθηκεύει τον άνω αριστερά κόμβο του κάθε κυβικού στοιχείου. Τέλος το μητρώο edofVec , όπως και στον κώδικα top88, με χρήση του nodeids αποθηκεύει τον πρώτο βαθμό ελευθερίας του κάθε στοιχείου.

$$edofMat = \begin{bmatrix} 4 & 5 & 6 & \cdots & 31 & 32 & 33 \\ 10 & 11 & 12 & \cdots & 37 & 38 & 39 \\ 16 & 17 & 18 & \cdots & 43 & 44 & 45 \\ 22 & 23 & 24 & \cdots & 49 & 50 & 51 \\ 34 & 35 & 36 & \cdots & 61 & 62 & 63 \\ 40 & 41 & 42 & \cdots & 67 & 68 & 69 \\ 46 & 47 & 48 & \cdots & 73 & 74 & 75 \\ 52 & 53 & 54 & \cdots & 79 & 80 & 81 \end{bmatrix}$$

Ο πρώτος βαθμός ελευθερίας κάθε στοιχείου στο τοπικό σύστημα του στοιχείου είναι ο οριζόντιος βαθμός ελευθερίας της κάτω αριστερά γωνίας του κύβου.



Εικόνα 23. Τοπικοί αρίθμηση κόμβων στοιχείου.

- Για την εξοικονόμηση χρόνου, και επειδή οι μετακινήσεις που προκύπτουν από την επίλυση της εξίσωσης ισορροπίας δεν χρειάζεται να είναι υπολογισμένες με πλήρη ακρίβεια, χρησιμοποιείται η Precondition Conjugated Gradient Method (PCG) για την επίλυση της εξίσωσης ισορροπίας. Η μέθοδος αυτή προσεγγίζει της μετατοπίσεις, χωρίς όμως να επιλύει την εξίσωση ισορροπίας, και επομένως εξοικονομεί χρόνο γιατί δεν χρειάζεται να αντιστρέφει το μητρώο δυσκαμψίας σε κάθε επανάληψη. Η PCG υπάρχει στις βιβλιοθήκες της Matlab και καλείται απ' ευθείας. Τα δεδομένα που απαιτούνται για την χρήση της PCG είναι το μητρώο δυσκαμψίας της κατασκευής, το μητρώο δυνάμεων, το όριο σύγκλισης, ο μέγιστος αριθμός των επαναλήψεων και ο προρρυθμιστήρας (preconditioner) ο οποίος επιλέγεται να είναι, σύμφωνα με τον Jacobi, ο διαγώνιος πίνακας του μητρώου δυσκαμψίας.

```
U(freedofs,:)                                                    =
pcg(K(freedofs,freedofs),F(freedofs,:),tolit,maxit,M);
```

- Για την απεικόνιση του σχήματος χρησιμοποιείται η συνάρτηση display_3D η οποία δέχεται ως μεταβλητές το μητρώο των πυκνοτήτων xPhys και επιστρέφει το σχήμα της κατασκευής. Επειδή το μητρώο xPhys είναι τρισδιάστατο προκειμένου να γίνει η απεικόνιση των πυκνοτήτων αξιοποιείται η εντολή patch για κάθε πεπερασμένο στοιχείο. Για να γίνει αυτό χρησιμοποιούνται οι επαναλήψεις

```
for k = 1:nelz
   z = (k-1)*hz;
   for i = 1:nelx
      x = (i-1)*hx;
      for j = 1:nely
        y = nely*hy - (j-1)*hy;
       end
     end
  end
```

όπου οι μεταβλητές x,y,z είναι οι συντεταγμένες μια ακμής του κάθε πεπερασμένου στοιχείου όπως φαίνεται στην εικόνα παρακάτω.
Εντός της επανάληψης αρχικά τοποθετείτε ένα φίλτρο έτσι ώστε να απεικονίζονται πυκνότητες μεγαλύτερες από το 0,5 μόνο. Έπειτα γίνεται χρήση της εντολής pach.

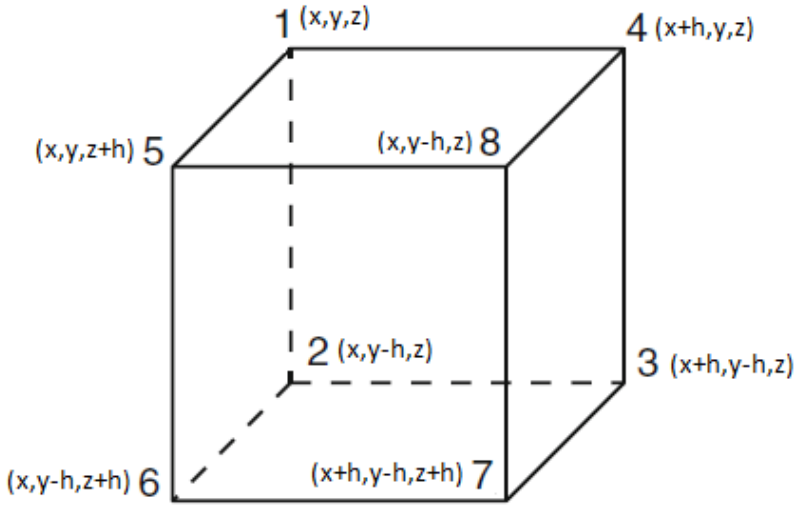
```
patch('Faces',face,'Vertices',vert,'FaceColor',[0.2+0.8*(
1-rho(j,i,k)),0.2+0.8*(1-rho(j,i,k)),0.2+0.8*(1-
rho(j,i,k))]);
```

Οι εντολή αυτή απεικονίζει πολύγονα των οποίων τις συντεταγμένες λαμβάνει από το μητρώο σειρές του μητρώου vert οι οποίες προσδιορίζονται από το μητρώο Face κάθε φορά.
Το μητρώο Vertices υπολογίζεται ως

```
vert = [x y z; x y-hx z; x+hx y-hx z; x+hx y z; x y
z+hx;x y-hx z+hx; x+hx y-hx z+hx;x+hx y z+hx];
```

όπου τα x,y,z είναι οι συντεταγμένες του αντίστοιχου στοιχείου που απεικονίζεται. Το hx είναι το ύψος του στοιχείου και είναι ίσο με ένα. Για να αντιστοιχιστούν οι επιφάνειες με κάθε γραμμή του μητρώου vert χρησιμοποιείτε το μητρώο face το οποίο ορίζεται ως

```
face = [1 2 3 4; 2 6 7 3; 4 3 7 8; 1 5 8 4; 1 2 6 5; 5 6
7 8];
```



Εικόνα 24. Οι αριθμοί αντιστοιχούν στους αριθμούς του μητρώου Face και τα x,y,z αντιστοιχούν τις τιμές το μητρώου vert.

ΚΕΦΑΛΑΙΟ 5

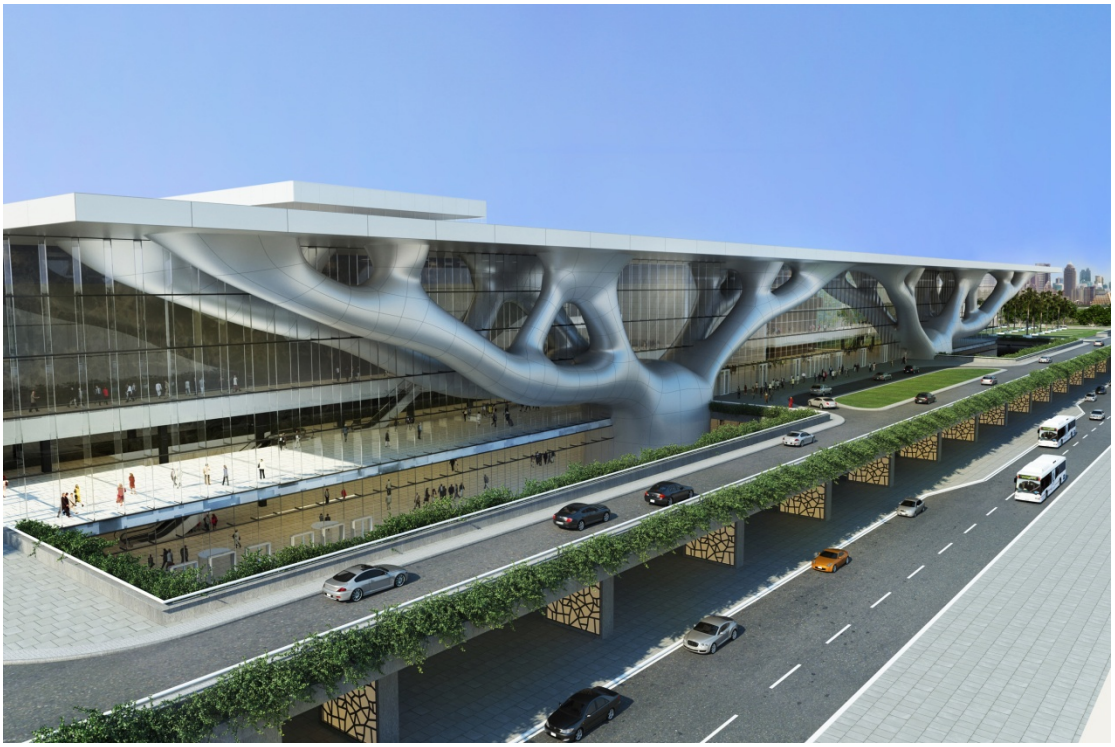
ΕΠΕΜΒΑΣΕΙΣ ΚΑΙ ΜΟΡΦΟΠΟΙΗΣΕΙΣ ΣΤΟΥΣ ΚΩΔΙΚΕΣ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗΣ ΤΟΠΟΛΟΓΙΑΣ

5.1 Εισαγωγή

Στο παρόν κεφάλαιο γίνεται αναφορά στις μορφοποιήσεις που έγιναν στους κώδικες που αναλύθηκαν στο κεφάλαιο 4. Οι επεμβάσεις χωρίζονται σε δύο μέρη. Στο πρώτο κομμάτι αναλύεται και εφαρμόζεται ο συνδυασμός στατικών και θερμικών φορτίσεων καθώς και εφαρμογή φορτίσεων εξαρτώμενων από τη μάζα της κατασκευής. Στο δεύτερο κομμάτι εξετάζονται οι τρόποι με τους οποίους θα μπορούσε να βελτιστοποιηθεί ο χρόνος περάτωσης του προγράμματος βελτιστοποίησης. Οι δοκιμές γίνονται στο κώδικα top3D λόγο του ότι είναι και ο κώδικας με το μεγαλύτερο υπολογιστικό βάρος.

5.2 ΕΦΑΜΡΟΓΗ ΘΕΡΜΙΚΩΝ ΔΥΝΑΜΕΩΝ ΣΤΟΝ ΚΩΔΙΚΑ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗΣ ΤΟΠΟΛΟΓΙΑΣ TOP88

Ορίζοντας ως βάση τη μορφή του στεγάστρου του συνεδριακού κέντρου του Qatar που σχεδιάστηκε από τον αρχιτέκτονα Arata Isozaki προσπαθήσαμε να εισάγουμε στον κώδικα βελτιστοποίησης τοπολογίας μια θερμική φόρτιση προκειμένου να αποκτήσουμε ένα δενδροειδές σχήμα στη κατασκευή ενώ παράλληλα να ικανοποιείται η εξίσωση ισορροπίας. Για να πραγματοποιηθεί αυτό δοκιμάσαμε μαζί με το στατικό πρόβλημα βελτιστοποίησης να μπορεί να επιλυθεί και ένα θερμικό πρόβλημα και μετά να πραγματοποιείται βελτιστοποίηση στο σύνολο των δύο προβλημάτων.



Εικόνα 25. Στέγαστρο συνεδριακού κέντρου

Αρχικά επιλέγεται ένα παράδειγμα προκειμένου να μπορεί να γίνει σύγκριση των διαφορών που επιφέρουν στην κατασκευή οι επεμβάσεις που γίνονται στο κώδικα.

Στο παράδειγμα αυτό η στήριξη βρίσκεται στο κάτω αριστερά μέρος της κατασκευής και οι φορτίσεις είναι τρεις συγκεντρωμένες δυνάμεις στο άνω μέρος της κατασκευής.

Ο αριθμός των πεπερασμένων στοιχείων, το ποσοστό του όγκου που θα καλυφθεί, η δύναμη της μεθόδου SIMP που θα χρησιμοποιηθεί, η ακτίνα του φίλτρου καθώς και το είδος του φίλτρου ορίζονται από την εντολή

```
top88(250,100,0.3,3.0,1.5,1)
```

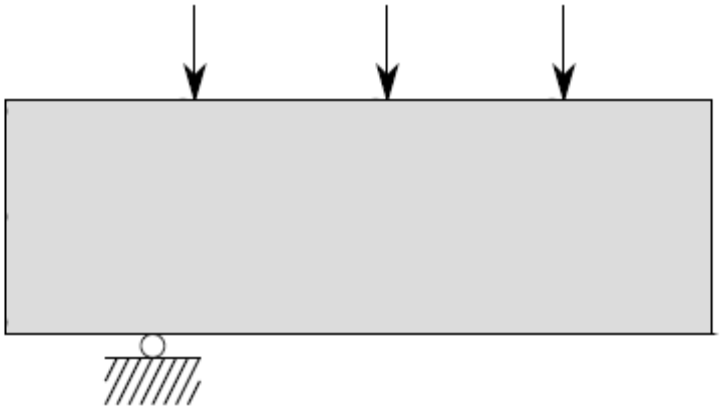
Για να οριστούν στον κώδικα βελτιστοποίησης οι δυνάμεις, χρησιμοποιούνται οι εντολές

```
iF_h = 1:2*(nely+1):2*(nely+1)*nelx+1;
iF_v = 2:2*(nely+1):2*(nely+1)*nelx+2;
F = sparse(iF_v,1,-1,2*(nely+1)*(nelx+1),1)
+sparse(iF_h,1,-0.3,2*(nely+1)*(nelx+1),1);
```

ενώ για τους δεσμευμένους βαθμούς ελευθερίας της στήριξης οι παρακάτω εντολές

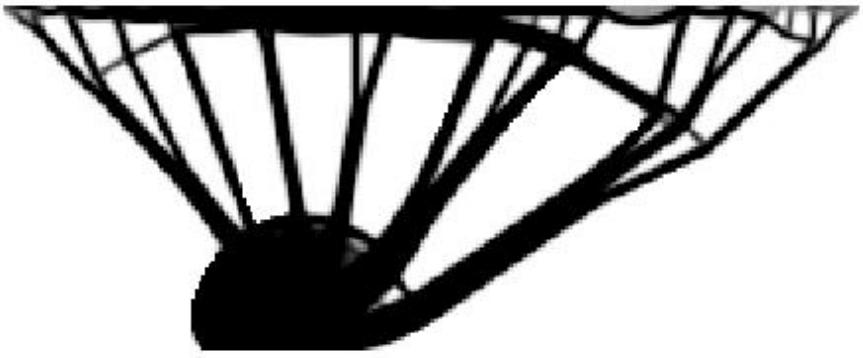
```
fixeddofs = union(2*(nely+1)*floor(15/60*(nelx+1)):
```

```
2*(nely+1):2*(nely+1)*floor(20/60*(nelx+1)),2*(nely+1)*fl
oor(15/60*(nelx+1))-
1:2*(nely+1):2*(nely+1)*floor(20/60*(nelx+1))-1);
```



Εικόνα 26. Σκαρίφημα στήριξης και δυνάμεων επιλεγμένου παραδείγματος

Το αποτέλεσμα φαίνεται παρακάτω



Εικόνα 27. Αποτέλεσμα κώδικα top88 για στατικές φορτίσεις.

Στη συνέχεια γίνεται διατύπωση του θερμικού προβλήματος και εφαρμογή του στον κώδικα βελτιστοποίησης τοπολογίας.

5.2.1 ΔΙΑΤΥΠΩΣΗ ΘΕΡΜΙΚΟΥ ΠΡΟΒΛΗΜΑΤΟΣ

Οι μηχανισμοί με τους οποίους μεταφέρεται η θερμότητα είναι:

- Αγωγή θερμότητας (heat conduction)
- Με συναγωγή (convective energy transport)
- Με διάχυση (diffusive energy transport)
- Μέσο ακτινοβολίας (radiative energy transport)

Στην περίπτωσή μας θεωρούμε μεταφορά θερμότητας με αγωγή από στοιχείο σε στοιχείο. Στο θερμικό πρόβλημα έχουμε ένα βαθμό ελευθερίας σε κάθε κόμβο της κατασκευής, ο οποίος είναι η θερμοκρασία Τ. Ως φόρτιση έχουμε θερμικές δυνάμεις οι οποίες τροφοδοτούν την κατασκευή με θερμότητα η οποία μεταφέρεται μέσω της κατασκευής στα σημεία που θεωρούμε τη θερμοκρασία σταθερή και ίση με μηδέν και διαφεύγει στο περιβάλλον ή το έδαφος.

Για να ορίσουμε το πρόβλημα πρέπει αρχικά να προσδιορίσουμε το μητρώο θερμικής αγωγιμότητας του κάθε στοιχείου. Τα στοιχεία που χρησιμοποιούμε είναι τετραγωνικά επομένως το μητρώο θερμικής αγωγιμότητας είναι

$$k\begin{bmatrix} 0.66 & -0.16 & -0.33 & -0.16 \\ -0.16 & 0.66 & -0.16 & -0.33 \\ -0.33 & -0.16 & 0.66 & -0.16 \\ -0.16 & -0.33 & -0.16 & 0.66 \end{bmatrix}$$

Όπως και στο στατικό πρόβλημα, το μέγεθος που επιδιώκουμε να ελαχιστοποιήσουμε προκειμένου να αποκτήσουμε τη βέλτιστη μορφή της κατασκευής είναι το γινόμενο της θερμικής δύναμης με τις θερμοκρασίες που αναπτύσσονται στη κατασκευή. Ονομάζουμε αυτό το μέγεθος "θερμικό έργο".

$$C(\mathrm{x}) = F^{\mathrm{T}}T(x) \tag{5.1}$$

Προκειμένου να υπολογίσουμε τις θερμοκρασίες που αναπτύσσονται στους κόμβους της κατασκευής πρέπει να μορφώσουμε το συνολικό μητρώο θερμικής αγωγιμότητας της κατασκευής. Οι θερμοκρασίες προκύπτουν

$$T(x) = \mathrm{K}^{-1}(x)F \tag{5.2}$$

Επομένως το πρόβλημα βελτιστοποίησης της θερμικής φόρτισης είναι

$$\begin{aligned} &\min_x C(x) = F^{\mathrm{T}}T(\mathrm{x}) \\ &s.t. \\ &\sum_1^n x_e a = V \\ &T(x) = K(x)^{-1}_{thermal} F \\ &0 \le x_e \le 1 \end{aligned} \tag{5.3}$$

Όπως και στο στατικό πρόβλημα.

Η μέθοδος με την οποία λαμβάνουμε υπόψην μας την πυκνότητα x του κάθε στοιχείου στο μητρώο θερμικής αγωγιμότητας είναι η τροποποιημένη μέθοδος SIMP που αξιοποιείται στον κώδικα top88 θεωρώντας θερμικές σταθερές

k0 ως το συντελεστή θερμικής αγωγιμότητας του υλικού,

kmin ως μια ελάχιστη τιμή για να μην μηδενίζεται ο συντελεστής θερμικής αγωγιμότητας $k(x_e)$.

Επομένως:

$$\begin{aligned} \mathrm{k}(\mathrm{x}_e) &= k_{\min} + x_e^p(\mathrm{k}_0 - k_{\min}) \\ \mathrm{K}_{thermal} &= \mathrm{k}(\mathrm{x}_e)K \end{aligned} \tag{5.4}$$

Όπου $K_{thermal}$ είναι το τελικό μητρώο θερμικής αγωγιμότητας του κάθε στοιχείου και K είναι το κλασικό μητρώο θερμικής αγωγιμότητας του κάθε στοιχείου.

Για να ελαχιστοποιήσουμε τη συνάρτηση αξιοποιούμε τη μέθοδο Optimality Criteria (OC). Επομένως οι πυκνότητες ανανεώνονται μέσω του τύπου

$$x_e = \begin{cases} 0 & if \quad (\dfrac{ab_e^k}{\lambda\alpha_e})^{\frac{1}{1+a}} < 0 \\ (\dfrac{ab_e^k}{\lambda\alpha_e})^{\frac{1}{1+a}} & if \quad (\dfrac{ab_e^k}{\lambda\alpha_e})^{\frac{1}{1+a}} = 0 \\ 1 & if \quad (\dfrac{ab_e^k}{\lambda\alpha_e})^{\frac{1}{1+a}} > 1 \end{cases} \tag{5.5}$$

όπου

$$\begin{aligned} b_e^k &= \frac{(x_e^k)^{1+a}}{a}\frac{\partial C}{\partial x_e}\bigg|_{x=x^k} \\ \frac{dC}{dx_e} &= T(x_e)^T K_{thermal}(\mathrm{x}_e)T(\mathrm{x}_e)\Big|_{x_e=x_e^k} \end{aligned} \tag{5.6}$$

5.2.2 ΕΦΑΡΜΟΓΗ ΘΕΡΜΙΚΟΥ ΠΡΟΒΛΗΜΑΤΟΣ ΣΤΟΝ ΚΩΔΙΚΑ ΜΕΛΤΙΣΤΟΠΟΙΗΣΗΣ ΤΟΠΟΛΟΓΙΑΣ TOP88

Για να εφαρμόσουμε τα παραπάνω στον κώδικα πρέπει να κάνουμε τις εξής αλλαγές:

Αρχικά να ορίσουμε τις θερμικές σταθερές k0 και kmin

k0 = 1;

kmin = 1e-5;

Έπειτα να ορίσουμε το τοπικό μητρώο θερμικής αγωγιμότητας του τετραγωνικού στοιχείου

```
KE_th = [  0.666666666666667  -0.166666666666667
          -0.333333333333333  -0.166666666666667 ; ...
          -0.166666666666667   0.666666666666667
          -0.166666666666667  -0.333333333333333 ; ...
          -0.333333333333333  -0.166666666666667
           0.666666666666667  -0.166666666666667 ; ...
          -0.166666666666667  -0.333333333333333
           -0.166666666666667   0.666666666666667 ];
```

και να κάνουμε τις απαραίτητες αλλαγές στα μητρώα nodenrs,edofVec,edofMat,iK,jK έτσι ώστε να θεωρούν ένα βαθμό ελευθερίας σε κάθε κόμβο αντί για δύο.

Στη συνέχεια η εντολή του αντίστοιχου ελαστικού προβλήματος

```
sK = reshape(KE(:)*(Emin+(xPhys(:))'.^penal*(E0-Emin)),64*nelx*nely,1);
```

πρέπει να πάρει την μορφή

```
sK = reshape(KE_th(:)*(kmin+xPhys(:)'.^penal*(k0-kmin)),16*nelx*nely,1);
```

Έτσι ώστε να λαμβάνονται υπόψη στο μητρώο sk οι θερμικές σταθερές k0, kmin. Με αυτό τον τρόπο δημιουργείται ένα μητρώο στήλη sk με διαστάσεις 16 φορές τον αριθμό των πεπερασμένων στοιχείων. Αυτό συμβαίνει διότι το τοπικό μητρώο KE_th αποτελείτε από 16 στοιχεία αντί για 64.

Τέλος για τον υπολογισμό των c και dc αντί για τα μέτρα ελαστικότητας E0 και Emin χρησιμοποιούνται οι θερμικές σταθερές k0 και kmin.

```
c_th = sum(sum((kmin+ (xPhys.^penal)*(k0-kmin)).*ce));
```

```
dc_th =-(k0-kmin)*(penal*xPhys.^(penal-1)).*ce;
```

Με τις αλλαγές αυτές ο κώδικας βελτιστοποίησης τοπολογίας top88 μετατρέπεται ώστε να επιλύει θερμικά προβλήματα.

Παρακάτω παρουσιάζονται παραδείγματα θερμικών φορτίσεων πάνω στο παράδειγμα που είχαμε ορίσει:

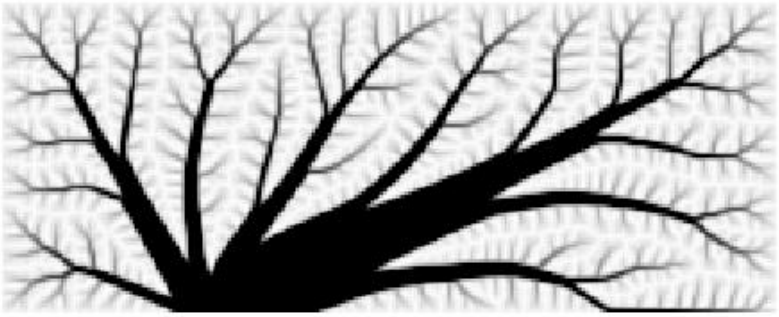
- Εφαρμογή θερμικής δύναμης σε όλη την επιφάνεια

Για την εφαρμογή της επιφανειακής δύναμης αρκεί η εντολή

```
F_th = sparse(1:(nelx+1)*(nely+1),1,-0.01,(nely+1)*(nelx+1),1);
```

Η οποία εφαρμόζει δύναμη 0,01 σε όλη την επιφάνεια της κατασκευής.

Το αποτέλεσμα που μας δίνει το πρόγραμμα φαίνεται στην παρακάτω εικόνα.



Εικόνα 28. Θερμική δύναμη σε όλη την επιφάνεια.

Η θερμική δύναμη σε όλη την επιφάνεια προκαλεί τη δημιουργία κλαδιών τα οποία προσπαθούν να παραλάβουν τη θερμική δύναμη και να τη μεταφέρουν στη στήριξη.

- Εφαρμογή θερμικής δύναμης σε όλη την επιφάνεια με φθίνουσα τιμή από άνω προς τα κάτω.

Για να αποφύγουμε τα πολυάριθμα κλαδιά στο κάτω μέρος προσαρμόζουμε την επιφανειακή δύναμη ώστε να μειώνεται όσο κατεβαίνει μέχρι που στο κάτω μέρος μηδενίζεται. Αυτό πραγματοποιείτε με τις εντολές

```
ivec = (1:nely+1);
ind = repmat(ivec,1,nelx+1);
ind = abs((ind-(nely+1)));
F_th = ind(:)*1e-3;
```

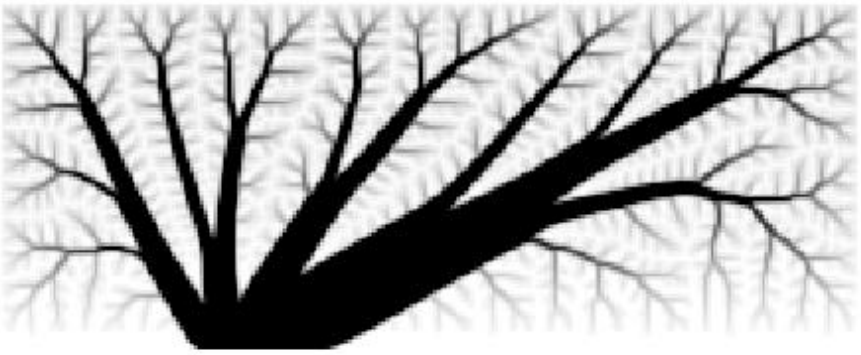
Το μητρώο ivec είναι ένα μητρώο γραμμή που περιέχει τους αριθμούς από το 1 μέχρι το nely+1. Με τη χρήση της εντολής repmat το μητρώο ivec αντιγράφεται nelx+1 φορές και δημιουργείται το μητρώο γραμμή ind διαστάσεων 1 x (nely * nelx).

Στη συνέχεια επειδή εμείς θέλουμε οι μεγάλοι συντελεστές να είναι από την άνω πλευρά μορφοποιούμε το μητρώο ind και τέλος το πολλαπλασιάζουμε με 10^{-3} για να του δώσουμε μια δύναμη στις διαστάσεις του προβλήματος μας.

$$ivec = \begin{bmatrix} 1 & 2 & \cdots & nely \end{bmatrix} \tag{5.7}$$

$$ind = \begin{bmatrix} ivec - nely & ivec - nely & \cdots & ivec - nely \end{bmatrix} * 10^{-3} \tag{5.8}$$

Παρακάτω φαίνεται η εικόνα της κατασκευής.



Εικόνα 29. Θερμική δύναμη κλιμακωμένη κατά ύψος της επιφάνειας

Τα κλαδιά μειώθηκαν στο κάτω μέρος όπως θέλαμε.

- Εφαρμογή θερμικής δύναμης στο άνω μέρος της κατασκευής.

Σε αυτή την περίπτωση εφαρμόζουμε δύναμη μόνο στο άνω μέρος της κατασκευής. Αυτό πραγματοποιείται με τις εντολές

```
sF_th = 1:(nely+1):nelx*(nely+1)+1;
F_th = sparse(sF_th,1,1,(nely+1)*(nelx+1),1);
```

Όπου στο sF_th τοποθετούνται οι άνω κόμβοι τις κατασκευής και μετά με την εντολή sparse στους κόμβους αυτούς τοποθετείτε θερμική δύναμη 1.

Τα αποτελέσματα που λαμβάνουμε φαίνονται παρακάτω.

Εικόνα 30. Θερμική δύναμη που ασκείτε στο πάνω μέρος της κατσκευής

5.2.3 ΣΥΝΔΙΑΣΜΟΣ ΘΕΡΜΙΚΩΝ ΚΑΙ ΣΤΑΤΙΚΩΝ ΔΥΝΑΜΕΩΝ

Προκειμένου να εντάξουμε στον κώδικα top88 το θερμικό πρόβλημα αρκεί να του εισάγουμε το θερμικό κώδικα που δημιουργήσαμε. Έξω από την επανάληψης της βελτιστοποίησης χωρίζουμε τον κώδικα σε δύο μέρη. Στο πρώτο κομμάτι εκτελούνται οι εντολές που αντιστοιχούν στο κομμάτι της προετοιμασίας του στατικού προβλήματος βελτιστοποίησης και στο δεύτερο κομμάτι εισάγουμε τις εντολές που χρησιμοποιούνται για την προετοιμασία του θερμικού κομματιού βελτιστοποίησης. Έπειτα υπολογίζεται το φίλτρο και ξεκινάει η επανάληψη.

Για να συνδυάσουμε το στατικό και θερμικό πρόβλημα θα πρέπει να υπολογίσουμε το συνολικό έργο C ως συνδυασμό του στατικού και θερμικού έργου. Προκειμένου να το κάνουμε αυτό θα χρησιμοποιήσουμε μια μεταβλητή weight. Η μεταβλητή αυτή θα αντιπροσωπεύει το ποσοστό που θα λαμβάνεται υπόψη στο τελικό συνδυαστικό πρόβλημα το κάθε επιμέρους πρόβλημα. Επομένως η σχέση για το συνολικό έργο θα είναι

C_tot = weight * C_structural + (1-weight) * C_thermal.

Ανάλογα με την τιμή του weight που εισάγουμε στο κώδικα θα προκύπτουν και τα ποσοστά που θα λάβει υπόψη του το κάθε πρόβλημα. Στον κώδικα έχουμε επιλέξει να υπάρχει μια επανάληψη η οποία θα δίνει στο weight διαφορές τιμές και τα αποτελέσματα θα αποθηκεύονται σε ξεχωριστούς φακέλους. Εντός της επανάληψης γίνεται ξεχωριστή επίλυση των πεπερασμένων στοιχείων για το κάθε πρόβλημα και υπολογίζονται η αντικειμενική συνάρτηση και η παράγωγος της ξεχωριστά για το κάθε ένα. Μετά μέσω των σχέσεων

```
% total objective function
c_tot = weight_s*c+weight_th*c_th;
%total sensitivities
dc_tot = weight_s*dc+weight_th*dc_th;
```

υπολογίζονται η συνολική συνάρτηση βελτιστοποίησης και η παράγωγος της. Με τη χρήση αυτών των μεταβλητών υπολογίζονται οι καινούριες πυκνότητες μέσω της μεθόδου Optimality Criteria.

Κάνοντας δοκιμή στον κώδικα θα δούμε πως το έργο που προκύπτει από το θερμικό και στατικό πρόβλημα έχουν μεγάλη διαφορά τάξης μεγέθους. Για να αντιμετωπιστεί αυτό πριν την επανάληψη βελτιστοποίησης γίνεται ο υπολογισμός του έργου και για τις δυο περιπτώσεις με πυκνότητες 1.0 για όλα τα στοιχεία. Έτσι υπολογίζεται το Cs_min και Cth_min και η σχέση που μας δίνει το συνολικό έργο μετασχηματίζεται στην παρακάτω σχέση:

C_tot = weight * C_structural/Cs_min + (1-weight) * C_thermal/Cth_min.

Προτού εφαρμόσουμε τον συνδυασμό των δύο φορτίσεων εισάγουμε κάποιες εντολές δημιουργίας φακέλων που θα αποθηκεύουν τα αποτέλεσμα που δίνει ο κώδικας.

5.2.4 ΕΙΣΑΓΩΓΗ ΕΝΤΟΛΩΝ ΔΗΜΙΟΥΡΓΙΑΣ ΦΑΚΕΛΩΝ ΓΙΑ ΤΗΝ ΑΠΟΘΗΚΕΥΣΗ ΤΩΝ ΑΠΟΤΕΛΕΣΜΑΤΩΝ

Η πρώτη αλλαγή είναι η αφαίρεση του volfrac από τις μεταβλητές εισαγωγής του προγράμματος. Η αφαίρεση αυτή έγινε έτσι ώστε η βελτιστοποίηση να γίνεται αυτόματα για παραπάνω από ένα volfrac. Για να επιτευχθεί αυτό τοποθετήθηκε μια επανάληψη έξω από την επανάληψη βελτιστοποίησης η οποία δίνει τιμές στη μεταβλητή volfrac.

Στη συνέχεια προστέθηκε η εντολή:

```
mkdir('results')
```

στην αρχή του προγράμματος , οι εντολές:

```
  vlfc = num2str(volfrac);
  vlfcfolder = ['volfrac = ' vlfc ];
```

μετά την έναρξη των επαναλήψεων για κάθε volfrac και οι εντολές:

```
  mkdir('results',vlfcfolder);
  path1          =          ['results\'          vlfcfolder
'\physical_densities.xlsx'];
  xlswrite(path1,xPhys);
  path2 = ['results\' vlfcfolder '\densities.xlsx'];
  xlswrite(path2,x);
  path3 = ['results\' vlfcfolder '\figure.png'];
  print(path3,'-dpng')
```

μετά το τέλος της επανάληψης βελτιστοποίησης για κάθε volfrac.

Έτσι κατά την εκτέλεση του προγράμματος δημιουργείται ο φάκελος results ο οποίος περιέχει φακέλους για κάθε volfrac στους οποίους αποθηκεύονται η μεταβλητή xPhys και x σε αρχεία xlsx καθώς και η εικόνα της τελικής κατασκευής σε αρχείο png. Με αυτόν τον τρόπο έχουμε τη δυνατότητα να κρατάμε την εικόνα της κατασκευής καθώς και τις πυκνότητες για κάθε ποσοστό του όγκου που χρησιμοποιεί το πρόγραμμα.

5.3 ΣΤΑΤΙΚΕΣ ΚΑΙ ΘΕΡΜΙΚΕΣ ΦΟΡΤΙΣΕΙΣ

Αν συνδυάσουμε αρχικά τη στατική φόρτιση του προηγούμενου παραδείγματος με τη θερμική φόρτιση του τελευταίου παραδείγματος με ποσοστό συμμετοχής 50%, για κάθε πρόβλημα θα πάρουμε τη παρακάτω εικόνα.

Εικόνα 31. Στατική και θερμική φόρτιση στο πάνω μέρος της κατασκευής

Η στήριξη παραμένει η ίδια και είναι κοινή και για τα δυο προβλήματα. Το αποτέλεσμα αυτού του συνδυασμού δίνει μια κατασκευή με πιο έντονη δενδροειδή μορφή. Αν συνδυάσουμε το ίδιο στατικό πρόβλημα με το θερμικό πρόβλημα της κατανεμημένης θερμικής δύναμης σε όλη την διατιθέμενη επιφάνεια τότε η εικόνα της κατασκευής παίρνει τη μορφή

Εικόνα 32. Στατική φόρτιση στο πάνω μέρος της κατασκευής και θερμική φόρτιση κατανεμημένη επιφανειακά στο φθίνουσα κατά ύψος

Η μορφή αυτή παρουσιάζει κλαδιά τα οποία εκτείνονται από την κατασκευή ,για να παραλάβουν τη θερμότητα που δέχεται η γύρω επιφάνεια και να τη μεταφέρουν

στο σημείο της στήριξης. Στο σημείο αυτό η δημιουργία των περαιτέρω κλάδων δεν είναι η επιθυμητή. Επόμενο βήμα είναι να δοκιμάσουμε να συνδυάσουμε τη θερμική φόρτιση στο άνω μέρος της κατασκευής με μια θερμική επιφανειακή φόρτιση που να ασκείται σε μια μικρή επιφάνεια στο άνω μέρος της κατασκευής. Αυτό μπορεί να πραγματοποιηθεί με χρήση των εντολών

```
iF_th = zeros(nely+1,nelx+1);
list_F = find(full(F)~=0);
list_F = floor(list_F/2-1e-8)+1;
list = setdiff(1:(nely+1):(nelx*(nely+1)+1),list_F);
iF_th(list) = 10;
% constant thermal load
iF_th = reshape(iF_th,nely+1,nelx+1);
iF_th((1:ceil(nely+1)/10),:)        =        1        +
iF_th((1:ceil(nely+1)/10),:);
```

Έτσι στο άνω μέρος ασκείτε θερμική δύναμη στους κόμβους όπου δεν ασκείτε η στατική καθώς και επιφανειακή θερμική δύναμη στο ένα δέκατο της επιφάνειας από πάνω.

Εικόνα 33. Στατικές φορτίσεις ασκούμενες στο πάνω μέρος και θερμικές ασκούμενες επιφανειακά σε ένα κομμάτι του άνω μέρους της επιφάνειας

Με αυτό τον τρόπο τα κλαδιά μειώνονται αλλά ακόμα υπάρχουν.

Προκειμένου να λάβουμε υπόψη μας τις επιφανειακές θερμικές δυνάμεις αλλά να αποφύγουμε την δημιουργία περαιτέρω κλαδιών θα πρέπει να αναπτυχτεί σχέση εξάρτησης μεταξύ της επιφανειακής θερμικής φόρτισης και της πυκνότητα των στοιχείων της κατασκευής. Με αυτό τον τρόπο αν κάποιο στοιχείο αποκτήσει πυκνότητα θα δεχθεί θερμική δύναμη η οποία θα πρέπει να μεταφερθεί στη στήριξη ,ενώ αν η πυκνότητα του γίνει μηδέν δεν θα υπάρχει δύναμη και έτσι ούτε και η ανάγκη δημιουργίας κλάδων για την παραλαβή της.

5.4 ΕΦΑΡΜΟΓΗ ΜΑΖΙΚΩΝ ΔΥΝΑΜΕΩΝ

Με τη θεώρηση δυνάμεων εξαρτώμενων από τη μάζα (πυκνότητα) F(x), η εξίσωση του έργου των δυνάμεων παίρνει τη μορφή

$$C(x) = \mathrm{F}^{T}(\mathrm{x})\,\mathrm{U} \tag{5.9}$$

Στην περίπτωση των θερμικών δυνάμεων το U γίνεται T.

Η παράγωγος της συνάρτησης C(x) όμως διαφέρει από εκείνη που χρησιμοποιήθηκε στο κεφάλαιο 3 διότι τώρα η δύναμη F(x) εξαρτάται από την πυκνότητα. Για να υπολογίζουμε την παράγωγο, θα χρησιμοποιήσουμε την ίδια μέθοδο όπως και στο κεφάλαιο 3 ,απαλείφοντας την παράγωγο των μετακινήσεων ως προς τις πυκνότητες. Ξεκινώντας από τη σχέση:

$$C(x) = F(x)^{\mathrm{T}}U = F(x)^{\mathrm{T}}U - \Lambda(\mathrm{KU} - \mathrm{F}) \qquad \forall\Lambda \tag{5.10}$$

Η παράγωγος είναι

$$\begin{aligned}\frac{dC}{dx_e} &= \frac{dF(x)}{dx_e}U + F(x)*\frac{dU}{dx_e} - \Lambda\frac{dK}{dx_e}U - \Lambda K\frac{dU}{dx_e} + \Lambda\frac{dF(x)}{dx_e} = \\ &= -\Lambda\frac{dK}{dx_e}U + 2\frac{dF(x)}{dx_e}U + (F(x) - \Lambda K)\frac{dU}{dx_e}\end{aligned} \tag{5.11}$$

Θεωρώντας Λ=U προκύπτει

$$\frac{dC}{dx_e} = -U^{T}\frac{dK}{dx_e}U + 2\frac{dF(x)}{dx_e}U \tag{5.12}$$

Επομένως παρατηρούμε ότι εκτός από το κομμάτι της παραγώγου που υπήρχε τώρα προκύπτει και ένα νέο μέλος που εξαρτάται από τη παράγωγο της δύναμης. Η δύναμη που θα χρησιμοποιήσουμε έχει τη μορφή

$$F(x) = a * x \tag{5.13}$$

Όπου α είναι ένας συντελεστής.

Για την εφαρμογή αυτών των δυνάμεων ως στατικές φορτίσεις θεωρούμε ως α=-1 για τις κατακόρυφες δυνάμεις και α=-0,3 για τις οριζόντιες. Το αρνητικό πρόσημο υποδηλώνει ότι ασκούνται προς τα κάτω. Αυτές οι δυνάμεις μπορούν να θεωρηθούν ως βαρυτικές δυνάμεις της κατασκευής και ανάλογα το υλικό να πάρουν και την αντίστοιχη τιμή. Για την εφαρμογή τους ως θερμικές φορτίσεις επιλέγουμε α=1. Στις θερμικές φορτίσεις το πρόσημο δεν έχει κάποιο νόημα. Αυτές οι θερμικές φορτίσεις μπορούν να θεωρηθούν ως η θερμότητα την οποία παραλαμβάνει η κατασκευή λόγο της επιφάνειας την οποία καταλαμβάνει.

5.4.1 ΕΦΑΡΜΟΓΗ ΜΑΖΙΚΩΝ ΔΥΝΑΜΕΩΝ ΣΤΟΝ ΚΩΔΙΚΑ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗΣ ΤΟΠΟΛΟΓΙΑΣ TOP88

Για την εφαρμογή των μαζικών δυνάμεων στον κώδικα βελτιστοποίησης τοπολογίας θα πρέπει να δημιουργηθεί ένα μητρώο F2 το οποίο θα αντιπροσωπεύει το κομμάτι της δύναμης που ασκείται λόγω της πυκνότητας του κάθε στοιχείου. Το μητρώο αυτό θα περιέχει για κάθε βαθμό ελευθερίας τον μέσο όρο των a*x των στοιχείων που βρίσκονται γύρο του, όπου α=-1 για κατακόρυφο βαθμό και α=-0,3 για οριζόντιο.

Για να δημιουργηθεί το μητρώο αυτό θα χρειαστεί για κάθε ένα βαθμό ελευθερίας, να βρίσκονται τα στοιχεία τα οποία βρίσκονται γύρω του, να υπολογίζεται το γινόμενο του a*x και μετά να λαμβάνεται ο μέσος όρος για όλα αυτά τα στοιχεία. Η διαδικασία αυτή πρέπει να πραγματοποιείται σε κάθε επανάληψη επειδή η πυκνότητα του κάθε στοιχείου αλλάζει. Για να πραγματοποιηθεί ο υπολογισμός αυτός θα μπορούσαμε να δημιουργήσουμε μια διπλή επανάληψη

```
for i=1:nely

   for j=1:nelx



   end

end
```

Η οποία με αυτό το τρόπο θα περάσει από όλα τα στοιχεία της κατασκευής. Εντός της επανάληψης υπολογίζουμε το κάθε στοιχείο με την εντολή

```
k2 = (j-1)*(nely+1)+i;
```

και τους οριζόντιους και κατακόρυφους βαθμούς ελευθερίας του στοιχείου αυτού με τις εντολές

verticaldofs = [2*(k2+1);2*(k2+1+(nely+1));2*(k2+1+nely);2*k2];

horizontaldofs = [2*(k2+1)-1;2*(k2+1+(nely+1))-1;2*(k2+1+nely)-1;2*k2-1];

Για κάθε ένα από αυτούς τους βαθμούς ελευθερίας θα πρέπει να προσθέσουμε στο μητρώο δυνάμεων F2 το γινόμενο

-1*x για κατακόρυφο

-0,3*x για οριζόντιο

και να έχουμε ένα μετρητή counter, ο οποίος θα μετρήσει σε κάθε βαθμό ελευθερίας πόσες φορές θα προστεθεί κάποιο από τα παραπάνω γινόμενα έτσι ώστε να πάρουμε το μέσο όρο έξω από την επανάληψη.

F2(verticaldofs,1) = F2(verticaldofs,1)-ones(max(size(verticaldofs)),1)*xPhys_tmp(i,j);

F2(horizontaldofs,1) = 0.3 * F2(verticaldofs,1);

counter2(verticaldofs,1) = counter2(verticaldofs,1)

+ones(max(size(verticaldofs)),1);

counter2(horizontaldofs,1) = counter2(horizontaldofs,1)

+ones(max(size(horizontaldofs)),1);

Εκτός της επανάληψης παίρνουμε το μέσο όρο του μητρώου F2 για κάθε βαθμό ελευθερίας με την εντολή

F2 = F2./counter2;

και μετά προσθέτουμε το μητρώο F2 στο μητρώο των δυνάμεων F.

Για τη περίπτωση της θερμικής φόρτισης ,επειδή έχουμε μόνο ένα βαθμό ελευθερίας ανά κόμβο, αρκεί να αντικαταστήσουμε τις εντολές που υπολογίζουν τους βαθμούς ελευθερίας του στοιχείου και τις δυνάμεις με τις εντολές

nodes = [k2+1;k2+1+(nely+1);k2+1+nely;k2];

F_th(nodes,1) = F_th(nodes,1)+ones(max(size(nodes)),1)

*xPhys_tmp(i,j);

Η διαδικασία αυτή θα μας δώσει τις μαζικές δυνάμεις όμως θα πρέπει να πραγματοποιείται εντός κάθε επανάληψης. Προκειμένου να την αποφύγουμε

μπορούμε να δημιουργήσουμε εκτός επανάληψης ένα μητρώο το οποίο θα περιέχει αυτή τη σχέση α=-1,-0,3 ανάμεσα σε κάθε βαθμό ελευθερίας και κάθε στοιχείο. Το μητρώο αυτό θα έχει τη μορφή

$$sF2 = \begin{matrix} x_1 & x_2 & \cdots & x_n \end{matrix} \begin{bmatrix} -0.3 & 0 & \cdots & 0 \\ -1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & -1 \end{bmatrix} \begin{matrix} \beta\alpha\theta\mu\acute{o}\varsigma\varepsilon\lambda\varepsilon\upsilon\theta\varepsilon\rho\acute{\iota}\alpha\varsigma 1 \\ \beta\alpha\theta\mu\acute{o}\varsigma\varepsilon\lambda\varepsilon\upsilon\theta\varepsilon\rho\acute{\iota}\alpha\varsigma 2 \\ \vdots \\ \beta\alpha\theta\mu\acute{o}\varsigma\varepsilon\lambda\varepsilon\upsilon\theta\varepsilon\rho\acute{\iota}\alpha\varsigma n \end{matrix} \quad (5.14)$$

Έχοντας δημιουργήσει αυτό το μητρώο εκτός επανάληψης ,έχουμε τη δυνατότητα να υπολογίσουμε τις μαζικές δυνάμεις εντός της επανάληψης βελτιστοποίησης, μέσω της πράξης:

```
F2 = (sF2*xPhys(:))./count(:);
```

όπου count είναι ένα μητρώο στήλη που περιέχει τον αριθμό των στοιχείων κάθε γραμμής του μητρώου sF2. Η πράξη sF2*x(:) μας δίνει ένα μητρώο στήλη το οποίο περιέχει το άθροισμα των γινόμενων της κάθε γραμμής του μητρώου sF2 με τη στήλη του μητρώο x(:). Το πλεονέκτημα αυτού του τρόπου υπολογισμού είναι ότι η μόνη πράξη που πραγματοποιείται εντός της επανάληψης είναι η παραπάνω, με την κατασκευή του μητρώου sF2 να γίνεται εκτός. Για να εφαρμοστεί και για τις θερμικές φορτίσεις τοποθετούμε στη σχέση βαθμού-στοιχείου, το α=1.

Η επαναληπτική διαδικασία που οδηγεί στη δημιουργία του μητρώο sF2 παρατίθεται παρακάτω.

Η βασική ιδέα έχει παρθεί από τη δημιουργία του μητρώου H για το φίλτρο. Επομένως θα χρησιμοποιήσουμε μια επανάληψη προκειμένου να δώσουμε σε τρία μητρώα iF,jF,sF τιμές με τρόπο έτσι ώστε sF2(iF(:),jF(:)) = sF(:). Η επανάληψη θα γίνει για κάθε βαθμό ελευθερίας.

```
for i1 = 1:(nelx+1)
      for j1 = 1:(nely+1)
          n = (i1-1)*(nely+1)+j1;
          e1 = 2*n-1;
          nelxel = max(i1-1,1);
          nelyel = max(j1-1,1);
          for i2 = nelxel:min(i1,nelx)
              for j2 = nelyel:min(j1,nely)
                  e2 = (i2-1)*nely+j2;
                  iF(k) = e1;
                  iF(k+1) = e1+1;
                  jF(k) = e2;
```

```
                jF(k+1) = e2;
                sF(k) = -0.3/(nelx*nely);
                sF(k+1) = -1/(nelx*nely);
                k = k+2;
                count(e1) = count(e1) + 1;
                count(e1+1) = count(e1+1) + 1;
            end
        end


    end
end
```

όπου e1 είναι ο βαθμός ελευθερίας κάθε φορά. Οι μεταβλητές nelxel και nelyel μας δείχνουν τον αριθμό των στοιχείων που βρίσκονται γύρο από τον κάθε βαθμό ελευθερίας. Στη δεύτερη επανάληψη με τη χρήση των nelxel και nelyel κινούμαστε σε όλα τα στοιχεία e2 γύρω από το βαθμό ελευθερίας και δίνουμε τις κατάλληλες τιμές στα iF,jF,sF. Επίσης υπολογίζουμε το μητρώο count, που μετράει τον αριθμό των στοιχείων που έχει γύρω του ο κάθε βαθμός ελευθερίας. Οι τιμές -1 και -0,3 διαιρούνται με τον αριθμό των στοιχείων προκειμένου να μειωθούν. Αυτό συμβαίνει επειδή διαφορετικά οι συγκεντρωμένες δυνάμεις που ασκούνται με τιμές -1 στο άνω μέρος της κατασκευής δεν θα επηρέαζαν καθόλου τη κατασκευή λόγο του μικρού μεγέθους τους. Τέλος για να υπολογιστεί το τελικό μητρώο sF2 χρησιμοποιείτε η εντολή sparse εκτός της επανάληψης

```
sF2 = sparse(iF,jF,sF);
```

Στη συνέχεια θα πρέπει να υπολογιστεί το καινούριο κομμάτι της παραγώγου dc/dx

$$2\frac{dF(x)}{dx_e}U \tag{5.15}$$

Για να πραγματοποιηθεί αυτό χρειαζόμαστε ένα μητρώο με τις παραγώγους της δύναμης ως προς το x. Το μητρώο αυτό περιέχει το συντελεστή α για κάθε ένα βαθμό ελευθερίας. Ο υπολογισμός αυτού του μητρώου μπορεί να πραγματοποιηθεί εντός της επανάληψης, που παρουσιάστηκε παραπάνω, στο κομμάτι που τελειώνει η διπλή εσωτερική επανάληψη μέχρι να τελειώσει η διπλή εξωτερική επανάληψη. Οι εντολές που πρέπει να προστεθούν είναι

```
dcF(e1) = -0.3/(nelx*nely);
dcF(e1+1) = -1/(nelx*nely);
```

Έπειτα υπολογίζεται το μητρώο Forcece μέσω της εντολής

```
Forcece = reshape(sum(2*dcF(edofMat).
*U(edofMat),2)/8,nely,nelx);
```

και προστίθεται στη συνολική παράγωγο.

```
dc = dc + Forcece;
```

Για το θερμικό πρόβλημα οι αλλαγές λόγο του ενός βαθμού ελευθερίας ανά κόμβο είναι

```
for i1 = 1:(nelx+1)
      for j1 = 1:(nely+1)
          n_th = (i1-1)*(nely+1)+j1;
          e1 = n_th;
          nelxel = max(i1-1,1);
          nelyel = max(j1-1,1);
          for i2 = nelxel:min(i1,nelx)
              for j2 = nelyel:min(j1,nely)
                  e2 = (i2-1)*nely+j2;
                  iF_th(k) = e1;
                  jF_th(k) = e2;
                  sF_th(k) = 1/(nelx*nely);
                  k = k+1;
                  count_th(e1) = count_th(e1) + 1;
              end
          end
          dcF_th(e1) = 1/(nelx*nely);
      end
end
sF2_th = sparse(iF_th,jF_th,sF_th);
```

και

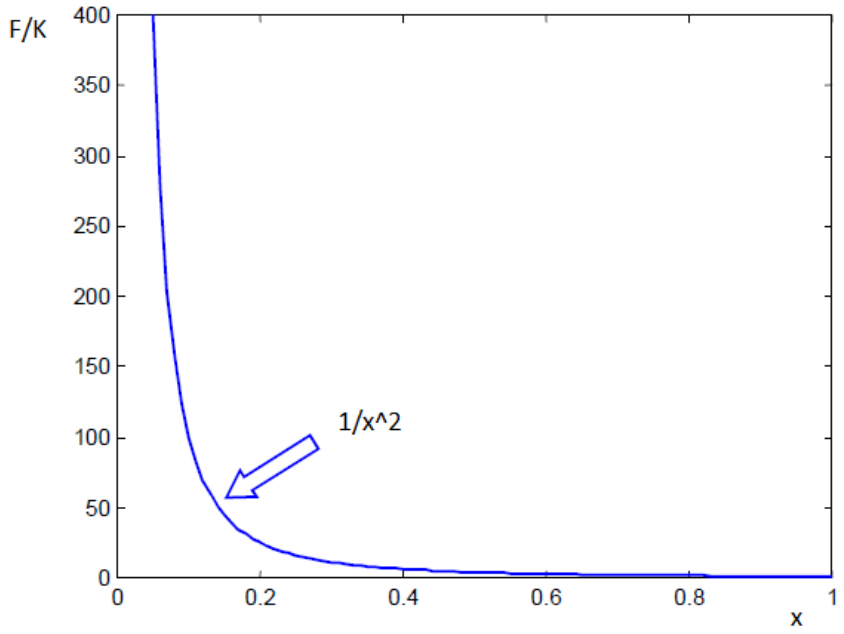
```
Forcece_th = reshape(sum(2*dcF_th(edofMat_th).
*U_th(edofMat_th),2)/4,nely,nelx);
dc_th = dc_th + Forcece_th;
```

Με αυτό τον τρόπο μπορούμε να λάβουμε υπόψη μας δυνάμεις οι οποίες εξαρτώνται από τη μάζα της κατασκευής.

Κάνοντας τις παραπάνω αλλαγές στον κώδικα έχουμε εφαρμόσει τις δυνάμεις που εξαρτώνται από τη μάζα της κατασκευής. Όμως επειδή ο λόγος δύναμης προς δυσκαμψία όταν εφαρμόζονται και οι μαζικές δυνάμεις δίνει:

$$\frac{F}{K} = \frac{x}{x^3} C$$

Αυτό έχει ως αποτέλεσμα για μικρές πυκνότητες ο λόγος αυτός να απειρίζεται.



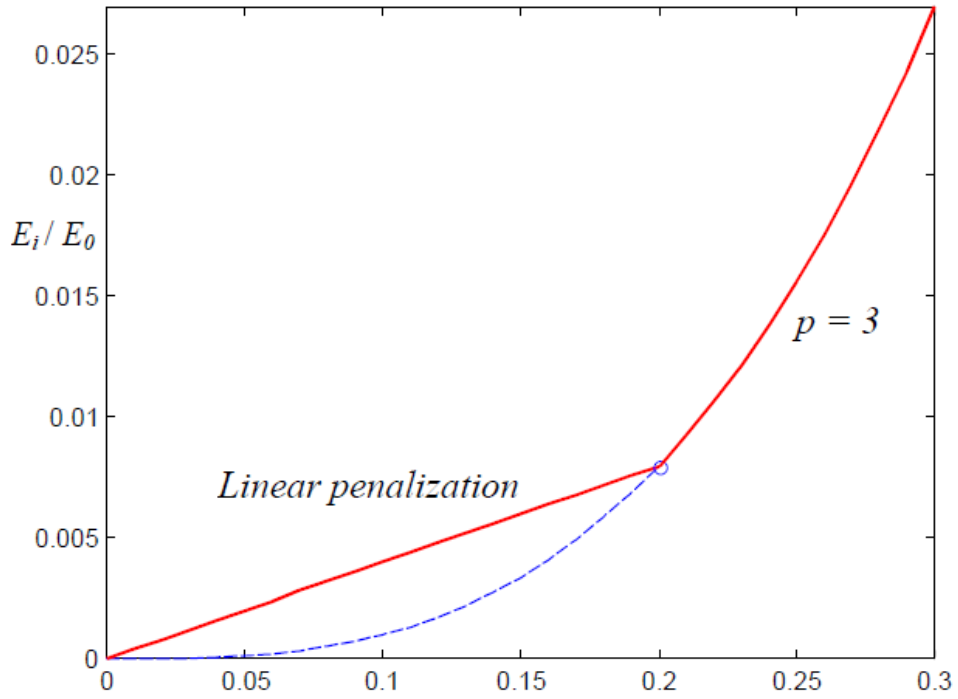
Εικόνα 34.Διάγραμμα F/K-1/x^2

Για να αντιμετωπίσουμε αυτό το πρόβλημα θα πρέπει να τροποποιήσουμε τη μέθοδο SIMP.

Η τροποποίηση που υιοθετήσαμε προήρθε από τη δημοσίευση TOPOLOGY OPTIMIZATION WITH SELF-WEIGHT LOADING: UNEXPECTED PROBLEMS AND SOLUTIONS [9]. Στη δημοσίευση αυτή προκειμένου να αντιμετωπιστεί το πρόβλημα του γκρι υλικού προτείνεται η εξής αλλαγή στη μέθοδο SIMP.

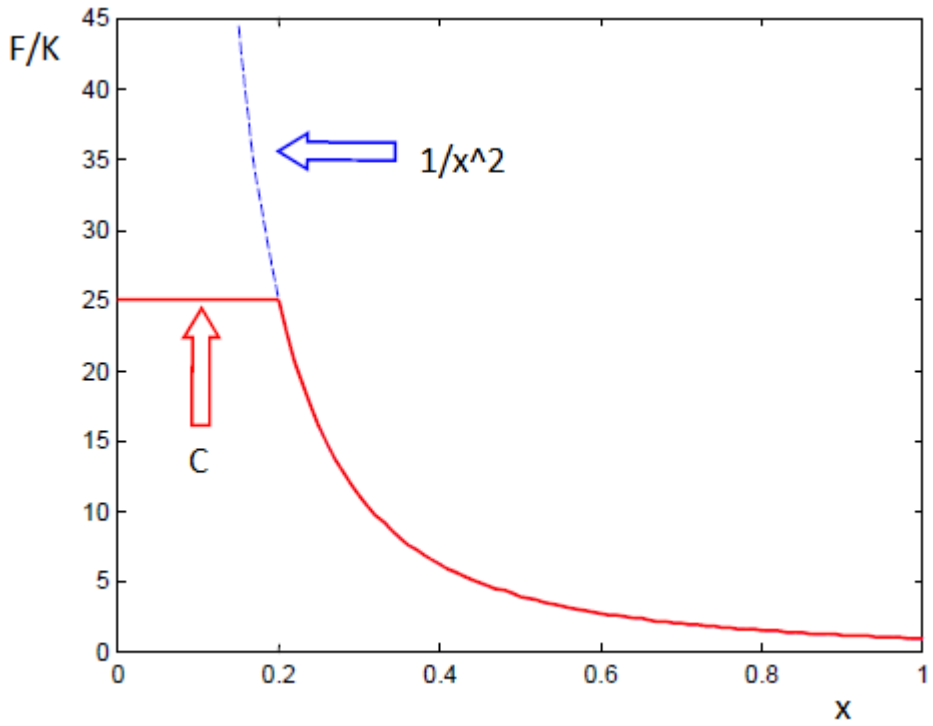
$$E = \begin{cases} E_{\min} + x^P (\mathrm{E}_0 - \mathrm{E}_{\min}) & \alpha\nu \quad x > 0.2 \\ E_{\min} + 0.2^{P-1} x (\mathrm{E}_0 - \mathrm{E}_{\min}) & \alpha\nu \quad x < 0.2 \end{cases} \tag{5.16}$$

Επομένως με αυτό τον τρόπο η εκθετική σχέση πυκνότητας-μέτρου ελαστικότητας γίνεται γραμμική για πυκνότητες κάτω του 0,2.



Εικόνα 35. Διάγραμμα λόγου E/E0 και x.

Αυτό έχει ως αποτέλεσμα ο λόγος δύναμης προς δυσκαμψίας να δίνει σταθερό αριθμό για x<0.2



Για να εφαρμοστεί η αλλαγή αυτή στον κώδικα πρέπει να αντικατασταθούν οι εντολές

```
sK = reshape(KE(:)*(Emin+(xPhys(:))'.^penal*(E0-Emin)),64*nelx*nely,1);

c = sum(sum((Emin+xPhys.^penal*(E0-Emin)).*ce));

dc =-penal*(E0-Emin)*xPhys.^(penal-1).*ce;
```

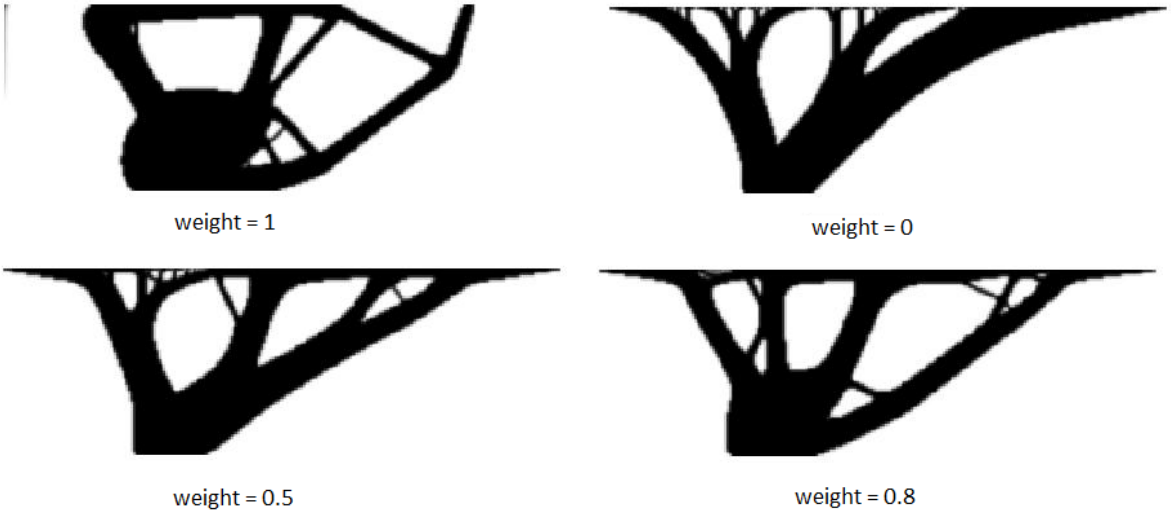
Με τις εντολές

```
sK = reshape(KE(:)*(Emin+((xPhys(:)>=rho_crit)'.*(xPhys(:)'.^penal)+(xPhys(:)<rho_crit)'.*(rho_crit^(penal-1)*xPhys(:)'))*(E0-Emin)),64*nelx*nely,1);

c = sum(sum((Emin+((xPhys>=rho_crit).*(xPhys.^penal)+(xPhys<rho_crit).*(rho_crit^(penal-1)*xPhys))*(E0-Emin)).*ce));

dc =-(E0-Emin)*((xPhys>=rho_crit).*(penal*xPhys.^(penal-1))+(xPhys<rho_crit).*(rho_crit^(penal-1))).*ce;
```

Αν ο αρχικός κώδικας δεχθεί τις παραπάνω αλλαγές και δοκιμαστεί θα παρατηρήσουμε ότι παρουσιάζει σφάλμα στην Optimality Criteria στο σημείο που υπολογίζει τις καινούριες πυκνότητες. Αυτό συμβαίνει επειδή η παράγωγος dc/dx αλλάζει πρόσημο ανάλογα με το στοιχείο. Όταν η παράγωγος γίνεται θετική τότε η OC αδυνατεί να δώσει το σωστό αποτέλεσμα. Θετική παράγωγος σημαίνει ότι για να ελαττωθεί η αντικειμενική συνάρτηση θα πρέπει η πυκνότητα σε εκείνο το σημείο να μειωθεί. Επομένως επεμβαίνουμε στη μέθοδο και όταν η παράγωγος γίνεται θετική μηδενίζουμε την αντίστοιχη πυκνότητα. Με τις αλλαγές αυτές το αποτέλεσμα που παίρνουμε φαίνονται παρακάτω:



Εικόνα 36. Weight = 1, περίπτωση μόνο στατικών φορτίσεων. Weight = 0 περίπτωση μόνο θερμικών φορτίσεων. Weight = 0.5, περίπτωση όπου λαμβάνονται θερμικές και στατικές φορτίσεις κατά 50%. Weight = 0.8, περίπτωση όπου οι στατικές φορτίσεις λαμβάνονται υπόψη 80% ενώ οι θερμικές 20%

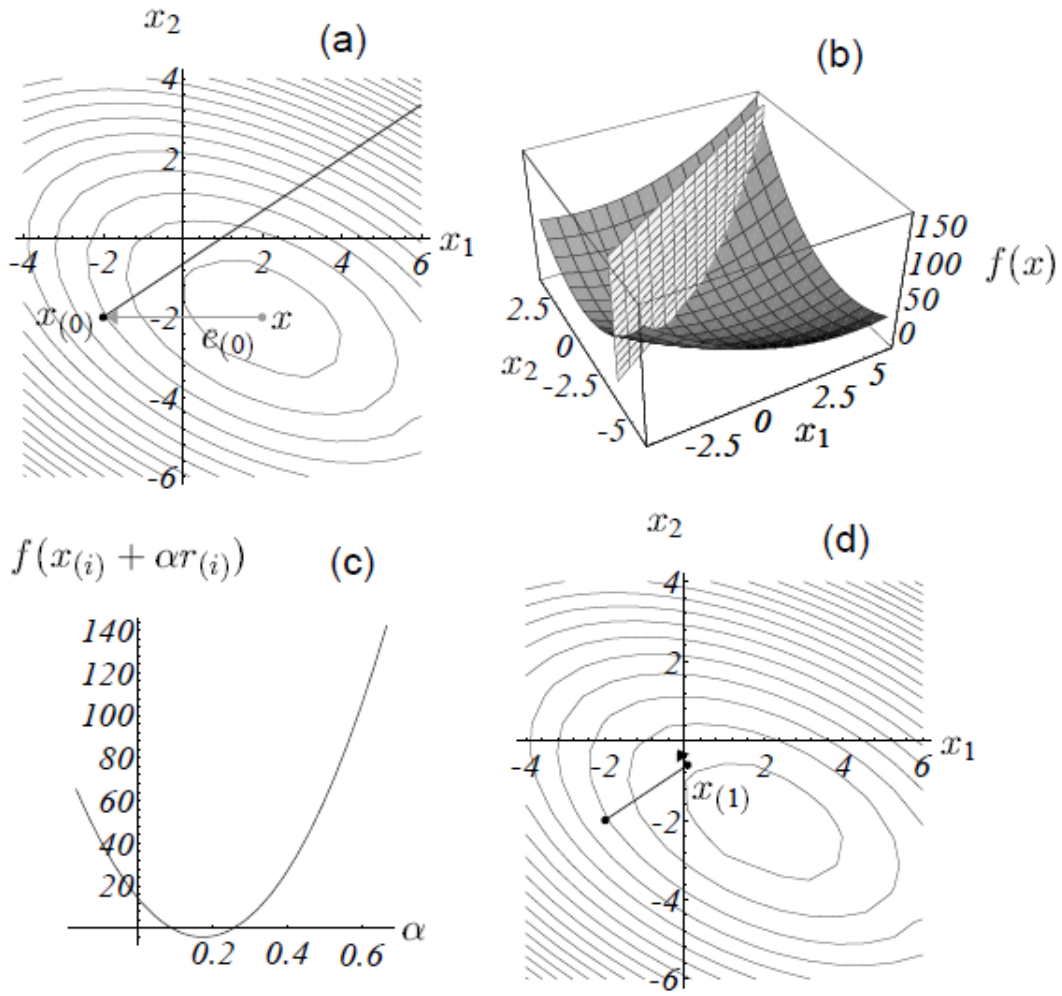
Προκειμένου να εξετάσουμε πόσο η αλλαγή στην OC επηρεάζει το τελικό σχήμα θα εφαρμόσουμε μια διαφορετική μέθοδο προσδιορισμού του βέλτιστου σχήματος. Εμείς επιλέξαμε την Gradient Descent.

5.4.2 ΘΕΩΡΗΤΙΚΗ ΕΦΑΡΜΟΓΗ ΜΕΘΟΔΟΥ GRADIENT DESCENT

Για την εφαρμογή της μεθόδου Gradient descent θα πρέπει να αφαιρέσουμε από τον κώδικα το κομμάτι της OC και να αλλάξουμε τη μορφή του, ώστε οι πυκνότητες να ανανεώνονται μέσω της Gradient Descent. Σύμφωνα με τη μέθοδο της Gradient Descent οι καινούριες πυκνότητες δίνονται από τον τύπο

$$x_{new} = x - m\frac{dC}{dx} \tag{5.17}$$

όπου m είναι το βήμα με το οποίο κινούμαστε προς την αρνητική κατεύθυνση της παραγώγου της συνάρτησης C. Για προβλήματα όπως είναι ο προσδιορισμός της λύσης μιας γραμμικής εξίσωσης Ax=B το βήμα μπορεί να προσδιοριστεί ως η απόσταση η οποία ελαχιστοποιεί τη συνάρτηση προς τη κατεύθυνση της παραγώγου. Όμως στα προβλήματα της βελτιστοποίησης κατασκευών δεν είναι δυνατόν να προσδιορίσουμε το βήμα με αυτό τον τρόπο. Επομένως ξεκινάμε με ένα βήμα και ελέγχουμε σε κάθε επανάληψη αν η συνάρτηση C μειώνεται. Αν σε κάποια επανάληψη η συνάρτηση αυξηθεί αντί να μειωθεί αυτό σημαίνει ότι το βήμα είναι πολύ μεγάλο και μειώνεται στο μισό.



Εικόνα 37.Για την επίλυση ενός γραμμικού συστήματος Ax=B γίνεται η θεώρηση ελαχιστοποίησης της παράγουσας συνάρτησης F(x)=1/2Αχ^2-Βχ+Γ όπως φαίνεται στην εικόνα b. Με τη χρήση της Gradient Descent οι καινούριες μεταβλητές x υπολογίζονται από τις παλιές ως xnew = x-m*dc/dx. Ο υπολογισμός του βήματος m γίνεται προσδιορίζοντας το σημείο στο οποίο η ευθεία που δημιουργείται από τη τομή της συνάρτησης f(x) και του επιπέδου που δείχνει τη κατεύθυνση μείωσης της παραγώγου.

Το πρόβλημα βελτιστοποίησης είναι

$$\begin{aligned} & C(x) = F^T U \\ & s.t. \\ & KU = F \\ & a * x = V \end{aligned} \tag{5.18}$$

Επομένως η αντικειμενική συνάρτηση προκειμένου να συμπεριλάβει τον περιορισμό παίρνει τη μορφή

$$C(x) = F^T U - \lambda(\mathrm{a}^* \mathrm{x} - \mathrm{V}) \tag{5.19}$$

και η παράγωγος της είναι

$$\frac{dC}{dx} = -U^T \frac{dK}{dx} U + 2\frac{dF}{dx} U - \lambda a \tag{5.20}$$

Επομένως οι καινούριες πυκνότητες προκύπτουν από τη σχέση

$$x_{new} = x - m(-\mathrm{U}^T \frac{dK}{dx} \mathrm{U} + 2\frac{dF}{dx} U - \lambda a) \tag{5.21}$$

Με αυτό τον τρόπο λαμβάνουμε υπόψη μας και τον περιορισμό του όγκου στον υπολογισμό των νέων πυκνοτήτων.

5.4.3 ΕΦΑΡΜΟΓΗ ΜΕΘΟΔΟΥ GRADIENT DESCENT ΣΤΟΝ ΚΩΔΙΚΑ TOP88

Η δομή του νέου κώδικα θα είναι η εξής:

-Αρχικά γίνεται ο ορισμός του θερμικού και στατικού προβλήματος όπως και στο προηγούμενο.

-Έπειτα γίνεται ο υπολογισμός του φίλτρου όπως και στον αρχικό κώδικα.

-Σε αυτό το κομμάτι γίνεται ένας πρώτος υπολογισμός της συνολικής αντικειμενικής συνάρτησης C_tot και της παραγώγου της dc_tot προκειμένου να αξιοποιηθεί παρακάτω.

Εισερχόμαστε στην επανάληψη βελτιστοποίησης

-Υπολογίζονται οι νέες πυκνότητες μέσω της παρακάτω διαδικασίας.

Αρχικά υπολογίζεται το πρώτο κομμάτι των νέων πυκνοτήτων από τη σχέση

```
xnew_in = x - m*dc_tot;
```

Όπου dc_tot είναι η παράγωγος της συνάρτησης C χωρίς το συντελεστή λ.

Αφού υπολογιστεί το πρώτο κομμάτι μετά ,μέσω μιας επανάληψης που θα προσδιορίζει το λ, υπολογίζονται οι νέες πυκνότητες

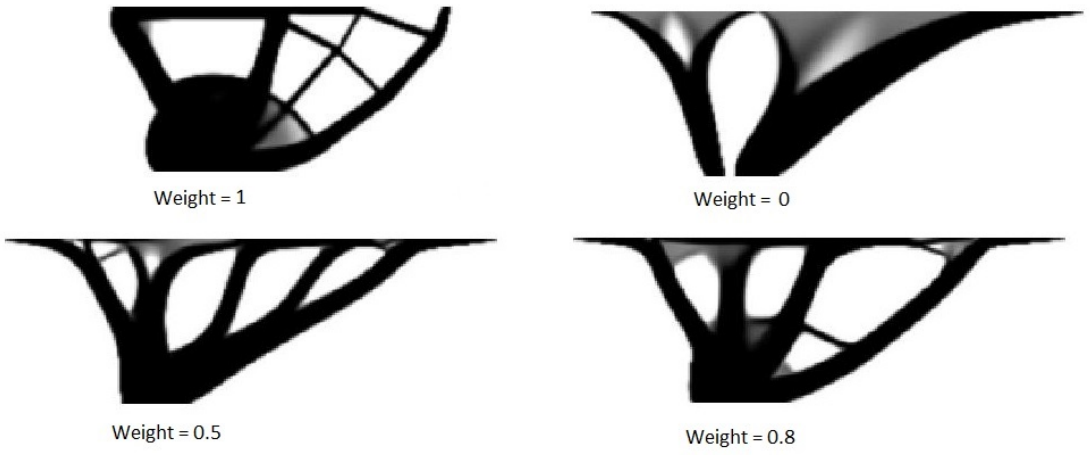
```
xnew = xnew_in - lmid*dv;
```

έτσι ώστε το σύνολο των νέων πυκνοτήτων να ικανοποιεί τον περιορισμό του όγκου.

-Στη συνέχεια υπολογίζεται η νέα αντικειμενική συνάρτηση C_tot και η παράγωγος της dc_tot.

-Τέλος, γίνεται έλεγχος αν η καινούρια τιμή της αντικειμενικής συνάρτησης είναι μικρότερη από τη παλιά. Αν είναι τότε αυξάνεται το βήμα κατά 1,1 και η επανάληψη βελτιστοποίησης επαναλαμβάνεται. Αν δεν είναι τότε η πυκνότητες δεν γίνονται δεκτές, το βήμα μειώνεται στο μισό και η επανάληψη επαναλαμβάνεται με τις προηγούμενες πυκνότητες.

Με αυτές τις αλλαγές τα αποτελέσματα που μας δίνει η εφαρμογή της μεθόδου Gradient Descent απεικονίζονται παρακάτω:



Εικόνα 38. Εικόνα 14. Weight = 1, περίπτωση μόνο στατικών φορτίσεων. Weight = 0 περίπτωση μόνο θερμικών φορτίσεων. Weight = 0.5, περίπτωση όπου λαμβάνονται θερμικές και στατικές φορτίσεις κατά 50%. Weight = 0.8, περίπτωση όπου οι στατικές φορτίσεις λαμβάνονται υπό

Παρατηρούμε ότι τα αποτελέσματα της μεθόδου Gradient Descent δίνουν περισσότερο γκρι υλικό από την OC. Αυτό συμβαίνει επειδή με την αλλαγή που κάναμε στην OC στα σημεία που η μέθοδος θέλει να μειώσει τη πυκνότητα, η πυκνότητα μηδενίζεται. Το σχήμα παρ' όλα αυτά και στις δύο μεθόδους έχει μικρή διαφορά επομένως με την αλλαγή της OC δεν αποκλίνουμε πολύ από το βέλτιστο σχήμα που επιζητούμε για την κατασκευή μας.

5.6 ΕΙΣΑΓΩΓΗ ΣΥΝΑΡΤΗΣΗΣ ΑΠΟΦΥΓΗΣ ΥΛΙΚΟΥ

Προκειμένου να μπορούμε να καθορίζουμε οι ίδιοι τα σημεία στα οποία δεν επιθυμούμε να τοποθετηθεί υλικό έχουμε τη δυνατότητα να χρησιμοποιήσουμε την επιλογή που μας δίνουν οι σημειώσεις του κώδικα top88 [1]. Η εφαρμογή αναγράφεται στο κομμάτι για τα παθητικά στοιχεία. Προκειμένου να ορίσουμε ότι μια περιοχή δεν θα πάρει υλικό ,αρκεί να δημιουργήσουμε έναν πίνακα passive. Στον πίνακα αυτό θα περιέχεται η τιμή ένα για τα στοιχεία που δεν θέλουμε να τοποθετηθεί υλικό και μηδέν για τα υπόλοιπα. Με αυτό τον τρόπο αν εισάγουμε μέσα στην OC μια εντολή η οποία να δίνει στα στοιχεία, των οποίων ο πίνακας passive είναι ένα, την τιμή μηδέν στη πυκνότητα τους θα πάρουμε ως αποτέλεσμα τη δημιουργία περιοχών που δεν έχουν υλικό.

Για να εφαρμόσουμε τα παραπάνω στον κώδικα αρκεί να εισάγουμε τις παρακάτω γραμμές:

```
%% PASSIVE
 passive = zeros(nely,nelx);
 for i=1:nelx
     for j=1:nely
         if sqrt((j-x)^2+(i-y)^2)<r
             passive(j,i) = 1;
         end
     end
 end
```

όπου x,y είναι οι συντεταγμένες του κέντρου του κύκλου των στοιχείων που δεν θέλουμε να τοποθετηθεί υλικό και r είναι η ακτίνα του κύκλου αυτού.

```
xPhys(passive==1) = 0;
```

Η εντολή αυτή τοποθετείτε αμέσως πριν τον έλεγχο του όγκου της κατασκευής με τον επιτρεπόμενο. Με τις παραπάνω αλλαγές για ένα κύκλο με κέντρο τις συντεταγμένες (nely,2.5/6nelx) και ακτίνα nely/3 η κατασκευή παίρνει τη μορφή:

Εικόνα 39. Το κυκλικό κομμάτι που δεν καλύπτεται από υλικό είναι το κομμάτι όπου το μητρώο passive έχει τη τιμή ένα.

Μια τέτοια εφαρμογή είναι θα ήταν χρήσιμη σε περιπτώσεις στεγάστρων τα οποία δεν επιθυμούμε να έχουν υλικό στο κάτω μέρος, κοντά στη στήριξη. Όμως παρατηρούμε πως παρά το γεγονός ότι δεν τοποθετείτε υλικό εντός του κύκλου τα στοιχεία που συνορεύουν με αυτόν παίρνουν κάποιο υλικό. Αυτό συμβαίνει επειδή στα στοιχεία από τα οποία αποτελείτε ο κύκλος του "επιβάλλεται" η τιμή μηδέν. Για να αποφύγουμε αυτό το πρόβλημα αντί να χρησιμοποιήσουμε τον πίνακα passive θα δημιουργήσουμε έναν πίνακα avoid. Στον πίνακα αυτό θα τοποθετήσουμε την τιμή 1 για όλα τα στοιχεία εκτός αυτών που θέλουμε να μην τοποθετηθεί υλικό. Για τα στοιχεία αυτά δίνουμε πολύ μεγάλη τιμή στο avoid η

οποία όσο απομακρυνόμαστε από το κεντρικό στοιχείο μειώνεται εκθετικά μέσο της συνάρτησης της κανονικής κατανομής σε δύο διαστάσεις.

$$f = c * e^{\frac{-(i-y)^2-(j-x)^2}{2r^2}} \tag{5.22}$$

όπου c είναι η μεγάλη τιμή που θέλουμε να πάρουν τα στοιχεία γύρω από το κέντρο, x,y είναι οι συντεταγμένες του κέντρου γύρω από το οποίο δεν τοποθετείται υλικό και r είναι η ακτίνα του κύκλου, με ρόλο τυπικής απόκλισης της κανονικής κατανομής. Ο πίνακας αυτός θα πρέπει να πολλαπλασιάζεται με την παράγωγο του όγκου dv, με αυτό τον τρόπο τα στοιχεία με μεγάλο dv δεν θα υφίστανται μεγάλη αλλαγή στην OC. Επίσης θα πρέπει να πολλαπλασιάζεται με το μητρώο των πυκνοτήτων έτσι ώστε τα στοιχεία που δεν θέλουμε να πάρουν υλικό να έχουν τεράστιες πυκνότητες με αποτέλεσμα να τους αφαιρείτε όλο το υλικό. Για να εφαρμοστεί η παραπάνω μεθοδολογία εισάγονται οι εξής γραμμές στο κώδικα.

```
%% f(x,y) generation
% define center of circle
radius=7;
coef = 90;
xc = floor(25/60*(nelx))+1;
yc = (nely)-radius;
sdv = radius;
i_man_constr = 1;
avoid = ones(nely,nelx);
 for i=1:(nely)
     for j=1:(nelx)
         if (i_man_constr == 1)
            avoid(i,j) = (1.0+coef * exp((-(i-yc)^2-(j-
                        xc)^2)/(2*sdv^2)));
         end
     end
 end
```

Οι συντεταγμένες του κέντρου του κύκλου βρίσκονται στο σημείο x_c,y_c και η ακτίνα είναι 7 στοιχεία. Στο σημείο που υπολογίζεται ο όγκος ,η εντολή

```
dv = ones(nely,nelx);
```

παίρνει τη μορφή

```
dv = ones(nely,nelx).*avoid;
```

και το φίλτρο που επιβάλλεται εντός της OC παίρνει τη μορφή

```
if ft == 1
      xPhys = xnew;
      xPhys_vol = xnew.*avoid;
    elseif ft == 2
```

```
      xPhys(:) = (H*(xnew(:).*avoid(:)))./Hs;
      xPhys_vol(:) = (H*(xnew(:).*avoid(:)))./Hs;
    end
```

όπου xPhys_vol είναι οι πυκνότητες με τις οποίες θα γίνει ο έλεγχος

```
if sum(xPhys_vol(:)) > volfrac*nelx*nely
```

Με τις παραπάνω επεμβάσεις, η κατασκευή παίρνει τη μορφή:

Εικόνα 40. Ο πίνακας avoid εμποδίζει την τοποθέτηση υλικού στο κάτω μέρος κοντά στη στήριξη της κατασκευής.

Για να δώσουμε ένα πιο δενδροειδές σχήμα στην κατασκευή μπορούμε να εφαρμόσουμε την δημιουργία των παρακάτω κύκλων στο avoid

```
%% f(x,y) generation
% define center of circle
radius=2;
coef = 90;
xc1 = floor(15/60*(nelx+1));
yc1 = (nely+1)/2;
xc2 = floor(20/60*(nelx+1));
yc2 = (nely+1)/2;
xc3 = floor(25/60*(nelx+1));
yc3 = (nely+1)/2;
xc4 = floor(30/60*(nelx+1));
yc4 = (nely+1)/2;
xc5 = floor(18/60*(nelx+1));
yc5 = (nely+1)/4;
xc6 = floor(23/60*(nelx+1));
yc6 = (nely+1)/4;
xc7 = floor(28/60*(nelx+1));
yc7 = (nely+1)/4;
sdv = radius;
i_man_constr = 1;
avoid = ones(nely,nelx);
 for i=1:(nely)
     for j=1:(nelx)
         if (i_man_constr == 1)
            avoid(i,j) = 1.0+coef *...
```

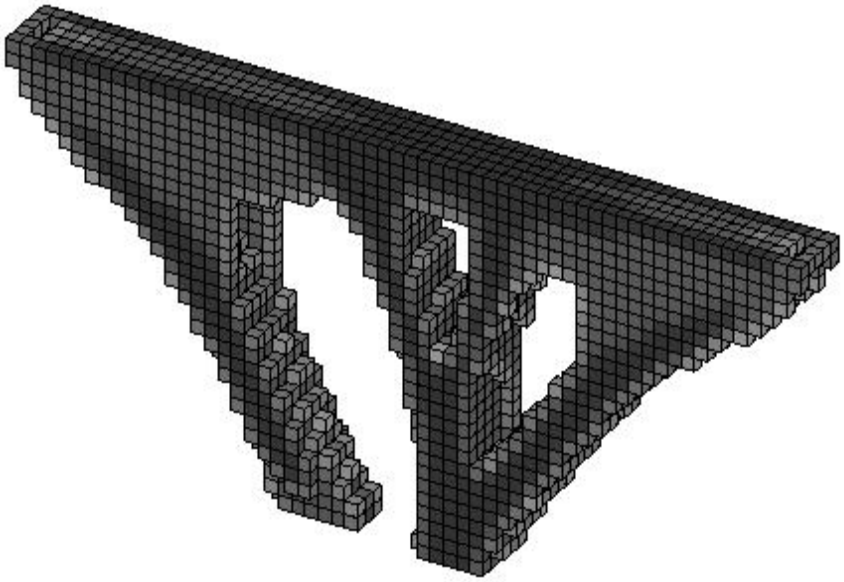
```
                (exp((-(i-yc1)^2-(j-xc1)^2)/(2*sdv^2))...
                +exp((-(i-yc2)^2-(j-xc2)^2)/(2*sdv^2))...
                +exp((-(i-yc3)^2-(j-xc3)^2)/(2*sdv^2))...
                +exp((-(i-yc4)^2-(j-xc4)^2)/(2*sdv^2))...
                +exp((-(i-yc5)^2-(j-xc5)^2)/(2*sdv^2))...
                +exp((-(i-yc6)^2-(j-xc6)^2)/(2*sdv^2))...
                +exp((-(i-yc7)^2-(j-xc7)^2)/(2*sdv^2))...
                );
        end
    end
end
```

Δίνοντας διάφορα κέντρα κατά μήκος του (nely+1)/2 και του (nely+1)/4 επιβάλουμε στην κατασκευή να διακλαδωθεί και έτσι προκύπτει η εικόνα:

Εικόνα 41. Η δενδροειδής μορφή επιτυγχάνεται με κύκλους οι οποίοι αναγκάζουν την κατασκευή να αναπτύξει κλαδιά για να παραλάβει τις δυνάμεις.

5.7 ΕΦΑΡΜΟΓΗ ΜΑΖΙΚΩΝ ΔΥΝΑΜΕΩΝ ΣΤΟΝ ΚΩΔΙΚΑ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗΣ ΤΟΠΟΛΟΓΙΑΣ TOP3D

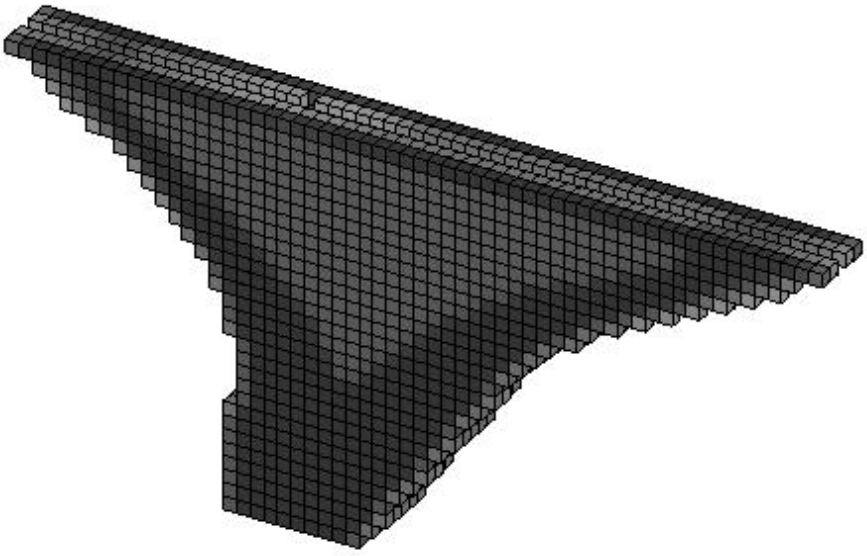
Για να γίνει η εφαρμογή των μαζικών δυνάμεων στον κώδικα βελτιστοποίησης τοπολογίας top3D αρκεί αρχικά να προσθέσουμε στο σημείο στο οποίο γίνεται η προετοιμασία του φίλτρου και την προετοιμασία του μητρώου sF2. Στη συνέχεια θα πρέπει να εφαρμοστεί η γραμμηκοποίηση της μεθόδου SIMP για πυκνότητες μικρότερες από το 0,2. Τέλος ,να προστεθεί στην παράγωγο dc το κομμάτι της δύναμης. Οι αλλαγές αυτές καθώς και η εφαρμογή τους είναι παρόμοιες με εκείνες που πραγματοποιήθηκαν στον κώδικα top88 αν λάβουμε υπόψη μας τα τρισδιάστατα κυβικά πεπερασμένα στοιχεία καθώς και τους παραπάνω βαθμούς ελευθερίας.



Εικόνα 42. Παράδειγμα εφαρμογής μαζικών δυνάμεων σε τρεις διαστάσεις.

5.8 ΕΦΑΜΡΟΓΗ ΜΑΖΙΚΩΝ ΚΑΙ ΘΕΡΜΙΚΩΝ ΔΥΝΑΜΕΩΝ ΣΤΟΝ ΚΩΔΙΚΑ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗΣ ΤΟΠΟΛΟΓΙΑΣ TOP3D

Η δομή του κώδικα top3D με συμμετοχή μαζικών αλλά και θερμικών δυνάμεων είναι παρόμοια με τη δομή του top88. Οι αλλαγές που θα πρέπει να πραγματοποιηθούν είναι οι ίδιες, προσαρμοσμένες στα τρισδιάστατα κυβικά πεπερασμένα στοιχεία.



Εικόνα 43. Παράδειγμα εφαρμογής μαζικών και θερμικών δυνάμεων σε τρεις διαστάσεις.

5.9 ΕΛΑΧΙΣΤΟΠΟΙΗΣΗ ΧΡΟΝΟΥ ΣΤΟΝ ΚΩΔΙΚΑ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗΣ ΤΟΠΟΛΟΓΙΑΣ TOP3D

Αρχικά για να δούμε πιο κομμάτι του κώδικα θα χρειαστεί να αλλάξουμε προκειμένου να βελτιστοποιήσουμε το χρόνο ,τρέχουμε το αρχικό παράδειγμα με την εντολή profiler. Με αυτό τον τρόπο παρατηρούμε ότι τον περισσότερο χρόνο χρησιμοποιεί η μέθοδος που εφαρμόζεται για την επίλυση της εξίσωσης ισορροπίας. Η μέθοδος αυτή είναι η Preconditioned Conjugated Gradient Method (PCG).Το αρχικό πρόγραμμα χρησιμοποιεί τη PCG που είναι ενσωματωμένη στη matlab. Προκειμένου να μπορούμε να επέμβουμε εσωτερικά της PCG και να αποφύγουμε τις παραπάνω εντολές που εκτελεί λόγο της γενικής χρήσης που τη δίνει η matlab επιλέξαμε να φτιάξουμε μια function PCG η οποία θα μας επιλύει την εξίσωση ισορροπίας. Παρακάτω εξηγείται η μέθοδος PCG καθώς και η δημιουργία της συνάρτησης pcg που χρησιμοποιείτε από το πρόγραμμα.

5.9.1 PRECONDITIONED CONJUGATED GRADIENT METHOD

Προκειμένου να επιλυθεί η γραμμική εξίσωση

$$F = Ax \tag{5.23}$$

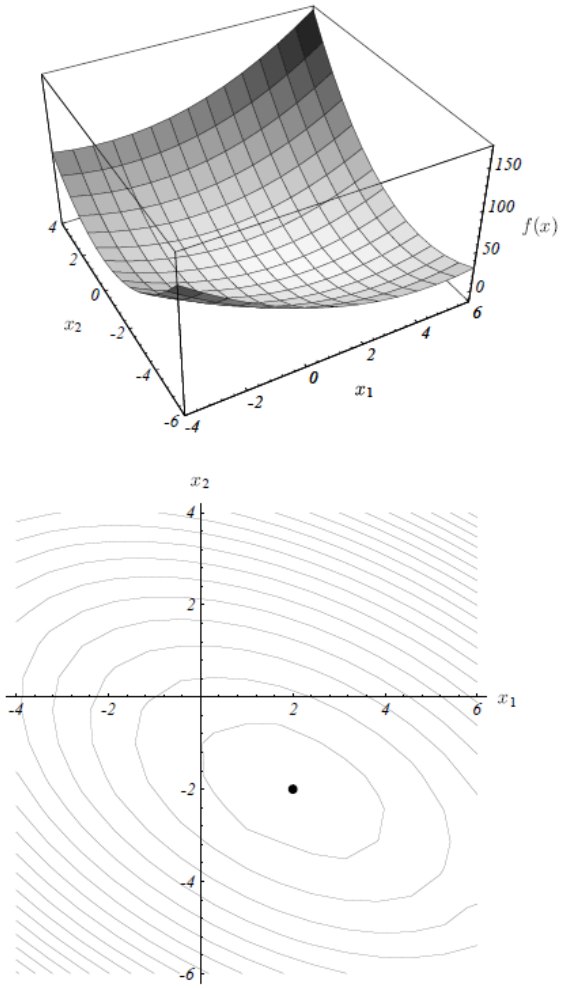
Η PCG θεωρεί τη παράγουσα της συνάρτησης

$$f\,'(x) = Ax - F \tag{5.24}$$

που είναι η συνάρτηση

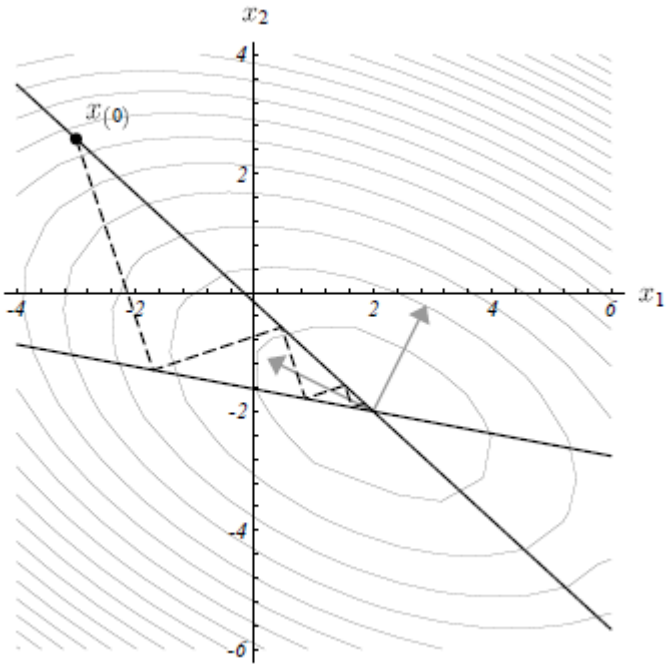
$$\mathrm{f}(x) = \frac{1}{2} x^T Ax - F^T x + C \tag{5.25}$$

και την ελαχιστοποιεί. Επομένως εκεί που η f(x) ελαχιστοποιείτε η f'(x) μηδενίζεται και παίρνουμε τη λύση της γραμμικής εξίσωσης.



Εικόνα 44. Στο επάνω σχήμα απεικονίζεται η συνάρτηση F(x) για μητρώο x δύο μεταβλητών. Το κατώτερο σημείο της εξίσωσης είναι και η λύση του συστήματος. Στο κάτω σχήμα φαίνεται η κάτοψη της συνάρτησης F(x).

Η εύρεση του ελαχίστου της συνάρτησης f(x) γίνεται μέσο μιας επαναληπτικής διαδικασίας. Η διαδικασία ξεκινά από ένα αρχικό σημείο x_0 και προχωρεί σε καινούρια x μέχρι να φτάσει στη λύση.



Εικόνα 45. Παράδειγμα μετακίνησης από το x_0 με κατευθύνσεις d_i μέχρι τη λύση x.

Προτού αναλυθεί η διαδικασία με την οποία λαμβάνονται τα καινούρια x θα πρέπει πρώτα να οριστούν δύο έννοιες.

- Σφάλμα (error) $e_i = x_i - x$,είναι η διαφορά της κάθε τιμής που λαμβάνουμε από τη λύση. Επομένως δείχνει πόσο μακριά βρισκόμαστε από τη λύση της εξίσωσης.

- Υπόλοιπο (residual) $r_i = F - Ax_i$,μας δείχνει πόσο μεγάλη είναι η διαφορά του γινομένου Ax_i από το μητρώο F. Το υπόλοιπο μπορεί να γραφεί και στη μορφή $r_i = -Ae_i$. Τέλος, το υπόλοιπο μπορεί να θεωρηθεί ως η αρνητική παράγωγος της συνάρτησης F(x).

$$r_i = -f(x) \tag{5.26}$$

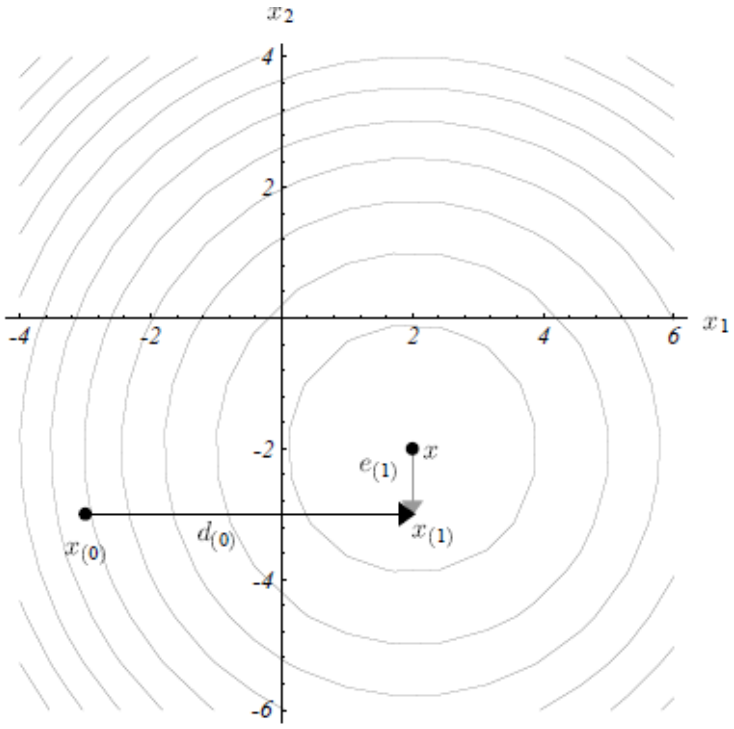
Ο τύπος που στη pcg μας δίνει κάθε φορά το καινούριο x είναι:

$$x_{i+1} = x_i + a_i d_i \tag{5.27}$$

όπου a είναι το βήμα και d είναι η κατεύθυνση πάνω στη συνάρτηση f(x) στην οποία κινούμαστε για να βρούμε το νέο x.

Επομένως παρόμοια και στη μέθοδο Gradient Descent το κάθε καινούριο x λαμβάνεται αν στο παλιό προσθέσουμε το γινόμενο a*d. Η διαφορά με τη GD είναι οι τιμές που λαμβάνουν οι μεταβλητές a,d.

Το βήμα της μετατόπισης a λαμβάνεται τέτοιο ώστε η κατεύθυνση d_i στην οποία κινούμαστε να είναι A-ορθογωνική με το σφάλμα $e_{i+1,}$ του καινούριου σημείου x_{i+1}.



Εικόνα 46. Παράδειγμα μετακίνησης πάνω στη κατεύθυνση d_0 μέχρι το d_0 να γίνει κάθετο με το σφάλμα του νέου σημείου e_1.

Αυτό συμβαίνει όταν:

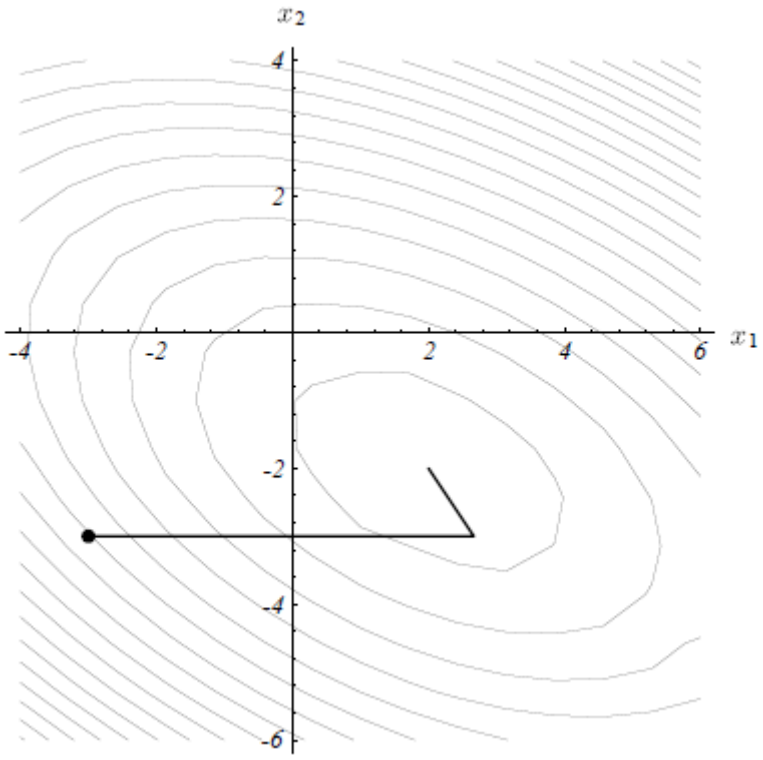
$$a_i = \frac{d_i r_i}{d_i A d_i} \tag{5.28}$$

Η διευθύνσεις d λαμβάνονται από τη διαδικασία που ονομάζεται conjugate Gram-Schmidt process. Επειδή στη CG οι διευθύνσεις πρέπει να είναι A-ορθογωνικές μεταξύ τους η διαδικασία των conjugate Gram-Schmidt παίρνει τις παραγώγους -f'(x) οι οποίες ισούται με r_i και αφαιρεί από αυτές τα μη A-ορθογωνικά στοιχεία τους για να αποκτήσει τις κατευθύνσεις d_i. Αυτό επιτυγχάνεται μέσω της σχέσης:

$$d_i = r_i + \beta_i d_{i-1} \tag{5.29}$$

όπου β είναι ένας συντελεστής που υπολογίζεται από τη σχέση:

$$\beta_i = \frac{r_i^T r_i}{r_{i-1}^T r_{i-1}} \tag{5.30}$$



Εικόνα 47.Παράδειγμα μετακίνησης α μέχρι η κατεύθυνση κίνησης d να γίνει A-ορθογωνική με το σφάλμα e του νέου σημείου x .

Με αυτό τον τρόπο το μόνο που πρέπει ακόμα να βρούμε είναι κάθε καινούριο r_i. Το κάθε r_i προκύπτει από το προηγούμενο αν στη σχέση ! πολλαπλασιάσουμε με -A και προσθέσουμε το μητρώο F:

$$\begin{aligned} & x_{i+1} = x_i + a_i d_i \Leftrightarrow \\ & F - Ax_{i+1} = F - Ax_i - Aa_i d_i \Leftrightarrow \\ & r_{i+1} = r_i - a_i A d_i \end{aligned} \tag{5.31}$$

Επομένως η διαδικασία που ακολουθεί η CG έχει τη μορφή

$$d_0 = r_0 = F - Ax_0 \tag{5.32}$$

$$a_i = \frac{r_i^T r_i}{d_i A d_i} \tag{5.33}$$

$$x_{i+1} = x_i + a_i d_i \tag{5.34}$$

$$r_{i+1} = r_i - a_i A d_i \tag{5.35}$$

$$\beta_{i+1} = \frac{r_{i+1}^T r_{i+1}}{r_i^T r_i} \tag{5.36}$$

$$d_{i+1} = r_{i+1} + \beta_{\iota+1} d_i \tag{5.37}$$

Προκειμένου να γίνει το πρόβλημα σαφέστερα και ορθότερα ορισμένο και επομένως να καταλήξουμε στη λύση σε μικρότερο αριθμό επαναλήψεων, μπορούμε να ορίσουμε έναν preconditioner M. Ο preconditioner M είναι ένας τετραγωνικός πίνακας ίδιων διαστάσεων με τις διαστάσεις του μητρώου A, ο οποίος πρέπει να είναι εύκολο να αντιστραφεί. Η διαδικασία με χρήση του preconditioner παίρνει τη παρακάτω μορφή:

$$d_0 = M^{-1} r_0 = M^{-1}(F - A x_0) \tag{5.38}$$

$$a_i = \frac{r_i^T M^{-1} r_i}{d_i A d_i} \tag{5.39}$$

$$x_{i+1} = x_i + a_i d_i \tag{5.40}$$

$$r_{i+1} = r_i - a_i A d_i \tag{5.41}$$

$$\beta_{i+1} = \frac{r_{i+1}^T M^{-1} r_{i+1}}{r_i^T M^{-1} r_i} \tag{5.42}$$

$$d_{i+1} = M^{-1} r_{i+1} + \beta_{\iota+1} d_i \tag{5.43}$$

Ο preconditioner που χρησιμοποιείτε αυτόματα από τη συνάρτηση PCG της matlab είναι ο Jacobi preconditioner. Για να τον υπολογίσουμε αρκεί να πάρουμε μόνο τις διαγώνιες τιμές του μητρώου A.

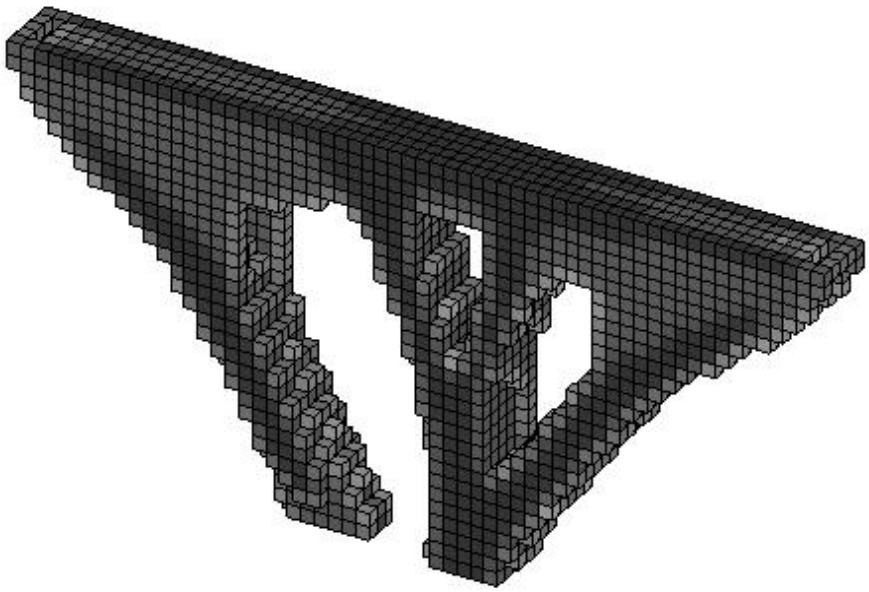
Αλγόριθμος υλοποίησης της Preconditioner Conjugated Gradient

Αρχικά δημιουργούμε μια function με όνομα gcp.

function [x, k] = gcp(A, b, stol, mit, C, x0)

Η συνάρτηση αυτή έχει ως δεδομένα το μητρώο δυσκαμψίας ως A, το μητρώο των δυνάμεων ως b, την απόκλιση του γινομένου Ax από το A, τον μέγιστο αριθμό επαναλήψεων, το Preconditioner C καθώς και το αρχικό μητρώο μετακινήσεων x0. Τα αποτελέσματα που επιστρέφει η συνάρτηση είναι το μητρώο των μετακινήσεων και ο αριθμός των επαναλήψεων που πραγματοποιήθηκαν. Η διαδικασία υπολογισμού των μετακινήσεων γίνεται ακλουθώντας τον αλγόριθμο που περιγράφηκε. Για τη πρώτη επανάληψη θεωρείτε $d_0=r_0$. Αυτό μπορεί να πραγματοποιηθεί με δύο τρόπους. Είτε με την πρώτη ανανέωση των μετακινήσεων εκτός επανάληψης είτε με χρήση της εντολή if για να γίνει ο διαχωρισμός εντός της επαναληπτικής διαδικασίας. Η επαναληπτική διαδικασία συνεχίζεται μέχρι το μέγιστο r να είναι μικρότερο από το stol ή εξαντληθούν οι επαναλήψεις.

Στη συνέχεια με βάση τη δημοσίευση των O. Amir και O. Sigmund [8] με όνομα On reducing computational effort in topology optimization: how far can we go? αλλάζουμε την μέγιστη απόκληση από 10^{-8} σε 10^{-4}. Παρατηρούμε ότι δεν υπάρχει κάποια διαφορά στο τελικό σχήμα που μας δίνει η κατασκευή. Υπάρχει όμως μεγάλη διαφορά στο χρόνο που χριάζεται για να υπολογίσει το σχήμα αυτό.



Εικόνα 48. Παράδειγμα εφαρμογής μαζικών δυνάμεων με ακρίβεια PCG 10^{-4}

5.10 ΧΡΗΣΗ ΚΑΡΤΑΣ ΕΠΕΞΕΡΓΑΣΙΑΣ ΓΡΑΦΙΚΩΝ

Προκειμένου να μειώσουμε περεταίρω το χρόνο που χρειάζεται ο κώδικας για να ολοκληρωθεί θα προσπαθήσουμε να μεταφέρουμε ολόκληρη ή και ένα κομμάτι της υπολογιστικής διαδικασίας στην κάρτα γραφικής επεξεργασίας GPU (Graphic Processor Unit). Η κάρτα επεξεργασίας γραφικών στον υπολογιστή χρησιμοποιείτε για την επεξεργασία των γραφικών απεικονίσεων. Λόγο των υπολογιστικών απαιτήσεων των γραφικών απεικονίσεων η κάρτα γραφικών έχει περισσότερους

πυρήνες από τους επεξεργαστές του υπολογιστή CPU (Computer Processor Unit). Το μειονέκτημα της GPU είναι η μνήμη της. Ενώ η CPU χρησιμοποιεί τη μνήμη RAM (Random Access Memory) ,η GPU μπορεί να χρησιμοποιήσει μόνο τη δική της μνήμη. Για αυτό το λόγο στα προβλήματα βελτιστοποίησης τοπολογίας των κατασκευών το μεγαλύτερο πρόβλημα που πρέπει να αντιμετωπιστεί προκειμένου να μπορεί να μεταφερθεί ο όγκος της υπολογιστικής διαδικασίας στη GPU είναι η αποθήκευση του ολικού μητρώου δυσκαμψίας της κατασκευής.

Για να αντιμετωπίσουμε το πρόβλημα αυτό δοκιμάσαμε αρχικά δύο μεθόδους.

Η πρώτη μέθοδος βασίστηκε κυρίως στη δημοσίευση High-Performance GPU computing for density-based topology optimization. Στη δημοσίευση αυτή προτείνεται η αποφυγή δημιουργίας του ολικού μητρώου δυσκαμψίας της κατασκευής. Για να γίνει αυτό, η επίλυση της εξίσωσης ισορροπίας γίνεται υπολογίζοντας τη σχέση βαθμού ελευθερίας με βαθμό ελευθερίας βασιζόμενη στο τοπικό μητρώο δυσκαμψίας του κάθε στοιχείου. Η τροποποίηση του κώδικα με αυτό τον τρόπο ευνοεί τη λειτουργία των kernel που χρησιμοποιούνται από τη γλώσσα προγραμματισμού C για να χωρίσουν τη διαδικασία βελτιστοποίησης σε κομμάτια και μετά να χρησιμοποιήσουν παράλληλες επεξεργασίες για την επίλυση του κάθε κομματιού. Η εφαρμογή στη matlab δεν έδωσε τα επιθυμητά αποτελέσματα σε χρόνους και για αυτό απορρίφθηκε.

Η δεύτερη μέθοδος είναι η αξιοποίηση της μεθόδου αποθήκευσης μεγάλων πινάκων με πολλά μηδενικά Skyline. Η μέθοδος αυτή αποθηκεύει από το μητρώο δυσκαμψίας τα μη μηδενικά στοιχεία μαζί με μερικά μηδενικά. Με αυτόν τον τρόπο εξοικονομεί χώρο στη μνήμη. Η διαδικασία ανανέωσης των στοιχείων του μητρώου δυσκαμψίας στο μητρώο Skyline καθώς και το γινόμενο του μητρώου Skyline με άλλα διανύσματα αποδείχθηκαν αρκετά χρονοβόρες διεργασίες με αποτέλεσμα και η δεύτερη μέθοδος να θεωρηθεί απορριπτέα.

Έχοντας απορρίψει τις δύο μεθόδους αυτές προχωρήσαμε στην αντιμετώπιση του προβλήματος της αποθήκευσης του μητρώου δυσκαμψίας με χρήση της εντολής sparse της matlab. Η εντολή αυτή υποστηρίζεται από τις βιβλιοθήκες της matlab για αποθήκευση μητρώων στη GPU με το μειονέκτημα ότι δεν είναι δυνατή η χρήση δεικτών. Αυτό αποτελεί πρόβλημα επειδή η PCG δέχεται ως δεδομένα το

K(freedofs,freedofs)

Επίσης οι τιμές του μητρώου K αλλάζουν σε κάθε επανάληψη. Επομένως η μεταφορά ολόκληρου του μητρώου και η ανανέωση του στη GPU δεν είναι δυνατή. Για το λόγο αυτό γίνεται η ανανέωση των τιμών του μητρώου K στη CPU και μετά η μεταφορά του άνω μητρώου στη GPU για να λειτουργήσει η PCG. Παρακάτω αναφέρονται τρεις λύσεις.

Στη πρώτη το κομμάτι που μεταφέρεται και υπολογίζεται στη GPU είναι το γινόμενο αAd της σχέσης 5.41. Το γινόμενο αυτό καταλαμβάνει το μεγαλύτερο ποσοστό υπολογιστικού χρόνου σε όλη τη διαδικασία βελτιστοποίησης. Το μειονέκτημα είναι ότι θα πρέπει να υπάρχει συνεχή μεταφορά δεδομένων από τη CPU στη GPU επειδή ο λόγος αυτός υπολογίζεται αρκετές φορές. Η μεταφορά αυτή γίνεται με τις εντολές

```
Agpu = gpuArray(A);
dgpu = gpuArray(d);
Adgpu = Agpu * dgpu;
Ad = gather(Adgpu);
```

Στη δεύτερη λύση μεταφέρεται ολόκληρη η μέθοδος PCG στη GPU. Αυτό πραγματοποιείτε με μεταφορά των δεδομένων της PCG πριν να καλεστεί και μεταφορά των μετακινήσεων από τη GPU στη CPU μετά το τέλος της PCG. Το μειονέκτημα και πάλι είναι ότι η PCG καλείτε αρκετές φορές κατά τη βελτιστοποίηση. Επομένως υπάρχει μεγάλη μεταφορά δεδομένων από τη GPU στη CPU. Για αν γίνει η μεταφορά της PCG στη κάρτα γραφικών πρέπει να εισαχθούν οι παρακάτω εντολές

```
Ugpu = gpuArray(U(freedofs,:));
Kgpu = gpuArray(K(freedofs,freedofs));
Fgpu = gpuArray(F(freedofs,:));
Mgpu = gpuArray(M);
Ugpu = gcp(Kgpu, Fgpu, tolit, maxit, Mgpu, Ugpu);
U(freedofs,:) = gather(Ugpu(:,1));
```

Στη τρίτη λύση μεταφέρεται όλο το κομμάτι της επανάληψης βελτιστοποίησης εκτός από την ανανέωση των τιμών του μητρώου δυσκαμψίας στη GPU. Αυτό πραγματοποιείτε με τη δημιουργία του παρακάτω κομματιού:

```
% GPUARRAYS

change = gpuArray(1);

Ugpu = gpuArray(U);

xPhysgpu = gpuArray(xPhys);

KEgpu = gpuArray(KE);

edofMatgpu = gpuArray(edofMat);

dcFgpu = gpuArray(dcF);

Hgpu = gpuArray(H);
```

Hsgpu = gpuArray(full(Hs));

xgpu = gpuArray(x);

EO = gpuArray(E0);

EMIN = gpuArray(Emin);

PENAL = gpuArray(penal);

VOLFRAC = gpuArray(volfrac);

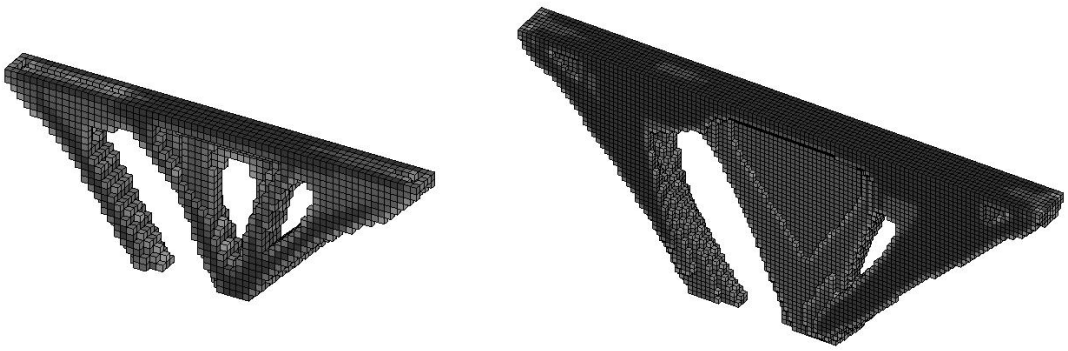
NELE = gpuArray(nele);

Με αυτό τον τρόπο οι μεταβλητές που θα αξιοποιηθούν εντός της επανάληψης βελτιστοποίησης μεταφέρονται στη GPU. Το μειονέκτημα σε αυτή την περίπτωση είναι ότι η ανανέωση των τιμών του μητρώου δυσκαμψίας γίνεται ακόμα στη CPU. Επομένως αρκετός χρόνος δαπανάτε στη μεταφορά του μητρώου K(freedofs,freedofs) σε κάθε επανάληψη.

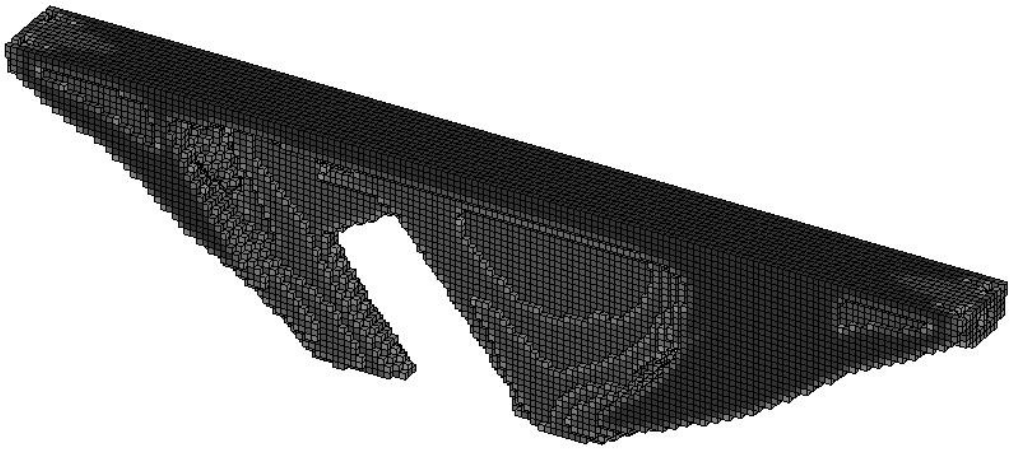
Οι μέθοδοι αυτοί εφαρμόζονται και στον αλγόριθμο με την εφαρμογή των θερμικών δυνάμεων. Στη περίπτωση αυτή η χρήση της PCG γίνεται και εκτός της επανάληψης βελτιστοποίησης στον υπολογισμό των Csmin και Cthmin. Για τα κομμάτια αυτά χρησιμοποιείτε η δεύτερη λύση που αναφέρθηκε.

Παρακάτω παρουσιάζονται τρία παραδείγματα για το απλό στατικό πρόβλημα με μαζικές δυνάμεις και τρία παραδείγματα για τον συνδυασμό του στατικού και θερμικού προβλήματος. Ο φορέας είναι ένα στέγαστρο με φορτίσεις στο άνω μέρος και τη στήριξη κάτω. Αυτό που αλλάζει κάθε φορά είναι ο αριθμός των στοιχείων για τα οποία εκτελείτε η βελτιστοποίηση. Σκοπός είναι να φανεί η επιτάχυνση που μπορεί να επιτευθεί από τις επεμβάσεις που έγιναν στον αλγόριθμο βελτιστοποίησης τοπολογίας, καθώς και η σημασία της χρήσης μεγάλου αριθμού στοιχείων. Αρχικά για το στατικό πρόβλημα με μαζικές δυνάμεις τα τρία παραδείγματα αποτελούνται από 8000 στοιχεία, 54000 στοιχεία και 83200 στοιχεία. Για το συνδυασμό του θερμικού και στατικού προβλήματος τα παραδείγματα αποτελούνται από 5850 στοιχεία, 17500 στοιχεία και 31850 στοιχεία. Ο αριθμός στοιχείων σε αυτή την περίπτωση είναι μικρότερος λόγο των απαιτήσεων σε μνήμη της κάρτα γραφικών.

Ακολουθούν τα στατικά παραδείγματα

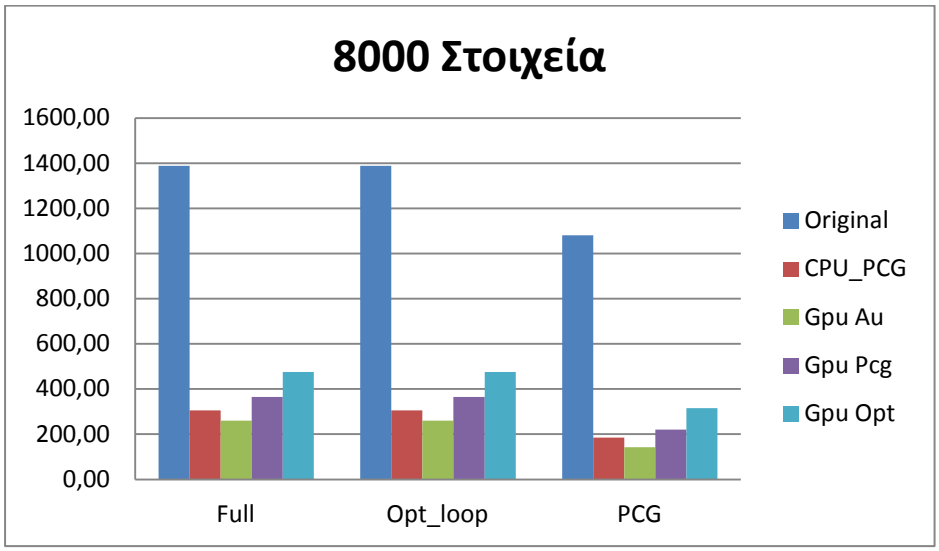


Εικόνα 49. α) Παράδειγμα εφαρμογής στατικών και μαζικών δυνάμεων 8000 στοιχείων β) Παράδειγμα εφαρμογής στατικών και μαζικών δυνάμεων 54000 στοιχείων.

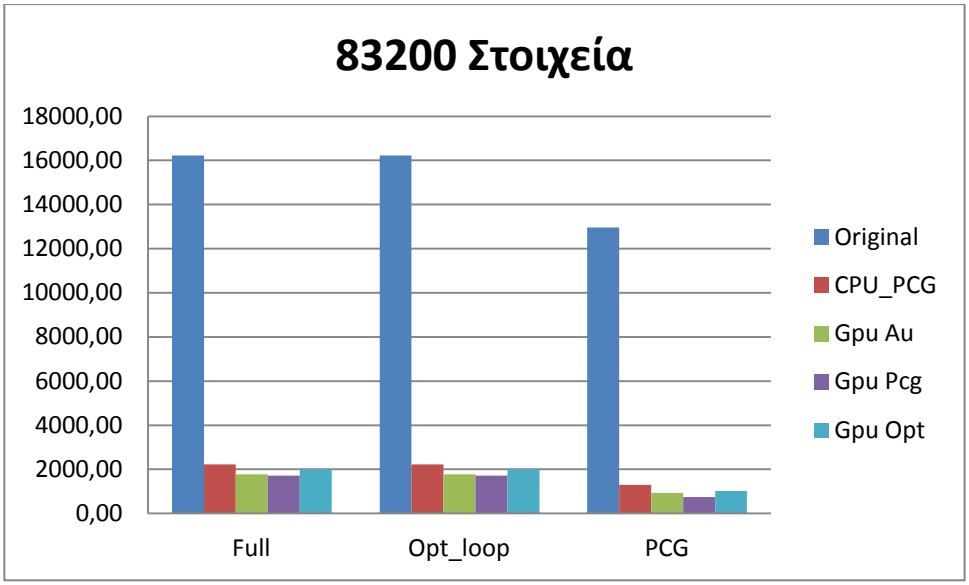


Εικόνα 50. Παράδειγμα εφαρμογής στατικών και μαζικών δυνάμεων 83200 στοιχείων.

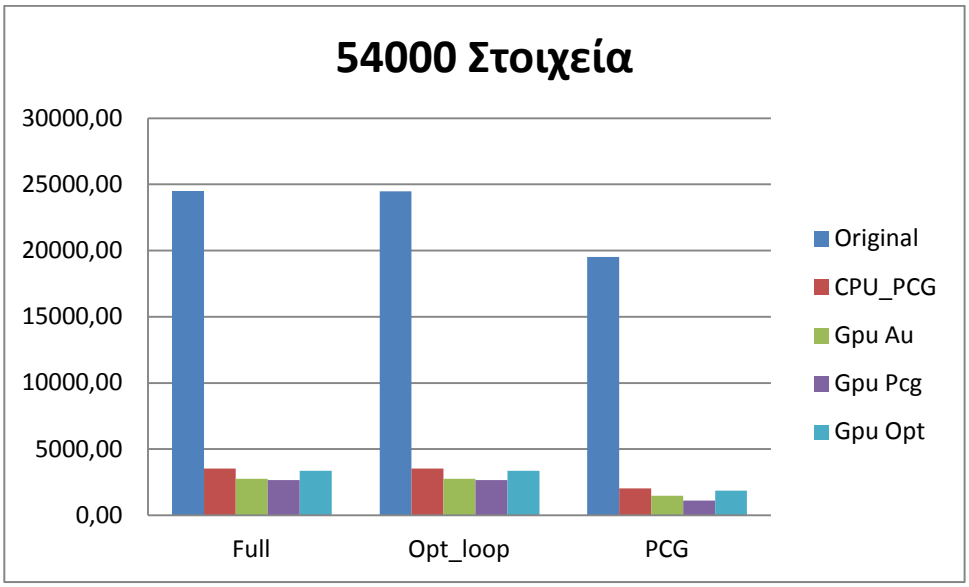
Στους παρακάτω πίνακες φαίνονται οι χρόνοι σε δευτερόλεπτα που χρειάστηκαν για τη κάθε περίπτωση.



Εικόνα 51. Χρόνοι ολοκλήρωσης κάθε κομματιού της βελτιστοποίησης για κάθε μια περίπτωση.



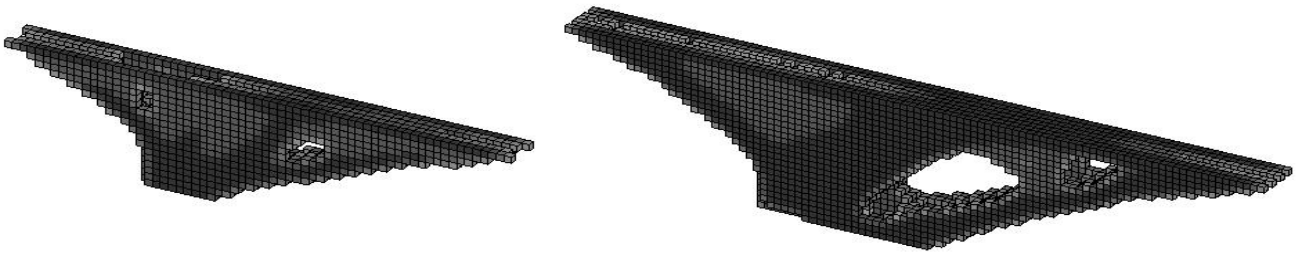
Εικόνα 52. Χρόνοι ολοκλήρωσης κάθε κομματιού της βελτιστοποίησης για κάθε μια περίπτωση.



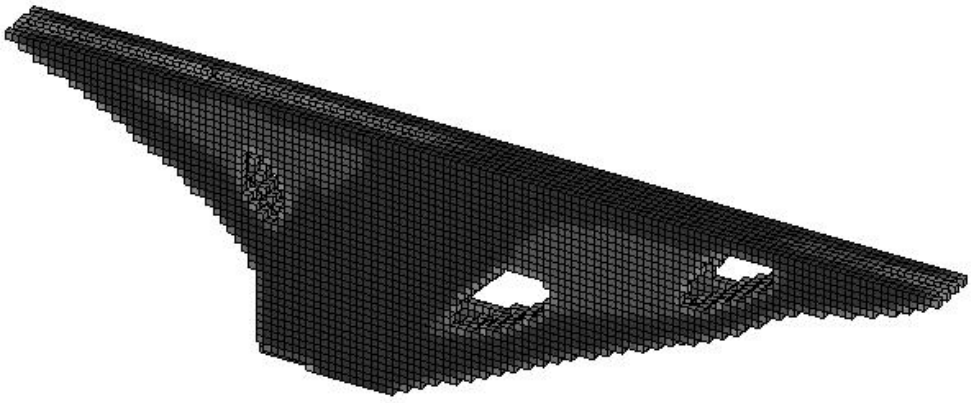
Εικόνα 53. Χρόνοι ολοκλήρωσης κάθε κομματιού της βελτιστοποίησης για κάθε μια περίπτωση.

Παρατηρούμε ότι για μικρό αριθμό στοιχείων η κάρτα γραφικής επεξεργασίας δεν επιτυγχάνει την επιτάχυνση που επιθυμούμε. Όμως και το αποτέλεσμα της διαδικασίας βελτιστοποίησης σταθεροποιείτε όταν ο αριθμός των στοιχείων αυξηθεί αρκετά. Για το μεγάλο αριθμό στοιχείων η κάρτα γραφικών είναι γρηγορότερη σε όλες τις εκδοχές που δοκιμάσαμε. η πιο αποτελεσματική είναι η δεύτερη λύση στην οποία η PCG υπολογίζεται κάθε φορά στη GPU.

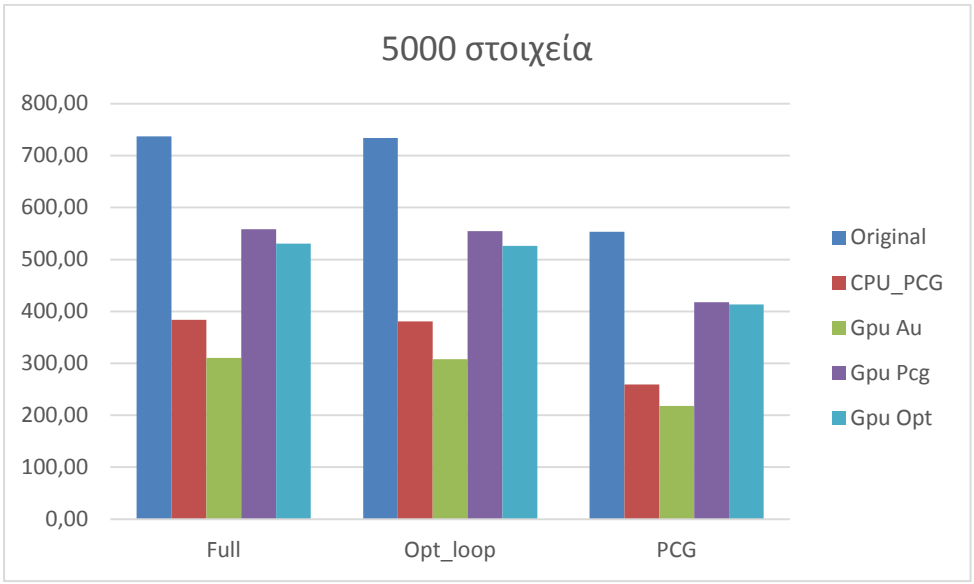
Ακολουθούν τα θερμικά παραδείγματα



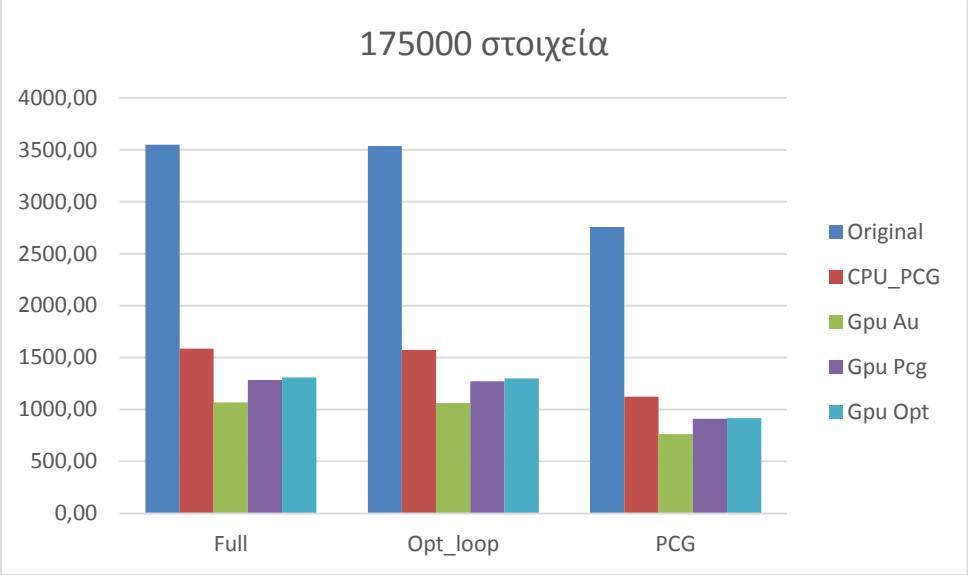
Εικόνα 54. α) Παράδειγμα εφαρμογής θερμικών και στατικών δυνάμεων 5850 στοιχείων β) Παράδειγμα εφαρμογής θερμικών και στατικών δυνάμεων 17500 στοιχείων



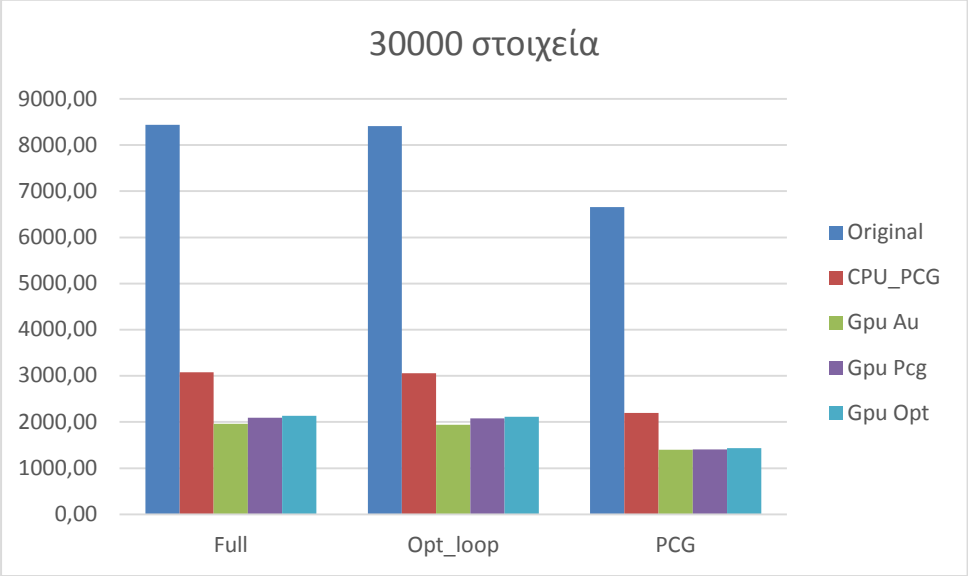
Εικόνα 55. Παράδειγμα εφαρμογής θερμικών και στατικών δυνάμεων 31850 στοιχείων.



Εικόνα 56. Χρόνοι ολοκλήρωσης κάθε κομματιού της βελτιστοποίησης για κάθε μια περίπτωση.



Εικόνα 57. Χρόνοι ολοκλήρωσης κάθε κομματιού της βελτιστοποίησης για κάθε μια περίπτωση.



Εικόνα 58. Χρόνοι ολοκλήρωσης κάθε κομματιού της βελτιστοποίησης για κάθε μια περίπτωση.

Επειδή ο αριθμός των στοιχείων είναι μικρός στα θερμικά παραδείγματα δεν βλέπουμε τη περίπτωση που η PCG εκτελείται στη GPU να δίνει τα γρηγορότερα αποτελέσματα.

ΚΕΦΑΛΑΙΟ 6

ΣΥΜΠΕΡΑΣΜΑΤΑ

Αντικείμενο της παρούσας διπλωματικής αποτέλεσε αρχικά η εφαρμογή μαζικών και θερμικών δυνάμεων με σκοπό η τελική κατασκευή να πάρει δενδροειδή μορφή. Στη συνέχεια ερευνήθηκαν τρόποι με τους οποίους θα μπορούσε να αξιοποιηθεί η υπολογιστική δύναμη της κάρτας επεξεργασίας γραφικών προκειμένου να μειωθεί ο χρόνος που χρειάζεται για να ολοκληρωθεί η βελτιστοποίηση.

Η εφαρμογή θερμικών δυνάμεων σε ένα μέρος της επιφάνειας έχει ως αποτέλεσμα τη δημιουργία κλαδιών προκειμένου να καταφέρει η κατασκευή να μεταφέρει τη θερμότητα στο έδαφος. Ένα τέτοιο παράδειγμα φαίνεται στην εικόνα 32. Για να αντιμετωπιστεί το πρόβλημα αυτό εφαρμόζονται δυνάμεις οι οποίες εξαρτώνται από τη μάζα. Στο στατικό πρόβλημα οι δυνάμεις αυτές αντιπροσωπεύουν το βάρος της κατασκευής. Στο το θερμικό αντιπροσωπεύουν τη ροή θερμότητας από την επιφάνεια της κατασκευής στο έδαφος. Με την εφαρμογή των μαζικών δυνάμεων εμφανίζονται δύο προβλήματα. Το πρώτο είναι το πρόβλημα που αναφέρεται στη δημοσίευση Topology Optimization with self-weight loading: unexpected problems and solutions [9]. Η αντιμετώπιση του γίνεται με τη τροποποίηση της μεθόδου SIMP.Το δεύτερο είναι ότι η παράγωγος της αντικειμενικής συνάρτησης διατηρεί σταθερό θετικό πρόσημο για όλα τα στοιχεία της κατασκευής. Αυτό δημιουργεί πρόβλημα στο τύπο ανανέωσης πυκνοτήτων της ΟC. Στα σημεία που η παράγωγος παίρνει θετική τιμή για να μειωθεί η αντικειμενική συνάρτηση θα πρέπει η πυκνότητα τους να μειωθεί. Επομένως μπορούμε να αντιμετωπίσουμε το παραπάνω πρόβλημα μηδενίζοντας κάθε φορά αυτές τις πυκνότητες. Τα αποτελέσματα εφαρμογής των μαζικών δυνάμεων φαίνονται στην εικόνα 36. Παρατηρούμε ότι ο συνδυασμός των προβλημάτων δίνει ως αποτέλεσμα τη δημιουργία μικρού αριθμού φορέων με μεγάλες διατομές για τη παραλαβή των δυνάμεων σε αντίθεση του καθαρά στατικού προβλήματος στο οποίο ο φορέας έχει τη μορφή δικτυώματος. Η θεώρηση των μηδενικών πυκνοτήτων δεν αποκλίνει το αποτέλεσμα από το βέλτιστο και βοηθάει στην αποφυγή της δημιουργίας γκρι υλικού. Τα αποτελέσματα στο ίδιο παράδειγμα με χρήση της μεθόδου Gradient Descent φαίνονται στην εικόνα 38.

Αν θέλουμε να έχουμε εμείς τη δυνατότητα να ορίσουμε επιφάνειες στις οποίες δεν επιθυμούμε να τοποθετηθεί υλικό έχουμε τη δυνατότητα να εφαρμόσουμε τη μέθοδο που αναφέρεται στη δημοσίευση που συνοδεύει τον κώδικα 88 γραμμών [1]. Η εφαρμογή αυτή όμως μπορεί να οδηγήσει σε αποτελέσματα όπως της εικόνας 39. Για την αποφυγή τέτοιων αποτελεσμάτων μπορούμε να εφαρμόσουμε μια συνάρτηση αύξησης της παραγώγου του όγκου στα σημεία που δεν θέλουμε υλικό. Μια τέτοια εφαρμογή φαίνεται στις εικόνες 40 και 41.

Για τη χρήση της κάρτας επεξεργασίας γραφικών το μεγαλύτερο πρόβλημα αποτελεί η αποθήκευση του μητρώου δυσκαμψίας της κατασκευής. Η χρήση του μητρώου αυτού γίνεται για την επίλυση της εξίσωσης ισορροπίας σε κάθε κύκλο της επανάληψης βελτιστοποίησης. Για να αντιμετωπιστεί το πρόβλημα αυτό έχουν αναπτυχθεί μέθοδοι που δεν κατασκευάζουν το μητρώο δυσκαμψίας. Οι μέθοδοι αυτές βρίσκουν αποτελεσματική εφαρμογή σε γλώσσες προγραμματισμού όπως είναι η C++. Στην περίπτωση της matlab τα αποτελέσματα που δίνουν δεν είναι ικανοποιητικά. Προκειμένου να είναι δυνατή η χρήση της κάρτας επεξεργασίας γραφικών προτάθηκαν τρεις μετατροπές. Τα χρονικά αποτελέσματα αυτών των μετατροπών φαίνονται στους πίνακες !.

Από τις εικόνες 49,50,51 και 52 παρατηρούμε τα αποτελέσματα της μεθόδου βελτιστοποίησης αλλάζουν όσο ο αριθμός των στοιχείων αυξάνεται. Από ένα αριθμό στοιχείων και μετά τα αποτελέσματα σταθεροποιούνται ποιά. Επομένως η εφαρμογή μεθόδων και η χρήση hardware του υπολογιστή προκειμένου να μειωθεί ο χρόνος που χρειάζεται για την εφαρμογή των αλγορίθμων βελτιστοποίησης είναι πολύ σημαντικό κομμάτι.

Βιβλιογραφία

[1] E. Andreassen, A. Clausen, M. Schevenels, B. S. Lazarov and O. Sigmund, “Efficient topology optimization in Matlab using 88 line of code,” 2010.

[2] G. Allaire and A. Kelly, “Thickness or sizing optimization of a 3-d elastic cantilever by SIMP Method,” 2010.

[3] O. Sigmund and K. Maute, Topology Optimization approaches:a comparative review, 2013.

[4] M. Bendsoe and O. Sigmund, Topology Optimization Theory, Methods and Applications, 2004.

[5] P. W. Christensen and A. Klarbing, An Introduction to Structural Optimization, Solid Mechanics and its Applications, 2009.

[6] P. Dombernowsky and A. Sondergaard, “Three-Dimensional topology optimization in architectural and structural design of concrete structures,” 2009.

[7] K. D. Tsavdaridis, J. J. Kingman and V. V. Toropov, “Structural Topology Optimization Study And Novel Perforated Beams”.

[8] O. Amir and O. Sigmund, “On reducing computational effort in topology optimization: how far can we go?,” 2010.

[9] M. Bruyneel and P. Duysinx, “Topology Optimization with self-weight loading: unexpected problems and solutions”.

[10] J. R. Shewchuk, Introduction to the Conjugated Gradient Method Without the agonizing Pain, 1994.

[11] M. Frutos, M. Castejon and H. Perez, “High-Performance GPU computing for density-based topology optimization,” *International Journal for Numerical Methods in Engineering,* 2016.

[12] O. Sigmond, “A 99 line topology Optimization code written in Matlab,” 2001.

[13] K. Liu and A. Tovar, “An efficient 3D topology optimization code written in

Matlab," 2013.

[14] Ι. Κ. Κούκος, Βελτιστοποίηση Διεργασιών και συστημάτων, 2014.

[15] Μ. Παπαδρακάκης, Ανάλυση Φορέων με τη Μέθοδο των Πεπερασμένων Στοιχείων, 2001.

[16] A. K. Kaw, E. E. Kalu and D. Nguyen, Numerical Methods with Applications.

[17] A. Mahdavi, R. Balaji, M. Frecker and E. M. Mockensturm, "Parallel Optimality Criteria-based Topology Optimization for minimum Compliance Design," 2005.