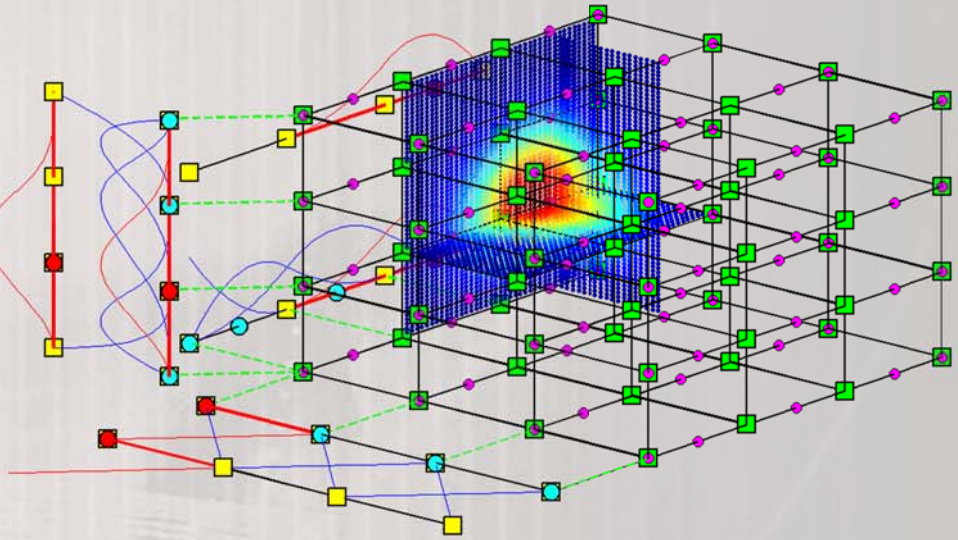
National Technical University of Athens
School of Civil Engineering
Department of Structural Engineering
Institute of Structural Analysis & Antiseismic Research

Hierarchical Formulation of the Isogeometric Analysis Method for Structures



Emmanouil Trypakis

Supervisors

Manolis Papadrakakis, Professor NTUA
Panagiotis Karakitsios, PhD Candidate

Athens, November 2014

National Technical University of Athens

School of Civil Engineering

Department of Structural Engineering

Institute of Structural Analysis and Antiseismic Research

Diploma Thesis

Hierarchical Formulation of the Isogeometric Analysis Method for Structures

Supervisors:

Manolis Papadrakakis, NTUA Professor

Panagiotis Karakitsios, NTUA PhD Candidate

Emmanouil Trypakis

Athens, November 2014

Εθνικό Μετσόβιο Πολυτεχνείο

Σχολή Πολιτικών Μηχανικών

Τομέας Δομοστατικής

Εργαστήριο Στατικής και Αντισεισμικών Ερευνών

Διπλωματική Εργασία

Ιεραρχική Διατύπωση της Μεθόδου Ισογεωμετρικής Ανάλυσης Κατασκευών

Επιβλέποντες:

Μανόλης Παπαδρακάκης, Καθηγητής ΕΜΠ

Παναγιώτης Καρακίτσιος, Υποψήφιος Δρ. ΕΜΠ

Εμμανουήλ Τρυπάκης

Αθήνα, Νοέμβριος 2014

To my family
George, Argyro, Eleni

Acknowledgements.

This thesis, the result of a long period of two years' struggle, is the final piece that completes my undergraduate studies in the School of Civil Engineering in the National Technical University of Athens. I am leaving this institution and looking back I realize how richer I am now than when I first entered it. I had excellent professors to guide me and best friends to support me. In that aspect I feel the need to thank them.

First, I would like to thank my supervisor, Professor Manolis Papadrakakis for accepting me as his student and supporting me during the last three years. His advice, encouragement and patient guidance helped me to complete this thesis and allowed me to grow as a research engineer on Isogeometric Analysis. I have been extremely lucky to have a supervisor who cared so much about my work and responded to my questions and queries so promptly. He is a tremendous mentor for me and a great teacher and supporter in the academic community.

This thesis would not have the same depth and extent if it were not for the PhD candidate Panagiotis Karakitsios. Panagiotis has been a strong and supportive advisor to me throughout my undergraduate years. He demonstrated his faith in my ability to rise to the occasion and do the necessary work. He is an excellent researcher who introduced me to the world of Isogeometric Analysis and helped me greatly in my first steps. I will always be grateful for this opportunity he gave me.

I would also like to thank all my friends and fellow www.mqn.gr moderators who have been a constant source of encouragement and enthusiasm. They made me live the most unique and meaningful university years. For their help in the final formulation of this thesis, I thank my friends Dimitra, Panagiotis , Dimitris T. and Dimitris P.

Finally, my deepest gratitude goes to my family, for their unflagging love and unconditional support throughout my life and my studies.

You are all brilliant!

Manos Trypakis,

November 2014.

Abstract

The aim of the present thesis is to investigate the linear static Isogeometric Analysis with NURBS and propose a Hierarchical scheme for a faster and efficient way of forming the Stiffness Matrix after a knot refinement. Isogeometric Analysis, a field recently introduced by Cottrell, Hughes and Bazilevs, aims to the complete integration of CAD and FEA technologies. The NURBS and the Finite Element Method have been examined separately, as the two components of Isogeometric Analysis. The necessary code for the analysis of the presented applications and drawings was developed in the programming language "MATLAB" and throughout the thesis linear elasticity was assumed. Chapter 1 is a general introduction to the topic of Isogeometric analysis. In chapter 2, we examine B-Splines and NURBS geometries. In chapter 3, the formulation of the stiffness matrix is investigated and in chapter 4 the application of external loads, the enforcement of boundary conditions and the resulting displacement, strain and stress fields. In chapter 5 we review refinement techniques. Furthermore, in chapter 6 we propose a new hierarchical refinement scheme for h-refinement. Finally, in chapter 7, three 2D applications are investigated and numerically tested and in chapter 8 we present the conclusions of this thesis.

Σύνοψη

Σκοπός της παρούσας διπλωματικής είναι η διερεύνηση της γραμμικής στατικής Ισογεωμετρικής Ανάλυσης με NURBS και η παρουσίαση μιας νέας μεθόδου για γρήγορο ιεραρχικό υπολογισμό του Μητρώου Στιβαρότητας μετά από την εισαγωγή νέων κόμβων. Η Ισογεωμετρική ανάλυση είναι μια περιοχή έρευνας που προτάθηκε πρόσφατα απ' τους Cottrell, Hughes και Bazilevs που στοχεύει στην πλήρη συγχώνευση των τεχνολογιών CAD και FEA. Οι συναρτήσεις σχήματος NURBS και η Μέθοδος των Πεπερασμένων Στοιχείων μελετήθηκαν ξεχωριστά ως οι δύο επιμέρους συνιστώσες της Ισογεωμετρικής Ανάλυσης. Ο απαραίτητος κώδικας για την ανάλυση και την παρουσίαση των σχημάτων και των εφαρμογών αναπτύχθηκε στη γλώσσα προγραμματισμού «MATLAB» και παντού έγινε η παραδοχή γραμμικής ελαστικότητας. Στο κεφάλαιο 1 γίνεται μια εισαγωγή στην Ισογεωμετρική Ανάλυση σαν σύγκλιση των τεχνολογιών των Πεπερασμένων στοιχείων και της Ψηφιακής Σχεδίασης. Αντικείμενο του κεφαλαίου 2 είναι η σχεδίαση με B-Spline και NURBS, στο κεφάλαιο 3 εξετάζεται η μόρφωση του Μητρώου Στιβαρότητες και στο κεφάλαιο 4 η επιβολή εξωτερικών φορτίων, συνοριακών συνθηκών και τα πεδία μετατοπίσεων, τροπών και τάσεων. Στο κεφάλαιο 5 αναλύονται τεχνικές επαναδιακριτοποίησης και βελτίωσης της λύσης και στο κεφάλαιο 6 προτείνεται μια νέα μέθοδος για την Ιεραρχική διατύπωση του μητρώου στιβαρότητας μετά από την προσθήκη νέων κόμβων. Τέλος, στο κεφάλαιο 7, διερευνώνται τρεις διδιάστατες εφαρμογές με τη μέθοδο της ισογεωμετρικής ανάλυσης και στο κεφάλαιο 8 διατυπώνονται τα συμπεράσματα της εργασίας αυτής.

Contents

Extended Abstract in Greek

Εκτεταμένη Περίληψη στα ελληνικά

Εισαγωγή στην Ισογεωμετρική Ανάλυση

Η ισογεωμετρική ανάλυση εμφανίστηκε το 2005 από τους Cottrell, Hughes, Bazilevs [1] και από τότε έχει προκαλέσει διεθνές ερευνητικό ενδιαφέρον. Χρησιμοποιώντας τις συναρτήσεις σχήματος της ψηφιακής σχεδίασης για την ανάλυση, προσπαθεί να γεφυρώσει τις τεχνολογίες CAD και FEA. Στην ισογεωμετρική προσέγγιση, η ανάλυση γίνεται πάντα με την ακριβή αρχική γεωμετρία, ακόμα και σε πολύ αραιές διακριτοποιήσεις. Οι πιο εδραιωμένες συναρτήσεις σχήματος για την ισογεωμετρική ανάλυση είναι οι ευρέως διαδεδομένες συναρτήσεις NURBS, οι οποίες λόγω των καλών τους ιδιοτήτων μπορούν να χρησιμοποιηθούν για την ανάλυση χωρίς ιδιαίτερα μεγάλες αλλαγές στις καθιερωμένες μορφές των κωδίκων για πεπερασμένα στοιχεία.

Χώροι Physical, Parameter και Index

Στην ισογεωμετρική ανάλυση με NURBS, χρησιμοποιούμε ισοπαραμετρικά στοιχεία και τρεις χώροι είναι απαραίτητοι. Ο πραγματικός χώρος (Physical Space), ο παραμετρικός χώρος (Parameter Space) και ο χώρος των δεικτών (Index Space). Ο Physical Space είναι ο χώρος που το αντικείμενο έχει την πραγματική του μορφή και στον οποίο σχεδιάζουμε τον φορέα χρησιμοποιώντας τα προγράμματα CAD ενώ ο Parameter Space, το αντίστοιχο του φυσικού χώρου στα πεπερασμένα στοιχεία, είναι ο χώρος στον οποίο γίνεται η διακριτοποίηση του φορέα και η αριθμητική ολοκλήρωση για τη μόρφωση του μητρώου στιβαρότητας. Ο Δεικτικός Χώρος (Index Space) επιτελεί βοηθητικό ρόλο στις συναρτήσεις NURBS. Χρησιμοποιείται για τον προσδιορισμό των παραμετρικών συντεταγμένων των Control Points, των αντίστοιχων των «κόμβων» στην ισογεωμετρική ανάλυση, και για την καλύτερη εποπτεία του Knot Value Vector.

Ορισμός και ιδιότητες των συναρτήσεων και γεωμετριών B-Splines

Οι συναρτήσεις NURBS είναι μια γενικότερη, ρητή μορφή με βάρη, των B-Splines. Οι συναρτήσεις βάσης των B-Splines, για να οριστούν, χρειάζονται ένα σύνολο n+p+1 μη μειούμενων παραμετρικών συντεταγμένων που αποθηκεύονται στο Knot Value Vector, $\Xi = \left\{\xi_1, \xi_2, \cdots \xi_{n+p}, \xi_{n+p+1}\right\}$. n είναι ο αριθμός των συναρτήσεων βάσεως και p ο πολυωνυμικός βαθμός τους. Στο Knot Value Vector αντιστοιχίζουμε ένα Knot Vector το

οποίο έχει τα ίδια στοιχεία αλλά μόνο μια φορά. Με βάση το Knot Value Vector ορίζονται οι συναρτήσεις βάσης με τον αναδρομικό αλγόριθμο Cox de Boor, ως εξής:

$$N_{i,0}(\xi)=\begin{cases}1, & \text{if } \xi_i \le \xi < \xi_{i+1}\\ 0, & \text{otherwise}\end{cases}, \text{ p=0 και } N_{i,p}(\xi)=\frac{\xi-\xi_i}{\xi_{i+p}-\xi_i}N_{i,p-1}(\xi)+\frac{\xi_{i+p+1}-\xi}{\xi_{i+p+1}-\xi_{i+1}}N_{i+1,p-1}(\xi), \text{ p>0}$$

Η k παράγωγός τους προσδιορίζεται από τη σχέση: $\frac{d^k}{d^k\xi}N_{i,p}(\xi)=\frac{p!}{(p-k)!}\sum_{j=0}^{k}a_{k,j}N_{i+j,p-k}(\xi)$

Οι B-Splines έχουν την ιδιότητα του πλήρους τανυστικού γινομένου (Full Tensor Product). Υπό την έννοια αυτή, είναι εύκολο να συνδυάσουμε τις συναρτήσεις βάσης των επιμέρους παραμετρικών συντεταγμένων για να χτίσουμε τη Shape Function. Η i B-Spline συνάρτηση σχήματος ορίζεται ως:

$R_i^p(\xi)=N_{i,p}(\xi)$ σε μονοδιάστατα προβλήματα,

$R_{i,j}^{p,q}(\xi,\eta)=N_{i,p}(\xi)M_{j,q}(\eta)$ σε διδιάστατα και

$R_{i,j,k}^{p,q,r}(\xi,\eta,\zeta)=N_{i,p}(\xi)M_{j,q}(\eta)L_{k,r}(\zeta)$ σε τριδιάστατα.

Οι B-Splines έχουν μια σειρά από πολύ καλές ιδιότητες που τις κάνουν τόσο δημοφιλείς:

1. Είναι μη μηδενικές σε περιορισμένο διάστημα (support): $N_{i,p}(\xi)=0 \quad \forall\xi\notin\left[\xi_i,\xi_{i+p+1}\right)$
2. Στο δοσμένο διάστημα $\left[\xi_j,\xi_{j+1}\right)$ το πολύ (p+1) εκ των συναρτήσεων $N_{i,p}$ είναι μη μηδενικές, και οι πιθανές μη μηδενικές είναι οι: $N_{j-p,p},\ldots,N_{j,p}$.
3. Κάθε συνάρτηση μοιράζεται μέρη του μη μηδενικού διαστήματός της με 2p άλλες.
4. Οι συναρτήσεις είναι παντού μη αρνητικές $N_{i,p}(\xi)\ge 0$
5. Το άθροισμα των συναρτήσεων σχήματος σε οποιοδήποτε σημείο είναι 1. $\sum_{i=1}^{n}N_{i,p}(\xi)=1$
6. Οι συναρτήσεις έχουν συνέχεια C^{p-k} πάνω σε έναν κόμβο πολλαπλότητας k και είναι απείρως διαφορίσιμες στο εσωτερικό του διαστήματος μεταξύ των κόμβων.
7. Εκτός απ' την περίπτωση των συναρτήσεων σχήματος για p=0, όλες οι άλλες έχουν ακριβώς ένα μέγιστο.

Τα Control Points στον παραμετρικό χώρο, βρίσκονται στο μέσον του support της συναρτήσεως βάσεως που αντιστοιχούν. Με βάση την ιδιότητα (1) των B-Spline, οι παραμετρικές συντεταγμένες του i Control Point στον άξονα ξ ορίζονται ως:

$$\xi_{CP_i}=0.5\left(\xi_{i+\frac{p}{2}}+\xi_{i+\frac{p}{2}+1}\right).$$

Οι γεωμετρίες των B-Splines μπορεί να είναι καμπύλες (1-Δ), επιφάνειες (2-Δ) ή όγκοι (3-Δ) και ορίζονται ως το άθροισμα των συναρτήσεων σχήματος πολλαπλασιασμένων με τις συντεταγμένες των αντίστοιχων Control Point. Μια γεωμετρία B-Spline ορίζεται ως

$$(x,y,z) = C(\xi) = \underset{(1xn)}{\left\{N_{i,p}(\xi)\right\}^T} \underset{(nx3)}{\left\{P_i\right\}} = \sum_{i=1}^{n} N_{i,p}(\xi) \underset{(1x3)}{\left\{P_i\right\}} \quad , \quad \xi_1 \le \xi \le \xi_{n+p+1} \text{ για καμπύλη,}$$

$$(x,y,z) = S(\xi,\eta) = \sum_{i=1}^{n}\sum_{j=1}^{m} R_{i,j}^{p,q}(\xi,\eta) \underset{(1x3)}{\left\{P_{i,j,k}\right\}} = \sum_{i=1}^{n}\sum_{j=1}^{m} N_{i,p}(\xi) N_{j,q}(\eta) \underset{(1x3)}{\left\{P_{i,j}\right\}} \text{ για επιφάνεια}$$

και ένας όγκος ως

$$(x,y,z) = V(\xi,\eta,\zeta) = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{l} R_{i,j,k}^{p,q,r}(\xi,\eta,\zeta) \underset{(1x3)}{\left\{P_{i,j,k}\right\}} = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{l} N_{i,p}(\xi) N_{j,q}(\eta) N_{k,r}(\xi) \underset{(1x3)}{\left\{P_{i,j,k}\right\}}$$

Οι καμπύλες B-Splines, και κατά επέκταση οι επιφάνειες και οι όγκοι, έχουν μια σειρά από θεμιτές ιδιότητες, απόρροια των ιδιοτήτων των συναρτήσεων σχήματος από τις οποίες φτιάχνονται.

1. Είναι μια γενίκευση των καμπυλών Bezier.
2. Η C(ξ) είναι μια τμηματική πολυωνυμική καμπύλη.
3. Το πρώτο και το τελευταίο Control Point βρίσκονται πάνω στη καμπύλη και μάλιστα είναι το πρώτο και το τελευταίο σημείο της, $C(\xi_1) = P_1$ και $C(\xi_{n+p+1}) = P_n$.
4. Οι καμπύλες B-Spline έχουν την ιδιότητα το τμήμα τους $\left[\xi_{i,}\xi_{i+1}\right)$ να περιέχεται εντός του "Convex Hull" των Control Point $P_{i-p}, \ldots, P_i$.
5. Η μετακίνηση ενός Control Point P αλλάζει επηρεάζει μόνο την περιοχή κοντά στο Control point.
6. Το πολύγωνο των Control Points είναι μια γραμμική προσέγγιση της καμπύλης
7. Ενας οποιοδήποτε συνδυασμός γραμμικών μετασχηματισμών μπορεί να εφαρμοστεί στην καμπύλη με την εφαρμογή του συνδυασμού αυτού στα Control Points.
8. Κανένα επίπεδο δεν έχει περισσότερες τομές με την καμπύλη από ότι με το πολύγωνο των Control Points (Σε διδιάστατες καμπύλες, καμία γραμμή δεν έχει περισσότερες τομές με την καμπύλη από ότι με το πολύγωνο των Control Points).
9. C(ξ) είναι γραμμικός συνδυασμός των $N_{i,p}(\xi)$, επομένως η συνέχεια και η παραγωγισιμότητα της καμπύλης προκύπτουν από τις αντίστοιχες συναρτήσεις βάσης.
10. Είναι δυνατό και πολλές φορές χρήσιμο να χρησιμοποιήσουμε πολλαπλά Control Points με τις ίδιες καρτεσιανές συντεταγμένες.

Όλες οι παραπάνω ιδιότητες γενικεύονται στις 2-Δ επιφάνειες και στα 3-Δ στερεά σώματα.

Ορισμός και ιδιότητες των συναρτήσεων και γεωμετριών NURBS

Μπορούμε να θεωρήσουμε τις καμπύλη NURBS ως την προβολή της αντίστοιχης καμπύλης B-Spline με Control Points $P_i^w = (X_i, Y_i, w_i)$ σε ένα συγκεκριμένο επίπεδο με Control Points $P_i = \left(\frac{X_i}{w_i}, \frac{Y_i}{w_i} \right)$ και βάρος w_i. Στη γενικότερη περίπτωση, μπορούμε να προβάλλουμε μια γεωμετρία NURBS από το ρητό χώρο d-διαστάσεων στο χώρο των B-Splines των (d+1) διαστάσεων και αντίστροφα.

Η προβολή των Control Points των NURBS στο χώρο των B-Splines γίνεται με τον πολλαπλασιασμό του Control Point με το αντίστοιχο βάρος και με διαίρεση για την αντίστροφη προβολή. Τα Control Points που ανήκουν στις B-Splines τα ονομάζουμε προβολικά (projective) και τα συμβολίζουμε P_i^w.

$$(3D - \text{ρητός χώρος}) \qquad \underset{(nx3)}{\{P\}} = \underset{(nx3)}{\{X_i, Y_i, Z_i\}} \text{ , με βάρος } w_i \underset{/w_i}{\overset{\cdot w_i}{\rightleftarrows}}$$

$$(4D - \text{ μη ρητός χώρος}) \quad \underset{(nx4)}{\left\{P^w\right\}} = \underset{(nx4)}{\{w_i X_i, w_i Y_i, w_i Z_i, w_i\}}$$

Για να ορίσουμε τις συναρτήσεις NURBS πρέπει να ορίσουμε πρώτα μια συνάρτηση βάρους. Οι συναρτήσεις σχήματος NURBS ορίζονται με την προβολή των συναρτήσεων B-Splines ανάλογα με τη διάσταση του χώρου που δουλεύουμε ως:

1-Δ: $$W(\xi) = \sum_{i=1}^{n} N_{i,p}(\xi)\, w_i \,, \qquad R_i^p(\xi) = \frac{w_i\, N_{i,p}(\xi)}{W(\xi)}$$

2-Δ: $$W(\xi,\eta) = \sum_{i=1}^{n}\sum_{j=1}^{m} N_{i,p}(\xi) M_{j,q}(\eta)\, w_{i,j} \,, \qquad R_{i,j}^{p,q}(\xi,\eta) = \frac{w_{i.j}\, N_{i,p}(\xi) M_{j,q}(\eta)}{W(\xi,\eta)}$$

και 3-Δ: $$W(\xi,\eta,\zeta) = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{l} N_{i,p}(\xi) M_{j,q}(\eta) L_{k,r}(\zeta)\, w_{i,j,k}$$

$$R_{i,j,k}^{p,q,r}(\xi,\eta,\zeta) = \frac{w_{i.j,k}\, N_{i,p}(\xi) M_{j,q}(\eta) L_{k,r}(\zeta)}{W(\xi,\eta,\zeta)}$$

Για να βρούμε τις παραγώγους τον NURBS, απλά παραγωγίζουμε τις συναρτήσεις και προκύπτει

1-Δ: $$\frac{d}{d\xi} R_i^p(\xi) = \frac{\left(\frac{d}{d\xi} N_{i,p}(\xi)\right) W(\xi) - N_{i,p}(\xi)\left(\frac{d}{d\xi} W(\xi)\right)}{\left(W(\xi)\right)^2} w_i$$

2-Δ:
$$\frac{\partial}{\partial\xi}R_{i,j}^{p,q}(\xi,\eta)=\frac{\left(\frac{\partial}{\partial\xi}N_{i,p}(\xi)\right)W(\xi,\eta)-N_{i,p}(\xi)\left(\frac{\partial}{\partial\xi}W(\xi,\eta)\right)}{\left(W(\xi,\eta)\right)^2}w_{i,j}M_{j,q}(\eta)$$

$$\frac{\partial}{\partial\eta}R_{i,j}^{p,q}(\xi,\eta)=\frac{\left(\frac{\partial}{\partial\eta}M_{j,q}(\eta)\right)W(\xi,\eta)-M_{j,q}(\eta)\left(\frac{\partial}{\partial\eta}W(\xi,\eta)\right)}{\left(W(\xi,\eta)\right)^2}w_{i,j}N_{i,p}(\xi)$$

και ομοίως για την 3-Δ περίπτωση.

Αντίστοιχα με τον ορισμό των γεωμετριών B-Splines, οι γεωμετρίες μπορεί να είναι καμπύλες (1-Δ), επιφάνειες (2-Δ) και όγκοι (3-Δ) και ορίζονται ως άθροισμα των συναρτήσεων σχήματος πολλαπλασιασμένων με τις συντεταγμένες των αντίστοιχων Control Points. Μια γεωμετρία NURBS ορίζεται ως:

$$(x,y,z)=C(\xi)=\underset{(1xn)}{\left\{N_{i,p}(\xi)\right\}^T}\underset{(nx3)}{\left\{P_i\right\}}=\sum_{i=1}^{n}N_{i,p}(\xi)\underset{(1x3)}{\left\{P_i\right\}}\quad,\quad \xi_1\le\xi\le\xi_{n+p+1}\ \text{για καμπύλη}$$

$$(x,y,z)=S(\xi,\eta)=\sum_{i=1}^{n}\sum_{j=1}^{m}R_{i,j}^{p,q}(\xi,\eta)\underset{(1x3)}{\left\{P_{i,j,k}\right\}}=\sum_{i=1}^{n}\sum_{j=1}^{m}N_{i,p}(\xi)N_{j,q}(\eta)\underset{(1x3)}{\left\{P_{i,j}\right\}}\ \text{για επιφάνεια}$$

και ένας όγκος ως

$$(x,y,z)=V(\xi,\eta,\zeta)=\sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{l}R_{i,j,k}^{p,q,r}(\xi,\eta,\zeta)\underset{(1x3)}{\left\{P_{i,j,k}\right\}}=\sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{l}N_{i,p}(\xi)N_{j,q}(\eta)N_{k,r}(\xi)\underset{(1x3)}{\left\{P_{i,j,k}\right\}}$$

Λόγω του τρόπου δημιουργίας των NURBS από τις B-Splines, οι NURBS κληρονομούν όλες τις καλές ιδιότητες των B-Splines που είδαμε παραπάνω. Επιπλέον, παρατηρούμε ότι στην περίπτωση που τα βάρη ισούνται με 1, οι NURBS είναι απλές B-Splines που επιβεβαιώνει την παραπάνω πρότασή μας ότι τα NURBS είναι μια γενίκευση των B-Spline.

Στις γεωμετρίες NURBS μπορούμε να έχουμε περισσότερα του ενός πλέγματα με 2 τρόπους, είτε με ένα και μόνο Knot Value Vector που στο σημείο ένωσης των 2 πλεγμάτων έχουμε κόμβο (knot) με multiplicity p και συνέχεια $C^{p-p}=C^0$, είτε με διαφορετικό Knot Value Vector για κάθε πλέγμα. Ενώ η πρώτη περίπτωση είναι εύκολα εφαρμόσιμη γιατί η σύνδεση έχει ήδη επιτευχθεί, στη δεύτερη περίπτωση η σύνδεση πολλές φορές μπορεί να είναι προβληματική και όχι στεγανή (να μην συμπίπτουν απόλυτα οι επιφάνειες σύνδεσης στον πραγματικό χώρο).

Μόρφωση του μητρώου Στιβαρότητας.

Στην ισογεωμετρική ανάλυση ακολουθείται η ίδια λογική με τα πεπερασμένα στοιχεία για τη μόρφωση του μητρώου στιβαρότητας. Ο φορέας διακριτοποιείται σε στοιχεία που δημιουργούνται από τα Knot Vectors και χρησιμοποιούμε αριθμητική ολοκλήρωση Gauss για υπολογίσουμε αριθμητικά το μητρώο στιβαρότητας. Αν ο μεγαλύτερος εκ των βαθμών των πολυωνύμων στους παραμετρικούς άξονες είναι p, τότε αριθμός των Gauss Points κατά τον άξονα σε κάθε στοιχείο είναι:

$$n_{GP}^{perKnotSpan} = \begin{cases} p, \text{ για 1-Δ προβλήματα} \\ p+1, \text{ για 2-Δ ,3-Δ προβλήματα} \end{cases}$$

Οι συντεταγμένες ξ^R και τα βάρη w_ξ^R των Gauss Point συναρτήσει του αριθμού τους υπάρχουν στη βιβλιογραφία για ένα διάστημα ολοκλήρωσης (-1,1). Ανάγοντάς τις στο παραμετρικό διάστημα του στοιχείου που μας ενδιαφέρει, παίρνουμε τις παραμετρικές συντεταγμένες και τα αντίστοιχα βάρη:

$$\xi = \frac{(\xi_{i+1} - \xi_i)\xi^R + (\xi_{i+1} + \xi_i)}{2}$$

$$w^{GP\xi} = \frac{(\xi_{i+1} - \xi_i)}{2} w_\xi^R$$

Θεωρώντας γραμμική ελαστικότητα, μπορούμε να μορφώσουμε τον πίνακα [E] για περιπτώσεις μονοδιάστατης (δικτυώματος), διδιάστατης (επίπεδης έντασης ή επίπεδης παραμόρφωσης) και τριδιάστατης ελαστικότητας.

$$\underset{(1x1)}{\left[E_{Truss}\right]} = E$$

$$\underset{(3x3)}{\left[E_{Plane\ Stress}\right]} = \frac{E}{1-\nu^2}\begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{1-\nu}{2} \end{bmatrix}, \underset{(3x3)}{\left[E_{Plane\ Strain}\right]} = \frac{E}{(1+\nu)(1-2\nu)}\begin{bmatrix} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & \frac{1-2\nu}{2} \end{bmatrix}$$

$$\underset{(6x6)}{\left[E_{3D\ Elasticity}\right]} = \frac{E}{(1-\nu)(1-2\nu)}\begin{bmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0 \\ \nu & 1-\nu & \nu & 0 & 0 & 0 \\ \nu & \nu & 1-\nu & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1-2\nu}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1-2\nu}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1-2\nu}{2} \end{bmatrix}$$

Ο Ιακωβιανός πίνακας και ο αντίστροφός του για τις μεταβάσεις από τον πραγματικό στον παραμετρικό χώρο είναι ο ακόλουθος για τις 3 περιπτώσεις ελαστικότητας:

1D Elasticity	2D Elasticity	3D Elasticity
$\underset{(1x1)}{[J]} = \underset{(1x1)}{\left[\frac{dx}{d\xi}\right]} = \underset{(1xn)}{\{N_\xi\}^T}\underset{(nx1)}{\{P\}}$	$\underset{(2x2)}{[J]} = \underset{(2x2)}{\begin{bmatrix}\frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi}\\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta}\end{bmatrix}} = \underset{(2xn)}{\begin{Bmatrix}\{N_\xi\}^T_{(1xn)}\\ \{N_\eta\}^T_{(1xn)}\end{Bmatrix}}\underset{(nx2)}{\{P\}}$	$\underset{(3x3)}{[J]} = \underset{(3x3)}{\begin{bmatrix}\frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} & \frac{\partial z}{\partial \xi}\\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta}\\ \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} & \frac{\partial z}{\partial \zeta}\end{bmatrix}} = \underset{(3xn)}{\begin{Bmatrix}\{N_\xi\}^T_{(1xn)}\\ \{N_\eta\}^T_{(1xn)}\\ \{N_\zeta\}^T_{(1xn)}\end{Bmatrix}}\underset{(nx3)}{\{P\}}$
$\underset{(1x1)}{[J]^{-1}} = \frac{1}{\underset{(1x1)}{[J]}} = \frac{1}{\det([J])}$	$\underset{(2x2)}{[J]^{-1}} = \underset{(2x2)}{\begin{bmatrix}J^*_{11} & J^*_{12}\\ J^*_{21} & J^*_{22}\end{bmatrix}} = \frac{1}{\det[J]}\begin{bmatrix}J_{22} & -J_{12}\\ -J_{21} & J_{11}\end{bmatrix}$ where $\det[J] = J_{11}J_{22} - J_{21}J_{12}$	$\underset{(3x3)}{[J]^{-1}} = \begin{bmatrix}J^*_{11} & J^*_{12} & J^*_{13}\\ J^*_{21} & J^*_{22} & J^*_{23}\\ J^*_{31} & J^*_{32} & J^*_{33}\end{bmatrix}$

Τα μητρώα παραμορφώσεως για μονοδιάστατη ελαστικότητα 1-Δ είναι:

Τα επιμέρους μητρώα παραμόρφωσης: $\underset{(1x1)}{[B_1]} = \left[\frac{1}{J_{11}}\right]$ και $\underset{(1xn)}{[B_2(\xi)]} = \underset{(1xn)}{\{R_{,\xi}\}^T}$.

Το ολικό μητρώο παραμόρφωσης: $\underset{(1xn)}{[B(\xi)]} = \underset{(1x1)}{[B_1(\xi)]}\underset{(1xn)}{[B_2(\xi)]}$

Το μητρώο στιβαρότητας:

$$\underset{(nxn)}{[K]} = \int_1 \underset{(nx1)}{[B(\xi)]^T}\underset{(1x1)}{[E]}\underset{(1xn)}{[B(\xi)]}A dx = \int_{\xi_1}^{\xi_{n+p+1}} \underset{(nx1)}{[B(\xi)]^T}\underset{(1x1)}{[E]}\underset{(1xn)}{[B(\xi)]}A\det\left([J(\xi)]\right)d\xi$$

Και η αριθμητική του ολοκλήρωση:

$$\underset{(nxn)}{[K]} = \sum_{i=1}^{n_{GP\xi}}\left(\underset{(nx1)}{[B(\xi_i)]^T}\underset{(1x1)}{[E]}\underset{(1xn)}{[B(\xi_i)]}A\det\left([J(\xi_i)]\right)w_i^{GP}\right)$$

Στην περίπτωση 2-Δ:

$$\underset{(3x4)}{[B_1(\xi,\eta)]} = \frac{1}{\det\left(\underset{(2x2)}{[J]}\right)}\underset{(3x4)}{\begin{bmatrix}J_{22} & -J_{12} & 0 & 0\\ 0 & 0 & -J_{21} & J_{11}\\ -J_{21} & J_{11} & J_{22} & -J_{12}\end{bmatrix}}$$

$$\underset{(4x2N)}{[B_2(\xi,\eta)]} = \begin{bmatrix}R_{1,\xi} & 0 & R_{2,\xi} & 0 & \cdots & \cdots & R_{N,\xi} & 0\\ R_{1,\eta} & 0 & R_{2,\eta} & 0 & \cdots & \cdots & R_{N,\eta} & 0\\ 0 & R_{1,\xi} & 0 & R_{2,\xi} & \cdots & \cdots & 0 & R_{N,\xi}\\ 0 & R_{1,\eta} & 0 & R_{2,\eta} & \cdots & \cdots & 0 & R_{N,\eta}\end{bmatrix}$$

$$\underset{(3x2N)}{[B(\xi,\eta)]} = \underset{(3x4)}{[B_1(\xi,\eta)]}\underset{(4x2N)}{[B_2(\xi,\eta)]}$$

και το μητρώο στιβαρότητας:

$$\underset{(2Nx2N)}{[K]} = \int_{\xi_1}^{\xi_{n+p+1}} \int_{\eta_1}^{\eta_{m+q+1}} \underset{(2Nx3)}{[B(\xi,\eta)]^T} \underset{(3x3)}{[E]} \underset{(3x2N)}{[B(\xi,\eta)]} t \det\left([J(\xi,\eta)]\right) d\xi d\eta$$

και με αριθμητική ολοκλήρωση:

$$\underset{(2Nx2N)}{[K]} = \sum_{i=1}^{n_{GP\xi}} \sum_{j=1}^{n_{GP\eta}} \left(\underset{(2Nx3)}{\left[B(\xi_i,\eta_j)\right]^T} \underset{(3x3)}{[E]} \underset{(3x2N)}{\left[B(\xi_i,\eta_j)\right]} t \det\left(\left[J(\xi_i,\eta_j\right]\right) w_i^{GP\xi} w_j^{GP\eta} \right)$$

Για 3-Δ ελαστικότητα, τα μητρώα παραμορφώσεως είναι:

$$\underset{(6x9)}{\left[B_{1(\xi,\eta,\zeta)}\right]} = \begin{bmatrix} J_{11}^* & J_{12}^* & J_{13}^* & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & J_{21}^* & J_{22}^* & J_{23}^* & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & J_{31}^* & J_{32}^* & J_{33}^* \\ J_{21}^* & J_{22}^* & J_{23}^* & J_{11}^* & J_{12}^* & J_{13}^* & 0 & 0 & 0 \\ 0 & 0 & 0 & J_{31}^* & J_{32}^* & J_{33}^* & J_{21}^* & J_{22}^* & J_{23}^* \\ J_{31}^* & J_{32}^* & J_{33}^* & 0 & 0 & 0 & J_{11}^* & J_{12}^* & J_{13}^* \end{bmatrix}_{(6x9)}$$

$$\underset{(9x3n)}{[B_{2(\xi,\eta,\zeta)}]} = \begin{bmatrix} R_{1,\xi} & 0 & 0 & R_{2,\xi} & 0 & 0 & \cdots & R_{n,\xi} & 0 & 0 \\ R_{1,\eta} & 0 & 0 & R_{2,\eta} & 0 & 0 & \cdots & R_{n,\eta} & 0 & 0 \\ R_{1,\zeta} & 0 & 0 & R_{2,\zeta} & 0 & 0 & \cdots & R_{n,\zeta} & 0 & 0 \\ 0 & R_{1,\xi} & 0 & 0 & R_{2,\xi} & 0 & \cdots & 0 & R_{n,\xi} & 0 \\ 0 & R_{1,\eta} & 0 & 0 & R_{2,\eta} & 0 & \cdots & 0 & R_{n,\eta} & 0 \\ 0 & R_{1,\zeta} & 0 & 0 & R_{2,\zeta} & 0 & \cdots & 0 & R_{n,\zeta} & 0 \\ 0 & 0 & R_{1,\xi} & 0 & 0 & R_{2,\xi} & \cdots & 0 & 0 & R_{n,\xi} \\ 0 & 0 & R_{1,\eta} & 0 & 0 & R_{2,\eta} & \cdots & 0 & 0 & R_{n,\eta} \\ 0 & 0 & R_{1,\zeta} & 0 & 0 & R_{2,\zeta} & \cdots & 0 & 0 & R_{n,\zeta} \end{bmatrix}$$

και $\underset{(6x3N)}{\left[B(\xi,\eta,\zeta)\right]} = \underset{(6x9)}{\left[B_1(\xi,\eta,\zeta)\right]} \underset{(9x3N)}{[B_2(\xi,\eta,\zeta)]}$

Το μητρώο στιβαρότητας προσδιορίζεται ως:

$$\underset{(3Nx3N)}{[K]} = \int_{\xi_1}^{\xi_{n+p+1}} \int_{\eta_1}^{\eta_{m+q+1}} \int_{\zeta_1}^{\zeta 1+r+1} \underset{(3Nx6)}{[B(\xi,\eta,\zeta)]^T} \underset{(6x6)}{[E]} \underset{(6x3N)}{[B(\xi,\eta,\zeta)]} \det\left([J(\xi,\eta,\zeta)]\right) d\xi d\eta d\zeta$$

και με αριθμητική ολοκλήρωση:

$$\underset{(3Nx3N)}{[K]} = \sum_{i=1}^{n_{GP\xi}} \sum_{j=1}^{n_{GP\eta}} \sum_{k=1}^{n_{GP\zeta}} \left(\underset{(3Nx6)}{\left[B(\xi_i,\eta_j,\zeta_k)\right]^T} \underset{(6x6)}{[E]} \underset{(6x3N)}{\left[B(\xi_i,\eta_j,\zeta_k)\right]} \det\left(\left[J(\xi_i,\eta_j,\zeta_k\right]\right) w_i^{GP\xi} w_j^{GP\eta} w_k^{GP\zeta} \right)$$

Εξωτερικά φορτία και συνοριακές συνθήκες

Δεν είναι εύκολο να δούμε τη φυσική σημασία όταν ασκούμε φορτία στα Control Points, γιατί κάθε Control Point γενικά δεν είναι πάνω στη γεωμετρία και φόρτιση σε αυτό αντιστοιχεί σε φόρτιση σε μια ολόκληρη περιοχή του φορέα. Όταν όμως επιβάλλουμε φορτία σε γωνιακά Control Points (γωνιακά με την έννοια ότι βρίσκονται στα άκρα του Knot Value Vector και επομένως είναι παρεμβολικά στη γεωμετρία μας) εφαρμόζονται πράγματι στο ίδιο σημείο του φορέα λόγω των παρεμβολικών συναρτήσεων σχήματος. Σε διαφορετική περίπτωση, όταν είναι κατανεμημένα πάνω στον φορέα, βρίσκουμε τις ισοδύναμες δράσεις που ασκούνται στα Control Points:

$$\underset{(Nx3)}{\{F\}} = \int_V \underset{(Nx1)}{\{R(x,y,z)\}}\, \underset{(1x3)}{f(x,y,z)}\, dV = \int_{\xi_1}^{\xi_{n+p+1}} \int_{\eta_1}^{\eta_{m+q+1}} \int_{\zeta_1}^{\zeta_{l+r+1}} \underset{(Nx1)}{\{R(\xi,\eta,\zeta)\}}\, \underset{(1x3)}{f(\xi,\eta,\zeta)} \det[J]\, d\zeta d\eta d\xi$$

Οι συνοριακές συνθήκες εμφανίζουν αντίστοιχη δυσκολία. Δεσμεύοντας ένα Control Point δεσμεύουμε μερικώς την περιοχή του φορέα που επηρεάζει. Μπορούμε όμως να δεσμεύσουμε αποτελεσματικά μια ολόκληρη πλευρά του φορέα. Αυτό το κάνουμε δεσμεύοντας όλα τα Control Points στην πλευρά εκείνη.

Στη συνέχεια, αφού έχουμε μορφώσει το μητρώο στιβαρότητας, έχουμε επιβάλλει εξωτερικές φορτίσεις και συνοριακές συνθήκες, μπορούμε να λύσουμε την εξίσωση και να βρούμε τις άγνωστες μετακινήσεις: $\{L_f\} = [K_{ff}]\{D_f\} \Rightarrow \{D_f\} = [K_{ff}]^{-1}\{L_f\}$

Όταν πάρουμε τις μετακινήσεις μπορούμε να προσδιορίσουμε τα πεδία μετακινήσεων, τάσεων και τροπών σε όλο το φορέα:

Μετακινήσεις: $d(\xi,\eta,\zeta) = \{R(\xi,\eta,\zeta)\}^T \{D\}$

Τροπές: $\{\varepsilon(\xi,\eta,\zeta)\} = [B(\xi,\eta,\zeta)]\{D\}$

Τάσεις: $\{\sigma\} = [E][B(\xi,\eta,\zeta)]\{D\}$

Επαναδιακριτοποίηση

Για να πάρουμε καλύτερα αποτελέσματα από την ανάλυσή μας αλλά και για να εφαρμόσουμε μια σειρά από χρήσιμους αλγορίθμους χρησιμοποιούμε τεχνικές επαναδιακριτοποίησης. Έχουμε 5 είδη επαναδιακριτοποίησης:

1. h-refinement
2. reverse h-refinement
3. p-refinement
4. reverse p-refinement
5. k-refinement

Όλες οι τεχνικές επαναδιακριτοποίησης βασίζονται στη λογική ότι πριν και μετά τη διαδικασία θα έχω την ίδια αρχική απαραμόρφωτη γεωμετρία. Σε κάθε περίπτωση βρίσκουμε τις νέες καρτεσιανές συντεταγμένες των Control Point ως

$$\underset{(mx3)}{\left\{P^F\right\}} = \underset{(mxn)}{\left[T^{FI}\right]} \underset{(nx3)}{\left\{P^I\right\}}$$

και επαναλαμβάνουμε τη διαδικασία μόρφωσης του μητρώου στιβαρότητας, επιβολής εξωτερικών συνθηκών και επίλυσης. Επίσης, σε όλες τις περιπτώσεις επαναδιακριτοποίησης, λόγω της Full Tensor Product φύσης των NURBS, μπορούμε να επαναδιακριτοποιήσουμε ξεχωριστά σε κάθε παραμετρικό άξονα και με όποια σειρά αξόνων θέλουμε.

Αναλυτικότερα:

Το h-refinement, η εισαγωγή κόμβου (knot) που συνεπάγεται και εισαγωγή Control Point, υπάρχει σαν τεχνική και στα FEA. Εισάγουμε Knot Values στο Knot Value Vector και παίρνουμε νέες συναρτήσεις σχήματος και Control Points (πυκνότερα).

Το reverse h-refinement είναι η αφαίρεση κόμβων. Δεν είναι πάντα δυνατό να αφαιρέσουμε κόμβους, για το λόγο αυτό ο αλγόριθμος για reverse h-refinement πρέπει αρχικά να ελέγξει ότι οι κόμβοι που επιλέξαμε προς αφαίρεση πράγματι αφαιρούνται και αφετέρου να βρει τις νέες συντεταγμένες των Control Point.

Το p-refinement ή order elevation, εμπλουτίζει τη βάση των συναρτήσεων σχήματος με την αύξηση του πολυωνυμικού βαθμού. Η διαδικασία περιλαμβάνει την εισαγωγή κόμβων (h-refinement) για τη μετατροπή σε ένα σύνολο από Bezier curves, στη συνέχεια την αύξηση του βαθμού των πολυωνύμων Bezier και κατόπιν αφαίρεση των περιττών knot values για μετατροπή ξανά σε B-Spline. Η τελική πολλαπλότητα των ενδιάμεσων κόμβων αυξάνεται όσο και ο βαθμός των πολυωνύμων ώστε να διατηρηθεί η συνέχεια στους κόμβους.

Το reverse p-refinement ή order reduction, μειώνει το βαθμό των πολυωνύμων. Η διαδικασία είναι αντίστοιχη με το p-refinement. Αποδόμηση των B-Splines σε Bezier, μείωση του βαθμού του πολυωνύμου και ξανά αφαίρεση των περιττών knot values ώστε να μετατραπεί ξανά σε B-Spline. Ομοίως με την αντίστοιχη τεχνική reverse h-refinement, γίνεται έλεγχος αν πραγματικά μπορεί να μειωθεί ο βαθμός χωρίς να χαθεί η ακρίβεια στη γεωμετρία. Αν γίνεται, τότε μειώνεται και η τελική πολλαπλότητα των κόμβων στο knot value vector ώστε η συνέχεια να παραμείνει η ίδια.

Τέλος, το k-refinement είναι μια νέα τεχνική μοναδική στην ισογεωμετρική ανάλυση. Εφαρμόζεται αρχικά ένα p-refinement για την αύξηση του βαθμού των πολυωνύμων και στη συνέχεια ένα h-refinement το οποίο εισάγει κόμβους με υψηλή συνέχεια. Ο λόγος της υψηλής συνέχειας, είναι ότι οι κόμβοι εισήχθησαν μετά το p-refinement και έτσι δεν αυξήθηκε η πολλαπλότητά τους.

Στη διπλωματική αυτή προσπαθήσαμε να εισάγουμε και έναν νέο τρόπο ιεραρχικής επαναδιακριτοποίησης. Ο λόγος είναι ότι σε μεγάλες κατασκευές, οι βαθμοί ελευθερίας είναι πολλοί και το κόστος εξαρχής υπολογισμού και αντιστροφής του μητρώου στιβαρότητας είναι αρκετά μεγάλο. Μορφώνοντας ιεραρχικά το μητρώο στιβαρότητας, δηλαδή εκφράζοντάς το συναρτήσει του αρχικού, κερδίζουμε αυτό το κόστος και έχουμε να υπολογίσουμε μόνο τα νέα K.

$$\underset{(MxM)}{\left[K^{Final}\right]} \simeq \begin{bmatrix} \underset{(NxN)}{\left[K^{Initial}\right]} & \underset{(NxQ)}{\left[K_{en}^{Final}\right]} \\ \underset{(QxN)}{\left[K_{ne}^{Final}\right]} & \underset{(QxQ)}{\left[K_{nn}^{Final}\right]} \end{bmatrix}$$

Συνδέοντας τις αρχικές με τις τελικές συναρτήσεις σχήματος φτάσαμε στην παραπάνω έκφραση του νέου μητρώου στιβαρότητας. Παρατηρούμε ότι δεν υπάρχει σύμβολο ίσον αλλά περίπου ίσον. Ο λόγος είναι ότι το ακριβές μητρώο είναι της μορφής

$$\underset{(MxM)}{\left[K^{Final}\right]} \simeq \begin{bmatrix} \underset{(NxN)}{\left[K^{Initial}\right]}+\underset{(NxN)}{\left[\delta K\right]} & \underset{(NxQ)}{\left[K_{en}^{Final}\right]} \\ \underset{(QxN)}{\left[K_{ne}^{Final}\right]} & \underset{(QxQ)}{\left[K_{nn}^{Final}\right]} \end{bmatrix}$$

Ωστόσο εμείς επιλέγουμε να αγνοήσουμε το μητρώο [δK] για να αποκτήσουμε την ιεραρχική μορφή. Φροντίζουμε φυσικά το [δK] να είναι αρκετά μικρό ώστε να μην επηρεάσει τη σύγκλιση. Αναλυτικά οι μορφές των νέων μητρώων που χρειάζεται να υπολογιστούν, δίνονται παρακάτω:

$$\underset{(NxN)}{\left[\delta K\right]} = \underset{(NxN)}{\left[T_e^m\right]^T} \underset{(NxN)}{\left[K^I\right]} \underset{(NxN)}{\left[T_e^m\right]} - \underset{(NxN)}{\left[K^I\right]}$$

$$\underset{(NxQ)}{\left[K_{en}^F\right]} = \underset{(NxN)}{\left[T_e^m\right]^T} \underset{(NxN)}{\left[K^I\right]} \underset{(NxQ)}{\left[T_n^m\right]}$$

$$\underset{(QxN)}{\left[K_{ne}^F\right]} = \underset{(QxN)}{\left[T_n^m\right]^T} \underset{(NxN)}{\left[K^I\right]} \underset{(NxN)}{\left[T_e^m\right]}$$

$$\underset{(QxQ)}{\left[K_{nn}^F\right]} = \underset{(QxN)}{\left[T_n^m\right]^T} \underset{(NxN)}{\left[K^I\right]} \underset{(NxQ)}{\left[T_n^m\right]}$$

όπου οι πίνακες $\left[T_e^m\right], \left[T_n^m\right]$ προσδιορίζονται από τους πίνακες μετασχηματισμού των συντεταγμένων των Control Point.

1 The concept of Isogeometric Analysis

1.1 Finite Element Method

The finite element analysis method is a way of numerically solving partial differential equations. Its general nature allows us to deploy it with special adjustments in countless applications in all fields of engineering. Solid and fluid mechanics, bioengineering, heat transfer and acoustics are just a few fields of interest. Today, any modern engineering project in one way or another uses finite element software.

1.1.1 Evolution of the Finite Element Method

1.1.1.1 Historical Overview

Finite element method is originated from the need to solve complicated elasticity and structural analysis problems in civil and aeronautical engineering. The first results for PDEs were by Rayleigh, Ritz and Galerkin followed by early attempts by Courant and Hrennikoff to solve numerically partial differential equations. All such successful attempts shared common ground in that all of them involved mesh discretization of a continuous domain into a set of discrete sub-domains. From that base developed the finite elements along with other methods called finite differences and finite volumes.

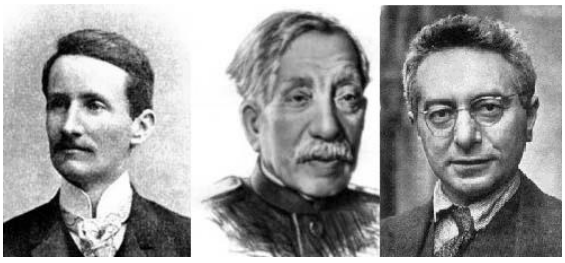


Figure 1.1. Engineer Ritz and mathematicians Galerkin and Courant.

The finite element method obtained its real impetus in the 1960s and 1970s. The linear triangular element, the simplest and still widely used element, can be traced back to Courant who coined it to solve the torsion problem in 1943. The method was firstly applied by J. Argyris during WWII with linear triangular elements in his attempt to simulate swept-back airplane wings for the Royal Aeronautical Society of London. Argyris' research was very successful and was classified at the time. His results and his work with Keley were published years later in 1960 and the method gained momentum.

The name "FiniteElements" was introduced by Clough in1960. The triangular elements' volume extension, the linear tetrahedron, appeared in 1962 by Gallagher and, the following years, the work of Taig 1961, Irons 1966, Zienkiewicz and Cheung 1968 resulted in the isoparametric elements, probably the most important concept in the history of FEA.

Based on the isoparametric concept, there were developed regular elements such as a square or a cube in the parent domain that had the capacity to take on a smoothly curved shape in the physical space. The spaces constructed, satisfied both basic mathematical convergence criteria and also useful physical attributes in problems of mechanics. The isoparametric elements became widely used and the curved quadrilateral and hexahedral elements became quite popular in solid mechanics applications. From then on, numerous researchers have devoted their work in the development of finite element technology.

1.1.1.2 Nowadays

The method started in 1940 has developed tremendously up to date. When first implemented by John Argyris for analysis of airplane wings, the current advanced computer used, could solve a linear system of maximum 64 unknowns. At present, with the massive investment and evolution on computers, we are looking forward to the 2018 target of one hexaflop computing power, meaning 10^{18} floating point operations per second. We are now able to solve equations with thousands or millions of unknowns and that has taken capabilities and potential of finite elements and generally numerical methods to a whole new level.

The evolution of computer technology led to powerful but affordable computers and thus nowadays every big industry has its own computer department. The available computer infrastructure in businesses increased the interest and made finite elements approachable to a great many engineers. Furthermore, released open source codes, constantly invented new techniques and governmental funds supporting financially the research interest in the field have given finite elements further impetus.

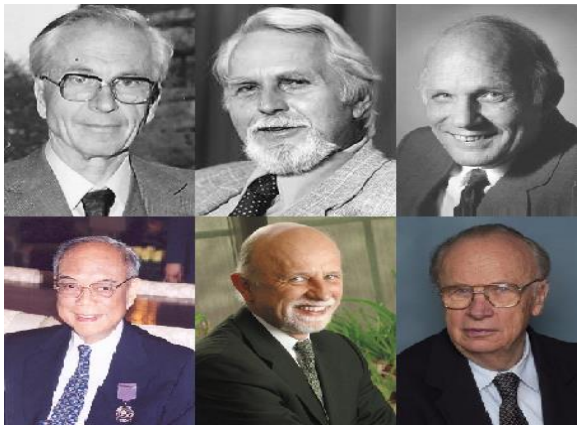


Figure 1.2. Argyris, Zinkiewicz, Taylor, Cheung, Belytchko, Babuska
All great researchers with significant contribution in the development of FEM.

1.1.1.3 Challenges

When a problem is solved, several others take its place, always growing in number. Computer power is increased but instead of solving static equilibrium we now try to simulate dynamic systems and optimize the design of a structure. With modern applications there is need for accurate and efficient software solutions. Fortunately, both aspects attract research interest. The hope for efficiency seems to reside in parallel programming, at first with CPUs with more than one cores and currently with GPU's great numbers of processors. Such implementations have been made by [4].

1.1.2 Basic Idea

The FEM is based on the concept of approximating the solution field of displacements in solid mechanics on any internal point of the model through a number of nodal displacements. The method is general and the displacement field in solid mechanics could be the pressure field in fluid mechanics or difference of voltage in electromagnetism problems.

FEM usually generates a mesh and through that defines the elements. Less often meshless methods are utilized and instead of a mesh we have just a cloud of nodes dispersed over the model. The standard ordinary methods use piecewise polynomials called shape functions to express the displacement of any internal point in respect to element's nodal displacements.

$$\underset{(3x1)}{\{U(x,y,z)\}} = \underset{(3x3n_e)}{[N^e]}\underset{(3n_ex1)}{\{d^e\}}$$

where d are the displacements of the internal point, n_e is the number of nodes of the element, D are the nodal displacements and N are the corresponding shape functions.

Generalizing the above equation for the whole structure, we get the displacement at any point as a function of the structure's nodal displacements. In this case n is the number of all the nodes in the structure.

$$\underset{(3x1)}{\{U(x,y,z)\}} = \underset{(3x3n)}{[N^S]}\underset{(3nx1)}{\{d^S\}}$$

Through differentiation of the point's displacement we can get the strain there as a function of the nodal displacements. The matrix B connecting nodal displacements and strains on a point, is called the Deformation Matrix.

$$\underset{(6x1)}{\{\varepsilon(x,y,z)\}} = \underset{(6x3n)}{[B]}\underset{(3nx1)}{\{d^S\}}$$

$$\text{where } \underset{(6x1)}{\{\varepsilon(x,y,z)\}} = \{\varepsilon_X \quad \varepsilon_Y \quad \varepsilon_Z \quad \gamma_{XY} \quad \gamma_{YZ} \quad \gamma_{ZX}\}^T$$

Depending on the constitutive law, we can connect the stress and strain vectors at any point.

$$\underset{(6x1)}{\{\sigma(x,y,z)\}} = \underset{(6x6)}{[E]}\,\underset{(6x1)}{\{\varepsilon(x,y,z)\}}$$

$$\underset{(6x1)}{\{\sigma(x,y,z)\}} = \{\sigma_X \quad \sigma_Y \quad \sigma_Z \quad \sigma_{XY} \quad \sigma_{YZ} \quad \sigma_{ZX}\}^T$$

The local stiffness matrix for each element of the model is calculated as

$$\underset{(3n_e x 3n_e)}{[k]} = \int_V \underset{(3n_e x 6)}{[B(x,y,z)]^T}\,\underset{(6x6)}{[E]}\,\underset{(6x3n_e)}{[B(x,y,z)]}\,dV$$

and due to the usual inability to get $[k]$ analytically we deploy numerical methods to evaluate it.

Afterwards, by adding each element's contribution to the total stiffness matrix of the structure, we get the stiffness matrix $\underset{(3nx3n)}{[K]}$.

We determine the element's equivalent nodal loads of any internal distributed loading

$$\underset{(3x1)}{\{r_e\}} = \int_{V_e} \underset{(3nex3)}{[N]^T}\,\underset{(3x1)}{\{f\}}\,dV_e$$

The global load vector $\underset{(3n_e x1)}{\{R\}}$ is formed by the local equivalent loads and any concentrated loads applied directly on the nodes.

Finally, we determine the unknown displacements by inverting the stiffness matrix of the free Control Points

$$\underset{(3nx1)}{\{R_f^S\}} = \underset{(3nx3n)}{[K_{ff}]}\,\underset{(3nx1)}{\{D_f^S\}}$$

1.2 Computer Aided Design

1.2.1 The evolution of CAD

1.2.1.1 Historical Overview

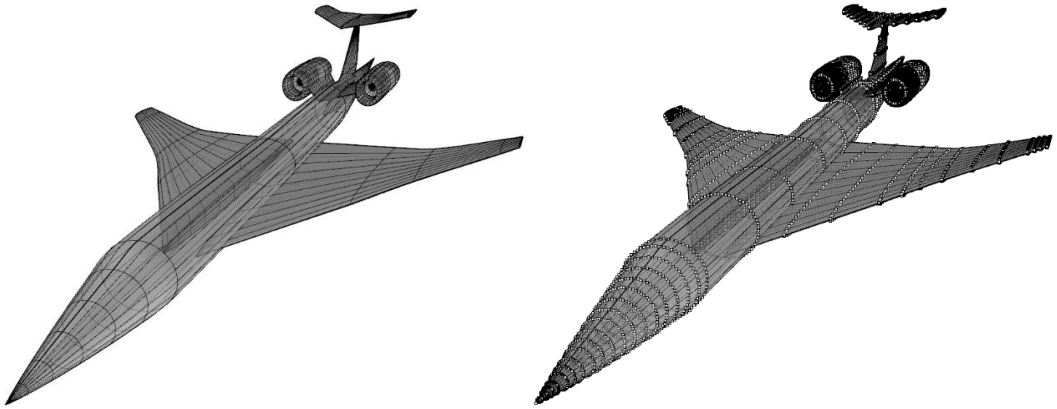
With the industrial evolution, difficult projects to be processed in the conventional way of hand-drawing emerged. In 1950s, in automotive, shipyard and aircraft industry, large scale projects that required high precision could not be handled anymore. First, smooth polynomial lines called splines by Schoenberg in 1946 were introduced to provide a curve fitting tool. Two French engineers from the automotive industry, Pierre Bezier from Renault and Paul de Casteljau from Citroen, set the foundations of the current spline toolbox by introducing the Bezier curves, named after the first who published his work in the field. With the fast development of computers and the new capabilities of CAD systems, far beyond their initial purpose of simply reproducing manual with electronic drafting, the cost benefit for companies to adopt them became apparent.

The development was rapid. Bezier curves were capable of representing a wide variety of curves but were not flexible enough. Riesenfeld in 1972 introduced the B-splines, standing for basis splines, which were much more flexible as they were defined in a way to have minimal support with respect to a given degree, smoothness and domain partition. Their generalization, called NURBS, were introduced by Versprille in 1975 and overcame the B-spline's inability to accurately represent specific geometris, like conic sections. Boeing was the first to industrialize NURBS and, in time, their native flexibility, stable mathematical procedures and simplicity had successfully established NURBS representation as a standard in CAD industry. Of course, CAD is an ever growing technology with new techniques and types of splines like T-and Polycube splines or Subdivision surfaces emerging constantly. Nonetheless, NURBS technology with its already developed and advanced software, still holds the majority of the market.

1.2.1.2 Nowadays.

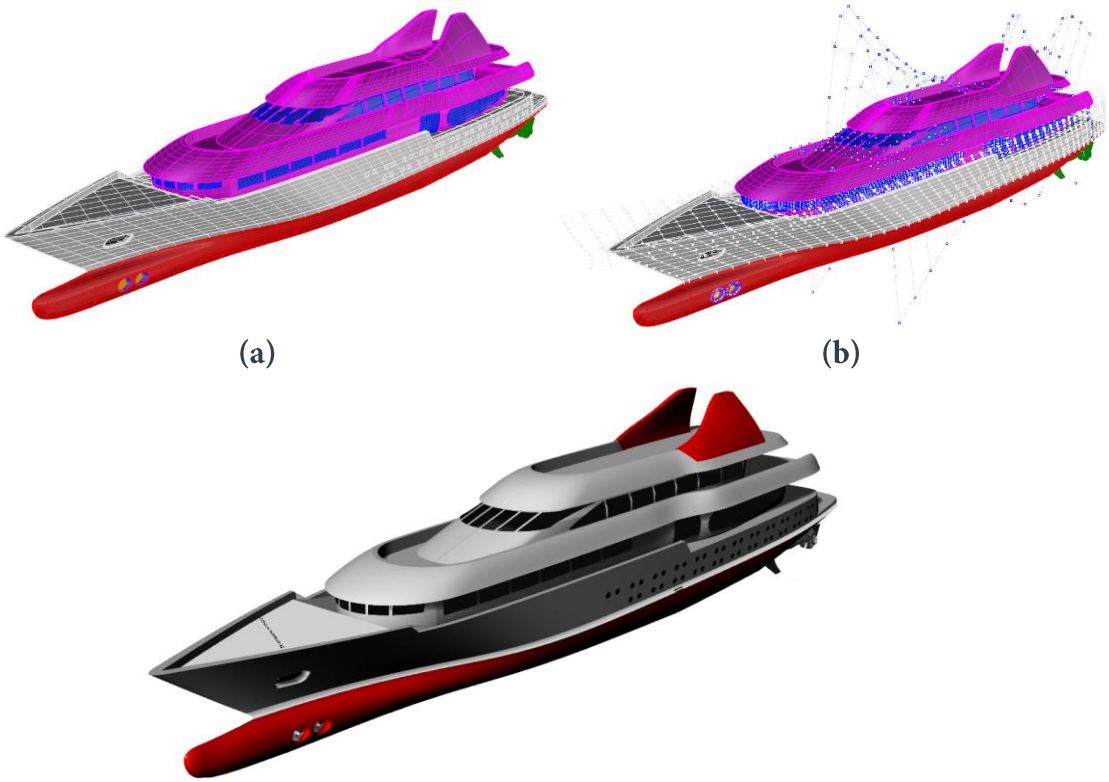
Nowadays we can say that Computer Aided Design or CAD is the use of computer systems to assist in the creation, modification, analysis or optimization of a design. The description incorporates a vast number of applications but the CAD indeed has impact in any number of industrial products in our everyday life. Its use is obvious in large scale projects like cars, buildings, aerospace engineering but is hidden in other projects too that we do not think of, namely in computer animation for special effects in movies, transistor arrangement on a CPU or prosthetics in medicine. The numerous applications it has and the technologies dependent on CAD, have given it great economical significance and today the industry has

gained an ever increasing impetus overcoming more and more difficult challenges presented.



(a) (b)

Figure 1.3. Business Jet modeled with NURBS.
(a) NURBS model with knot lines.
(b) NURBS model with knot lines and control points.
(https://grabcad.com/library/sst-1)



(a) (b)

(c)

Figure 1.4. Luxurious Yacht modeled with NURBS.
(a) NURBS model with Knot lines.
(b) NURBS model with Knot lines and Control Points.
(c) NURBS model rendered.
(https://grabcad.com/library/megayacht)

1.3 Isogeometric Analysis

For a successful simulation of a problem nowadays, we usually require both exact geometry representation and accurate engineering results. The two methods, however, Finite Elements (the CAE technologies– Computer Aided Engineering) and CAD (Computer Aided Design) evolved differently in the flow of time. Currently, the two technologies use different methods and different shape functions. That way, after the designer finishes simulating the model with exact representation, passes the result to the engineer who tries to approximate the geometry given by the CAD designer with a new mesh generation and the use of new shape functions. It is obvious that this is a tedious and time consuming procedure. By looking up in the literature we see that today, this integration can sum up to 80% of the total analysis time. If we take into consideration that this integration process has to be repeated several times as the engineer gets a solution and then meshes again to get a better one, we realize that we can save a lot of time by investing in reducing that time.

The engineers' answer came in the form of a publication back in 2005 by Hughes, Cottrell and Bazilevs. Although earlier attempts had been made, those where the first researchers to systematically approach the problem. Their idea was to utilize the same shape functions widely used in CAD literature, NURBS, to approximate the solution field. They practically reversed the isoparametric concept. Instead of using the shape functions of the solution field approximation to define the geometry, they used the CAD's shape functions for geometry for analysis. The first results were very encouraging, the CAD Shape functions had very good native attributes suitable for their use in finite element analysis. From that point, the research in the field is rapidly attracting interest and the amount of publications in the field grows exponentially each year.

Figure 1.5. A.Cottrell, T.Hughes and Y.Bazilevs.
Originators of Isogeometric Analysis and authors of the first and only book on the topic.

In Isogeometric Analysis (IGA), CAD and FEA tend to merge, outlining a bright future of closer collaboration and ideally integration of the two technologies. Fortunately, that doesn't mean that FEA has to be reinvented. Changing the basis functions leads to a series of adaptations in existing codes but the codes' architectures remain the same in principle. There is also an extensive variety of shape functions candidates for use in IGA. Shape function types develop along the IGA as CAD industry surely benefits from this integration attempt. In CAD industry, most widely used are still NURBS, but many new spline technologies are announced every year, overcoming the drawbacks of their predecessors. NURBS are not very efficient for local refinement or patch merging but the recent technology of T-Splines overcomes those drawbacks and manages to offer truly local refinement schemes and water tight patch connections. It naturally has its own drawbacks like the complexity of the algorithms used, the cases it can be applied and the fact that is a relatively new technology, not yet widely adopted by software companies. There are several promising shape functions coming along with IGA, suitable for more specific problems, namely hierarchical B-Splines, Polycube Splines or Subdivision surfaces, ensuring that the engineer using IGA will have alternatives to approach the problem's solution if they are needed.

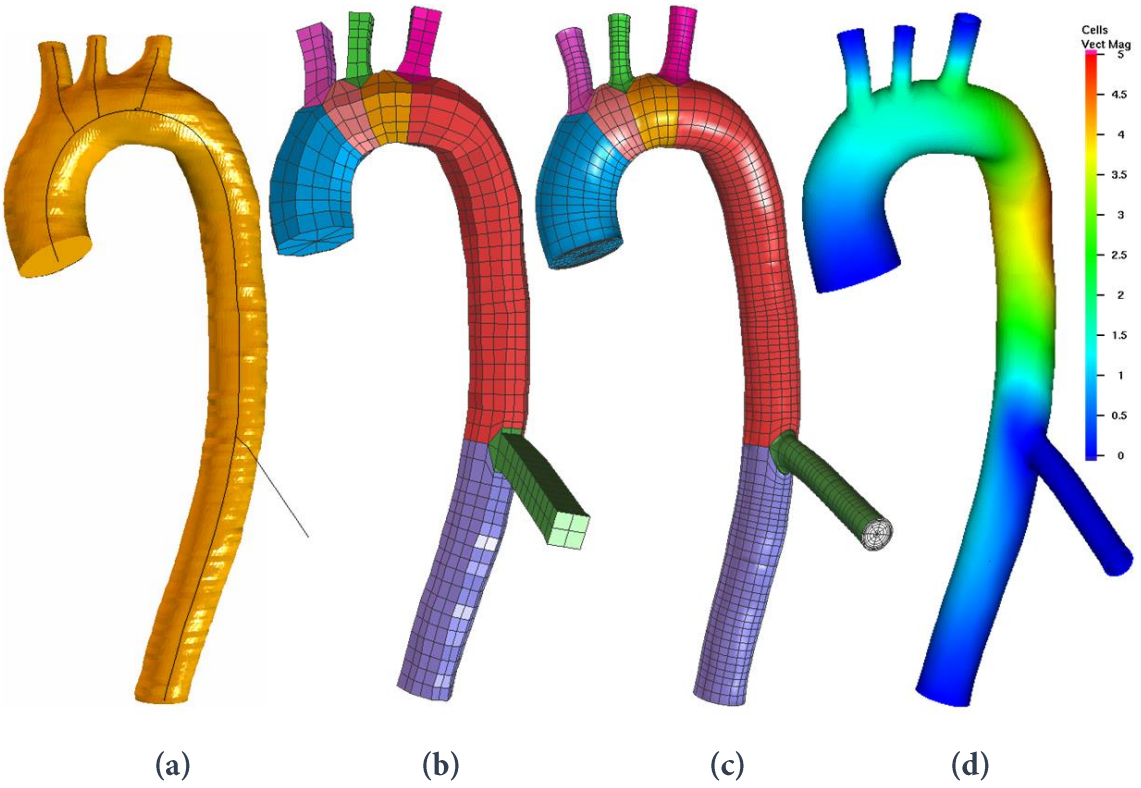


Figure 1.6. Simulation and Isogeometric Analysis of the Thoracic aorta.
(a) Surface model and path, **(b)** Control mesh
(c) Solid NURBS, **(d)** Simulation Results
(http://www.andrew.cmu.edu/user/jessicaz/publication/vascularmodel/)

2 Basic Ingredients of Isogeometric Analysis

2.1 Introduction

In this thesis, we decided to exclusively use NURBS. In this chapter we will study thoroughly how NURBS are formulated, their properties and behavior. Although there are newer technologies and other splines with better behavior under certain circumstances, the majority of the CAD industry, from software companies to amateur users, still uses NURBS shape functions to model geometry as billions have been invested in the industry and both extensive knowledge and advanced software has been developed on the subject.

In Isogeometric analysis we generally use isoparametric elements. The term isoparametric is explained by the fact that we use the shape functions which describe the solution field to describe the geometry of the structure. In isogeometric analysis the isoparametric concept is reversed. We use the shape functions which describe the geometry to approximate the solution field. We should highlight, however, that in Isogeometric analysis the shape functions are used to describe the exact CAD geometry, unlike Finite Elements where the shape functions are used to only approximate the CAD geometry.

Directly by the CAD mesh, since the same Shape functions are used, we have the mesh of control points and knots. The Control points are the coefficients which multiplied with the values of the shape functions at a certain point define the geometry. The Knot mesh provides a suitable discretization of the domain for numerical integration and defines the boundaries of the support of the Shape functions.

2.2 Index, Parameter and Physical Space

In most Spline geometries, three different spaces play a crucial role when defining them. The Index, the Parameter and the Physical Space.

In the process of working with NURBS geometries, several projections take place. The space we are more familiar with is the Physical Space. This space is based on the Cartesian coordinate system and therein the accurate model is designed using CAD technology. When analyzing a complex structure it is easier to reduce it to a simpler shape to be able to process it. Taking advantage of the isoparametric concept we do just that, we project the complex structure in a simpler space where it has a very basic shape of a line, a rectangle or a cuboid, depending on the structure's dimensions. This is the Parameter space, equivalent to the natural system in FEA, where the shape functions are defined and numerical quadrature takes place. Lastly, the Index space plays an auxiliary role in NURBS formulation, depicting better the Shape functions' support and assisting in calculating the Control Points parametric coordinates.

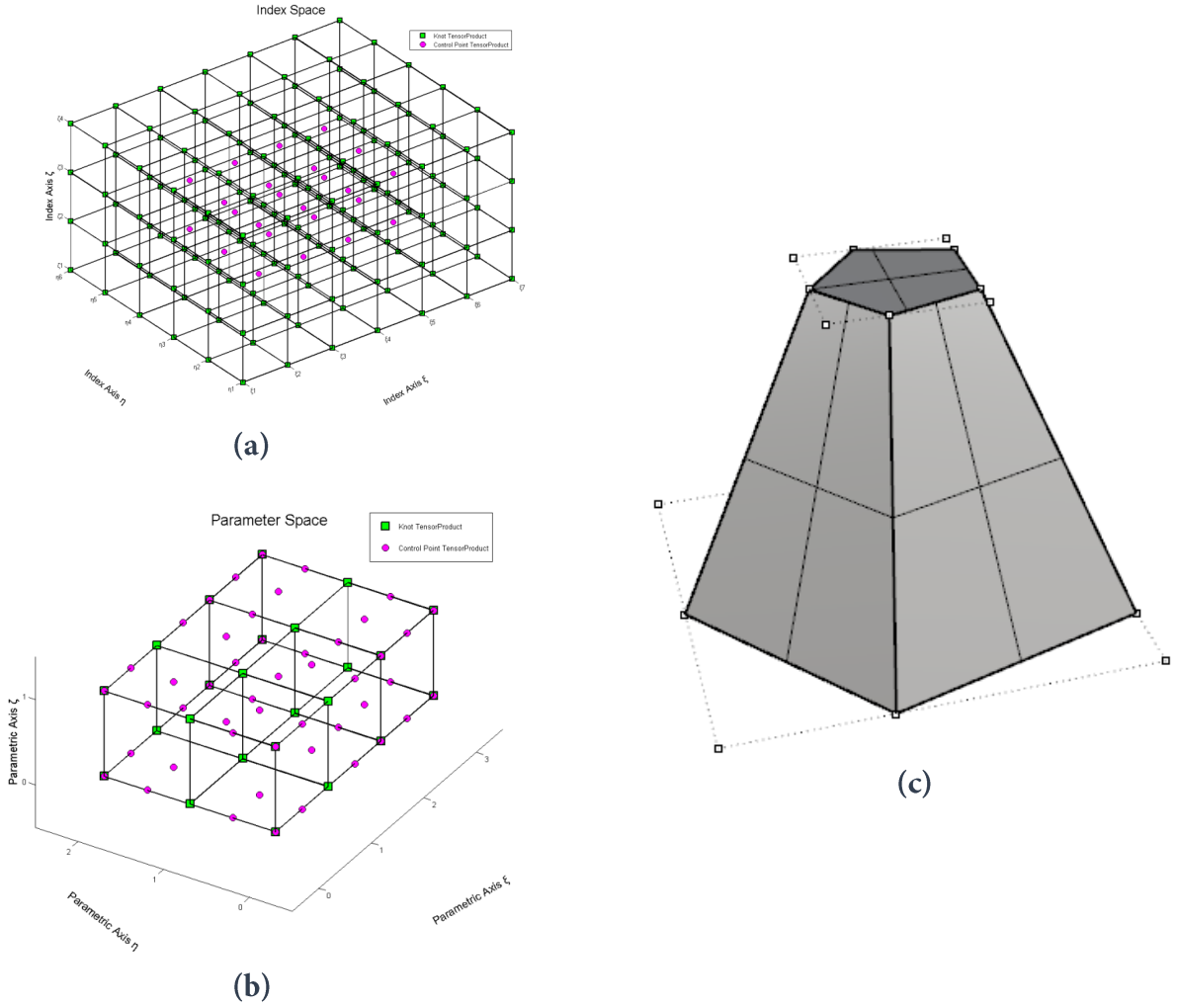


Figure 2.1. **(a)** Index, **(b)** Parameter and **(c)** Physical Space

2.2.1 Index Space

Index space is a representation of the model in respect with the Knot Values. Its purpose is to better present the sequence of Knot Values and potentially repeated Knots. That way we can clearly see the support of each basis function and which knot spans are trivial. To do that, it shows the Knot Value sequence with equal distances between the Knot Values. In the Index Space we also determine the Control Point's parametric coordinates as the middle of the support of its basis function.

Index Space is a line in 1D, a rectangle in 2D and a cuboid in 3D.

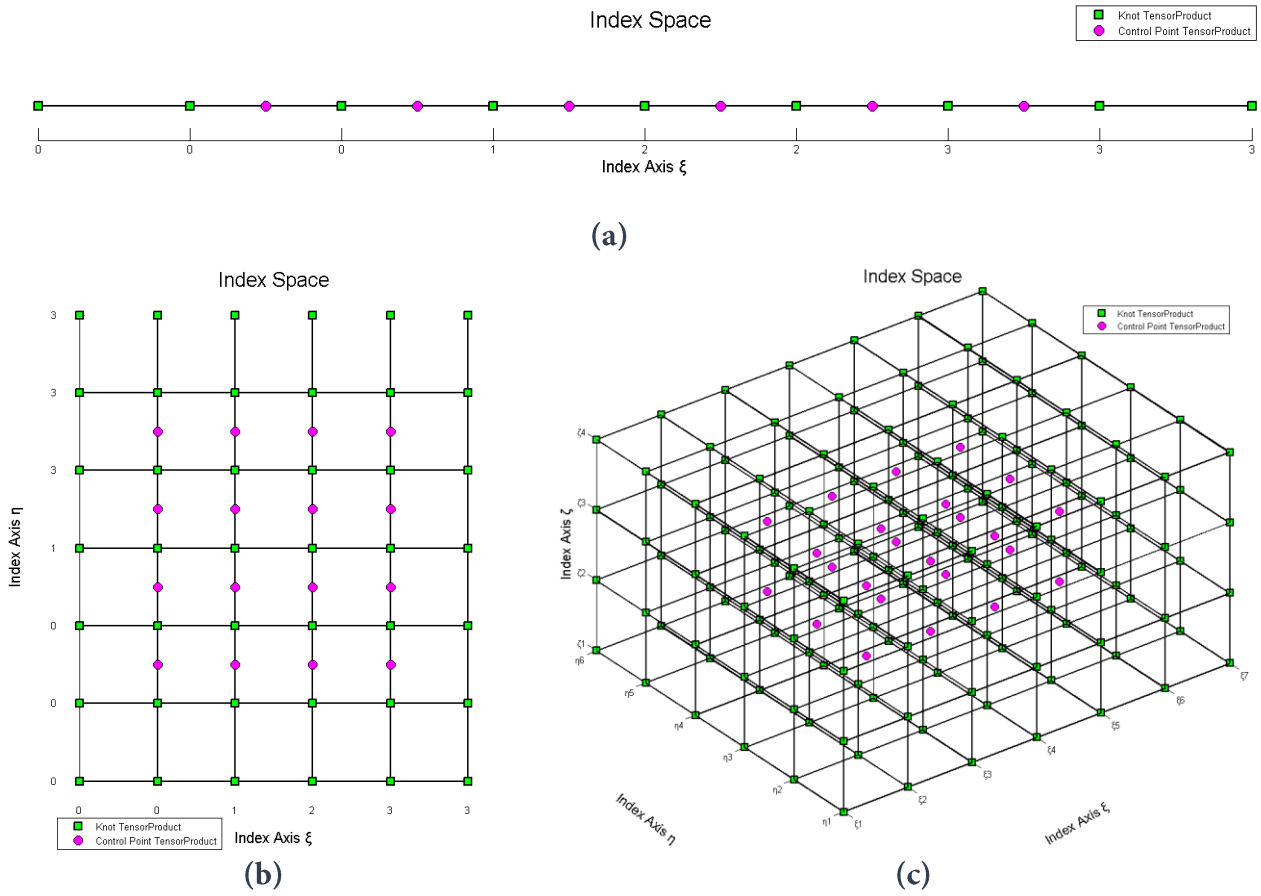


Figure 2.2. Index Space in the cases of 1D, 2D, 3D.
(a) Index Space 1D Line. Knot Value Vector Ξ = { 0 0 0 1 2 2 3 3 3 }
(b) Index Space 2D Rectangle. Knot Value Vectors Ξ = { 0 0 1 2 3 3 } , H = { 0 0 0 1 3 3 3 }
(c) Index Space 3D Cuboid. Knot Value Vectors Ξ={ ξ1 . . . ξ7}, H={ η1 . . . η6 } , Z = { ζ1 . . . ζ4 }

2.2.2 Parameter Space

Parameter space is a representation of the model in respect with Knots. Now the Knots are presented in their exact positions as their numerical contents indicate, unlike the equally distanced Knot Values in Index Space. We could see it as a "contraction" of the Index Space but only roughly, as that description would not be accurate.

Parameter Space is the analogous to Natural System in Finite Elements. The Jacobian matrix and its inverse are used for the transition from Parameter Space to Physical Space and vice versa. After the mapping of the model from Physical Space to Parameter Space where it will be a line in 1D, a rectangle in 2D or a cuboid in 3D, the Gaussian Quadrature to calculate the Stiffness Matrix takes place.

In Parameter Space we usually also plot the Basis and Shape functions as in this space it is easy to see their support and observe their properties with clarity.

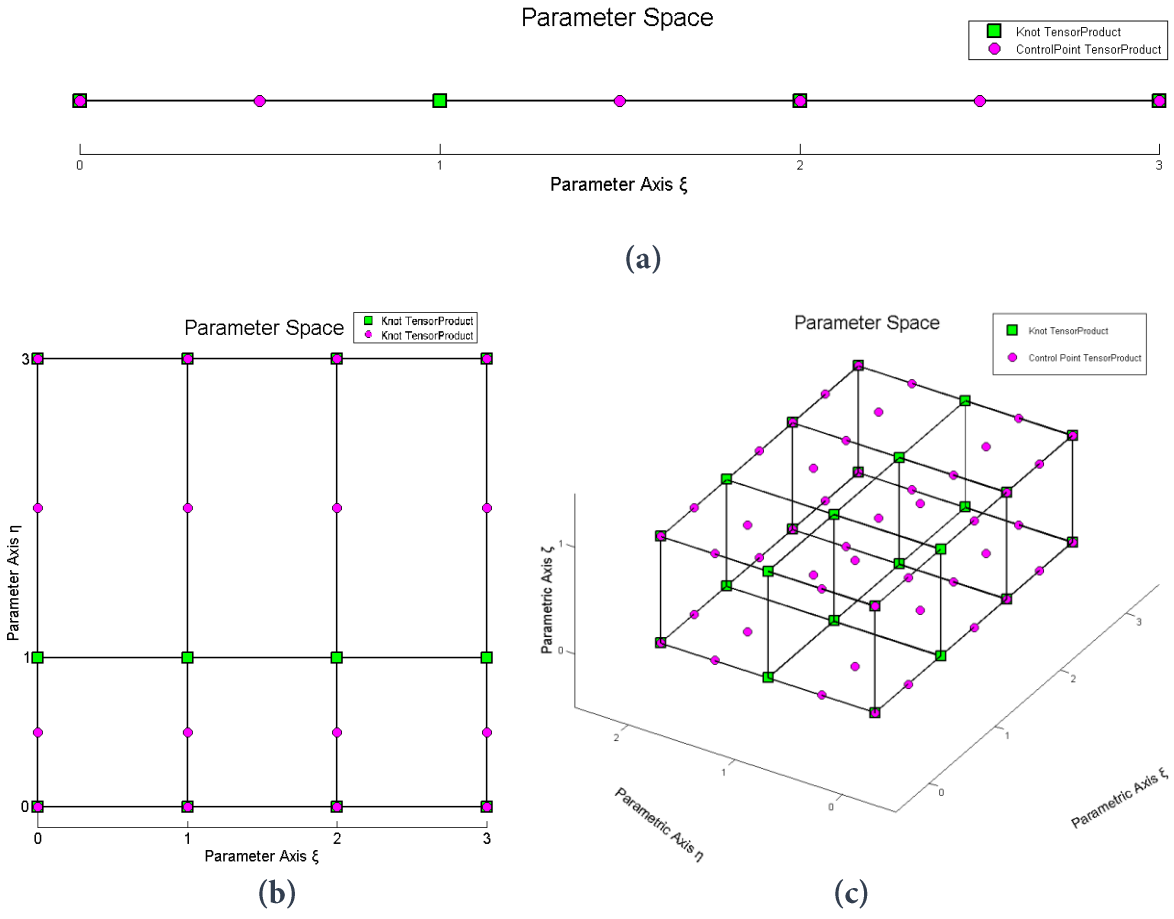


Figure 2.3. Parameter Space in the cases of 1D, 2D, 3D.
(a) Index Space 1D Line. Knot Value Vector Ξ = { 0 0 0 1 2 2 3 3 3 }
(b) Index Space 2D Rectangle. Knot Value Vectors Ξ = { 0 0 1 2 3 3 } , H = { 0 0 0 1 3 3 3 }
(c) Index Space 3D Cuboid. Knot Value Vectors Ξ={ 0 0 0 1 2 2 3 3 3}, H={ 0 0 0 1 2 2 2 } , Z = { 0 0 1 1 }

2.2.3 Physical Space

Physical Space is the Cartesian Space where the real model is designed. Here we can see all kinds of shapes, random curves, surfaces and solids.

In contrast to conventional FEA with C^0 continuity where the nodes lay upon the model, control points in the Physical Space are not restricted to the inside or edge of the model. We can intuitively think of the control points as weights that pull the model toward them.

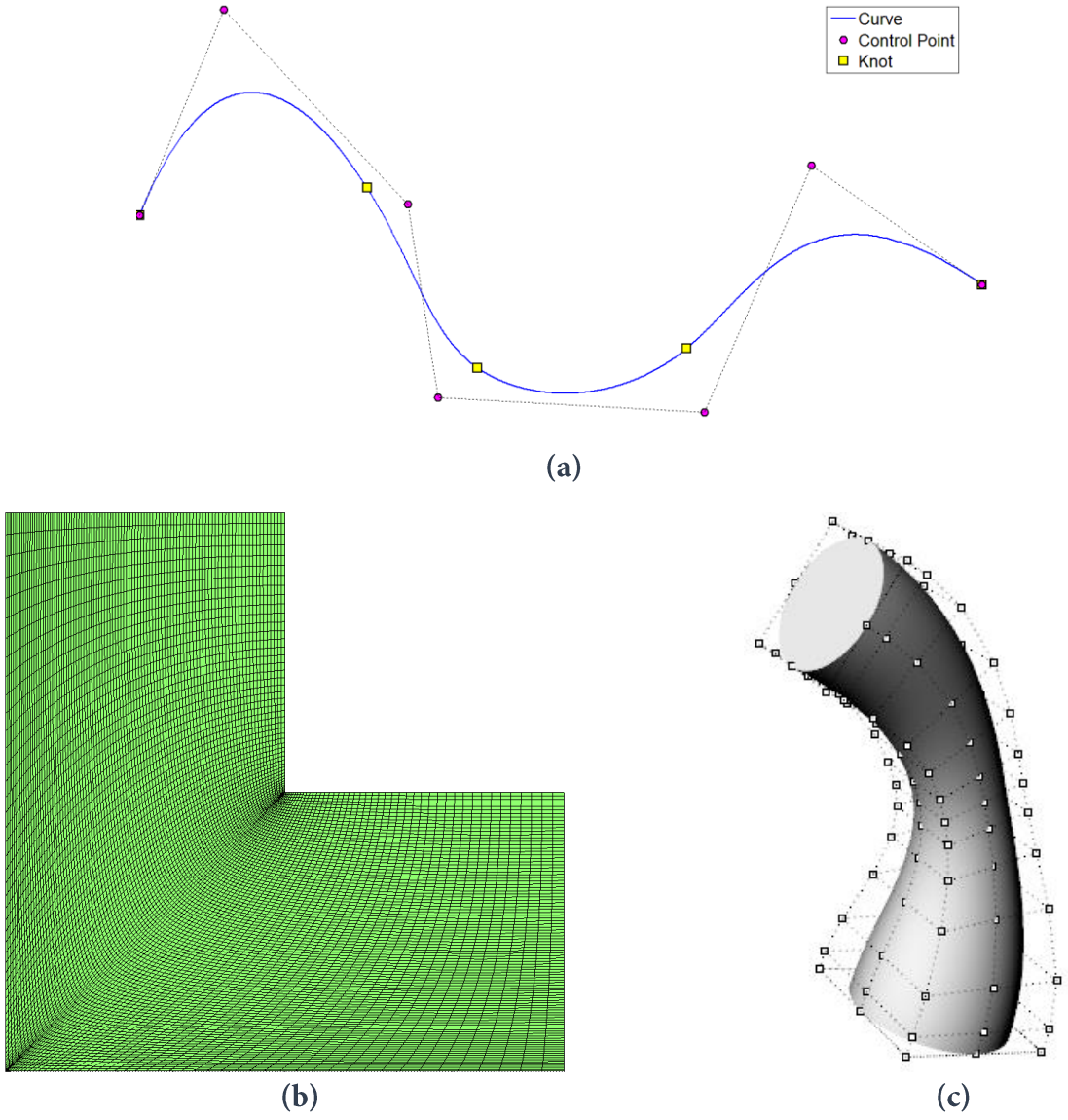


Figure 2.4. Physical Space in cases of 1D, 2D, 3D.
(a) Curve **(b)** L-Shape **(c)** Solid Pipe

2.3 B-Spline Geometries

2.3.1 Introduction

B-Splines is the foundation on which the NURBS are built and thus we will have to study them in depth. In fact, NURBS are a generalization of B-Splines and share most of their properties. A B-Spline function is a Spline function that has minimal support with respect to a given degree, smoothness and domain partition. B-Splines are piecewise polynomials and therefore are characterized by their degree p and their support.

2.3.2 Knot Vector

The Knot Vector, in one dimension is a set of non-decreasing coordinates in the parameter space. The first and last coordinate define the model's boundaries in its projection in the index and parameter space.

In bibliography the general term is "Knot Vector" whether we are referring to the non-decreasing set of coordinates or the unique values of those coordinates. To be unambiguous as to which of those two vectors we are referring to, we introduce the terminology:

"Knot Value vector": This is the whole set of non-decreasing coordinates.

"Knot Vector": This is the non-decreasing set of unique coordinates.

An example of a Knot Value Vector would be $\{0\ 0\ 0\ 1\ 2.5\ 3\ 4\ 4\ 5\ 6\ 6\ 6\}$

and the respective Knot Vector $\{0\ 1\ 2.5\ 3\ 4\ 5\ 6\}$

We refer to the values of the Knot Value Vector as "Knot Values" and to the values of Knot Vector as "Knots".

A knot value vector is named **uniform** when all knot values are equally distanced as in the case of $\{0\ 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\}$ and **open** when the first and last value is repeated (p+1) times as in the case of $\{0\ 0\ 0\ 1\ 2.5\ 3\ 4\ 4\ 5\ 6\ 6\ 6\}$ (for p=2). By a strict translation of the above terminology, there cannot be an open uniform Knot Vector. We may, however, occasionally throughout this Thesis use the term "open uniform" Knot Value Vector, meaning that the Knot Value Vector is open and its Knots are equally distanced.

In B-Splines and NURBS we will **generally use open, non-uniform** knot value vectors. The knot values of such a vector are (n+p+1).

$$\Xi = \left\{\xi_1, \xi_2, \cdots \xi_{n+p}, \xi_{n+p+1}\right\}$$

As we will see further on, the actual numerical content of a knot value is trivial, it is the relative distance between the knot values that is important. That said, a knot vector can be translated or scaled by any number and the resulting basis will still be the same. All the information needed to build a B-Spline basis function can be obtained from the Knot Value Vector.

2.3.3 Definition of B-Splines

Given a knot value vector in Parameter space $\Xi=\{\xi_1, \xi_2, \cdots \xi_{n+p}, \xi_{n+p+1}\}$ we have all the information needed to evaluate the B-Splines. We can determine the degree p of the B-Splines by counting how many repeated values we have at the beginning and end of the knot value vector recalling they should be (p+1). Then we get the number n of the basis functions as we know that there are n+p+1 knot values in total.

There are a number of ways to evaluate B-Splines but here we present the widely used Cox de Boor recursive formula.

Cox de Boor Recursive Formula:

First, for **degree p=0** (piecewise constant, box B-Splines)

$$N_{i,0}(\xi) = \begin{cases} 1, & \text{if } \xi_i \le \xi < \xi_{i+1} \\ 0, & \text{otherwise} \end{cases}$$

We note that the piecewise constant function does not include the right edge. This ensures partition of unity as the next basis function begins at that edge. The last function of degree zero however includes both left and right edge in order to ensure partition of unity even in the right edge of the Knot Vector.

$$N_{n+p,0}(\xi) = \begin{cases} 1, & \text{if } \xi_i \le \xi \le \xi_{i+1} \\ 0, & \text{otherwise} \end{cases}$$

Afterwards, for **degree p>0**

$$N_{i,p}(\xi)=\frac{\xi-\xi_i}{\xi_{i+p}-\xi_i} N_{i,p-1}(\xi)+\frac{\xi_{i+p+1}-\xi}{\xi_{i+p+1}-\xi_{i+1}} N_{i+1,p-1}(\xi)$$

with the assumption that $\frac{0}{0} \doteq 0$

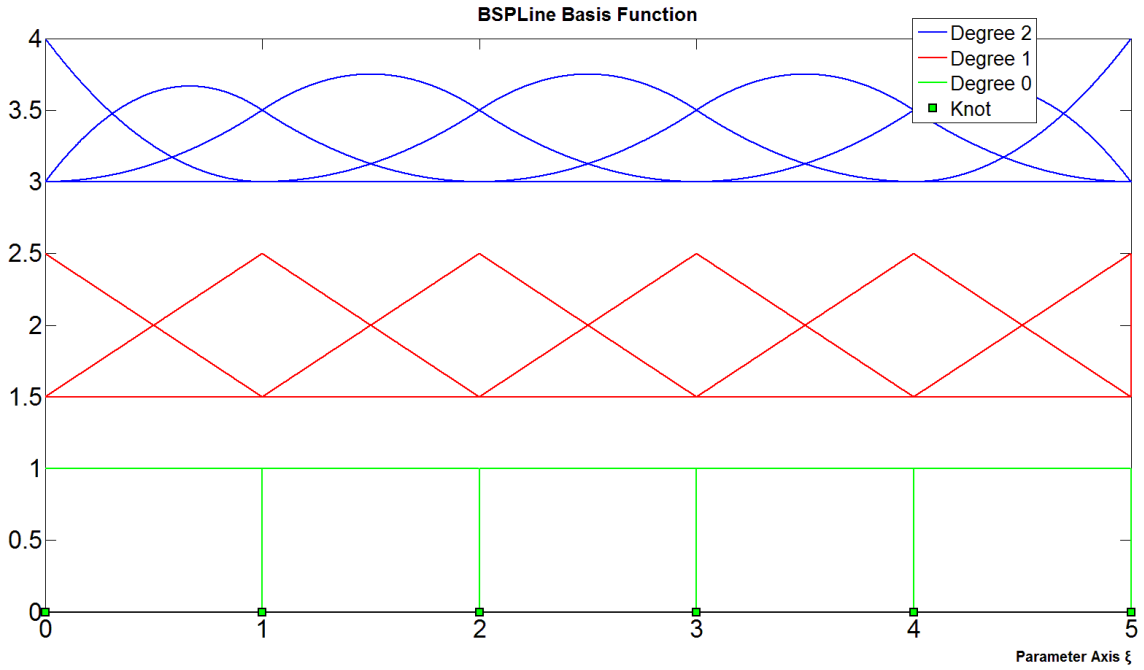


Figure 2.5. B-Spline Basis functions
Degree zero (green), one (red) and two (blue)
Knot Value Vector Ξ={0 0 0 1 2 3 4 5 5 5}

2.3.4 Control Points

The vector valued coefficients of the basis functions are referred to as Control points. Although it is not a complete analogous, Control points are close to the notion of nodes in Finite Elements in that they are the coefficients multiplied with the basis functions to define geometry. They exist in all three spaces: Index, Parameter and Physical Space. Their parametric coordinates are defined as the center of the support of the basis function in Index space. The i^{th} basis function of degree p, has support $[\xi_i, \xi_{i+p+1})$ as will be discussed later on in the B-Spline properties. The support contains (p+1) knot value spans and (p+2) knot values (including the right boundary value ξ_{i+p+1}).

For **even degrees**, the center of the support in the Index Space lies between two sequential knot values and therefore the control point parametric coordinate is:

$$\xi_{CP_i} = 0.5\left(\xi_{i+\frac{p}{2}} + \xi_{i+\frac{p}{2}+1}\right)$$

Thus a control point of even degree can either be on a knot or in the middle of a knot span.

For **odd degrees**, the center of the support is the knot value $\left(i+\frac{p+1}{2}\right)$. Thus the control point is always coincident with the knot:

$$\xi_{CP_i} = \xi_{i+\frac{p+1}{2}}$$

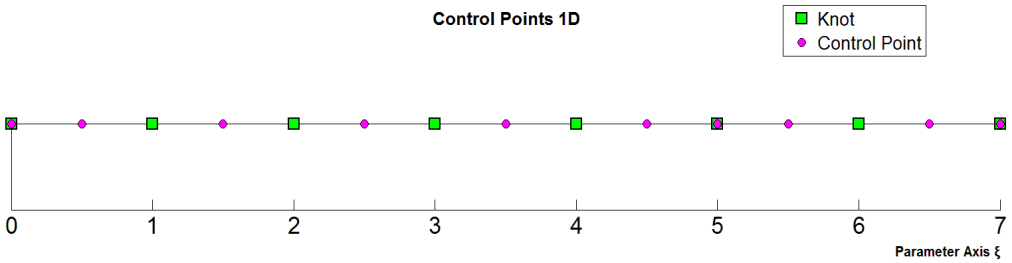


Figure 2.6. Parameter Space. Control Points and Knots for even Degree=2.
Knot Value Vector Ξ = {0 0 0 1 2 3 4 5 5 6 7 7 7}

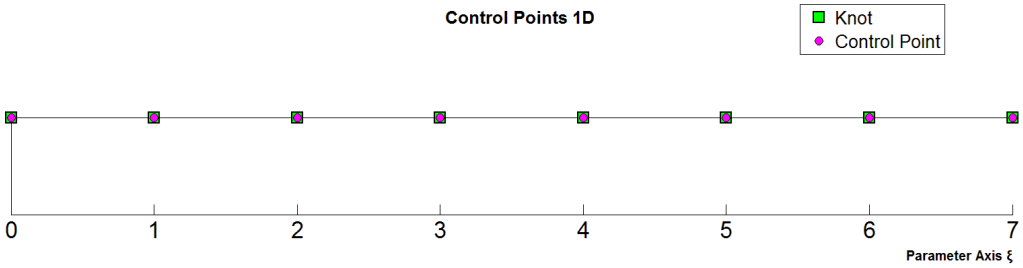


Figure 2.7. Parameter Space. Control Points and Knots for odd Degree=3.
Knot Value Vector Ξ = { 0 0 0 0 1 2 3 4 5 6 7 7 7 7 }

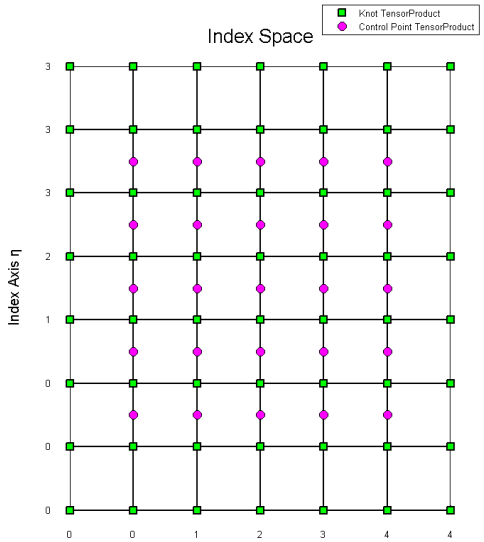


Figure 2.8. Index Space. 2D Control Points at the middle of the Basis function support.
Degree ξ = 1 (odd) , Degree η = 2 (even)

2.3.5 B-Spline Shape functions and their Full Tensor Product Nature

B-Spline basis functions along with their important properties are of full tensor product nature. That means evaluating the B-Spline Shape Function is very easy by combining B-Spline basis functions along different parametric direction.

In the 1D case, taking into account only the B-Spline basis function along the ξ parametric direction, the B-Spline Shape function is:

$$R_i^p(\xi) = N_{i,p}(\xi)$$

In the 2D case combining B-Spline basis functions on parametric directions ξ, η, the B-Spline Shape function is:

$$R_{i,j}^{p,q}(\xi,\eta) = N_{i,p}(\xi)M_{j,q}(\eta)$$

In the 3D case, combining B-Spline basis functions on parametric directions ξ, η, ζ, the B-Spline Shape function is:

$$R_{i,j,k}^{p,q,r}(\xi,\eta,\zeta) = N_{i,p}(\xi)M_{j,q}(\eta)L_{k,r}(\zeta)$$

As a result, the Control Points and Knot Vectors, used to define and better represent the B-Spline basis functions along a parametric axis are also of a full tensor product nature in the Parameter Space. In that way, we can refer to any control point or knot value with its coordinate along the corresponding parametric direction: The control point (i,j) in a 2D problem has the parametric coordinates of the i^{th} control point on axis ξ and of the j^{th} control point on axis η.

The full tensor product definition of the Shape functions allows the other properties of the B-Splines to be inherited in the Shape functions as well and be applied.

2.3.6 B-Spline Basis Function Properties

Piegl and Tiller [14], do a thorough examination of the properties of B-Spline basis functions and their proofs but we also noted here an extra property we found useful, property 3. We will review them briefly and then examine each property separately.

1. Local support property

$$N_{i,p}(\xi) = 0 \quad \forall \xi \notin \left[\xi_i, \xi_{i+p+1}\right)$$

2. In any given knot span $\left[\xi_j, \xi_{j+1}\right)$ at most (p+1) of the functions $N_{i,p}$ are non-zero and those non-zero candidates are $N_{j-p,p}, \ldots, N_{j,p}$.
3. Every function shares support with 2p other functions plus itself.
4. Non-negativity

$$N_{i,p}(\xi) \geq 0$$

5. Partition of unity.

$$\sum_{i=1}^{n} N_{i,p}(\xi) = 1$$

6. C^{p-k} Continuity on a knot of multiplicity k and infinitely differentiable in the internal of a knot span.
7. Except for the case of p=0, $N_{i,p}(\xi)$ attains exactly one maximum.
8. Linear Independence.

2.3.6.1 Local support property

The local support property means that a B-Spline Basis function $N_{i,p}$ is non-zero only in a specific interval in the Parameter Space.

$$N_{i,p}(\xi) = 0 \quad \forall \xi \notin \left[\xi_i, \xi_{i+p+1}\right)$$

The local support is derived from the recursive definition of a B-Spline.

$$N_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi)$$

The i^{th} B-Spline of degree p is a combination of the B-splines i and i+1 of degree p-1. By induction we can see that the p degree B-Spline has support (p+1) degree zero box functions and by recalling that the degree zero functions have support only one knot span, the p degree B-Splines have support (p+1) knot spans or (p+2) knot values. It is easy to prove it with induction on p. We can see it represented with clarity in Figure 2.9.

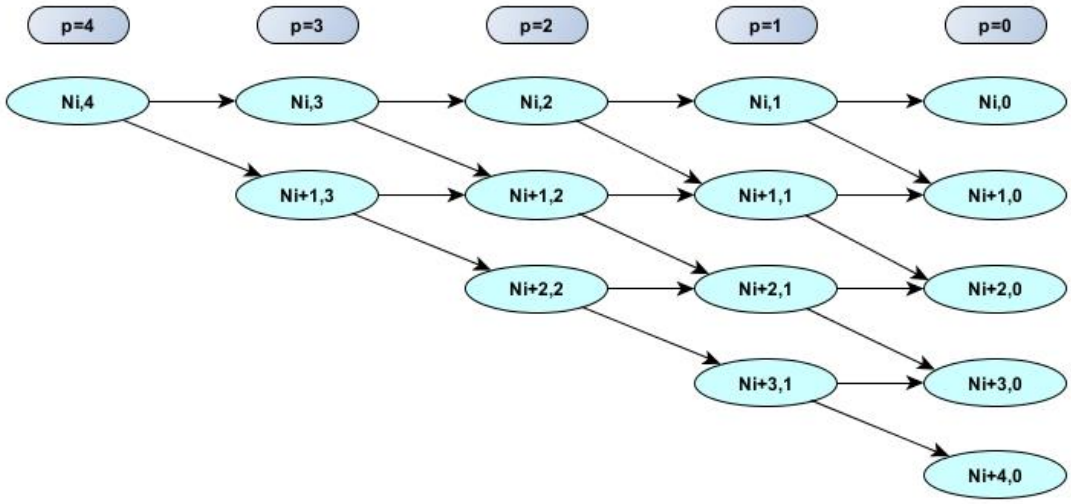


Figure 2.9. Lower degree basis functions that influence a certain basis function of degree p.

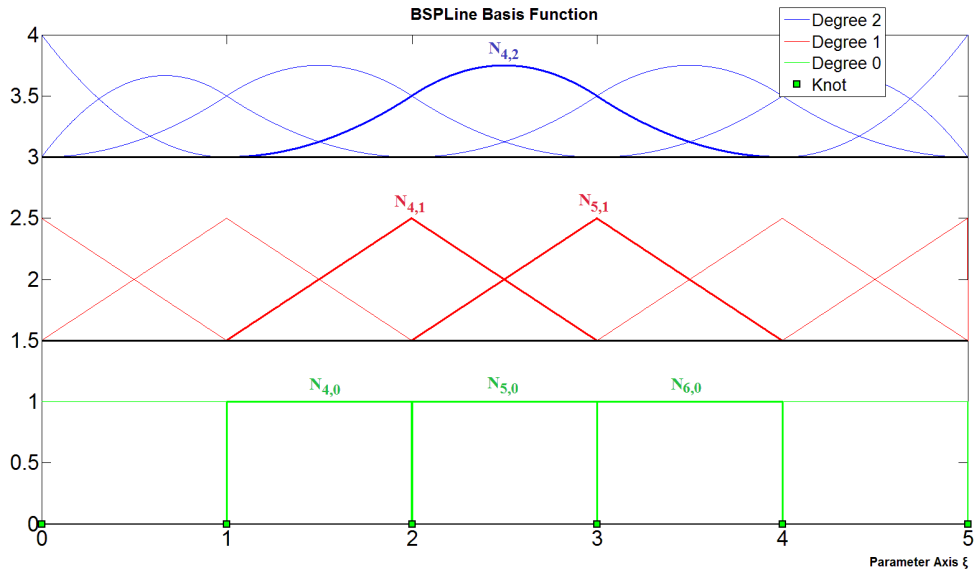


Figure 2.10. Lower degree basis functions that influence the quadratic basis function $N_{4,2}$. Knot Value Vector $\Xi = \{0\ 0\ 0\ 1\ 2\ 3\ 4\ 5\ 5\ 5\}$

In Figure 2.10, we see that, due to the recursive character of the B-Splines, the i^{th} B-Spline of degree p=4 depends on p+1=5 box B-Splines of degree zero who have support only one knot value span. $N_{4,1}(\xi)$ is defined by $N_{4,0}(\xi)$ and $N_{5,0}(\xi)$ which have support the interval $[1,2)\cup[2,3)=[1,3)$ $[1,2)\cup[2,3)=[1,3)$ and $N_{5,1}(\xi)$ by $N_{5,0}(\xi)$ and $N_{6,0}(\xi)$ which have support the interval $[2,3)\cup[3,4)=[2,4)$. The basis function $N_{4,2}(\xi)$ is defined by the functions $N_{4,1}(\xi)$ and $N_{5,1}(\xi)$ which hold together the support $[1,3)\cup[2,4)=[1,4)$. Finally the basis function $N_{4,2}(\xi)$ has support of (p+1)=3 knot spans equal to the interval $[1,4)$.

In 2D and 3D,

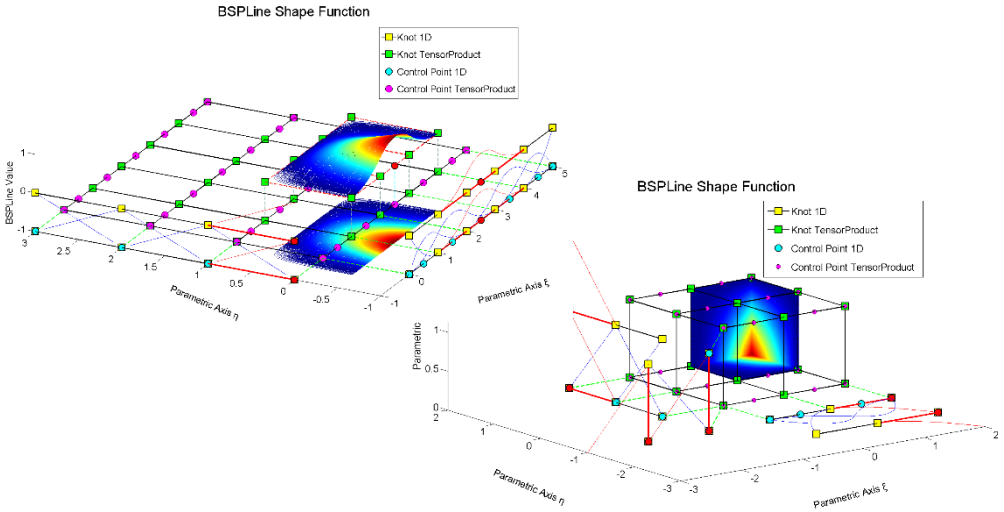


Figure 2.11. Local support of Shape Functions 2D and 3D.

2.3.6.2 In any given knot span $[\xi_j,\xi_{j+1})$ at most (p+1) of the functions $N_{i,p}$ are nonzero and those non-zero candidates are $N_{j-p,p},\ldots,N_{j,p}$

In the above statement we declared "at most" (p+1) basis functions will be nonzero. The "at most" augmentation takes into consideration the case when $\xi_j=\xi_{j+1}$, when there is a knot with multiplicity>1 and the interval $[\xi_j,\xi_{j+1})$ does not exist. The functions $N_{i,p}$ who are not zero in the interval $[\xi_j,\xi_{j+1})$ are those who have the interval $[\xi_j,\xi_{j+1})$ in their support. When the definition of $N_{i,p}$ is traced recursively back to the degree zero box B-Spline Basis functions, to have the knot span $[\xi_j,\xi_{j+1})$ in its support, the $N_{j,0}$ must be among those supporting degree zero basis functions. Otherwise the $N_{i,p}$ B-Spline will surely be zero in the $[\xi_j,\xi_{j+1})$ knot span.

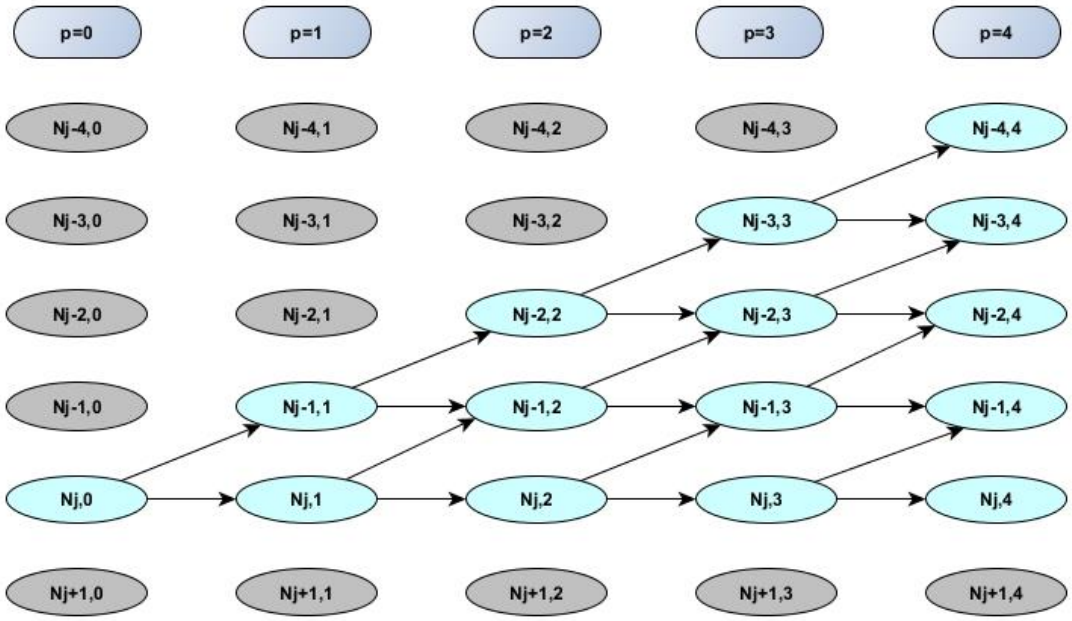


Figure 2.12. Influence of a single B-Spline box function to the higher degree B-Spline basis functions.

In Figure 2.12 we can see that each box B-Spline affects only two linear basis functions, only three quadratic basis function, only four degree 3 basis functions and generalizing the statement (p+1) of degree p basis functions. More precisely, we can see in the above figure that those basis functions who will be potentially non-zero are only the $N_{j-p},\ldots N_j$.

In Figure 2.13 we can see which B-Spline basis functions of each degree 0, 1 and 2, have the interval [2,3) in their support.

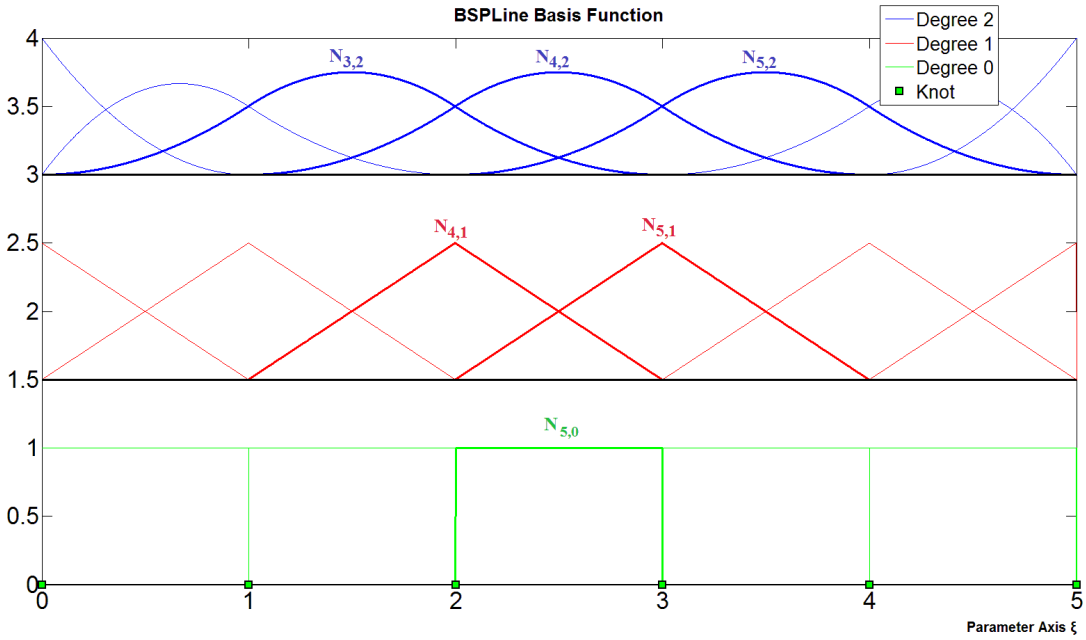


Figure 2.13. Influence of a single B-Spline box function to the higher degree linear and quadratic basis functions.

2.3.6.3 Every basis function shares support with maximum 2p others

This derives directly from the local support property: $N_{i,p}(\xi)=0 \quad \forall \xi \notin \left[\xi_i, \xi_{i+p+1}\right)$. Then the function $N_{i-p,p}(\xi)=0$ has support $\left[\xi_{i-p}, \xi_{i+1}\right)$ and the function $N_{i+p,p}(\xi)=0$ has support $\left[\xi_{i+p}, \xi_{i+2p+1}\right)$. They are the first and last function that share support with $N_{i,p}(\xi)$ and thus the set of functions that share support with $N_{i,p}(\xi)$ are the $N_{j,p}(\xi)$ with

j=i-p, …, i-1, i+1, …, i+2p+1 which are 2p in number.

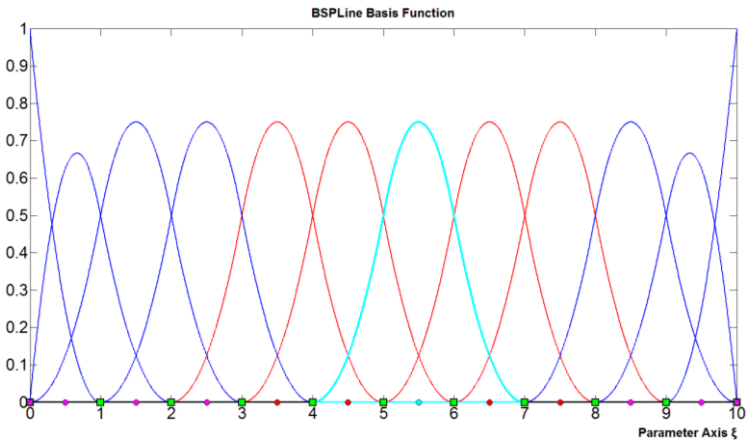


Figure 2.14. Every B-Spline basis function shares support with 2p other.
Knot Value Vector Ξ = { 0 0 0 1 2 3 4 5 6 7 8 9 10 10 10}.
2p=2*2=4 other functions

Instead, in the case of trivial knot spans, when knot values have multiplicity greater than 1, we may have functions who share support with less than 2p others. If the trivial knot span is internal to the support of the B-Spline we are inspecting, then the B-Spline shares support with exactly 2p others. We can see that in Figure 2.15.

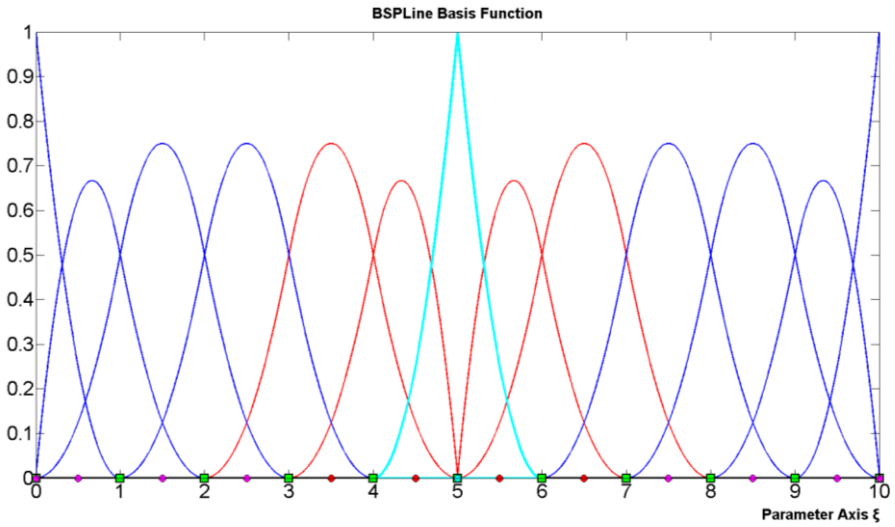


Figure 2.15. Every B-Spline basis function shares support with 2p other.
Knot Value Vector Ξ = { 0 0 0 1 2 3 4 5 5 6 7 8 9 10 10 10}. Knot 5 multiplicity=2.

In the case though the trivial knot span is at the edge of the B-Spline's support then it shares support with less than 2p others. We can see such examples in the following Figures 2.16, 2.17, 2.18.

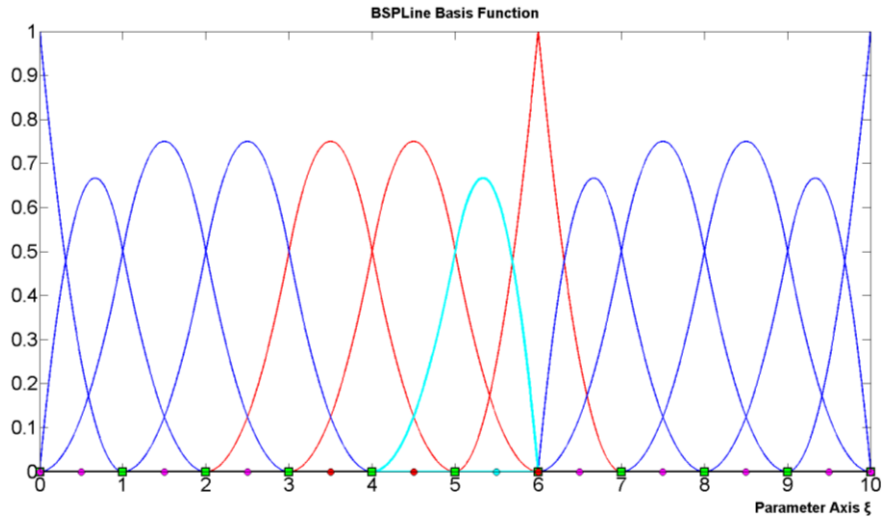


Figure 2.16. Every B-Spline basis function does not share support with 2p other.
Knot Value Vector Ξ = { 0 0 0 1 2 3 4 5 5 6 7 8 9 10 10 10}. Knot 6 multiplicity=2.

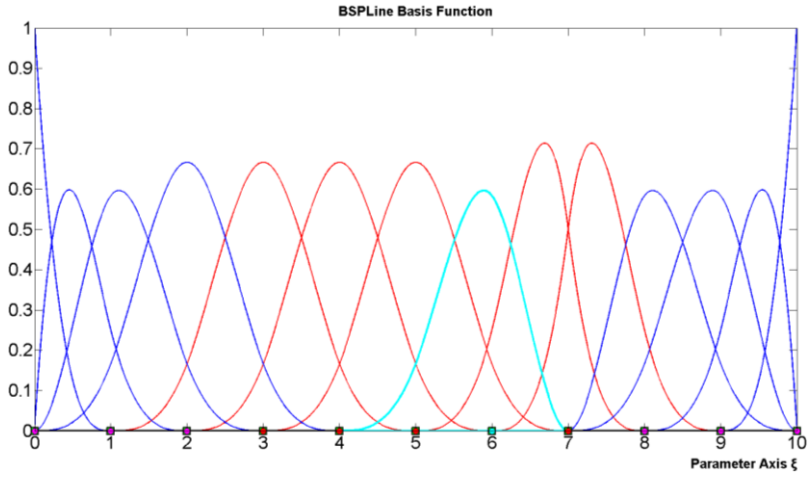
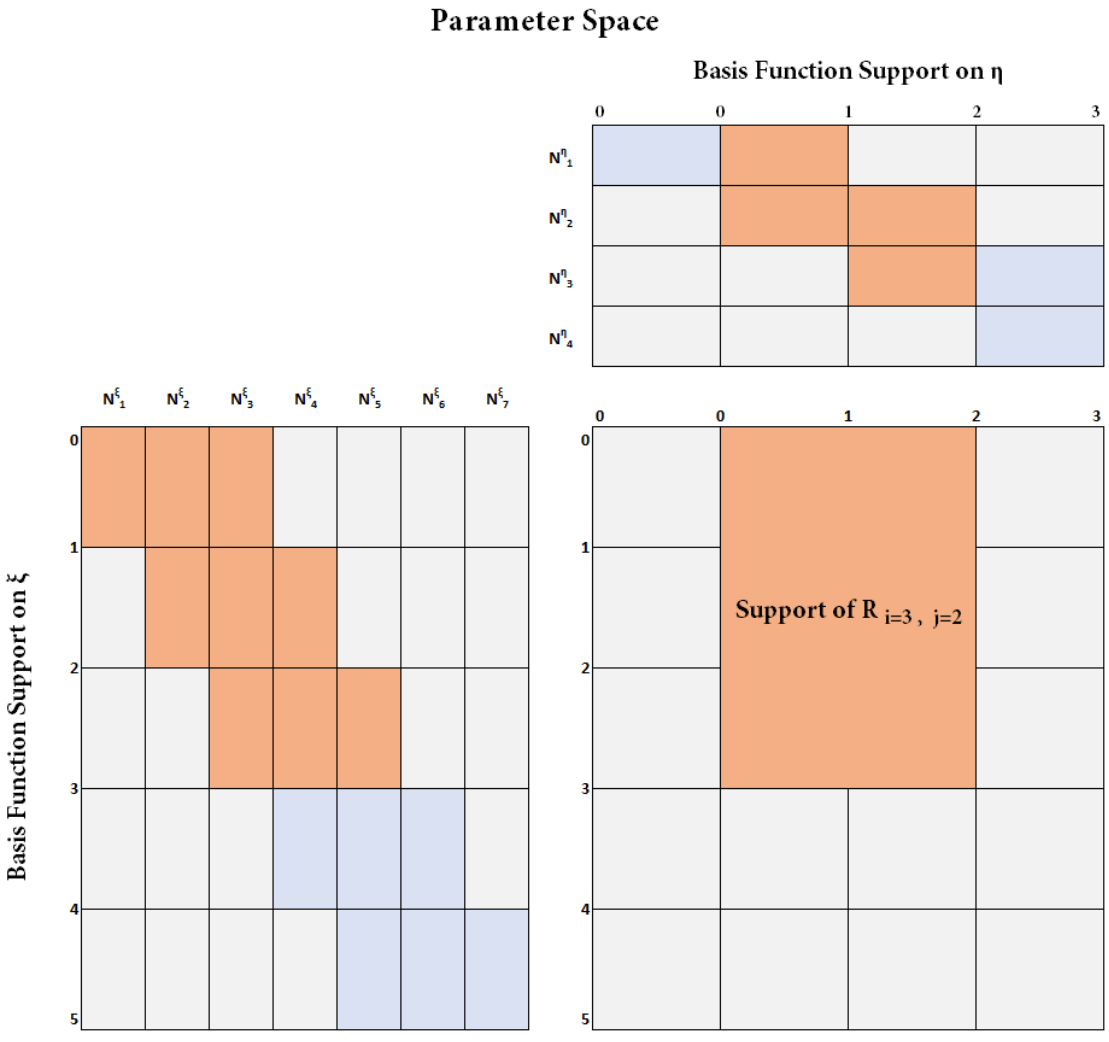


Figure 2.17. Every B-Spline basis function does not share support with 2p other. Knot Value Vector Ξ = { 0 0 0 0 1 2 3 4 5 6 7 7 8 9 10 10 10 10 }. Knot 6 multiplicity=2.



(a)

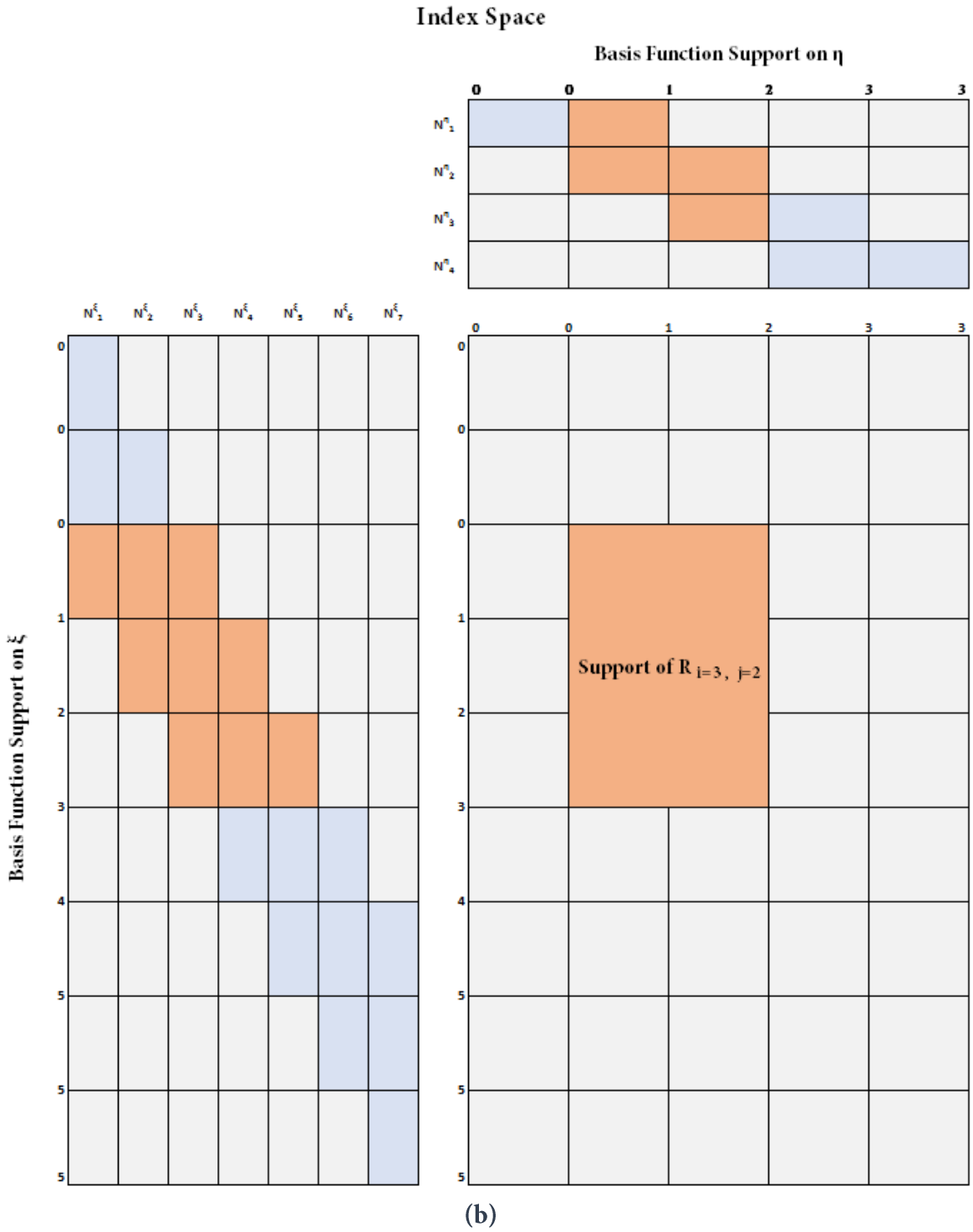


Figure 2.18. Maximum Shared Support of a 2D B-Spline in **(a)** Parameter Space **(b)** Index Space Knot Value Vectors Ξ = { 0 0 0 1 2 3 4 5 5 5 } and H = { 0 0 1 2 3 3 }

2.3.6.4 Non-negativity

$N_{i,p}(\xi) \geq 0 \ \forall i,p,\xi \in \Xi$ where Ξ is the Knot Value Vector. This is proven by induction on p.

It is clearly true for p=0 and we assume it is true for p-1.

By definition $N_{i,p}(\xi)=\frac{\xi-\xi_i}{\xi_{i+p}-\xi_i} N_{i,p-1}(\xi)+\frac{\xi_{i+p+1}-\xi}{\xi_{i+p+1}-\xi_{i+1}} N_{i+1,p-1}(\xi)$.

By the property of local support, either $\xi \notin \left[\xi_i, \xi_{i+p}\right)$ and $N_{i,p-1}(\xi)=0$ or $\xi \in \left[\xi_i, \xi_{i+p}\right)$ in which case $\xi-\xi_i \geq 0$ and with the assumption made that $N_{i,p-1}(\xi) \geq 0$, the first term $\frac{\xi-\xi_i}{\xi_{i+p}-\xi_i} N_{i,p-1}(\xi) \geq 0$. The same is true for the second term, $\frac{\xi_{i+p+1}-\xi}{\xi_{i+p+1}-\xi_{i+1}} N_{i+1,p-1}(\xi) \geq 0$. Thus $N_{i,p}(\xi) \geq 0$ and is valid for any i, p, ξ.

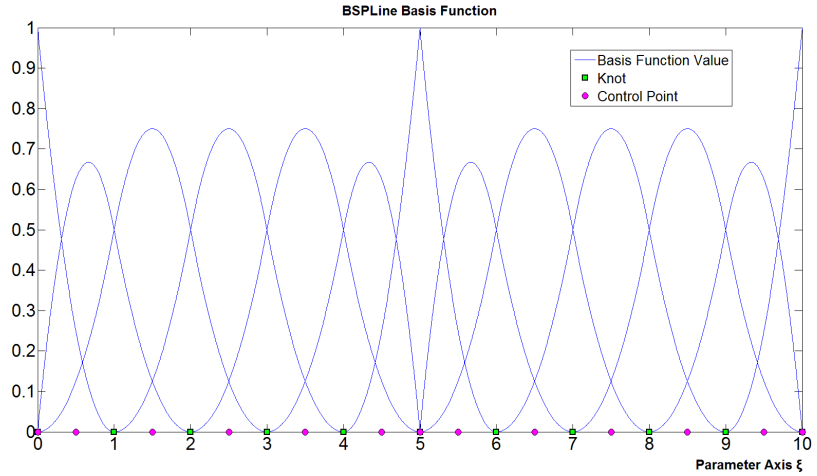


Figure 2.19. Non Negativity Property of B-Spline Basis Functions.
Knot Value Vector Ξ = { 0 0 0 1 2 3 4 5 5 6 7 8 9 10 10 10 }

2.3.6.5 Partition of Unity

For any ξ in the knot vector Ξ, the sum of all basis function at that point is one.

$$\sum_{i=1}^{n} N_{i,p}(\xi) = 1$$
$$\forall \xi \in \Xi = \{\xi_1, \ldots, \xi_{n+p+1}\}$$

Partition of unity is essential in any set of basis function in any finite element scheme. The reason is simple. In isogeometric analysis as in the general case of Finite Elements, the displacement at a point C(ξ,η,ζ)=(x,y,z) in the solid model, will be approximated by $\sum_{i=1}^{n} N_{i,p}(\xi)\, u_i$ where u_i are the displacements on the nodes or in the case of isogeometric analysis the displacements on the control points. If $u_i = \bar{u}$ for all the control points, we have

rigid body displacements (property Affine Invariance of B-Spline Curves in 2.3.9) and if those in conjunction with the laws of the phenomenon we study cause no strain, it is clear that $\bar{u}\sum_{i=1}^{n} N_{i,p}(\xi) = \text{constant}$. In our case of isoparametric elements, where the same functions are used to describe geometry, it would be necessary that all points are translated by $\bar{u}$, thus $\bar{u}\sum_{i=1}^{n} N_{i,p}(\xi) = \bar{u} \Rightarrow \sum_{i=1}^{n} N_{i,p}(\xi) = 1$.

Proof. We will prove it in an arbitrary knot span.

$\forall \xi \in \left[\xi_i, \xi_{i+1}\right)$,

$$\sum_{j=1}^{n} N_{j,p}(\xi) = \sum_{j=i-p}^{i} N_{j,p}(\xi) = \sum_{j=i-p}^{i} \left(\frac{\xi - \xi_j}{\xi_{j+p} - \xi_j} N_{j,p-1}(\xi) + \frac{\xi_{j+p+1} - \xi}{\xi_{j+p+1} - \xi_{j+1}} N_{j+1,p-1}(\xi) \right)$$

$$= \sum_{j=i-p}^{i} \frac{\xi - \xi_j}{\xi_{j+p} - \xi_j} N_{j,p-1}(\xi) + \sum_{j=i-p}^{i} \frac{\xi_{j+p+1} - \xi}{\xi_{j+p+1} - \xi_{j+1}} N_{j+1,p-1}(\xi)$$

Changing variable in the second sum from (i-p) to (i-p+1) and considering that $N_{i-p,p-1} = N_{i+1,p-1} = 0$ for $\xi \in \left[\xi_i, \xi_{i+1}\right)$:

$$\sum_{j=1}^{n} N_{j,p}(\xi) = \sum_{j=i-p+1}^{i} \left(\frac{\xi - \xi_j}{\xi_{j+p} - \xi_j} + \frac{\xi_{j+p} - \xi}{\xi_{j+p} - \xi_j} \right) N_{j,p-1}(\xi) = \sum_{j=i-p+1}^{i} N_{j,p-1}(\xi)$$

Applying the same procedure recursively:

$$\sum_{j=1}^{n} N_{j,p}(\xi) = \sum_{j=i-p+1}^{i} N_{j,p-1}(\xi) = \sum_{j=i-p+2}^{i} N_{j,p-2}(\xi) = \ldots = \sum_{j=i}^{i} N_{j,0}(\xi) = 1$$

Of course, it is also valid and easily proved for 2D and 3D cases by having in mind that it also applies for every separate parametric direction ξ, η, ζ.

$$2D: \quad \sum_{i=1}^{n}\sum_{j}^{m} N_{i,p}(\xi) M_{j,q}(\eta) = 1$$

$$3D: \quad \sum_{i=1}^{n}\sum_{j}^{m}\sum_{k=1}^{l} N_{i,p}(\xi) M_{j,q}(\eta) L_{k,r}(\zeta) = 1$$

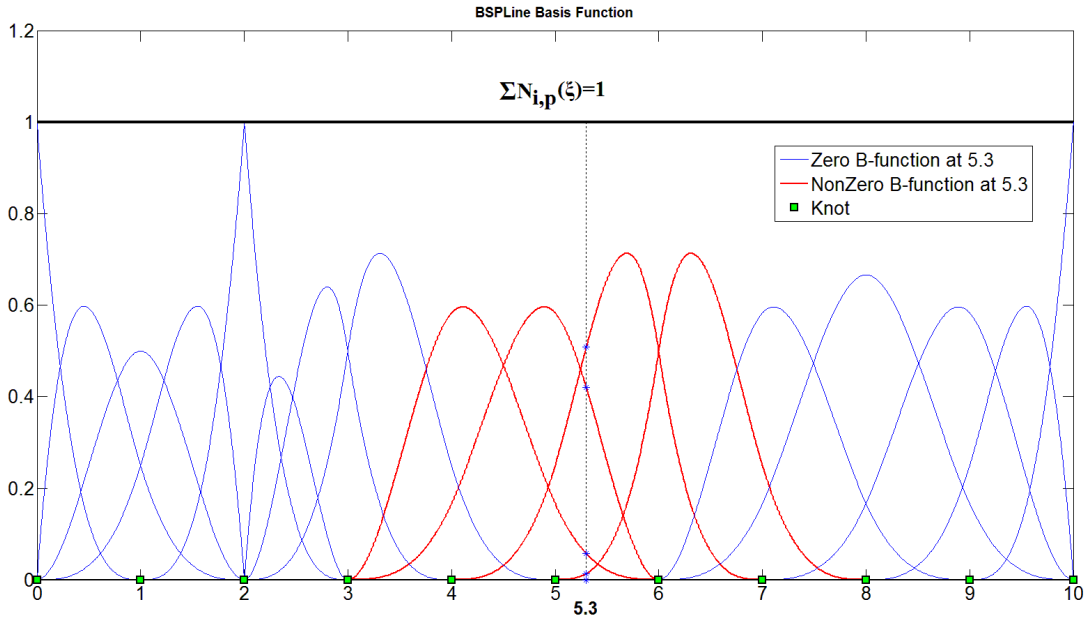


Figure 2.20. B-Spline Basis functions for Knot Value Vector
Ξ = {0 0 0 0 1 2 2 2 3 3 4 5 6 6 7 8 9 10 10 10 10}

We lay here a demonstration. We check the sum of the basis functions at ξ=5.3.

$$N_{i,p}(5.3)=0,\ i=1,..,8$$
$$N_{9,p}(5.3)=0.05717$$
$$N_{10,p}(5.3)=0.42058$$
$$N_{11,p}(5.3)=0.50875$$
$$N_{12,p}(5.3)=0.01350$$
$$N_{i,p}(5.3)=0,\ i=13,...,17$$
$$\sum_{i=1}^{17} N_{i,p}(5.3)=1$$

2.3.6.6 C^{p-k} Continuity on a knot of multiplicity k and infinitely differentiable in the internal of a knot span

All derivatives of $N_{i,p}(\xi)$ exist in the interior of a knot span where it is a continuous polynomial with standard formula and consequently indefinitely differentiable. At the knots, which are the elements' boundaries, the polynomial changes formula and the functions $N_{i,p}(\xi)$ are only (p-k) times continuous where k is the multiplicity of the knot. Hence, increasing the degree increases the continuity and increasing the multiplicity of a knot decreases the continuity along that element boundary. By definition, the knot vector at the start and end has (p+1) multiplicity which stands for continuity $C^{p-k}=C^{p-(p+1)}=C^{-1}$ meaning that the functions are not even continuous there.

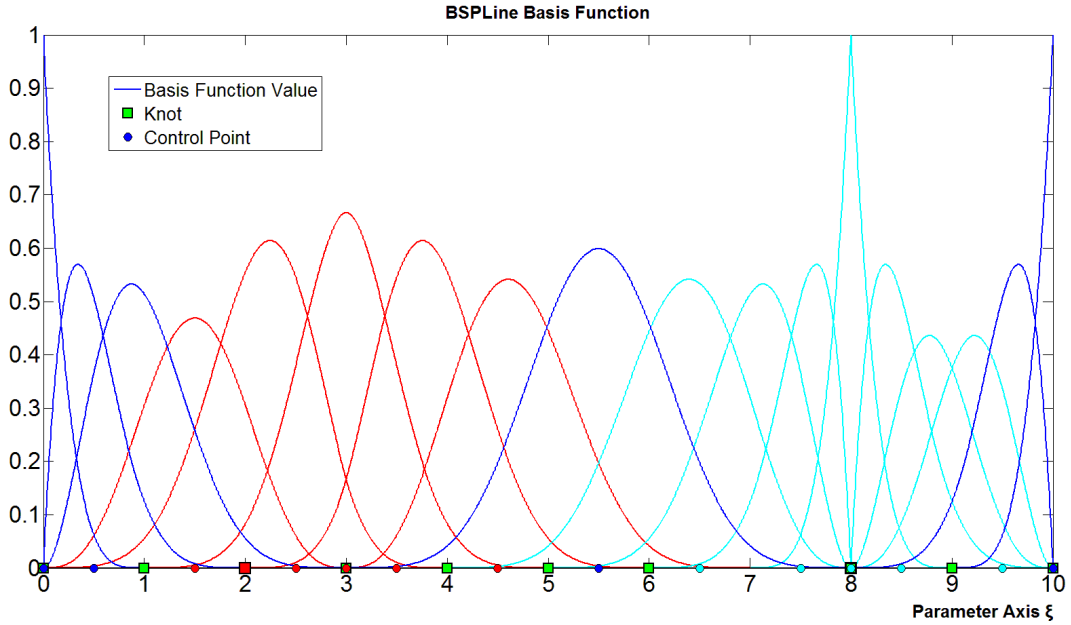


Figure 2.21. B-Spline Basis functions on a Knot Value Vector with different internal knot multiplicities. Knot Value Vector Ξ = { 0 0 0 0 0 1 2 3 3 4 5 6 7 8 8 8 8 9 10 10 10 10 10}

In the above Figure 2.21 we have a standard knot value vector of degree 4:

$$\Xi = \{\ 0\ 0\ 0\ 0\ 0\ 1\ 2\ 3\ 3\ 4\ 5\ 6\ 7\ 8\ 8\ 8\ 8\ 9\ 10\ 10\ 10\ 10\ 10\ \}$$

We notice the knots 3 and 8 have multiplicity 2 and 4 respectively. That means that while at the other knots with multiplicity 1 the continuity is $C^{4-1} = C^3$, on knot 3 we have continuity $C^{4-2} = C^2$ and on knot 8 $C^{4-4} = C^0$. Indeed we can see that in knot 8, the function has a pointy summit and thus cannot be differentiated but is still continuous.

We painted blue the basis functions who were not affected by the multiplicity of the knots 3,8, red those who were affected by the multiplicity 2 of knot 3 and cyan those affected by the multiplicity=4 of the knot 8. To match the above convention, the same colors were also applied to the corresponding control points and the knots 3 and 8.

2.3.6.7 $N_{i,p}(\xi)$ attains exactly one maximum, except for p=0

Each B-Spline attains exactly one maximum and if it is fully developed, meaning no trivial knot spans exist in its support, then it is at the center of its support. If the $N_{i,p}(\xi)$ basis function is fully developed then its maximum is at the center of its support which is the parametric coordinate of its corresponding Control Point.

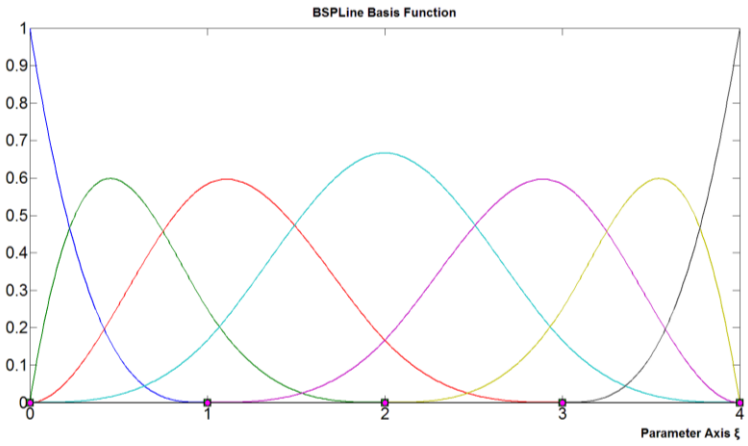


Figure 2.22. Each B-Spline attains exactly one maximum.
Knot Value Vector Ξ = { 0 0 0 0 1 2 3 4 4 4 4 }

2.3.6.8 Linear independence

A set of n functions: $f_1(\xi)\quad f_1(\xi)\quad \cdots\quad f_n(\xi)$ are called linear independent when for

$$\sum_{1}^{n} c_i f_i(\xi) = c_1 f_1(\xi) + c_2 f_2(\xi) + \cdots + c_n f_n(\xi), \text{ for all } \xi \text{ in an interval } \Xi$$

it entails that $c_1 = c_2 = \cdots = c_n = 0$.

It is proved that no B-Spline basis function can be expressed as a linear combination of the other basis functions and thus the B-Spline Basis functions are linear independent.

2.3.7 B-Spline Basis Function Derivatives

With the simple quotient rule applied on the Cox de Boor recursive formula we can express the derivatives of a basis function recursively to the same derivative of its previous degree basis functions.

$$\frac{d}{d\xi}N_{i,p}(\xi)=\frac{p}{\xi_{i+p}-\xi_i}\left(\frac{d}{d\xi}N_{i,p-1}(\xi)\right)-\frac{p}{\xi_{i+p+1}-\xi_{i+1}}\left(\frac{d}{d\xi}N_{i+1,p-1}(\xi)\right)$$

We can generalize the above equation to higher derivatives

$$\frac{d^k}{d^k\xi}N_{i,p}(\xi)=\frac{p}{\xi_{i+p}-\xi_i}\left(\frac{d^k}{d^k\xi}N_{i,p-1}(\xi)\right)-\frac{p}{\xi_{i+p+1}-\xi_{i+1}}\left(\frac{d^k}{d^k\xi}N_{i+1,p-1}(\xi)\right)$$

and finally end up in the following expression

$$\frac{d^k}{d^k\xi}N_{i,p}(\xi)=\frac{p!}{(p-k)!}\sum_{j=0}^{k}a_{k,j}N_{i+j,p-k}(\xi)$$

where

$$a_{0,0}=1$$

$$a_{k,0}=\frac{a_{k-1,0}}{\xi_{i+p-k+1}-\xi_i}$$

$$a_{k,j}=\frac{a_{k-1,j}-a_{k-1,j-1}}{\xi_{i+p+j-k+1}-\xi_{i+j}}\quad,\quad j=1,\ldots,k\text{-}1$$

$$a_{k,k}=\frac{-a_{k-1,k-1}}{\xi_{i+p+1}-\xi_{i+k}}$$

When the denominator is zero, in case of repeated knot values, we consider the coefficient zero.

In case of more parametric directions, we get the partial derivatives of the shape functions.

For **2D** Shape Functions: $$\frac{\partial}{\partial\xi}R_{i,j}^{p,q}(\xi,\eta)=\left(\frac{\partial}{\partial\xi}N_{i,p}(\xi)\right)M_{j,q}(\eta)$$

and in the same way we get $\frac{\partial}{\partial\eta}R_{i,j}^{p,q}(\xi,\eta)$

For **3D** Shape Functions: $$\frac{\partial}{\partial\xi}R_{i,j,k}^{p,q,r}(\xi,\eta,\zeta)=\left(\frac{\partial}{\partial\xi}N_{i,p}(\xi)\right)M_{j,q}(\eta)\,L_{k,r}(\zeta)$$

and in the same way we get $\frac{\partial}{\partial\eta}R_{i,j,k}^{p,q,r}(\xi,\eta,\zeta)$, $\frac{\partial}{\partial\zeta}R_{i,j,k}^{p,q,r}(\xi,\eta,\zeta)$

2.3.8 B-Spline Geometries

B-Spline geometries can be Curves, Surfaces or Solids and are created with the combination of B-Spline Shape functions and their corresponding Control Points.

B-Spline Curve.

A p^{th} degree curve is defined by

$$(x,y,z) = C(\xi) = \underset{(1xn)}{\left\{N_{i,p}(\xi)\right\}^T} \underset{(n\,x\,3)}{\left\{P_i\right\}} = \sum_{i=1}^{n} N_{i,p}(\xi) \underset{(1x3)}{\left\{P_i\right\}} \quad , \quad \xi_1 \le \xi \le \xi_{n+p+1}$$

where the $\left\{P_i\right\}$ are the control points' Cartesian coordinates of the curve and $\underset{(nx1)}{\left\{N_{i,p}(\xi)\right\}}$ the p^{th} degree B-Spline basis functions defined on the non-periodic open knot value vector $\Xi = \left\{\xi_1, \xi_2, \cdots \xi_{n+p}, \xi_{n+p+1}\right\}$.

B-Spline Surface.

A B-Spline surface is obtained by taking a bidirectional net of control points, two knot vectors Ξ and H with the respective polynomial degrees p, q and the products of the univariate B-Spline functions:

$$(x,y,z) = S(\xi,\eta) = \sum_{i=1}^{n}\sum_{j=1}^{m} R_{i,j}^{p,q}(\xi,\eta) \underset{(1x3)}{\left\{P_{i,j,k}\right\}} = \sum_{i=1}^{n}\sum_{j=1}^{m} N_{i,p}(\xi) N_{j,q}(\eta) \underset{(1x3)}{\left\{P_{i,j}\right\}}$$

B-Spline Solid.

In the same fashion, a B-Spline volume is obtained by taking a three directional net of control points, three knot vectors Ξ, H, Z with the respective polynomial degrees p, q, r and the products of the univariate B-Spline functions:

$$(x,y,z) = V(\xi,\eta,\zeta) = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{l} R_{i,j,k}^{p,q,r}(\xi,\eta,\zeta) \underset{(1x3)}{\left\{P_{i,j,k}\right\}} = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{l} N_{i,p}(\xi) N_{j,q}(\eta) N_{k,r}(\xi) \underset{(1x3)}{\left\{P_{i,j,k}\right\}}$$

Point Iversion

Note that the inverse procedure, finding the parametric coordinates ξ, η, ζ of a point (x,y,z) in physical space, is more difficult and may require an iterative procedure. The topic is examined thoroughly in [14] in the chapter "Point Inversion".

2.3.9 B-Spline Curve Properties.

According to Piegl Tiller [14], the B-Spline Curves have the following properties:

1. B-Spline curves are a generalization of Bezier Curves.
2. C(ξ) is a piecewise polynomial curve.
3. Endpoint interpolation $C(\xi_1) = P_1$ and $C(\xi_{n+p+1}) = P_n$.
4. B-Spline curves possess strong convex hull property.
5. Local modification scheme: Moving a control point changes only a part of the curve near the control point.
6. The control polygon represents a piecewise linear approximation to the curve.
7. Affine invariance: An affine transformation is applied to the curve by applying it to the control points.
8. Variation diminishing property: No plane has more intersections with the curve than with the control polygon (in 2D curves, no line has more intersections than the control polygon).
9. C(ξ) is a linear combination of $N_{i,p}(\xi)$, thus the Curve's continuity and differentiability follow from that of the basis functions.
10. It is possible and sometimes useful to use multiple (coincident) control points.

The above properties all generalize in 2D surface and 3D solid B-Spline geometries.

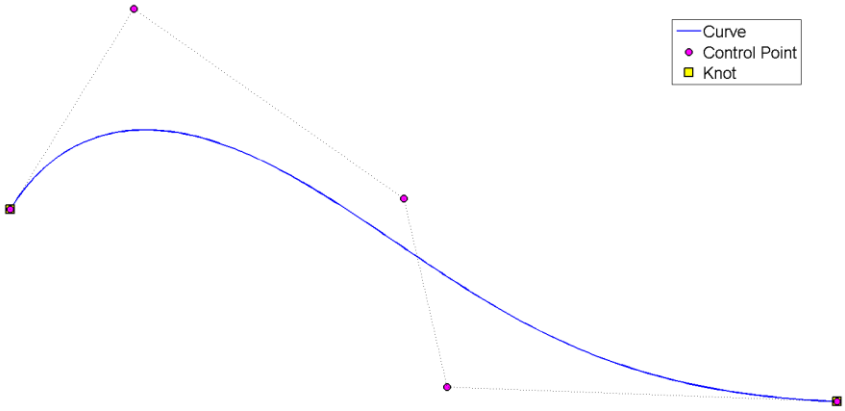
2.3.9.1 B-Spline curves are a generalization of Bezier Curves

We could refer to Bezier Curves as ancestors to B-Splines. Bezier Curves are B-Splines with a Knot Value Vector of a single knot span.

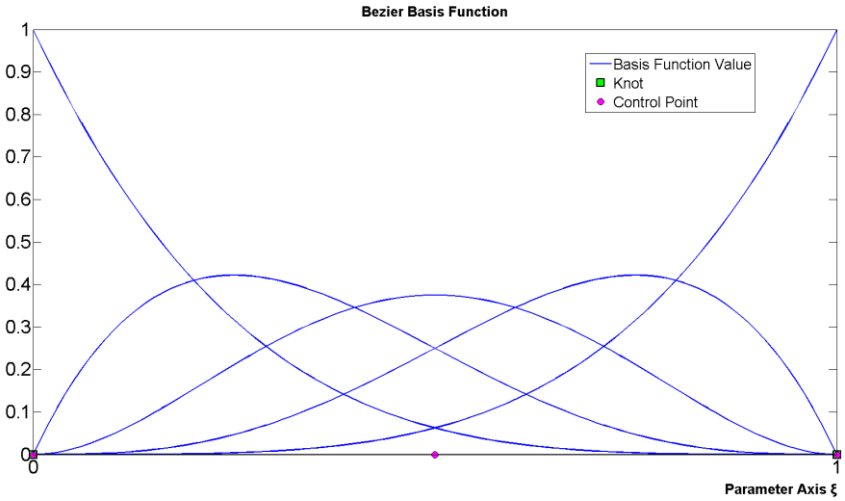
$$\Xi = \{\xi_1 = ... = \xi_{p+1} = 0,\ \xi_{p+2} = ... = \xi_{2p+2} = 1\}$$

This results in n=p+1 control points. Being defined over only one knot span all basis functions are non zero over the entire domain and thus every control point affects the whole model.

We can see an example of a Bezier Curve and its basis functions in Figure 2.23.



(a)



(b)

Figure 2.23. Bezier Curve
Bezier Curve in **(a)** Physical Space and **(b)** Basis functions in Parameter Space
Knot Value Vector: Ξ = { 0 0 0 0 0 1 1 1 1 1 }

2.3.9.2 C(ξ) is a piecewise polynomial curve

We can see that through the definition of the Curve,

$$(x,y,z)=C(\xi)=\underset{(1xn)}{\left\{N_{i,p}(\xi)\right\}^{T}}\underset{(nx3)}{\left\{P_i\right\}}=\sum_{i=1}^{n}N_{i,p}(\xi)\underset{(1x3)}{\left\{P_i\right\}}\quad,\quad \xi_1\le\xi\le\xi_{n+p+1}$$

We know that the B-Spline Basis functions $N_{i,p}(\xi)$ are piecewise polynomials which multiplied by coefficients $\underset{(1x3)}{\left\{P_i\right\}}$ and then summed, also result in a piecewise polynomial of ξ, the Curve C(ξ). Of course the tensor product surfaces and solids from combining B-Spline Basis functions along different directions ξ, η, ζ are also piecewise polynomials in respect to those directions. In general they are piecewise polynomials of ξ, η and ζ.

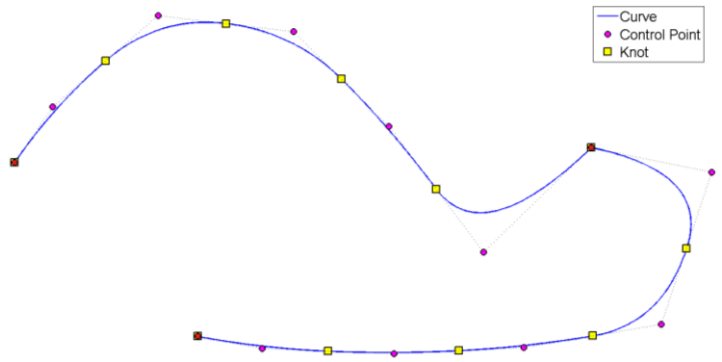
2.3.9.3 Endpoint interpolation $C(\xi_1)=P_1$ and $C(\xi_{n+p+1})=P_n$

The endpoint interpolation property states that the first and last control point of the curve lay upon the curve, on its first and last point. We will support this argument and also augment it with the statement that for every knot of C^0 continuity internal to the Knot Vector a control point also lies upon the Curve.

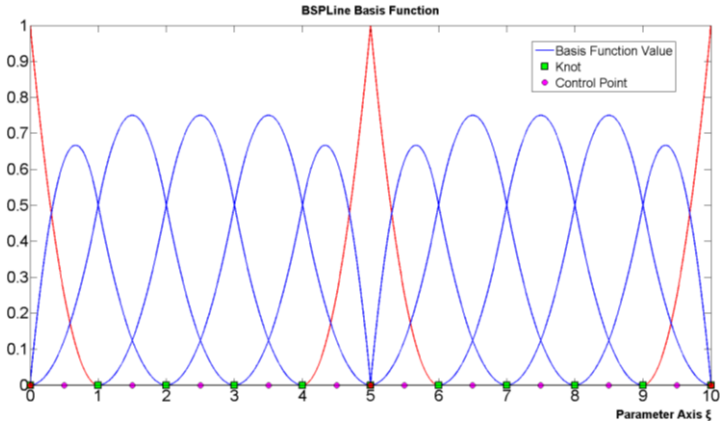
At points of C^0 continuity internally to the knot vector and C^{-1} continuity at the edges of the knot vector the shape functions are all zero except one that is 1.

$$C(\xi)=\sum_{i=1}^{n}N_{i,p}(\xi)\underset{(1x3)}{\left\{P_i\right\}}$$

From the definition of the Curve, that means that at that point $\bar{\xi}$ of C^0 or C^{-1} continuity $C(\bar{\xi})=N_{j,p}(\xi)\underset{(1x3)}{\left\{P_j\right\}}=1\cdot\underset{(1x3)}{\left\{P_j\right\}}=\underset{(1x3)}{\left\{P_j\right\}}$ where j is the j^{th} control point corresponding to the j^{th} Basis function that is 1 while all other functions $N_{i,p}(\xi)$, $i\neq j$ are zero. In the case of the first and last point, it is clear that the j^{th} basis function in the above expression is the first (j=1) and the last (j=n) basis function corresponding to the first and last control points. Thus the first and last control point are interpolatory to the curve. We can see that in Figure 2.24.



(a)



(b)

Figure 2.24. Interpolation of the curve at points of C^0 or C^{-1} Continuity. **(a)** Curve in Physical Space. **(b)** Basis functions in Parameter Space. Knot Value Vector Ξ = { 0 0 0 1 2 3 4 5 5 6 7 8 9 10 10 10 }

In cases of 2D surfaces or 3D solids, to actually have a control point interpolate the model there must be a C^0 or C^{-1} continuity on all available directions on the point $\bar{\xi}$, $\bar{\eta}$, $\bar{\zeta}$ in order to have only one shape function $R_{i,j,k}^{p,q,r}$ at that point be one and all others zero, resulting to

$$V(\bar{\xi},\ \bar{\eta},\ \bar{\zeta}) = R_{i,j,k}^{p,q,r} \underset{(1x3)}{\left\{P_{i,j,k}\right\}} = 1 \cdot \underset{(1x3)}{\left\{P_{i,j,k}\right\}} = \underset{(1x3)}{\left\{P_{i,j,k}\right\}}$$

In cases that all control points interpolate the curve, we practically approach the curve with standard finite elements with interpolatory nodes on the edges of each element. However this is a special case and thankfully is not found very often as the lower continuity brings up some drawbacks as well.

2.3.9.4 B-Spline curves possess strong convex hull property

The strong convex hull property states that the curve is contained in the convex hull of its control polygon and more specifically that:

If $\xi \in [\xi_i, \xi_{i+1})$ with $p \le i < n$, so that ξ is not in the starting and ending (p+1) trivial knot spans, then C(ξ) is in the convex hull of the control points P_{i-p}, P_i.

The strong convex hull property follows from the properties of non-negativity, partition of unity and local support of the B-Spline Basis functions. In Figure 2.25 we demonstrate the strong convex hull property at each knot span and over the whole curve.

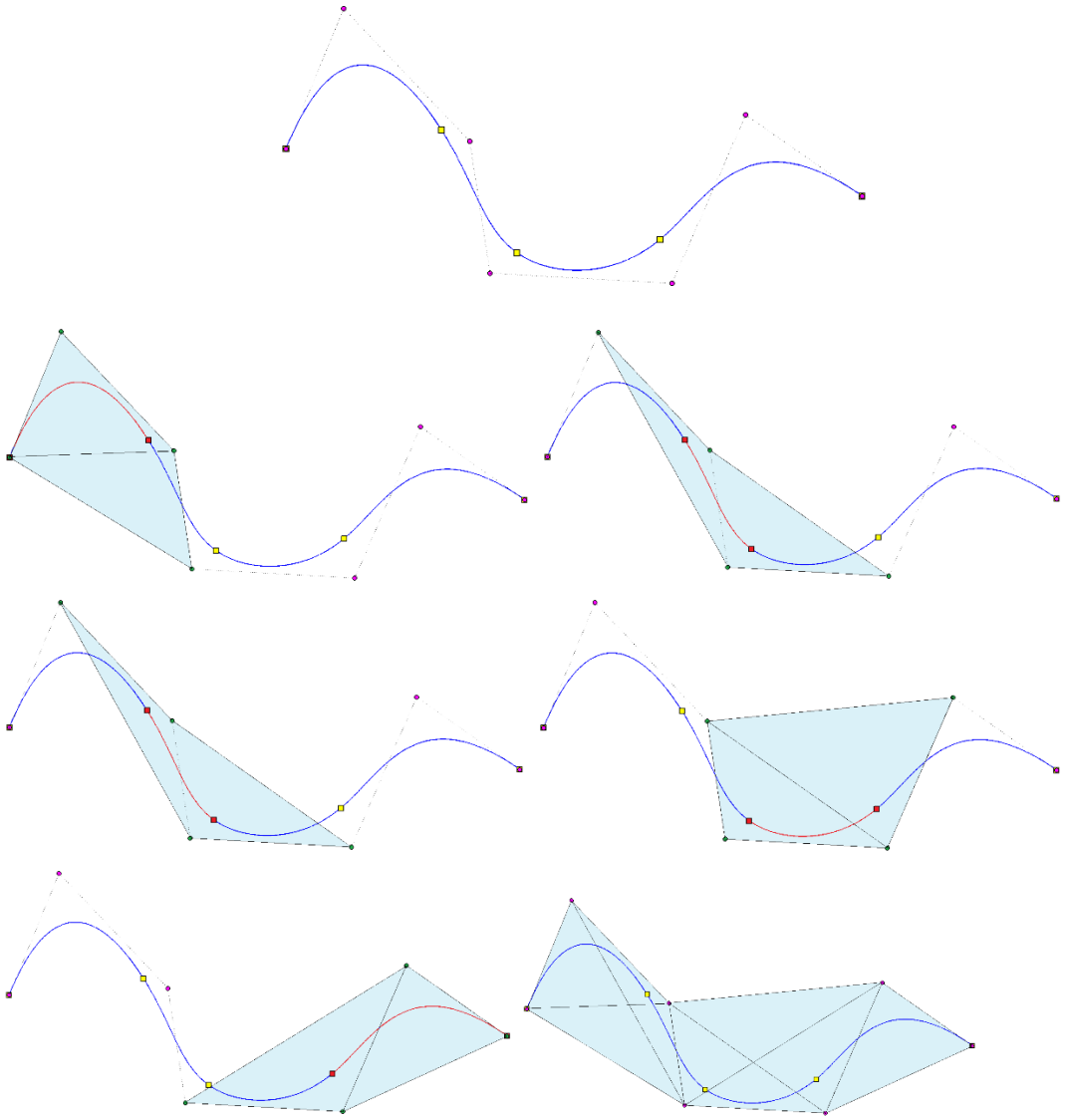
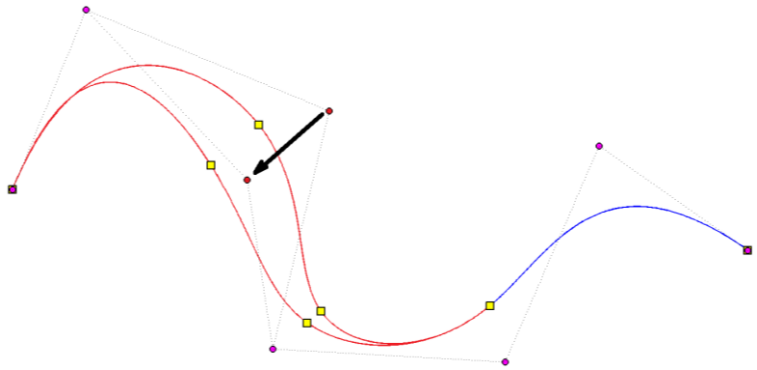


Figure 2.25. Demonstration of the Strong Convex Hull Property in B-Spline Curves. Knot Value Vector Ξ = { 0 0 0 0 1 2 3 4 4 4 4}

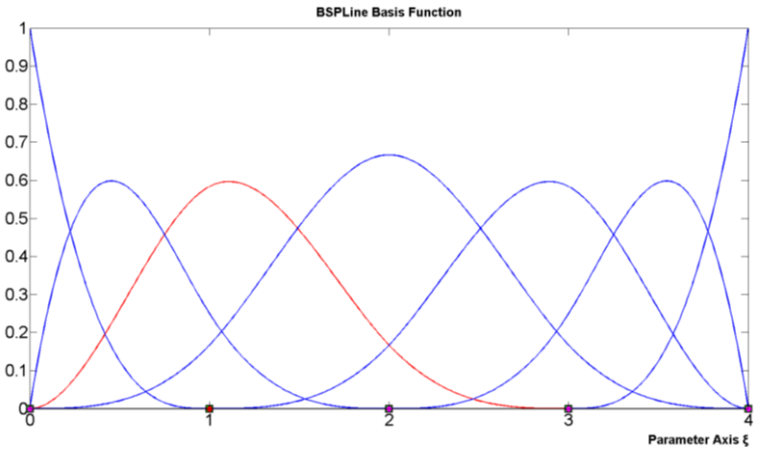
2.3.9.5 Local modification scheme: Moving a control point Pi changes only a part of the curve near the control point.

Moving the control point P_i only affects a local part of the curve. That follows after the local support properties of B-Spline Basis functions $N_{i,p}(\xi) = 0 \quad \forall \xi \notin \left[\xi_i, \xi_{i+p+1}\right)$

Changing the i^{th} control point's Cartesian coordinate doesn't matter outside the interval $\left[\xi_i, \xi_{i+p+1}\right)$ as only there the corresponding B-Spline $N_{i,p}(\xi) \neq 0$. Thus outside $\left[\xi_i, \xi_{i+p+1}\right)$ the contribution $N_{i,p}(\xi)\underset{(1x3)}{\{P_i\}} = 0$ to the curve is zero whether we change the coordinate or not. So indeed, moving a control point P_i changes only a part of the curve, the mapping of the interval $\left[\xi_i, \xi_{i+p+1}\right)$ to the Physical Space.



(a)



(b)

Figure 2.26. The control point local support in (a) Physical and (b) Parameter Space. Moving a Control Point affects only part of the curve.

The 3rd B-Spline and the corresponding 3rd control point have support the interval $[0,3)$ and that is the interval in the Physical space that is influenced by the change of the control point's Cartesian coordinate. We notice that in the interval $[3,4)$, out of the control points'

support, the Curve remains intact. In 2D and 3D cases the property applies in the same manner.

2.3.9.6 The control polygon represents a piecewise linear approximation to the curve

The control polygon represents a piecewise linear approximation to the curve. With knot refinement or order elevation more control points are added to the control polygon and due to convex hull property the control polygon is forced to get closer to the curve. In 2D and 3D cases the property is also valid. The control polygon approximates better the surface or the solid with control point insertions, with knot refinement or order elevation, which are refinement procedures of inserting Control Points and we will study in Chapter 5. In the following Figure 2.27 we performed consecutive knot insertions and order elevations to demonstrate this property.

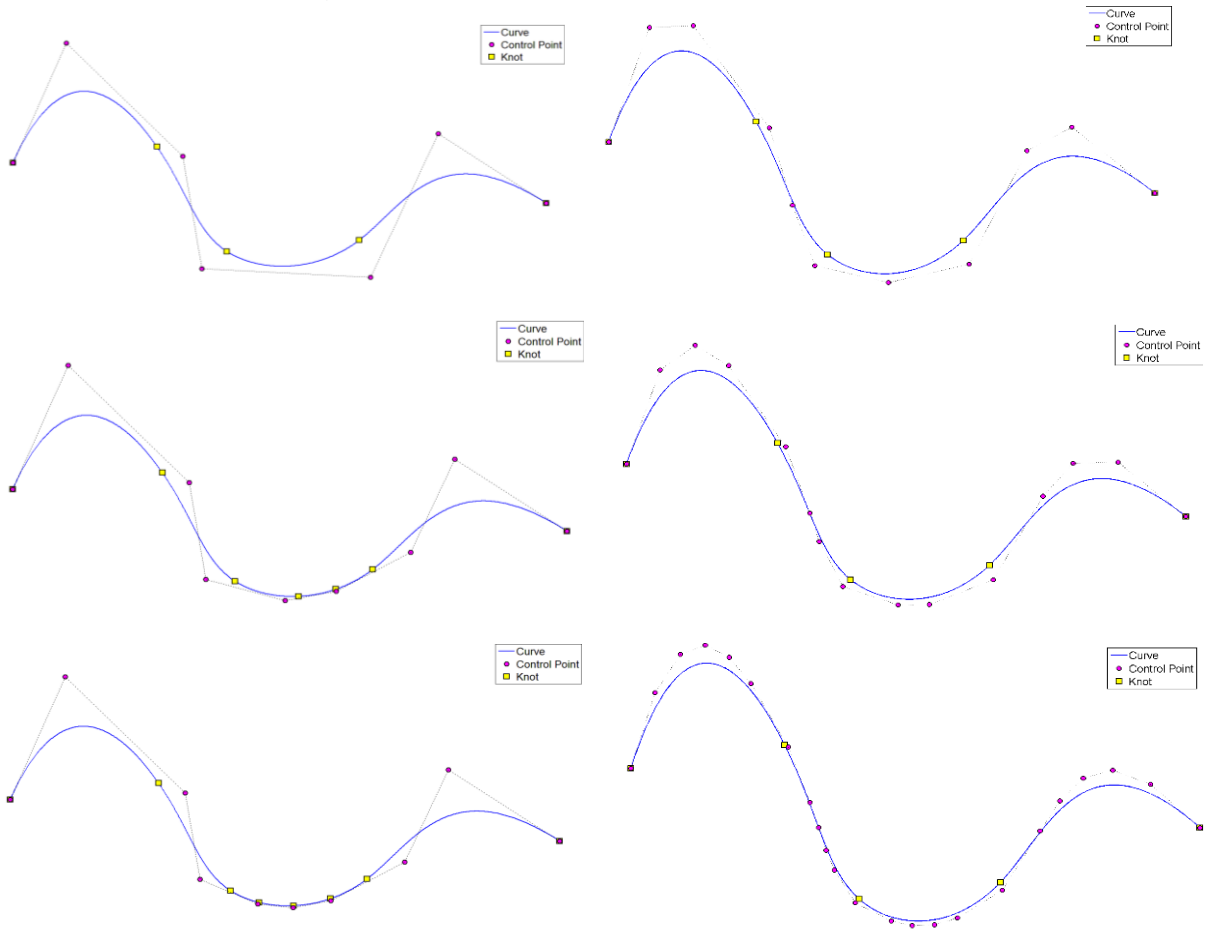


Figure 2.27. Approximation of the Control Polygon to the Curve with two consecutive knot insertions (Left column) and consecutive order elevations (right column) from degree 3 to 6.
Initial Knot Value Vector Ξ = { 0 0 0 0 1 2 3 4 4 4 4 }

2.3.9.7 Affine invariance: An affine transformation is applied to the curve by applying it to the control points

This is a very useful property on which we base our attempt to use NURBS and B-Splines as Basis functions for FEA. A direct result of that property is that by considering the Control Points as nodes and applying the displacements there, the whole model will have an analogous deformation. It is also necessary for partition of unity to make sense and have rigid body displacements by applying the same displacement over all the control points.

2.3.9.8 It is possible and sometimes useful to use multiple (coincident) control points.

An example is the widely used paradigm, in Isogeometric Analysis, of the plate with a hole. A way of representing it, is with a double control point on the upper left corner. The curve's convex hull forces the mapping of $[\xi_i, \xi_{i+1})$ in physical space to be inside the convex hull of the Control Points $P_{i-p},\ldots,P_i$. For p=2, it has to be in the convex hull of P_{i-2}, P_{i-1}, P_i. Therefore, by using the same cartesian coordinates for control points 2 and 3, the first knot span is the left vertical line and the second is the horizontal upper edge. In both cases, the curve is forced to connect 2 points with a straight line. The drawback when using multiple coincident control points is that we have points of singularity as multiple different points in Parameter space are mapped on the same point in Physical space.

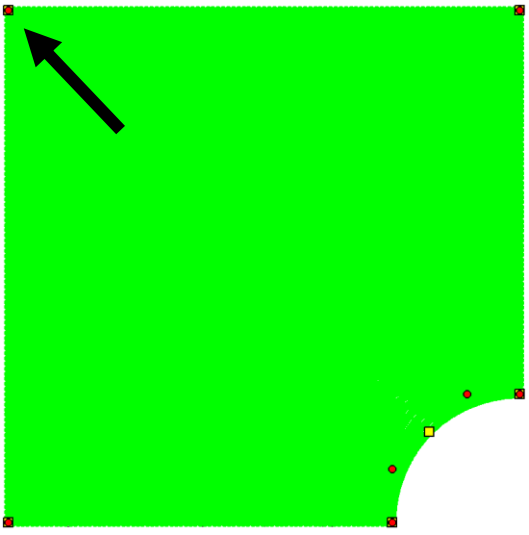


Figure 2.28. Plate with a hole. Represented with a double control point at the upper left corner.

2.4 Non Uniform Rational B-Splines

2.4.1 NURBS Concept

B-Splines where a breakthrough technology in CAD but unable to accurately interpolate some geometries, namely the conic sections. Those drawbacks are solved with the natural evolution of B-Splines, the NURBS standing for Non Uniform Rational B-Splines. B-Splines are already non uniform, recall that the knot value vectors we used until now are not uniform, but the extra term "rational" is the one making the difference. We can think the NURBS as a projection of B-Splines in a certain plane.

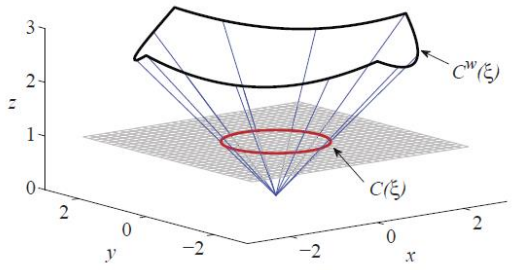


Figure 2.29.
Projection of the B-Spline Curve to plane z=1, forming the NURBS Curve: a circle.
(Image: Isogeometric analysis: toward integration of CAD and FEA)

In Figure 2.29, the B-Spline Curve in the 3D non-rational space with control points $P_i = (X_i, Y_i, w_i)$ is projected to the 2D rational space with control points $P_i = \left(\frac{X_i}{w_i}, \frac{Y_i}{w_i}\right)$ and weight w_i, forming the 2D NURBS Curve.

To project a NURBS geometry in 4D non-rational space, we project the geometry's Control Points to their corresponding projective non-rational, their weight equals to one, Control Points. Thus a Control Point $P_i = (x_i, y_i, z_i)$ with weight w_i from the 3D Cartesian space is projected to the non-rational 4D space control point $P_i^w = (w_i x_i, w_i y_i, w_i z_i, w_i)$.

$$(3D-\text{rational space}) \qquad \underset{(nx3)}{\{P\}} = \underset{(nx3)}{\{X_i, Y_i, Z_i\}} \text{ , with weight } w_i \xrightarrow{\cdot w_i}$$

$$(4D-\text{nonrational space}) \qquad \underset{(nx4)}{\{P^w\}} = \underset{(nx4)}{\{w_i X_i, w_i Y_i, w_i Z_i, w_i\}}$$

In general the d-dimensional NURBS are a projection of the (d+1) dimensional non-rational B-Splines.

2.4.2 NURBS Shape Functions

To define the NURBS Shape functions, we will also need to define the Weight function $W(\xi,\eta,\zeta)$.

For a **NURBS Curve** on parametric axis ξ the Weight function and the NURBS Shape function is:

$$W(\xi) = \sum_{i=1}^{n} N_{i,p}(\xi)\, w_i \,, \qquad R_i^p(\xi) = \frac{w_i\, N_{i,p}(\xi)}{W(\xi)}$$

For a **NURBS Surface** on parametric axes ξ, η, the Weight function and the NURBS Shape function is:

$$W(\xi,\eta) = \sum_{i=1}^{n}\sum_{j=1}^{m} N_{i,p}(\xi) M_{j,q}(\eta)\, w_{i,j} \qquad R_{i,j}^{p,q}(\xi,\eta) = \frac{w_{i.j}\, N_{i,p}(\xi) M_{j,q}(\eta)}{W(\xi,\eta)}$$

For a NURBS Solid on parametric axes ξ, η, ζ, the Weight function and the NURBS Shape function is:

$$W(\xi,\eta,\zeta) = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{l} N_{i,p}(\xi) M_{j,q}(\eta) L_{k,r}(\zeta)\, w_{i,j,k}$$

$$R_{i,j,k}^{p,q,r}(\xi,\eta,\zeta) = \frac{w_{i.j,k}\, N_{i,p}(\xi) M_{j,q}(\eta) L_{k,r}(\zeta)}{W(\xi,\eta,\zeta)}$$

Note that each NURBS Shape function is the corresponding B-Spline Shape function multiplied with the term $\frac{w_{i.j,k}}{W(\xi,\eta,\zeta)}$. In that way the NURBS are called rational, with the same meaning as in rational numbers. Also note, that when $w_{i,j,k}=1$ for all i, j, k, the NURBS Shape functions are identical with the B-Spline Shape functions. That means the NURBS are indeed a generalization of B-Splines who are able to represent accurately more geometries. In practice, most geometries can be adequately reproduced with B-Splines but NURBS are used whenever a shape close to conic section emerges.

2.4.3 NURBS Shape Function Derivatives

Applying the quotient rule we get the expression for the derivatives of NURBS Shape functions.

$$\frac{d}{d\xi}R_i^p(\xi)=\frac{\left(\frac{d}{d\xi}N_{i,p}(\xi)\right)W(\xi)-N_{i,p}(\xi)\left(\frac{d}{d\xi}W(\xi)\right)}{\left(W(\xi)\right)^2}w_i$$

In the case of more than one parametric directions we get the partial derivatives in the same way.

For 2D Shape Functions:

$$\frac{\partial}{\partial\xi}R_{i,j}^{p,q}(\xi,\eta)=\frac{\left(\frac{\partial}{\partial\xi}N_{i,p}(\xi)\right)W(\xi,\eta)-N_{i,p}(\xi)\left(\frac{\partial}{\partial\xi}W(\xi,\eta)\right)}{\left(W(\xi,\eta)\right)^2}w_{i,j}M_{j,q}(\eta)$$

$$\frac{\partial}{\partial\eta}R_{i,j}^{p,q}(\xi,\eta)=\frac{\left(\frac{\partial}{\partial\eta}M_{j,q}(\eta)\right)W(\xi,\eta)-M_{j,q}(\eta)\left(\frac{\partial}{\partial\eta}W(\xi,\eta)\right)}{\left(W(\xi,\eta)\right)^2}w_{i,j}N_{i,p}(\xi)$$

And for 3D Shape Functions

$$\frac{\partial}{\partial\xi}R_{i,j,k}^{p,q,r}(\xi,\eta,\zeta)=\frac{\left(\frac{\partial}{\partial\xi}N_{i,p}(\xi)\right)W(\xi,\eta,\zeta)-N_{i,p}(\xi)\left(\frac{\partial}{\partial\xi}W(\xi,\eta,\zeta)\right)}{\left(W(\xi,\eta,\zeta)\right)^2}w_{i,j,k}M_{j,q}(\eta)L_{k,r}(\zeta)$$

$$\frac{\partial}{\partial\eta}R_{i,j,k}^{p,q,r}(\xi,\eta,\zeta)=\frac{\left(\frac{\partial}{\partial\eta}M_{j,q}(\eta)\right)W(\xi,\eta,\zeta)-M_{j,q}(\eta)\left(\frac{\partial}{\partial\eta}W(\xi,\eta,\zeta)\right)}{\left(W(\xi,\eta,\zeta)\right)^2}w_{i,j,k}N_{i,p}(\xi)L_{k,r}(\zeta)$$

$$\frac{\partial}{\partial\zeta}R_{i,j,k}^{p,q,r}(\xi,\eta,\zeta)=\frac{\left(\frac{\partial}{\partial\zeta}L_{k,r}(\zeta)\right)W(\xi,\eta,\zeta)-L_{k,r}(\zeta)\left(\frac{\partial}{\partial\zeta}W(\xi,\eta,\zeta)\right)}{\left(W(\xi,\eta,\zeta)\right)^2}w_{i,j,k}N_{i,p}(\xi)M_{j,q}(\eta)$$

For higher derivatives exist some convenient recursive formulas and are available in Piegl and Tiller [14].

2.4.4 NURBS Geometries

NURBS entities can be curves surfaces or solids. They are created in the same way as B-Spline geometries with the use of NURBS Shape functions and their corresponding control points.

Curves:

$$C(\xi) = \sum_{i=1}^{n} R_i^p(\xi) P_i$$

Surfaces:

$$S(\xi, \eta) = \sum_{i=1}^{n} \sum_{j=1}^{m} R_{i,j}^{p,q}(\xi, \eta) P_{i,j}$$

Solids:

$$V(\xi, \eta, \zeta) = \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{l} R_{i,j,k}^{p,q,r}(\xi, \eta, \zeta) P_{i,j,k}$$

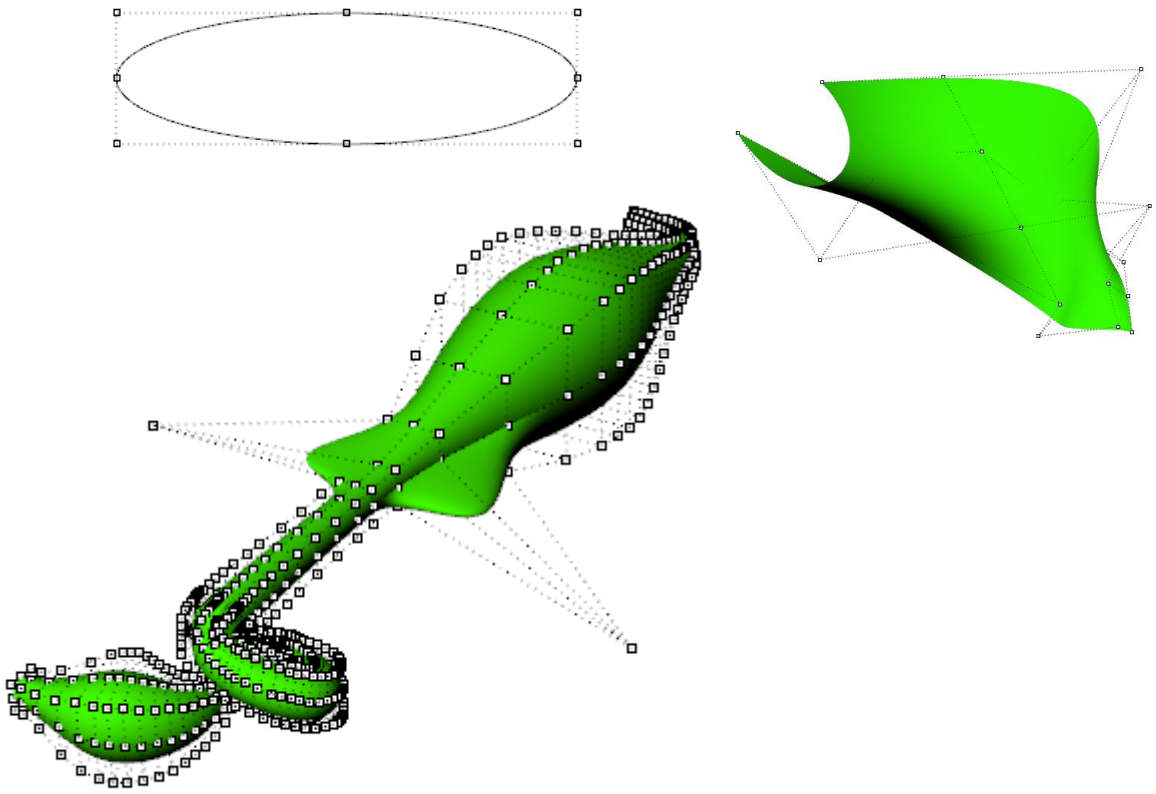
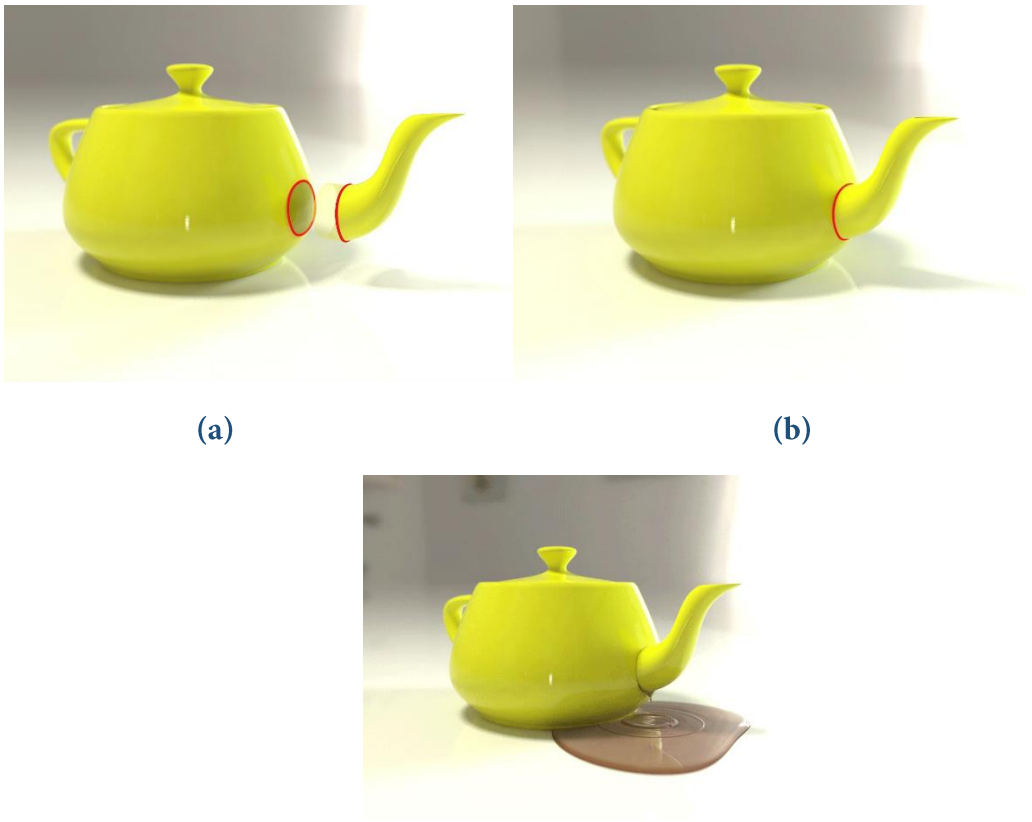


Figure 2.30. NURBS Entities.
Ellipse Curve, a Surface and a Solid.

2.4.5 Patches.

The Patches emerged in CAD as a way to represent very complex geometries with accuracy. Major changes in the Physical Space geometry, namely the changing of a circle into another conic section, cannot always be exactly represented with a single knot value or with a simple line, rectangle or a cube in Parameter Space. Furthermore, even when the geometry is simple, if the material changes then we have to draw the model with different patches in order to transition from modelling to analysis. If such major changes take place it is easier to cut the model into separate pieces, model each one separately and then glue them together again.

Generally, in Finite Elements, in the merging of two meshes we can have cases of overlapping or non-overlapping meshes, with coincident or non-coincident nodes at the edge. In Isogeometric analysis with NURBS, the connection of two patches with equal degree of polynomials is still an issue and problems occur even in the simplest case of non-overlapping meshes with coincident control points. The connection of the patches in NURBS can be difficult and there are serious native problems involved, not allowing us to ensure "water-tight" connections in the general case.



(a) **(b)**

(c)

Figure 2.31. The connection between patches in NURBS can be problematic. Water tight connection cannot be achieved in the general case. (http://www.siam.org/news/news.php?id=1874)

In the case we attempt to merge two patches of equal degrees and coincident control points at the merging edge, we in fact merge their knot vectors into one.

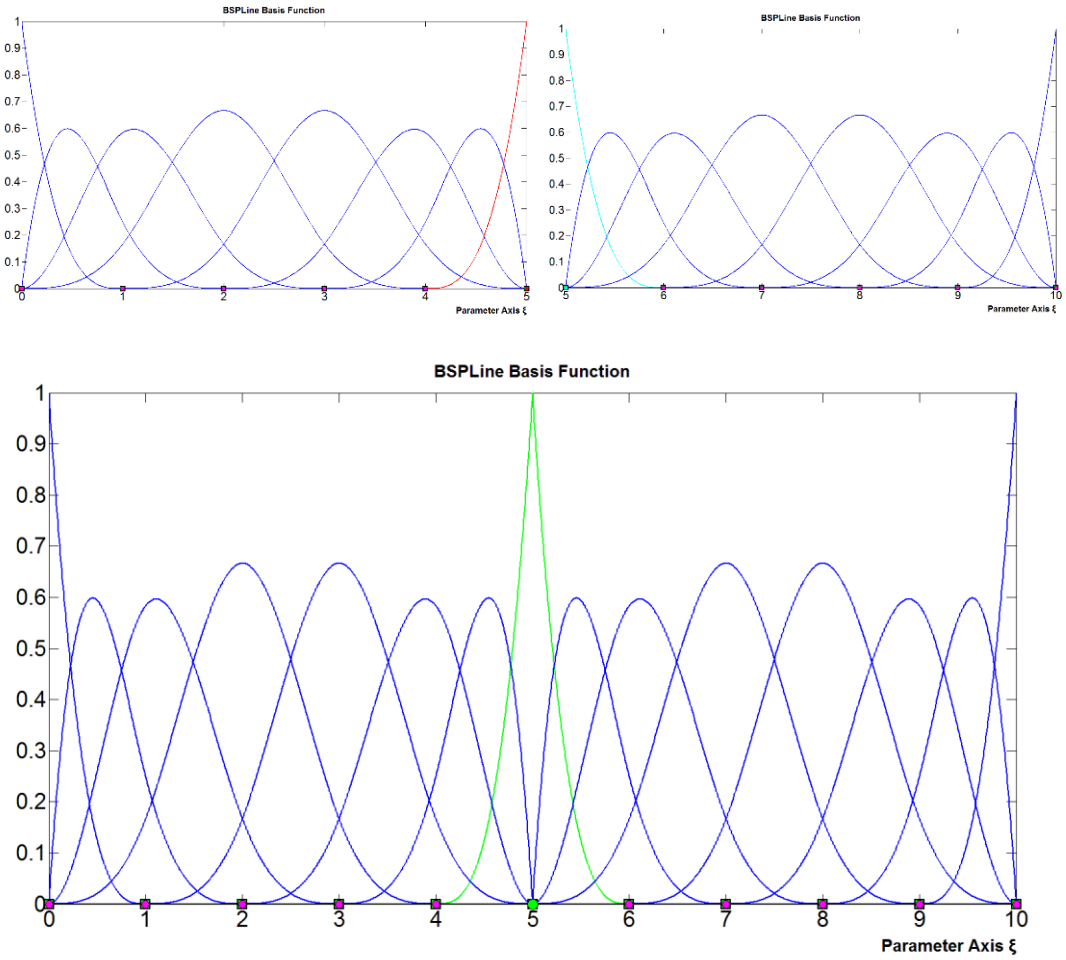


Figure 2.32. Two separate patches with coincident edge control points and are merged into one in Parameter Space. At the merging knot we have C^0 continuity.
Knot Value Vector Ξ = { 0 0 0 0 1 2 3 4 5 5 5 6 7 8 9 10 10 10 10 }

We are able to connect them that way, because due to the multiplicity at the start and end of the separate knot vectors, all other functions are zero and the geometry and displacement are defined only by the certain control point at the edge.

3 Stiffness Matrix

3.1 Preliminary Steps for Analysis

3.1.1 Shape Functions

The use of the CAD Shape functions for Analysis is necessary to integrate CAD and FEA but these basis functions have some special characteristics in respect with the standard FEA Lagrange Polynomial functions. FEA standard Shape functions are interpolatory with C^0 continuity at all nodes and furthermore with C^{-1} continuity at the edges. That prohibits FEA from correctly defining stresses and strains at the boundaries as no derivatives exist there and to overcome that, we resort to auxiliary corrective methods. Instead, in Isogeometric Analysis with NURBS, natively high continuity shape functions are provided, with all derivatives continuous inside the elements and C^{p-k} continuity along the element boundaries. That results in elements with overlapping, an asset that gives us the opportunity to better simulate the actual connections between regions of the structure. It is a common misconception that this attribute leads to a greater bandwidth. But whether in FEA or IGA with NURBS, any given function of degree p shares support with 2p other functions, as we have already seen in previous chapters, and thus the bandwidth is the same in both cases.

3.1.2 Control Points

Generally, in Finite Elements we assemble a stiffness matrix where the unknowns are the degrees of freedom on the nodes of the mesh. The corresponding concept in IGA are the Control Points. They serve as coefficients of the geometry for the Shape functions and in combination with them, they approximate the initial geometry and the solution field.

3.1.3 Elements.

In isogeometric analysis, the patches are sometimes referred to as elements, but we usually consider the different knot spans as elements. Many try to find a direct analogous of FEA in IGA and, having the interpolatory basis functions at the end of the finite element in mind, they consider the patches as elements. Indeed, at the edges of the patch the functions are interpolatory too and that way, with continuity C^{-1} at the edges, there is no overlapping between the patches. However, there have been cases where the patch has higher continuity and overlapping exists. For this thesis we decided, as a more direct analogous to FEM, to consider knot spans as elements. We can justify that, as at the end of the knot spans, the continuity, even if it is not C^0, changes due to the piecewise nature of the polynomial basis

and as a result the Gauss quadrature uses polynomials laid in the interior of a single knot span.

3.1.4 Quadrature

Due to the difficulty of an analytical integration, we generally resort to numerical solutions for the integrals, especially in the formulation of the stiffness matrix. The Gauss Quadrature is used widely in such cases and will be described in the following paragraphs.

For the integration of a function over the patch, we compute the coordinates and weights of a number of gauss points in the patch. We compute the numerical value of the function there and we multiply it with the gauss point's weight. The sum of those multiplications over the patch is approximate to the integral.

$$I = \int_{\xi_1}^{\xi_n} f(\xi) d\xi = \sum_{i=1}^{n_{GP\xi}} f(\xi_i) w_i^{GP\xi}$$

3.1.5 Number of Gauss Point

We want to minimize the computational cost and therefore we want the minimum number of Gauss Points per Knot Span that will give us the exact value of the integral, provided that the function f is a polynomial of degree q. According to Hughes, Reali and Sangalli [2], that number is:

$$n_{GP}^{perKnotSpan} = \begin{cases} \frac{q+1}{2}, \text{ for q odd} \\ \frac{q+2}{2}, \text{ for q even} \end{cases}$$

The numerical integration is referring to the stiffness matrix for which the integral has the form $[K] = \iiint_{\xi,\eta,\zeta} [B]^T [E][B] \det([J]) d\xi d\eta d\zeta$.

Let p be the maximum degree the basis functions have along the parametric axes.

In the case of **1D**,

The maximum degree of the Deformation Matrix $[B]$ is defined from the degree of the derivatives of the shape functions which is (p-1). Then the product of $[B]^T [E][B]$ is a polynomial of maximum degree (p-1)+(p-1)=2p-2 which is always an even number. Subsequently, the minimum number of Gauss Points needed for exact integration is:

$$n_{GP}^{1D} = \frac{(2p-2)+2}{2} = p$$

In the case of **2D or 3D**,

the maximum degree of the Deformation Matrix $[B]$ is defined from the degree of the partial derivatives of the shape functions, which is p. Then the product of $[B]^T[E][B]$ is a polynomial of maximum degree p+p=2p which is always an even number. Subsequently the minimum number of Gauss Points needed for exact integration is:

$$n_{GP}^{3D} = n_{GP}^{2D} = \frac{2p+2}{2} = p+1$$

Conclusively we need:

p Gauss Points per Knot Span in **1D** problems

p+1 Gauss Points per Knot Span in **2D** or **3D** problems

3.1.6 Parametric Coordinates and Weights of Gauss Points

After we get the number of the Gauss points we will use for each knot span, we compute the Gauss point coordinates in the knot span. The coordinates of the Gauss points are determined as the roots of the Legendre Polynomial and its weights as a function of its derivative. Both coordinates and weights, however, lie in a reference knot span [-1,1].

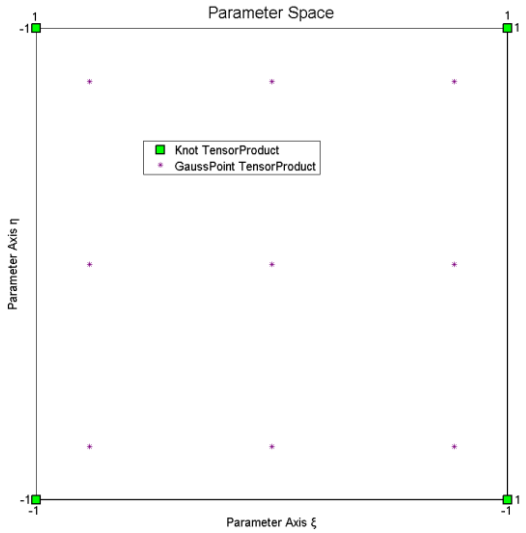


Figure 3.1. Reference Knot Span [-1, 1]
Degree on ξ and η = 2. Gauss Point Number = 3x3 = 9
Gauss Point Parametric Coordinates Ξ={-0.77459 , 0 , 0.77459} x H={-0.77459 , 0 , 0.77459}

Those coordinates and weights are already computed in bibliography for sufficient cases of Gauss point numbers. We have the following matrix of Gauss point coordinates and weights in the Reference interval (-1,1) in respect with the Gauss Point number n.

n	ξ_i^R	w_i
1	0.0	2.0
2	±0.57735	1.0
3	±0.77459	0.55555
	0.0	0.88888
4	±0.86113	0.34785
	±0.33998	0.65214
5	±0.90617	0.23692
	0.53846	0.47862
	0.0	0.56888
6	±0.93246	0.17132
	±0.66120	0.36076
	±0.23861	0.46791

Figure 3.2. Parametric Coordinates and Weights
for Gauss numerical integration in the Reference interval (-1,1)

Now that we have the coordinates $\xi^R$and weight w^R of each gauss point we have to move them from the Reference knot span to our desired knot span $[\xi_i, \xi_{i+1})$.

$$\xi = \frac{(\xi_{i+1} - \xi_i)\xi^R + (\xi_{i+1} + \xi_i)}{2}$$

$$w^{GP\xi} = \frac{(\xi_{i+1} - \xi_i)}{2} w_\xi^R$$

The full tensor product property applies to gauss points too, allowing us to calculate coordinates and weight the same way in all axes ξ,η,ζ and combine them.

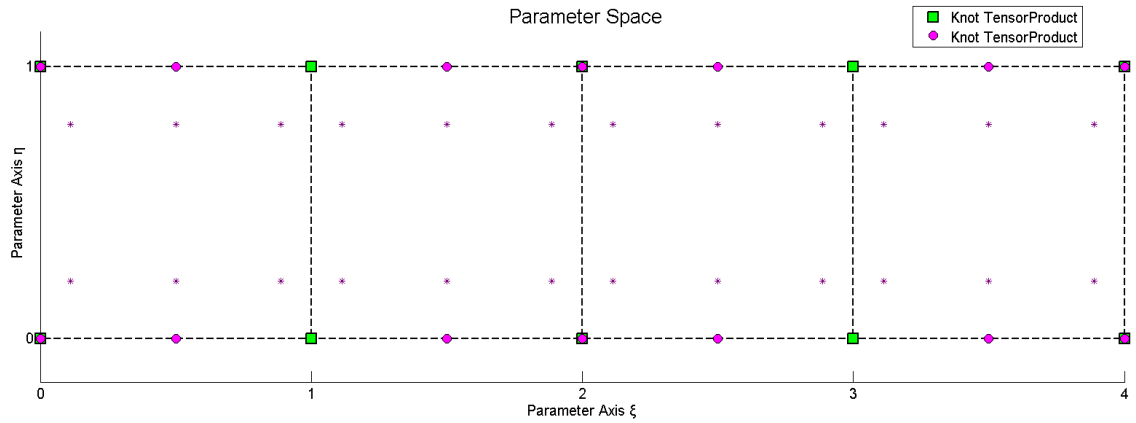


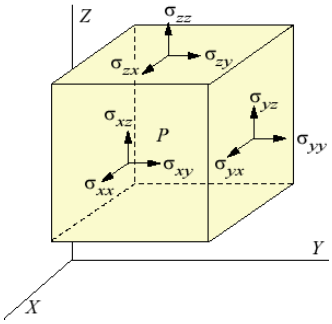
Figure 3.3. Gauss Points in Parameter Space.
Knot Value Vectors Ξ = {0 0 0 1 2 2 3 4 4 4}, H= {0 0 1 1}

3.1.7 Theory of Elasticity and the Elasticity Matrix

Depending on the stress and strain field for each case we use different elasticity matrices. We will present elasticity matrices for the standard cases of 1D elasticity, 2D elasticity (plane stress and plane strain) and 3D elasticity. The symbol E stands for Young's modulus of elasticity and ν for Poisson's ratio.

At any point in a solid, there are six independent stress components and six corresponding strain components. The stress components are

$$\underset{(1x6)}{\{\sigma\}^T} = \{\sigma_{xx} \quad \sigma_{yy} \quad \sigma_{zz} \quad \sigma_{xy} \quad \sigma_{yz} \quad \sigma_{zy}\}$$



and the strain components

$$\underset{(1x6)}{\{\varepsilon\}^T} = \{\varepsilon_{xx} \quad \varepsilon_{yy} \quad \varepsilon_{zz} \quad \gamma_{xy} \quad \gamma_{yz} \quad \gamma_{zy}\}$$

Hooke's law connects these stresses and strains with a relation easier expressed through a matrix equation $\underset{(6x1)}{\{\sigma\}} = \underset{(6x6)}{[E]}\,\underset{(6x1)}{\{\varepsilon\}}$.

3D elasticity

In the general case of **3D elasticity** that can be expressed as

$$\underset{(6x1)}{\{\sigma\}} = \underset{(6x6)}{[E]}\,\underset{(6x1)}{\{\varepsilon\}} \Rightarrow \underset{(6x1)}{\begin{bmatrix}\sigma_{xx}\\ \sigma_{yy}\\ \sigma_{zz}\\ \sigma_{xy}\\ \sigma_{yz}\\ \sigma_{zx}\end{bmatrix}} = \underbrace{\frac{E}{(1-\nu)(1-2\nu)}\begin{bmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0\\ \nu & 1-\nu & \nu & 0 & 0 & 0\\ \nu & \nu & 1-\nu & 0 & 0 & 0\\ 0 & 0 & 0 & \frac{1-2\nu}{2} & 0 & 0\\ 0 & 0 & 0 & 0 & \frac{1-2\nu}{2} & 0\\ 0 & 0 & 0 & 0 & 0 & \frac{1-2\nu}{2}\end{bmatrix}}_{\underset{(6x6)}{[E_{3D\ Elasticity}]}} \underset{(6x1)}{\begin{bmatrix}\varepsilon_{xx}\\ \varepsilon_{yy}\\ \varepsilon_{zz}\\ \gamma_{xy}\\ \gamma_{yz}\\ \gamma_{zx}\end{bmatrix}}$$

All the other cases are derived from the general case of 3D elasticity by marking the strains or stresses we know do not exist as nil.

The case of 2D elasticity

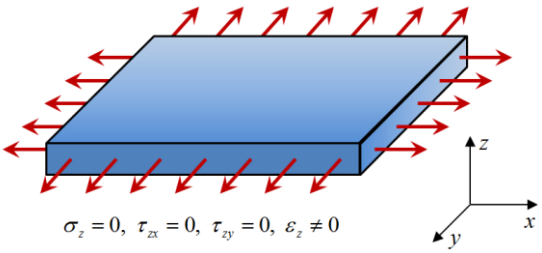
For a 2D solid let us assume that all dependent variables are independent of the z coordinate and all external loads are applied in the xy plane.

We have now 2 major cases: Plane Stress and Plain Strain.

2D Elasticity, Plane stress

This is the case where a plate is loaded in its midplane and all loads are symmetric with respect to the midplane, the support conditions are symmetric about the midplane, the in-plane displacements, strains and stresses can be assumed uniform along the thickness and the normal and shear stress components in the z direction can be considered zero or negligible. Then the plate is said to be in a state of plane stress, or a membrane state.
In that case, components σ_z, σ_{xz}, σ_{yz} and γ_{xz}, γ_{yz} are zero. The state of stress in that case is specified by stresses σ_x, σ_y, σ_{xy} and strains ε_x, ε_y, γ_{xy}.

$$\underset{(3x1)}{\begin{bmatrix}\sigma_{xx}\\ \sigma_{yy}\\ \tau_{xy}\end{bmatrix}} = \underbrace{\frac{E}{1-\nu^2}\begin{bmatrix}1 & \nu & 0\\ \nu & 1 & 0\\ 0 & 0 & \frac{1-\nu}{2}\end{bmatrix}}_{[E_{\text{Plane Stress}}]\ (3x3)} \underset{(3x1)}{\begin{bmatrix}\varepsilon_{xx}\\ \varepsilon_{yy}\\ \gamma_{xy}\end{bmatrix}}$$



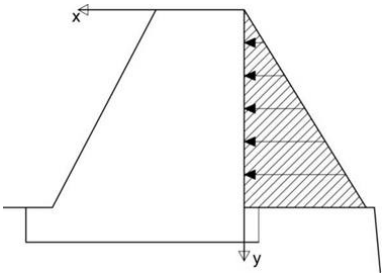
and strain $\varepsilon_z = -\frac{\nu}{1-\nu}(\varepsilon_x + \varepsilon_y)$

2D Elasticity, Plane strain

This is the case where the dimension of the structure in direction z is much larger than the other two dimensions in x and y direction, all external forces are applied parallel to the xy plane and do not vary in the z direction. Typical cases of interest for an engineer are retaining walls, dams, tunnels or, in smaller scale, bars and rollers with forces normal to their cross section.

For isotropic materials we can assume that stresses $\sigma_{xz}=\sigma_{yz}=0$ and strains $\varepsilon_z=\gamma_{xz}=\gamma_{yz}=0$. The stress-strain components are related as:

$$\underset{(3x1)}{\begin{bmatrix}\sigma_{xx}\\ \sigma_{yy}\\ \tau_{xy}\end{bmatrix}} = \underbrace{\frac{E}{(1+\nu)(1-2\nu)}\begin{bmatrix}1-\nu & \nu & 0\\ \nu & 1-\nu & 0\\ 0 & 0 & \frac{1-2\nu}{2}\end{bmatrix}}_{[E_{\text{Plane Strain}}]\ (3x3)} \underset{(3x1)}{\begin{bmatrix}\varepsilon_{xx}\\ \varepsilon_{yy}\\ \gamma_{xy}\end{bmatrix}}$$



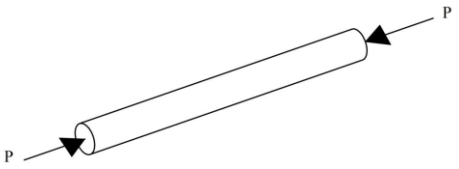
and stress $\sigma_z = \frac{E\nu}{(1+\nu)(1-2\nu)}(\varepsilon_{xx} + \varepsilon_{yy})$

1D Elasticity, truss

It is the case of a truss member, a solid whose dimension in one direction (axial direction) is much larger than the other two and is subjected only to axial forces.

$$\underset{(1x1)}{\{\sigma_x\}} = \underset{(1x1)}{[E]}\underset{(1x1)}{\{\varepsilon_x\}} \Rightarrow \sigma_x = E\varepsilon_x$$

with $\underset{(1x1)}{\left[E_{1D\ Elasticity}\right]} = E$



3.1.8 Mapping of Parameter to Physical space

The isoparametric elements where originally invented by Taig and Irons in their attempt to build not orthogonal elements with curved sides. The isoparametric concept is based on a use of a second coordinate system defined over the Cartesian system through a mapping. It is the natural system in FEM, the equivalent to parameter space in IGA.

The mapping between Parameter and Physical space must be "one to one", that is every point with coordinates (ξ,η,ζ) in parameter has one and only one corresponding point with coordinates (x,y,z) in physical space, and vice versa. To proceed with the change of coordinates we will need a Jacobian matrix $[J]$ for the integrals and more precisely we will need $[J]^{-1}$ as the integrals from physical space will be computed in the parameter space.

The $[J]^{-1}$ can only exist if the aforementioned mapping is "one to one". To that end, special care has to be taken so that the positive direction of the axes in Physical and Parameter space coincide or else the determinant of the Jacobian will be negative.

The normal mapping from Parameter to Physical Space will give us the coordinates (x,y,z) as a function of the parametric coordinates ξ,η,ζ:

$$(x,y,z) = \begin{Bmatrix} x = x(\xi,\eta,\zeta) \\ y = y(\xi,\eta,\zeta) \\ z = z(\xi,\eta,\zeta) \end{Bmatrix} = S(\xi,\eta,\zeta) = \sum_{1}^{n} N_i(\xi,\eta,\zeta)\underset{(1x3)}{\{P_i\}}$$

and the inverse mapping from Physical to Parameter Space will give us the coordinates ξ,η,ζ as a function of the physical coordinates x,y,z:

$$(\xi,\eta,\zeta) = \begin{Bmatrix} \xi(x,y,z) \\ \eta(x,y,z) \\ \zeta(x,y,z) \end{Bmatrix}$$

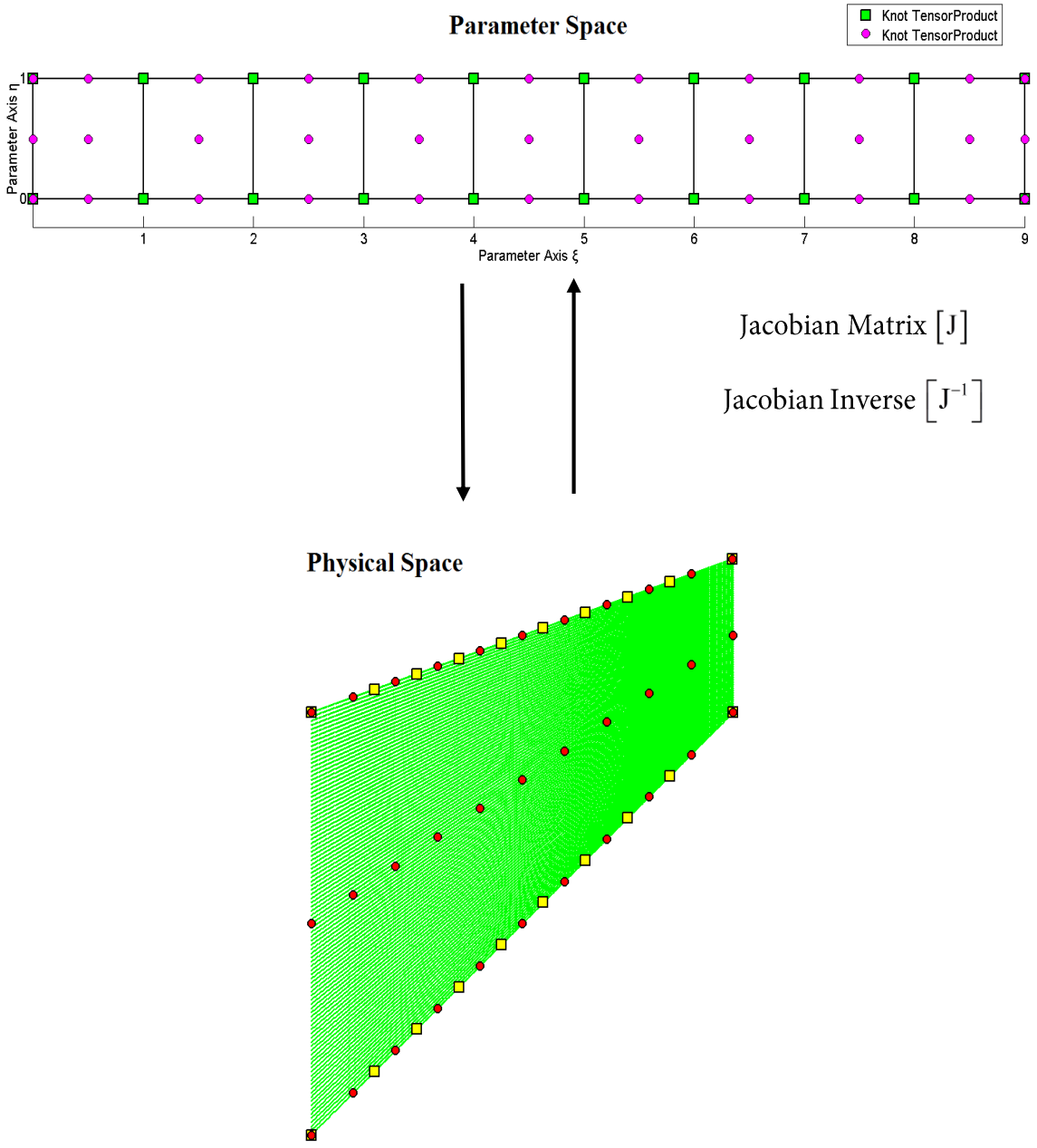
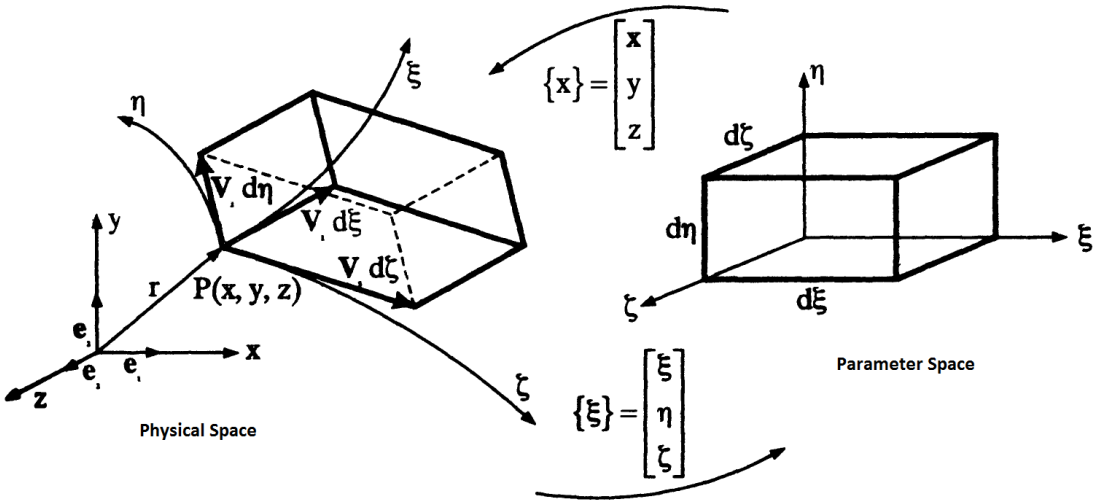


Figure 3.4. Cook's Cantilever 2D.Transition from Physical to Parameter Space and vice versa. Knot Value Vectors: Ξ ={ 0 0 0 1 2 3 4 5 6 7 8 9 9 9 } , H = { 0 0 0 1 1 1 }

3.1.9 Integrals, Jacobian matrix, dV,dA,dx

Projecting the parametric system into the physical space gives us a curved coordinate system in every point P, as shown in the figure below:



The point vector of the point P is given by the relation

$$\{r\} = x\{e_1\} + y\{e_2\} + z\{e_3\}$$

where x,y,z are the coordinates of P and $\{e_i\}$ the unitary vector on axis x,y,z respectively.

The tangent vectors on curves ξ,η,ζ in physical space are given by the relations:

$$\{V_1\} = \frac{\partial\{r\}}{\partial\xi} = \frac{\partial x}{\partial\xi}\{e_1\} + \frac{\partial y}{\partial\xi}\{e_2\} + \frac{\partial z}{\partial\xi}\{e_3\}$$

$$\{V_2\} = \frac{\partial\{r\}}{\partial\eta} = \frac{\partial x}{\partial\eta}\{e_1\} + \frac{\partial y}{\partial\eta}\{e_2\} + \frac{\partial z}{\partial\eta}\{e_3\}$$

$$\{V_3\} = \frac{\partial\{r\}}{\partial\zeta} = \frac{\partial x}{\partial\zeta}\{e_1\} + \frac{\partial y}{\partial\zeta}\{e_2\} + \frac{\partial z}{\partial\zeta}\{e_3\}$$

Then, the elementary parallelepiped has sides $\{V_1\}d\xi$, $\{V_2\}d\eta$, $\{V_3\}d\zeta$ and volume

$$dV = (\{V_1\}d\xi)\left[(\{V_2\}d\eta)\times(\{V_3\}d\zeta)\right] = \{V_1\}(\{V_2\}\times\{V_3\})d\xi\,d\eta\,d\zeta \xrightarrow{\text{replacing } \{V_1\},\{V_2\},\{V_3\}}$$

$$dV = \left(\frac{\partial x}{\partial \xi}\{e_1\} + \frac{\partial y}{\partial \xi}\{e_2\} + \frac{\partial z}{\partial \xi}\{e_3\} \right) \begin{vmatrix} \{e_1\} & \{e_2\} & \{e_3\} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta} \\ \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} & \frac{\partial z}{\partial \zeta} \end{vmatrix} d\xi d\eta d\zeta \Rightarrow$$

$$dV = \begin{vmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta} \\ \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} & \frac{\partial z}{\partial \zeta} \end{vmatrix} d\xi d\eta d\zeta = \det \underset{(3x3)}{[J]}\, d\xi d\eta d\zeta \quad \text{and} \quad \underset{(3x3)}{[J]} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta} \\ \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} & \frac{\partial z}{\partial \zeta} \end{bmatrix}$$

In the case of 2D elasticity, the area dA of the elementary parallelogram in physical space is:

$$dA = \begin{vmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{vmatrix} d\xi d\eta = \det \underset{(2x2)}{[J]}\, d\xi d\eta$$

Finally in the case of 1D elasticity, the length dx of the elementary line in physical space is:

$$dx = \left| \frac{dx}{d\xi} \right| d\xi = \det \underset{(1x1)}{[J]}\, d\xi$$

In each of the above cases, in order to compute the integral we need to evaluate the Jacobian matrix.

We have that

$$(x, y, z) = S(\xi, \eta, \zeta) = \sum_1^n N_i(\xi, \eta, \zeta) \underset{(1x3)}{\{P_i\}} \Rightarrow$$

$$\underset{(3x3)}{[J]} = \underset{(3x3)}{\begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta} \\ \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} & \frac{\partial z}{\partial \zeta} \end{bmatrix}} = \underset{(3x3)}{\begin{Bmatrix} \underset{(1x3)}{\{S_\xi(\xi, \eta, \zeta)\}} \\ \underset{(1x3)}{\{S_\eta(\xi, \eta, \zeta)\}} \\ \underset{(1x3)}{\{S_\zeta(\xi, \eta, \zeta)\}} \end{Bmatrix}} = \underset{(3x3)}{\begin{bmatrix} \left\{ \sum_1^n N_{\xi,i}(\xi, \eta, \zeta) \underset{(1x3)}{\{P_i\}} \right\} \\ \left\{ \sum_1^n N_{\eta,i}(\xi, \eta, \zeta) \underset{(1x3)}{\{P_i\}} \right\} \\ \left\{ \sum_1^n N_{\zeta,i}(\xi, \eta, \zeta) \underset{(1x3)}{\{P_i\}} \right\} \end{bmatrix}} = \underset{(3xn)}{\begin{Bmatrix} \underset{(nx1)}{\{N_\xi\}^T} \\ \underset{(nx1)}{\{N_\eta\}^T} \\ \underset{(nx1)}{\{N_\zeta\}^T} \end{Bmatrix}} \underset{(nx3)}{\{P\}}$$

In the same way we can get the Jacobian matrix for 2D and 1D elasticity.

Presenting all the cases along with the inverse Jacobian matrix, which we will use later:

1D Elasticity	2D Elasticity	3D Elasticity
$\underset{(1x1)}{[J]} = \underset{(1x1)}{\left[\frac{dx}{d\xi}\right]} = \underset{(1xn)}{\{N_\xi\}^T} \underset{(nx1)}{\{P\}}$	$\underset{(2x2)}{[J]} = \underset{(2x2)}{\begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix}} = \underset{(2xn)}{\begin{Bmatrix} \{N_\xi\}^T \\ \{N_\eta\}^T \end{Bmatrix}} \underset{(nx2)}{\{P\}}$	$\underset{(3x3)}{[J]} = \underset{(3x3)}{\begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta} \\ \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} & \frac{\partial z}{\partial \zeta} \end{bmatrix}} = \underset{(3xn)}{\begin{Bmatrix} \{N_\xi\}^T \\ \{N_\eta\}^T \\ \{N_\zeta\}^T \end{Bmatrix}} \underset{(nx3)}{\{P\}}$
$\underset{(1x1)}{[J]^{-1}} = \frac{1}{\underset{(1x1)}{[J]}} = \frac{1}{\det([J])}$	$\underset{(2x2)}{[J]^{-1}} = \begin{bmatrix} J^*_{11} & J^*_{12} \\ J^*_{21} & J^*_{22} \end{bmatrix} = \frac{1}{\det[J]} \begin{bmatrix} J_{22} & -J_{12} \\ -J_{21} & J_{11} \end{bmatrix}$ where $\det[J] = J_{11}J_{22} - J_{21}J_{12}$	$\underset{(3x3)}{[J]^{-1}} = \begin{bmatrix} J^*_{11} & J^*_{12} & J^*_{13} \\ J^*_{21} & J^*_{22} & J^*_{23} \\ J^*_{31} & J^*_{32} & J^*_{33} \end{bmatrix}$

Figure 3.5. Jacobian and Jacobian Inverse Matrix in each case of Elasticity.

Where $\underset{(1x3)}{\{P_i\}} = \underset{(1x3)}{\{X_i \quad Y_i \quad Z_i\}}$ are the i[th] control point's Cartesian coordinates and

$\underset{(nx1)}{\{N_\xi\}} = \frac{\partial \underset{(nx1)}{\{N\}}}{\partial \xi}$. Similarly for $\underset{(nx1)}{\{N_\eta\}}, \underset{(nx1)}{\{N_\zeta\}}$

3.2 Stiffness Matrix 1D

One dimensional problems are mostly of academic interest as they are only used in truss systems with axial loading. They serve well, nonetheless, when we first try to understand a new concept or as a step towards the more general and difficult cases of 2D and 3D elasticity.

In the 1D case, only axial deformation for each point of the truss exists. The displacement is $u(x) = u(C(\xi))$. The respective strain matrix is:

$$\underset{(1x1)}{\{\varepsilon\}} = [\varepsilon_x] = \left[\frac{\partial u}{\partial x}\right]$$

In Finite Element Analysis, we want to integrate throughout the volume/surface/length of the model. As this is hard to do in Physical space we change the basis of space, transitioning to the Parameter space. In the parameter space the integration is rather easy. We need the

$$\left[\frac{\partial \varphi}{\partial x}\right] = [J]^{-1}\left[\frac{\partial \varphi}{\partial \xi}\right].$$

$$\frac{\partial u}{\partial \xi} = \frac{\partial u}{\partial x}\frac{\partial x}{\partial \xi} = \frac{\partial u}{\partial x}\underset{(1x1)}{[J]} \Rightarrow \frac{\partial u}{\partial x} = \underset{(1x1)}{[J]^{-1}}\frac{\partial u}{\partial \xi} = \underset{(1x1)}{\frac{1}{[J]}}\frac{\partial u}{\partial \xi} \Rightarrow \underset{(1x1)}{\{\varepsilon_x\}} = \underset{(1x1)}{[B_1]}\underset{(1x1)}{\{u_\xi\}} \Rightarrow$$

$$\underset{(1x1)}{\{\varepsilon_x\}} = \underset{(1x1)}{[B_1]}\underset{(1x1)}{\{u_\xi\}}$$

where the **Deformation Matrix [B_1]:**

$$\underset{(1x1)}{[B_1]} = \left[\frac{1}{J_{11}}\right]$$

$$\underset{(1x1)}{\{u_\xi\}} = \underset{(1x1)}{\left[\frac{\partial u}{\partial \xi}\right]} = \underset{(1x1)}{\left[\frac{\partial\left(\sum_1^n R_i u_i\right)}{\partial \xi}\right]} = \underset{(1xn)}{\begin{bmatrix} R_{1,\xi} & R_{2,\xi} & \cdots & \cdots & R_{n,\xi} \end{bmatrix}}\underset{(nx1)}{\begin{bmatrix} u_1 & u_2 & \cdots & \cdots & u_5 \end{bmatrix}^T} = \underset{(1xn)}{\{R_{,\xi}\}^T}\underset{(nx1)}{\{d\}} \Rightarrow$$

$$\underset{(1x1)}{[u_\xi]} = \underset{(1xn)}{[B_2(\xi)]}\underset{(nx1)}{\{d\}}$$

where the **Deformation Matrix [B_2]:**

$$\underset{(1xn)}{[B_2(\xi)]} = \underset{(1xn)}{\{R_{,\xi}\}^T}$$

In this way, the **Deformation Matrix** [**B**] is evaluated

$$\underset{(1xn)}{[B(\xi)]} = \underset{(1x1)}{[B_1(\xi)]}\underset{(1xn)}{[B_2(\xi)]}$$

and we are able to express strain values anywhere in the model as a function of control points' displacements.

Then, the **Stiffness Matrix** [**K**] is evaluated as

$$\underset{(nxn)}{[K]} = \int_1 \underset{(nx1)}{[B(\xi)]^T}\underset{(1x1)}{[E]}\underset{(1xn)}{[B(\xi)]}A dx = \int_{\xi_1}^{\xi_{n+p+1}} \underset{(nx1)}{[B(\xi)]^T}\underset{(1x1)}{[E]}\underset{(1xn)}{[B(\xi)]}A \det\left([J(\xi)]\right) d\xi$$

But analytical integration is almost never applicable in finite elements and we generally proceed with numerical integration. In the case of Gauss Numerical Integration, we have to calculate the function at all the gauss points in the patch.

Finally,

$$\underset{(nxn)}{[K]} = \sum_{i=1}^{n_{GP\xi}} \left(\underset{(nx1)}{[B(\xi_i)]^T}\underset{(1x1)}{[E]}\underset{(1xn)}{[B(\xi_i)]}A \det\left([J(\xi_i)]\right) w_i^{GP} \right)$$

where

ξ_i: the parametric coordinate of the i^{th} gauss point

$n_{GP\xi}$: the total number of gauss points in the patch

w_i^{GP}: the weight of the i^{th} gauss point.

A: the area of the cross section

3.3 Stiffness Matrix 2D

In 2D elasticity the Physical space is a plane with axes (x,y) and the Parameter space a plane with axes (ξ,η). Though we have one more dimension to take into consideration, the basic logic and steps are the same as in 1D elasticity.

$$\underset{(3x1)}{\{\varepsilon\}} = \begin{bmatrix} \varepsilon_x \\ \varepsilon_y \\ \gamma_{xy} \end{bmatrix} = \begin{bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial v}{\partial y} \\ \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

In the normal mapping from Parameter to Physical

$$\begin{bmatrix} \frac{\partial}{\partial \xi} \\ \frac{\partial}{\partial \eta} \end{bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} \begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{bmatrix} = \underset{(2x2)}{[J]} \begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{bmatrix}$$

and in the inverse mapping (Physical->Parameter)

$$\begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{bmatrix} = \underset{(2x2)}{[J]^{-1}} \begin{bmatrix} \frac{\partial}{\partial \xi} \\ \frac{\partial}{\partial \eta} \end{bmatrix}$$

Substituting $\frac{\partial}{\partial x}$ and $\frac{\partial}{\partial y}$ we get the deformations in physical space as a function of deformations in parameter space:

$$\underset{(3x1)}{\{\varepsilon\}} = \underbrace{\frac{1}{\det\left(\underset{(2x2)}{[J]}\right)} \underbrace{\begin{bmatrix} J_{22} & -J_{12} & 0 & 0 \\ 0 & 0 & -J_{21} & J_{11} \\ -J_{21} & J_{11} & J_{22} & -J_{12} \end{bmatrix}}_{(3x4)}}_{\underset{(3x4)}{[B_1(\xi,\eta)]}} \underset{(4x1)}{\begin{bmatrix} \frac{\partial u}{\partial \xi} \\ \frac{\partial u}{\partial \eta} \\ \frac{\partial v}{\partial \xi} \\ \frac{\partial v}{\partial \eta} \end{bmatrix}} = \underset{(3x4)}{[B_1(\xi,\eta)]} \underset{(4x1)}{\begin{bmatrix} u_\xi \\ u_\eta \\ v_\xi \\ v_\eta \end{bmatrix}} \Rightarrow$$

$$\underset{(3x1)}{\{\varepsilon\}} = \underset{(3x4)}{[B_1(\xi,\eta)]} \underset{(4x1)}{\begin{bmatrix} u_\xi \\ u_\eta \\ v_\xi \\ v_\eta \end{bmatrix}}$$

where the **Deformation Matrix [B_1]:**

$$\underset{(3x4)}{[B_1(\xi,\eta)]} = \frac{1}{\det\left(\underset{(2x2)}{[J]}\right)} \underset{(3x4)}{\begin{bmatrix} J_{22} & -J_{12} & 0 & 0 \\ 0 & 0 & -J_{21} & J_{11} \\ -J_{21} & J_{11} & J_{22} & -J_{12} \end{bmatrix}}$$

The deformations in parameter space as a function of control points' displacements:

$$\underset{(4x1)}{\begin{bmatrix} u_\xi \\ u_\eta \\ v_\xi \\ v_\eta \end{bmatrix}} = \underset{(4x1)}{\begin{bmatrix} \frac{\partial u}{\partial \xi} \\ \frac{\partial u}{\partial \eta} \\ \frac{\partial v}{\partial \xi} \\ \frac{\partial v}{\partial \eta} \end{bmatrix}} = \underset{(4x1)}{\begin{bmatrix} \sum_{i=1}^{n} R_{i,\xi} u_i \\ \sum_{i=1}^{n} R_{i,\eta} u_i \\ \sum_{i=1}^{n} R_{i,\xi} v_i \\ \sum_{i=1}^{n} R_{i,\eta} v_i \end{bmatrix}} = \underbrace{\underset{(4xN)}{\begin{bmatrix} R_{1,\xi} & 0 & R_{2,\xi} & 0 & \cdots & \cdots & R_{N,\xi} & 0 \\ R_{1,\eta} & 0 & R_{2,\eta} & 0 & \cdots & \cdots & R_{N,\eta} & 0 \\ 0 & R_{1,\xi} & 0 & R_{2,\xi} & \cdots & \cdots & 0 & R_{N,\xi} \\ 0 & R_{1,\eta} & 0 & R_{2,\eta} & \cdots & \cdots & 0 & R_{N,\eta} \end{bmatrix}}}_{\underset{(4x2N)}{[B_2(\xi,\eta)]}} \underset{(2Nx1)}{\begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ \vdots \\ \vdots \\ u_N \\ v_N \end{bmatrix}} \Rightarrow$$

$$\underset{(4x1)}{\begin{bmatrix} u_\xi \\ u_\eta \\ v_\xi \\ v_\eta \end{bmatrix}} = \underset{(4x2N)}{[B_2(\xi,\eta)]} \underset{(2Nx1)}{\{d\}}$$

where the **Deformation Matrix [B_2]:**

$$\underset{(4x2N)}{[B_2(\xi,\eta)]} = \underset{(4x2N)}{\begin{bmatrix} R_{1,\xi} & 0 & R_{2,\xi} & 0 & \cdots & \cdots & R_{N,\xi} & 0 \\ R_{1,\eta} & 0 & R_{2,\eta} & 0 & \cdots & \cdots & R_{N,\eta} & 0 \\ 0 & R_{1,\xi} & 0 & R_{2,\xi} & \cdots & \cdots & 0 & R_{N,\xi} \\ 0 & R_{1,\eta} & 0 & R_{2,\eta} & \cdots & \cdots & 0 & R_{N,\eta} \end{bmatrix}}$$

In this way, the **Deformation Matrix [B]** is evaluated

$$\underset{(3x2N)}{[B(\xi,\eta)]} = \underset{(3x4)}{[B_1(\xi,\eta)]} \underset{(4x2N)}{[B_2(\xi,\eta)]}$$

and we are able to express strain values anywhere in the model as a function of control points' displacements.

Then, the **Stiffness Matrix** [**K**] is evaluated as

$$\underset{(2Nx2N)}{[K]} = \int_{\xi_1}^{\xi_{n+p+1}} \int_{\eta_1}^{\eta_{m+q+1}} \underset{(2Nx3)}{[B(\xi,\eta)]^T} \underset{(3x3)}{[E]} \underset{(3x2N)}{[B(\xi,\eta)]} t \det\left([J(\xi,\eta)]\right) d\xi d\eta$$

The gauss points are full tensor product over the parameter space, that way the numerical integration is performed in the following way.

$$\underset{(2Nx2N)}{[K]} = \sum_{i=1}^{n_{GP\xi}} \sum_{j=1}^{n_{GP\eta}} \left(\underset{(2Nx3)}{\left[B(\xi_i,\eta_j)\right]^T} \underset{(3x3)}{[E]} \underset{(3x2N)}{\left[B(\xi_i,\eta_j)\right]} t \det\left(\left[J(\xi_i,\eta_j\right]\right) w_i^{GP\xi} w_j^{GP\eta} \right)$$

where

ξ_i, η_j: the parametric coordinates of the tensor product gauss point i, j.

$n_{GP\xi}$, $n_{GP\eta}$: the number of gauss points along the axes ξ and η respectively, in the patch.

$w_i^{GP\xi}$, $w_j^{GP\eta}$: the tensor product weights of the gauss point i,j.

t: the thickness of the cross section

3.4 Stiffness Matrix 3D

3D elasticity is the general case.

$$\underset{(3x1)}{\{\varepsilon\}} = \begin{bmatrix} \varepsilon_x \\ \varepsilon_y \\ \varepsilon_z \\ \gamma_{xy} \\ \gamma_{yz} \\ \gamma_{zx} \end{bmatrix} = \begin{bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial v}{\partial y} \\ \frac{\partial w}{\partial z} \\ \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \\ \frac{\partial v}{\partial z} + \frac{\partial w}{\partial y} \\ \frac{\partial w}{\partial x} + \frac{\partial u}{\partial z} \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix}$$

In the straight mapping (Parameter->Physical)

$$\begin{bmatrix} \frac{\partial}{\partial \xi} \\ \frac{\partial}{\partial \eta} \\ \frac{\partial}{\partial \zeta} \end{bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta} \\ \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} & \frac{\partial z}{\partial \zeta} \end{bmatrix} \begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{bmatrix} = \underset{(3x3)}{[J]} \begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{bmatrix}$$

and in the inverse mapping (Physical->Parameter)

$$\begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{bmatrix} = \underset{(3x3)}{[J]^{-1}} \begin{bmatrix} \frac{\partial}{\partial \xi} \\ \frac{\partial}{\partial \eta} \\ \frac{\partial}{\partial \zeta} \end{bmatrix} \text{ and } \underset{(3x3)}{[J]^{-1}} = \begin{bmatrix} J^*_{11} & J^*_{12} & J^*_{13} \\ J^*_{21} & J^*_{22} & J^*_{23} \\ J^*_{31} & J^*_{32} & J^*_{33} \end{bmatrix}$$

Substituting $\frac{\partial}{\partial x}$, $\frac{\partial}{\partial y}$ and $\frac{\partial}{\partial z}$ we get the deformations in physical space as a function of deformations in parameter space:

$$\underset{(6x1)}{\{\varepsilon\}} = \begin{bmatrix} \varepsilon_x \\ \varepsilon_y \\ \varepsilon_z \\ \gamma_{xy} \\ \gamma_{yz} \\ \gamma_{zx} \end{bmatrix} = \begin{bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial v}{\partial y} \\ \frac{\partial w}{\partial z} \\ \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \\ \frac{\partial v}{\partial z} + \frac{\partial w}{\partial y} \\ \frac{\partial w}{\partial x} + \frac{\partial u}{\partial z} \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \underbrace{\begin{bmatrix} J^*_{11} & J^*_{12} & J^*_{13} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & J^*_{21} & J^*_{22} & J^*_{23} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & J^*_{31} & J^*_{32} & J^*_{33} \\ J^*_{21} & J^*_{22} & J^*_{23} & J^*_{11} & J^*_{12} & J^*_{13} & 0 & 0 & 0 \\ 0 & 0 & 0 & J^*_{31} & J^*_{32} & J^*_{33} & J^*_{21} & J^*_{22} & J^*_{23} \\ J^*_{31} & J^*_{32} & J^*_{33} & 0 & 0 & 0 & J^*_{11} & J^*_{12} & J^*_{13} \end{bmatrix}}_{\underset{(6x9)}{\left[B_{1(\xi,\eta,\zeta)}\right]}} \underbrace{\begin{bmatrix} \frac{\partial u}{\partial \xi} \\ \frac{\partial v}{\partial \xi} \\ \frac{\partial w}{\partial \xi} \\ \frac{\partial u}{\partial \eta} \\ \frac{\partial v}{\partial \eta} \\ \frac{\partial w}{\partial \eta} \\ \frac{\partial u}{\partial \zeta} \\ \frac{\partial v}{\partial \zeta} \\ \frac{\partial w}{\partial \zeta} \end{bmatrix}}_{(9x1)} \Rightarrow$$

$$\underset{(6x1)}{\{\varepsilon\}} = \underset{(6x9)}{\left[B_{1(\xi,\eta,\zeta)}\right]} \underset{(9x1)}{\begin{bmatrix} u_\xi & v_\xi & w_\xi & u_\eta & v_\eta & w_\eta & u_\zeta & v_\zeta & w_\zeta \end{bmatrix}^T}$$

where the **Deformation Matrix [B_1]**:

$$\underset{(6x9)}{\left[B_{1(\xi,\eta,\zeta)}\right]} = \begin{bmatrix} J^*_{11} & J^*_{12} & J^*_{13} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & J^*_{21} & J^*_{22} & J^*_{23} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & J^*_{31} & J^*_{32} & J^*_{33} \\ J^*_{21} & J^*_{22} & J^*_{23} & J^*_{11} & J^*_{12} & J^*_{13} & 0 & 0 & 0 \\ 0 & 0 & 0 & J^*_{31} & J^*_{32} & J^*_{33} & J^*_{21} & J^*_{22} & J^*_{23} \\ J^*_{31} & J^*_{32} & J^*_{33} & 0 & 0 & 0 & J^*_{11} & J^*_{12} & J^*_{13} \end{bmatrix}$$

and

$$\underset{(3x3)}{\left[J\right]^{-1}} = \begin{bmatrix} J^*_{11} & J^*_{12} & J^*_{13} \\ J^*_{21} & J^*_{22} & J^*_{23} \\ J^*_{31} & J^*_{32} & J^*_{33} \end{bmatrix}$$

The deformations are given by:

$$\underset{(9x1)}{\begin{bmatrix} u_\xi \\ v_\xi \\ w_\xi \\ u_\eta \\ v_\eta \\ w_\eta \\ u_\zeta \\ v_\zeta \\ w_\zeta \end{bmatrix}} = \underset{(9x1)}{\begin{bmatrix} \sum_{i=1}^{N} R_{i,\xi} u_i \\ \sum_{i=1}^{N} R_{i,\xi} v_i \\ \sum_{i=1}^{N} R_{i,\xi} w_i \\ \sum_{i=1}^{N} R_{i,\eta} u_i \\ \sum_{i=1}^{N} R_{i,\eta} v_i \\ \sum_{i=1}^{N} R_{i,\eta} w_i \\ \sum_{i=1}^{N} R_{i,\zeta} u_i \\ \sum_{i=1}^{N} R_{i,\zeta} v_i \\ \sum_{i=1}^{N} R_{i,\zeta} w_i \end{bmatrix}} = \underbrace{\underset{(9x3N)}{\begin{bmatrix} R_{1,\xi} & 0 & 0 & R_{2,\xi} & 0 & 0 & \cdots & R_{n,\xi} & 0 & 0 \\ R_{1,\eta} & 0 & 0 & R_{2,\eta} & 0 & 0 & \cdots & R_{n,\eta} & 0 & 0 \\ R_{1,\zeta} & 0 & 0 & R_{2,\zeta} & 0 & 0 & \cdots & R_{n,\zeta} & 0 & 0 \\ 0 & R_{1,\xi} & 0 & 0 & R_{2,\xi} & 0 & \cdots & 0 & R_{n,\xi} & 0 \\ 0 & R_{1,\eta} & 0 & 0 & R_{2,\eta} & 0 & \cdots & 0 & R_{n,\eta} & 0 \\ 0 & R_{1,\zeta} & 0 & 0 & R_{2,\zeta} & 0 & \cdots & 0 & R_{n,\zeta} & 0 \\ 0 & 0 & R_{1,\xi} & 0 & 0 & R_{2,\xi} & \cdots & 0 & 0 & R_{n,\xi} \\ 0 & 0 & R_{1,\eta} & 0 & 0 & R_{2,\eta} & \cdots & 0 & 0 & R_{n,\eta} \\ 0 & 0 & R_{1,\zeta} & 0 & 0 & R_{2,\zeta} & \cdots & 0 & 0 & R_{n,\zeta} \end{bmatrix}}}_{[B_{2(\xi,\eta,\zeta)}]_{(9x3n)}} \underset{(3Nx1)}{\begin{bmatrix} u_1 \\ v_1 \\ w_1 \\ u_2 \\ v_2 \\ w_2 \\ \vdots \\ \vdots \\ u_N \\ v_N \\ w_N \end{bmatrix}} \Rightarrow$$

$$\underset{(9x1)}{\begin{bmatrix} u_\xi & v_\xi & w_\xi & u_\eta & v_\eta & w_\eta & u_\zeta & v_\zeta & w_\zeta \end{bmatrix}^T} = \underset{(9x3N)}{[B_{2(\xi,\eta,\zeta)}]} \underset{(3Nx1)}{\{d\}}$$

where **Deformation Matrix [B₂]**:

$$\underset{(9x3N)}{[B_{2(\xi,\eta,\zeta)}]} = \underset{(9x3N)}{\begin{bmatrix} R_{1,\xi} & 0 & 0 & R_{2,\xi} & 0 & 0 & \cdots & R_{n,\xi} & 0 & 0 \\ R_{1,\eta} & 0 & 0 & R_{2,\eta} & 0 & 0 & \cdots & R_{n,\eta} & 0 & 0 \\ R_{1,\zeta} & 0 & 0 & R_{2,\zeta} & 0 & 0 & \cdots & R_{n,\zeta} & 0 & 0 \\ 0 & R_{1,\xi} & 0 & 0 & R_{2,\xi} & 0 & \cdots & 0 & R_{n,\xi} & 0 \\ 0 & R_{1,\eta} & 0 & 0 & R_{2,\eta} & 0 & \cdots & 0 & R_{n,\eta} & 0 \\ 0 & R_{1,\zeta} & 0 & 0 & R_{2,\zeta} & 0 & \cdots & 0 & R_{n,\zeta} & 0 \\ 0 & 0 & R_{1,\xi} & 0 & 0 & R_{2,\xi} & \cdots & 0 & 0 & R_{n,\xi} \\ 0 & 0 & R_{1,\eta} & 0 & 0 & R_{2,\eta} & \cdots & 0 & 0 & R_{n,\eta} \\ 0 & 0 & R_{1,\zeta} & 0 & 0 & R_{2,\zeta} & \cdots & 0 & 0 & R_{n,\zeta} \end{bmatrix}}$$

In this way, the deformation matrix is evaluated

$$\underset{(6x3N)}{[B(\xi,\eta,\zeta)]} = \underset{(6x9)}{[B_1(\xi,\eta,\zeta)]} \underset{(9x3N)}{[B_2(\xi,\eta,\zeta)]}$$

and we are able to express strain values anywhere in the model as a function of control points' displacements.

Then, the **Stiffness Matrix** [K] is evaluated as

$$\underset{(3Nx3N)}{[K]} = \int_{\xi_1}^{\xi_{n+p+1}} \int_{\eta_1}^{\eta_{m+q+1}} \int_{\zeta_1}^{\zeta_{l+r+1}} \underset{(3Nx6)}{[B(\xi,\eta,\zeta)]^T} \underset{(6x6)}{[E]} \underset{(6x3N)}{[B(\xi,\eta,\zeta)]} \det\left([J(\xi,\eta,\zeta)]\right) d\xi d\eta d\zeta$$

The gauss points are full tensor product over the parameter space, that way the numerical integration is performed in the following way.

$$\underset{(3Nx3N)}{[K]} = \sum_{i=1}^{n_{GP\xi}} \sum_{j=1}^{n_{GP\eta}} \sum_{k=1}^{n_{GP\zeta}} \left(\underset{(3Nx6)}{\left[B(\xi_i,\eta_j,\zeta_k)\right]^T} \underset{(6x6)}{[E]} \underset{(6x3N)}{\left[B(\xi_i,\eta_j,\zeta_k)\right]} \det\left(\left[J(\xi_i,\eta_j,\zeta_k\right]\right) w_i^{GP\xi} w_j^{GP\eta} w_k^{GP\zeta} \right)$$

where

ξ_i, η_j,ζ_k: the parametric coordinates of the tensor product gauss point i, j, k.

$n_{GP\xi}$, $n_{GP\eta}$, $n_{GP\zeta}$: the number of gauss points along the axes ξ, η and ζ respectively, in the patch.

$w_i^{GP\xi}, w_j^{GP\eta}, w_k^{GP\zeta}$: the tensor product weights of the gauss point i, j, k.

We now present a flowchart of the general architecture of an Isogeometric Analysis Code.

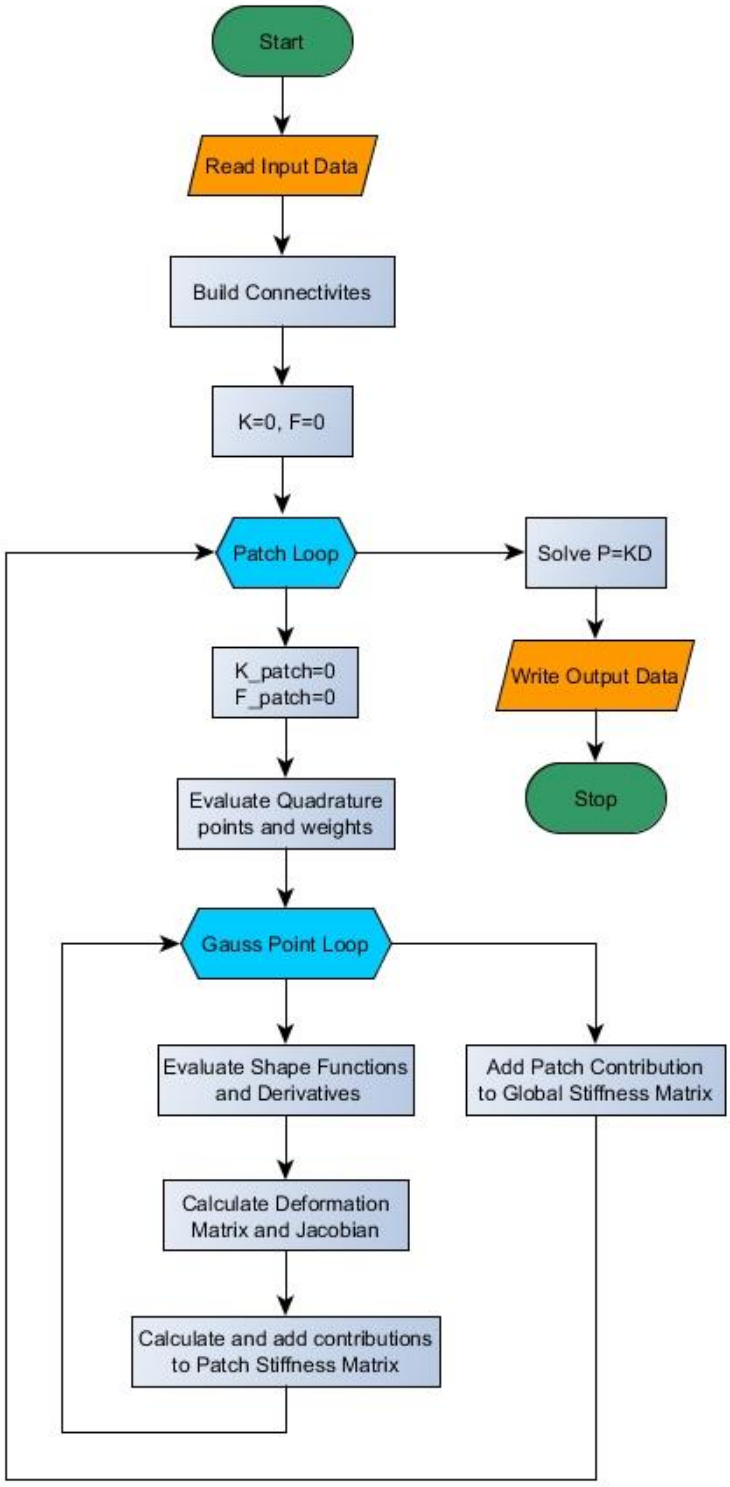


Figure 3.6. Architecture of an Isogeometric Analysis Code in the form of a flowchart.

4 External Loads Boundary Conditions Solution Field

4.1 External Loads

The loading upon the model may vary; it can be concentrated, distributed over a surface or distributed over the volume of the model like the self-weight. But we have to solve a discrete equation where all loading is applied on the nodes, or, in IGA, on the Control Points. We will now assemble the equivalent load vector $\underset{(Nx1)}{\{F\}}$.

4.1.1 Concentrated loads

Unlike Finite Element Analysis, where the degrees of freedom are actually on the model and on the nodes the shape functions are interpolatory, in Isogeometric Analysis neither is that way. In every point several shape functions are not zero. Therefore we cannot assign in general concentrated loads directly on the degrees of freedom even if they are applied on a point whose parametric coordinates coincide with the coordinates of a control point. They have to be transformed into equivalent loads through their multiplication with shape functions values on the point of application.

There is one case though where we can directly assign them to the degrees of freedom. At the edge of the patch we have C^{-1} continuity and the control points lay on the model in physical space. If the loads are applied on the edge of the patch, meaning on the first or last control point in both parametric axes, and they are tangent to one of the parametric axes in physical space, then we can assign it to the respective degree of freedom. For example, let us say that we have a 2D problem with a concentrated load on the last control point on axis ξ and first in η. If the load is tangent to the axis η in physical space, then we can directly assign it on that control point on degree freedom η.

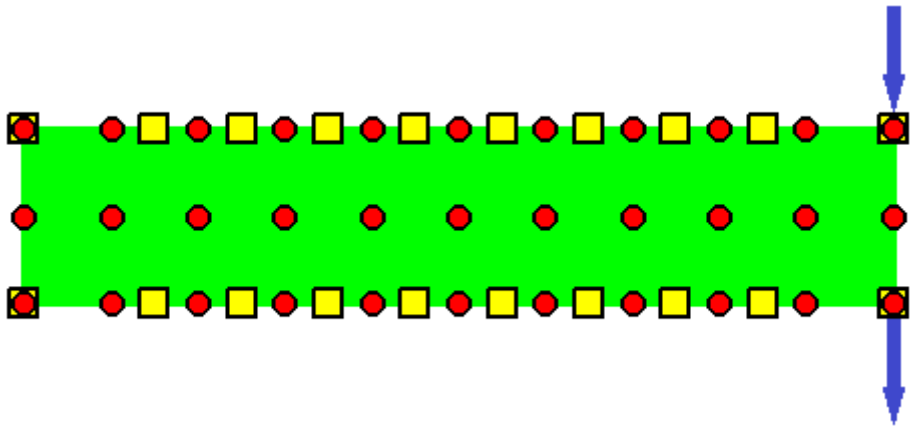


Figure 4.1. Physical Space. Cantilever 2D with concentrated loads on the two right edge corner control points. All but one shape function are zero and thus the loads applied in the corner material point are directly transferred to the control point.

This can be easily verified. The first and last control point in each parametric axis have the same parametric coordinates as the first and last knot. Thus only one Basis function in each axis is one and all others zero, resulting in only one NURBS Shape function being one and all others zero. That way the whole load is assigned to that control point on its respective degree of freedom.

4.1.2 Distributed loads

In a similar way we have to compute the equivalent control point loads for all distributed loads. Assuming that the function:

$$\left[\mathrm{LoadX}(\xi,\eta,\zeta),\mathrm{LoadY}(\xi,\eta,\zeta),\mathrm{LoadZ}(\xi,\eta,\zeta)\right] = \underset{(1x3)}{f(x,y,z)} = \underset{(1x3)}{f(\xi,\eta,\zeta)}$$

describes the load distribution at any point (x,y,z) or in the parameter space (ξ,η,ζ), the equivalent loads will be computed as:

$$\underset{(Nx3)}{\{F\}} = \int_V \underset{(Nx1)}{\{R(x,y,z)\}}\, \underset{(1x3)}{f(x,y,z)}\, dV = \int_{\xi_1}^{\xi_{n+p+1}} \int_{\eta_1}^{\eta_{m+q+1}} \int_{\zeta_1}^{\zeta_{l+r+1}} \underset{(Nx1)}{\{R(\xi,\eta,\zeta)\}}\, \underset{(1x3)}{f(\xi,\eta,\zeta)} \det[J]\, d\zeta d\eta d\xi$$

and in detail for each case of elasticity:

1D Elasticity:
$$\underset{(nx1)}{\{F\}} = \int_{\xi_1}^{\xi_{n+p+1}} \underset{(nx1)}{\{R(\xi,\eta,\zeta)\}}\, \underset{(1x1)}{f(\xi,\eta,\zeta)} \det[J]\, d\xi$$

2D Elasticity:
$$\underset{(Nx2)}{\{F\}} = \int_{\xi_1}^{\xi_{n+p+1}} \int_{\eta_1}^{\eta_{m+q+1}} \underset{(Nx1)}{\{R(\xi,\eta,\zeta)\}}\, \underset{(1x2)}{f(\xi,\eta,\zeta)} \det[J]\, d\eta d\xi$$

3D Elasticity:
$$\underset{(Nx3)}{\{F\}} = \int_{\xi_1}^{\xi_{n+p+1}} \int_{\eta_1}^{\eta_{m+q+1}} \int_{\zeta_1}^{\zeta_{l+r+1}} \underset{(Nx1)}{\{R(\xi,\eta,\zeta)\}}\, \underset{(1x3)}{f(\xi,\eta,\zeta)} \det[J]\, d\zeta d\eta d\xi$$

4.2 Boundary Conditions

Depending on the differential equation we want to solve, it is possible to have a variety of boundary conditions on the edges, Dirichlet, Neumann or other. In Computational Mechanics, boundary conditions in strong form are generally not suitable for implementation. We usually transform them to a weak form and then discretize in a finite number of points in the mesh; in IGA on the control points. In FEM it is relatively simple to decide which nodes are to be constrained because they coincide with material points. In IGA, due to the greater overlapping between the control points' domains of influence and the non interpolatory to the geometry character of control points, displacements of control points, generally, do not represent displacements of a certain material point. That said, the enforcement of boundary conditions in the general case is harder.

However, there are simple cases where the implementation of boundary conditions is straightforward. For example, when dealing with a 2D shape with straight sides, the control points are on the geometry. The displacements on a side of such a model are only dependent on the displacements of the control points on that side. Thus, we can easily constrain a side of the model by constraining all the control points on those sides.

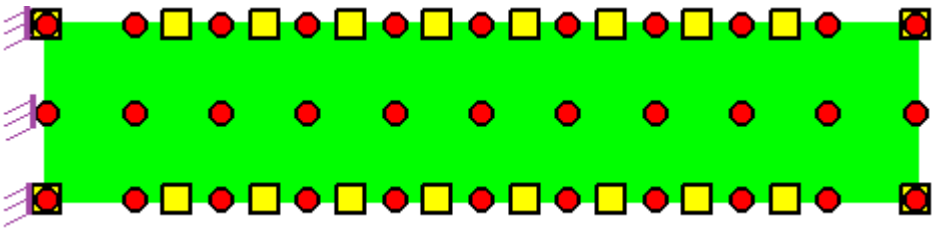


Figure 4.2. The left edge control points are constrained in both directions X,Y. That way, the whole left side is constrained.

In its simplest form, the boundary conditions are Dirichlet, fixed displacements on a part of the model. Those degrees of freedom are called stationary and the corresponding rows and columns are deleted from the Stiffness Matrix and the Load Vector. Then we can compute the remaining unknown displacements by solving the linear system:

$$\{L_f\} = [K_{ff}]\{D_f\} \Rightarrow \{D_f\} = [K_{ff}]^{-1}\{L_f\}$$

The fixed zero displacements $\{D_s\}$ of the fixed degrees of freedom are added back to the result, thus forming the Displacement Vector $\{D\} = \begin{Bmatrix}\{D_f\}\\ \{D_s\}\end{Bmatrix}$. The constraint's loads are evaluated as $\{L_s\} = [K_{sf}]\{D_f\}$ and then added back to the result, forming the Load Vector $\{L\} = \begin{Bmatrix}\{L_f\}\\ \{L_s\}\end{Bmatrix}$

4.3 Solution Field

4.3.1 Displacement Field

After solving the equation we get a vector $\{D\}$ of the displacements on Control Points. Again, if the shape functions were interpolatory at every control point then those displacements would actually have a physical meaning and correspond to an actual displacement of a material point on the model. Nonetheless, apart from the boundary knots, the shape functions are not interpolatory which means that each displacement of vector $\{D\}$ is just a part of the actual displacement at a material point ξ. In any point in parameter space (ξ,η,ζ), corresponding on a material point (x,y,z) on the model, we can get the displacement field as

$$\underset{(1xN)}{d(\xi,\eta,\zeta)} = \underset{(1xN)}{\{R(\xi,\eta,\zeta)\}^T}\ \underset{(Nx1)}{\{D\}}$$

1D Elasticity:

$$\underset{(1xn)}{d(\xi)} = \underset{(1xn)}{\{R(\xi)\}^T}\ \underset{(nx1)}{\{D\}}$$

2D Elasticity:

$$\underset{(1x2n)}{d(\xi,\eta)} = \underset{(1x2n)}{\{R(\xi,\eta)\}^T}\ \underset{(2nx1)}{\{D\}}$$

3D Elasticity:

$$\underset{(1x3n)}{d(\xi,\eta,\zeta)} = \underset{(1x3n)}{\{R(\xi,\eta,\zeta)\}^T}\ \underset{(3nx1)}{\{D\}}$$

4.3.2 Strain Field

By definition of the deformation matrix, the strain on any point ξ, η, ζ is given by the equation $\underset{(dimx1)}{\{\varepsilon\}} = \underset{(dimxN)}{[B]}\ \underset{(Nx1)}{\{D\}}$, where dim is the number of strains calculated.

For each of the three cases of linear elasticity, for n control points:

1D Elasticity: $\underset{(1x1)}{\{\varepsilon(\xi)\}} = \underset{(1xn)}{[B(\xi)]}\ \underset{(nx1)}{\{D\}}$

2D Elasticity: $\underset{(3x1)}{\{\varepsilon(\xi,\eta)\}} = \underset{(3x2n)}{[B(\xi,\eta)]}\ \underset{(2nx1)}{\{D\}}$

3D Elasticity: $\underset{(6x1)}{\{\varepsilon(\xi,\eta,\zeta)\}} = \underset{(6x3n)}{[B(\xi,\eta,\zeta)]}\ \underset{(3nx1)}{\{D\}}$

4.3.3 Stress Field

By definition again, elasticity matrix transforms the strains on a point to the respective stresses. $\underset{(\dim x1)}{\{\sigma\}} = \underset{(\dim x\dim)}{[E]}\ \underset{(\dim x1)}{\{\varepsilon\}} = \underset{(\dim x\dim)}{[E]}\ \underset{(\dim xN)}{[B]}\ \underset{(Nx1)}{\{D\}}$ where dim is the number of stresses and strains calculated in that case and N the number of the degrees of freedom.

In detail, for the three cases of linear elasticity:

1D Elasticity: $$\underset{(1x1)}{\{\sigma\}} = \underset{(1x1)}{[E]}\underset{(1x1)}{\{\varepsilon(\xi)\}} = \underset{(1x1)}{[E]}\underset{(1xn)}{[B(\xi)]}\underset{(nx1)}{\{D\}}$$

2D Elasticity: $$\underset{(3x1)}{\{\sigma\}} = \underset{(3x3)}{[E]}\underset{(3x1)}{\{\varepsilon(\xi,\eta)\}} = \underset{(3x3)}{[E]}\underset{(3x2n)}{[B(\xi,\eta)]}\underset{(2nx1)}{\{D\}}$$

3D Elasticity: $$\underset{(6x1)}{\{\sigma\}} = \underset{(6x6)}{[E]}\underset{(6x1)}{\{\varepsilon(\xi,\eta,\zeta)\}} = \underset{(6x6)}{[E]}\underset{(6x3n)}{[B(\xi,\eta,\zeta)]}\underset{(3nx1)}{\{D\}}$$

We notice that stresses and strains are expressed as a function of the deformation matrix [B] which uses the derivatives of the shape functions on the point we inspect. Ordinary Finite Elements use functions interpolatory at the end of each element which have continuity C^{-1} there. For that reason, they are not able to calculate stress and strain on those points from the above equations as they cannot define the shape functions' derivatives there. To overcome this problem other corrective procedures are deployed. In isogeometric analysis however, when using degree p continuity we have continuity C^{p-m} and the derivatives C^{p-m-1}. That way, with the right choice of the degree, such a problem will not arise. Using higher degree functions instead, we can have C^1 or higher continuity and have smooth and more natural approach of the actual stress and strain field.

5 Refinement

5.1 Introduction

B-Spline and NURBS geometries were initially developed to support the needs of the CAD industry. The evolution of CAD sets increasingly difficult goals and nowadays we are able to design complex shapes with the use of CAD shape functions. At first, when designing a new shape, a small number of control points able to exactly represent it was convenient, both for the designer and for the computational time. But as the designers target more complex shapes and the engineers try to approximate with better accuracy the solution, the need for flexibility of the model and thus interactive design is now imperative. Designers need a way to exactly represent geometry and, after the initial model representation, refine small details. In that way, we can initially draw the coarse shape of a face and later on define details such as eyes. This is the natural way of drawing and how modern designers work.

To accomplish that, we initially define the whole geometry we need to describe with as few shape functions as possible. Afterwards, we add the flexibility needed to enable us to interactively work on details which is equivalent to more control points describing the geometry in the area. More control points means that the meshing is denser and each control point has a smaller influence and a local character on defining the geometry of the area. For that, recall property 5 of the B-Spline curves. The designer can now change the Cartesian coordinates of the control points, thus conveniently working in a CAD environment in physical space, and affect only a small area of the model and represent details with accuracy.

The engineer would be seemingly content to have the initial model exactly represented with the minimum number of control points and less flexibility. That would enable them to finish their job seemingly sooner as less control points mean less degrees of freedom and thus much less computational time, which in several cases can require entire days if the project is complicated enough. On the contrary, less flexibility in the initial shape, even if we are not making use of it, means less capacity to represent accurately enough the analysis result field, displacements, loads and every result depending on those. In that new perspective, the engineer also needs flexibility, and in fact equal or more flexibility than the initial undeformed model requires.

That way, refinement in all its forms, is a great asset for shape functions, aiding both the CAD designer and the CAE analyst and thus proving its importance in both aspects of Isogeometric Analysis. In this chapter we will present in detail the different methods on how to successfully perform refinement.

5.2 Knot Value Insertion.

We recall, that in FEM, h-refinement is the technique which enriches the basis functions but keeps the same degree of the polynomials. In isogeometric analysis, knot value insertion or h-refinement enriches the basis and we get a new knot value vector $\bar{\Xi}$ by inserting knot values in the initial knot value vector Ξ. An inserted knot value may either be already present in which case we increase the multiplicity of the knot or it may be a itself a brand new knot. In any case though, it has to be a value internal to the initial knot value vector, preserving the boundary knot values' numerical content and multiplicity.

We consider our initial geometry accurate and with h-refinement we only want to add flexibility to the curve. The curve does not change geometrically or parametrically:

$$C^{I}_{(\xi)} = C^{F}_{(\xi)} \Rightarrow \underset{(1xn)}{\left\{R^{I}_{(\xi)}\right\}^{T}} \cdot \underset{(nx3)}{\left\{P^{I}\right\}} = \underset{(1xm)}{\left\{R^{F}_{(\xi)}\right\}^{T}} \cdot \underset{(mx3)}{\left\{P^{F}\right\}} \text{ with m>n.}$$

$\underset{(nx1)}{\left\{R^{I}_{(\xi)}\right\}}, \underset{(mx1)}{\left\{R^{F}_{(\xi)}\right\}}$ are the known initial and final basis functions of ξ and $\underset{(nx3)}{\left\{P^{I}\right\}}$, $\underset{(mx3)}{\left\{P^{F}\right\}}$ are the initial and final Cartesian coordinates (x,y,z) of the control points. We do not know the coordinates $\underset{(mx3)}{\left\{P^{F}\right\}}$ so to form a linear system we get m equations by giving m values to ξ within the limits of the Knot Vector. We take care, that every basis function is non zero at least at one of those values given to ξ. Due to the linear independence of the basis functions, we do know that the system has a unique solution giving us the new control points' cartesian coordinates $\underset{(mx3)}{\left\{P^{F}\right\}}$. In terms of matrix equation:

$$\underset{(mxn)}{\begin{bmatrix} \left\{R^{I}_{(\xi_1)}\right\}^{T} \\ \left\{R^{I}_{(\xi_2)}\right\}^{T} \\ \vdots \\ \left\{R^{I}_{(\xi_m)}\right\}^{T} \end{bmatrix}} \cdot \underset{(nx3)}{\left\{P^{I}\right\}} = \underset{(mxm)}{\begin{bmatrix} \left\{R^{F}_{(\xi_1)}\right\}^{T} \\ \left\{R^{F}_{(\xi_2)}\right\}^{T} \\ \vdots \\ \left\{R^{F}_{(\xi_m)}\right\}^{T} \end{bmatrix}} \cdot \underset{(mx3)}{\left\{P^{F}\right\}} \Rightarrow \underset{(mxn)}{\left[A^{I}\right]} \cdot \underset{(nx3)}{\left\{P^{I}\right\}} = \underset{(mxm)}{\left[A^{F}\right]} \cdot \underset{(mx3)}{\left\{P^{F}\right\}} \Rightarrow \underset{(mx3)}{\left\{P^{F}\right\}} = \underbrace{\underset{(mxm)}{\left[A^{F}\right]^{-1}} \underset{(mxn)}{\left[A^{I}\right]}}_{\underset{(mxn)}{\left[T^{FI}\right]}} \underset{(nx3)}{\left\{P^{I}\right\}}$$

The proven relation states that:

$$\underset{(mx3)}{\left\{P^{F}\right\}} = \underset{(mxn)}{\left[T^{FI}\right]} \underset{(nx3)}{\left\{P^{I}\right\}}$$

Cottrell, Hughes, Bazilevs [2] have automated this procedure of forming the transformation matrix $\underset{(mxn)}{\left[T^{FI}\right]}$ with the following algorithm.

Algorithm for Knot Value Insertion:

Given:

- the initial knot vector $\Xi = \{\xi_1, \xi_2, ..., \xi_{n+p+1}\}$
- the final knot vector $\bar{\Xi} = \{\bar{\xi}_1 = \xi_1, \bar{\xi}_2, ..., \bar{\xi}_{n+m+p+1} = \xi_{n+p+1}\}$

The transformation matrix $\underset{(mxn)}{[T^{FI}]}$ is built as:

$$T_{ij}^{0} = \begin{cases} 1, & \bar{\xi}_i \in [\xi_j, \xi_{j+1}) \\ 0, & \text{otherwise} \end{cases}$$

and

$$T_{ij}^{q} = \frac{\bar{\xi}_{i+q} - \xi_j}{\xi_{j+q} - \xi_j} T_{ij}^{q-1} + \frac{\xi_{j+q+1} - \bar{\xi}_{i+q}}{\xi_{j+q+1} - \xi_{j+1}} T_{ij+1}^{q-1}, \text{ for q=1,2,...,p}$$

When working on a surface or a solid, note that due to the tensor product nature of the shape functions, inserting a control point in axis ξ, is equal to inserting a whole set of control points in η and ζ. This is a major drawback of using traditional NURBS, not allowing local refinement. On the bright side, the tensor product nature gives us the option to perform refinement on every axis separately and in the order we prefer.

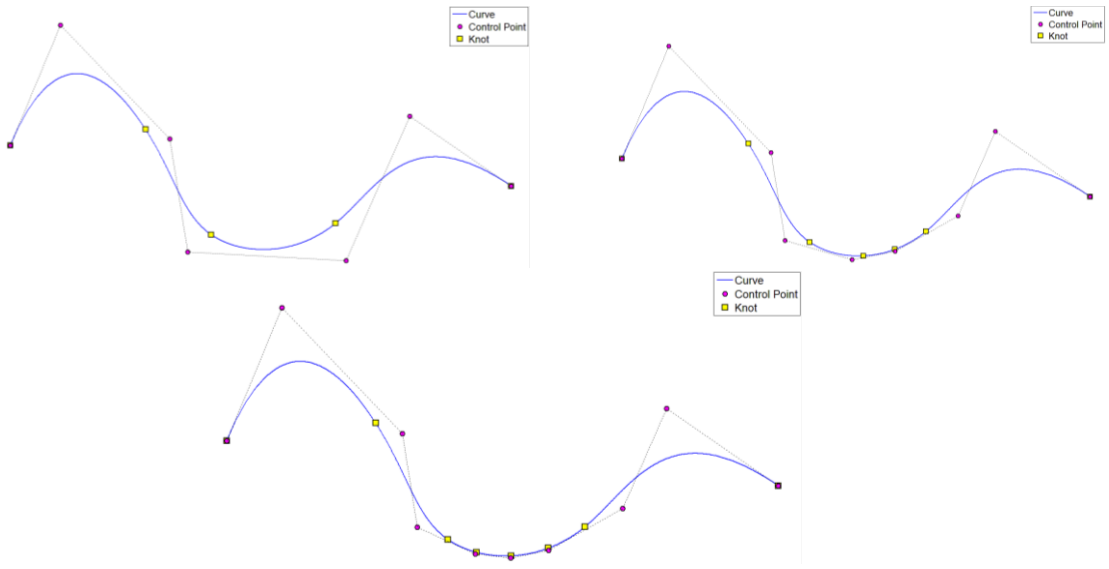


Figure 5.1. Consecutive knot insertions in a Curve.
Knot Value Vector Ξ_1 = {0 0 0 0 1 2 3 4 4 4 4}
Knot Value Vector Ξ_2 = {0 0 0 0 1 2 2.5 2.75 3 4 4 4 4}
Knot Value Vector Ξ_3 = {0 0 0 0 1 2 2.25 2.5 2.75 3 4 4 4 4}

5.3 Knot Value Removal

Knot removal or reverse h-refinement is very useful in many cases. The designer may need to express the curve accurately but with less control points than he originally estimated were needed or the engineer may want to decrease the number of degrees of freedom of its model if it is too time consuming. It is also necessary to change the curve's nature: for example, when transforming with knot insertion the B-Spline curve in a set of Bezier curves and then with knot removal the set of Bezier curves to a B-Spline curve again. This transition back and forth is very useful as there is a wide variety of algorithms designed for Bezier curves that do not apply in B-Splines.

Knot removal, in contrast with knot insertion, cannot be always implemented without changing the actual curve. Thus a knot value removal algorithm must do two things, firstly determine if a knot value is removable and how many times and secondly, if applicable, compute the new control points' Cartesian coordinates. There exist several efficient algorithms and the matter is thoroughly investigated in [14].

There are cases though when we know in advance which knot values can be removed. If we perform a knot refinement, then we do know that the knot values we inserted can be removed without losing accuracy in the curve representation. We here lay a handy tip for the transition back and forth from a coarse to a finer mesh.

In the standard h-refinement, the control points' Cartesian coordinates are transformed as follows:

$$\underset{(n_F x 1)}{\{P^F\}} = \underset{(n_F x n_C)}{[T^{FC}]} \underset{(n_C x 1)}{\{P^C\}}$$

We will attempt to build a transformation matrix $\underset{(n_C x n_F)}{[T^{CF}]}$ that $\underset{(n_C x 1)}{\{P^C\}} = \underset{(n_C x n_F)}{[T^{CF}]} \underset{(n_F x 1)}{\{P^F\}}$.

$$\underset{(n_F x 1)}{\{P^F\}} = \underset{(n_F x n_C)}{[T^{FC}]} \underset{(n_C x 1)}{\{P^C\}} \Rightarrow \underset{(n_C x n_F)}{[T^{FC}]^T} \underset{(n_F x 1)}{\{P^F\}} = \underset{(n_C x n_C)}{\left(\underset{(n_C x n_F)}{[T^{FC}]^T} \underset{(n_F x n_C)}{[T^{FC}]} \right)} \underset{(n_C x 1)}{\{P^C\}}$$

We know that the knot values can be removed as we were the ones who inserted them and thus the matrix $\underset{(n_C x n_C)}{\left([T^{FC}]^T [T^{FC}] \right)}$ is invertible, therefore:

$$\underset{(n_C x 1)}{\{P^C\}} = \underbrace{\underset{(n_C x n_C)}{\left(\underset{(n_C x n_F)}{[T^{FC}]^T} \underset{(n_F x n_C)}{[T^{FC}]} \right)^{-1}} \underset{(n_C x n_F)}{[T^{FC}]^T}}_{\underset{(n_C x n_F)}{[T^{CF}]}} \underset{(n_F x 1)}{\{P^F\}}$$

and

$$\underset{(n_C \, x \, n_F)}{\left[T^{CF}\right]} = \left(\underset{(n_C x n_F)}{\left[T^{FC}\right]^T} \underset{(n_F x n_C)}{\left[T^{FC}\right]} \right)^{-1} \underset{(n_C x n_F)}{\left[T^{FC}\right]^T}$$

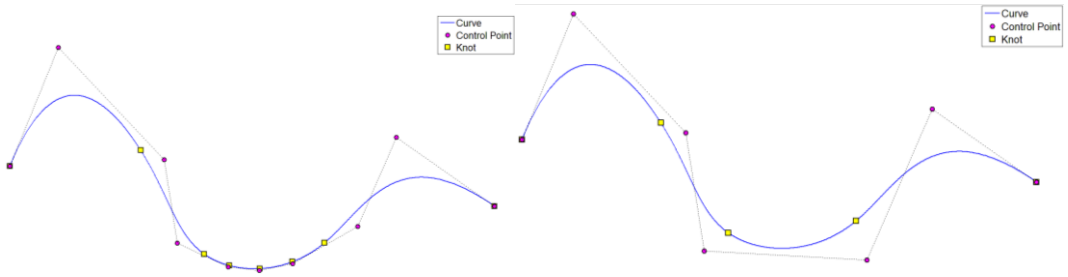


Figure 5.2. Successful Reverse h-refinement. The Curve remains the same.
Knot Value Vector Ξ_1 = { 0 0 0 0 1 2 2.25 2.5 2.75 3 4 4 4 4}
Knot Value Vector Ξ_2 = { 0 0 0 0 1 2 3 4 4 4 4}

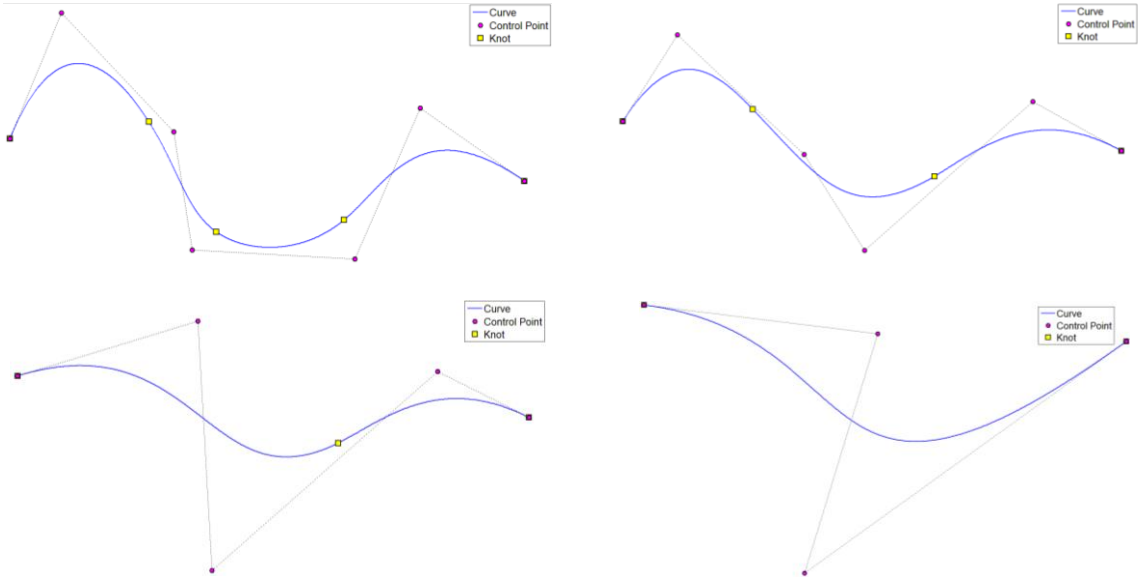


Figure 5.3. Unsuccessful Reverse Knot Refinement. The Curve's geometry is altered.
Knot Vector Ξ_1 = { 0 0 0 0 1 2 3 4 4 4 4}
Knot Vector Ξ_2 = { 0 0 0 0 1 3 4 4 4}
Knot Vector Ξ_3 = { 0 0 0 0 3 4 4 4 4}
Knot Vector Ξ_4 = { 0 0 0 0 4 4 4 4}

5.4 Order Elevation

Order elevation or degree elevation or p-refinement is the finite elements' equivalent of p-refinement in isogeometric analysis. It allows us to enrich the basis by raising the polynomial order of the basis functions used to represent the geometry. The support of the shape functions is increased and there is a stronger interconnection between the elements.

By order elevating a curve we do not want to alter it parametrically or geometrically. Let us assume we want to increase the degree by $t = \bar{p} - p$. The knot value vector has to have its first and last knot value repeated (p+t+1) times. We therefore insert extra t repetitions of the first and last knot value in the vector. Furthermore, we know that along element boundaries the continuity of a curve is C^{p+t-m} but initially it was only C^{p-m}, where m is the multiplicity of the knot. To preserve that continuity along the element boundaries we have to repeat those knots extra t times.

Conclusively, to get the **new knot value vector** we **increase every knot's multiplicity $\mathbf{t = \bar{p} - p}$ times in the initial knot value vector**.

We now only have to obtain the new control point Cartesian coordinates. An obvious but very inefficient solution would be to solve a system of linear equations. The curve is geometrically unchanged:

$$C(\xi) = \sum_{i=1}^{\bar{n}} \bar{N}_{i,p+t}(\xi)\,\bar{P}_i = \sum_{i}^{n} N_{i,p}(\xi) P_i$$

Evaluating the previous equation at $\bar{n}$ appropriate ξ values yields a banded system of $\bar{n}$ linear equations in the unknowns $\bar{P}_i$. In the same way as h-refinement that would give us a matrix equivalent equation between the higher and lower degree curve's control points.

$$\underset{(\bar{n}x1)}{\left\{P^H\right\}} = \underset{(\bar{n}xn)}{\left[T^{HL}\right]} \underset{(nx1)}{\left\{P^L\right\}}$$

More efficient algorithms, but also mathematically complicated exist for raising the degree by 1. Here we will present a mathematically simple and efficient algorithm given in [14] for raising the degree by t.

Algorithm for Order Elevation.

Step1. We replicate existing knots until their multiplicity is equal to the polynomial order (p+1) with knot refinement, thus subdividing the curve into a number of Bezier curves.

Step2. We elevate the order of the polynomial on all those Bezier curves.

Step3. We perform reverse knot refinement (knot removal) by removing the excessive knot values and combining the separate Bezier curves into one, order elevated, B-Spline curve.

We note here that the knot removal is not very expensive as we know which knots are removable, those who we inserted. We in fact know in advance what the final knot value vector is, we have defined it previously.

The algorithm for step 2, to elevate a Bezier curve from (p) to (p+t) degree in one step is:

$$P_i^t = \sum_{j=\max(0,i-t)}^{\min(p,i)} \frac{\binom{p}{j}\binom{t}{i-j}}{\binom{p+t}{i}} P_j, \quad i=0,\ldots,p+t$$

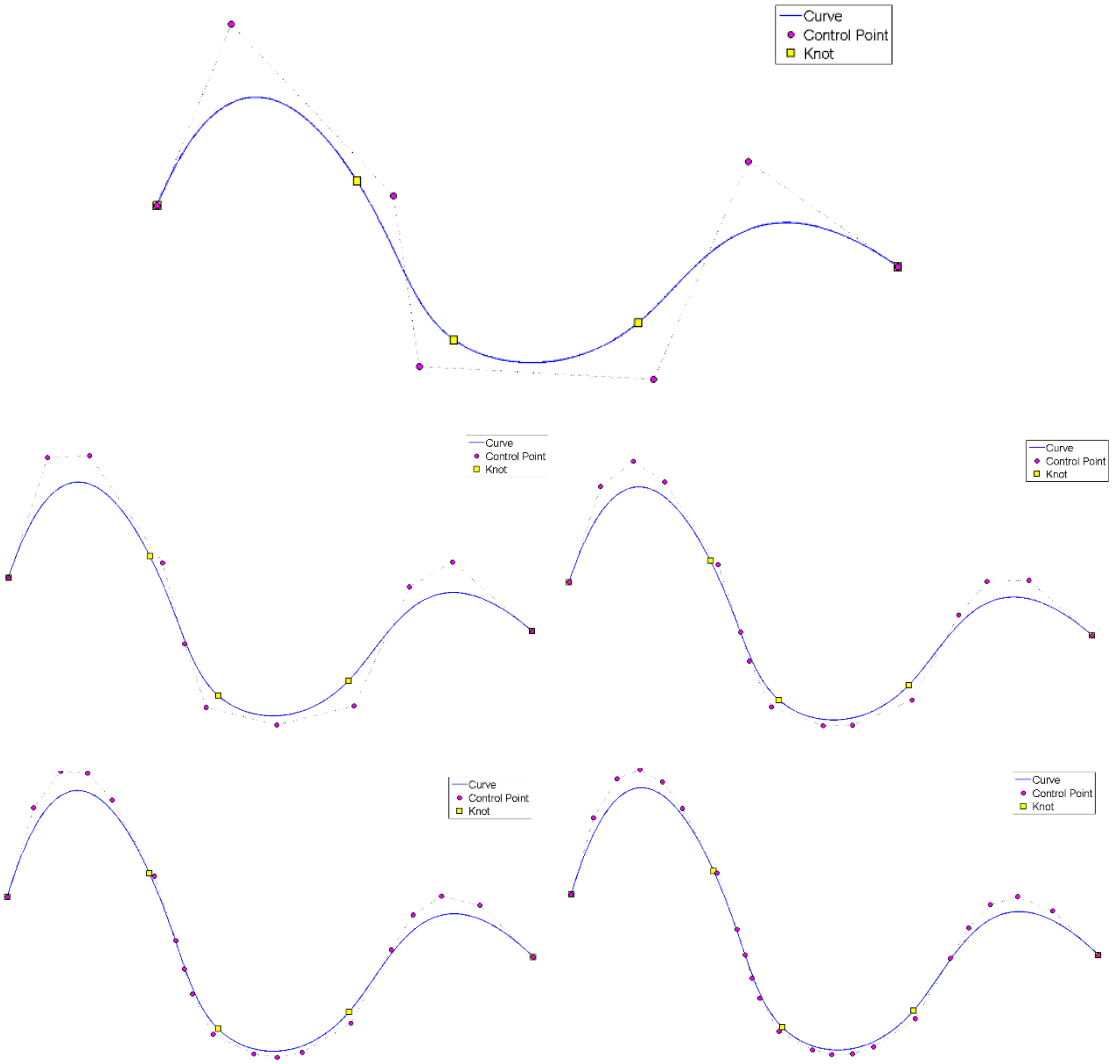


Figure 5.4. Consecutive Degree Elevations from Degree=3 to Degree=7
Initial Knot Value Vector = { 0 0 0 0 1 2 3 4 4 4 4}

5.5 Order reduction

As its name designates, order reduction or degree reduction is the reverse process of p-refinement. We try to reduce the degree of the basis functions without changing it geometrically or parametrically. Degree reduction, as knot value removal, is a problem overdetermined and cannot always be applied with preservation of the same geometry.

The existing algorithms mainly reverse the order elevation procedure by following again three steps:

Algorithm for Order Reduction:

Step1. Decompose the B-Spline curve in Bezier segments,

Step2. Check if degree reduction is applicable and if it is, perform degree reduction.

Step3. Compose again the B-Spline curve from the separate Bezier segments by removing unnecessary knot values.

Of course, if the degree reduction is not applicable in a Bezier segment then it is not applicable for the whole B-Spline. The degree reduction in Bezier curves is a matter well known but we will not elaborate further in this text. An analysis of that topic exists in [14].

In case, however, we have previously performed degree elevation we know that degree reduction is applicable and by exactly how many degrees. We here lay a handy tip for the transition back and forth from a lower degree to a higher degree mesh.

In the standard h-refinement the control points' Cartesian coordinates are transformed as follows:

$$\underset{(n_H x 1)}{\{P^H\}} = \underset{(n_H x n_L)}{[T^{HL}]} \underset{(n_L x 1)}{\{P^L\}}$$

In the same way as in knot removal there is a matrix:

$$\underset{(n_L x n_H)}{[T^{LH}]} = \underset{(n_L x n_L)}{\left(\underset{(n_L x n_H)}{[T^{HL}]^T} \underset{(n_H x n_L)}{[T^{HL}]} \right)^{-1}} \underset{(n_L x n_H)}{[T^{HL}]^T}$$

such that

$$\underset{(n_L x 1)}{\{P^L\}} = \underset{(n_L x n_H)}{[T^{LH}]} \underset{(n_H x 1)}{\{P^H\}}$$

5.6 k-refinement

k-refinement is a new refinement type unique in isogeometric analysis. It was introduced by Cottrell, Hughes, Bazilevs [1]. If we perform a knot refinement and we insert a new knot with multiplicity 1 in the knot value vector of a curve with degree p, then the curve would be C^{p-1} continuous on that new knot. If we afterwards order elevate the curve by $t = \overline{p} - p$ degrees, the continuity there will be preserved by the design of the order elevation process. If we instead, perform order elevation on the curve by $t = \overline{p} - p$ degrees and only then perform a knot insertion, the new knot value will have multiplicity 1 and the continuity there will be higher by t, C^{p+t-1}. This new refinement technique is called k-refinement. That way we are able to enrich the basis with order elevation and in the same time use basis functions of higher continuity achieving efficiency and robustness of the solution space towards a high precision analysis. It is the combination of the p- and h- refinement and to apply it we apply first p- and then h- refinement.

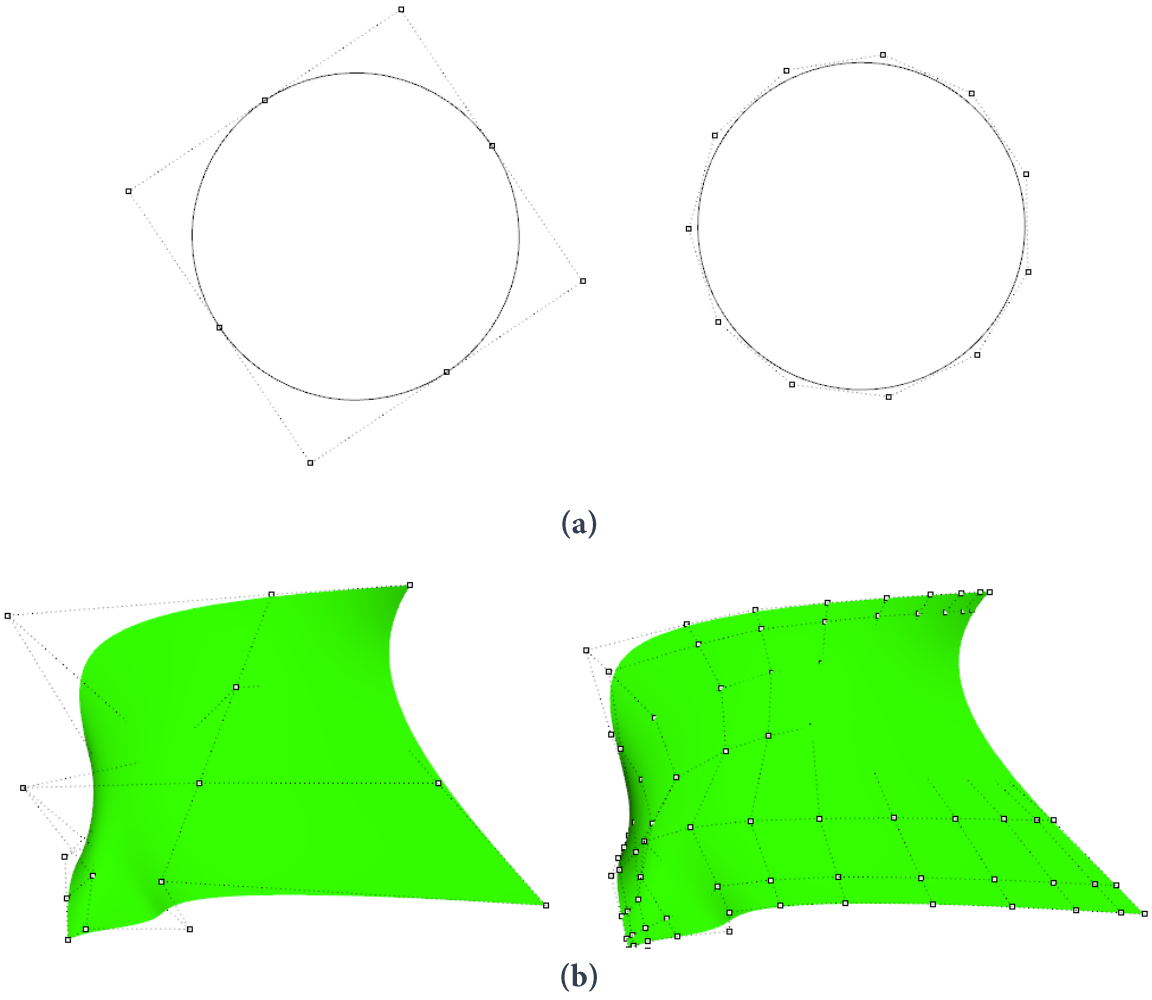


Figure 5.5.
(a) k-refined Circle. The degree elevated from 2 to3.
(b) k-refined Surface. The degree elevated from 2 to5.

5.7 NURBS Refinement

NURBS in d-dimensional space are a projection of B-Splines in the (d+1) dimensional space. In bibliography all refinement methods and algorithms are built for B-Spline non-rational curves. An easy way to refine a NURBS entity is to project it to the (d+1) dimensional B-Spline space, refine it there and then again project it back to the d-dimensional NURBS space.

To project a NURBS geometry in 4D non-rational space, we project the geometry's Control Points to their corresponding projective non-rational, their weight equals to one, Control Points. Thus a Control Point $P_i = (x_i, y_i, z_i)$ with weight w_i from the 3D Cartesian space is projected to the non-rational 4D space control point $P_i^w = (w_i\, x_i, w_i\, y_i, w_i\, z_i, w_i)$.

$$(3D-\text{rational space}) \qquad \underset{(nx3)}{\{P\}} = \underset{(nx3)}{\{X_i, Y_i, Z_i\}} \text{ , with weight } w_i \xrightarrow{\cdot w_i}$$

$$(4D-\text{nonrational space}) \qquad \underset{(nx4)}{\{P^w\}} = \underset{(nx4)}{\{w_i\, X_i, w_i Y_i, w_i Z_i, w_i\}}$$

We could represent that projection of the three coordinates X,Y,Z in a matrix form:

$$\underset{(nx3)}{\{P^w\}} = \underset{(nxn)}{[W]}\,\underset{(nx3)}{\{P\}}$$

where $\underset{(nxn)}{[W]} = \text{diag}(w_1, w_2, ..., w_n) = \text{diag}\left(\underset{(nx1)}{\{w\}}\right)$

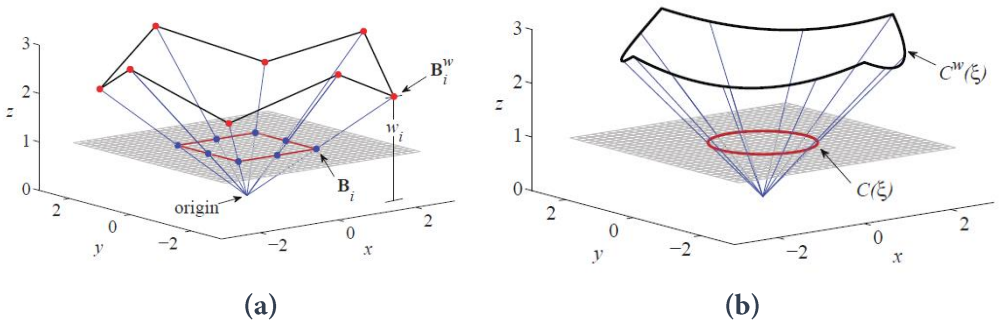


Figure 5.6.
(a) Projection of the control polygon to plane z=1
(b) Projection of the B-Spline Curve to plane z=1, forming the NURBS Curve: a circle.
(Image: Isogeometric analysis: toward integration of CAD and FEA)

During the refinement of the B-Spline curve, we treat all four coordinates, including the weight w_i, the same. If there exists a transformation matrix $\underset{(mxn)}{[T^w]}$ for refinement, then it refines all 4 coordinates:

$$\underset{(mx4)}{\{\bar{P}^w\}} = \underset{(mxn)}{[T^w]}\underset{(nx4)}{\{P^w\}}$$

After the coordinates' refinement we project the new control point $\bar{P}_i^w = (\bar{w}_i\,\bar{x}_i, \bar{w}_i\,\bar{y}_i, \bar{w}_i\,\bar{z}_i, \bar{w}_i) = (\bar{x}_i^w, \bar{y}_i^w, \bar{z}_i^w, \bar{w}_i)$ back to the 3D cartesian space by dividing each coordinate with the new weight $\bar{w}_i$, gaining the rational, refined control point $\bar{P}_i = (\bar{x}_i, \bar{y}_i, \bar{z}_i) = \left(\frac{\bar{x}_i^w}{\bar{w}_i}, \frac{\bar{y}_i^w}{\bar{w}_i}, \frac{\bar{z}_i^w}{\bar{w}_i}\right)$ with weight $\bar{w}_i$.

$$(4D-\text{nonrational space}) \quad \underset{(nx4)}{\{\bar{P}^w\}} = \underset{(nx4)}{\{\bar{w}_i\,\bar{X}_i, \bar{w}_i\,\bar{Y}_i, \bar{w}_i\,\bar{Z}_i, \bar{w}_i\}} \begin{array}{c} /\bar{w}_i \\ \longrightarrow \\ \longleftarrow \\ \cdot\bar{w}_i \end{array}$$

$$(3D-\text{rational space}) \quad \underset{(nx3)}{\{\bar{P}\}} = \underset{(nx3)}{\{\bar{X}_i, \bar{Y}_i, \bar{Z}_i\}} \text{ , with weight } \bar{w}_i$$

We could represent that projection of the three coordinates X,Y,Z in a matrix form:

$$\underset{(mx3)}{\{\bar{P}^w\}} = \underset{(mxm)}{[\bar{W}]}\underset{(mx3)}{\{\bar{P}\}} \Rightarrow \underset{(mx3)}{\{\bar{P}\}} = \underset{(mxm)}{[\bar{W}]^{-1}}\underset{(mx3)}{\{\bar{P}^w\}}$$

where $\underset{(mxm)}{[\bar{W}]} = \text{diag}(\bar{w}_1, \bar{w}_2, ..., \bar{w}_m) = \text{diag}\left(\underset{(mx1)}{\{\bar{w}\}}\right)$ and $\underset{(mx1)}{\{\bar{w}\}} = \underset{(mxn)}{[T^w]}\underset{(nx1)}{\{w\}}$

Conclusively, we can perform the NURBS Control Point refinement in one step,

$$\underset{(mx3)}{\{\bar{P}\}} = \underset{(mxn)}{[T]}\underset{(nx3)}{\{P\}}$$

where $\underset{(mxn)}{[T]} = \underset{(mxm)}{[\bar{W}]^{-1}}\underset{(mxn)}{[T^w]}\underset{(nxn)}{[W]}$ and $\underset{(mx1)}{\{\bar{w}\}} = \underset{(mxn)}{[T^w]}\underset{(nx1)}{\{w\}}$

5.8 Shape function transformation matrices in knot refinement

For further investigation on the refinement process and to develop Hierarchical Refinement in the next chapter, we will need to relate the shape functions before (initial) and after (final) the refinement process.

5.8.1 B-Spline Basis Functions related.

Equalizing the initial and final geometry of the curve in the physical space we get:

$$C^{I}_{(\xi)} = C^{F}_{(\xi)} \Rightarrow \underset{(1xn)}{\left\{N^{I}_{(\xi)}\right\}^{T}} \cdot \underset{(nx3)}{\left\{P^{I}\right\}} = \underset{(1xm)}{\left\{N^{F}_{(\xi)}\right\}^{T}} \cdot \underset{(mx3)}{\left\{P^{F}\right\}} = \underset{(1xm)}{\left\{N^{F}_{(\xi)}\right\}^{T}} \underset{(mxn)}{\left[T^{FI}\right]} \cdot \underset{(nx3)}{\left\{P^{I}\right\}}$$

And thus we suspect that a relation $\underset{(1xn)}{\left\{N^{I}_{(\xi)}\right\}^{T}} = \underset{(1xm)}{\left\{N^{F}_{(\xi)}\right\}^{T}} \underset{(mxn)}{\left[T^{FI}\right]} \Leftrightarrow \underset{(nx1)}{\left\{N^{I}_{(\xi)}\right\}} = \underset{(nxm)}{\left[T^{FI}\right]^{T}} \underset{(mx1)}{\left\{N^{F}_{(\xi)}\right\}}$ could be connecting the initial and fine mesh basis functions. The proof is rather complicated but we will lay it here for the sake of completeness.

Assuming an initial knot value vector, $\Xi = \left\{\xi_1, \xi_2, ..., \xi_{n+p+1}\right\}$,and a final knot value vector with only one new knot value $\bar{\xi}_{\kappa+1} = \bar{\xi}$,

$$\bar{\Xi} = \left\{\bar{\xi}_1 = \xi_1, \bar{\xi}_2 = \xi_2, .., \bar{\xi}_{\kappa-1} = \xi_{\kappa-1}, \bar{\xi}_{\kappa} = \xi_{\kappa}, \bar{\xi}_{\kappa+1} = \bar{\xi}, \bar{\xi}_{\kappa+2} = \xi_{\kappa+1}, .., \bar{\xi}_{n+p+2} = \xi_{n+p+1}\right\}$$

we can use the following formulas from [14] in the chapter “Knot Insertion”, page 142.

The existing initial basis functions can be expressed as a function of the new ones.

$$N_{i,p} = \begin{cases} \bar{N}_{i,p} & \text{for i=1,2,...,k-p-1} \\ \bar{N}_{i+1,p} & \text{for i=k+1,....,n} \\ \\ \dfrac{\bar{\xi} - \bar{\xi}_i}{\bar{\xi}_{i+p+1} - \bar{\xi}_i} \bar{N}_{i,p} + \dfrac{\bar{\xi}_{i+p+2} - \bar{\xi}}{\bar{\xi}_{i+p+2} - \bar{\xi}_{i+1}} \bar{N}_{i+1,p} & \text{for i=k-p,....,k} \end{cases}$$

Equation 5.1. Initial and Refined shape functions related in the case of a single Knot Insertion.

The first two equations are easy to derive by taking into account that the support of each basis function is (p+1) knot value spans and a basis function will be affected only if the new knot value resides in that support. The third equation is proved by induction on p and the Cox de Boor algorithm we presented earlier, but we omit the proof as it is quite messy. Proofs using divided differences are found in [15], [17], [18].

We name $\underset{((n+1)xn)}{[T]} = \left[T_{ij}\right]_{((n+1)xn)}$ the transformation matrix giving us the new control points from the initial control points for one knot insertion:

$$\underset{((n+1)x1)}{\{P^F\}} = \underset{((n+1)xn)}{[T]} \underset{(nx1)}{\{P^I\}}$$

From [14] we have that:

$$Q_i = (1-a_i)P_{i-1} + a_i P_i$$

$$\text{where } a_i = \begin{cases} 1, & i \le k-p \\ \dfrac{\overline{\xi} - \xi_i}{\xi_{i+p} - \xi_i}, & k-p+1 \le i \le k \\ 0, & i \ge k+1 \end{cases}$$

We assumed $\overline{\xi}_{k+1} = \overline{\xi}$ is the new value of the knot value vector, therefore:

$$\overline{\xi}_i = \begin{cases} \xi_i & i \le k \\ \overline{\xi} & = k+1 \\ \xi_{i-1} & \ge k+2 \end{cases} \text{ and } \xi_i = \begin{cases} \overline{\xi}_i & i \le k \\ \overline{\xi}_{i+1} & i \ge k+1 \end{cases}$$

Thus we can express the new control points as a function of the new knot value vector:

$$Q_i = (1-a_i)P_{i-1} + a_i P_i$$

$$\text{where } a_i = \begin{cases} 1, & i \le k-p \\ \dfrac{\overline{\xi} - \overline{\xi}_i}{\overline{\xi}_{i+p+1} - \overline{\xi}_i}, & k-p+1 \le i \le k \\ 0, & i \ge k+1 \end{cases}$$

The control points' transformation matrix $\underset{((n+1)x1)}{\{Q\}} = \underset{((n+1)xn)}{[T]} \underset{(nx1)}{\{P\}}$ for this single knot value insertion is shown in Figure 5.7.

We want to prove that $\underset{(nx1)}{\{N^I\}} = \underset{(nx(n+1))}{[T]}^T \underset{((n+1)xn)}{\{N^F\}} \Leftrightarrow N^I_{i,p} = \sum_{j=1}^{n+1} T_{ji} N^F_{j,p}$

Cases:

- For **i=1,2,…,k-p-1**

  the column $\underset{((n+1)x1)}{\{T_i\}}$ has only one element non-zero, $T_{ii} = 1$. Thus

  $N^I_{i,p} = \sum_{j=1}^{n+1} T_{ji} N^F_{j,p} = T_{ii} N^F_{i,p} = 1 \cdot N^F_{i,p} = N^F_{i,p}$ which is valid from Equation 5.1.

$$[T]_{((n+1)xn)} = \begin{array}{c|cccccccccccccc} \text{Row Index} \backslash \text{Column Index} & 1 & 2 & \cdots & \cdots & k-p & k-p+1 & \cdots & \cdots & k-1 & k & k+1 & \cdots & \cdots & n-1 & n \\ \hline 1 & 1 & & & & & & & & & & & & & & \\ 2 & & 1 & & & & & & & & & & & & & \\ \vdots & & & \ddots & & & & & & & & & & & & \\ \vdots & & & & \ddots & & & & & & & & & & & \\ k-p & & & & & 1 & & & & & & & & & & \\ k-p+1 & & & & & 1-a_{k-p+1} & a_{k-p+1} & & & & & & & & & \\ \vdots & & & & & & 1-a_{k-p+2} & \ddots & & & & & & & & \\ \vdots & & & & & & & \ddots & & & & & & & & \\ k-1 & & & & & & & & 1-a_{k-1} & a_{k-1} & & & & & \\ k & & & & & & & & & 1-a_k & a_k & & & & \\ k+1 & & & & & & & & & & 1 & & & & & \\ \vdots & & & & & & & & & & & \ddots & & & & \\ \vdots & & & & & & & & & & & & \ddots & & & \\ n-1 & & & & & & & & & & & & & \ddots & & \\ n & & & & & & & & & & & & & & 1 & \\ n+1 & & & & & & & & & & & & & & & 1 \end{array}$$

Figure 5.7. Control Point Transformation Matrix for a single Knot Value Insertion.

- For **i=k+1,…,n**

  The column $\underset{((n+1)x1)}{\{T_i\}}$ has only one element non-zero, $T_{i+1i} = 1$. Thus

  $N^I_{i,p} = \sum_{j=1}^{n+1} T_{ji} N^F_{j,p} = T_{i+1i} N^F_{i+1,p} = 1 \cdot N^F_{i+1,p} = N^F_{i+1,p}$ which is valid from Equation 5.1.

- For **i=k-p+1,…,k-1**

  The column $\underset{((n+1)x1)}{\{T_i\}}$ has only two elements non-zeros,

  $$T_{ii} = a_i = \frac{\overline{\xi} - \overline{\xi}_i}{\overline{\xi}_{i+p+1} - \overline{\xi}_i} \text{ and } T_{i+1i} = 1 - a_{i+1} = \frac{\overline{\xi}_{i+p+1} - \overline{\xi}}{\overline{\xi}_{i+p+1} - \overline{\xi}_i}$$

  Thus $N^I_{i,p} = \sum_{j=1}^{n+1} T_{ji} N^F_{j,p} = T_{ii} N^F_{i,p} + T_{i+1i} N^F_{i+1,p} = \frac{\overline{\xi} - \overline{\xi}_i}{\overline{\xi}_{i+p+1} - \overline{\xi}_i} N^F_{i,p} + \frac{\overline{\xi}_{i+p+1} - \overline{\xi}}{\overline{\xi}_{i+p+1} - \overline{\xi}_i} N^F_{i+1,p}$ which is valid from **Equation 5.1.**

- In the special case of **i=k-p**:

  The column $\underset{((n+1)x1)}{\{T_i\}}$ has only two non-zero elements,

$$T_{ii}=1 \text{ and } T_{i+1i}=\frac{\bar{\xi}_{i+p+1}-\bar{\xi}}{\bar{\xi}_{i+p+1}-\bar{\xi}_i}$$

$$T_{ii}=1=\frac{\bar{\xi}_{i+p+1}-\bar{\xi}_i}{\bar{\xi}_{i+p+1}-\bar{\xi}_i} \overset{i=k-p}{=} \frac{\bar{\xi}_{k+1}-\bar{\xi}_i}{\bar{\xi}_{k+1}-\bar{\xi}_i} \overset{\bar{\xi}_{k+1}=\bar{\xi}}{=} \frac{\bar{\xi}-\bar{\xi}_i}{\bar{\xi}_{k+1}-\bar{\xi}_i}$$ and the proof is reduced to the previous more general case of i=k-p+1,…,k-1.

- In the special case of **i=k**:

  The column $\underset{((n+1)x1)}{\{T_i\}}$ has only two elements non-zeros,

  $$T_{ii}=\frac{\bar{\xi}-\bar{\xi}_i}{\bar{\xi}_{i+p+1}-\bar{\xi}_i} \text{ and } T_{i+1i}=1$$

  $$T_{i+1i}=1=\frac{\bar{\xi}_{i+p+1}-\bar{\xi}_i}{\bar{\xi}_{i+p+1}-\bar{\xi}_i} \overset{i=k}{=} \frac{\bar{\xi}_{i+p+1}-\bar{\xi}_{k+1}}{\bar{\xi}_{i+p+1}-\bar{\xi}_i} \overset{\bar{\xi}_{k+1}=\bar{\xi}}{=} \frac{\bar{\xi}_{i+p+1}-\bar{\xi}_{k+1}}{\bar{\xi}_{k+1}-\bar{\xi}_i}$$

  and the proof is reduced to the previous more general case of i=k-p+1,…,k-1.

Thus we proved that for a single knot insertion with control points' transformation matrix $\underset{((n+1)x1)}{\{P^F\}} = \underset{((n+1)xn)}{[T]} \underset{(nx1)}{\{P^I\}}$, the basis functions are transformed as $\underset{(nx1)}{\{N^I\}} = \underset{(nx(n+1))}{[T]^T} \underset{((n+1)x1)}{\{N^F\}}$.

The general case of r knot value insertions is equivalent to successive r single knot value insertions, so:

$$\underset{((n+r)x1)}{\{P^F\}} = \left(\underset{((n+r)x(n+r-1))}{[T^r]} \cdots \underset{((n+2)x(n+1))}{[T^2]} \underset{((n+1)xn)}{[T^1]} \right) \underset{(nx1)}{\{P\}} = \underset{((n+r)xn)}{[T^{FI}]} \underset{(nx1)}{\{P^I\}}$$

where $\underset{((n+r)xn)}{[T^{FI}]} = \underset{((n+r)x(n+r-1))}{[T^r]} \cdots \underset{((n+2)x(n+1))}{[T^2]} \underset{((n+1)xn)}{[T^1]}$ is the control points' transformation matrix.

For the initial and final shape functions after all r knot insertions:

$$\underset{(nx1)}{\{N^I\}} = \left(\underset{((n+r)x(n+r-1))}{[T^r]^T} \cdots \underset{((n+2)x(n+1))}{[T^2]^T} \underset{((n+1)xn)}{[T^1]^T} \right) \underset{((n+1)x1)}{\{N^F\}} = \left(\underset{((n+1)xn)}{[T^1]} \underset{((n+2)x(n+1))}{[T^2]} \cdots \underset{((n+r)x(n+r-1))}{[T^r]} \right)^T \underset{((n+r)x1)}{\{N^F\}} \Rightarrow$$

$$\underset{(nx1)}{\{N^I\}} = \underset{(nx(n+r))}{[T^{FI}]^T} \underset{((n+r)x1)}{\{N^F\}}$$

or in the form we are usually accustomed to, with n initial control points and m final control points $\underset{(mx1)}{\{P^F\}} = \underset{(mxn)}{[T^{FI}]} \underset{(nx1)}{\{P^I\}} \Rightarrow$

$$\underset{(nx1)}{\{N^I\}} = \underset{(nxm)}{[T^{FI}]^T} \underset{(mx1)}{\{N^F\}}$$

5.8.2 NURBS Shape functions related

The NURBS Shape functions are defined as:

$$\underset{(nx1)}{\{R_i\}} = \underset{(nx1)}{\left\{ \frac{w_i N_i}{\sum_{j=1}^{n} w_j N_j} \right\}} = \underset{(nx1)}{\left\{ \frac{w_i N_i}{W(\xi)} \right\}} = \frac{1}{W(\xi)} \underset{(nx1)}{\{w_i N_i\}}$$

But the weight function $W(\xi)$ is immune to refinement:

$$W^I(\xi) = \sum_{j=1}^{n} w^I_j N^I_j = \underset{(1xn)}{\{w^I\}^T} \underset{(nx1)}{\{N^I\}} = \underset{(1xn)}{\{w^I\}^T} \left(\underset{(nxm)}{[T^{FI}]^T} \cdot \underset{(mx1)}{\{N^F_{(\xi)}\}} \right) = \underset{(1xm)}{\left(\underset{(mxn)}{[T^{FI}]} \underset{(nx1)}{\{w^I\}} \right)^T} \underset{(mx1)}{\{N^F_{(\xi)}\}} =$$

$$\underset{(1xm)}{\{w^F\}^T} \underset{(mx1)}{\{N^F_{(\xi)}\}} = \sum_{j=1}^{m} w^F_j N^F_j = W^F(\xi) \Rightarrow$$

$$W^I(\xi) = W^F(\xi) = W(\xi)$$

By using the diagonal matrices $\underset{(nxn)}{[W]} = \text{diag}(w_1, w_2, ..., w_n) = \text{diag}\left(\underset{(nxn)}{\{w\}}\right)$ again we proceed as:

$$\underset{(nx1)}{\{R_i\}} = \frac{1}{W(\xi)} \underset{(nx1)}{\{w_i N_i\}} \Rightarrow$$

$$\underset{(nxn)}{[W^I]^{-1}} \underset{(nx1)}{\{R^I\}} = \frac{1}{W(\xi)} \underset{(nx1)}{\{N^I_i\}} = \frac{1}{W(\xi)} \underset{(nxm)}{[T^{FIw}]^T} \underset{(mx1)}{\{N^F\}} = \frac{1}{W(\xi)} \underset{(nxm)}{[T^{FIw}]^T} \underset{(mxm)}{[W^F]^{-1}} \underset{(mx1)}{\{R^F\}} \Rightarrow$$

$$\underset{(nx1)}{\{R^I\}} = \underset{(nxn)}{[W^I]} \underset{(nxm)}{[T^{FIw}]^T} \underset{(mxm)}{[W^F]^{-1}} \underset{(mx1)}{\{R^F\}} = \underset{(nxn)}{[W^I]^T} \underset{(nxm)}{[T^{FIw}]^T} \underset{(mxm)}{[W^F]^{-T}} \underset{(mx1)}{\{R^F\}} =$$

$$\left(\underset{(mxm)}{[W^F]^{-1}} \underset{(mxn)}{[T^{FIw}]} \underset{(nxn)}{[W^I]} \right)^T = \underset{(nxm)}{[T^{FI}]^T} \underset{(mx1)}{\{R^F\}} \Rightarrow$$

$$\underset{(nx1)}{\{R^I\}} = \underset{(nxm)}{[T^{FI}]^T} \underset{(mx1)}{\{R^F\}}$$

Conclusively, in the case of NURBS, which also incorporates the special case of B-Splines, the new shape functions can be expressed as a function of the initial ones as

$\underset{(nx1)}{\{R^I\}} = \underset{(nxm)}{[T^{FI}]^T} \underset{(mx1)}{\{R^F\}}$ where $\underset{(mxn)}{[T^{FI}]}$ is the control points' transformation $\underset{(mx1)}{\{P^F\}} = \underset{(mxn)}{[T^{FI}]} \underset{(nx1)}{\{P^I\}}$

Furthermore, as we previously stated in the chapter 5.3, Knot Value Removal, we were the ones who did the knot refinement so the reverse knot refinement, that transitions from the final to the initial mesh, is possible. The transformation matrix in that case will be

$$\underset{(nxm)}{\left[T^{IF}\right]}=\underset{(nxn)}{\left(\underset{(nxm)}{\left[T^{FI}\right]^{T}}\underset{(mxn)}{\left[T^{FI}\right]}\right)^{-1}}\underset{(nxm)}{\left[T^{FI}\right]^{T}}.$$

Of course the final and initial Shape functions can be related in the same way

$$\underset{(nx1)}{\left\{R^{I}\right\}}=\underset{(nxm)}{\left[T^{FI}\right]^{T}}\underset{(mx1)}{\left\{R^{F}\right\}}\Rightarrow\underset{(mxn)}{\left[T^{IF}\right]^{T}}\underset{(nx1)}{\left\{R^{I}\right\}}=\left(\left(\underset{(nxm)}{\left[T^{FI}\right]^{T}}\underset{(mxn)}{\left[T^{FI}\right]}\right)^{-1}\underset{(nxm)}{\left[T^{FI}\right]^{T}}\right)^{T}\underset{(nxm)}{\left[T^{FI}\right]^{T}}\underset{(mx1)}{\left\{R^{F}\right\}}\Rightarrow$$

$$\underset{(mxn)}{\left[T^{IF}\right]^{T}}\underset{(nx1)}{\left\{R^{I}\right\}}=\underset{(mxn)}{\left[T^{FI}\right]}\left(\underset{(nxm)}{\left[T^{FI}\right]^{T}}\underset{(mxn)}{\left[T^{FI}\right]}\right)^{-1}\underset{(nxm)}{\left[T^{FI}\right]^{T}}\underset{(mx1)}{\left\{R^{F}\right\}}\Rightarrow$$

$$\underset{(mxn)}{\left[T^{IF}\right]^{T}}\underset{(nx1)}{\left\{R^{I}\right\}}=\left(\underset{(mxn)}{\left[T^{FI}\right]}\underset{(nxm)}{\left[T^{FI}\right]^{T}}\right)^{-1}\underset{(mxn)}{\left[T^{FI}\right]}\underset{(nxm)}{\left[T^{FI}\right]^{T}}\underset{(mxn)}{\left[T^{FI}\right]}\left(\underset{(nxm)}{\left[T^{FI}\right]^{T}}\underset{(mxn)}{\left[T^{FI}\right]}\right)^{-1}\underset{(nxm)}{\left[T^{FI}\right]^{T}}\underset{(mx1)}{\left\{R^{F}\right\}}\Rightarrow$$

$$\underset{(mx1)}{\left\{R^{F}\right\}}=\underset{(mxn)}{\left[T^{IF}\right]^{T}}\underset{(nx1)}{\left\{R^{I}\right\}}$$

5.9 Load Refinement

If we have already evaluated the equivalent loads on a coarse mesh there is no need to compute them again on a finer mesh. We can do that by using the proved equation $\underset{(mx1)}{\left\{R^F(\xi,\eta,\zeta)\right\}} = \underset{(mxn)}{\left[T^{CF}\right]^T} \underset{(nx1)}{\left\{R^C(\xi,\eta,\zeta)\right\}}$

The evaluation of the equivalent loads in the fine mesh is:

$$\underset{(mx3)}{\left\{L^F\right\}} = \int_\Omega \underset{(mx1)}{\left\{R^F(\xi,\eta,\zeta)\right\}} \underset{(1x3)}{f(\xi,\eta,\zeta)} \det\left[J\right] d\Omega =$$

$$\int_\Omega \underset{(mxn)}{\left[T^{CF}\right]^T} \underset{(nx1)}{\left\{R^C(\xi,\eta,\zeta)\right\}} \underset{(1x3)}{f(\xi,\eta,\zeta)} \det\left[J\right] d\Omega =$$

$$\underset{(mxn)}{\left[T^{CF}\right]^T} \int_\Omega \underset{(nx1)}{\left\{R^C(\xi,\eta,\zeta)\right\}} \underset{(1x3)}{f(\xi,\eta,\zeta)} \det\left[J\right] d\Omega \Rightarrow$$

$$\underset{(mx3)}{\left\{L^F\right\}} = \underset{(mxn)}{\left[T^{CF}\right]^T} \underset{(nx3)}{\left\{L^C\right\}}$$

Of course, the above process can be applied in the same way for reverse refinement resulting in $\underset{(nx3)}{\left\{L^C\right\}} = \underset{(nxm)}{\left[T^{FC}\right]^T} \underset{(mx3)}{\left\{L^F\right\}}$.

We should have in mind though, that if we described the load distribution with the coarse mesh shape functions and those were not sufficiently flexible to describe the load distribution, then the function $\underset{(1x3)}{f(\xi,\eta,\zeta)}$ will not be accurate. In that case, we should describe the load distribution with the new more flexible shape functions in the CAD software again and then proceed with calculating the equivalent loads.

6 Hierarchical Refinement

6.1 Introduction

In the previous chapter we reviewed several standard refinement techniques in knot insertion and removal, degree elevation and reduction. In those procedures, the shape function coefficients, Control Points' Cartesian coordinates, are calculated again and therefore the whole procedure of stiffness matrix assembly is repeated. In the case we have a large and complicated structure, the number of degrees of freedom could be huge and reassembling and inversing the stiffness matrix could add a substantial computational cost.

In this chapter, we will attempt to develop a new method to tackle that computational cost in the case of knot refinement (h-refinement), by managing to express the new Stiffness matrix in a hierarchical form:

$$\underset{(MxM)}{\left[K^{Final}\right]}=\begin{bmatrix}\underset{(NxN)}{\left[K^{Initial}\right]} & \underset{(NxQ)}{\left[K_{en}^{Final}\right]}\\ \underset{(QxN)}{\left[K_{ne}^{Final}\right]} & \underset{(QxQ)}{\left[K_{nn}^{Final}\right]}\end{bmatrix}$$

where N, M are the number of the degrees of freedom in the initial and final mesh and Q=M-N the number of new degrees of freedom. Also the subscripts n, e stand for new and enriched referring to the degrees of freedom.

That way, we will have to assemble only the much smaller matrices $\left[K_{en}^{Final}\right]$, $\left[K_{ne}^{Final}\right]$, $\left[K_{nn}^{Final}\right]$ and we will also be able to utilize the already computed inverse matrix $\left[K^{Initial}\right]$.

The knot values have a local influence, that is when a knot value is inserted only the Control Points whose support includes that new knot value are influenced and the rest of the control points remain the same. So when expressing the new control points' coordinates $\underset{(mx3)}{\left\{P^{F}\right\}}=\underset{(mxn)}{\left[T^{FI}\right]}\underset{(nx3)}{\left\{P^{I}\right\}}$, the matrix $\underset{(mxn)}{\left[T^{FI}\right]}$ is mainly unitary but for the area where the control points are influenced by the knot insertion.

6.2 Refinement of the Stiffness Matrix

In the previous chapter "Refinement," we established a relation between the initial and final shape functions for knot refinement and reverse knot refinement. We will use those relations to also relate the initial and final stiffness matrices.

6.2.1 Refinement of the Stiffness Matrix 1D

The curve is given by the equation $C(\xi) = \underset{(1xn)}{\{R\}^T}\ \underset{(nx1)}{\{P\}}$. The initial and final curve are the same in physical space: $C^I(\xi) = C^F(\xi)$.

Jacobian [J]

The Jacobian of the transformation from parametric coordinates to physical (Cartesian) coordinates:

$$\underset{(1x1)}{[J^F]} = \underset{(1xm)}{\left\{R^F_{\xi(\xi)}\right\}^T}\ \underset{(mx3)}{\left\{P^F\right\}} = C^F_{\xi(\xi)} = C^I_{\xi(\xi)} = \underset{(1xn)}{\left\{R^I_{\xi(\xi)}\right\}^T}\ \underset{(nx3)}{\left\{P^I\right\}} = \underset{(1xn)}{[D^I_N]^T} \cdot \underset{(nx3)}{\left\{P^I\right\}} = \underset{(1x1)}{[J^I]}$$

The Jacobian stays the same with the knot refinement.

Deformation Matrix [B_1]

$$\underset{(1x1)}{\left[B^F_{1(\xi)}\right]} = \underset{(1x1)}{\left[J^F_{(\xi)}\right]^{-1}} = \underset{(1x1)}{\left[J^I_{(\xi)}\right]^{-1}} = \underset{(1x1)}{\left[B^I_{1(\xi)}\right]}$$

Deformation Matrix [B_2]

$$\underset{(1xm)}{[B^F_{2(\xi)}]} = \underset{(1xm)}{\left[R^F_{\xi(\xi)}\right]^T} = \underset{(1xn)}{\left\{R^I_{(\xi)}\right\}^T}\ \underset{(nxm)}{\left[T^{IF}\right]} = \underset{(1xn)}{[B^I_{2(\xi)}]}\ \underset{(nxm)}{\left[T^{IF,B}\right]}$$

$$\text{and } \underset{(1xn)}{[B^I_{2(\xi)}]} = \underset{(1xn)}{\left[R^I_{\xi(\xi)}\right]^T} = \underset{(1xm)}{\left\{R^F_{(\xi)}\right\}^T}\ \underset{(mxn)}{\left[T^{FI}\right]} = \underset{(1xm)}{[B^F_{2(\xi)}]}\ \underset{(mxn)}{\left[T^{FI,B}\right]}$$

where

$$\underset{(nxm)}{\left[T^{IF,B}\right]} = \underset{(nxm)}{\left[T^{IF}\right]} \text{ and } \underset{(nxm)}{\left[T^{FI,B}\right]} = \underset{(nxm)}{\left[T^{FI}\right]}$$

Deformation Matrix [B]=[B_1][B_2]

$$\underset{(1xm)}{\left[B^F\right]} = \underset{(1x1)}{\left[B^F_1\right]}\ \underset{(1xm)}{\left[B^F_2\right]} = \underset{(1x1)}{\left[B^I_1\right]}\ \underset{(1xn)}{\left[B^I_2\right]}\ \underset{(nxm)}{\left[T^{IF,B}\right]} = \underset{(1xn)}{\left[B^I\right]}\ \underset{(nxm)}{\left[T^{IF,B}\right]}$$

$$\text{and } \underset{(1xn)}{\left[B^I\right]} = \underset{(1x1)}{\left[B^I_1\right]}\ \underset{(1xn)}{\left[B^I_2\right]} = \underset{(1x1)}{\left[B^F_1\right]}\ \underset{(1xm)}{\left[B^F_2\right]}\ \underset{(mxn)}{\left[T^{FI,B}\right]} = \underset{(1xm)}{\left[B^F\right]}\ \underset{(mxn)}{\left[T^{FI,B}\right]}$$

Stiffness Matrix [K]

$$\underset{(mxm)}{\left[K^{F}\right]}=\left(\int_{\xi}\underset{(mx1)}{\left[B^{F}_{(\xi)}\right]^{T}}\underset{(1x1)}{[E]}\underset{(1xm)}{\left[B^{F}_{(\xi)}\right]}\det(J)\,d\xi\right)\cdot \text{Height}\cdot\text{Thickness}$$

$$=\left(\int_{\xi}\left(\underset{(1xn)}{\left[B^{I}\right]}\underset{(nxm)}{\left[T^{IF,B}\right]}\right)^{T}\underset{(1x1)}{[E]}\left(\underset{(1xn)}{\left[B^{I}\right]}\underset{(nxm)}{\left[T^{IF,B}\right]}\right)\det(J)\,d\xi\right)\cdot \text{Height}\cdot\text{Thickness}$$

$$=\underset{(mxn)}{\left[T^{IF,B}\right]^{T}}\left(\int_{\xi}\underset{(nx1)}{\left[B^{I}\right]^{T}}\underset{(1x1)}{[E]}\underset{(1xn)}{\left[B^{I}\right]}\det(J)\,d\xi\right)\cdot \text{Height}\cdot\text{Thickness}\cdot\underset{(nxm)}{\left[T^{IF,B}\right]}$$

$$=\underset{(mxn)}{\left[T^{IF,B}\right]^{T}}\underset{(nxn)}{\left[K^{I}\right]}\underset{(nxm)}{\left[T^{FI,B}\right]}\Rightarrow$$

$$\underset{(mxm)}{\left[K^{F}\right]}=\underset{(mxn)}{\left[T^{IF,B}\right]^{T}}\underset{(nxn)}{\left[K^{I}\right]}\underset{(nxm)}{\left[T^{IF,B}\right]}$$

In the same way

$$\underset{(nxn)}{\left[K^{I}\right]}=\underset{(nxm)}{\left[T^{FI,B}\right]^{T}}\underset{(mxm)}{\left[K^{F}\right]}\underset{(mxn)}{\left[T^{FI,B}\right]}$$

6.2.2 Refinement of the Stiffness Matrix 2D

In order to form a hierarchical procedure as in 1D elasticity we need to determine a control point transformation matrix for all the control points in a form $\underset{(mx1)}{\{P^F\}} = \underset{(mxn)}{[T^{FI}]}\underset{(nx1)}{\{P^I\}}$

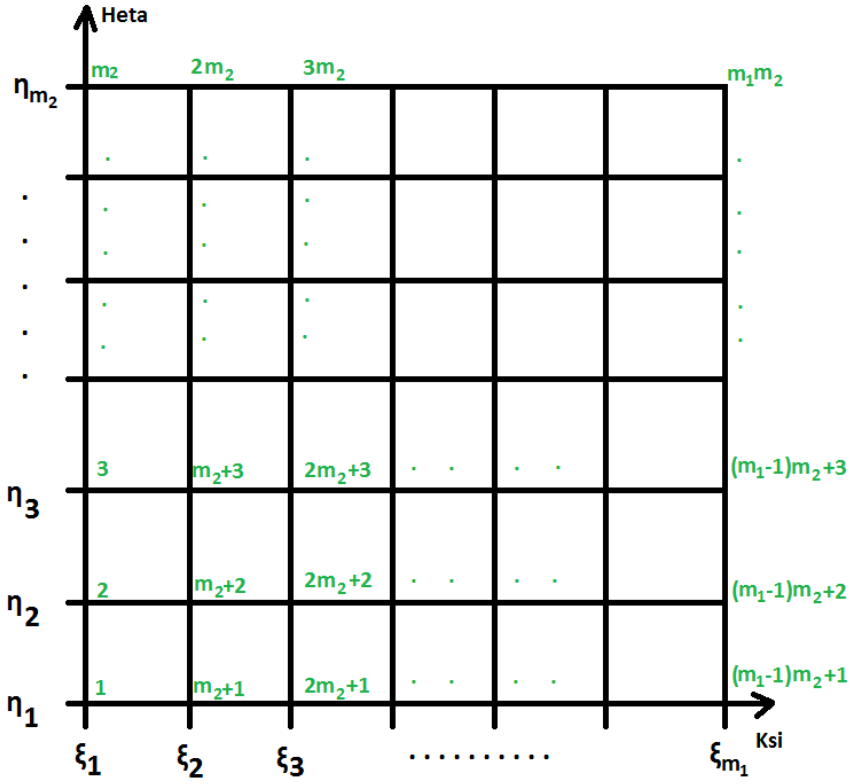
6.2.2.1 Control Point Transformation Matrix 2D

A fundamental characteristic of the analysis with NURBS is the full tensor product property. In that way we can perform the refinement (and the control point coordinate transformation) on each parametric axis ξ, η separately. For the needs of hierarchical refinement we need to build a Transformation Matrix $[T^{IF}]$ that directly refines the control points of the Final mesh to the Initial mesh, with their global numbering. From the refinement on each axis ξ, η we have already obtained the two transformation matrices, $[T_\xi^{FI}]$ and $[T_\eta^{FI}]$, and subsequently the reverse matrices for knot removal, $[T_\xi^{IF}]$ and $[T_\eta^{IF}]$.

We assume that in the global numbering, we first count the control points on η and then on ξ. We name this global numbering ξη.

i. Refinement on Heta

In the following figure, per axis numbering is in black and the global numbering in green letters.



Then the global numbering of the control points would be like:

$$\underset{(mx3)}{\left\{P^{I}_{\xi\eta\ \text{numbering}}\right\}} = \left[\begin{array}{c:c} \xi_1 & \underset{(m_2x3)}{\left\{\begin{matrix}P_1\\P_2\\\vdots\\P_{m_2}\end{matrix}\right\}} \\ \hdashline \xi_2 & \underset{(m_2x3)}{\left\{\begin{matrix}P_{m_2+1}\\P_{m_2+2}\\\vdots\\P_{2m_2}\end{matrix}\right\}} \\ \hdashline \xi_3,\cdots,\xi_{m_1-1} & P_{2m_2+1},\cdots,P_{(m_1-1)m_2} \\ \hdashline \xi_{m_1} & \underset{(m_2x3)}{\left\{\begin{matrix}P_{(m_1-1)m_2+1}\\P_{(m_1-1)m_2+2}\\\vdots\\P_{m_1m_2}\end{matrix}\right\}} \end{array}\right]_{(mx3)}$$

As the Control Point coordinates are arrayed like that, we can perform the refinement on axis η by multiplying with the matrix:

$$\underset{((m_1n_2)xm)}{\left[T^{IF,tot}_{\eta}\right]} = \underset{(m_1n_2)x(m_1m_2)}{\begin{bmatrix} \underset{(n_2xm_2)}{\left[T^{IF}_{\eta}\right]} & [0] & \cdots & [0] \\ [0] & \underset{(n_2xm_2)}{\left[T^{IF}_{\eta}\right]} & [0] & \vdots \\ \vdots & [0] & \ddots & [0] \\ [0] & \cdots & [0] & \underset{(n_2xm_2)}{\left[T^{IF}_{\eta}\right]} \end{bmatrix}} = \underset{(m_1xm_1)}{[I]} \otimes \underset{(n_2xm_2)}{\left[T^{IF}_{\eta}\right]}$$

The refinement on η for all control points is performed at once as:

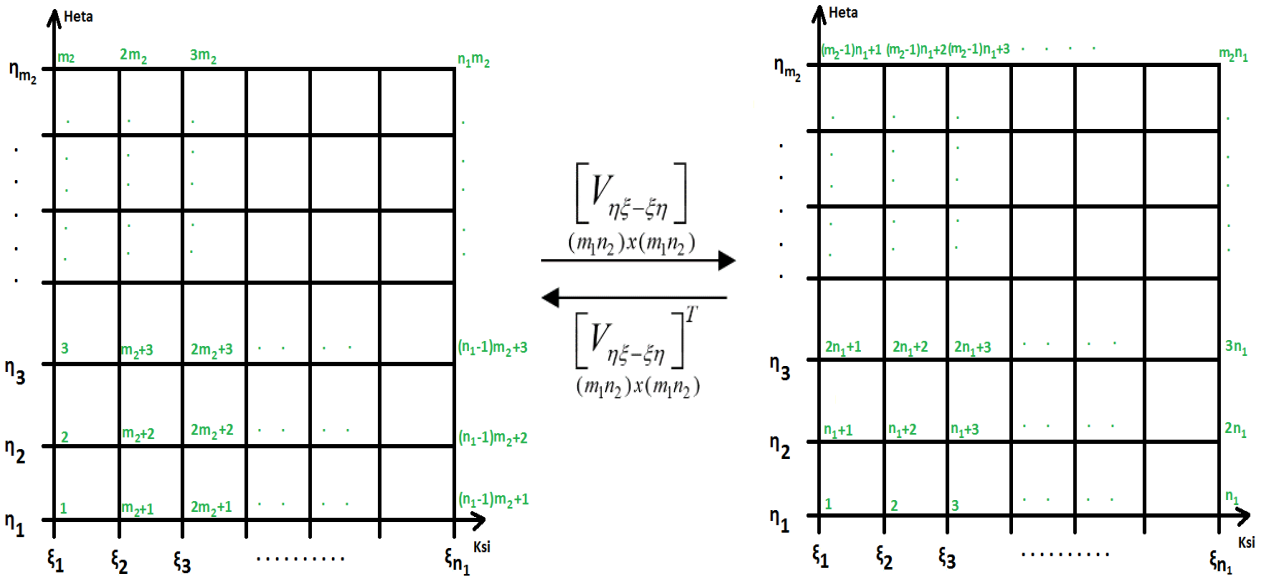
$$\underset{((m_1n_2)x3)}{\left\{P^{\xi\ \text{refined}}_{\xi\eta\ \text{numbering}}\right\}} = \underset{((m_1n_2)xm)}{\left[T^{IF,tot}_{\eta}\right]} \underset{(mx3)}{\left\{P^{F,\ \xi,\eta\ \text{refined}}_{\xi\eta\ \text{numbering}}\right\}}$$

ii. Changing the global numbering, ξη to ηξ

To proceed in the same way to reverse refine on ξ, we need to change the global numbering from ξη to ηξ. We therefore introduce a permutation matrix $\underset{(m_1n_2xm_1n_2)}{\left[V_{\eta\xi-\xi\eta}\right]}$, which changes the rows in such a way that after the permutation we first count ξ and then η.

ξη Numbering

ηξ Numbering



And the permutation matrix's numerical content:

$$\left[V_{\eta\xi-\xi\eta}\right]_{((m_1n_2)x(m_1n_2))} =$$

	1	2	3	⋯	n_2	n_2+1		⋯	$2n_2$	⋯	$(m_1-1)n_2+1$		⋯	m_1n_2
1	1	0	0	⋯	0	0	0	⋯	0	0 0 ⋯ 0	0	0	⋯	0 0
2	0	0	0	⋯	0	1	0	⋯	0	0 0 ⋯ 0	0	0	⋯	0 0
3	0	0	0	⋯	0	0	0	⋯	0	1 0 ⋯ 0	0	0	⋯	0 0
⋮	⋮	⋮	⋮	⋱	⋮	0	0	⋯	0	⋮ ⋮ ⋱ ⋮	⋮	⋮	⋱	⋮ ⋮
m_1	0	0	0	⋯	0	0	0	⋯	0	0 0 ⋯ 0	1	0	⋯	0 0
m_1+1	0	1	0	⋯	0	0	0	⋯	0	0 0 0 ⋯ 0	0	0	⋯	0
n_1+2	0	0	0	⋯	0	0	1	⋯	0	0 0 0 ⋯ 0	0	0	⋯	0
⋮	⋮	⋮	⋮	⋱	⋮	⋮	⋮	⋱	⋮	⋮ ⋮ ⋮ ⋱ ⋮	⋮	⋮	⋱	⋮
$2m_1$	0	0	0	⋯	0	0	0	⋯	0	0 0 0 ⋯ 0	0	1	⋯	0
⋮			⋱					⋱		⋱			⋱	
$(n_2-1)m_1+1$	0	0	0	⋯	1	0	0	⋯	0	0 0 ⋯ 0	0	0	⋯	0 0
⋮	⋮	⋮	⋮	⋱	⋮	⋮	⋮	⋱	⋮	⋮ ⋮ ⋱ ⋮	⋮	⋮	⋱	⋮ ⋮
n_2m_1	0	0	0	⋯	0	0	0	⋯	0	0 0 ⋯ 0	0	0	⋯	0 1

We also observe that the permutation matrix satisfies all the orthonormal tests and therefore $\left[V_{\eta\xi-\xi\eta}\right]^{-1}_{(m_1n_2)x(m_1n_2)} = \left[V_{\eta\xi-\xi\eta}\right]^{T}_{(m_1n_2)x(m_1n_2)}$.

After the multiplication, we now have the control points' coordinates with a global ξη numbering.

$$\left\{P^{\xi\text{ refined}}_{\eta\xi\text{ numbering}}\right\}_{((m_1n_2)x3)} = \left[V_{\eta\xi-\xi\eta}\right]_{(m_1n_2)x(m_1n_2)} \left\{P^{\xi\text{ refined}}_{\xi\eta\text{ numbering}}\right\}_{((m_1n_2)x3)}$$

iii. Refinement on ξ

Now the global numbering of the control points would be like:

$$\underset{((m_1n_2)x3)}{\left\{P^{\xi\ \text{refined}}_{\eta\xi\ \text{numbering}}\right\}} = \underset{(m_1n_2x3)}{\left[\begin{array}{c|c} \eta_1 & \underset{(m_1x2)}{\left\{\begin{matrix}P_1\\P_2\\\vdots\\P_{m_1}\end{matrix}\right\}} \\ \hline \eta_2 & \underset{(m_1x3)}{\left\{\begin{matrix}P_{m_1+1}\\P_{m_1+2}\\\vdots\\P_{2m_1}\end{matrix}\right\}} \\ \hline \eta_3,\cdots,\eta_{n_2-1} & P_{2m_1+1},\cdots,P_{(n_2-1)m_1} \\ \hline \eta_{n_2} & \underset{(m_1x3)}{\left\{\begin{matrix}P_{(n_2-1)m_1+1}\\P_{(n_2-1)m_1+2}\\\vdots\\P_{n_2m_1}\end{matrix}\right\}} \end{array}\right]}$$

At this point we can perform the refinement on ξ in the same way as we did in η. We multiply the control points' coordinates with the matrix:

$$\underset{(n\ x\ m_1n_2)}{\left[T^{IF,tot}_{\xi}\right]} = \underset{(n\ x\ (m_1n_2))}{\begin{bmatrix} \underset{(n_1xm_1)}{\left[T^{IF}_{\xi}\right]} & [0] & \cdots & [0] \\ [0] & \underset{(n_1xm_1)}{\left[T^{IF}_{\xi}\right]} & [0] & \vdots \\ \vdots & [0] & \ddots & [0] \\ [0] & \cdots & [0] & \underset{(n_1xm_1)}{\left[T^{IF}_{\xi}\right]} \end{bmatrix}} = \underset{(n_2xn_2)}{[I]} \otimes \underset{(n_1xm_1)}{\left[T^{IF}_{\xi}\right]}$$

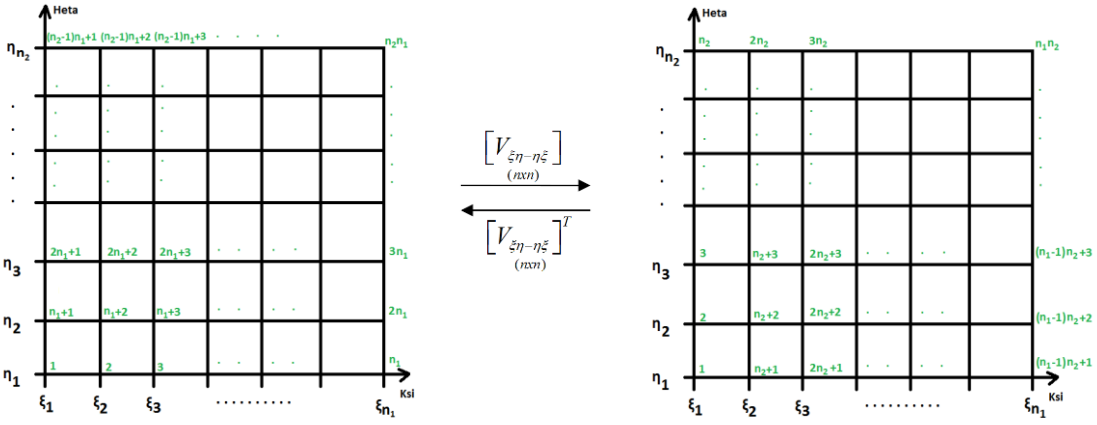
$$\underset{(nx3)}{\left\{P^{I}_{\eta\xi\ \text{numbering}}\right\}} = \underset{(n\ x\ (m_1n_2))}{\left[T^{IF,tot}_{\xi}\right]} \underset{((m_1n_2)x3)}{\left\{P^{\xi\ \text{refined}}_{\eta\xi\ \text{numbering}}\right\}}$$

Now the refinement is complete and all we have to do is return the control point coordinates to the original global ξη numbering.

iv. Changing the global numbering, ηξ to ξη.

We observe that we cannot use the $\left[V_{\eta\xi-\xi\eta}\right]^{T}_{(m_1 n_2)x(m_1 n_2)}$ matrix we built before as we now have only n control points. We build a new permutation matrix $\left[V_{\xi\eta-\eta\xi}\right]_{(nxn)}$.

ηξ Numbering **ξη** Numbering



And the permutation matrix's numerical content:

$$\left[V_{\xi\eta-\eta\xi}\right]_{(nxn)}=\left[\begin{array}{c|ccccc|ccccc|ccccc|ccccc}
 & 1 & 2 & 3 & \cdots & n_1 & n_1+1 & & \cdots & & 2n_1 & & & \cdots & & & (n_2-1)n_1+1 & & \cdots & & n_2n_1 \\
\hline
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & & 0 & 0 & \cdots & 0 & 0 \\
2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & & 0 & 0 & \cdots & 0 & 0 \\
3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 & & 0 & 0 & \cdots & 0 & 0 \\
\vdots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & \cdots & \cdots & \cdots & \cdots & & \cdots & \cdots & \cdots & \cdots & \cdots \\
n_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & & 1 & 0 & \cdots & 0 & 0 \\
\hline
n_2+1 & 0 & 1 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 \\
n_2+2 & 0 & 0 & 0 & \cdots & 0 & 0 & 1 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 \\
\vdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
2n_2 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 1 & 0 & \cdots & 0 \\
\hline
\cdots & & & \cdots & & & & & \cdots & & & & & \cdots & & & & & \cdots & & \\
\hline
(n_1-1)n_2+1 & 0 & 0 & 0 & \cdots & 1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & & 0 & 0 & \cdots & 0 & 0 \\
\vdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & & \cdots & \cdots & \cdots & \cdots & \cdots \\
n_1n_2 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 0 & & 0 & 0 & \cdots & 0 & 1
\end{array}\right]$$

This permutation matrix also satisfies all the orthonormal tests and therefore

$$\left[V_{\xi\eta-\eta\xi}\right]^{-1}_{(nxn)}=\left[V_{\xi\eta-\eta\xi}\right]^{T}_{(nxn)}$$

Finally after the multiplication $\left\{P^{I}\right\}_{(nx3)}=\left\{P^{I}_{\xi\eta\ \text{numbering}}\right\}_{(nx3)}=\left[V_{\xi\eta-\eta\xi}\right]_{(nxn)}\cdot\left\{P^{I}_{\eta\xi\ \text{numbering}}\right\}_{(nx3)}$ we have the coordinates of all the control points after the reverse refinement in the original global numbering ξη.

$$\underset{(nx3)}{\left\{P^{I}\right\}} = \underset{(nx3)}{\left\{P^{I}_{\xi\eta\ \text{numbering}}\right\}} = \left[\begin{array}{c:c} \xi_1 & \underset{(n_2 x 3)}{\left\{\begin{matrix} P_1 \\ P_2 \\ \vdots \\ P_{n_2} \end{matrix}\right\}} \\ \hdashline \xi_2 & \underset{(n_2 x 3)}{\left\{\begin{matrix} P_{n_2+1} \\ P_{n_2+2} \\ \vdots \\ P_{2n_2} \end{matrix}\right\}} \\ \hdashline \xi_3, \cdots, \xi_{n_1-1} & P_{2n_2+1}, \cdots, P_{(n_1-1)n_2} \\ \hdashline \xi_{n_1} & \underset{(n_2 x 3)}{\left\{\begin{matrix} P_{(n_1-1)n_2+1} \\ P_{(n_1-1)n_2+2} \\ \vdots \\ P_{n_1 n_2} \end{matrix}\right\}} \end{array}\right]_{(n_1 n_2 x 3)}$$

v. The whole procedure of control points' coordinates 2D refinement, in a glance.

We can summarize the refinement procedure getting the new control point coordinates as

$$\underset{(nx3)}{\left\{P^{I}\right\}} = \underbrace{\underset{(nxn)}{\left[V_{\xi\eta-\eta\xi}\right]}\underset{(n\ x\ (m_1 n_2))}{\left[T^{IF,tot}_{\xi}\right]}\underset{((m_1 n_2)x(m_1 n_2))}{\left[V_{\eta\xi-\xi\eta}\right]}\underset{((m_1 n_2)xm)}{\left[T^{IF,tot}_{\eta}\right]}}_{\underset{(nxm)}{\left[T^{IF,2D}\right]}}\underset{(mx3)}{\left\{P^{F}\right\}} \Rightarrow$$

$$\underset{(nx3)}{\left\{P^{I}\right\}} = \underset{(nxm)}{\left[T^{IF,2D}\right]}\underset{(mx3)}{\left\{P^{F}\right\}}$$

6.2.2.2 Refinement of the Stiffness Matrix 2D

The surface is given by the equation $S(\xi,\eta) = \underset{(1xn)}{\{R\}^T}\ \underset{(nx2)}{\{P\}}$ and the initial and final surfaces are the same in physical space: $S^I(\xi,\eta) = S^F(\xi,\eta)$

Jacobian [J]

The Jacobian of the transformation from physical (Cartesian) coordinates to the parametric coordinates is

$$\underset{(2x2)}{[J^F]} = \underset{(2xm)}{[D_R^F]^T} \cdot \underset{(mx2)}{\{P^F\}} = \begin{bmatrix} \underset{(1xm)}{\{R^F_{\xi(\xi,\eta)}\}^T}\ \underset{(mx2)}{\{P^F\}} \\ \underset{(1xm)}{\{R^F_{\eta(\xi,\eta)}\}^T}\ \underset{(mx2)}{\{P^F\}} \end{bmatrix} = \begin{bmatrix} \underset{(1x2)}{[S^F_{\xi(\xi,\eta)}]} \\ \underset{(1x2)}{[S^F_{\eta(\xi,\eta)}]} \end{bmatrix}_{(2x2)} = \begin{bmatrix} \underset{(1x2)}{[S^I_{\xi(\xi,\eta)}]} \\ \underset{(1x2)}{[S^I_{\eta(\xi,\eta)}]} \end{bmatrix}_{(2x2)} = \underset{(2x2)}{[J^I]}$$

So the Jacobian stays the same in the knot refinement.

Deformation Matrix [B_1]

As the Jacobian remains the same after the refinement, $J^I_{ij} = J^F_{ij} = J_{ij}$

$$\underset{(3x4)}{\left[B_1^F(\xi,\eta)\right]} = \frac{1}{\det\left(\underset{(2x2)}{[J]}\right)} \underbrace{\begin{bmatrix} J_{22} & -J_{12} & 0 & 0 \\ 0 & 0 & -J_{21} & J_{11} \\ -J_{21} & J_{11} & J_{22} & -J_{12} \end{bmatrix}}_{(3x4)} = \underset{(3x4)}{\left[B_1^I(\xi,\eta)\right]}$$

Deformation Matrix [B_2]

Based on the 1D relation of the initial and final deformation matrix, $\underset{(1xm)}{[B^F_{2(\xi)}]} = \underset{(1xn)}{[B^I_{2(\xi)}]}\underset{(nxm)}{\left[T^{IF}\right]}$

we prove a similar relation for the 2D initial and final deformation matrix $[B_2]$:

$$\underset{(4x2m)}{[B^F_{2(\xi,\eta)}]} = \underset{(4x2n)}{[B^I_{2(\xi,\eta)}]}\underset{(2nx2m)}{\left[T^{IF,B}\right]} \text{ and } \underset{(4x2n)}{[B^I_{2(\xi,\eta)}]} = \underset{(4x2m)}{[B^F_{2(\xi,\eta)}]}\underset{(2mx2n)}{\left[T^{FI,B}\right]}$$

where

$$\underset{(2mx2n)}{\left[T^{IF,B}\right]}=\left[\begin{array}{cc|cc|c|cc} T_{11}^{IF,2D} & 0 & T_{12}^{IF,2D} & 0 & \cdots & T_{1m}^{IF,2D} & 0 \\ 0 & T_{11}^{IF,2D} & 0 & T_{12}^{IF,2D} & \cdots & 0 & T_{1m}^{IF,2D} \\ \hline T_{21}^{IF,2D} & 0 & T_{22}^{IF,2D} & 0 & \cdots & T_{2m}^{IF,2D} & 0 \\ 0 & T_{21}^{IF,2D} & 0 & T_{22}^{IF,2D} & \cdots & 0 & T_{2m}^{IF,2D} \\ \hline \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \hline T_{n1}^{IF,2D} & 0 & T_{n2}^{IF,2D} & 0 & \cdots & T_{nm}^{IF,2D} & 0 \\ 0 & T_{n1}^{IF,2D} & 0 & \underset{(2mx2n)}{T_{n2}^{IF,2D}} & \cdots & 0 & T_{nm}^{IF,2D} \end{array}\right]$$

and $\underset{(nxm)}{\left[T^{IF,2D}\right]}$ is the control point transformation matrix we formed in Ch. 6.2.2.1.

Indeed, by substituting $\underset{(2mx2n)}{\left[T^{IF,B}\right]}$ in $\underset{(4x2m)}{[B^{F}_{2(\xi,\eta)}]}=\underset{(4x2n)}{[B^{I}_{2(\xi,\eta)}]}\underset{(2nx2m)}{\left[T^{IF,B}\right]}$ we get

$$\underset{(4x2n)}{[B^{I}_{2(\xi,\eta)}]}\underset{(2nx2m)}{\left[T^{IF,B}\right]}=$$

$$=\left[\begin{array}{cccccc} R^{I}_{1,\xi} & 0 & R^{I}_{2,\xi} & 0 & \cdots & R^{I}_{n,\xi} & 0 \\ R^{I}_{1,\eta} & 0 & R^{I}_{2,\eta} & 0 & \cdots & R^{I}_{n,\eta} & 0 \\ 0 & R^{I}_{1,\xi} & 0 & R^{I}_{2,\xi} & \cdots & 0 & R^{I}_{n,\xi} \\ 0 & R^{I}_{1,\eta} & 0 & \underset{(4x2n)}{R^{I}_{2,\eta}} & \cdots & 0 & R^{I}_{n,\eta} \end{array}\right]\left[\begin{array}{cc|cc|c|cc} T_{11}^{IF,2D} & 0 & T_{12}^{IF,2D} & 0 & \cdots & T_{1m}^{IF,2D} & 0 \\ 0 & T_{11}^{IF,2D} & 0 & T_{12}^{IF,2D} & \cdots & 0 & T_{1m}^{IF,2D} \\ \hline T_{21}^{IF,2D} & 0 & T_{22}^{IF,2D} & 0 & \cdots & T_{2m}^{IF,2D} & 0 \\ 0 & T_{21}^{IF,2D} & 0 & T_{22}^{IF,2D} & \cdots & 0 & T_{2m}^{IF,2D} \\ \hline \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \hline T_{n1}^{IF,2D} & 0 & T_{n2}^{IF,2D} & 0 & \cdots & T_{nm}^{IF,2D} & 0 \\ 0 & T_{n1}^{IF,2D} & 0 & \underset{(2mx2n)}{T_{n2}^{IF,2D}} & \cdots & 0 & T_{nm}^{IF,2D} \end{array}\right]$$

$$=\left[\begin{array}{cccccc} \sum_{i=1}^{n} T_{i1}R^{I}_{i,} & 0 & \sum_{i=1}^{n} T_{i2}R^{I}_{i,\xi} & 0 & \cdots & \sum_{i=1}^{n} T_{im}R^{I}_{i,\xi} & 0 \\ \sum_{i=1}^{n} T_{i1}R^{I}_{i,\eta} & 0 & \sum_{i=1}^{n} T_{i2}R^{I}_{i,\eta} & 0 & \cdots & \sum_{i=1}^{n} T_{im}R^{I}_{i,\eta} & 0 \\ 0 & \sum_{i=1}^{n} T_{i1}R^{I}_{i,\xi} & 0 & \sum_{i=1}^{n} T_{i2}R^{I}_{i,\xi} & \cdots & 0 & \sum_{i=1}^{n} T_{im}R^{I}_{i,\xi} \\ 0 & \sum_{i=1}^{n} T_{i1}R^{I}_{i,\eta} & 0 & \underset{(4x2n)}{\sum_{i=1}^{n} T_{i2}R^{I}_{i,\eta}} & \cdots & 0 & \sum_{i=1}^{n} T_{im}R^{I}_{i,\eta} \end{array}\right]$$

$$=\left[\begin{array}{cccccc} R^{F}_{1,\xi} & 0 & R^{F}_{2,\xi} & 0 & \cdots & R^{F}_{n,\xi} & 0 \\ R^{F}_{1,\eta} & 0 & R^{F}_{2,\eta} & 0 & \cdots & R^{F}_{n,\eta} & 0 \\ 0 & R^{F}_{1,\xi} & 0 & R^{F}_{2,\xi} & \cdots & 0 & R^{F}_{n,\xi} \\ 0 & R^{F}_{1,\eta} & 0 & \underset{(4x2m)}{R^{F}_{2,\eta}} & \cdots & 0 & R^{F}_{n,\eta} \end{array}\right]$$

$$=\underset{(4x2m)}{[B^{F}_{2(\xi,\eta)}]}$$

Deformation Matrix $[B]=[B_1][B_2]$

$$\underset{(3x2m)}{[B^F_{(\xi,\eta)}]}=\underset{(3x4)}{[B^F_{1(\xi,\eta)}]}\underset{(4x2m)}{[B^F_{2(\xi,\eta)}]}=\underset{(3x4)}{[B^I_{1(\xi,\eta)}]}\underset{(4x2n)}{[B^I_{2(\xi,\eta)}]}\underset{(2nx2m)}{\left[T^{IF,B}\right]}=\underset{(3x2n)}{[B^I_{(\xi,\eta)}]}\underset{(2nx2m)}{\left[T^{IF,B}\right]}\Rightarrow \boxed{\underset{(3x2m)}{[B^F_{(\xi,\eta)}]}=\underset{(3x2n)}{[B^I_{(\xi,\eta)}]}\underset{(2nx2m)}{\left[T^{IF,B}\right]}}$$

and

$$\underset{(3x2n)}{[B^I_{(\xi,\eta)}]}=\underset{(3x4)}{[B^I_{1(\xi,\eta)}]}\underset{(4x2n)}{[B^I_{2(\xi,\eta)}]}=\underset{(3x4)}{[B^F_{1(\xi,\eta)}]}\underset{(4x2m)}{[B^F_{2(\xi,\eta)}]}\underset{(2mx2n)}{\left[T^{FI,B}\right]}=\underset{(3x2m)}{[B^F_{(\xi,\eta)}]}\underset{(2mx2n)}{\left[T^{FI,B}\right]}\Rightarrow\underset{(3x2n)}{[B^I_{(\xi,\eta)}]}=\underset{(3x2m)}{[B^F_{(\xi,\eta)}]}\underset{(2mx2n)}{\left[T^{FI,B}\right]}$$

Stiffness Matrix $[K]=[B]^T[E][B]$

$$\underset{(2mx2m)}{\left[K^F\right]}=\int_\Omega \underset{(2mx3)}{\left[B^F\right]^T}\underset{(3x3)}{[E]}\underset{(3x2m)}{\left[B^F\right]}\det[J]\,d\Omega=\int_\Omega\left(\underset{(3x2n)}{[B^I]}\underset{(2nx2m)}{\left[T^{IF,B}\right]}\right)^T\underset{(3x3)}{[E]}\left(\underset{(3x2n)}{[B^I]}\underset{(2nx2m)}{\left[T^{IF,B}\right]}\right)\det[J]\,d\Omega=$$

$$\int_\Omega \underset{(2mx2n)}{\left[T^{IF,B}\right]^T}\underset{(2nx3)}{[B^I_{(\xi,\eta)}]^T}\underset{(3x3)}{[E]}\underset{(3x2n)}{[B^I_{(\xi,\eta)}]}\underset{(2nx2m)}{\left[T^{IF,B}\right]}\det[J]\,d\Omega=$$

$$\underset{(2mx2n)}{\left[T^{IF,B}\right]^T}\int_\Omega \underset{(2nx3)}{[B^I_{(\xi,\eta)}]^T}\underset{(3x3)}{[E]}\underset{(3x2n)}{[B^I_{(\xi,\eta)}]}\det[J]\,d\Omega\cdot\underset{(2mx2n)}{\left[T^{IF,B}\right]}=$$

$$\underset{(2mx2n)}{\left[T^{IF,B}\right]^T}\underset{(2nx2n)}{\left[K^I\right]}\underset{(2nx2m)}{\left[T^{IF,B}\right]}\Rightarrow$$

$$\underset{(2mx2m)}{\left[K^F\right]}=\underset{(2mx2n)}{\left[T^{IF,B}\right]^T}\underset{(2nx2n)}{\left[K^I\right]}\underset{(2nx2m)}{\left[T^{IF,B}\right]}$$

Similarly:

$$\underset{(2nx2n)}{\left[K^I\right]}=\underset{(2nx2m)}{\left[T^{FI,B}\right]^T}\underset{(2mx2m)}{\left[K^F\right]}\underset{(2mx2n)}{\left[T^{FI,B}\right]}$$

6.2.3 Refinement of the Stiffness Matrix 3D.

6.2.3.1 Control Point Transformation Matrix.

In the same way as in 2D, the 3D control point transformation matrix will be:

$$\underset{(nx3)}{\left\{P^I\right\}} = \underbrace{\underset{(mxm)}{\left[V_{\xi\eta\zeta-\eta\zeta\xi}\right]} \underset{(mx(n_1m_2m_3))}{\left[T_\xi^{FI,tot}\right]} \underset{((n_1m_2m_3)x(n_1m_2m_3))}{\left[V_{\eta\zeta\xi-\zeta\xi\eta}\right]} \underset{((n_1m_2m_3)x(n_1m_2m_3))}{\left[T_\eta^{FI,tot}\right]} \underset{((n_1n_2m_3)x(n_1n_2m_3))}{\left[V_{\zeta\xi\eta-\xi\eta\zeta}\right]} \underset{((n_1n_2m_3)xn)}{\left[T_\zeta^{FI,tot}\right]}}_{\underset{(mxn)}{\left[T^{FI,3D}\right]}} \underset{(mx3)}{\left\{P^F\right\}} \Rightarrow$$

$$\underset{(nx3)}{\left\{P^I\right\}} = \underset{(nxm)}{\left[T^{IF,3D}\right]} \underset{(mx3)}{\left\{P^F\right\}}$$

6.2.3.2 Refinement of the Stiffness Matrix 3D.

The volume is given by the equation $V(\xi,\eta,\zeta) = \underset{(1xn)}{\{R\}^T} \underset{(nx3)}{\{P\}}$. The initial and final volume are the same in physical space: $V^I(\xi,\eta,\zeta) = V^F(\xi,\eta,\zeta)$.

Jacobian [J]

The Jacobian of the transformation from physical (Cartesian) coordinates to the parametric coordinates is

$$\underset{(3x3)}{[J^F]} = \underset{(3xm)}{[D_R^F]^T} \cdot \underset{(mx3)}{\{P^F\}} = \begin{bmatrix} \underset{(1xm)}{\left\{R^F_{\xi(\xi,\eta,\zeta)}\right\}^T} \underset{(mx3)}{\{P^F\}} \\ \underset{(1xm)}{\left\{R^F_{\eta(\xi,\eta,\zeta)}\right\}^T} \underset{(mx3)}{\{P^F\}} \\ \underset{(1xm)}{\left\{R^F_{\zeta(\xi,\eta,\zeta)}\right\}^T} \underset{(mx3)}{\{P^F\}} \end{bmatrix} = \begin{bmatrix} \underset{(1x3)}{\left[V^F_{\xi(\xi,\eta,\zeta)}\right]} \\ \underset{(1x3)}{\left[V^F_{\eta(\xi,\eta,\zeta)}\right]} \\ \underset{(1x3)}{\left[V^F_{\zeta(\xi,\eta,\zeta)}\right]} \end{bmatrix}_{(3x3)} = \begin{bmatrix} \underset{(1x3)}{\left[V^I_{\xi(\xi,\eta,\zeta)}\right]} \\ \underset{(1x3)}{\left[V^I_{\eta(\xi,\eta,\zeta)}\right]} \\ \underset{(1x3)}{\left[V^I_{\zeta(\xi,\eta,\zeta)}\right]} \end{bmatrix}_{(3x3)} = \underset{(3x3)}{[J^I]}$$

The Jacobian stays the same in knot refinement.

Deformation Matrix [B_1]

As the Jacobian remains the same after the refinement $\underset{(3x3)}{\left[J^I\right]} = \underset{(3x3)}{\left[J^F\right]} = \underset{(3x3)}{\left[J\right]}$ and

$$\underset{(6x9)}{\left[B_1^F(\xi,\eta,\zeta)\right]} = \begin{bmatrix} J^*_{11} & J^*_{12} & J^*_{13} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & J^*_{21} & J^*_{22} & J^*_{23} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & J^*_{31} & J^*_{32} & J^*_{33} \\ J^*_{21} & J^*_{22} & J^*_{23} & J^*_{11} & J^*_{12} & J^*_{13} & 0 & 0 & 0 \\ 0 & 0 & 0 & J^*_{31} & J^*_{32} & J^*_{33} & J^*_{21} & J^*_{22} & J^*_{23} \\ J^*_{31} & J^*_{32} & J^*_{33} & 0 & 0 & 0 & J^*_{11} & J^*_{12} & J^*_{13} \end{bmatrix}_{(6x9)} = \underset{(6x9)}{\left[B_1^I(\xi,\eta,\zeta)\right]}$$

$$\text{where } \underset{(3x3)}{[J]^{-1}} = \begin{bmatrix} J^*_{11} & J^*_{12} & J^*_{13} \\ J^*_{21} & J^*_{22} & J^*_{23} \\ J^*_{31} & J^*_{32} & J^*_{33} \end{bmatrix}$$

Deformation Matrix $[B_2]$

$$\underset{(9x3m)}{[B^F_{2(\xi,\eta)}]} = \underset{(9x3n)}{[B^I_{2(\xi,\eta)}]} \underset{(3nx3m)}{\left[T^{IF,B}\right]}$$

and $\underset{(9x3n)}{[B^I_{2(\xi,\eta)}]} = \underset{(9x3m)}{[B^F_{2(\xi,\eta)}]} \underset{(3mx3n)}{\left[T^{FI,B}\right]}$

In a the same way as in 2D case, the deformation transformation matrix $\underset{(3nx3m)}{\left[T^{IF,B}\right]}$ is

$$\underset{(3nx3m)}{\left[T^{IF,B}\right]} = \left[\begin{array}{ccc:ccc:c:ccc} T^{IF,3D}_{11} & 0 & 0 & T^{IF,3D}_{12} & 0 & 0 & \cdots & T^{IF,3D}_{1m} & 0 & 0 \\ 0 & T^{IF,3D}_{11} & 0 & 0 & T^{IF,3D}_{12} & 0 & \cdots & 0 & T^{IF,3D}_{1m} & 0 \\ 0 & 0 & T^{IF,3D}_{11} & 0 & 0 & T^{IF,3D}_{12} & \cdots & 0 & 0 & T^{IF,3D}_{1m} \\ \hdashline T^{IF,3D}_{21} & 0 & 0 & T^{IF,3D}_{22} & 0 & 0 & \cdots & T^{IF,3D}_{2m} & 0 & 0 \\ 0 & T^{IF,3D}_{21} & 0 & 0 & T^{IF,3D}_{22} & 0 & \cdots & 0 & T^{IF,3D}_{2m} & 0 \\ 0 & 0 & T^{IF,3D}_{21} & 0 & 0 & T^{IF,3D}_{22} & \cdots & 0 & 0 & T^{IF,3D}_{2m} \\ \hdashline \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \hdashline T^{IF,3D}_{n1} & 0 & 0 & T^{IF,3D}_{n2} & 0 & 0 & \cdots & T^{IF,3D}_{nm} & 0 & 0 \\ 0 & T^{IF,3D}_{n1} & 0 & 0 & T^{IF,3D}_{n2} & 0 & \cdots & 0 & T^{IF,3D}_{nm} & 0 \\ 0 & 0 & T^{IF,3D}_{n1} & 0 & 0 & T^{IF,3D}_{n2} & \cdots & 0 & 0 & T^{IF,3D}_{nm} \end{array}\right] \text{ and}$$

$\underset{(3nx3m)}{}$

$\underset{(nxm)}{\left[T^{IF,3D}\right]}$ is the control point transformation matrix.

Deformation Matrix $[B]=[B_1][B_2]$

$$\underset{(6x3m)}{[B^F_{(\xi,\eta,\zeta)}]} = \underset{(6x9)}{[B^F_{1(\xi,\eta,\zeta)}]} \underset{(9x3m)}{[B^F_{2(\xi,\eta,\zeta)}]} = \underset{(6x9)}{[B^I_{1(\xi,\eta,\zeta)}]} \underset{(9x3n)}{[B^I_{2(\xi,\eta,\zeta)}]} \underset{(3nx3m)}{\left[T^{IF,B}\right]} = \underset{(9x3n)}{[B^I_{(\xi,\eta,\zeta)}]} \underset{(3nx3m)}{\left[T^{IF,B}\right]} \Rightarrow \underset{(6x3m)}{[B^F_{(\xi,\eta,\zeta)}]} = \underset{(9x3n)}{[B^I_{(\xi,\eta,\zeta)}]} \underset{(3nx3m)}{\left[T^{IF,B}\right]}$$

and

$$\underset{(9x3n)}{[B^I_{(\xi,\eta,\zeta)}]} = \underset{(6x9)}{[B^I_{1(\xi,\eta,\zeta)}]} \underset{(9x3n)}{[B^I_{2(\xi,\eta,\zeta)}]} = \underset{(6x9)}{[B^F_{1(\xi,\eta,\zeta)}]} \underset{(9x3m)}{[B^F_{2(\xi,\eta,\zeta)}]} \underset{(3mx3n)}{\left[T^{FI,B}\right]} = \underset{(6x3m)}{[B^F_{(\xi,\eta,\zeta)}]} \underset{(3mx3n)}{\left[T^{FI,B}\right]} \Rightarrow \underset{(9x3n)}{[B^I_{(\xi,\eta,\zeta)}]} = \underset{(6x3m)}{[B^F_{(\xi,\eta,\zeta)}]} \underset{(3mx3n)}{\left[T^{FI,B}\right]}$$

Stiffness Matrix [K]

$$\underset{(3mx3m)}{\left[K^F\right]} = \int_{\Omega} \underset{(3mx6)}{\left[B^F\right]^T} \underset{(6x6)}{\left[E\right]} \underset{(6x3m)}{\left[B^F\right]} \det\left[J\right] d\Omega = \int_{\Omega} \left(\underset{(6x3n)}{\left[B^I\right]} \underset{(3nx3m)}{\left[T^{IF,B}\right]} \right)^T \underset{(6x6)}{\left[E\right]} \left(\underset{(6x3n)}{\left[B^I\right]} \underset{(3nx3m)}{\left[T^{IF,B}\right]} \right) \det\left[J\right] d\Omega =$$

$$\int_{\Omega} \underset{(3mx3n)}{\left[T^{IF,B}\right]^T} \underset{(3nx6)}{[B^I_{(\xi,\eta)}]^T} \underset{(6x6)}{\left[E\right]} \underset{(6x3n)}{[B^I_{(\xi,\eta)}]} \underset{(3nx3m)}{\left[T^{IF,B}\right]} \det\left[J\right] d\Omega =$$

$$\underset{(3mx3n)}{\left[T^{IF,B}\right]^T} \int_{\Omega} \underset{(3nx6)}{[B^I_{(\xi,\eta)}]^T} \underset{(6x6)}{\left[E\right]} \underset{(6x3n)}{[B^I_{(\xi,\eta)}]} \det\left[J\right] d\Omega \cdot \underset{(3mx3n)}{\left[T^{IF,B}\right]} =$$

$$\underset{(3mx3n)}{\left[T^{IF,B}\right]^T} \underset{(3nx3n)}{\left[K^I\right]} \underset{(3nx3m)}{\left[T^{IF,B}\right]} \Rightarrow$$

$$\underset{(3mx3m)}{\left[K^F\right]} = \underset{(3mx3n)}{\left[T^{IF,B}\right]^T} \underset{(3nx3n)}{\left[K^I\right]} \underset{(3nx3m)}{\left[T^{IF,B}\right]}$$

Similarly:

$$\underset{(3nx3n)}{\left[K^I\right]} = \underset{(3nx3m)}{\left[T^{FI,B}\right]^T} \underset{(3mx3m)}{\left[K^F\right]} \underset{(3mx3n)}{\left[T^{FI,B}\right]}$$

6.3 Hierarchical Formulation

To work generally in any case of 1D, 2D or 3D elasticity, for the rest of the chapter we will consider as:

n, N the initial control points and degrees of freedom,

m, N the final control points and degrees of freedom,

q, Q the new control points and degrees of freedom

It is obvious that q=m-n and Q=M-N. We will name the dimension of the elasticity matrix in each case as d.

For 1D elasticity d=1, N=n, Q=q and M=m.

For 2D elasticity d=3, N=2n, Q=2q and M=2m.

For 3D elasticity d=6, N=3n, Q=3q and M=3m.

The subscript index 1 refers to parametric direction ξ, 2 to η and 3 to ζ.

Thus $n_1 \times n_2 \times n_3=n$, $q_1 \times q_2 \times q_3=q$, $m_1 \times m_2 \times m_3=m$.

6.3.1 Hierarchical Deformation Matrix

For the new $\underset{(MxM)}{[K]}$ we build a permutation matrix $\underset{(MxM)}{[V]}$ so that the degrees of freedom and the control points inserted with the knot refinement are transferred at the end. From the knot value insertion, many degrees of freedom are influenced, more than just the **Q=M-N** that were inserted. Nevertheless, we use the permutation matrix to transfer (M-N) degrees of freedom to the end of the stiffness matrix. These will be called the **new degrees of freedom** and the **respective q=m-n control points the new control points**. We can even do it by moving non-influenced control points to the end, but then the error at the calculation of the stiffness matrix would be greater. In order to do that in one step, we modify the transformation matrix to get us from the initial degrees of freedom to the final modified degrees of freedom.

$$\underset{(NxM)}{\left[T^m\right]}=\underset{(NxM)}{\left[T^{IF}\right]}\cdot\underset{(MxM)}{[V]}^T$$

We can thus partition the transformation matrix in two areas, the enriched area $\underset{(NxN)}{[T_e]}$ referring to the existing enriched (influenced or not influenced) degrees of freedom and the new area $\underset{(NxQ)}{[T_n]}$ referring to the new degrees of freedom.

$$\underset{(NxM)}{\left[T^m\right]}=\underset{(NxM)}{\left[\underset{(NxN)}{\left[T_e^m\right]}\quad\underset{(NxQ)}{\left[T_n^m\right]}\right]}$$

Subsequently,

$$\underset{(dxM)}{\left[B^F\right]}=\underset{(dxN)}{\left[B^I\right]}\underset{(NxM)}{\left[T^m\right]}=\underset{(dxN)}{\left[B^I\right]}\underset{(NxM)}{\left[\underset{(NxN)}{\left[T_e^m\right]}\quad\underset{(NxQ)}{\left[T_n^m\right]}\right]}=\left[\underset{(dxN)}{\left[B^I\right]}\underset{(NxN)}{\left[T_e^m\right]}\quad\underbrace{\underset{(dxN)}{\left[B^I\right]}\underset{(NxQ)}{\left[T_n^m\right]}}_{\underset{(dxQ)}{\left[\Delta B_n\right]}}\right]=\left[\underset{(dxN)}{\left[B^I\right]}\underset{(NxN)}{\left[T_e^m\right]}\quad\underset{(dxQ)}{\left[\Delta B_n\right]}\right]$$

But we want to express $[B^F]$ as a sum of $[B^I]$ and another term. We can proceed by expressing $\left[T_e^m\right]$ in the following way:

$$\underset{(dxN)}{\left[B^I\right]}\underset{(NxN)}{\left[T_e^m\right]}=\underset{(dxN)}{\left[B^I\right]}\left(\underset{(NxN)}{\left[I\right]}+\underset{(NxN)}{\left[T_e^m\right]}-\underset{(NxN)}{\left[I\right]}\right)=\underset{(dxN)}{\left[B^I\right]}+\underset{(dxN)}{\left[B^I\right]}\left(\underset{(NxN)}{\left[T_e^m\right]}-\underset{(NxN)}{\left[I\right]}\right)=\underset{(dxN)}{\left[B^I\right]}+\underset{(dxN)}{\left[\Delta B_e\right]}$$

Finally we can write that

$$\underset{(dxM)}{\left[B^F\right]}=\left[\underset{(dxN)}{\left[B^I\right]}+\underset{(dxN)}{\left[\Delta B_e\right]}\quad\underset{(dxQ)}{\left[\Delta B_n\right]}\right]$$

where:

$\underset{(dxN)}{\left[B^I\right]}$ the initial deformation matrix.

$$\underset{(dxN)}{\left[\Delta B_e\right]}=\underset{(dxN)}{\left[B^I\right]}\left(\underset{(NxN)}{\left[T_e^m\right]}-\underset{(NxN)}{\left[I\right]}\right)$$

$$\underset{(dxQ)}{\left[\Delta B_n\right]}=\underset{(dxN)}{\left[B^I\right]}\underset{(NxQ)}{\left[T_n^m\right]}$$

$$\underset{(NxM)}{\left[T^m\right]}=\underset{(NxM)}{\left[\underset{(NxN)}{\left[T_e^m\right]}\quad\underset{(NxQ)}{\left[T_n^m\right]}\right]}$$

6.3.2 Hierarchical Stiffness Matrix

$$\underset{(MxM)}{\left[K^F\right]} = \int_{\Omega} \underset{(Mxd)}{\left[B^F\right]^T} \underset{(dxd)}{[E]} \underset{(dxM)}{\left[B^F\right]} d\Omega =$$

$$\left(\int_{\Omega} \left[\underset{(dxN)}{\left[B^I\right]} + \underset{(dxN)}{\left[\Delta B_e\right]} \quad \underset{(dxQ)}{\left[\Delta B_n\right]} \right]^T \underset{(dxd)}{[E]} \left[\underset{(dxN)}{\left[B^I\right]} + \underset{(dxN)}{\left[\Delta B_e\right]} \quad \underset{(dxQ)}{\left[\Delta B_n\right]} \right] d\Omega \right) =$$

$$\int_{\Omega} \begin{bmatrix} \underset{(Nxd)}{\left[B^I\right]^T} + \underset{(Nxd)}{\left[\Delta B_e\right]^T} \\ \underset{(Qxd)}{\left[\Delta B_n\right]^T} \end{bmatrix} \underset{(dxd)}{[E]} \left[\underset{(dxN)}{\left[B^I\right]} + \underset{(dxN)}{\left[\Delta B_e\right]} \quad \underset{(dxQ)}{\left[\Delta B_n\right]} \right] d\Omega =$$

$$\int_{\Omega} \begin{bmatrix} \underset{(Nxd)}{\left[B^I\right]^T} + \underset{(Nxd)}{\left[\Delta B_e\right]^T} \\ \underset{(Qxd)}{\left[\Delta B_n\right]^T} \end{bmatrix} \underset{(dxd)}{[E]} \left[\underset{(dxN)}{\left[B^I\right]} + \underset{(dxN)}{\left[\Delta B_e\right]} \quad \underset{(dxQ)}{\left[\Delta B_n\right]} \right] d\Omega \Rightarrow$$

$$\underset{(MxM)}{\left[K^F\right]} = \int_{\Omega} \begin{pmatrix} \left(\underset{(Nxd)}{\left[B^I\right]^T} + \underset{(Nxd)}{\left[\Delta B_e\right]^T} \right) \underset{(dxd)}{[E]} \left(\underset{(dxN)}{\left[B^I\right]} + \underset{(dxN)}{\left[\Delta B_e\right]} \right) & \left(\underset{(Nxd)}{\left[B^I\right]^T} + \underset{(Nxd)}{\left[\Delta B_e\right]^T} \right) \underset{(dxd)}{[E]} \underset{(dxQ)}{\left[\Delta B_n\right]} \\ \underset{(Qxd)}{\left[\Delta B_n\right]^T} \underset{(dxd)}{[E]} \left(\underset{(dxN)}{\left[B^I\right]} + \underset{(dxN)}{\left[\Delta B_e\right]} \right) & \underset{(Qxd)}{\left[\Delta B_n\right]^T} \underset{(dxd)}{[E]} \underset{(dxQ)}{\left[\Delta B_n\right]} \end{pmatrix} d\Omega$$

We substitute $\underset{(dxQ)}{\left[\Delta B_n\right]} = \underset{(dxN)}{\left[B^I\right]} \underset{(NxQ)}{\left[T_n^m\right]}$ and $\underset{(dxN)}{\left[\Delta B_e\right]} = \underset{(dxN)}{\left[B^I\right]} \left(\underset{(NxN)}{\left[T_e^m\right]} - \underset{(NxN)}{[I]} \right)$, and do the multiplications and sums.

- $\underset{(NxN)}{\left[K_{ee}^{F}\right]}$

$$\underset{(NxN)}{\left[K_{ee}^{F}\right]} =$$

$$\underbrace{\int_{\Omega} \underset{(Nxd)}{\left[B^{I}\right]^{T}} \underset{(dxd)}{[E]} \underset{(dxN)}{\left[B^{I}\right]} d\Omega}_{\underset{(NxN)}{\left[K^{I}\right]}} + \int_{\Omega} \underset{(Nxd)}{\left[B^{I}\right]^{T}} \underset{(dxd)}{[E]} \underset{(dxN)}{[\Delta B_e]} d\Omega + \int_{\Omega} \underset{(Nxd)}{[\Delta B_e]^{T}} \underset{(dxd)}{[E]} \underset{(dxN)}{\left[B^{I}\right]} d\Omega + \int_{\Omega} \underset{(Nxd)}{[\Delta B_e]^{T}} \underset{(dxd)}{[E]} \underset{(dxN)}{[\Delta B_e]} d\Omega =$$

$$\underset{(NxN)}{\left[K^{I}\right]} + \underbrace{\int_{\Omega} \underset{(Nxd)}{\left[B^{I}\right]^{T}} \underset{(dxd)}{[E]} \underset{(dxN)}{\left[B^{I}\right]} d\Omega}_{\underset{(NxN)}{\left[K^{I}\right]}} \left(\underset{(NxN)}{\left[T_e^{m}\right]} - \underset{(NxN)}{[I]} \right) + \left(\underset{(NxN)}{\left[T_e^{m}\right]^{T}} - \underset{(NxN)}{[I]} \right) \underbrace{\int_{\Omega} \underset{(Nxd)}{\left[B^{I}\right]^{T}} \underset{(dxd)}{[E]} \underset{(dxN)}{\left[B^{I}\right]} d\Omega}_{\underset{(NxN)}{\left[K^{I}\right]}} +$$

$$+ \left(\underset{(NxN)}{\left[T_e^{m}\right]^{T}} - \underset{(NxN)}{[I]} \right) \underbrace{\int_{\Omega} \underset{(Nxd)}{\left[B^{I}\right]^{T}} \underset{(dxd)}{[E]} \underset{(dxN)}{\left[B^{I}\right]} d\Omega}_{\underset{(NxN)}{\left[K^{I}\right]}} \left(\underset{(NxN)}{\left[T_e^{m}\right]} - \underset{(NxN)}{[I]} \right) =$$

$$\underset{(NxN)}{\left[K^{I}\right]} + \underbrace{\underset{(NxN)}{\left[K^{I}\right]} \left(\underset{(NxN)}{\left[T_e^{m}\right]} - \underset{(NxN)}{[I]} \right) + \left(\underset{(NxN)}{\left[T_e^{m}\right]^{T}} - \underset{(NxN)}{[I]} \right) \underset{(NxN)}{\left[K^{I}\right]} + \left(\underset{(NxN)}{\left[T_e^{m}\right]^{T}} - \underset{(NxN)}{[I]} \right) \underset{(NxN)}{\left[K^{I}\right]} \left(\underset{(NxN)}{\left[T_e^{m}\right]} - \underset{(NxN)}{[I]} \right)}_{\underset{(NxN)}{[\delta K]}} \Rightarrow$$

$$\underset{(NxN)}{\left[K_{ee}^{F}\right]} = \underset{(NxN)}{\left[K^{I}\right]} + \underset{(NxN)}{[\delta K]}$$

We can get a better expression for $\underset{(NxN)}{[\delta K]}$ by doing the multiplications on the matrices,

$$\underset{(NxN)}{[\delta K]} =$$

$$\underset{(NxN)}{\left[K^{I}\right]} \left(\underset{(NxN)}{\left[T_e^{m}\right]} - \underset{(NxN)}{[I]} \right) + \left(\underset{(NxN)}{\left[T_e^{m}\right]^{T}} - \underset{(NxN)}{[I]} \right) \underset{(NxN)}{\left[K^{I}\right]} + \left(\underset{(NxN)}{\left[T_e^{m}\right]^{T}} - \underset{(NxN)}{[I]} \right) \underset{(NxN)}{\left[K^{I}\right]} \left(\underset{(NxN)}{\left[T_e^{m}\right]} - \underset{(NxN)}{[I]} \right) =$$

$$\underset{(NxN)}{\left[K^{I}\right]} \left(\underset{(NxN)}{\left[T_e^{m}\right]} - \underset{(NxN)}{[I]} \right) + \left(\underset{(NxN)}{\left[T_e^{m}\right]^{T}} - \underset{(NxN)}{[I]} \right) \underset{(NxN)}{\left[K^{I}\right]} \left(\underset{(NxN)}{[I]} + \left(\underset{(NxN)}{\left[T_e^{m}\right]} - \underset{(NxN)}{[I]} \right) \right) =$$

$$\underset{(NxN)}{\left[K^{I}\right]} \left(\underset{(NxN)}{\left[T_e^{m}\right]} - \underset{(NxN)}{[I]} \right) + \left(\underset{(NxN)}{\left[T_e^{m}\right]^{T}} - \underset{(NxN)}{[I]} \right) \underset{(NxN)}{\left[K^{I}\right]} \underset{(NxN)}{\left[T_e^{m}\right]} =$$

$$\underset{(NxN)}{\left[K^{I}\right]} \underset{(NxN)}{\left[T_e^{m}\right]} - \underset{(NxN)}{\left[K^{I}\right]} + \underset{(NxN)}{\left[T_e^{m}\right]^{T}} \underset{(NxN)}{\left[K^{I}\right]} \underset{(NxN)}{\left[T_e^{m}\right]} - \underset{(NxN)}{\left[K^{I}\right]} \underset{(NxN)}{\left[T_e^{m}\right]} \Rightarrow$$

$$\underset{(NxN)}{[\delta K]} = \underset{(NxN)}{\left[T_e^{m}\right]^{T}} \underset{(NxN)}{\left[K^{I}\right]} \underset{(NxN)}{\left[T_e^{m}\right]} - \underset{(NxN)}{\left[K^{I}\right]}$$

- $\underset{(NxQ)}{\left[K_{en}^{F}\right]}$

$$\underset{(NxQ)}{\left[K_{en}^{F}\right]}=$$

$$\int_{\Omega}\left(\underset{(Nxd)}{\left[B^{I}\right]}^{T}+\underset{(Nxd)}{\left[\Delta B_{e}\right]}^{T}\right)\underset{(dxd)}{\left[E\right]}\underset{(dxQ)}{\left[\Delta B_{n}\right]}d\Omega=$$

$$\int_{\Omega}\underset{(Nxd)}{\left[\Delta B_{e}\right]}^{T}\underset{(dxd)}{\left[E\right]}\underset{(dxQ)}{\left[\Delta B_{n}\right]}d\Omega+\int_{\Omega}\underset{(Nxd)}{\left[B^{I}\right]}^{T}\underset{(dxd)}{\left[E\right]}\underset{(dxQ)}{\left[\Delta B_{n}\right]}d\Omega=$$

$$\left(\underset{(NxN)}{\left[T_{e}^{m}\right]}^{T}-\underset{(NxN)}{\left[I\right]}\right)\underbrace{\int_{\Omega}\underset{(Nxd)}{\left[B^{I}\right]}^{T}\underset{(dxd)}{\left[E\right]}\underset{(dxN)}{\left[B^{I}\right]}d\Omega}_{\underset{(NxN)}{\left[K^{I}\right]}}\underset{(NxQ)}{\left[T_{n}^{m}\right]}+\underbrace{\int_{\Omega}\underset{(Nxd)}{\left[B^{I}\right]}^{T}\underset{(dxd)}{\left[E\right]}\underset{(dxN)}{\left[B^{I}\right]}d\Omega}_{\underset{(NxN)}{\left[K^{I}\right]}}\underset{(NxQ)}{\left[T_{n}^{m}\right]}=$$

$$\underset{(NxN)}{\left[T_{e}^{m}\right]}^{T}\underset{(NxN)}{\left[K^{I}\right]}\underset{(NxQ)}{\left[T_{n}^{m}\right]}-\underset{(NxN)}{\left[K^{I}\right]}\underset{(NxQ)}{\left[T_{n}^{m}\right]}+\underset{(NxN)}{\left[K^{I}\right]}\underset{(NxQ)}{\left[T_{n}^{m}\right]}\Rightarrow$$

$$\underset{(NxQ)}{\left[K_{en}^{F}\right]}=\underset{(NxN)}{\left[T_{e}^{m}\right]}^{T}\underset{(NxN)}{\left[K^{I}\right]}\underset{(NxQ)}{\left[T_{n}^{m}\right]}$$

- $\underset{(QxN)}{\left[K_{ne}^{F}\right]}$

$$\underset{(QxN)}{\left[K_{ne}^{F}\right]}=$$

$$\int_{\Omega}\underset{(Qxd)}{\left[\Delta B_{n}\right]}^{T}\underset{(dxd)}{\left[E\right]}\left(\underset{(dxN)}{\left[\Delta B_{e}\right]}+\underset{(dxN)}{\left[B^{I}\right]}\right)d\Omega=$$

$$\int_{\Omega}\underset{(Qxd)}{\left[\Delta B_{n}\right]}^{T}\underset{(dxd)}{\left[E\right]}\underset{(dxN)}{\left[B^{I}\right]}d\Omega+\int_{\Omega}\underset{(Qxd)}{\left[\Delta B_{n}\right]}^{T}\underset{(dxd)}{\left[E\right]}\underset{(dxN)}{\left[\Delta B_{e}\right]}d\Omega=$$

$$\underset{(QxN)}{\left[T_{n}^{m}\right]}^{T}\underbrace{\int_{\Omega}\underset{(Nxd)}{\left[B^{I}\right]}^{T}\underset{(dxd)}{\left[E\right]}\underset{(dxN)}{\left[B^{I}\right]}d\Omega}_{\underset{(NxN)}{\left[K^{I}\right]}}+\underset{(QxN)}{\left[T_{n}^{m}\right]}^{T}\underbrace{\int_{\Omega}\underset{(Nxd)}{\left[B^{I}\right]}^{T}\underset{(dxd)}{\left[E\right]}\underset{(dxN)}{\left[B^{I}\right]}d\Omega}_{\underset{(NxN)}{\left[K^{I}\right]}}\left(\underset{(NxN)}{\left[T_{e}^{m}\right]}-\underset{(NxN)}{\left[I\right]}\right)=$$

$$\underset{(QxN)}{\left[T_{n}^{m}\right]}^{T}\underset{(NxN)}{\left[K^{I}\right]}+\underset{(QxN)}{\left[T_{n}^{m}\right]}^{T}\underset{(NxN)}{\left[K^{I}\right]}\underset{(NxN)}{\left[T_{e}^{m}\right]}-\underset{(QxN)}{\left[T_{n}^{m}\right]}^{T}\underset{(NxN)}{\left[K^{I}\right]}\Rightarrow$$

$$\underset{(QxN)}{\left[K_{ne}^{F}\right]}=\underset{(QxN)}{\left[T_{n}^{m}\right]}^{T}\underset{(NxN)}{\left[K^{I}\right]}\underset{(NxN)}{\left[T_{e}^{m}\right]}$$

- $\underset{(QxQ)}{\left[K^F_{nn}\right]}$

$$\underset{(QxQ)}{\left[K^F_{nn}\right]} =$$

$$\int_{\Omega} \underset{(Qxd)}{\left[\Delta B_n\right]}^T \underset{(dxd)}{\left[E\right]} \underset{(dxQ)}{\left[\Delta B_n\right]} d\Omega =$$

$$\underset{(QxN)}{\left[T^m_n\right]}^T \underbrace{\int_{\Omega} \underset{(Nxd)}{\left[B^I\right]}^T \underset{(dxd)}{\left[E\right]} \underset{(dxN)}{\left[B^I\right]} d\Omega}_{\underset{(NxN)}{\left[K^I\right]}} \underset{(NxQ)}{\left[T^m_n\right]} \Rightarrow$$

$$\underset{(QxQ)}{\left[K^F_{nn}\right]} = \underset{(QxN)}{\left[T^m_n\right]}^T \underset{(NxN)}{\left[K^I\right]} \underset{(NxQ)}{\left[T^m_n\right]}$$

Finally, in any case of linear elasticity, 1D, 2D or 3D, we managed to express the Stiffness Matrix of the Fine mesh as a function of the Stiffness Matrix of the Initial mesh in the following way:

$$\underset{(MxM)}{\left[K^F\right]} = \begin{bmatrix} \underset{(NxN)}{\left[K^I\right]} + \underset{(NxN)}{\left[\delta K\right]} & \underset{(NxQ)}{\left[K^F_{en}\right]} \\ \underset{(QxN)}{\left[K^F_{ne}\right]} & \underset{(QxQ)}{\left[K^F_{nn}\right]} \end{bmatrix}$$

where, as we proved above, the separate matrices are:

$$\underset{(NxN)}{\left[\delta K\right]} = \underset{(NxN)}{\left[T^m_e\right]}^T \underset{(NxN)}{\left[K^I\right]} \underset{(NxN)}{\left[T^m_e\right]} - \underset{(NxN)}{\left[K^I\right]}$$

$$\underset{(NxQ)}{\left[K^F_{en}\right]} = \underset{(NxN)}{\left[T^m_e\right]}^T \underset{(NxN)}{\left[K^I\right]} \underset{(NxQ)}{\left[T^m_n\right]}$$

$$\underset{(QxN)}{\left[K^F_{ne}\right]} = \underset{(QxN)}{\left[T^m_n\right]}^T \underset{(NxN)}{\left[K^I\right]} \underset{(NxN)}{\left[T^m_e\right]}$$

$$\underset{(QxQ)}{\left[K^F_{nn}\right]} = \underset{(QxN)}{\left[T^m_n\right]}^T \underset{(NxN)}{\left[K^I\right]} \underset{(NxQ)}{\left[T^m_n\right]}$$

With the previous form of the Stiffness Matrix of the refined mesh,

$$\underset{(MxM)}{\left[K^F\right]}=\begin{bmatrix}\underset{(NxN)}{\left[K^I\right]}+\underset{(NxN)}{\left[\delta K\right]} & \underset{(NxQ)}{\left[K^F_{en}\right]}\\ \underset{(QxN)}{\left[K^F_{ne}\right]} & \underset{(QxQ)}{\left[K^F_{nn}\right]}\end{bmatrix}$$

we have the opportunity to tackle large scale problems with a great number of degrees of freedom. In those cases, inversing a Stiffness Matrix $[K^I]+[\delta K]$ can be a critical undertaking in the process of solving the differential equations and requires a great amount of time. For that reason, if the control points we consider "new" are chosen with care and the values of the matrix $\left[\delta K\right]$ are very small in comparison with the respective values of the matrix $[K^I]$ then we can consider the matrix $\left[\delta K\right]$ negligible and the new matrix of the refined mesh:

$$\underset{(MxM)}{\left[K^F\right]}\simeq\begin{bmatrix}\underset{(NxN)}{\left[K^I\right]} & \underset{(NxQ)}{\left[K^F_{en}\right]}\\ \underset{(QxN)}{\left[K^F_{ne}\right]} & \underset{(QxQ)}{\left[K^F_{nn}\right]}\end{bmatrix}$$

That way we can take advantage firstly of the already formulated matrix $\left[K^I\right]$ and secondly of the fact that we have already inversed it. Then with modern algorithms we can directly use the inversed $\left[K^I\right]^{-1}$ and also inverse much smaller matrices instead of the huge $\left[K^F\right]$.

6.4 New Control Points

6.4.1 New Control Point Number

When a new Knot Value is inserted in a Knot Value Vector, either a preexisting Knot Value whose multiplicity is increased or a new Knot, it affects the surrounding NURBS and their corresponding control points. We recall the second B-Spline basis function property, that in any given knot span $\left[\xi_j, \xi_{j+1}\right)$ at most (p+1) of the functions $N_{i,p}$ are nonzero and those candidates are $N_{j-p,p},\ldots,N_{j,p}$. Thus initially, (p+1) basis functions had support over the knot span$\left[\xi_j, \xi_{j+1}\right)$. Now we insert a new knot in the span $\left[\xi_j, \xi_{j+1}\right)$ and it now consists of two separate Knot Spans $\left[\xi_j, \overline{\xi}\right)$ and $\left[\overline{\xi}, \xi_{j+1}\right)$. The basis functions influenced by the knot insertion are the (p+1) initial basis functions who had support over the $\left[\xi_j, \xi_{j+1}\right)$, plus the new basis function starting at the inserted knot value $\overline{\xi}$. That means there are (p+2) different control points from the initial mesh. Obviously, in more parametric directions, with the insertion of a control point on axis ξ, we insert a whole set of control points on the control net.

In the cases of 1D, 2D, 3D elasticity with a (n x m x r) control points on the axes ξ, η, ζ and a knot value insertion $\overline{\xi}$ on axis ξ with degree p:

In 1D,

1 control point new

p+1 other control points influenced

In 2D,

1xm control points new

(p+1) x m control points influenced

In 3D,

1 x m x r control points new

(p+1) x m x r control points influenced

We note that, since we can consider as "new" only one of the (p+1) control points that are influenced by a knot insertion, including the inserted control point, it is seemingly convenient for us if the insertion of multiple control points is in a small area where they interact with each other. That way, there is overlapping between the control points influenced and we can name a larger percentage of the influenced degrees of freedom as "new". In example, if we make a knot insertion for degree 2, then 3 control points are influenced but only one is considered new. If we insert a second knot and thus an extra control point, 3 control points are influenced again. If 1 of those influenced control points is the same in both knot insertions, then instead of having 2*3=6 influenced control points and considering 2 of them "new", we have 2*3-1=5 influenced control points (one is common in both cases) and we consider 2 "new".

But this is only one possible outcome, the presence of many new knot values in the area could affect too much the nearby control points and thus the influenced "enriched" degrees of freedom in the neighborhood may have totally different interaction with each other that the initial stiffness matrix $[K^I]$ does not describe it satisfactorily any more.

6.4.2 Selecting the New Control Points

In the Hierarchical Scheme we stated that we have to choose which of the influenced Control Points we will consider **new** and which **enriched**. For the enriched ones, we make the assumption that their interaction has not changed with the insertion of new knots and the matrix $\left[K_{ee}^{F}\right]=\left[K^{I}\right]+\left[\delta K\right]\simeq\left[K^{I}\right]$ is the same. Alternatively, we can evaluate the deviation of the contribution of the enriched degrees of freedom after the refinement in matrix $\left[\delta K\right]$.

We will on purpose neglect the contribution of $\left[\delta K\right]$ and we have choose the "new" control points with care in order for $\left[\delta K\right]$ to be small.

We recall that $\underset{(NxN)}{\left[\delta K\right]}=\underset{(NxN)}{\left[T_e^m\right]^T}\underset{(NxN)}{\left[K^I\right]}\underset{(NxN)}{\left[T_e^m\right]}-\underset{(NxN)}{\left[K^I\right]}$ and that $\underset{(NxM)}{\left[T^m\right]}=\underset{(NxM)}{\left[\underset{(NxN)}{\left[T_e^m\right]}\quad\underset{(NxQ)}{\left[T_n^m\right]}\right]}$, where [T] is the transformation matrix for the deformation matrix which we name $\left[T^m\right]$ after we move the "new" Q degrees of freedom at its end. We want to name "new" the degrees of freedom, and subsequently the control points, who were influenced most by the knot refinement. That way, $\left[T_e^m\right]$ will have mostly not very influenced degrees of freedom and be close to the identity matrix which makes the error zero, $\underset{(NxN)}{\left[\delta K\right]}=\underset{(NxN)}{\left[K^I\right]}-\underset{(NxN)}{\left[K^I\right]}=\underset{(NxN)}{\left[0\right]}$

In the general case where many knot values are inserted and basis functions supports are influenced by more than one knot insertion, determining which control point is influenced more is not a task we can easily predict in advance. Instead, as we easily know which control points were influenced (but not how much) we can search the control point Transformation matrix $\left[T^{IF}\right]$ on the positions we know that are affected and determine the q control points that were most influenced.

One criterion to choose q new control points, could be to check the columns of $\left[T^{IF}\right]$, as each column corresponds to the final control points, and select as new the first q control points with the lowest maximum number in their column. That way, the control points in the fine mesh which are influenced by more control points in the initial mesh will be considered new and those who are influenced by mostly one control point, in that case there would be a single 1 in the column, will be considered enriched. The modified matrix $\left[T_e^m\right]$ will be close to the identity matrix and there will be a small error $\left[\delta K\right]$

The idea of the hierarchical refinement is new and we have not fully implemented it yet. There are many parameters to check, apply it in real world problems to see how it behaves and find a reliable criterion to automatically choose which control points will be considered as new.

7 Applications

In this chapter, we will see three 2D applications of Isogeometric Analysis. A simple cantilever, the Cook's cantilever and a plate with a hole.

7.1 Cantilever 2D

7.1.1 Initial Mesh

7.1.1.1 Geometry and Loading

The standard cantilever 2D Beam presented here is subjected to plane stress. The third dimension (thickness) is much smaller than the other two and all loads are placed in the XY midplane of the beam. The degrees of freedom on the left side are fixed and the load is applied on the right side of the cantilever. For the first representation we use a 3x11 control point mesh.

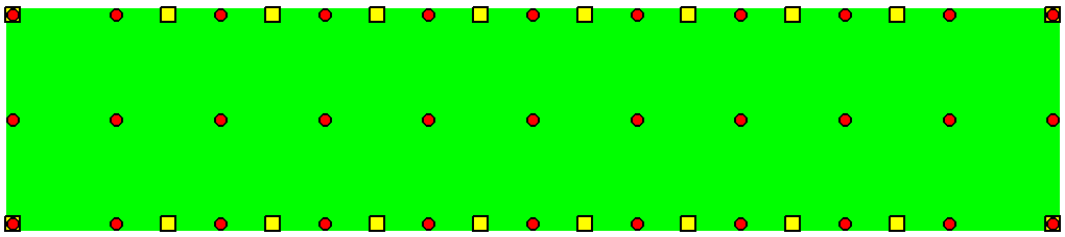
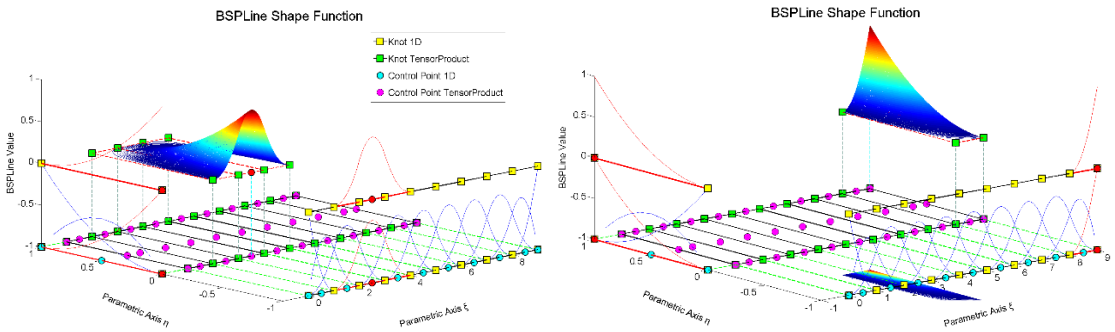


Figure 7.1. Cantilever 2D in Physical Space.
C^1 Continuity across the Knots.

Knots are displayed as yellow rectangles and control points as red circles.



(a) (b)

Figure 7.2. Shape functions of Cantilever 2D in Parameter Space.
We have C^1 Continuity across the knots.
(a) Shape function corresponding to (3,1) Control Point on the ξ, η grid.
(b) Shape function corresponding to (11,3) Control Point on the ξ, η grid.

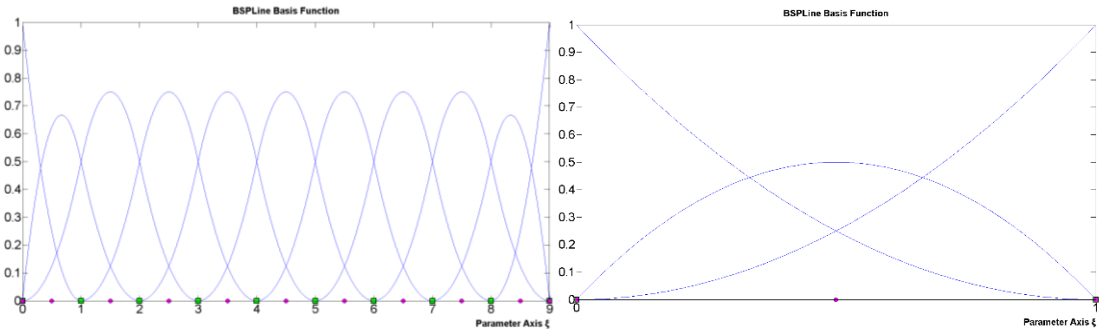


Figure 7.3. B-Spline Basis Function in Parameter Space. Continuity C^1
(a). B-Spline Basis Functions on axis ξ
(b). B-Spline Basis Functions on axis η

To avoid confusion over the loads distribution, we apply it directly on the far right edge of the beam, on the top and bottom corners. The control points there are interpolatory to the geometry and the shape functions correspond to the end of the knot vectors. That means in each corner only one shape function is 1 and all others zero. Subsequently the concentrated load applied on the corner material point is directly transferred to the control point. The shape function of the right up control point of the cantilever can be observed in Figure 7.2 (b).

7.1.1.2 Stiffness Matrix assembly

There are 3x11=33 control points and each of them has 2 degrees of freedom, summing up to a total of 66 degrees of freedom. In each parametric direction we have polynomial degree 2 and since it is a 2D problem, we need p+1=3 gauss points in each parametric direction in a knot span. That means for an element we will need 3x3=9 Gauss Points. We have 9 elements summing up a total of 9x9=81 Gauss Points.

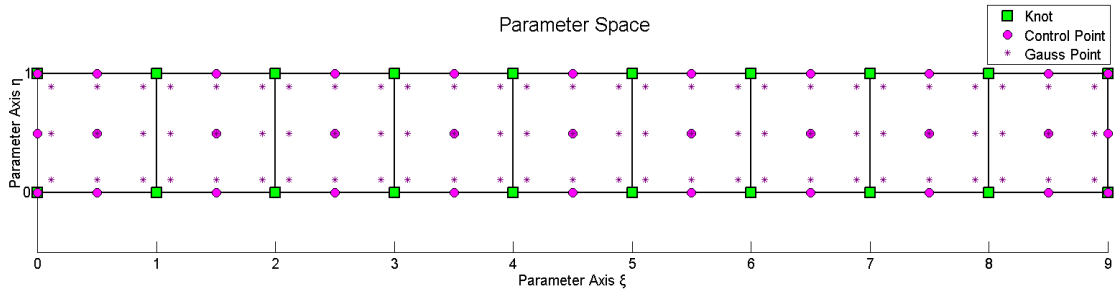


Figure 7.4. Gauss Points in Parameter Space.
Knot Value Vector Ξ = { 0 0 0 1 2 3 4 5 6 7 8 9 9 9} and H = { 0 0 0 1 1 1}

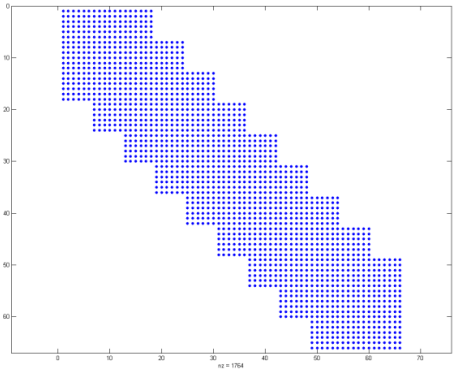


Figure 7.5. Initial Stiffness Matrix. 66 degrees of freedom.
Stiffness Matrix: 4356 cells. 1764 non-zero cells.

We can see in the Stiffness Matrix above, by counting the blue dots, that each degree interacts with maximum 30 others.

Due to its geometry and loading, the beam is subjected to transverse bending. For degree p=2 and Knot multiplicity 1 in all internal Knots, the surface is C^1 continuous across the knot boundaries. Below, we see the stresses σ_{xx}. We notice that the changes are smooth and the stress field is continuous.

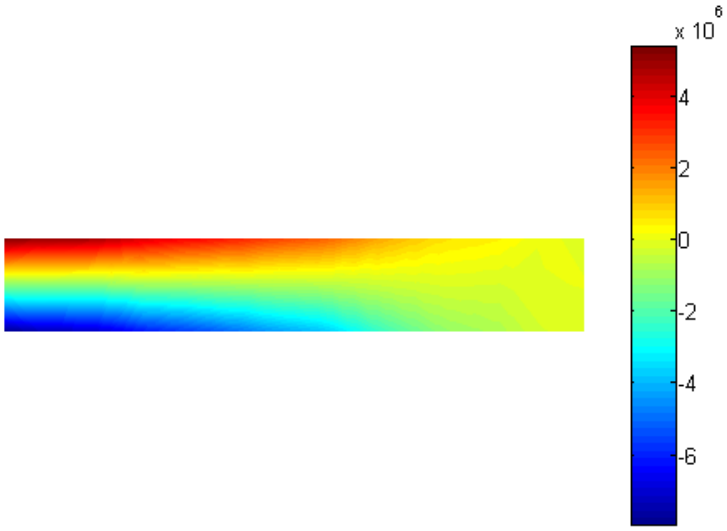


Figure 7.6. Smooth Stresses σ_{xx}.
C^1 continuity across the Knots.

7.1.2 h-Refinement for C^0 continuity

We will analyze the same model with C^0 continuity across its knots. We add no new Knots but instead insert already existing Knot values so that every internal Knot has multiplicity p. This results in continuity C^0 continuity along element boundaries which now become patch boundaries.

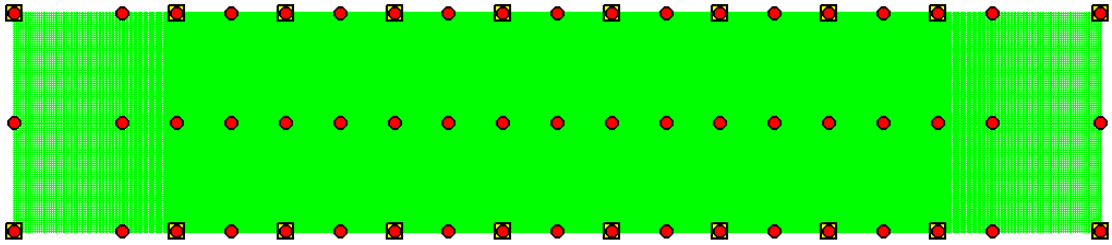


Figure 7.7. Physical Space.
C^0 continuity across the Knots.

The Shape and basis functions in the case of C^0 continuity are designed in the Figures 7.8, 7.9 below.

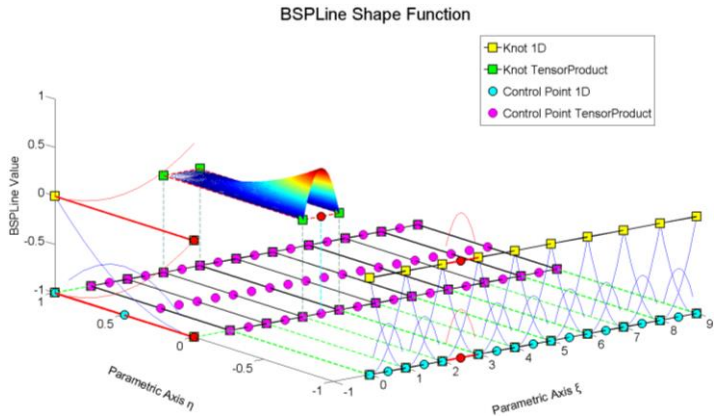


Figure 7.8. Shape function of Cantilever 2D in Parameter Space.
We have C^0 Continuity across the knots.
The shape function corresponds to the (6,1) Control Point on the ξ, η grid.

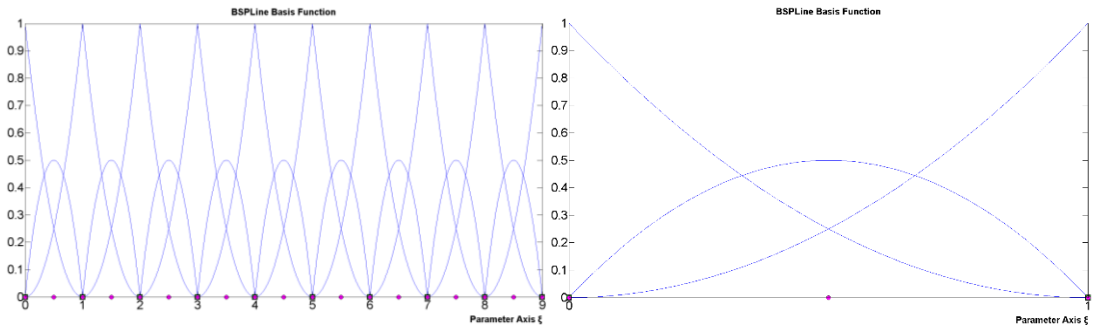
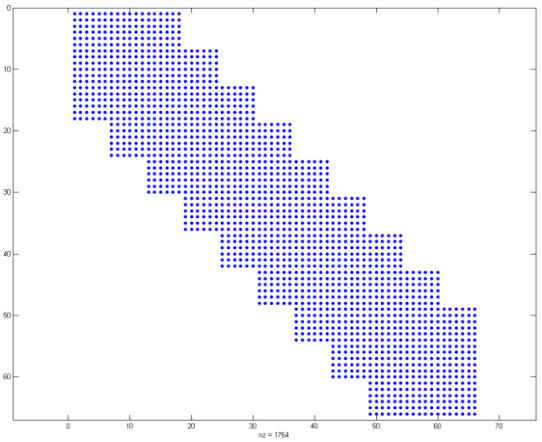
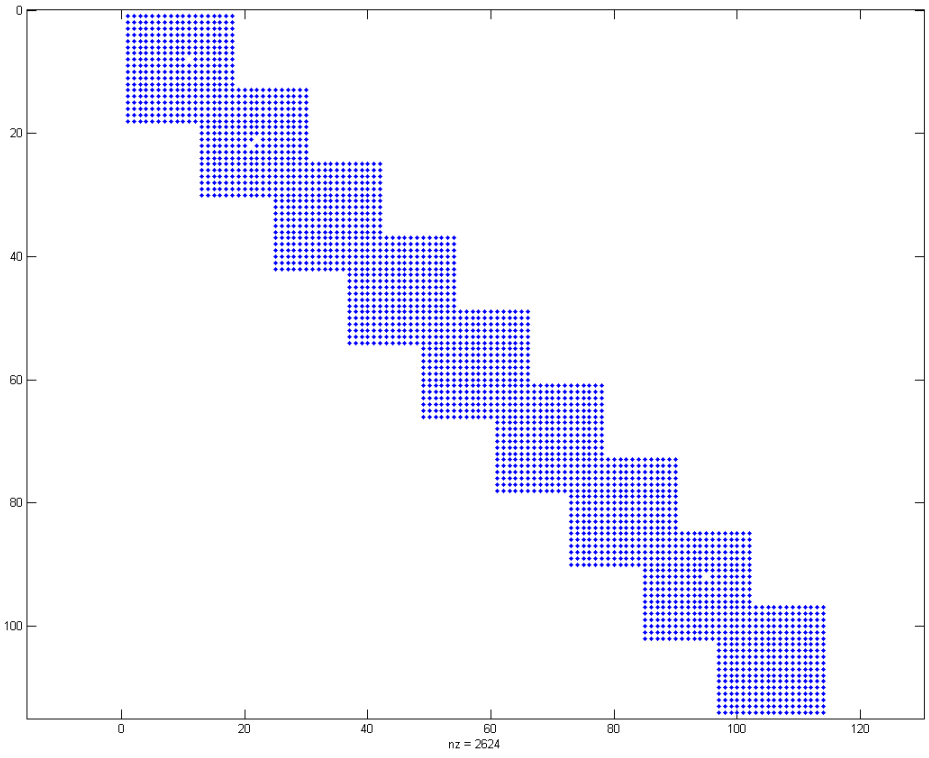


Figure 7.9. B-Spline Basis Function in Parameter Space. Continuity C^0
(a). B-Spline Basis Functions on axis ξ
(b). B-Spline Basis Functions on axis η

With the insertion of Knot Values we have also new Control Points. We now have a model with 19x3=57 Control Points and 57x2=114 degrees of freedom. The stiffness matrix, as we have expected, since we retained the same degree, has exactly the same bandwidth in both cases. Each degree of freedom interacts at maximum with 30 others. However, the C^1 continuity has more overlapping over the elements and thus the matrix is denser. We drew the two cases separately in Figure 7.10 with the same scale for greater clarity. The C^1 continuity is much smaller (66x66) compared to the matrix of C^0 continuity (114x114).



(a)



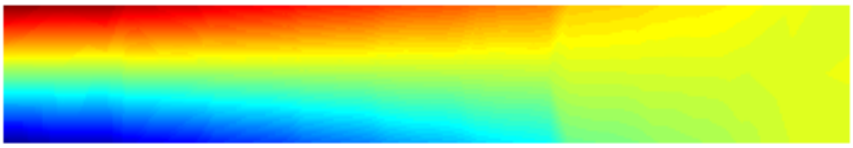
(b)

Figure 7.10. Stiffness Matrices with nonzero elements in blue.
(a) Denser C^1 Continuity Stiffness Matrix
(b) Sparse C^0 Continuity Stiffness Matrix

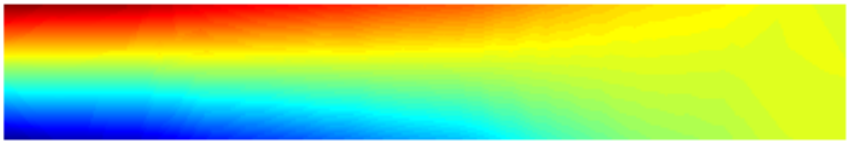
C^1 and C^0 Continuity Stiffness Matrix Comparison		
Continuity	C^1	C^0
Degree	2	2
Control Point	33	57
Degree of Freedom	66	114
Stiffness Matrix elements	4356	12996
Stiffness Matrix nonzero elements	1764	2624
Density= NonzeroElements/Elements	**0,40**	**0,20**

Figure 7.11. Comparison of the Stiffness Matrix properties in C^0 and C^1 continuity.

The stresses are a function of the surface's derivatives, and with continuity C^0 across the knot boundaries we will have discontinuities there in the stress field.



(a)



(b)

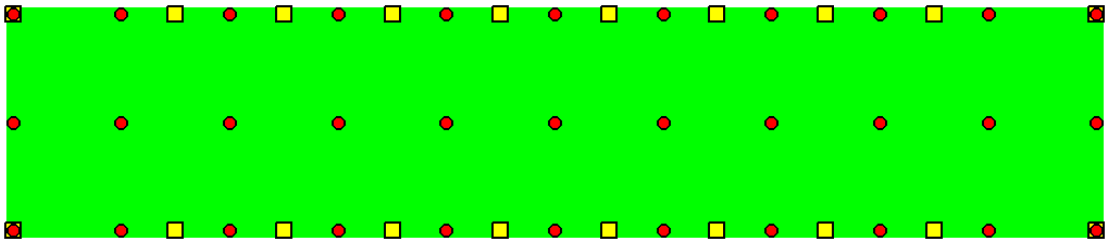
Figure 7.12. Stresses σ_{xx}.
(a). C^0 continuity across the Knots. The stresses are discontinuous.
(b). C^1 continuity across the Knots. The stresses are continuous.

7.1.3 Investigation with refinement

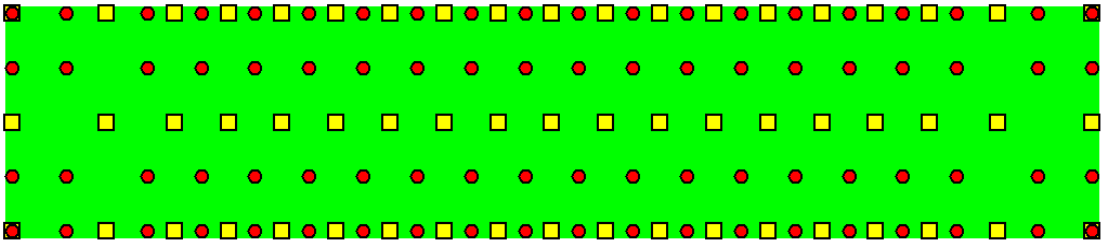
To compare the refinement h, p, k procedures we will compare the displacements on the far right and bottom corner control point. We choose that control point's displacement because it is interpolatory to the beam thus its displacements will always be the the actual displacement of the material point on the right down corner. The shape functions there are interpolatory and the control point's displacement is the actual displacement of the material point there.

Up to now we have performed two solutions with the initial mesh and an extra one with C^0 continuity. Both meshes were very rough and far from the actual solution. For that reason we will not include them in the investigation of the convergence.

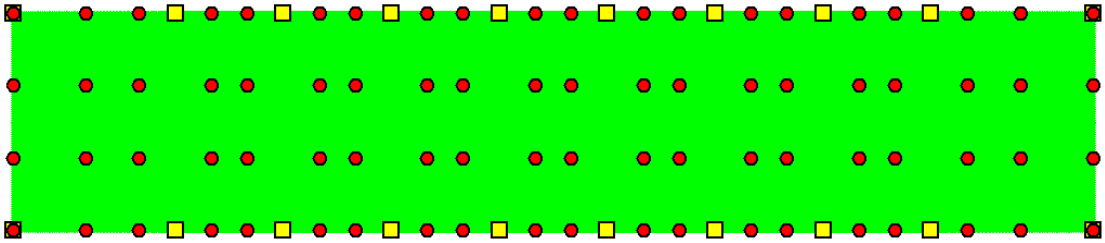
To parametrically study the problem and its convergence, we did several refinements on the initial mesh.



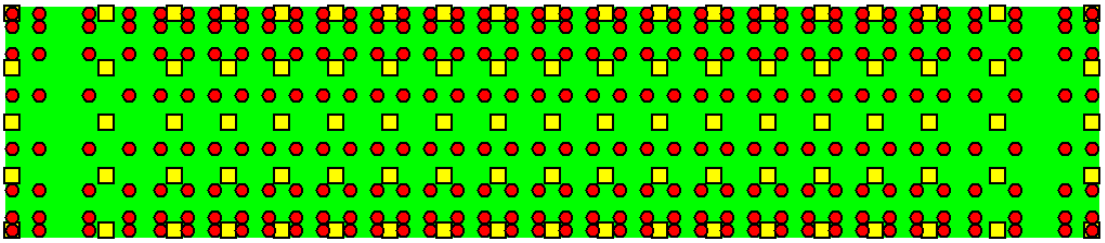
(a) Initial Mesh



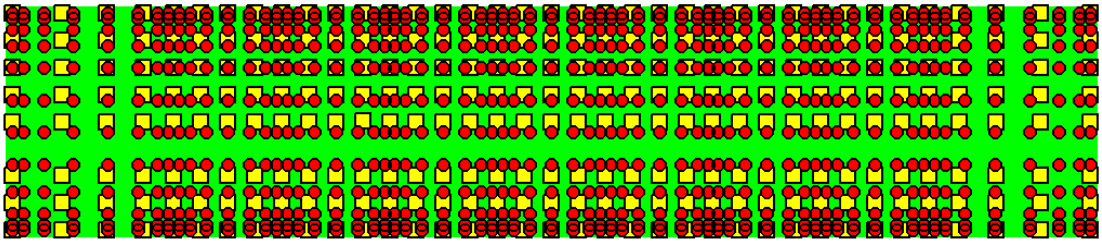
(b) h-refinement



(c) p-refinement, p=3



(d) k-refinement 1, p=4



(e) k-refinement 2, p=5

Figure 7.13. Meshes of Cantilever 2D in Physical Space after consecutive refinements.

In the above Figure the separate meshes are the result of:

(a) An **initial meshing** of degree 2 on each axis. Control grid 11x3. Degrees of freedom 160.

(b) A **uniform h-refinement** on axes ξ and η, by inserting a knot at the middle of each knot span. Control grid 20x4. Degrees of freedom 160.

(c) A **p-refinement** on axes ξ and η, increasing each degree from 2 to **3**. Control grid 20x4. Degrees of freedom 160.

(d) A **k-refinement** on axis ξ and η, by increasing each degree form 2 to **4** and afterwards subdividing the knot spans on ξ by 2 and on η by 4. Control grid 38x8. Degrees of freedom 608.

(e) A **k-refinement** on axes ξ and η, by increasing each degree from 2 to **5** and afterwards subdividing the knot spans on ξ and η uniformly by 2 and by 4 respectively. Control grid 65x12. Degrees of freedom 1560.

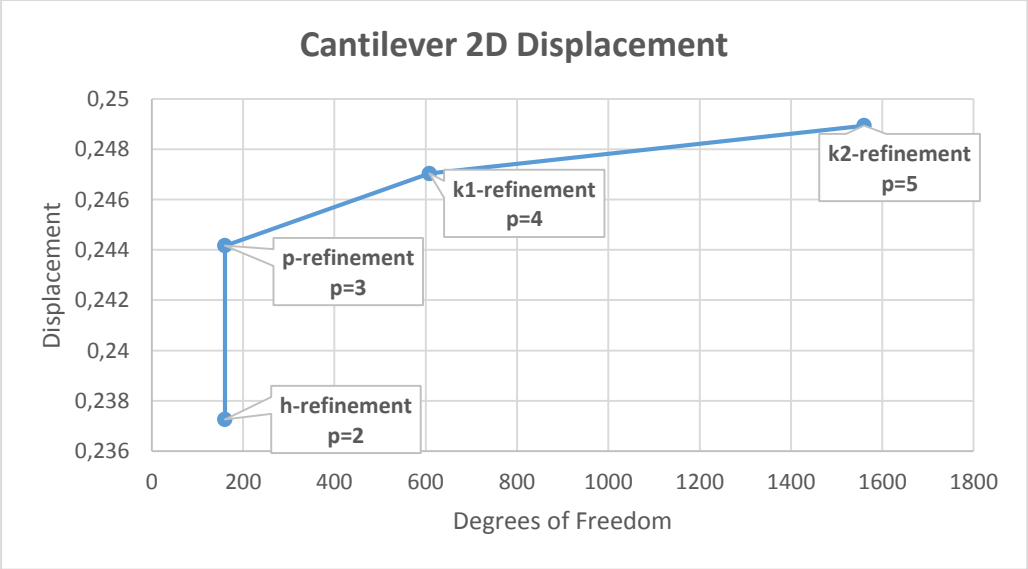


Figure 7.14. Investigation of the Convergence for the Cantilever 2D Beam, in respect with the degrees of freedom.

The initial solution has been omitted as it was very far (around 3) from the final solution and would require a scale that would not let us compare the other refinement cases. Also, the load value is not important as it is linear to the displacement, what is however important is the relative distance of the displacement values we obtained.

The initial solution was around 0.50, very far from the convergence value. The method instantly gives better results with an h-refinement which produces denser but more importantly roughly even elements, close to squares. With the same degrees of freedom we have better results with a p-refinement which enriches the basis in a way that better represents the interaction between the elements. Nonetheless the best results are taken from k-refinement. The solution converges rapidly with k refinement. With k-refinement, we firstly enrich the polynomial basis with p-refinement and then with an h-refinement we make the mesh denser. The final solution is 0.29.

7.2 Cook's Cantilever

7.2.1 Initial Mesh

Contrary to the simple Cantilever we saw in the previous paragraph, the Cook's Cantilever has not the same cross section across its length. Cook's Cantilever is generally a widely used application for testing numerical methods. Its analytical solution is also complicated as Euler assumptions are not applicable due to its comparable dimensions of length and height.

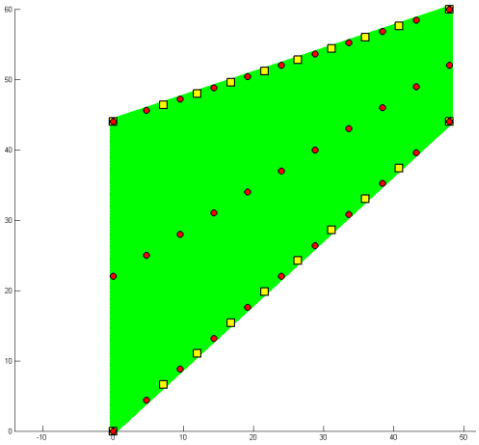


Figure 7.15. Cook's Cantilever 2D in Physical Space.

The geometry of Cook's Cantilever is described by the quadrilateral with vertices (0,0), (0,48), (44,48) and (44,60). The thickness is very small compared to length and height.

The cantilever is again subjected to transverse bending. For the same reasons as in simple cantilever 2D of the previous paragraph, the loads are applied on the right edge, top and bottom control points which coincide with the top and bottom material points. The loads have direction towards the negative Y axis.

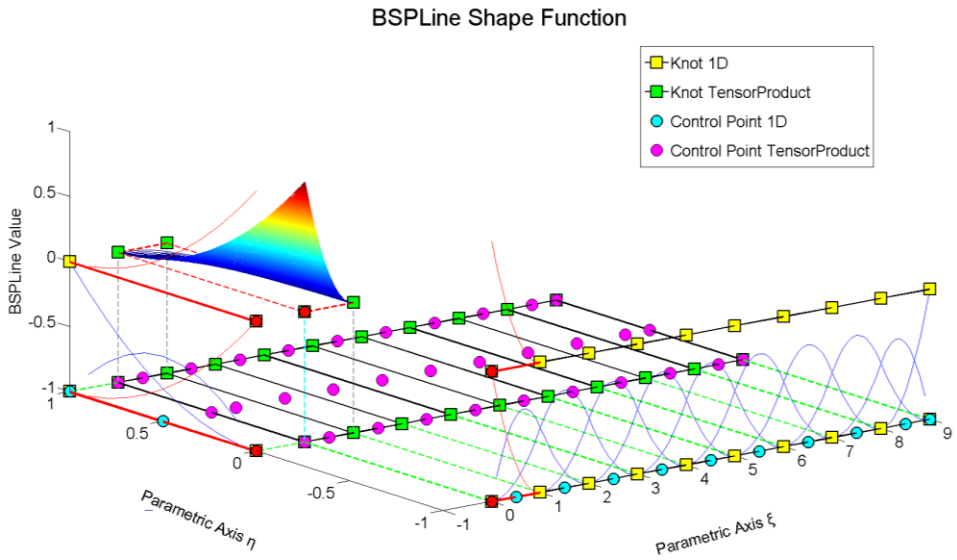


Figure 7.16. Cook's Cantilever Shape function in Parameter Space.

7.2.2 Stiffness Matrix assembly

The assembling of the stiffness matrix is almost the same to the simple cantilever, only the Control Point Cartesian coordinates change and thus we will not expand on that matter. The resulting Stiffness Matrix has the following form with 1764 non zero cells.

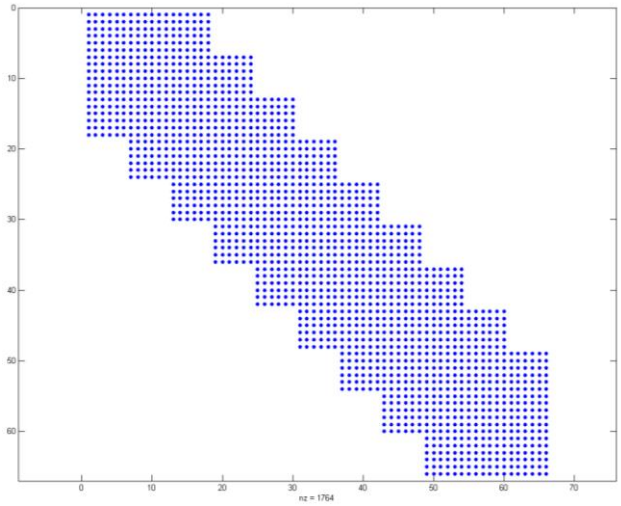
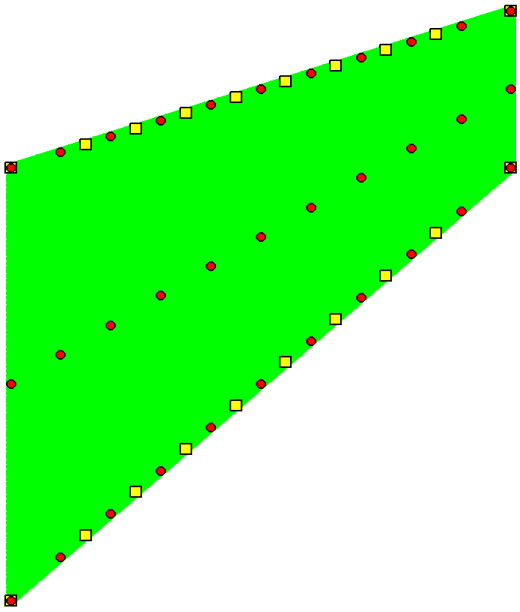


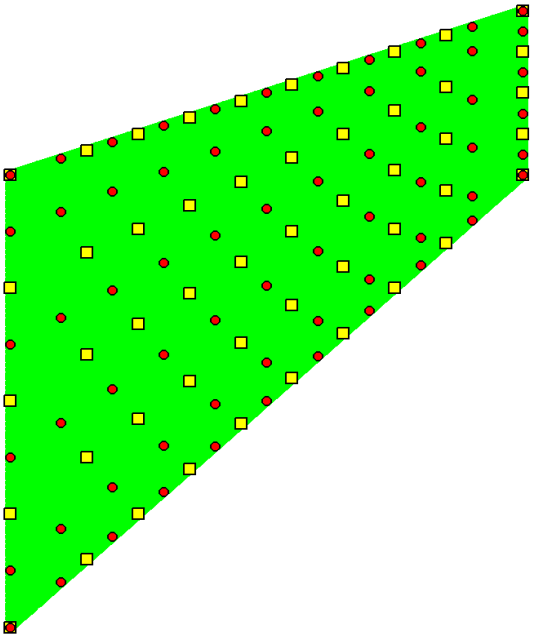
Figure 7.17. Cook's Cantilever Stiffness Matrix with nonzero elements in blue.

7.2.3 Investigation with Refinement

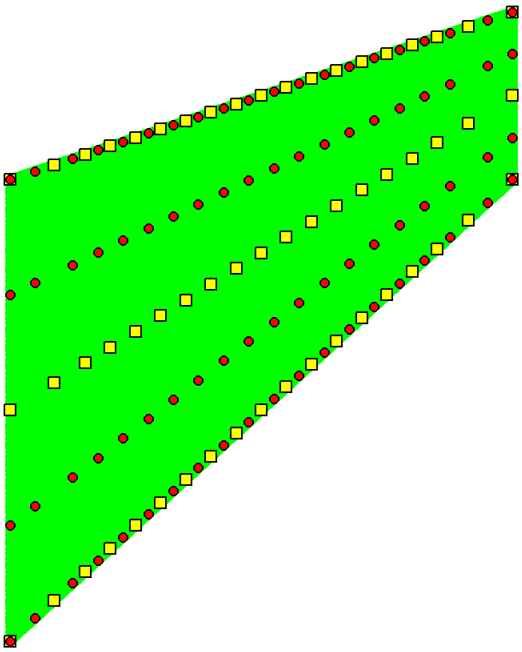
Apart from the Initial Mesh we perform 4 other refinements to converge to the solution. In this case we did several refinements and the mesh after each refinement is shown in the Figure below:



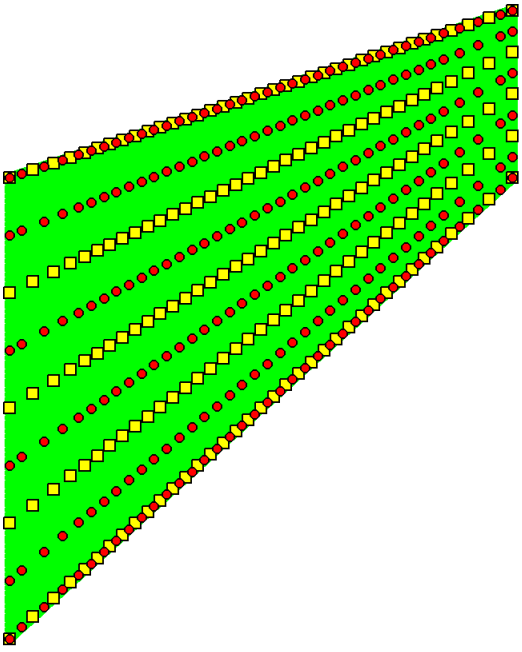
(a) Initial Mesh



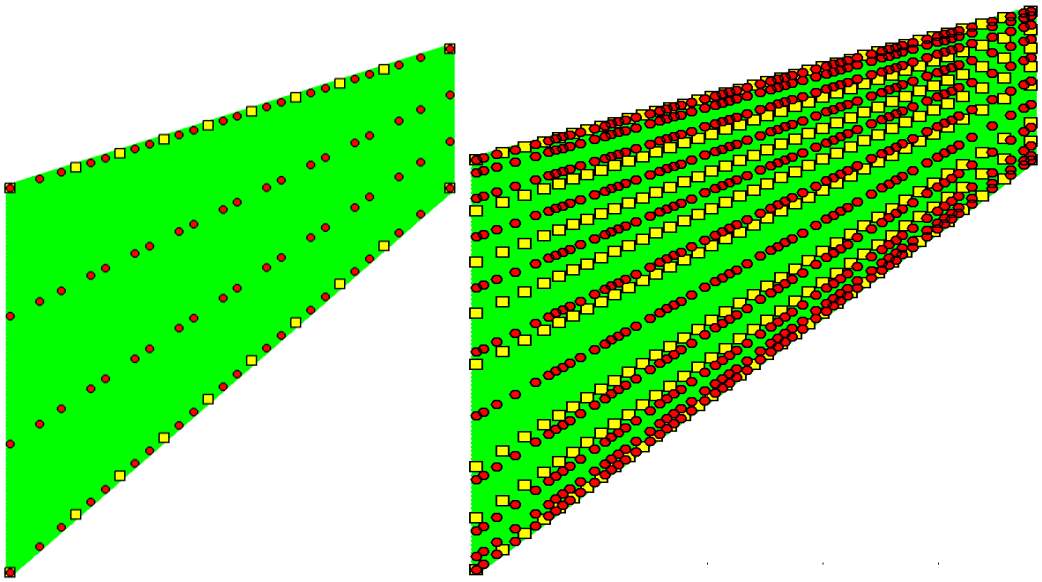
(b) h-refinement 1



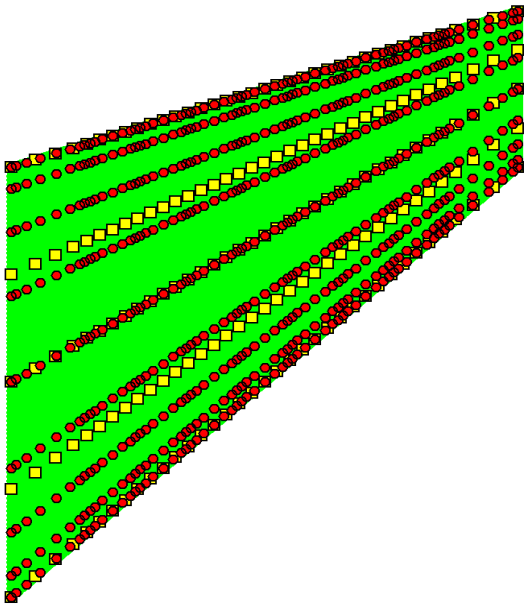
(c) h-refinement 2



(d) h-refinement 3



(e) p-refinement **(f)** k-refinement 1, p=5



(f) k- refinement 2, p=4

Figure 7.18. Meshes of Cook's Cantilever 2D in Physical Space after consecutive refinements.
(a) Initial Mesh, p=2 **(b)** h-refinement 1, p=2
(c) h-refinement 2, p=2 **(d)** h-refinement 3, p=2
(e) p-refinement, p=3 **(f)** k-refinement 1, p=4
(g) k-refinement 1, p=5

In the above Figure the separate meshes are the result of:

(a) An **initial meshing** of degree 2 on each axis. Control grid 11x3. Degrees of freedom 66.

(b) A uniform **h-refinement** on axis η, by inserting a knot at the quarters of each knot span. Control grid 11x6. Degrees of freedom 132.

(c) A uniform **h-refinement** on axes ξ and η, by inserting a knot at the middle of each knot span. Control grid 20x4. Degrees of freedom 160.

(d) A uniform **h-refinement** on axes ξ and η, by inserting two knot at the thirds of each knot span. Control grid 38x6. Degrees of freedom 456.

(e) A **p-refinement** elevating the degree from 2 to 3 in both parametric axes.

(f) A **k-refinement**, raising the degree to 4 and then uniform knot insertion at the quarters of the knot spans. Control grid 56x11. Degrees of freedom 1232.

(g) A **k-refinement** on axes ξ and η, by increasing each degree from 2 to **5** and afterwards subdividing the knot spans on ξ and η uniformly by 2 and by 4 respectively. Control grid 65x9. Degrees of freedom 1170.

We plotted the displacement in respect with the degrees of freedom in each refinement case in the figure below.

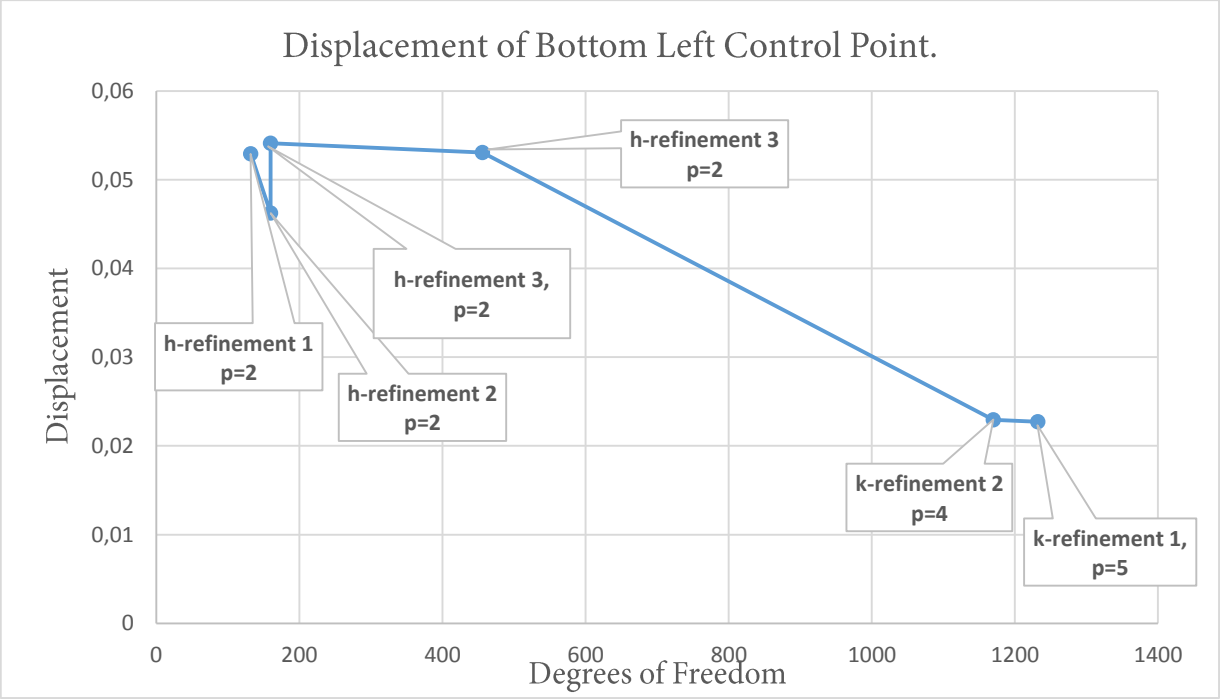


Figure 7.19. Investigation of the Convergence for the Cook Cantilever 2D Beam, in respect with the degrees of freedom.

We notice that although with the first 3 h-refinements and the p-refinement the solution is close to 0.05, with the k-refinement it diverges. The explanation is in the meshing. If we examine the meshes we will see that from the beginning, the elements are distorted, being longer along the η direction. In the 2nd h-refinement that the elements are even more distorted we see the solution diverging. When we make the situation even worse by refining the ξ axis and order elevating (k-refinements 1 and 2)**, even the higher degree cannot reduce the errors created by the extremely distorted elements. The engineer should** always judge which is the best refinement and when.

The stresses after the p-refinement are in the Figure below.

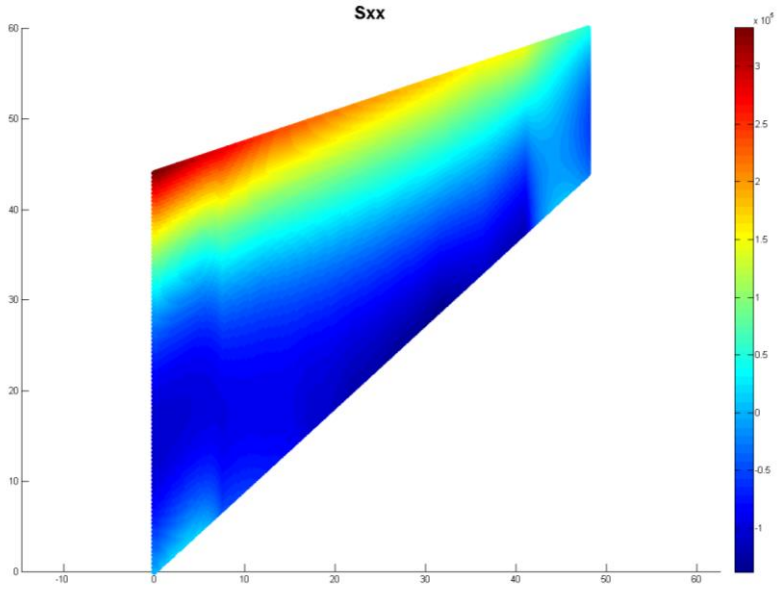


Figure 7.20. Stresses σ_{xx} in the Cook's Cantilever after p-refinement.

7.3 Plate with a circular hole.

An orthogonal plate with a hole in the center is subjected to axial tension in both left and upper side. The problem is symmetric and we can study instead only one quarter of the plate. The thickness is very small to the other dimensions, the forces are applied in the middle plane and subsequently it is a plane stress problem and the geometrical plate behaves as a disc. It is a problem characteristic and commonly studied in Isogeometric analysis.

There are two ways of representing it. The first way is using a double control point at the upper left corner with the cost of creating a singularity there. The second way, which we adopt here, is approximating the left and upper edge with a linear B-Spline. But we need quadratic B-Splines for the simulation of the quarter of a circle on the lower right side and thus we perform p-Refinement in the linear B-Spline to do it. The yellow line connecting the two knots is the patch boundary as the multiplicity of the knots there is p=2 and the continuity $C^{p-m}=C^0$

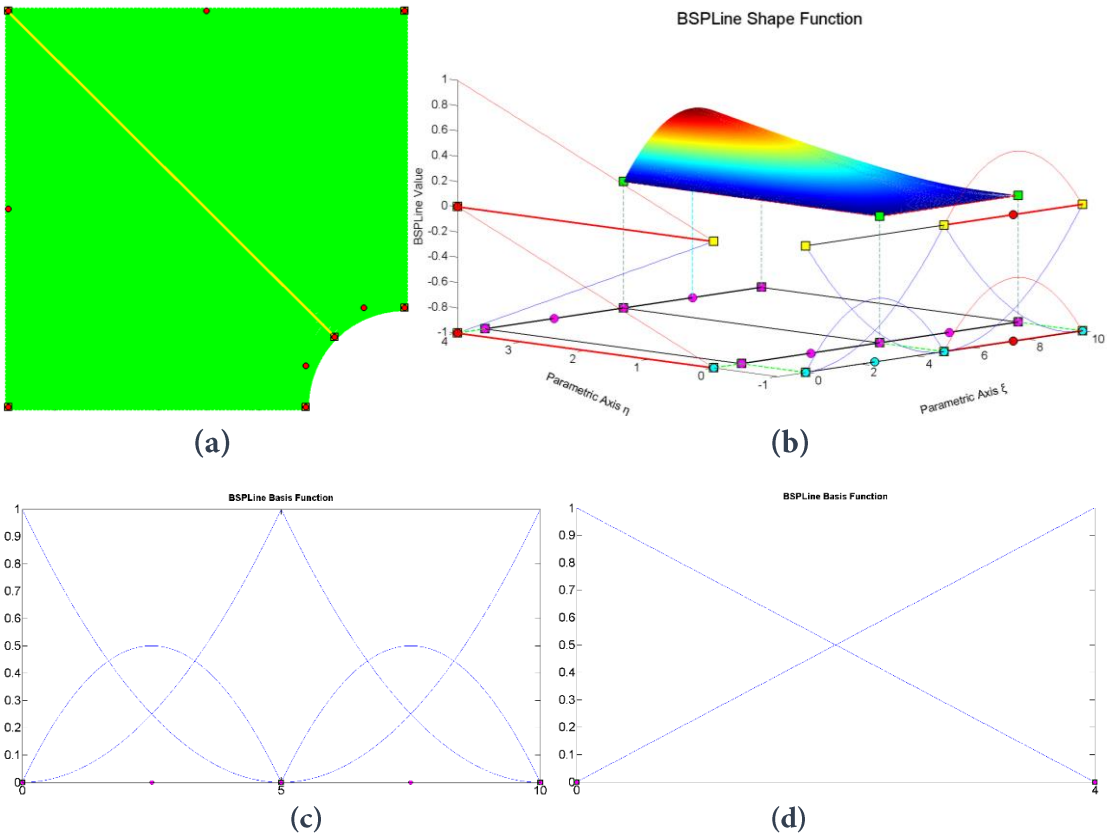


Figure 7.21. Plate with a hole.
(a). Physical Space.
(b). Shape function in Parameter Space.
(c). B-Spline Basis in Parameter Space on ξ.
(d). B-Spline Basis in Parameter Space on η.

A number of refinements was performed to achieve convergence. The meshes are displayed below:

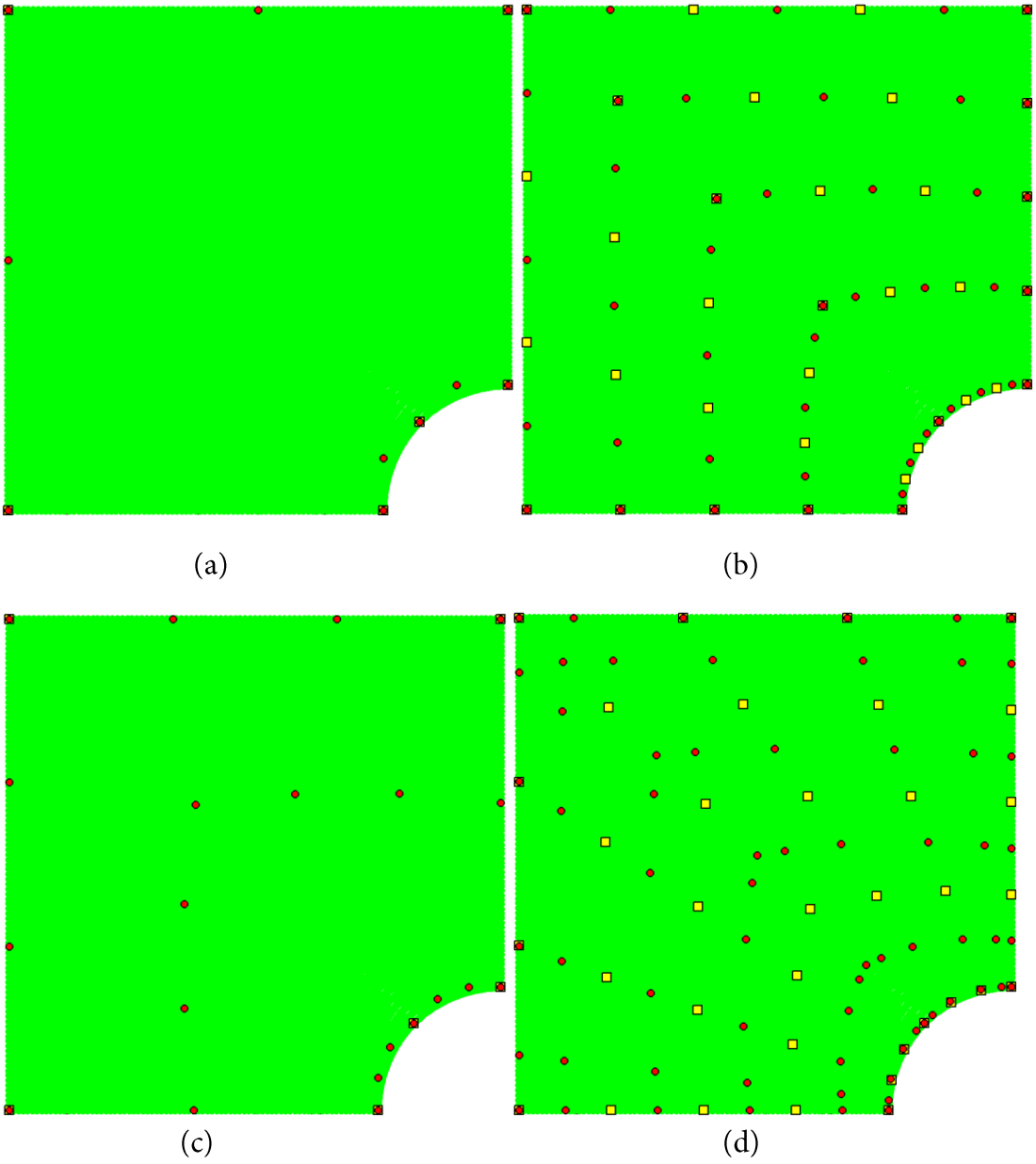


Figure 7.22. Meshes in Physical Space after refinements.
Knots are yellow and Control Points red.
(a). Initial Mesh
(b). h-refinement
(c). p-refinement
(d). k-refinement

In the above Figure the separate meshes are the result of:

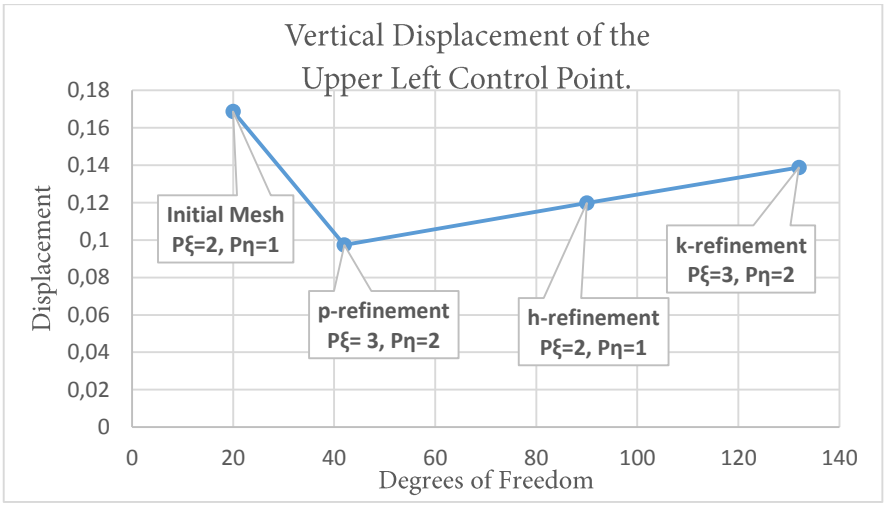
(a) An **initial meshing** of degree 2 on ξ and 1 on η. Control grid 5x2. Degrees of freedom 20.

(b) A uniform **h-refinement** on axes ξ and η, by inserting a knot at the thirds of each knot span. Control grid 9x5. Degrees of freedom 90.

(c) A **p-refinement** on axes ξ and η, raising the degree on ξ from 2 to 3 and in η from 1 to 2. Control grid 7x3. Degrees of freedom 42.

(d) A **k-refinement**, raising the degree on ξ from 2 to 3 and in η from 1 to 2, and then uniform knot insertion at the thirds of the each knot span on both axes. Control grid 11x6. Degrees of freedom 132.

We plotted the displacement in respect with the degrees of freedom in each refinement case in the figure below.



The meshing retains good analogies between the sides of the quadrilaterals. The solution converges fast at 0.14

The stresses after the final k-refinement are shown below in Figure 7.23. We notice the discontinuities in the stress field along the patch line.

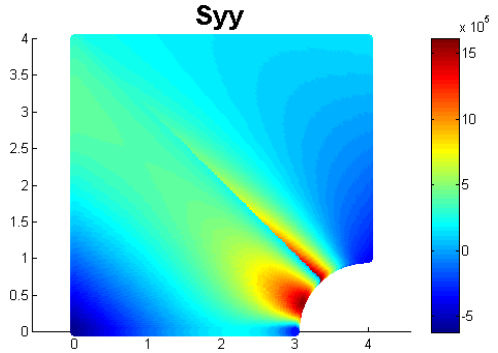


Figure 7.23. Plate with a hole. Stresses S_{yy}.

8 Conclusions

Exact Geometry

Isogeometric Analysis has the great advantage over the ordinary Finite Elements that there is no approximation in its geometry. The shape functions used for the CAD geometry are the same used for analysis and thus there is no reason for approximation. Transition from design to analysis model is geometrical accurate.

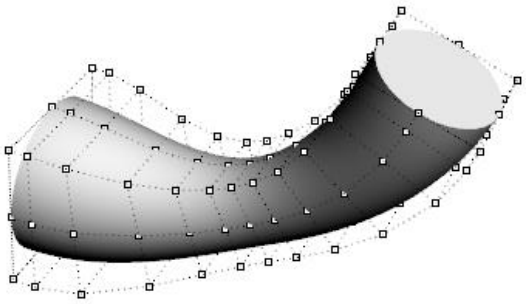


Figure 8.1. NURBS geometry and Control Net.

Improved Refinement

The Isogeometric analysis has the asset of solving from the very first meshing with the accurate CAD geometry. Thus when applying any kind of refinement there is no need to retreat back to the initial CAD geometry and remesh. We refine the mesh in the Parameter Space with h-,p- or k- refinement, and we apply that to the Control Points Coordinates in the Physical Space retaining the same accurate CAD geometry we had from the beginning for our analysis. Furthermore, the k- refinement technique, unique in IGA, seems a very competitive technique compared with h- or p- refinement.

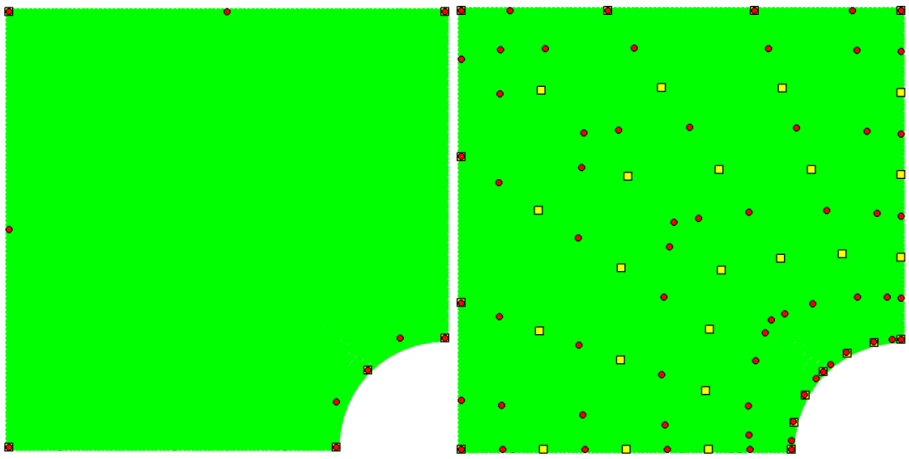
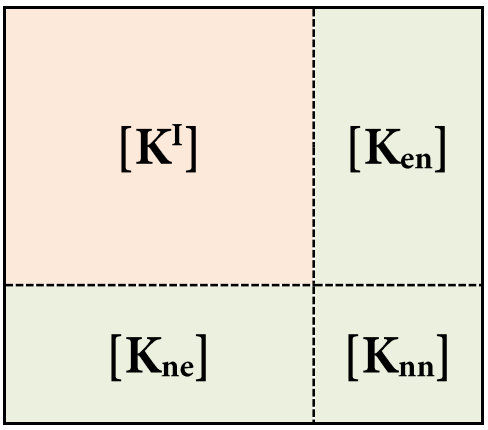


Figure 8.2. Plate with a hole. (a) Coarse mesh and (b) Fine k refined mesh.

Hierarchical Refinement

Hierarchical Refinement is a very hopeful technique for the application of h-refinement on large projects with many degrees of freedom. Especially in NURBS, where local refinement still proves to be a shortcoming. To refine an area we cannot simply insert a control point but have to insert a whole new set of control points due to the Full Tensor Product property. By Hierarchical refinement this process is accelerated a lot, by faster formulating and inversing the Stiffness Matrix.



Element Interconnectivity

The greater overlapping of isogeometric elements in comparison with FEA elements is very useful and leading to a greater interconnectivity. The derivatives are continuous across the boundaries, even with reduced continuity, and thus the stress and strain fields are continuous too. The higher continuity provides IGA with a great tool for fields of interest utilizing higher derivatives for the needed solution. Extrapolation and other methods to approximate stresses in areas they cannot be defined directly through the solution, are only used in special cases of C^0 continuity.

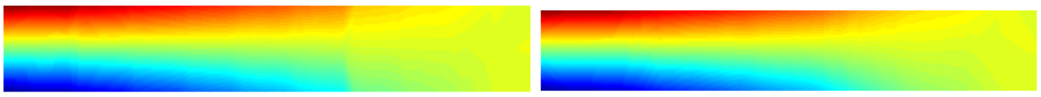


Figure 8.3. Stress σ_{xx} on C^0 and C^1 continuity.

Patches.

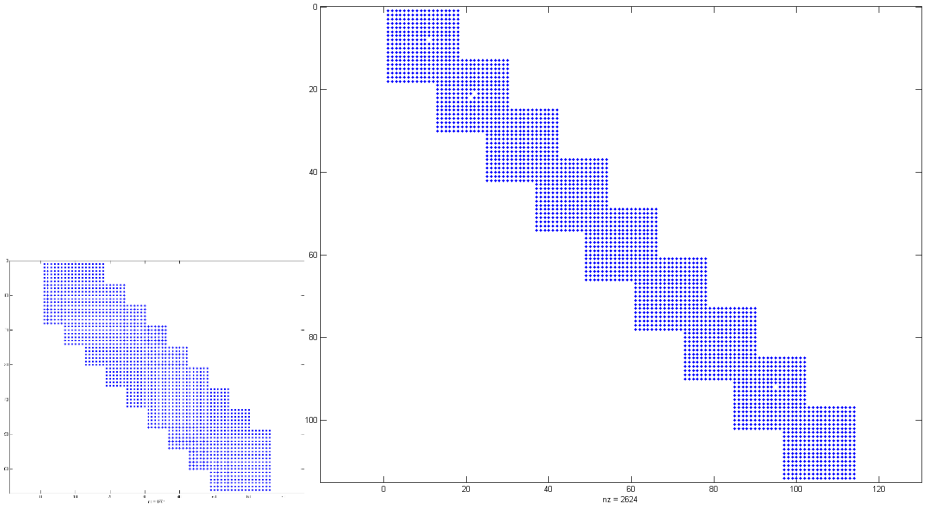
Patches are a very useful tool in IGA. They are used for handling complicated geometries, different materials in the same model or utilizing current methods with parallel computing for solving the equilibrium equation. At patch's edges we have C^{-1} continuity but we can also represent different patches along a single direction with a joined Knot Value Vector of C^0 continuity at the patch merging point.

In either case, C^{-1} and C^0 provide interpolatory control points at the patch edges making it easier to merge it with the rest of the model. There are schemes for patch merging for either overlapping or non overlapping edges with one to one control point consistency or not. However, in the general case the patch connections are problematic and not always able to support water tight connection.

Figure 8.4. Teapot. Failed to water tight connect the two patches. (http://www.siam.org/news/news.php?id=1874)

Stiffness Matrix

The Stiffness matrix formulation process follows the same principles as in FEM and thus existing efficient Finite Element codes can, with some adaptations, be implemented in IGA. The increased overlapping of the elements leads to stronger interconnectivity between them and to a denser Stiffness Matrix, more accurately representing the actual behavior of the model. Fortunately, the increased overlapping does not increase the bandwidth of the Stiffness Matrix in respect with FEM Stiffness Matrices.



(a) **(b)**

Figure 8.5. Stiffness Matrix comparison on the same model with **(a)** C^1 and **(b)** C^0 continuity.

The engineer

Isogeometric analysis proves to be a powerful tool in Computational Mechanics. However, the principles of Finite Elements apply here as well. Distorted elements or wrong choice of axes can lead the method to divergence. The engineer implementing the method should take into account its features and its limitations and thus he has to understand the method and the nature of the various techniques he is using. Only then, will he be a true user of the method, by using it efficiently and, if needed, making adaptations, to the problem's formulation or the method's implementation.

The Isogeometric Analysis Method

Isogeometric Analysis has been already established among the research community, with the number of engineers working on it increasing exponentially as the time passes. Isogeometric analysis has great assets, exact geometry representation, supports higher continuity for the solution than standard Finite Element Analysis and has powerful tools for refinement. In the case of NURBS, it utilizes a well understood and developed CAD technology which means there are plenty of codes and literature available and already present software on the market. The merging of FEA and CAD in the case of NURBS seems to progress smoothly. New emerging types of splines seem promising and are competitive to NURBS analysis by overcoming its drawbacks and provide new paths for further progress on the field. Isogeometric Analysis is a method that is currently being developed, has already gained research momentum and seems to be the natural evolution in the Finite Element Technology.

9 References

Published books and Scientific Papers

Isogeometric Analysis

1. Cottrell J.A., Hughes T.J.R., Bazilevs Y., Isogeometric Analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement, Comput. Methods in Appl. Mech. Eng. 194 (39-41) (2005)
2. Hughes T.J.R., Realli A., Sangalli G., Efficient quadrature for NURBS-based isogeometric analysis, Comput. Methods Appl. Mech. Eng. 199 (5-18), 2010
3. Cottrell J.A., Hughes T.J.R., Bazilevs Y., Isogeometric Analysis: Towards Integration of CAD and FEA, first edition, Wiley, 2009
4. Karatarakis A., Karakitsios P., Papadrakakis M., GPU accelerated computation of the isogeometric analysis stiffness matrix, 2013
5. Vuong A.V., Adaptive Hierarchical Isogeometric Finite Element Methods, 2012
6. Bornemann, Cirak A subdivision based implementation of the hierarchical b-spline finite element method, 2012
7. Kuru G. ,Verhoosel, van der Zee, E.H. van Brummelen, Goal-adaptive Isogeometric Analysis with hierarchical splines , 2013
8. Nguyena, Bordas, Rabczukb , An introduction to Isogeometric Analysis with Matlab implementation, FEM and XFEM formulations, 2012
9. Vuong, Giannelli, Juttler, Simeon, A hierarchical approach to adaptive local refinement in isogeometric analysis , 2011
10. Stamatis A., Isogeometric Linear Static Analysis with NURBS, Diploma Thesis, 2013

Finite Elements

11. Papadrakakis M., Analysis of Structures with the Finite Element Method, second edition, Papasotiriou, 2001
12. Zienkiewicz O.C., Taylor R.L., Zhu J.Z., The Finite Element Method, its Basis and Fundamentals, seventh edition, 2013
13. Metsis P., Lantzounis N., Papadrakakis M., Hierarchical formulation of EFG meshless methods, 2014

Computer Aided Design

14. Piegl L., Tiller W., The NURBS Book , second edition, Springer, 1997
15. de Boor C., A Practical Guide to Splines, first revised edition, 2001
16. Pan, Weng, Recursive representation and application of transformation matrices of B-Splines, 2006
17. Boehm, W., Inserting new knots into B-spline curves, CAD, Vol12, No.4, pp.199-201, 1980
18. Lee, E.T.Y., B-spline Primer, Boeing Document, 1983

Websites.

19. http://en.wikipedia.org/
20. http://tf3dm.com/
21. http://mathworld.wolfram.com/
22. http://docs.mcneel.com/rhino/5/help/en-us/index.htm
23. http://www.scopus.com/
24. http://www.mathworks.com/matlabcentral/fileexchange/26390-nurbs-toolbox-by-d-m-spink