Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Εφαρμοσμενων Μαθηματικων και Φυσικων Επιστημων
Τομεας Μαθηματικων
Δ.Π.Μ.Σ. Εφαρμοσμενες Μαθηματικες Επιστημες

Convergence of Adaptive Finite Elements
Σύγκλιση Προσαρμοστικών Πεπερασμένων Στοιχείων

Μεταπτυχιακή Διπλωματική Εργασία

του

Δημήτρη Κάτσικα

Επιβλέπων:

Εμμανουήλ Γεωργούλης
Καθηγητής Ε.Μ.Π.

Αθήνα, Φεβρουάριος 2022

Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Εφαρμοσμενων Μαθηματικων και Φυσικων Επιστημων
Τομεας Μαθηματικων
Δ.Π.Μ.Σ. Εφαρμοσμενες Μαθηματικες Επιστημες

Convergence of Adaptive Finite Elements
Σύγκλιση Προσαρμοστικών Πεπερασμένων Στοιχείων

Μεταπτυχιακή Διπλωματική Εργασία

του

Δημήτρη Κάτσικα

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή στις __/__/2022.

Εμμανουήλ Γεωργούλης	Κωνσταντίνος Χρυσαφίνος	Βασίλειος Κοκκίνης
Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Αναπληρωτής Καθηγητής Ε.Μ.Π.
.....	.....	.....
(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*

Ευχαριστίες

Πρώτα από όλα, ευχαριστώ τον επιβλέποντα καθηγητή της παρούσας διπλωματικής εργασίας, Εμμανουήλ Γεωργούλη για την στήριξη και την καθοδήγησή του καθ' όλη την διάρκεια εκπόνησής της.

Θα ήθελα να ευχαριστήσω επίσης τους καθηγητές Κωνσταντίνο Χρυσαφίνο και Βασίλειο Κοκκίνη για την συμμετοχή τους στην τριμελή επιτροπή.

Τέλος, ευχαριστώ την οικογένειά μου για την αμέριστη υποστήριξη που μου δίνει όλα αυτά τα χρόνια.

Contents

Περίληψη

Σε αυτήν την εργασία θα ασχοληθούμε με την επίλυση του ελλειπτικού προβλήματος της μερικής διαφορικής εξίσωσης Poisson, με μικτές συνοριακές συνθήκες (Dirichlet και Neumann), μέσω της μεθόδου πεπερασμένων στοιχείων.

Θα αναπτύξουμε δύο είδη μεθόδων πεπερασμένων στοιχείων, την κανονική μέθοδο (Finite Element Method (FEM)) και την προσαρμοστική μέθοδο (Adaptive Finite Element Method (AFEM)).

Η κανονική FEM είναι η κλασική μέθοδος πεπερασμένων στοιχείων, όπου λειτουργεί με συνεχείς τριγωνοποιήσεις ολόκληρου του πεδίου ορισμού και υπολογισμό της προσεγγιστικής λύσης και του σφάλματος σε κάθε τριγωνοποίηση, με σκοπό να μικρύνει το σφάλμα.

Η Adaptive FEM είναι ένα είδος μεθόδου πεπερασμένων στοιχείων, με το χαρακτηριστικό ότι σε κάθε τριγωνοποίηση του πεδίου ορισμού της επιλέγει ποια τρίγωνα θα τριγωνοποιήσει, με κριτήριο ποια θα έχουν το μεγαλύτερο αντίκτυπο στην μείωση του σφάλματος. Αυτό της επιτρέπει να είναι γρηγορότερη από την κανονική FEM.

0.1 Ορισμός ελλειπτικού προβλήματος, ασθενείς μορφές του και ύπαρξη ασθενούς λύσης

Στο κεφάλαιο 1 αναλύουμε το ελλειπτικό πρόβλημα της μερικής διαφορικής εξίσωσης **Poisson**, με μικτές συνοριακές συνθήκες (Dirichlet και Neumann):

$$-\Delta u = f \ \sigma\tau o\nu \ \Omega \subseteq R^n \ \mu\varepsilon \ u : \Omega \mapsto R, \ f : \Omega \mapsto R \tag{1}$$

$$u = g_D \ \sigma\tau o\nu \ \Gamma_D \ \mu\varepsilon \ g_D : \Gamma_D \subseteq R^n \mapsto R \tag{2}$$
(Dirichlet συνοριακή συνθήκη στον Γ_D)

$$\nabla u \cdot \vec{n} = g_N \ \sigma\tau o\nu \ \Gamma_N \ \mu\varepsilon \ g_N : \Gamma_N \subseteq R^n \mapsto R \tag{3}$$
(Neumann συνοριακή συνθήκη στον Γ_N)

με $\partial\Omega = \Gamma_D \cup \Gamma_N$, $\Gamma_D \cap \Gamma_N = \emptyset$ όπου Γ_D κλειστό σύνολο και Γ_N ανοιχτό.

Λόγω της πιθανότητας γωνιακών ιδιομορφιών στο παραπάνω πρόβλημα, θα αναζητήσουμε μία ασθενή λύση. Έτσι, με την αποδυνάμωση των απαιτήσεων διαφοροποίησης στο u, το πρόβλημα

γίνεται, βρες $u \in H^1_{g,D}(\Omega)$ τέτοιο ώστε

$$\int_\Omega \nabla u \cdot \nabla v dx = \int_\Omega f v dx + \int_{\Gamma_N} g_N v dS \ \ \forall v \in H^1_{0,D}(\Omega), \tag{4}$$

όπου $H^1_{0,D}(\Omega) = \{u \in H^1(\Omega) : u = 0 \text{ στον } \Gamma_D\}$ και $H^1_{g,D}(\Omega) = \{u \in H^1(\Omega) : u = g_D \text{ στον } \Gamma_D\}$. Ονομάζουμε αυτή την μορφή, ασθενή μορφή του προβλήματος.

Τώρα θα δείξουμε μια ισοδύναμη ασθενή μορφή του (4), υιοθετώντας τους ακόλουθους συμβολισμούς:

$$a(w,v) = \int_\Omega \nabla w \cdot \nabla v dx \qquad \gamma\iota\alpha\ w, v \in H^1_{0,D}(\Omega), \tag{5}$$

$$l(v) = \int_\Omega f v dx - \int_\Omega \nabla G \cdot \nabla v dx + \int_{\Gamma_N} g_N v dS \qquad \gamma\iota\alpha\ v \in H^1_{0,D}(\Omega), \tag{6}$$

όπου το G επιλέγεται με τις ιδιότητες που περιγράφονται στην παράγραφο 1.1.1. Έτσι το πρόβλημα (4) της αναζήτησης $u \in H^1_{g,D}(\Omega)$, μπορεί να γραφτεί στην ισοδύναμη μορφή της αναζήτησης $w \in H^1_{0,D}(\Omega)$ τέτοιο ώστε

$$a(w,v) = l(v) \ \ \forall v \in H^1_{0,D}(\Omega). \tag{7}$$

Αυτή η μορφή μας βοηθά καλύτερα να αποδείξουμε την ύπαρξη μοναδικής λύσης, χρησιμοποιώντας το θεώρημα Lax - Milgram (Appendix 5.2) και κατά προέκταση την ύπαρξη μοναδικής λύσης στο (4).

0.2 Μέθοδος πεπερασμένων στοιχείων

Στο κεφάλαιο 2 ορίζουμε δύο τρόπους γραμμικής μέθοδου πεπερασμένων στοιχείων για την λύση του προβλήματος που περιγράφεται από τα (1), (2), (3), η μία βασίζεται στο (4) και η άλλη στο (7), αλλά στην πραγματικότητα είναι ισοδύναμες. Πριν δείξουμε τις μεθόδους, ορίζουμε πρώτα τους χώρους και τις ιδιότητές τους που θα χρησιμοποιήσουμε.

Πρώτα ορίζουμε μια τριγωνοποίηση στο πεδίο ορισμού Ω και την καλούμε T, εδώ πρέπει να σημειώσουμε ότι ανάλογα την μορφή του Ω, η διαδικασία τριγωνοποίησης δεν μπορεί πάντα να τον χωρίσει σε ακριβή τρίγωνα. Στον εσωτερικό χώρο του Ω τα τρίγωνα που θα σχηματιστούν θα είναι κανονικά τρίγωνα, αλλά αν ο Ω δεν είναι πολύγωνο τότε θα έχουμε πρόβλημα στα σύνορα, εκεί τα τρίγωνα που θα σχηματιστούν μπορεί να έχουν για πλευρές καμπύλες αντί για γραμμές.

0.2.1 Ορισμός του πεπερασμένου γραμμικού χώρου V_T

Στην συνέχεια ορίζουμε στο T τον πεπερασμένων διαστάσεων γραμμικό χώρο V_T, ο οποίος παίζει σημαντικό ρόλο στην δημιουργία του χώρου των πεπερασμένων στοιχείων για το πρόβλημά μας. $V_T = \{v \in C(\overline{\Omega}) : v_{|\tau} \in P_1 \ \ \forall \tau \in T\}$, όπου P_1 είναι ο γραμμικός χώρος των γραμμικών πολυωνύμων.

Επίσης για κάθε κόμβο x_i στο T ορίζουμε $\phi_i(x) : \Omega \to R$ μια συνεχής και κατά τμήματα γραμμική συνάρτηση, τέτοια ώστε $\phi_i(x_i) = 1$ και $\phi_i(x_j) = 0$ εάν $i \neq j$, άρα είναι σαφές ότι $\phi_i \in V_T$. Ακόμη, το σύνολο $\{\phi_i\}_{i=1}^N$ είναι μια βάση του γραμμικού χώρου V_T, όπου N είναι ο αριθμός των κόμβων του T, συμπεραλαμβανομένων και των συνοριακών κόμβων.

Έτσι το V_T είναι ένας πεπερασμένης διάστασης γραμμικός χώρος, αυτό σημαίνει ότι για κάθε $v \in V_T$ υπάρχει ένα μοναδικό $\overrightarrow{a_v} = (a_{v_1}, a_{v_2}, \cdots, a_{v_N}) \in R^N$ τέτοιο ώστε $v = \sum_{i=1}^{N} a_{v_i}\phi_i(x)$.

0.2.2 Ορισμός του πεπερασμένου γραμμικού χώρου $V_T \cap H^1_{0,D}(\Omega)$

Θέτουμε

$$V_T \cap H^1_{0,D}(\Omega) = \{v \in H^1_{0,D}(\Omega) : v_{|\tau} \in P_1 \ \ \forall \tau \in T \text{ and } v = 0 \text{ on } \Gamma_D\}, \tag{8}$$

την τομή των γραμμικών χώρων V_T και $H^1_{0,D}(\Omega)$, έτσι $V_T \cap H^1_{0,D}(\Omega) \subseteq V_T,\ H^1_{0,D}(\Omega)$ και επειδή έχουμε τομή γραμμικών χώρων, ο $V_T \cap H^1_{0,D}(\Omega)$ είναι και αυτός γραμμικός χώρος.

Επίσης επειδή $V_T \cap H^1_{0,D}(\Omega) \subseteq V_T$ ένας γραμμικός υποχώρος του V_T (ένας περασμένος γραμμικός χώρος), μπορούμε για κάθε $v \in V_T \cap H^1_{0,D}(\Omega)$ να χρησιμοποιήσουμε την βάση του V_T και να το γράψουμε σαν $v = \sum_{i=1}^{N} a_{v_i}\phi_i$, όπου για κάθε i που αντιστοιχεί σε κόμβο στο σύνορο Γ_D έχουμε $a_{v_i} = 0$. Έτσι είναι εύκολο να δούμε ότι ο $V_T \cap H^1_{0,D}(\Omega)$ έχει σαν βάση ένα υποσύνολο της βάσης του V_T, δηλαδή $\{\phi_{i_k}\}_{k=1}^{M} \subseteq \{\phi_i\}_{i=1}^{N}$ όπου i_k για $k = 1, \cdots, M \leq N$ αντιστοιχεί σε όλα τα ϕ_{i_k} που δεν αντιστοιχούν στο Γ_D και όπου $N - M$ είναι ο αριθμός των κόμβων που ανήκουν στο σύνορο Γ_D.

0.2.3 Ορισμός του $V_{T,g,D}$

Ορίζουμε ως

$$V_{T,g,D} = \{v \in C(\overline{\Omega}) : v_{|\tau} \in P_1 \ \ \forall \tau \in T \text{ και } v = g'_D \text{ στον } \Gamma_D\} \subseteq V_T, \tag{9}$$

όπου είναι ένα υποσύνολο του V_T και όχι ένας υποχώρος. Επίσης σαν g'_D παίρνουμε μία πιο γραμμική προσέγγιση του g_D, υποθέτοντας ότι $g'_D(x_i) = g_D(x_i)$ για κάθε $i \in L$ (μία αρίθμηση των κόμβων στον Γ_D) με $x_i \in \Gamma_D$, το σημείο ενός κόμβου της τριγωνοποίησης T στον Γ_D.

Ο λόγος που δεν παίρνουμε $v = g_D$ στον Γ_D είναι επειδή η ομαλότητα του $v \in V_T$ στο σύνορο Γ_D μπορεί να είναι διαφορετική, πιο γραμμική ανά τμήματα, από ότι αυτή του g_D στον Γ_D και άρα τότε δεν μπορούν να είναι ίσα. Αυτό μπορεί αν συμβεί γιατί $v_{|\tau}$ ανήκει στο P_1 για κάθε $t \in T$ και v χρειάζεται να διατηρήσει την ιδιότητα αυτή και στο σύνορο Γ_D, κάτι από το οποίο δεν περιορίζεται το g_D και για αυτό καταλήγουμε στο τέλος να παίρνουμε $v = g'_D$ στον Γ_D. Επίσης αυτό σημαίνει ότι $V_{T,g,D}$ μπορεί να μην είναι υποσύνολο του $H^1_{g,D}(\Omega)$.

Τέλος για κάθε $v \in V_{T,g,D}$, επειδή $v \in V_T$ και $v = g'_D$ στον Γ_D, υπάρχει $(v_1, v_2, \cdots, v_N) \in R^N$ τέτοιο ώστε $v = \sum_{i=1}^{N} v_i\phi_i(x)$ και $v_i = g'_D(x_i) = g_D(x_i)$ για κάθε $i \in L$, όπου $x_i \in \Gamma_D$.

0.2.4 Ορισμός Μεθόδου Πεπερασμένων Στοιχείων (FEM)

'Ετσι για το ασθενές πρόβλημα (4) ορίζουμε την πρώτη μορφή της μεθόδου πεπερασμένων στοιχείων (Finite Element Method ή FEM). Αναζητούμε $u \in V_{T,g,D}$ το οποίο ικανοποιεί την εξίσωση

$$\int_\Omega \nabla u \cdot \nabla v dx = \int_\Omega f v dx + \int_{\Gamma_N} g_N v dS \ \ \forall v \in V_T \cap H^1_{0,D}(\Omega). \tag{10}$$

Αυτό είναι ισοδύναμο με το να βρούμε διάνυσμα $(u_1, u_2, \cdots u_N) \in R^N$ που σχηματίζει $u = \sum_{i=1}^{N} u_i\phi_i(x)$, με την παρατήρηση ότι για κάθε $i \in L$ έχουμε ότι $u_i = g_D(x_i)$ με $x_i \in \Gamma_D$, και το οποίο διάνυσμα

ικανοποιεί την εξίσωση

$$\sum_{i=1}^{N} u_i \int_\Omega \nabla\phi_i(x)\cdot\nabla v dx = \int_\Omega f v dx + \int_{\Gamma_N} g_N v dS \;\; \forall v \in V_T \cap H^1_{0,D}(\Omega).$$

Επίσης επειδή $V_T \cap H^1_{0,D}(\Omega)$ είναι ένας γραμμικός χώρος με βάση $\{\phi_{i_k}\}_{k=1}^{M} \subseteq \{\phi_i\}_{i=1}^{N}$, είναι το ίδιο να πάρουμε την παραπάνω εξίσωση μόνο για τα $\phi_{i_k}(x) \in V_T \cap H^1_{0,D}(\Omega)$ για $k = 1, \cdots, M$. Έτσι το πρόβλημα είναι ισοδύναμο με το να βρούμε $(u_1, u_2, \cdots u_N) \in R^N$ και $u_i = g_D(x_i)\ \forall i \in L$ όπου $x_i \in \Gamma_D$, τέτοιο ώστε

$$\sum_{i=1}^{N} u_i \int_\Omega \nabla\phi_i \cdot \nabla\phi_{i_k} dx = \int_\Omega f\phi_{i_k} dx + \int_{\Gamma_N} g_N \phi_{i_k} dS \;\; \forall k = 1, \cdots, M.$$

Για απλοποίηση, αναδιοργανώνουμε την αρίθμηση των κόμβων για να γράψουμε την μέθοδο όπως: βρες $(u_1, u_2, \cdots u_M, u_{M+1}, \cdots u_N) \in R^N$ με $u_i = g_D(x_i)$ για $i = M+1, \cdots N$, τέτοιο ώστε

$$\sum_{i=1}^{N} u_i \int_\Omega \nabla\phi_i \cdot \nabla\phi_j dx = \int_\Omega f\phi_j dx + \int_{\Gamma_N} g_N \phi_j dS \;\; \forall j = 1, \cdots, M. \tag{11}$$

Τώρα θέτοντας $a_{ji} = \int_\Omega \nabla\phi_i \nabla\phi_j dx \in R$ για $j = 1, \cdots, M,\ i = 1, \cdots, N$ μπορούμε να ορίσουμε τον $M \times N$ πίνακα $A = (a_{ji})_{M\times N}$, καθώς επίσης θέτοντας $F_J = \int_\Omega f\phi_j dx + \int_{\Gamma_N} g_N \phi_j dS$ για $j = 1, \cdots, M$ μπορούμε να πάρουμε το διάνυσμα $F = (F_1, F_2, \cdots F_M)^T \in R^M$. Έτσι το πρόβλημα γίνεται βρες $\vec{u} = (u_1, u_2, ..., u_N)^T \in R^N$ με $u_i = g_D(x_i)$ για $i = M+1 \cdots N$ τέτοιο ώστε

$$A\vec{u} = F \text{ με } A \in R^{M\times N} \text{ και } F \in R^M. \tag{12}$$

Μπορούμε ακόμη να απλοποιήσουμε περαιτέρω αυτή την μορφή παίρνοντας το (11) και γράφοντάς το ως

$$\sum_{i=1}^{M} u_i \int_\Omega \nabla\phi_i \cdot \nabla\phi_j dx + \sum_{i=M+1}^{N} u_i \int_\Omega \nabla\phi_i \cdot \nabla\phi_j dx = \int_\Omega f\phi_j dx + \int_{\Gamma_N} g_N \phi_j dS \;\; \forall j = 1, \cdots, M$$

ή

$$\sum_{i=1}^{M} u_i \int_\Omega \nabla\phi_i \cdot \nabla\phi_j dx = \int_\Omega f\phi_j dx + \int_{\Gamma_N} g_N \phi_j dS - \sum_{i=M+1}^{N} g_D(x_i) \int_\Omega \nabla\phi_i \cdot \nabla\phi_j dx \;\; \forall j = 1, \cdots, M.$$

Τέλος παίρνουμε έναν καινούργιο πίνακα $A'_{M\times M} = (a_{ji})_{M\times M} \in R^{M\times M}$, με $a_{ji} = \int_\Omega \nabla\phi_i \cdot \nabla\phi_j dx$ για $j, i = 1, \cdots, M$, μαζί με το διάνυσμα $F' = (F'_1, F'_2, \cdots, F'_M)^T \in R^M$, όπου $F'_j = \int_\Omega f\phi_j dx + \int_{\Gamma_N} g_N \phi_j dS - \sum_{i=M+1}^{N} g_D(x_i) \int_\Omega \nabla\phi_i \cdot \nabla\phi_j dx$, και το πρόβλημα γίνεται:

$$\text{βρες } \vec{u'} = (u_1, u_2, ..., u_M) \in R^M \text{ τέτοιο ώστε } A'\vec{u'} = F'. \tag{13}$$

0.2.5 Ορισμός εναλλακτικής FEM

Τώρα ορίζουμε τη γραμμική μέθοδο πεπερασμένων στοιχείων που βασίζεται στην δεύτερη μορφή του ασθενούς προβλήματος (7), το οποίο είναι να βρούμε $w \in V_T \cap H^1_{0,D}(\Omega)$ που ικανοποιεί

$$\int_\Omega \nabla w \cdot \nabla v dx = \int_\Omega f v dx - \int_\Omega \nabla G \cdot \nabla v dx + \int_{\Gamma_N} g_N v dS \ \ \forall v \in V_T \cap H^1_{0,D}(\Omega). \tag{14}$$

Βασιζόμενοι στο τι έχουμε δείξει για τον γραμμικό χώρο $V_T \cap H^1_{0,D}(\Omega)$, το $w(x)$ μπορεί να γραφτεί σαν $w(x) = \sum_{i=1}^M w_i \phi_i(x)$ με $\vec{w} = (w_1, w_2, \cdots, w_M) \in R^M$. Έτσι το πρόβλημα γίνεται ισοδύναμο με το πρόβλημα, βρες $\vec{w} \in R^M$ τέτοιο ώστε

$$\sum_{i=1}^M w_i \int_\Omega \nabla \phi_i(x) \cdot \nabla v dx = \int_\Omega f v dx - \int_\Omega \nabla G \cdot \nabla v dx + \int_{\Gamma_N} g_N v dS \ \ \forall v \in V_T \cap H^1_{0,D}(\Omega).$$

Επίσης επειδή $V_T \cap H^1_{0,D}(\Omega)$ είναι ένας γραμμικός χώρος με πεπερασμένη βάση $\{\phi_j(x)\}_{j=1,\cdots,M}$, είναι το ίδιο να πάρουμε την παραπάνω εξίσωση μόνο για τα $\phi_i(x) \in V_T \cap H^1_{0,D}(\Omega)$ με $i = 1, \cdots, M$. Έτσι το πρόβλημα γίνεται ισοδύναμο με το να βρούμε $\vec{w} \in R^M$ τέτοιο ώστε

$$\sum_{i=1}^M w_i \int_\Omega \nabla \phi_i \nabla \phi_j dx = \int_\Omega f \phi_j dx - \int_\Omega \nabla G \cdot \nabla \phi_j dx + \int_{\Gamma_N} g_N \phi_j dS \text{ για } j = 1, \cdots, M.$$

Τέλος παίρνουμε τον πίνακα $A = (a_{ij})_{M \times M}$ με $a_{ij} = \int_\Omega \nabla \phi_i \nabla \phi_j dx$, μαζί με το διάνυσμα $F = (F_1, F_2, \cdots, F_M)^T \in R^M$ όπου $F_i = \int_\Omega f \phi_i dx - \int_\Omega \nabla G \cdot \nabla \phi_i dx + \int_{\Gamma_N} g_N \phi_i dS$ για $i = 1, \cdots, M$ και το πρόβλημα γίνεται:

$$\text{βρες } \vec{w} \in R^M \text{ τέτοιο ώστε } A\vec{w} = F, \tag{15}$$

το οποίο είναι σχεδόν το ίδιο με αυτό που βρήκαμε με την πρώτη μορφή της μεθόδου πεπερασμένων στοιχείων (13).

0.2.6 Ισοδυναμία των δύο μορφών FEM

Στο τμήμα 2.2.2 δείχνουμε ότι το (13) και το (15) είναι ισοδύναμες εξισώσεις. Έτσι οι ισοδύναμες μέθοδοι πεπερασμένων στοιχείων (11), (12) (13) γίνονται ισοδύναμες με τα (14), (15).

0.2.7 Ύπαρξη και μοναδικότητα λύσης της FEM

Ο λόγος που δείξαμε την ισοδυναμία μεταξύ των δύο μορφών ασθενούς προβλήματος (4) και (7) καθώς και την ισοδυναμία των αντίστοιχων μεθόδων πεπρασμένων στοιχείων (12) και (14) ή (15), είναι επειδή χρησιμοποιούμε το (12) για τον υπολογισμό της λύση στο πρόγραμμα MATLAB, ενώ χρησιμοποιούμε το (7) και το (14) ή (15) για τις αποδείξεις της θεωρίας μας.

Έτσι τώρα βλέπουμε ότι η μέθοδός μας πεπερασμένων στοιχείων, ανεξαρτήτως της μορφής της, έχει μία μοναδική λύση. Αυτό είναι εύκολο να το δείξουμε γιατί το $V_T \cap H^1_{0,D}(\Omega)$, που ορίζεται στην δεύτερη μορφή πεπερασμένων στοιχείων (14), είναι ένας πεπερασμένης διάστασης γραμμικός υποχώρος του $H^1_{0,D}(\Omega)$, με την ίδια νόρμα που προκύπτει από το εσωτερικό γινόμενο

του $H^1_{0,D}(\Omega)$, αυτό σημαίνει ότι είναι ένας χώρος Hilbert και επειδή το θεώρημα Lax - Milgram Theorem (Appendix 5.2) ισχύει για τον χώρο $H^1_{0,D}(\Omega)$ στο (7), ισχύει επίσης και για τον υποχώρο του. Έτσι υπάρχει μοναδική λύση για την μορφή πεπερασμένων στοιχείων (14) και άρα για κάθε άλλη ισοδύναμη μορφή.

0.3 A-posteriori φράγμα σφάλματος

Στο κεφάλαιο 3 υπολογίζουμε το συνολικό σφάλμα μεταξύ της ασθενούς λύσης και της προσέγγισής της, τη λύση πεπερασμένων στοιχείων. Το κάνουμε αυτό υπολογίζοντας το a-posteriori φράγμα σφάλματος στο συνολικό σφάλμα.

Το ασθενές πρόβλημα ήταν να βρούμε $u \in H^1_{g,D}(\Omega)$ τέτοιο ώστε

$$\int_\Omega \nabla u \cdot \nabla v dx = \int_\Omega f v dx + \int_{\Gamma_N} g_N v dS \;\; \forall v \in H^1_{0,D}(\Omega), \tag{16}$$

ενώ για αυτό η πεπερασμένων στοιχείων προσέγγισή του είναι να βρούμε $u_n \in V_{T_n,g,D}$ που ικανοποιεί

$$\int_\Omega \nabla u_n \cdot \nabla v_n dx = \int_\Omega f v_n dx + \int_{\Gamma_N} g_N v_n dS \;\; \forall v_n \in V_{T_n} \cap H^1_{0,D}(\Omega). \tag{17}$$

Έτσι ας υποθέσουμε ότι έχουμε $u \in H^1_{g,D}(\Omega)$ που ικανοποιεί το (16) και $u_n \in V_{T_n,g,D}$ που ικανοποιεί το (17), τότε αποδεικνύουμε ότι

$$\|u - u_n\|_{H^1_{0,D}(\Omega)} \leq c^{1/2} \left(\sum_{t \in T} e^2_{tr}(u_n, f, t) \right)^{1/2}, \tag{18}$$

όπου $c \geq 0$ και

$$e_{tr}(u_n, f, t) = \left(h_t^2 \|f + \Delta u_n\|^2_{L^2(t)} + \sum_{e \in dt} h_e \|L(u_n, e)\|^2_{L^2(e)} \right)^{1/2}, \tag{19}$$

με $L(u_n, e) = [\nabla u_n]_e \; \forall e \in S(T) \backslash \Gamma_N$ και $L(u_n, e) = (\nabla u_n \cdot \vec{n} - g_N)_{|e} \; \forall e \in \Gamma_N$.

Έτσι στο (18) έχουμε ορίσει το a-posteriori φράγμα σφάλματος του συνολικού σφάλματος. Επίσης καλούμε το $e_{tr}(u_n, f, t)$ ως εκτιμητή σφάλματος για το τρίγωνο t, ή φράγμα σφάλματος ανά τρίγωνο t και έτσι μπορούμε να δούμε ότι το a-posteriori φράγμα σφάλματος είναι ένα άθροισμα που λαμβάνει υπόψην του τα φράγματα σφάλματος ανά τρίγωνο.

0.4 Κανονική και Προσαρμοστική Μέθοδος Πεπερασμένων Στοιχείων

Στο κεφάλαιο 4, συζητάμε δύο μεθόδους λύσης του ασθενούς προβλήματος, την κανονική Μέθοδο Πεπερασμένων Στοιχείων (Finite Element Method (FEM)) και την Προσαρμοστική Μέθοδο Πεπερασμένων Στοιχείων (Adaptive Finite Element Method (AFEM)).

Η κανονική FEM χρησιμοποιείται με σκοπό να βρούμε μια προσεγγιστική λύση $\overline{u}$ στο ασθενές μας πρόβλημα, όσο πιο κοντά θέλουμε στην πραγματική ασθενή λύση u. Για να το κάνουμε αυτό παίρνουμε μία τριγωνοποίηση T_n και υπολογίζουμε σε αυτή την προσεγγιστική λύση u_n και το a-posteriori φράγμα σφάλματος e_n που έχει με την πραγματική λύση u, τότε αν θέλουμε πιο μικρό σφάλμα ξανατριγωνοποιούμε *όλα* τα τρίγωνα του T_n για να δημιουργήσουμε ένα καινούργιο T_{n+1} για το οποίο υπολογίζουμε ξανά το u_{n+1} και το e_{n+1}. Επαναλαμβάνουμε μέχρι να φτάσουμε ένα n με το επιθυμητό φράγμα σφάλματος (e_n) και τότε για αυτό θα έχουμε $\overline{u} = u_n$.

Το πρόβλημα με την κανονική FEM είναι ότι μετά απο κάποιες τριγωνοποιήσεις θα έχουμε ένα μεγάλο αριθμό N τριγώνων και ο αλγόριθμος μπορεί να γίνει πολύ αργός. Το βασικό ζήτημα, ωστόσο, είναι ότι ο ρυθμός σύγκλισης του κανονικού FEM εξαρτάται από την κανονικότητα της ακριβούς λύσης του προβλήματος. Επομένως, παρουσία γωνιακών ιδιομορφιών, η σύγκλιση του κανονικού FEM είναι πιο αργή από τον ρυθμό σύγκλισης για ομαλές ακριβείς λύσεις.

Αυτό το πρόβλημα έρχεται ο AFEM να λύσει, είναι η ίδια λογική με την κανονικη FEM αλλά για να αποφύγουμε αυτή την βραδύτητα, σε κάθε βήμα αντί να τριγωνοποιούμε όλα τα τρίγωνα, θα επιλέξουμε **δυναμικά** ποια από αυτά χρειάζονται τριγωνοποίηση. Το κάνουμε αυτό με την ελπίδα να βρούμε το φράγμα του σφάλματος που ψάχνουμε δημιουργώντας μικρότερο αριθμό τριγώνων από την κανονική FEM.

Τα κριτήρια με τα οποία επιλέγουμε αυτά τα τρίγωνα βασίζονται στον εκτιμητή σφάλματος για κάθε τρίγωνο, τον οποίο ορίσαμε πιο πριν σαν e_{tr}, το φράγμα σφάλματος ανά τρίγωνο t. Αυτό μας λέει πόσο κάθε τρίγωνο συμβάλει στο συνολικό a - posteriori φράγμα σφάλματος. Τα τρίγωνα με το μεγαλύτερο e_{tr} είναι αυτά που χρειαζόμαστε να τριγωνοποιήσουμε. Αυτό θα μας επιτρέψει να αυξήσουμε την πυκνότητα της τριγωνοποίησής μας με σκοπό σταδιακά να μειώσουμε το σφάλμα, ενώ το κάνουμε ανά σημεία και μόνο γύρω από τα τρίγωνα που θα έχουν το μεγαλύτερο αντίκτυπο. Περισσότερα για τα κριτήρια αργότερα.

Στη συνέχεια ρίχνουμε μια πιο προσεκτική ματιά στα σχεδιαγράμματα των αλγορίθμων, κανονικού FEM και AFEM.

0.4.1 Regular FEM Αλγόριθμος

ΑΡΧΙΚΟ ΠΛΕΓΜΑ ΤΡΙΓΩΝΩΝ:
δημιουργία του αρχικού πλέγματος T_1

↓

ΕΠΙΛΥΣΗ ΠΡΟΒΛΗΜΑΤΟΣ:
επίλυση του προβλήματος πεπερασμένων στοιχείων για το πλέγμα (τριγωνοποίηση) T_1

↓

ΕΚΤΙΜΗΣΗ A-POSTERIORI ΦΡΑΓΜΑ ΣΦΑΛΜΑΤΟΣ:
αναθέτουμε στην μεταβλητή ***error*** *την τιμή του a-posteriori φράγμα σφάλματος για το πλέγμα* T_1

↓

WHILE (**error** > **errorBoundary**)

ΤΡΙΓΩΝΟΠΟΙΗΣΗ ΤΟΥ ΠΛΕΓΜΑΤΟΣ:
για ***ΟΛΑ*** *τα τρίγωνα, τριγωνοποιούμε το πλέγμα* T_n *στο πλέγμα* T_{n+1}

↓

ΕΠΙΛΥΣΗ ΠΡΟΒΛΗΜΑΤΟΣ:
επίλυση του προβλήματος πεπερασμένων στοιχείων για το πλέγμα T_{n+1}

↓

ΕΚΤΙΜΗΣΗ A-POSTERIORI ΦΡΑΓΜΑ ΣΦΑΛΜΑΤΟΣ:
αναθέτουμε στην μεταβλητή ***error*** *την τιμή του a-posteriori φράγμα σφάλματος για το πλέγμα* T_{n+1}

END_WHILE

0.4.2 Adaptive FEM Αλγόριθμος

ΑΡΧΙΚΟ ΠΛΕΓΜΑ ΤΡΙΓΩΝΩΝ:
δημιουργία του αρχικού πλέγματος T_1

↓

ΕΠΙΛΥΣΗ ΠΡΟΒΛΗΜΑΤΟΣ:
επίλυση του προβλήματος πεπερασμένων στοιχείων για το πλέγμα (τριγωνοποίηση) T_1

↓

ΕΚΤΙΜΗΣΗ A-POSTERIORI ΦΡΑΓΜΑ ΣΦΑΛΜΑΤΟΣ:
αναθέτουμε στην μεταβλητή ***error*** *την τιμή του a-posteriori φράγμα σφάλματος για το πλέγμα* T_1

↓

WHILE (**error** > **errorBoundary**)

ΜΑΡΚΑΡΙΣΜΑ ΤΡΙΓΩΝΩΝ:
με βάση το CRITERION μαρκάρουμε τα τρίγωνα που χρειάζονται επί μέρους τριγωνοποίηση

↓

ΤΡΙΓΩΝΟΠΟΙΗΣΗ ΤΟΥ ΠΛΕΓΜΑΤΟΣ:
για τα ***ΜΑΡΚΑΡΙΣΜΕΝΑ*** *τρίγωνα, τριγωνοποιούμε το πλέγμα* T_n *στο πλέγμα* T_{n+1}

↓

ΕΠΙΛΥΣΗ ΠΡΟΒΛΗΜΑΤΟΣ:
επίλυση του προβλήματος πεπερασμένων στοιχείων για το πλέγμα (τριγωνοποίηση) T_{n+1}

↓

ΕΚΤΙΜΗΣΗ A-POSTERIORI ΦΡΑΓΜΑ ΣΦΑΛΜΑΤΟΣ:
αναθέτουμε στην μεταβλητή ***error*** *την τιμή του a-posteriori φράγμα σφάλματος για το πλέγμα* T_{n+1}

END_WHILE

0.4.3 Παρατηρήσεις για τους κανονικό FEM και AFEM αλγορίθμους

Βασιζόμενοι στο (18) παίρνουμε σαν παράμετρο ***error*** το a-posteriori φράγμα σφάλματος που ορίζεται ως

$$\boldsymbol{error} = \left(\sum_{t \in T} e_{tr}^2(u_n, f, t) \right)^{1/2}. \tag{20}$$

Τότε και για τους δύο αλγορίθμους κανονικός και Adaptive FEM, τρέχουμε την επανάληψη *WHILE*, μέχρι η προϋπόθεση ***error*** > ***errorBoundary*** να σταματήσει να ισχύει, αυτό συμβαίνει όταν το a-posteriori φράγμα σφάλματος γίνει μικρότερο ή ίσο από το ελάχιστο φράγμα ***errorBoundary***.

Όταν συμβεί αυτό, σύμφωνα με το (18), σημαίνει ότι έχουμε πετύχει το ακόλουθο σφάλμα,

$$\|u - u_n\|_{H^1_{0,D}(\Omega)} \leq c \cdot \boldsymbol{error} \leq c \cdot \boldsymbol{errorBoundary},$$

όπου $c > 0$ είναι μια σταθερά.

Έτσι η παράμετρος ***errorBoundary*** είναι ένα όρισμα εισαγωγής στο πρόγραμμά μας, που μας βοηθάει να καθορίσουμε πόσο κοντά στην πραγματική λύση u θέλουμε να βρούμε την προσεγγιστική u_n.

Όπως δείχνουμε στο κεφάλαιο 4.5, και για τους δύο αλγορίθμους, κανονικό και Adaptive FEM το a-posteriori φράγμα σφάλματος (***error***) συγκλίνει στο μηδέν καθώς το $n \to \infty$, έτσι τελικά θα ικανοποιήσουν την προϋπόθεση εξόδου του ***errorBoundary*** και το πρόγραμμα θα τερματίσει.

0.4.4 Κριτήριο του Adaptive FEM Αλγορίθμου

Όπως είπαμε και πριν το ΚΡΙΤΗΡΙΟ με το οποίο σημειώνουμε ποια τρίγωνα να τριγωνοποιήσουμε περαιτέρω, είναι αυτό που θα δώσει την προσαρμοστικότητα στον αλγόριθμό μας. Στην πραγματικότητα παρουσιάζουμε δύο τρόπους για να το κάνουμε αυτό, με τον δεύτερο να μας δίνει ακόμη μεγαλύτερη ταχύτητα στον υπολογισμό από τον πρώτο:

1st περίπτωση: Στην αρχή το πρόγραμμά μας λαμβάνει μια σταθερή παράμετρο $0 < \textbf{errorPercentage} \leq 1$ ως είσοδο. Σε κάθε επανάληψη, ταξινομεί τους εκτιμητές σφαλμάτων ανά τρίγωνο στο πλέγμα μας, από το μεγαλύτερο στο μικρότερο. Συμβολίζουμε τους εκτιμητές σφαλμάτων ανά τρίγωνο ως e_t για κάθε $t \in T$, όπου t είναι ένα τρίγωνο και T το σύνολο τριγωνισμού (πλέγμα). Τέλος, λαμβάνουμε ως $T_H \subseteq T$, το ελάχιστο σύνολο τριγώνων t με το μεγαλύτερο e_t που ικανοποιεί τα ακόλουθα,

$$\sqrt{\sum_{t \in T_H} e_t^2} \geq errorPercentage \cdot error, \tag{21}$$

το οποίο σημαίνει

$$\sum_{t \in T_H} e_t^2 \geq errorPercentage^2 \cdot error^2. \tag{22}$$

Αυτό το T_H είναι το σύνολο των τριγώνων που σημειώνουμε για τριγωνοποίηση. Ονομάζουμε αυτή τη διαδικασία σήμανσης (μαρκαρίσματος) ως bulk-chasing marking strategy και εισήχθη από τον Dörfler.

2nd περίπτωση: Αυτό είναι παρόμοιο με την πρώτη περίπτωση, αλλά τώρα αντί για το **errorPercentage** να είναι μια σταθερά, είναι μια μεταβλητή που αλλάζει προσαρμοστικά σε κάθε επανάληψη. Επίσης το **errorPercentage** έχει ένα κατώτερο όριο που μπορεί να φτάσει, που ορίζεται ως η σταθερά **errorPercentageBoundary**. Παρακάτω είναι πώς το αποκτούμε σε κάθε επανάληψη:

$$errorPercentage = \frac{(error - errorBoundary)}{error} \tag{23}$$

$$\begin{aligned} &if(errorPercentage < errorPercentageBoundary) \\ &\quad errorPercentage = errorPercentageBoundary \\ &end \end{aligned} \tag{24}$$

Άρα το ΚΡΙΤΗΡΙΟ του Adaptive FEM εξαρτάται από την παράμετρο σήμανσης $errorPercentage$ και είναι είτε σταθερό είτε αλλάζει ανά επανάληψη.

0.4.5 Συμπεριφορά του κανονικού FEM και του Adaptive FEM

Όπως είπαμε προηγουμένως, η κύρια διαφορά μεταξύ αυτών των αλγορίθμων είναι ο τρόπος με τον οποίο επιλέγουν ποια τρίγωνα θα τριγωνοποιήσουν περαιτέρω σε κάθε βήμα. Ο κανονικός FEM επιλέγει όλα τα τρίγωνα στο πλέγμα, ενώ το Adaptive FEM επιλέγει μόνο ένα ποσοστό από αυτά. Ως αποτέλεσμα, αυτές οι ιδιότητες μπορούν να μεταφραστούν στις ακόλουθες συμπεριφορές για τους δύο αλγόριθμους.

Ο αλγόριθμος του κανονικού FEM καθώς εκτελείται, δημιουργεί ένα ομοιόμορφο πλέγμα στο πεδίο ορισμού, αλλά οι εκτιμητές σφαλμάτων ανά τρίγωνο δεν είναι απαραίτητα ίσοι.

Από την άλλη πλευρά, ο προσαρμοστικός αλγόριθμος FEM καθώς εκτελείται, μπορεί να δημιουργήσει περιοχές στο πλέγμα με διαφορετική πυκνότητα μεταξύ τους, αλλά οι εκτιμητές σφαλμάτων ανά τρίγωνο τείνουν να γίνονται ίσοι.

0.4.6 Σύγκλιση του κανονικού FEM στην ασθενή λύση

Στην υποενότητα 4.5.1 δείχνουμε τη σύγκλιση του Κανονικού FEM προς την ασθενή λύση. Αυτό το κάνουμε εκμεταλλευόμενοι την ιδιότητα του κανονικού FEM, ότι σε κάθε βήμα ο τριγωνισμός T_n γίνεται ομοιόμορφα πυκνότερος σε ολόκληρο το πεδίο ορισμού, αυτό μεταφράζεται στη συνέχεια στην ιδιότητα, ότι για κάθε $v \in V$ υπάρχει μια ακολουθία $\{\phi_n\}_{n=1}^{\infty}$ με $\phi_n \in V_n$, τέτοια ώστε $lim_{n\to\infty}\|v - \phi_n\|_V = 0$ και χρησιμοποιώντας αυτό αποδεικνύουμε το θεώρημα 4.5.1, το οποίο λέει ότι το u_n για το T_n συγκλίνει στο u.

0.4.7 Σύγκλιση του Adaptive FEM στην ασθενή λύση

Στην υποενότητα 4.5.2 δείχνουμε τη σύγκλιση του Adaptive FEM με την ασθενή λύση. Τώρα για τον αλγόριθμο Adaptive FEM έχουμε ένα διαφορετικό κριτήριο που τον περιγράφει, το οποίο είναι το κλειδί για να αποδείξουμε τη σύγκλισή του στην ασθενή λύση.

Βλέπουμε ότι στην υποενότητα 0.4.4, περιγράψαμε δύο περιπτώσεις Adaptive FEM με τα αντίστοιχα κριτήρια, αλλά μπορούμε να χρησιμοποιήσουμε ένα γενικό κριτήριο για να περιγράψουμε

και τα δύο, το οποίο στο βήμα n^{th} είναι

$$e_n(T_{M_n}) \geq \theta_n \cdot e_n, \tag{25}$$

όπου $0 < \theta_n \leq 1$ είναι η παράμετρος σήμανσης ($errorPercentage$) της υποενότητας 0.4.4, $T_{M_n} \subseteq T_n$ είναι το σύνολο των τριγώνων που σημειώνουμε για τριγωνοποίηση στο T_n και

$$e_n = e(u_n, T_n) = \left(\sum_{t \in T_n} e_{tr}^2(u_n, f, t) \right)^{1/2}, \tag{26}$$

$$e_n(T_{M_n}) = e(u_n, T_{M_n}) = \left(\sum_{t \in T_{M_n}} e_{tr}^2(u_n, f, t) \right)^{1/2}. \tag{27}$$

Όπως μπορούμε να δούμε το e_n είναι το φράγμα σφάλματος από το (18), όπου

$$\|u - u_n\|_{H^1_{0,D}(\Omega)} \leq c^{1/2} \left(\sum_{t \in T_n} e_{tr}^2(u_n, f, t) \right)^{1/2}.$$

Επομένως, το κριτήριο (25) είναι η ενσωματωμένη ιδιότητα του Adaptive FEM που το περιγράφει και αυτό που κάνει τον Adaptive διαφορετικό από το κανονικό FEM. Αυτή είναι και η βασική ιδιότητα που βοηθάει στην απόδειξη σύγκλισης, γιατί μας οδηγεί στην απόδειξη του παρακάτω θεωρήματος, που ονομάζεται ιδιότητα συστολής και περιγράφεται επίσης στο 4.5.4.

Theorem 0.4.1. *(Ιδιότητα συστολής)*
Έστω $\theta_n \in (0, 1]$, η παράμετρος σήμανσης της AFEM με $0 < \theta_{min} \leq \theta_n$ για κάθε n και θ_{min} ένας σταθερός αριθμός, και έστω $\{T_n, V_n, u_n\}_{n \geq 0}$ η ακολουθία χώρων πλέγματος, χώρων πεπερασμένων στοιχείων (όπως ορίζεται στην ενότητα 4.5.2) και διακριτές λύσεις (που παράγονται από το AFEM για το πρόβλημα πεπερασμένων στοιχείων (4.9) σε κάθε T_n).

Στη συνέχεια υπάρχουν $\gamma > 0$ και $0 < a_n < 1$, που εξαρτάται αποκλειστικά από την κανονικότητα σχήματος του T_0, το b (ο ελάχιστος αριθμός κομματιών που ένα στοιχείο χωρίζεται σε ένα βήμα τριγωνισμού) και το $0 < \theta_n \leq 1$, έτσι ώστε

$$\|\nabla(u - u_{n+1})\|^2_{L^2(\Omega)} + \gamma e^2_{n+1} \leq a_n^2 (\|\nabla(u - u_n)\|^2_{L^2(\Omega)} + \gamma e_n^2), \tag{28}$$

όπου $u \in H^1_{g,D}(\Omega)$ η ασθενής λύση του (4.8).

Να σημειώσουμε εδώ ότι το θ_{min} το έχουμε ήδη ορίσει στο κριτήριο του Adaptive FEM Algorithm (υποενότητα 4.4.3) όπου για την πρώτη περίπτωση έχουμε $\theta_{min} = \theta_n = errorPercentage$, ενώ για την δεύτερη έχουμε $\theta_{min} = errorPercentageBoundary$.

Τέλος με αυτό το θεώρημα βλέπουμε ότι, η ποσότητα $\|\nabla(u-u_n)\|^2_{L^2(\Omega)} + \gamma e^2_{n+1}$ γίνεται μικρότερη καθώς $n \to \infty$ και κατά προέκταση $lim_{n\to\infty} \|\nabla(u - u_n)\|_{L^2(\Omega)} = 0$, το οποίο αποδεικνύει στο θεώρημα 4.5.5 τη σύγκλιση της προσεγγιστικής λύσης u_n του AFEM στην ασθενή λύση u του ασθενούς προβλήματός μας.

0.5 Ασυμπτωτική ανάλυση και σύγκριση κανονικού και Adaptive FEM

Στην ενότητα 4.6 βλέπουμε την ασυμπτωτική ανάλυση του κανονικού και Adaptive FEM.

Στην υποενότητα 4.6.2 συμπεραίνουμε ότι για τον κανονικό FEM έχουμε την συνάρτηση υπολογιστικής πολυπλοκότητας

$$C_{regular} = O((k+1)^{fn}), \tag{29}$$

όπου fn είναι η συνάρτηση

$$fn = fn_{regular}(errorBoundary). \tag{30}$$

Ενώ για τον Adaptive FEM, έχουμε από την υποενότητα 4.6.3 την συνάρτηση υπολογιστικής πολυπλοκότητας

$$C_{adaptive} = O\left(\prod_{j=0}^{fn}(k \cdot l(j)+1)\right), \tag{31}$$

όπου για το Κριτήριο 1, fn είναι η συνάρτηση

$$fn = fn_{adaptive}(errorBoundary,\ errorPercentage) \tag{32}$$

και για το Κριτήριο 2 είναι

$$fn = fn_{adaptive}(errorBoundary,\ errorPercentageBoundary). \tag{33}$$

0.5.1 Σύγκριση μεταξύ του κανονικού FEM και του Adaptive FEM

Αυτό που βλέπουμε στην πραγματικότητα είναι ότι και οι δύο συναρτήσεις υπολογιστικής πολυπλοκότητας $C_{regular}$ και $C_{adaptive}$ είναι ασυμπτωτικά ίσες με τον αριθμό των τριγώνων που δημιουργεί κάθε αλγόριθμος και στην υποενότητα 4.6.4 βλέπουμε ότι, για πολύ μικρά $errorBoundary$ το $C_{adaptive}$ γίνεται ταχύτερο από το $C_{regular}$.

Επίσης για τον αλγόριθμο Adaptive FEM, με βάση τα δύο κριτήρια που αναφέραμε στην υποενότητα 0.4.4, έχουμε δύο περιπτώσεις και στην υποενότητα 4.6.5 βλέπουμε ότι για πολύ μικρό $errorBoundary$, το δεύτερο κριτήριο μπορεί να μας δώσει έναν πιο γρήγορο αλγόριθμο από τον πρώτο. Το επιτυγχάνει αυτό τριγωνοποιώντας μεγαλύτερο ποσοστό τριγώνων στις πρώτες βελτιώσεις πλέγματος, όπου το κόστος είναι ακόμα χαμηλό, ενώ μικραίνει το ποσοστό καθώς ο αριθμός των τριγώνων μεγαλώνει και το κόστος γίνεται μεγαλύτερο.

0.6 Ένα παράδειγμα

Ως παράδειγμα για την εφαρμογή της παραπάνω θεωρίας θα χρησιμοποιήσουμε το ακόλουθο πρόβλημα Poisson:

$$\begin{aligned} -\Delta u &= 1 \ in\ \Omega \subseteq R^2\ with\ u : \Omega \mapsto R, \\ u &= 0\ on\ \partial\Omega, \end{aligned}$$

όπου $\Omega = (-1,1)^2 \backslash ((0,1) \times (-1,0))$ ένας χώρος L σχήματος. Για να βρούμε μια προσεγγιστική λύση $u \in V_T \cap H_0^1(\Omega)$ για το αντίστοιχο πρόβλημα πεπερασμένων στοιχείων, χρησιμοποιούμε και τους δύο αλγόριθμους Adaptive και κανονικού FEM, και τους συγκρίνουμε.

Για να κατανοήσουμε καλύτερα τη συμπεριφορά αυτών των αλγορίθμων, δείχνουμε την εφαρμογή τους για διαφορετικές περιπτώσεις του *errorBoundary*. Για κάθε περίπτωση εφαρμόζουμε και τους δύο αλγόριθμους, Adaptive και κανονικού FEM, όπου για το Adaptive χρησιμοποιούμε τους δύο τρόπους επανάληψης που παρουσιάσαμε πριν, ένας για το κριτήριο 1 με *errorPercentage* και έναν για το κριτήριο 2 με *errorPercentageBoundary*. Στα παρακάτω σχεδιαγράμματα, πολλές από τις προαναφερόμενες μεταβλητές αντικαθίστανται ως:

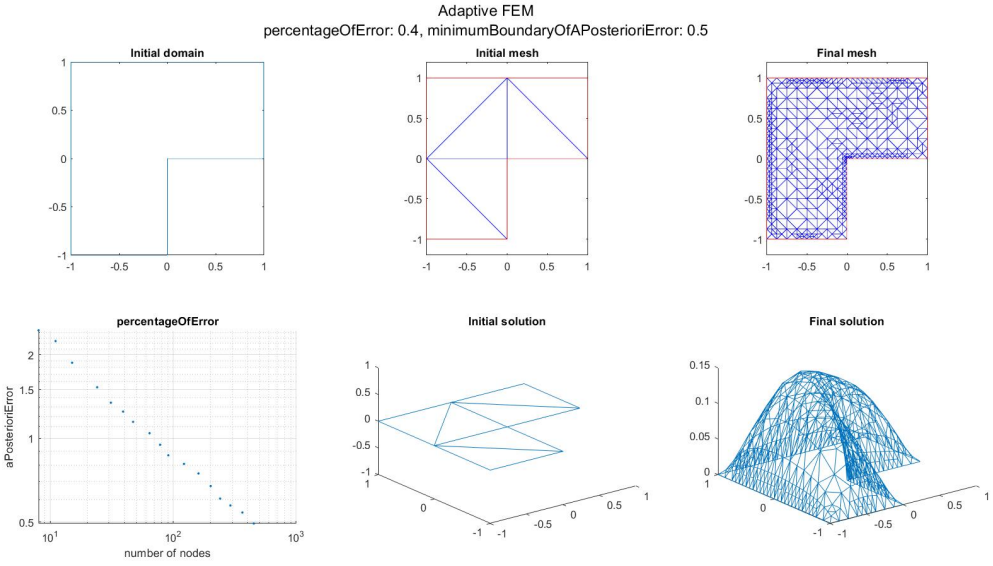
$$\begin{aligned} error &\longrightarrow aPosterioriError, \\ errorBoundary &\longrightarrow minimumBoundaryOfAPosterioriError, \\ errorPercentage &\longrightarrow percentageOfError, \\ errorPercentageBoundary &\longrightarrow adaptivePercentageOfError. \end{aligned}$$

0.6.1 Παράδειγμα για minimumBoundaryOfAPosterioriError = 0.5

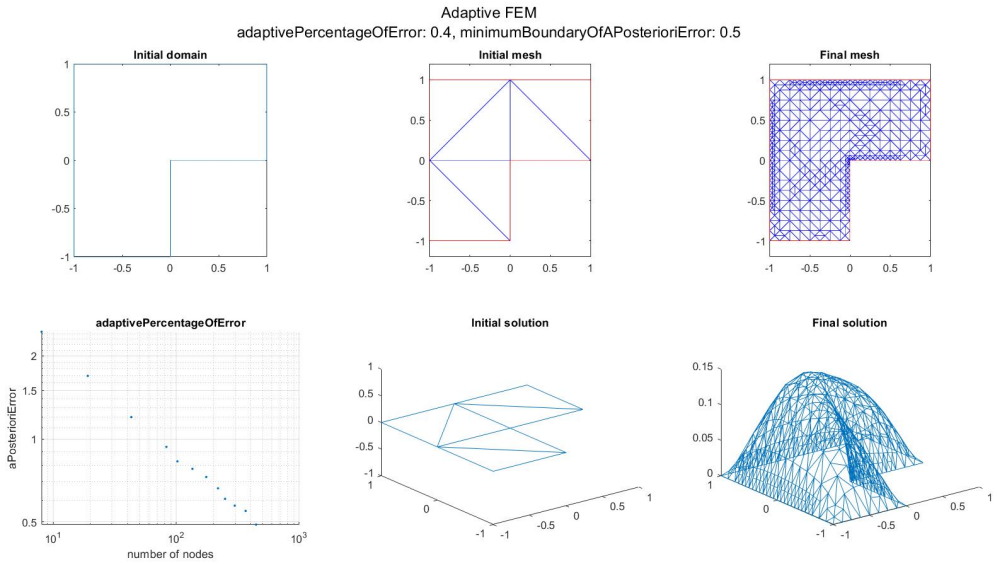
Adaptive FEM

Παίρνουμε κάποια παραδείγματα για διαφορετικά *percentageOfError* και *adaptivePercentageOfError*.

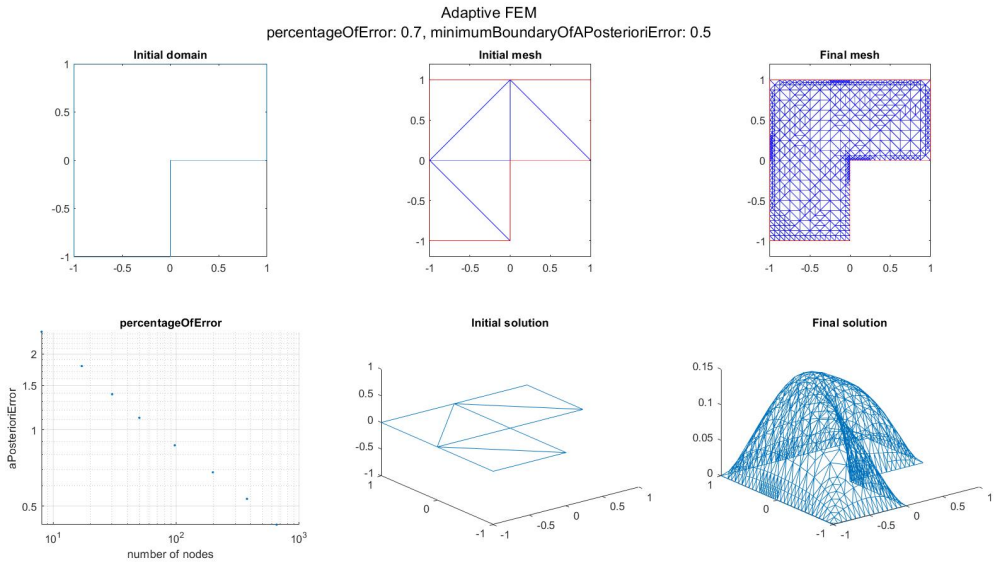
• Κριτήριο 1: $percentageOfError = 0.4$



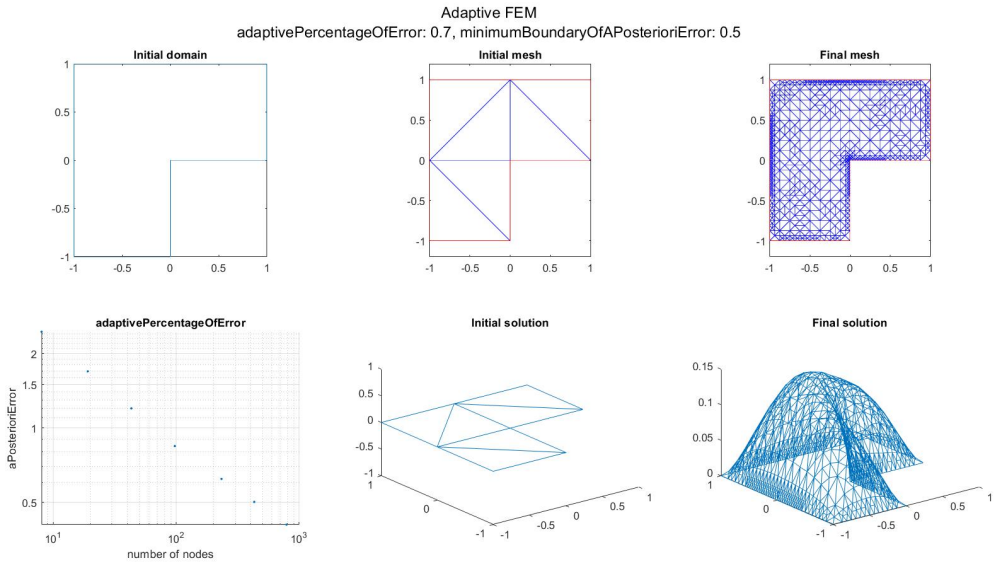
• Κριτήριο 2: $adaptivePercentageOfError = 0.4$



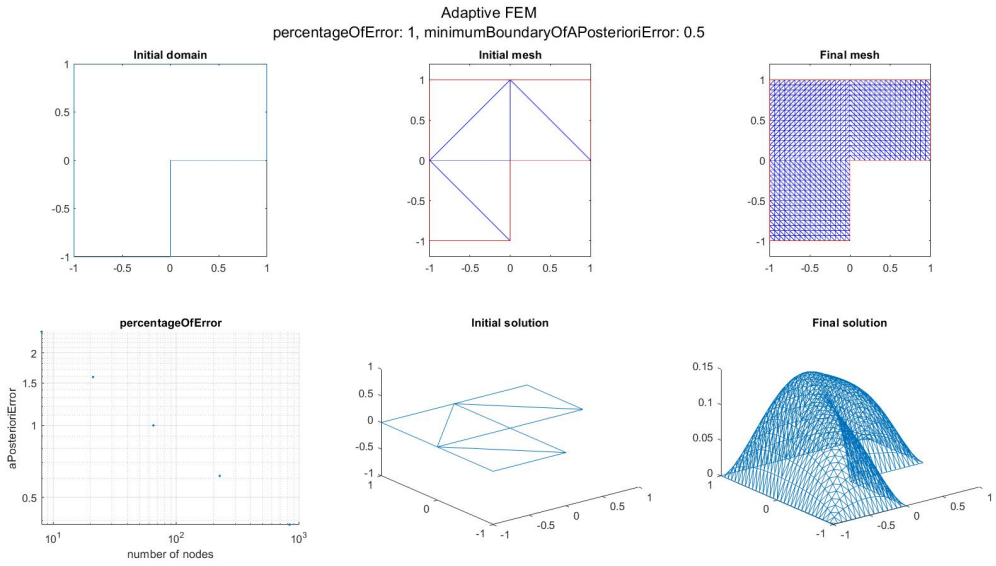
- Κριτήριο 1: $percentageOfError = 0.7$



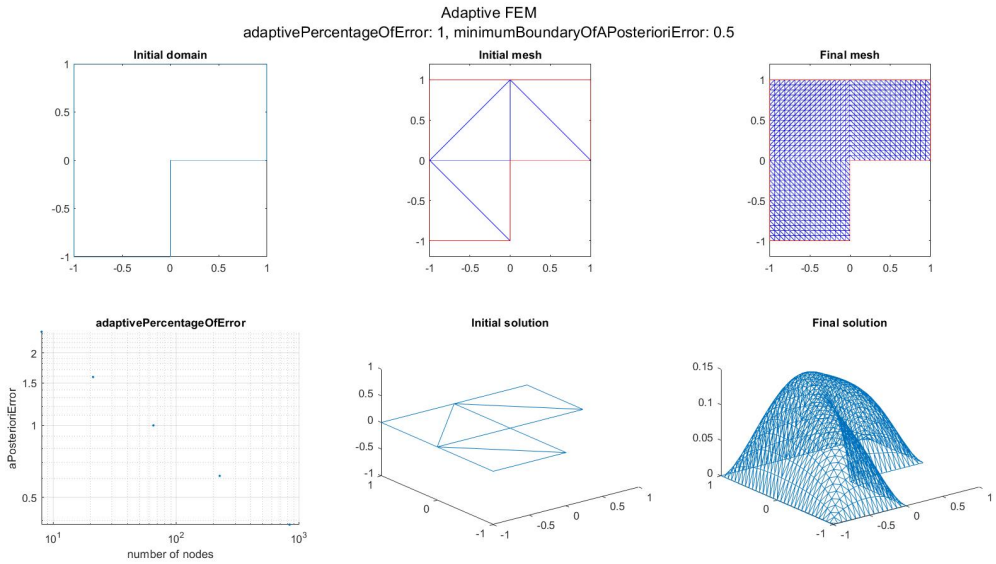
- Κριτήριο 2: $adaptivePercentageOfError = 0.7$



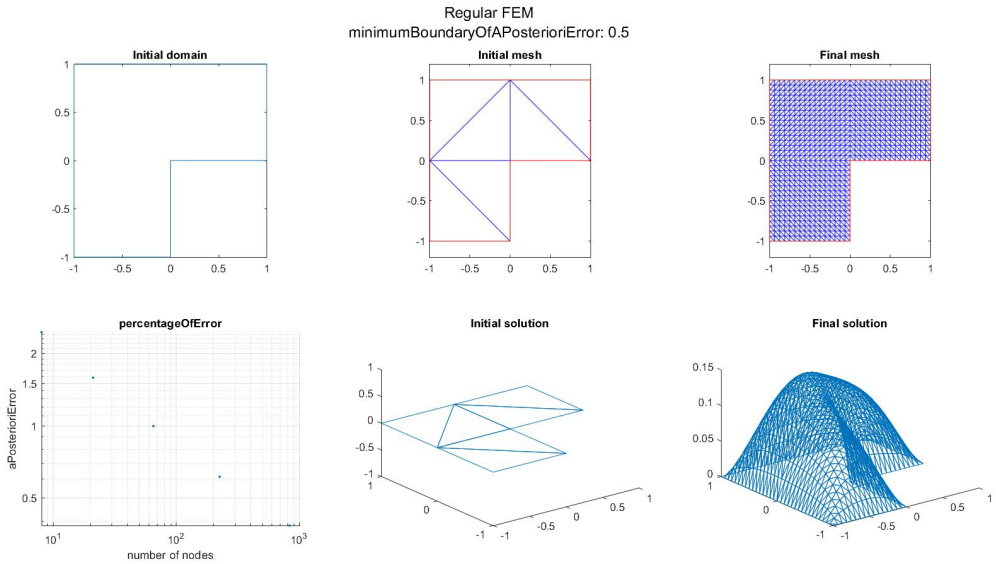
- Κριτήριο 1: $percentageOfError = 1$



- Κριτήριο 2: $adaptivePercentageOfError = 1$



Κανονικός FEM



Από τα παραπάνω διαγράμματα μπορούμε να παρατηρήσουμε μερικά από τα πράγματα που περιγράψαμε στην υποενότητα 4.6.5:

- Τα πλέγματα των Adaptive FEM παραδειγμάτων δεν είναι ομοιόμορφα, σε αντίθεση με το πλέγμα του regular FEM.
- Ο αριθμός των σημείων των scatter plots για το ***aPosterioriError vs number of nodes***, δείχνει πόσες φορές το πρόγραμμα θα τρέξει μέχρι να φτάσει το επιθυμητό *aPosterioriError*.

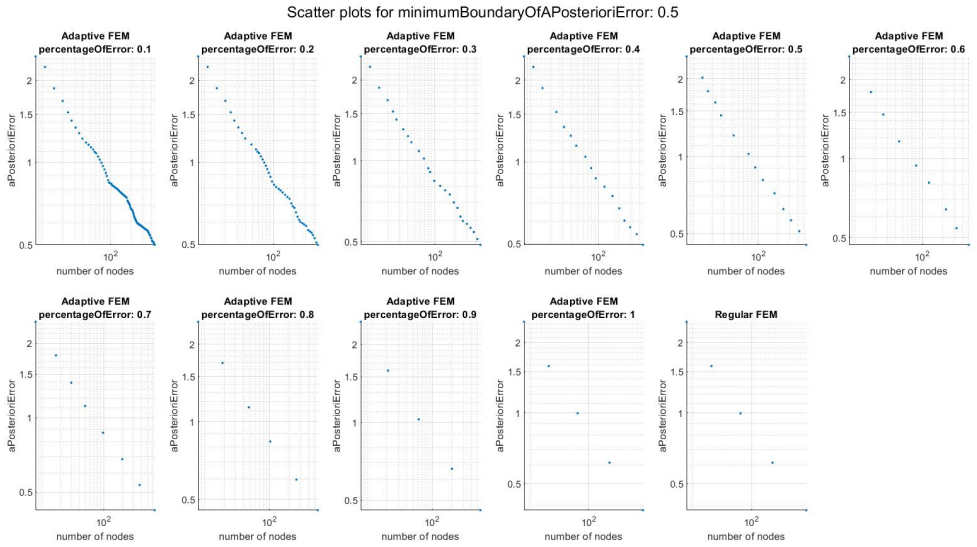
Μπορούμε να δούμε ότι για τον Adaptive FEM Criterion 2 θα τρέξει λιγότερες φορές από ότι για το Criterion 1.

- Καθώς το *percentageOfError* ή το *adaptivePercentageOfError* πλησιάζει το 1, ο Adaptive FEM γίνεται περισσότερο σαν τον regular FEM και όταν φτάσουν το 1, έχουν γίνει ίδιοι.

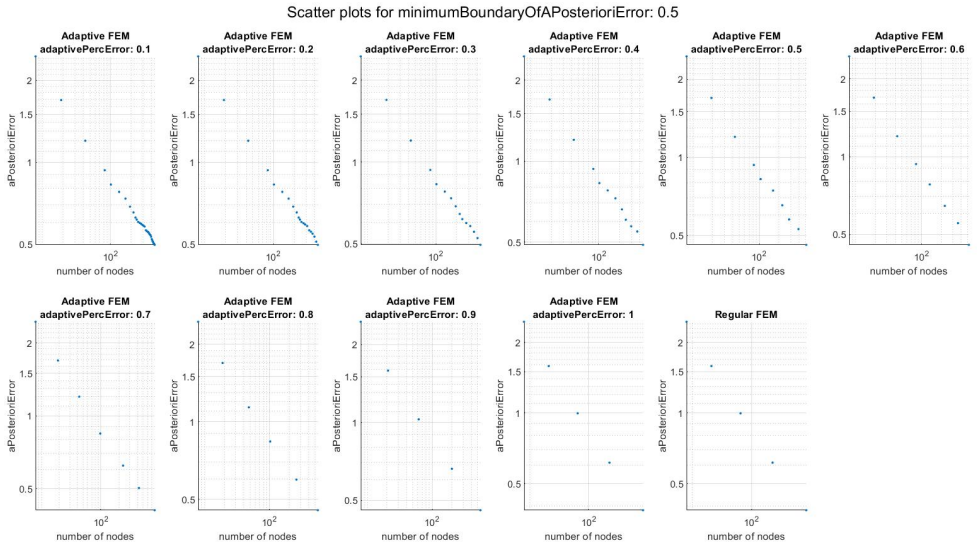
Scatter Log/Log γραφήματα για τον Adaptive και τον κανονικό FEM

Παρακάτω έχουμε μερικά ακόμη παραδείγματα σε scatter log/log γραφήματα (***aPosterioriError vs number of nodes***) για τον Adaptive FEM και τον κανονικό FEM, προκειμένου να δούμε πως το aPosterioriError μειώνεται καθώς οι κόμβοι αυξάνονται.Για τον Adaptive FEM, θα κοιτάξουμε και τα δύο κριτήρια.

• Adaptive FEM με Κριτήριο 1 και κανονικός FEM



• Adaptive FEM με Κριτήριο 2 και κανονικός FEM



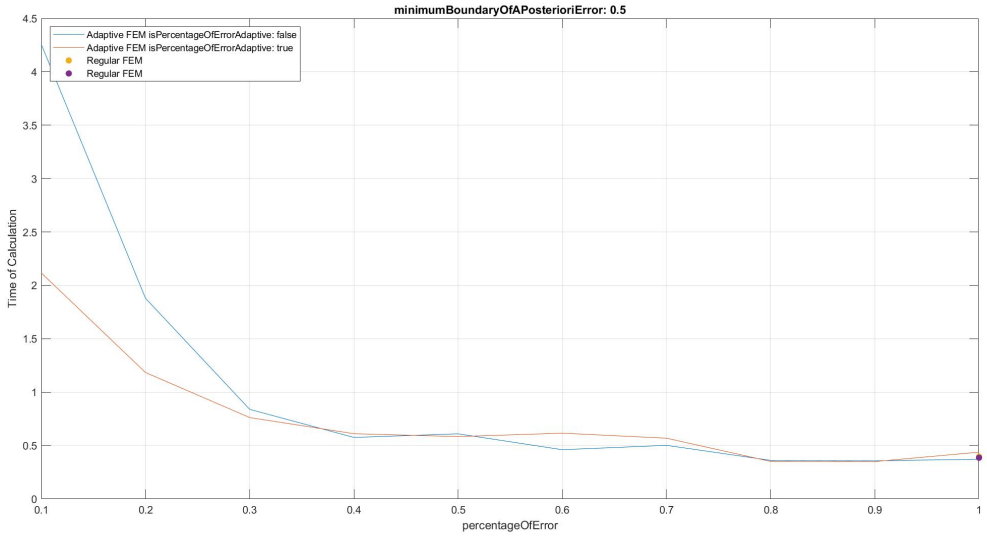
Έτσι τα γραφήματα μας δίνουν μια καλύτερη εικόνα πώς οι αλγόριθμοι συμπεριφέρονται όταν αλλάζουμε το *percentageOfError* ή το *adaptivePercentageOfError* και επιβεβαιώνουν ξανά τα σημεία από τη θεωρία που παρατηρήσαμε με τα προηγούμενα διαγράμματα.

Επίσης μπορούμε να δούμε ότι ο τελικός αριθμός των κόμβων που δημιουργούνται σε κάθε περίπτωση διαφέρουν, αυτός ο αριθμός είναι ανάλογος με τον τελικό αριθμό των τριγώνων που δημιουργούνται και όπως δείξαμε στην ενότητα 4.6, για μικρό *minimumBoundaryOfAPosteriori-Error* μπορεί να μας δώσει μία εικόνα του υπολογιστικού κόστους του αλγορίθμου.

Γράφημα υπολογιστικού χρόνου των Adaptive και κανονικού FEM

Παρακάτω είναι το γράφημα υπολογιστικού χρόνου σε δευτερόλεπτα καθώς το *percentageOfError* αλλάζει για τον Adaptive FEM με Κριτήριο 1, τον Adaptive FEM με Κριτήριο 2 καθώς και τον κανονικό FEM.

Για το Κριτήριο 1 ($isPercentageOfErrorAdaptive = false$) χρησιμοποιούμε κανονικά το *percentageOfError* που έχουμε ορίσει, αλλά για το Κριτήριο 2 ($isPercentageOfErrorAdaptive = true$) με το *percentageOfError* εννοούμε *adaptivePercentageOfError*. Ενώ για τον κανονικό FEM είναι το ίδιο όπως όταν παίρνουμε $percentageOfError = adaptivePercentageOfError = 1$.



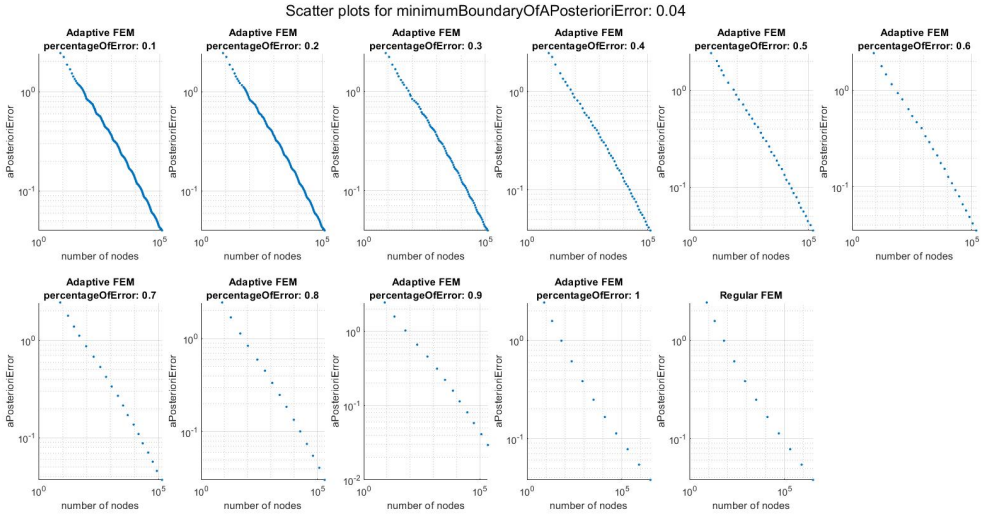
Όπως μπορούμε να δούμε ο Adaptive FEM για το Κριτήριο 2 είναι γρηγορότερος από τον Adaptive FEM για το Κριτήριο 1 (κυρίως λόγω των λιγότερων επαναλήψεων). Επίσης ο κανονικός FEM για τα περισσότερα *percentageOfError* φαίνεται να είναι ταχύτερος από οποιοδήποτε Adaptive, αυτό έρχεται σε αντίθεση με αυτό που είδαμε στην ενότητα 4.6, αλλά αυτό συμβαίνει επειδή είμαστε στην περίπτωση του $minimumBoundaryOfAPosterioriError = 0.5$ και παρόλο που ο αριθμός των τελικών κόμβων για τον κανονικό FEM είναι μεγαλύτερος απο του Adaptive FEM, εδώ το υπολογιστικό κόστος είναι ήδη μικρό σε κάθε επανάληψη. Έτσι ο αριθμός των επαναλήψεων είναι πιο σημαντικός σε αυτό το στάδιο. Καθώς το *minimumBoundaryOfAPosterioriError* γίνεται μικρότερο αυτό θα ξεκινήσει να αλλάζει και ο αριθμός των τελικών κόμβων θα είναι ο κύριος παράγοντας για το υπολογιστικό κόστος και έτσι ο κανονικός FEM θα γίνει πιο αργός.

Έτσι για να το δούμε αυτό ακολουθούν κάποια αποτελέσματα για την πολύ μικρότερη περίπτωση του *minimumBoundaryOfAPosterioriError*.

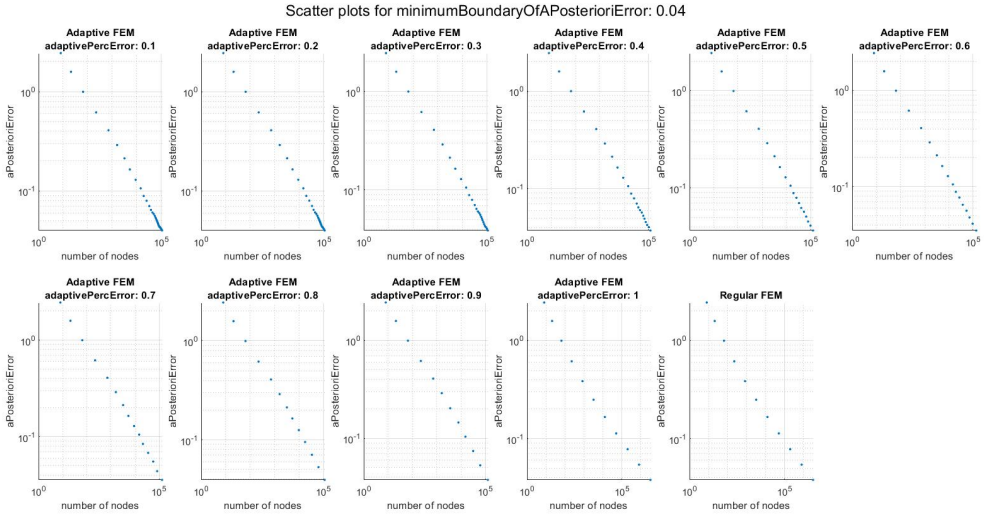
0.6.2 Παράδειγμα για minimumBoundaryOfAPosterioriError = 0.04

Scatter Log/Log γραφήματα για τον Adaptive και κανονικό FEM

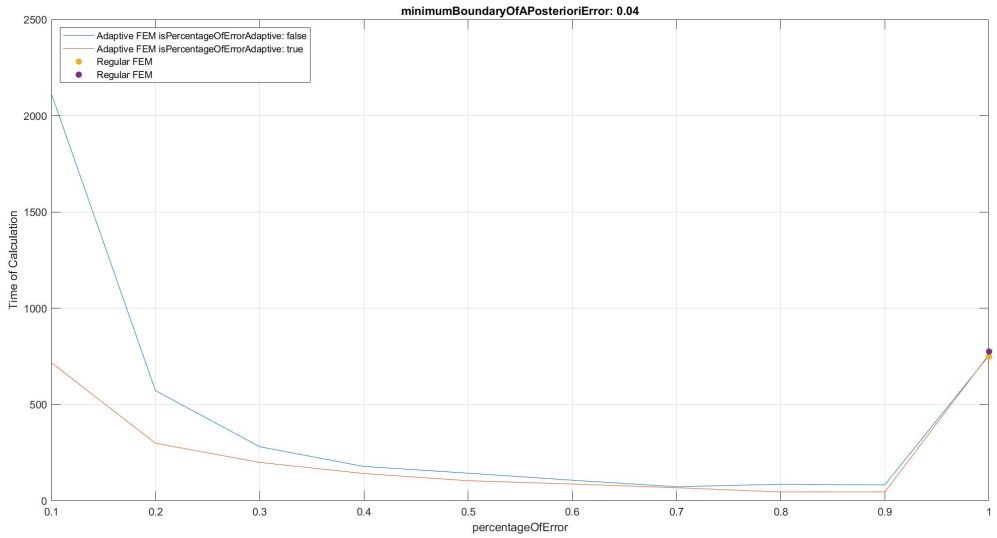
• Adaptive FEM με Κριτήριο 1 και κανονικός FEM



• Adaptive FEM με Κριτήριο 2 και κανονικός FEM



Γράφημα υπολογιστικής πολυπλοκότητας των Adaptive και κανονικού FEM



Τέλος βλέπουμε εδώ ότι για μικρό *minimumBoundaryOfAPosterioriError* ο Adaptive FEM γίνεται γρηγορότερος από τον κανονικό FEM για τα περισσότερα *percentageOfErrors*.

Abstract

In this work, we will deal with the solution of the problem of elliptic partial differential Poisson's equation, with mixed boundary conditions (Dirichlet and Neumann), through the finite element method.

We will develop two types of finite element methods, the regular Finite Element Method (FEM) and the Adaptive Finite Element Method (AFEM).

Regular FEM is the classic finite element method where it works by continuously triangulating the entire domain and calculating the approximate solution and error in each triangulation, in order to reduce the error.

Adaptive FEM is a kind of finite element method with the feature that in each triangulation of its domain, it chooses the triangles to refine, based on a criterion which of them will have the greatest impact on reducing the error. This allows it to be faster than regular FEM.

Chapter 1

Elliptic Boundary Value Problems

The problem we will analyze is the elliptic partial differential **Poisson's equation** with boundary conditions:

$$-\Delta u = f \; in \; \Omega \subseteq R^n \; with \; u : \Omega \mapsto R, \; f : \Omega \mapsto R \tag{1.1}$$

$$u = g_D \; on \; \Gamma_D \; with \; g_D : \Gamma_D \subseteq R^n \mapsto R \tag{1.2}$$
$$(Dirichlet \; boundary \; condition \; on \; \Gamma_D)$$

$$\nabla u \cdot \vec{n} = g_N \; on \; \Gamma_N \; with \; g_N : \Gamma_N \subseteq R^n \mapsto R \tag{1.3}$$
$$(Neumann \; boundary \; condition \; on \; \Gamma_N)$$

with $\partial\Omega = \Gamma_D \cup \Gamma_N$, $\Gamma_D \cap \Gamma_N = \emptyset$ and Γ_D closed set and Γ_N open.

Due to the possibility of corner singularities, we will be seeking weak solution to above Poisson differential equation with mixed boundaries (Dirichlet and Neumann) by weakening the differentiability requirements on u.

Let us suppose that u is a classical solution of the problem ($u \in C^2(\Omega) \cap C(\overline{\Omega})$) and let's take any $v \in C^1_{0,D}(\Omega) := \{u \in C^1(\Omega) : u = 0 \; on \; \Gamma_D\}$ then we multiply and integrate (1.1).

$$-\int_\Omega \Delta u v dx \; = \; \int_\Omega f v dx \;\; \forall v \in C^1_{0,D}(\Omega). \tag{1.4}$$

We apply integration by parts formula, following by the product rule ($\Delta uv = -\nabla u \cdot \nabla v + \nabla \cdot (\nabla u v)$) and the previous equation becomes

$$\int_\Omega \nabla u \cdot \nabla v dx \; - \; \int_\Omega \nabla \cdot ((\nabla u) v) dx \; = \; \int_\Omega f v dx \;\; \forall v \in C^1_{0,D}(\Omega); \tag{1.5}$$

from Gauss - Green Theorem (Appendix 5.1), we have

$$\int_\Omega \nabla u \cdot \nabla v dx \; - \; \int_{\partial\Omega} ((\nabla u) v) \cdot \vec{n} dS \; = \; \int_\Omega f v dx \;\; \forall v \in C^1_{0,D}(\Omega). \tag{1.6}$$

In order for this equality to make sense we no longer need to assume that $u \in C^2(\Omega)$, it is sufficient that $u \in L_2(\Omega)$ and $\frac{\partial u}{\partial x_i} \in L_2(\Omega), i = 1, 2, \cdots, n$ (weak derivatives), therefore $u \in H^1(\Omega)$ with $H^1(\Omega) = \{u \in L_2(\Omega) : \frac{\partial u}{\partial x_i} \in L_2(\Omega),\ i = 1, 2, \cdots, n\}$ a Sobolev space.

Also we need to keep in mind the Dirichlet boundary condition $u = g_D$ on Γ_D, so we need to restrict $u \in H^1_{g,D}(\Omega) = \{u \in H^1(\Omega) : u = g_D \text{ on } \Gamma_D\} \subset H^1(\Omega)$.

Also for the same reason we no longer need $v \in C^1_0(\Omega)$, we can simplify that and take $v \in H^1_{0,D}(\Omega)$ where $H^1_{0,D}(\Omega) = \{u \in H^1(\Omega) : u = 0 \ on \ \Gamma_D\}$ and $C^1_0(\Omega) \subset H^1_{0,D}(\Omega)$. So we transform the problem (1.1), (1.2), (1.3) of the search of the classical solution $u \in C^2(\Omega) \cap C(\overline{\Omega})$ to the below problem:

find $u \in H^1_{g,D}(\Omega)$ satisfying

$$\int_\Omega \nabla u \cdot \nabla v dx \ - \ \int_{\partial\Omega} ((\nabla u)v) \cdot \vec{n} dS \ = \ \int_\Omega f v dx \ \ \forall v \in H^1_{0,D}(\Omega), \tag{1.7}$$

$$\nabla u \cdot \vec{n} = g_n \ on \ \Gamma_N. \tag{1.8}$$

We modify further the equation (1.7) based on the boundary data (1.8) and the space $H^1_{0,D}(\Omega)$ where v belongs. So since $v = 0$ on Γ_D, we have

$$\int_\Omega \nabla u \cdot \nabla v dx \ - \ \int_{\Gamma_N} ((\nabla u)v) \cdot \vec{n} dS = \int_\Omega f v dx \ \ \forall v \in H^1_{0,D}(\Omega) \tag{1.9}$$

from (1.8), we have

$$\int_\Omega \nabla u \cdot \nabla v dx \ - \ \int_{\Gamma_N} g_N v dS = \int_\Omega f v dx \ \ \forall v \in H^1_{0,D}(\Omega) \tag{1.10}$$

and finally we have

$$\int_\Omega \nabla u \cdot \nabla v dx = \int_\Omega f v dx + \int_{\Gamma_N} g_N v dS \ \ \forall v \in H^1_{0,D}(\Omega). \tag{1.11}$$

So now the problem becomes find $u \in H^1_{g,D}(\Omega)$ such that

$$\int_\Omega \nabla u \cdot \nabla v dx = \int_\Omega f v dx + \int_{\Gamma_N} g_N v dS \ \ \forall v \in H^1_{0,D}(\Omega), \tag{1.12}$$

which we will call the weak form of the problem.

1.1 Existence and uniqueness of a solution to the weak problem

We need to prove the existence and uniqueness of the solution u on $H^1_{g,D}$ for the problem (1.12), but first we will find an equivalent form of our weak problem with its solution defined in a linear space $H^1_{0,D}$.

1.1.1 Equivalent form of the weak problem on space $H^1_{0,D}$

We can find a function G with the following properties:

- $G \in H^1(\Omega)$,
- $G = g_D$ on Γ_D.

That means that $G \in H^1_{g,D}(\Omega)$.

Now if $u \in H^1_{g,D}(\Omega)$ exists, then by taking a $G \in H^1_{g,D}(\Omega)$ as described above, the function $w = u - G$ would also exists. That would mean that $w = u - G \in H^1(\Omega)$ and $w = 0$ on Γ_D, so $w \in H^1_{0,D}(\Omega)$. For that $G \in H^1_{g,D}(\Omega)$, the reverse is also true, if a $w \in H^1_{0,D}(\Omega)$ exists then we can find $u \in H^1_{g,D}(\Omega)$ such that $u = w + G$. So for a specific $G \in H^1_{g,D}(\Omega)$, if we have either u or w we could find one another.

Using that idea we replace u with $w + G$ on (1.12) and we have

$$\int_\Omega \nabla(w+G)\cdot\nabla v dx = \int_\Omega f v dx + \int_{\Gamma_N} g_N v dS \;\; \forall v \in H^1_{0,D}(\Omega), \tag{1.13}$$

that means

$$\int_\Omega \nabla w\cdot\nabla v dx = \int_\Omega f v dx - \int_\Omega \nabla G\cdot\nabla v dx + \int_{\Gamma_N} g_N v dS \;\; \forall v \in H^1_{0,D}(\Omega). \tag{1.14}$$

So the problem (1.12) of finding $u \in H^1_{g,D}(\Omega)$ is equivalent of finding $w \in H^1_{0,D}(\Omega)$ for (1.14) (for a chosen G with the properties as above).

Finally we adopt the following notation:

$$a(w,v) = \int_\Omega \nabla w\cdot\nabla v dx \qquad for\ w, v \in H^1_{0,D}(\Omega), \tag{1.15}$$

$$l(v) = \int_\Omega f v dx - \int_\Omega \nabla G\cdot\nabla v dx + \int_{\Gamma_N} g_N v dS \qquad for\ v \in H^1_{0,D}(\Omega). \tag{1.16}$$

So the problem (1.14) can be written as find $w \in H^1_{0,D}(\Omega)$ such that

$$a(w,v) = l(v) \;\; \forall v \in H^1_{0,D}(\Omega). \tag{1.17}$$

We shall prove the existence of a unique solution to this problem by exploiting the Lax - Milgram Theorem (Appendix 5.2).

Proof of existence and uniqueness

1.1.2 $H^1_{0,D}(\Omega)$ is a Hilbert space

First we show that

$$H^1_{0,D}(\Omega) = \{u \in H^1(\Omega) : u = 0\ on\ \Gamma_D\} \subseteq H^1(\Omega) \tag{1.18}$$

is a Hilbert space with norm

$$\|\cdot\|_{H^1_{0,D}(\Omega)} = \|\cdot\|_{H^1(\Omega)} = (\cdot,\cdot)^{1/2}_{H^1(\Omega)}, \tag{1.19}$$

where $H^1(\Omega)$ is a known Hilbert space with inner product

$$(w,v)_{H^1(\Omega)} = \int_\Omega wv dx + \sum_{i=1}^{n} \left(\int_\Omega \frac{\partial w}{\partial x_i} \frac{\partial v}{\partial x_i} dx \right) \ \forall w, v \in H^1(\Omega). \tag{1.20}$$

Since $H^1_{0,D}(\Omega) \subset H^1(\Omega)$ we can take the same inner product i.e.,

$$(w,v)_{H^1_{0,D}(\Omega)} = (w,v)_{H^1(\Omega)} \ \forall w, v \in H^1_{0,D}(\Omega). \tag{1.21}$$

We begin by showing that $H^1_{0,D}(\Omega)$ is a linear subspace. Because $H^1_{0,D}(\Omega) \subseteq H^1(\Omega)$ and $H^1(\Omega)$ is a linear space, as a consequence that is a Hilbert space, it's sufficient to prove that $H^1_{0,D}(\Omega)$ is a linear subspace of $H^1(\Omega)$. So $\forall u, v \in H^1_{0,D}(\Omega) \subseteq H^1(\Omega),\ \forall \lambda, \mu \in R$, because $H^1(\Omega)$ is a linear space we have

$$\lambda u + \mu v \in H^1(\Omega)$$

and because $u = 0$, $v = 0$ on Γ_D this gives us

$$\lambda u + \mu v = 0 \ on \ \Gamma_D,$$

so $\lambda u + \mu v \in H^1(\Omega)$ and $\lambda u + \mu v = 0 \ on \ \Gamma_D$, i.e. $\lambda u + \mu v \in H^1_{0,D}(\Omega)$. Thus $H^1_{0,D}(\Omega)$ is a linear subspace of $H^1(\Omega)$.

Also the inner product we defined at the beginning $(\cdot,\cdot)_{H^1_{0,D}(\Omega)}$ and the norm derived from it $\|\cdot\|_{H^1_{0,D}(\Omega)} = (\cdot,\cdot)^{1/2}_{H^1_{0,D}(\Omega)}$ are all defined well because are the same with those on the space $H^1(\Omega)$, with the only difference that are defined in the linear subspace $H^1_{0,D}(\Omega)$. So all the defining properties of an inner product and a norm are still true on this subspace. So $H^1_{0,D}(\Omega)$ is a linear space with an inner product and a norm derived from the inner product.

Next, we need to show that $H^1_{0,D}(\Omega)$ is a closed space in $H^1(\Omega)$, this will be true if for every sequence $u_n \in H^1_{0,D}(\Omega)$ that converges on a $u \in H^1(\Omega)$, we have that $u \in H^1_{0,D}(\Omega)$. So we take a $u_n \in H^1_{0,D}(\Omega)$ that converges to a u on $H^1(\Omega)$ and we have that

$$\lim_{n\to\infty} \|u - u_n\|_{H^1(\Omega)} = 0,$$

then from continuity of trace function on boundary $\Gamma = \partial\Omega$ for $u - u_n \in H^1(\Omega)$ (Appendix 5.5), there is a $c \geq 0$ such as

$$\|u - u_n\|_{L_2(\Gamma)} \leq c\|u - u_n\|_{H^1(\Omega)}$$

and so we have

$$\lim_{n\to\infty} \|u - u_n\|_{L_2(\Gamma)} = 0.$$

This means

$$\lim_{n\to\infty} \left(\int_{\Gamma_D} (u - u_n)^2_{|\Gamma}(x) dx + \int_{\Gamma_N} (u - u_n)^2_{|\Gamma}(x) dx \right)^{1/2} = 0,$$

which implies

$$\lim_{n\to\infty} \int_{\Gamma_D} (u - u_n)^2_{|\Gamma_D}(x) dx = 0$$

and because $u_n \in H^1_{0,D}(\Omega) \subseteq H^1(\Omega)$, we have $u_n = 0$ on Γ_D, so the equation becomes

$$\lim_{n\to\infty} \int_{\Gamma_D} u^2_{|\Gamma_D}(x)dx = 0,$$

that means

$$\int_{\Gamma_D} u^2_{|\Gamma_D}(x)dx = 0$$

and because $u^2 \geq 0$, we have $u = 0$ on Γ_D for almost every $x \in \Gamma_D$, from which we conclude that $u \in H^1_{0,D}(\Omega)$. So $H^1_{0,D}(\Omega)$ is a closed space in $H^1(\Omega)$.

Finally because we have proved that $H^1_{0,D}(\Omega)$ is a closed linear subspace of the Hilbert space $H^1(\Omega)$, with the same norm derived from an inner product as $H^1(\Omega)$, we have that $H^1_{0,D}(\Omega)$ is also a Hilbert space.

1.1.3 Proof of condition (a) of Lax - Miligram Theorem

It is obvious that $a(w,v) = \int_\Omega \nabla w \cdot \nabla v dx$ is bilinear functional on $H^1_{0,D}(\Omega) \times H^1_{0,D}(\Omega)$ and we show that satisfies the condition (a) of Lax - Miligram Theorem (Appendix 5.2).

So for every $v \in H^1_{0,D}(\Omega)$ we have

$$a(v,v) = \int_\Omega \nabla v \cdot \nabla v dx = \int_\Omega |\nabla v|^2 dx = \sum_{i=1}^n \int_\Omega |\frac{\partial v}{\partial x_i}(x)|^2 dx$$

from Poincaré - Friedrichs inequality (Appendix 5.3), there exists a $c_*(\Omega) \geq 0$, such that, for every $v \in H^1_{0,D}(\Omega)$

$$\int_\Omega |v(x)|^2 dx \leq c_* \sum_{i=1}^n \int_\Omega |\frac{\partial v}{\partial x_i}(x)|^2 dx = c_* a(v,v).$$

So, if we take

$$c_* a(v,v) + a(v,v) \geq \int_\Omega |v(x)|^2 dx + \int_\Omega \nabla v \cdot \nabla v dx$$

we conclude

$$a(v,v) \geq \frac{1}{c_* + 1} \|v(x)\|^2_{H^1_{0,D}(\Omega)}, \tag{1.22}$$

so we proved the condition (a) with $c_0 = \frac{1}{c_*+1}$.

1.1.4 Proof of condition (b) of Lax - Miligram Theorem

Now we prove for $a(w,v)$ the condition (b) of Lax - Miligram Theorem (Appendix 5.2). So for every $w, v \in H^1_{0,D}(\Omega)$ we have

$$|a(w,v)| = |\sum_{i=1}^n \int_\Omega \frac{\partial w}{\partial x_i}\frac{\partial v}{\partial x_i} dx| \leq \sum_{i=1}^n |\int_\Omega \frac{\partial w}{\partial x_i}\frac{\partial v}{\partial x_i}| dx$$

and from Cauchy - Schwarz inequality (Appendix 5.4) and $\frac{\partial w}{\partial x_i}, \frac{\partial v}{\partial x_i} \in L^2(\Omega)$ for $i = 1, \cdots, n$, we have

$$|a(w,v)| \leq \sum_{i=1}^{n} \left(\left(\int_\Omega |\frac{\partial w}{\partial x_i}|^2 dx \right)^{1/2} \left(\int_\Omega |\frac{\partial v}{\partial x_i}|^2 dx \right)^{1/2} \right)$$

$$\leq \left(\int_\Omega \sum_{i=1}^{n} |\frac{\partial w}{\partial x_i}|^2 dx \right)^{1/2} \left(\int_\Omega \sum_{i=1}^{n} |\frac{\partial v}{\partial x_i}|^2 dx \right)^{1/2} \leq \|w\|_{H^1_{0,D}(\Omega)} \|v\|_{H^1_{0,D}(\Omega)}.$$

Therefore, for every $w, v \in H^1_{0,D}(\Omega)$ we have $|a(w,v)| \leq \|w\|_{H^1_{0,D}(\Omega)} \|v\|_{H^1_{0,D}(\Omega)}$ and so condition (b) is proved with $c_1 = 1$.

1.1.5 Proof of condition (c) of Lax - Miligram Theorem

It is obvious that $l(v)$ for $v \in H^1_{0,D}(\Omega)$ is a linear function and now we prove that satisfies the condition (c).

For every $v \in H^1_{0,D}(\Omega) \subseteq L_2(\Omega)$, we have

$$|l(v)| = |\int_\Omega fv dx - \int_\Omega \nabla G \cdot \nabla v dx + \int_{\Gamma_N} g_N v dS| \leq |\int_\Omega fv dx| + \sum_{i=1}^{n} |\int_\Omega \frac{\partial G}{\partial x_i} \frac{\partial v}{\partial x_i} dx| + |\int_{\Gamma_N} g_N v dS|,$$

also from Cauchy - Schwarz inequality (Appendix 5.4) for $f, v, \frac{\partial v}{\partial x_i}, \frac{\partial G}{\partial x_i} \in L_2(\Omega)$ with $i = 1, 2, \cdots, n$ and from Cauchy - Schwarz inequality (Appendix 5.4) for $g_N, v_{|\Gamma_N} \in L_2(\Gamma_N)$, we have

$$|l(v)| \leq \left(\int_\Omega |f|^2 dx \right)^{1/2} \left(\int_\Omega |v|^2 dx \right)^{1/2} + \sum_{i=1}^{n} \left(\left(\int_\Omega |\frac{\partial G}{\partial x_i}|^2 dx \right)^{1/2} \left(\int_\Omega |\frac{\partial v}{\partial x_i}|^2 dx \right)^{1/2} \right)$$
$$+ \left(\int_{\Gamma_N} |g_N|^2 dS \right)^{1/2} \left(\int_{\Gamma_N} |v|^2 dS \right)^{1/2}.$$

So, because $\|v\|_{L_2(\Omega)} \leq \|v\|_{H^1_{0,D}(\Omega)}$ and $\|\frac{\partial v}{\partial x_i}\|_{L_2(\Omega)} \leq \|v\|_{H^1_{0,D}(\Omega)}$, we have

$$|l(v)| \leq \left(\int_\Omega |f|^2 dx \right)^{1/2} \|v\|_{H^1_{0,D}(\Omega)} + \sum_{i=1}^{n} \left(\left(\int_\Omega |\frac{\partial G}{\partial x_i}|^2 dx \right)^{1/2} \|v\|_{H^1_{0,D}(\Omega)} \right)$$
$$+ \left(\int_{\Gamma_N} |g_N|^2 dx \right)^{1/2} \|v\|_{L_2(\Gamma_N)},$$

and by writing as $c_1 = \left(\int_\Omega |f|^2 dx \right)^{1/2} + \sum_{i=1}^{n} \left(\int_\Omega |\frac{\partial G}{\partial x_i}|^2 dx \right)^{1/2} \geq 0$, we have

$$|l(v)| \leq c_1 \|v\|_{H^1_{0,D}(\Omega)} + \left(\int_{\Gamma_N} |g_N|^2 dS \right)^{1/2} \|v\|_{L_2(\Gamma_N)}.$$

Now because of the theorem of continuity of trace function on Neumann boundary Γ_N for $v \in H^1_{0,D}(\Omega)$ (Appendix 5.6), there is a $c \geq 0$ such that

$$|l(v)| \leq c_1 \|v\|_{H^1_{0,D}(\Omega)} + \left(\int_{\Gamma_N} |g_N|^2 dS \right)^{1/2} c \|v\|_{H^1_{0,D}(\Omega)}.$$

Finally by taking as $c_2 = c_1 + c \left(\int_{\Gamma_N} |g_N|^2 dS \right)^{1/2} \geq 0$, we have

$$|l(v)| \leq c_2 \|v\|_{H^1_{0,D}(\Omega)}, \tag{1.23}$$

which implies that the condition (c) of Lax - Milgram Theorem (Appendix 5.2) is satisfied.

1.1.6 Conclusion for unique solution

So we have the below two equivalent weak forms: find $u \in H^1_{g,D}(\Omega)$ that satisfies

$$\int_\Omega \nabla u \cdot \nabla v dx = \int_\Omega f v dx + \int_{\Gamma_N} g_N v dS \;\; \forall v \in H^1_{0,D}(\Omega) \tag{1.24}$$

and, find $w \in H^1_{0,D}(\Omega)$ that satisfies

$$a(w, v) = l(v) \;\; \forall v \in H^1_{0,D}(\Omega). \tag{1.25}$$

Finally through Lax - Miligram Theorem (Appendix 5.2) we have proven that (1.25) has a unique solution and because (1.24), (1.25) are equivalent, that means (1.24) has also a unique solution.

Chapter 2

The Finite Element Method

Now based on the above we will define two ways to compute the linear finite element method of solving (1.1), (1.2), (1.3), one based on (1.24) and one based on (1.25); they are actually the same. First we will define the spaces and their properties we will use.

2.1 Prerequisites

2.1.1 Definition of linear space V_T

First we define a triangulation of Ω and call it T, but we must keep in mind that depending on Ω, the triangulation process cannot always divide it to exact triangles. This is possible for the inner space but in the boundaries if Ω is not a polygon we have a problem, so for the triangles that are created on the boundaries, their sides maybe are not lines but curves.

Then we define V_T the finite dimensional linear space on T which plays important part to create our finite element space for our problem. $V_T = \{v \in C(\overline{\Omega}) : v_{|\tau} \in P_1 \ \ \forall \tau \in T\}$ where P_1 is the linear space of linear polynomials.

Lemma 2.1.1. *V_T is a linear space.*

Proof. $V_T \subseteq C(\overline{\Omega})$ and $C(\overline{\Omega})$ a known linear space, it is enough to show that V_T is a linear subspace of $C(\overline{\Omega})$.

So $\forall v, w \in V_T$ and $\lambda, \mu \in R$ and because $C(\overline{\Omega})$ is a linear space, we have $\lambda v + \mu w \in C(\overline{\Omega})$. Also for each $\tau \in T$

$$(\lambda v + \mu w)_{|\tau} = (\lambda v)_{|\tau} + (\mu w)_{|\tau} = \lambda v_{|\tau} + \mu w_{|\tau}$$

and because $v_{|\tau}, w_{|\tau} \in P_1$ and P_1 is a known linear space, we have

$$\lambda v_{|\tau} + \mu w_{|\tau} \in P_1 \Rightarrow (\lambda v + \mu w)_{|\tau} \in P_1.$$

So that means V_T is a linear space. □

Also for each vertex x_i of T we define $\phi_i(x) : \Omega \to R$ a continuous piecewise linear function such that $\phi_i(x_i) = 1$ and $\phi_i(x_j) = 0$ if $i \neq j$, so it is clear that $\phi_i \in V_T$.

Lemma 2.1.2. *$\{\phi_i\}_{i=1}^N$ is a basis of linear space V_T, where N is the number of vertices of T including the boundary vertices.*

Proof. First we show that $\{\phi_i\}_{i=1}^N$ are linear independent. Lets assume that are not, then there is a ϕ_k with $k \in \{1, 2, 3 \cdots N\}$ such that $\phi_k = \sum_{i=1}^{k-1} a_i\phi_i + \sum_{i=k+1}^{N} a_i\phi_i$ with $a_i \in R$ $\forall i \in \{1, 2, \cdots, k-1, k+1, \cdots N\}$ not all zero. Then for the vertex $x_k \in T$ we have $\phi_k(x_k) = 1$, so

$$1 = \phi_k(x_k) = \sum_{i=1}^{k-1} a_i\phi_i(x_k) + \sum_{i=k+1}^{N} a_i\phi_i(x_k)$$

and because $\phi_i(x_k) = 0$ $\forall i \in \{1, 2, \cdots, k-1, k+1, \cdots N\}$, and so

$$1 = \phi_k(x_k) = \sum_{i=1}^{k-1} a_i\phi_i(x_k) + \sum_{i=k+1}^{N} a_i\phi_i(x_k) = 0,$$

which is not valid, so our assumption is wrong and $\{\phi_i\}_{i=1}^N$ are linear independent. Also it is easily shown that V_T is spanned by $\{\phi_i\}_{i=1}^N$, which means $V_T =< \{\phi_i\}_{i=1}^N >$. So $\{\phi_i\}_{i=1}^N$ is a basis of linear space V_T. □

So from all the previous we conclude that V_T is a finite linear space, that means for each $v \in V_T$ there is a unique $\overrightarrow{a_v} = (a_{v_1}, a_{v_2}, \cdots, a_{v_N}) \in R^N$ such that $v = \sum_{i=1}^N a_{v_i}\phi_i(x)$. The reverse is also true, for each $\overrightarrow{a} \in R^N$ there is a unique $v \in V_T$, such that $v = \sum_{i=1}^N a_i\phi_i(x)$. So that means that there is a bijection $B : V_T \to R^N$ with $B(v) = (a_{v_1}, a_{v_2}, \cdots, a_{v_N}) \in R^N$ for $v \in V_T$. We can also see that B preserves addition and scalar multiplication because for $\lambda, \mu \in R$ and $v, u \in V_T$

$$\begin{aligned} B(\lambda v + \mu u) &= B\left(\lambda \sum_{i=1}^{N} a_{v_i}\phi_i(x) + \mu \sum_{i=1}^{N} a_{u_i}\phi_i(x)\right) \\ &= B\left(\sum_{i=1}^{N} (\lambda a_{v_i} + \mu a_{u_i})\phi_i(x)\right) \\ &= (\lambda a_{v_1} + \mu a_{u_1}, \lambda a_{v_1} + \mu a_{u_1}, \cdots, \lambda a_{v_N} + \mu a_{u_N}) \\ &= \lambda(a_{v_1}, a_{v_2}, \cdots, a_{v_N}) + \mu(a_{u_1}, a_{u_2}, \cdots, a_{u_N}), \end{aligned}$$

therefore, B is an isomorphism between V_T and R^N ($V_T \cong R^N$).

2.1.2 Definition of linear space $V_T \cap H^1_{0,D}(\Omega)$

We set

$$V_T \cap H^1_{0,D}(\Omega) = \{v \in H^1_{0,D}(\Omega) : v_{|\tau} \in P_1 \ \ \forall \tau \in T \text{ and } v = 0 \text{ on } \Gamma_D\}, \tag{2.1}$$

which obviously is the intersection of the linear spaces V_T and $H^1_{0,D}(\Omega)$, thus $V_T \cap H^1_{0,D}(\Omega) \subseteq V_T$, $H^1_{0,D}(\Omega)$ and because we have an intersection of linear spaces, $V_T \cap H^1_{0,D}(\Omega)$ would also be a linear space.

Also because $V_T \cap H^1_{0,D}(\Omega) \subseteq V_T$ a linear subspace of V_T (a finite linear space), we can for every $v \in V_T \cap H^1_{0,D}(\Omega)$ use the basis of V_T and write it as $v = \sum_{i=1}^{N} a_{v_i}\phi_i$, where for every i that corresponds to a vertex on the boundary Γ_D we have $a_{v_i} = 0$. So it's easy to see that $V_T \cap H^1_{0,D}(\Omega)$ has as basis a subset of the basis of V_T, that means $\{\phi_{i_k}\}_{k=1}^{M} \subseteq \{\phi_i\}_{i=1}^{N}$ where i_k for $k = 1, \cdots, M \leq N$ match all the ϕ_{i_k} that don't correspond on Γ_D and where $N - M$ is the number of vertices that belong to Γ_D.

2.1.3 Definition of $V_{T,g,D}$

We define

$$V_{T,g,D} = \{v \in C(\overline{\Omega}) : v_{|\tau} \in P_1 \ \ \forall \tau \in T \text{ and } v = g'_D \text{ on } \Gamma_D\} \subseteq V_T, \tag{2.2}$$

a subset of V_T and not a subspace, where we take as g'_D a more linear approach to g_D, by assuming that $g'_D(x_i) = g_D(x_i)$ for each $i \in L$ (a numeration of the vertices on Γ_D) with $x_i \in \Gamma_D$ the point of a vertex at triangulation T on Γ_D.

The reason we don't take $v = g_D$ on Γ_D is because the smoothness of $v \in V_T$ on the boundary Γ_D may be different, more linear by parts, than that of g_D on Γ_D and they can't be equal. Remember that $v_{|\tau}$ belongs to P_1 for each $t \in T$ and v needs to retain that property also on the boundary Γ_D, something from which g_D is not limited and this is the reason in the end we take $v = g'_D$ on Γ_D. So keep in mind that $V_{T,g,D}$ may not be a subset of $H^1_{g,D}(\Omega)$.

Finally for each $v \in V_{T,g,D}$ we have that because $v \in V_T$ and $v = g'_D$ on Γ_D, there is $(v_1, v_2, \cdots, v_N) \in R^N$ such that $v = \sum_{i=1}^{N} v_i\phi_i(x)$ and $v_i = g'_D(x_i) = g_D(x_i)$ for each $i \in L$, where $x_i \in \Gamma_D$.

2.2 The finite element method

So for (1.24) we define our first form of finite element method such that find $u \in V_{T,g,D}$ that satisfies

$$\int_\Omega \nabla u \cdot \nabla v dx = \int_\Omega f v dx + \int_{\Gamma_N} g_N v dS \ \ \forall v \in V_T \cap H^1_{0,D}(\Omega). \tag{2.3}$$

This is equivalent of finding $(u_1, u_2, \cdots u_N) \in R^N$, such that $u = \sum_{i=1}^{N} u_i\phi_i(x)$ in which for every $i \in L$ we have $u_i = g_D(x_i)$ with $x_i \in \Gamma_D$, that satisfies

$$\sum_{i=1}^{N} u_i \int_\Omega \nabla\phi_i(x) \cdot \nabla v dx = \int_\Omega f v dx + \int_{\Gamma_N} g_N v dS \ \ \forall v \in V_T \cap H^1_{0,D}(\Omega).$$

Also because $V_T \cap H^1_{0,D}(\Omega)$ is a linear space with a basis $\{\phi_{i_k}\}_{k=1}^{M} \subseteq \{\phi_i\}_{i=1}^{N}$, it's the same to take the above equation only for its $\phi_{i_k}(x) \in V_T \cap H^1_{0,D}(\Omega)$ for $k = 1, \cdots, M$. So the problem is equivalent to find $(u_1, u_2, \cdots u_N) \in R^N$ and $u_i = g_D(x_i)\ \forall i \in L$ where $x_i \in \Gamma_D$ such that

$$\sum_{i=1}^{N} u_i \int_\Omega \nabla\phi_i \cdot \nabla\phi_{i_k} dx = \int_\Omega f\phi_{i_k} dx + \int_{\Gamma_N} g_N\phi_{i_k} dS \ \ \forall k = 1, \cdots, M.$$

To simplify things, we reorder the numbering of nodes so we can write the method as find $(u_1, u_2, \cdots u_M, u_{M+1}, \cdots u_N) \in R^N$ with $u_i = g_D(x_i)$ for $i = M+1, \cdots N$ such that

$$\sum_{i=1}^{N} u_i \int_\Omega \nabla\phi_i \cdot \nabla\phi_j dx = \int_\Omega f\phi_j dx + \int_{\Gamma_N} g_N \phi_j dS \;\; \forall j = 1, \cdots, M. \tag{2.4}$$

Now we set $a_{ji} = \int_\Omega \nabla\phi_i \cdot \nabla\phi_j dx \in R$ for $j = 1, \cdots, M$, $i = 1, \cdots, N$ and we can have the $M \times N$ matrix $A = (a_{ji})_{M\times N}$. Also we set $F_J = \int_\Omega f\phi_j dx + \int_{\Gamma_N} g_N\phi_j dS$ for $j = 1, \cdots, M$ and $F = (F_1, F_2, \cdots F_M)^T \in R^M$.

So the problem becomes find $\vec{u} = (u_1, u_2, ..., u_N)^T \in R^N$ with $u_i = g_D(x_i)$ for $i = M+1 \cdots N$ such that

$$A\vec{u} = F \text{ with } A \in R^{M\times N} \text{ and } F \in R^M. \tag{2.5}$$

We can simplify further that form by taking (2.4) and writing as

$$\sum_{i=1}^{M} u_i \int_\Omega \nabla\phi_i \cdot \nabla\phi_j dx + \sum_{i=M+1}^{N} u_i \int_\Omega \nabla\phi_i \cdot \nabla\phi_j dx = \int_\Omega f\phi_j dx + \int_{\Gamma_N} g_N\phi_j dS \;\; \forall j = 1, \cdots, M$$

or

$$\sum_{i=1}^{M} u_i \int_\Omega \nabla\phi_i \cdot \nabla\phi_j dx = \int_\Omega f\phi_j dx + \int_{\Gamma_N} g_N\phi_j dS - \sum_{i=M+1}^{N} g_D(x_i) \int_\Omega \nabla\phi_i \cdot \nabla\phi_j dx \;\; \forall j = 1, \cdots, M.$$

Finally we take a new matrix $A'_{M\times M} = (a_{ji})_{M\times M} \in R^{M\times M}$ with $a_{ji} = \int_\Omega \nabla\phi_i \cdot \nabla\phi_j dx$ for $j, i = 1, \cdots, M$, along with a vector $F' = (F'_1, F'_2, \cdots, F'_M)^T \in R^M$ where $F'_j = \int_\Omega f\phi_j dx + \int_{\Gamma_N} g_N\phi_j dS - \sum_{i=M+1}^{N} g_D(x_i) \int_\Omega \nabla\phi_i \cdot \nabla\phi_j dx$ and the problem becomes:

$$\text{find } \vec{u'} = (u_1, u_2, ..., u_M) \in R^M \text{ such that } A'\vec{u'} = F'. \tag{2.6}$$

2.2.1 Alternative form of the FEM

Now let's define the linear finite element form based on the second form of the weak problem (1.25), which is find $w \in V_T \cap H^1_{0,D}(\Omega)$ that satisfies

$$\int_\Omega \nabla w \cdot \nabla v dx = \int_\Omega f v dx - \int_\Omega \nabla G \cdot \nabla v dx + \int_{\Gamma_N} g_N v dS \;\; \forall v \in V_T \cap H^1_{0,D}(\Omega). \tag{2.7}$$

Based on what we have shown before for the linear space $V_T \cap H^1_{0,D}(\Omega)$, $w(x)$ can be written as $w(x) = \sum_{i=1}^{M} w_i\phi_i(x)$ with $\vec{w} = (w_1, w_2, \cdots, w_M) \in R^M$. So the problem becomes equivalent to, find $\vec{w} \in R^M$ such that

$$\sum_{i=1}^{M} w_i \int_\Omega \nabla\phi_i(x) \cdot \nabla v dx = \int_\Omega f v dx - \int_\Omega \nabla G \cdot \nabla v dx + \int_{\Gamma_N} g_N v dS \;\; \forall v \in V_T \cap H^1_{0,D}(\Omega).$$

Also because $V_T \cap H^1_{0,D}(\Omega)$ is a linear space with a finite basis $\{\phi_j(x)\}_{j=1,\cdots,M}$, it's the same to take the above equation only for it's $\phi_i(x) \in V_T \cap H^1_{0,D}(\Omega)$ for $i = 1, \cdots, M$. So the problem becomes equivalent to find $\vec{w} \in R^M$ such that

$$\sum_{i=1}^{M} w_i \int_\Omega \nabla\phi_i \nabla\phi_j dx = \int_\Omega f\phi_j dx - \int_\Omega \nabla G \cdot \nabla\phi_j dx + \int_{\Gamma_N} g_N \phi_j dS \text{ for } j = 1, \cdots, M.$$

Finally we take a matrix $A = (a_{ij})_{M\times M}$ with $a_{ij} = \int_\Omega \nabla\phi_i \nabla\phi_j dx$, along with a vector $F = (F_1, F_2, \cdots, F_M)^T \in R^M$ where $F_i = \int_\Omega f\phi_i dx - \int_\Omega \nabla G \cdot \nabla\phi_i dx + \int_{\Gamma_N} g_N \phi_i dS$ for $i = 1, \cdots, M$ and the problem becomes:

$$\text{find } \vec{w} \in R^M \text{ such that } A\vec{w} = F, \tag{2.8}$$

which is not identical to the first form of the finite element method (2.6).

2.2.2 Equivalence of the two forms of FEM

Now we show that (2.6) and (2.8) are equivalents equations. If for the second form (2.8) we take the same numbering for the basis functions ϕ_i as the first (2.6), then it is obvious that $A = A'$ and the space R^M, where we search for the solution u, it is the same for both. So the only difference we can spot are to the vectors F, F', therefore we just need to prove that $F = F'$.

For F on (2.8) we see that the formula has a G, which as we defined before is a function, such that $G \in H^1(\Omega)$ and $G = g$ on Γ_D. In (2.6) we used the set $V_{T,g,D}$; here we do correspondingly the same thing for G, that is we are less strict about the property $G = g_D$ on Γ_D i.e., we seek an approximation of g_D. So we take as G a continuous function in Ω and linear per triangle $t \in T$ with $G(x_i) = 0$ for every vertex x_i on Ω except those $x_i \in \Gamma_D$ for which we take $G(x_i) = g_D(x_i)$.

So we take a $G \in V_{T,g,D}$ and can be written as $G(x) = \sum_{i=1}^{N} k_i\phi_i(x)$ with $k_i = 0$ for $i = 1, \cdots, M$ and $k_i = g_D(x_i) = G(x_i)$ for $i = M+1, \cdots, N$ based on the numeration of the vertices we defined before. So from (2.8) we have

$$F_j = \int_\Omega f\phi_j dx - \int_\Omega \nabla G \cdot \nabla\phi_j dx + \int_{\Gamma_N} g_N \phi_j dS$$

and because $G(x) = \sum_{i=1}^{N} k_i\phi_i(x) = \sum_{i=M+1}^{N} k_i\phi_i(x)$, it becomes

$$\int_\Omega f\phi_j dx - \int_\Omega \nabla\left(\sum_{i=M+1}^{N} k_i\phi_i\right) \cdot \nabla\phi_j dx + \int_{\Gamma_N} g_N \phi_j dS,$$

that means

$$\int_\Omega f\phi_j dx - \sum_{i=M+1}^{N} k_i \int_\Omega \nabla\phi_i \cdot \nabla\phi_j dx + \int_{\Gamma_N} g_N \phi_j dS,$$

which is equivalent to F'_j defined at (2.6) and so $F = F'$. So the equivalent finite element methods (2.4), (2.5) (2.6) becomes equivalent with the (2.7), (2.8).

2.3 Existence and uniqueness of the solution at the finite element method

The reason we showed the equivalence between the two forms of our weak problem (1.24) and (1.25) as well as the equivalence of their respective finite element methods (2.5) and (2.7) or (2.8), is because we use (2.5) for the calculation of the solution on our MATLAB program, while we use (1.25) and the (2.7) or (2.8) for the proofs of our theory.

So now we see that our finite element method, no matter its form, has a unique solution. This is trivial, because $V_T \cap H^1_{0,D}(\Omega)$, is a finite dimensional linear subspace of $H^1_{0,D}(\Omega)$ with the same norm derived from an inner product as $H^1_{0,D}(\Omega)$, that means is a Hilbert space and because Lax - Milgram Theorem (Appendix 5.2) holds for the space $H^1_{0,D}(\Omega)$ on (1.25), it also holds for the subspace. So there is a unique solution for the finite element form (2.7) and so for every other equivalent form.

Chapter 3

A-posteriori error bounds

Now we estimate the global error between the weak solution and its approximation for a triangulation T_n [1], the finite element solution. We do that by finding an a-posteriori error bound to the global error.

The weak problem reads: find $u \in H^1_{g,D}(\Omega)$, such that

$$\int_\Omega \nabla u \cdot \nabla v dx = \int_\Omega f v dx + \int_{\Gamma_N} g_N v dS \;\; \forall v \in H^1_{0,D}(\Omega), \tag{3.1}$$

while its finite element approximation problem is find $u_n \in V_{T_n,g,D}$ that satisfies

$$\int_\Omega \nabla u_n \cdot \nabla v_n dx = \int_\Omega f v_n dx + \int_{\Gamma_N} g_N v_n dS \;\; \forall v_n \in V_{T_n} \cap H^1_{0,D}(\Omega); \tag{3.2}$$

here for simplicity we take $u_n = g_D$ on Γ_D instead of g'_D on Γ_D. If $g_D \neq g'_D$ we can use the stability of the PDE.

So let's assume we have $u \in H^1_{g,D}(\Omega)$ that satisfies (3.1) and $u_n \in V_{T_n,g,D}$ that satisfies (3.2). Then

$$\|\nabla(u - u_n)\|^2_{L^2(\Omega)} = \int_\Omega \nabla u \cdot \nabla(u - u_n) dx - \int_\Omega \nabla u_n \cdot \nabla(u - u_n) dx,$$

from (3.1). Since $u - u_n \in H^1_{0,D}(\Omega)$, we have

$$\|\nabla(u - u_n)\|^2_{L^2(\Omega)} = \int_\Omega f(u - u_n) dx + \int_{\Gamma_N} g_N(u - u_n) dS - \int_\Omega \nabla u_n \cdot \nabla(u - u_n) dx. \tag{3.3}$$

Also if we take an arbitrary $v_n \in V_{T_n} \cap H^1_{0,D}(\Omega)$, from (3.2) we have

$$\int_\Omega \nabla u_n \cdot \nabla v_n dx - \int_\Omega f v_n dx - \int_{\Gamma_N} g_N v_n dS = 0,$$

[1] Actually T_n belongs to series of triangulations in Ω and for each step n there is an approximate solution u_n, for now we just pick one of those n to find the a-posteriori error bound to the global error.

and, so, (3.3) becomes

$$\|\nabla(u-u_n)\|^2_{L^2(\Omega)} = \int_\Omega f(u-u_n)dx + \int_{\Gamma_N} g_N(u-u_n)dS - \int_\Omega \nabla u_n \cdot \nabla(u-u_n)dx + \int_\Omega \nabla u_n \cdot \nabla v_n dx - \int_\Omega f v_n dx - \int_{\Gamma_N} g_N v_n dS,$$

which means

$$\|\nabla(u-u_n)\|^2_{L^2(\Omega)} = \int_\Omega f(u-u_n-v_n)dx + \int_{\Gamma_N} g_N(u-u_n-v_n)dS - \int_\Omega \nabla u_n \cdot \nabla(u-u_n-v_n)dx.$$

Because on the n^{th} step we take the triangulation T_n of Ω, we can divide Ω to triangles t and have

$$\|\nabla(u-u_n)\|^2_{L^2(\Omega)} = \sum_{t\in T}\left(\int_t f(u-u_n-v_n)dx\right) + \int_{\Gamma_N} g_N(u-u_n-v_n)dS - \sum_{t\in T}\left(\int_t \nabla u_n \cdot \nabla(u-u_n-v_n)dx\right). \tag{3.4}$$

Now, because $u_n \in V_{T_n,g,D} \subseteq V_{T_n}$, which means that u_n is linear in each $t \in T_n$, we have that u_n is two times differentiable on t, so we can take integration by parts on t and have

$$\int_t \nabla u_n \cdot \nabla(u-u_n-v_n)dx = \int_t \nabla\cdot((u-u_n-v_n)\nabla u_n)dx - \int_t \Delta u_n(u-u_n-v_n)dx,$$

which transforms (3.4) into

$$\|\nabla(u-u_n)\|^2_{L^2(\Omega)} = \sum_{t\in T}\left(\int_t f(u-u_n-v_n)dx\right) + \int_{\Gamma_N} g_N(u-u_n-v_n)dS - \sum_{t\in T}\left(\int_t \nabla\cdot((u-u_n-v_n)\nabla u_n)dx - \int_t \Delta u_n(u-u_n-v_n)dx\right),$$

also from Gauss - Green Theorem (Appendix 5.1). If we denote by ∂t the edges of the triangle t, we have

$$\|\nabla(u-u_n)\|^2_{L^2(\Omega)} = \sum_{t\in T}\left(\int_t f(u-u_n-v_n)dx\right) + \int_{\Gamma_N} g_N(u-u_n-v_n)dS - \sum_{t\in T}\left(\int_{\partial t}(u-u_n-v_n)\nabla u_n\cdot\vec{n}dS - \int_t \Delta u_n(u-u_n-v_n)dx\right).$$

Simplifying further the equation and if we take as e_1, e_2, e_3 the edges of triangle t with $\partial t = e_1 \cup e_2 \cup e_3$, we have

$$\|\nabla(u-u_n)\|^2_{L^2(\Omega)} = \sum_{t\in T}\left(\int_t (f+\Delta u_n)(u-u_n-v_n)dx\right) + \int_{\Gamma_N} g_N(u-u_n-v_n)dS - \sum_{t\in T}\left(\sum_{i=1}^{3}\int_{e_i}(u-u_n-v_n)\nabla u_n\cdot\vec{n}dS\right).$$

Now let's take as $S(T)$ the set of edges for triangulation T of Ω and as $S(T)^o = S(T)\backslash(\Gamma_D \cup \Gamma_N)$ the edges of $S(T)$ without those that belongs to the boundary $\partial\Omega = \Gamma_D \cup \Gamma_N$, then for each edge $e \in S(T)^o$ based on the note here [2] we can take the jump of ∇u_n on e as $[\nabla u_n]_e = \nabla u_{n|t_{|e}} \cdot \vec{n_{t_e}} + \nabla u_{n|t'_{|e}} \cdot \vec{n_{t'_e}}$, where $e = \partial t \cap \partial t'$ and then we can have

$$\|\nabla(u-u_n)\|^2_{L^2(\Omega)} = \sum_{t\in T}\left(\int_t (f+\Delta u_n)(u-u_n-v_n)dx\right) + \int_{\Gamma_N} g_N(u-u_n-v_n)dS$$
$$- \sum_{e\in S(T)^o}\int_e [\nabla u_n]_e(u-u_n-v_n)dS - \sum_{e\in\Gamma_D\cup\Gamma_N}\int_e (u-u_n-v_n)\nabla u_n\cdot\vec{n}dS,$$

also because $u-u_n, v_n \in H^1_{0,D}(\Omega)$ we have that $u-u_n-v_n \in H^1_{0,D}(\Omega)$, which means that $u-u_n-v_n = 0$ on Γ_D and so we have

$$\|\nabla(u-u_n)\|^2_{L^2(\Omega)} = \sum_{t\in T}\left(\int_t (f+\Delta u_n)(u-u_n-v_n)dx\right) + \int_{\Gamma_N} g_N(u-u_n-v_n)dS$$
$$- \sum_{e\in S(T)^o}\int_e [\nabla u_n]_e(u-u_n-v_n)dS - \sum_{e\in\Gamma_N}\int_e (u-u_n-v_n)\nabla u_n\cdot\vec{n}dS,$$

simplifying further, we can have

$$\|\nabla(u-u_n)\|^2_{L^2(\Omega)} = \sum_{t\in T}\left(\int_t (f+\Delta u_n)(u-u_n-v_n)dx\right) + \sum_{e\in\Gamma_N}\int_e g_N(u-u_n-v_n)dS$$
$$- \sum_{e\in S(T)^o}\int_e [\nabla u_n]_e(u-u_n-v_n)dS - \sum_{e\in\Gamma_N}\int_e (u-u_n-v_n)\nabla u_n\cdot\vec{n}dS$$

and so

$$\|\nabla(u-u_n)\|^2_{L^2(\Omega)} = \sum_{t\in T}\left(\int_t (f+\Delta u_n)(u-u_n-v_n)dx\right) - \sum_{e\in\Gamma_N}\int_e (u-u_n-v_n)(\nabla u_n\cdot\vec{n}-g_N)dS$$
$$- \sum_{e\in S(T)^o}\int_e [\nabla u_n]_e(u-u_n-v_n)dS.$$

Also we set as $L(u_n,e) = [\nabla u_n]_e \ \forall e \in S(T)^o$ and $L(u_n,e) = (\nabla u_n\cdot\vec{n}-g_N)_{|e} \ \forall e\in\Gamma_N$, and we have

$$\|\nabla(u-u_n)\|^2_{L^2(\Omega)} = \sum_{t\in T}\left(\int_t (f+\Delta u_n)(u-u_n-v_n)dx\right) - \sum_{e\in S(T)\backslash\Gamma_D}\int_e L(u_n,e)(u-u_n-v_n)dS.$$

[2] If we take a triangulation T, then we can find two triangles $t, t' \in T$ with a common edge $e = \partial t \cap \partial t'$, where ∂t and $\partial t'$ are the edges of the triangles t and t' respectively. On them we can define $\vec{n_t}$ and $\vec{n_{t'}}$, which are the outward pointing unit normal vector fields for the edges of t and t' respectively. Now for a function ϕ defined on T with the correct weak differentiability at each triangle we can define

$$[\nabla\phi]_e = \nabla\phi_{|t_{|e}}\cdot\vec{n_{t_e}} + \nabla\phi_{|t'_{|e}}\cdot\vec{n_{t'_e}},$$

where if $\nabla\phi$ continuous in the area around e, because also $\vec{n_t} = -\vec{n_{t'}}$ we have $[\nabla\phi]_e = 0$ on e, while if $\nabla\phi$ not continuous in the area around e, then $[\nabla\phi]_e \neq 0$ on e and we call it jump of $\nabla\phi$ on e.

Now from Cauchy - Schwarz (Appendix 5.4) on $L^2(t)$, we have

$$\|\nabla(u-u_n)\|^2_{L^2(\Omega)} \leq \sum_{t\in T}\left(\left(\int_t (f+\Delta u_n)^2 dx\right)^{1/2}\left(\int_t (u-u_n-v_n)^2 dx\right)^{1/2}\right) + |\sum_{e\in S(T)\backslash\Gamma_D}\int_e L(u_n,e)(u-u_n-v_n)dS|,$$

which means

$$\|\nabla(u-u_n)\|^2_{L^2(\Omega)} \leq \sum_{t\in T}\|f+\Delta u_n\|_{L^2(t)}\|u-u_n-v_n\|_{L^2(t)} + |\sum_{e\in S(T)\backslash\Gamma_D}\int_e L(u_n,e)(u-u_n-v_n)dS|,$$

where from Cauchy - Schwarz (Appendix 5.4) on $L^2(e)$, we have

$$\|\nabla(u-u_n)\|^2_{L^2(\Omega)} \leq \sum_{t\in T}\|f+\Delta u_n\|_{L^2(t)}\|u-u_n-v_n\|_{L^2(t)} + \sum_{e\in S(T)\backslash\Gamma_D}\|L(u_n,e)\|_{L^2(e)}\|u-u_n-v_n\|_{L^2(e)}.$$

Also if we take as h_t the length of the largest edge on t and as h_e the length of edge e, we have

$$\|\nabla(u-u_n)\|^2_{L^2(\Omega)} \leq \sum_{t\in T} h_t\|f+\Delta u_n\|_{L^2(t)}h_t^{-1}\|u-u_n-v_n\|_{L^2(t)} + \sum_{e\in S(T)\backslash\Gamma_D} h_e^{1/2}\|L(u_n,e)\|_{L^2(e)}h_e^{-1/2}\|u-u_n-v_n\|_{L^2(e)}$$

Now from Cauchy - Schwarz (Appendix 5.4) on R^k, where k we mean in one case the number of triangles on T and on the other the number of edges on $S(T)\backslash\Gamma_D$, we have

$$\|\nabla(u-u_n)\|^2_{L^2(\Omega)} \leq \overbrace{\left(\sum_{t\in T} h_t^2\|f+\Delta u_n\|^2_{L^2(t)}\right)^{1/2}}^{A_1}\overbrace{\left(\sum_{t\in T} h_t^{-2}\|u-u_n-v_n\|^2_{L^2(t)}\right)^{1/2}}^{B_1} + \overbrace{\left(\sum_{e\in S(T)\backslash\Gamma_D} h_e\|L(u_n,e)\|^2_{L^2(e)}\right)^{1/2}}^{A_2}\overbrace{\left(\sum_{e\in S(T)\backslash\Gamma_D} h_e^{-1}\|u-u_n-v_n\|^2_{L^2(e)}\right)^{1/2}}^{B_2},$$

and because $A_1B_1 + A_2B_2 \leq (A_1^2+A_2^2)^{1/2}(B_1^2+B_2^2)^{1/2}$, we have

$$\|\nabla(u-u_n)\|^2_{L^2(\Omega)} \leq \left(\sum_{t\in T} h_t^2\|f+\Delta u_n\|^2_{L^2(t)} + \sum_{e\in S(T)\backslash\Gamma_D} h_e\|L(u_n,e)\|^2_{L^2(e)}\right)^{1/2} \left(\sum_{t\in T} h_t^{-2}\|u-u_n-v_n\|^2_{L^2(t)} + \sum_{e\in S(T)\backslash\Gamma_D} h_e^{-1}\|u-u_n-v_n\|^2_{L^2(e)}\right)^{1/2},$$

which becomes

$$\|\nabla(u-u_n)\|^2_{L^2(\Omega)} \leq \left(\sum_{t\in T} h_t^2 \|f+\Delta u_n\|^2_{L^2(t)} + \sum_{e\in S(T)\setminus\Gamma_D} h_e \|L(u_n,e)\|^2_{L^2(e)}\right)^{1/2} \left(\sum_{t\in T} h_t^{-2}\|u-u_n-v_n\|^2_{L^2(t)} + \sum_{e\in S(T)} h_e^{-1}\|u-u_n-v_n\|^2_{L^2(e)}\right)^{1/2}, \quad (3.5)$$

where $u-u_n \in H^1(\Omega)$ and $v_n \in V_T \cap H^1_{0,D}(\Omega) \subseteq H^1(\Omega)$. where $u-u_n \in H^1(\Omega)$ and $v_n \in V_T \cap H^1_{0,D}(\Omega) \subseteq H^1(\Omega)$. The above inequality is true for any $v_n \in V_T \cap H^1_{0,D}(\Omega)$, so from Existence of interpolator of Clément Theorem (Appendix 5.8) for $v = u-u_n \in H^1(\Omega)$ we can find a $v_n \in V_T \cap H^1_{0,D}(\Omega)$ (the H_n of Appendix 5.8) and a $c_1 \geq 0$, such that if we take the above inequality and apply the theorem, we have

$$\|\nabla(u-u_n)\|^2_{L_2(\Omega)} \leq \left(\sum_{t\in T} h_t^2 \|f+\Delta u_n\|^2_{L^2(t)} + \sum_{e\in S(T)\setminus\Gamma_D} h_e \|L(u_n,e)\|^2_{L^2(e)}\right) {c_1}^{1/2}\|\nabla(u-u_n)\|_{L_2(\Omega)},$$

which means

$$\|\nabla(u-u_n)\|_{L_2(\Omega)} \leq {c_1}^{1/2}\left(\sum_{t\in T} h_t^2 \|f+\Delta u_n\|^2_{L^2(t)} + \sum_{e\in S(T)\setminus\Gamma_D} h_e \|L(u_n,e)\|^2_{L^2(e)}\right)^{1/2}.$$

Also, because $u-u_n \in H^1_{0,D}(\Omega)$, from Poincaré - Friedrichs inequality (Appendix 5.3) we have that, there is $c_2 \geq 0$ such as

$$\|u-u_n\|_{H^1_{0,D}(\Omega)} \leq c_2 {c_1}^{1/2}\left(\sum_{t\in T} h_t^2 \|f+\Delta u_n\|^2_{L^2(t)} + \sum_{e\in S(T)\setminus\Gamma_D} h_e \|L(u_n,e)\|^2_{L^2(e)}\right)^{1/2}$$

and by setting as $e(u_n,f,\Omega) = \left(\sum_{t\in T} h_t^2 \|f+\Delta u_n\|^2_{L^2(t)} + \sum_{e\in S(T)\setminus\Gamma_D} h_e \|L(u_n,e)\|^2_{L^2(e)}\right)^{1/2}$ and $c = c_2 {c_1}^{1/2} \geq 0$, we can write the inequality as

$$\|u-u_n\|_{H^1_{0,D}(\Omega)} \leq c e(u_n,f,\Omega). \quad (3.6)$$

Now let's take a closer look at $e(u_n,f,\Omega)$; we have that

$$e^2(u_n,f,\Omega) = \sum_{t\in T} h_t^2 \|f+\Delta u_n\|^2_{L^2(t)} + \sum_{e\in S(T)\setminus\Gamma_D} h_e \|L(u_n,e)\|^2_{L^2(e)}$$

and we want to bound that quantity further. To do that we try to extend the function $L(u_n,e)$ on Γ_D, by setting $L(u_n,e) = [\nabla u_n]_e$ for $e \in \Gamma_D$. This seems problematic, because for the

edges e that belongs to the boundary Γ_D, the u_n is not defined on the side of the edge that's outside of T and that means that the jump of ∇u_n on e, as described here [13], is also not defined. So we extend the notion of $[\nabla u_n]_e$ on Γ_D and from now on for those edges, when we use $[\nabla u_n]_e = \nabla u_{n|t_{|e}} \cdot \vec{n_{t_e}} + \nabla u_{n|t'_{|e}} \cdot \vec{n_{t'_e}}$, we will mean that either $\nabla u_{n|t_{|e}}$ or $\nabla u_{n|t'_{|e}}$ is zero on e. So now we can have the following inequality

$$\begin{aligned} e^2(u_n, f, \Omega) &\leq \sum_{t \in T} h_t^2 \|f + \Delta u_n\|^2_{L^2(t)} + \sum_{t \in T} \sum_{e \in dt} h_e \|L(u_n, e)\|^2_{L^2(e)} \\ &= \sum_{t \in T} \left(h_t^2 \|f + \Delta u_n\|^2_{L^2(t)} + \sum_{e \in dt} h_e \|L(u_n, e)\|^2_{L^2(e)} \right) \end{aligned}$$

and by taking as

$$e_{tr}(u_n, f, t) = \left(h_t^2 \|f + \Delta u_n\|^2_{L^2(t)} + \sum_{e \in dt} h_e \|L(u_n, e)\|^2_{L^2(e)} \right)^{1/2}, \tag{3.7}$$

we can write the inequality as

$$e^2(u_n, f, \Omega) \leq \sum_{t \in T} e^2_{tr}(u_n, f, t).$$

Finally from (3.6) we have the following bound

$$\|u - u_n\|_{H^1_{0,D}(\Omega)} \leq c^{1/2} \left(\sum_{t \in T} e^2_{tr}(u_n, f, t) \right)^{1/2}. \tag{3.8}$$

So in (3.8) we have defined the a-posteriori error bound of the global error. We also call $e_{tr}(u_n, f, t)$ defined in (3.7) as the error estimator on triangle t, or error bound per triangle t and so we can see that the a-posteriori error bound is a sum that takes into account those error bounds per triangle. For a further simplification of (3.8) in a more calculable form, see the Appendix 5.9 on how to calculate $e_{tr}(u_n, f, t)$.

Chapter 4

Adaptive Finite Element Method

4.1 Introduction

We can now discuss an Adaptive Finite Element Method (AFEM).

A regular Finite Element Method (or FEM) is used in order to find an approximation solution $\overline{u}$ to our weak problem, as close as we want to the real weak solution u. To do that we take a triangulation T_n and calculate on it the approximation solution u_n and the a-posteriori error bound e_n it has with the real solution u, then if we want a smaller error we refine *all* the triangles of the triangulation T_n to create a new one T_{n+1} for which we calculate again the u_{n+1} and e_{n+1}. We repeat that until we reach an n with the desired error bound (e_n), then for that n we would have $\overline{u} = u_n$.

The problem here with the regular FEM is that after some refinement we will have a very large number N of triangles and this algorithm can become very slow. The key issue, however, is that the rate of convergence of the regular FEM depends on the global regularity of the exact solution of the problem. Therefore, in the presence of corner singularities, the convergence of regular FEM is slower than the rate of convergence for smooth exact solutions.

This is what AFEM is coming to solve, it's the same logic as before but to avoid slowness, at each step instead of refine all the triangles, we will choose **adaptively** which of them needs refinement. So we do that in hope to find the error bound we seek by creating a smaller set of triangles than the Regular FEM.

The criteria by which we choose those triangles are based to an error estimator for each triangle, called error bound per triangle (which we have defined before as e_{tr} for a triangle t). This tell us how much each triangle contributes to the global error bound (a - posteriori error bound). The triangles with higher e_{tr} are those that we need to refine. This will allow us to increase the density of the triangulation in order to gradually decrease the error, while also refine only those triangles that can have the most impact and not all of them. More for the criteria later.

Next we take a closer look at the blueprints of Regular and Adaptive FEM algorithms.

4.2 Regular FEM Algorithm

INITIAL MESH:
create the initial triangulation T_1

↓

SOLVE PROBLEM:
solve the finite element problem for triangulation T_1

↓

ESTIMATE A-POSTERIORI ERROR BOUND:
fill the variable ***error*** *with the a-posteriori error bound estimation of triangulation* T_1

↓

WHILE (**error** > **errorBoundary**)

REFINE MESH:
for ***ALL*** *triangles, refine the mesh of triangulation* T_n *to* T_{n+1}

↓

SOLVE PROBLEM:
solve the finite element problem for triangulation T_{n+1}

↓

ESTIMATE A-POSTERIORI ERROR BOUND:
fill the variable ***error*** *with the a-posteriori error bound estimation of triangulation* T_{n+1}

END_WHILE

4.3 Adaptive FEM Algorithm

INITIAL MESH:
create the initial triangulation T_1

↓

SOLVE PROBLEM:
solve the finite element problem for triangulation T_1

↓

ESTIMATE A-POSTERIORI ERROR BOUND:
fill the variable ***error*** *with the a-posteriori error bound estimation of triangulation* T_1

↓

WHILE (**error** > **errorBoundary**)

MARK TRIANGLES:
we mark the triangles that need refinement based on a CRITERION

↓

REFINE MESH:
for the ***MARKED*** *triangles, refine the mesh of triangulation* T_n *to* T_{n+1}

↓

SOLVE PROBLEM:
solve the finite element problem for triangulation T_{n+1}

↓

ESTIMATE A-POSTERIORI ERROR BOUND:
fill the variable ***error*** *with the a-posteriori error bound estimation of triangulation* T_{n+1}

END_WHILE

4.4 Notes for the Regular and Adaptive FEM Algorithms

4.4.1 The a-posteriori error parameter: error

Based on (3.8) we take as the parameter ***error*** the a-posteriori error bound defined as

$$\boldsymbol{error} = \left(\sum_{t \in T} e_{tr}^2(u_n, f, t) \right)^{1/2}. \tag{4.1}$$

4.4.2 Loop Condition

The condition $\boldsymbol{error} > \boldsymbol{errorBoundary}$, for both Regular and Adaptive FEM algorithms, is meant to run the WHILE loop until it breaks, that is when the a-posteriori error bound becomes less or equal to a minimum boundary, defined as ***errorBoundary***.

When that happens, based on (3.8), this would mean that we have achieved the following error,

$$\|u - u_n\|_{H^1_{0,D}(\Omega)} \leq c \cdot \boldsymbol{error} \leq c \cdot \boldsymbol{errorBoundary},$$

where $c > 0$ some constant.

So the ***errorBoundary*** parameter is an input argument to our program in order to specify, how close to the actual solution u we want to find the approximation solution u_n.

As we would show below, at section 4.5, for both Regular and Adaptive FEM Algorithms the a-posteriori error bound (***error***) converges to zero as $n \to \infty$, so eventually they would meet the exit condition of the ***errorBoundary*** and the program would terminate.

4.4.3 Criterion of Adaptive FEM Algorithm

As we told before the CRITERION by which we mark what triangles to refine, is what will give the adaptability to our algorithm. Actually we present two ways to do that, with the second giving us even more speed at computation than the first:

1st case: At the beginning our program takes a constant parameter $0 <$ **errorPercentage** ≤ 1 as input. At each iteration of our loop, it sorts the error estimators per triangle in our mesh, from largest to smallest. We symbolize error estimators per triangle as e_t for each $t \in T$, where t is a triangle and T the triangulation set (mesh). Finally we take as $T_H \subseteq T$, the minimal set of triangles t with the largest e_t that satisfies the following,

$$\sqrt{\sum_{t \in T_H} e_t^2} \geq errorPercentage \cdot error, \tag{4.2}$$

which means

$$\sum_{t \in T_H} e_t^2 \geq errorPercentage^2 \cdot error^2. \tag{4.3}$$

This T_H is the set of triangles we mark for refinement. We call this marking procedure as bulk-chasing marking strategy and was introduced by Dörfler.

2nd case: This is similar to first case but now instead of **errorPercentage** be a constant, it's a variable that adaptively changes at each iteration. Also **errorPercentage** has a lower bound it can reach, defined as the constant **errorPercentageBoundary**. Below is how we obtain it at each iteration:

$$errorPercentage = \frac{(error - errorBoundary)}{error} \tag{4.4}$$

$$\begin{aligned} &if(errorPercentage < errorPercentageBoundary) \\ &\quad errorPercentage = errorPercentageBoundary \\ &end \end{aligned} \tag{4.5}$$

So the CRITERION of Adaptive FEM is dependent on the marking parameter $errorPercentage$ and it is either constant or it changes per iteration.

4.4.4 Behavior of Regular and Adaptive FEM

As we said before, the main difference between those algorithms is how they select which triangles to refine at each step. The Regular FEM selects all the triangles in the mesh, while Adaptive FEM selects only a proportion of them. As a result those properties can be translated to the following behaviors for the two algorithms.

The Regular FEM algorithm as it runs, creates a uniform mesh on the domain, but the error estimators per triangle are not necessary even.

On the other hand Adaptive FEM algorithm as it runs, may create regions in the mesh with different density between them, but the error estimators per triangle tends to become even.

4.5 Analysis of the convergence of Regular and Adaptive FEM

Now let's take into consideration the above algorithms, for which at each step they create a triangulation T_n and calculate for that T_n the finite element approximation solution u_n of the original problem. Also at each step, the created triangulation T_n is denser than the previous triangulation T_{n-1}, based on a criterion, this means that the criterion will determine which part of the domain will be further triangulated. For Regular FEM the criterion is the uniform triangulation of the whole domain (that is the selection of All triangles for refinement), while for Adaptive FEM the criterion is as described at subsection 4.4.3.

4.5.1 Convergence of regular FEM to the weak solution

For regular FEM, we apply a simple criterion, that at each step the triangulation T_n becomes uniformly denser on the whole domain and we want to prove that u_n converges to u when

$n \longrightarrow +\infty$, where u_n is the solution of the finite element problem (2.7) on T_n, for the n^{th} triangulation step, while u is the solution of (1.25).

For quick reference we setup first some symbols and formulas. For the spaces we use the following symbols $V = H^1_{0,D}(\Omega)$ and $V_n = V_{T_n} \cap H^1_{0,D}(\Omega)$. We can see that $V_n \subseteq V$, so now we have the following sequence of subspaces $\{V_n\}_{n=1}^{\infty}$ of V. Also, we recall the following formulas

$$a(u,v) = \int_\Omega \nabla u \cdot \nabla v dx \tag{4.6}$$

and

$$l(v) = \int_\Omega f v dx - \int_\Omega \nabla G \cdot \nabla v dx + \int_{\Gamma_N} g_N v dS, \tag{4.7}$$

for $u, v \in V$ or V_n.

From the way we defined the triangulation here (at each step T_{n+1} to be uniformly denser on the whole domain than T_n), we have the property that for each $v \in V$ there is a sequence $\{\phi_n\}_{n=1}^{\infty}$ with $\phi_n \in V_n$, such as $lim_{n\to\infty}\|v - \phi_n\|_V = 0$, using standard interpolation estimates.

Theorem 4.5.1. *If for each $v \in V$ there is a sequence $\{\phi_n\}_{n=1}^{\infty}$ with $\phi_n \in V_n$, such as $lim_{n\to\infty}\|v-\phi_n\|_V = 0$, then for the sequence $\{u_n\}_{n=1}^{\infty}$, with $u_n \in V_n$, we have that $lim_{n\to\infty}\|u_n - u\|_V = 0$.*

Proof. From the property (a) of Lax - Milgram Theorem (Appendix 5.2), which hold for $a(\cdot,\cdot)$ on V and, since $u_n \in V_n \subseteq V \ \ \forall n \in N$ and $u - u_n \in V$, we have that there exist $c_0 > 0$ such that

$$\|u - u_n\|_V^2 \leq {c_0}^{-1} a(u - u_n, u - u_n),$$

while from the definition of the theorem, for $u \in V$ we can take a sequence $\{\phi_n\}_{n=1}^{\infty}$ with $\phi_n \in V_n$, such as $lim_{n\to\infty}\|v - \phi_n\|_V = 0$ and using the Galerkin's orthogonality (Appendix 5.7) we can have

$$\|u - u_n\|_V^2 \leq {c_0}^{-1} a(u - u_n, u - u_n) = {c_0}^{-1} a(u - u_n, u - \phi_n).$$

Now from the property (b) of Lax - Milgram Theorem (Appendix 5.2) we have

$$\|u - u_n\|_V^2 \leq {c_0}^{-1} a(u - u_n, u - \phi_n) \leq {c_0}^{-1} c_1 \|u - u_n\|_V \|u - \phi_n\|_V,$$

so we have

$$\|u - u_n\|_V \leq {c_0}^{-1} c_1 \|u - \phi_n\|_V$$

and due to $lim_{n\to\infty}\|v - \phi_n\|_V = 0$, we also get $lim_{n\to\infty}\|u - u_n\|_V = 0$, which proves our theorem. □

4.5.2 Convergence of Adaptive FEM to the weak solution

Now for Adaptive FEM algorithm we have a different mesh modification criterion, which is the key to prove its convergence to the weak solution.

Before that let's remind the weak problem and it's approximation at the n^{th} step we use here, with their spaces, as well as some properties.

We use the first form of our weak problem (1.24), which is find $u \in H^1_{g,D}(\Omega)$ such as

$$a(u,v) = l(v) \text{ for every } v \in H^1_{0,D}(\Omega), \tag{4.8}$$

with it's approximation problem (2.6) at n^{th} step be, find $u_n \in V_{T_n,g,D}$ such as

$$a(u_n,v) = l(v) \text{ for every } v \in V_{T_n} \cap H^1_{0,D}(\Omega), \tag{4.9}$$

where

$$a(u,v) = \int_\Omega \nabla u \cdot \nabla v dx \tag{4.10}$$

and

$$l(v) = \int_\Omega f v dx + \int_{\Gamma_N} g_N v dS. \tag{4.11}$$

As we said before, the spaces $H^1_{g,D}(\Omega)$ and $V_{T_n,g,D}$ are not linear, but subsets of the linear space $H^1(\Omega)$, on the other hand, their subtraction $u - u_n \in H^1_{0,D}(\Omega) \subset H^1(\Omega)$ belongs to the linear space $H^1_{0,D}(\Omega)$ and so to prove that u_n converges to u, we want to show that

$$\lim_{n\to\infty} \|u - u_n\|_{H^1_{0,D}(\Omega)} = 0. \tag{4.12}$$

Also from Poincaré - Friedrichs inequality (Appendix 5.3), because $u - u_n \in H^1_{0,D}(\Omega)$, there is $c_* \geq 0$ such as

$$\int_\Omega |u(x) - u_n(x)|^2 dx \leq c_* \|\nabla(u - u_n)\|^2_{L^2(\Omega)}$$

and so it would be enough to prove that

$$\lim_{n\to\infty} \|\nabla(u - u_n)\|_{L^2(\Omega)} = 0. \tag{4.13}$$

Also a note here is that

$$\|\nabla(u - u_n)\|_{L^2(\Omega)} = [a(u - u_n, u - u_n)]^{1/2}.$$

Returning to the criterion now, we see that at subsection 4.4.3, we described two cases of Adaptive FEM with their respective criteria, but we can use a general criterion to describe both, which at the n^{th} step is

$$e_n(T_{M_n}) \geq \theta_n \cdot e_n, \tag{4.14}$$

where $0 < \theta_n \leq 1$ is the marking parameter ($errorPercentage$) of 4.4.3, $T_{M_n} \subseteq T_n$ is the set of triangles we mark for refinement on T_n and

$$e_n = e(u_n, T_n) = \left(\sum_{t \in T_n} e^2_{tr}(u_n, f, t) \right)^{1/2}, \tag{4.15}$$

$$e_n(T_{M_n}) = e(u_n, T_{M_n}) = \left(\sum_{t \in T_{M_n}} e^2_{tr}(u_n, f, t) \right)^{1/2}. \tag{4.16}$$

As we can see e_n is the error bound from (3.8), where

$$\|\nabla(u-u_n)\|_{L^2(\Omega)} \leq \|u-u_n\|_{H^1_{0,D}(\Omega)} \leq c^{1/2}\left(\sum_{t\in T_n} e^2_{tr}(u_n,f,t)\right)^{1/2}. \tag{4.17}$$

So criterion (4.14) is the embedded property of Adaptive FEM that describes it and what makes the Adaptive, different from regular FEM, so it will be the key property to our proof of convergence.

Now before we proceed with the proof of convergence we show some helpful lemmas. For a triangulation of Ω defined as T_h, we will use the following symbols for the spaces, $V = H^1_{0,D}(\Omega)$ and $V_h = T_h \cap H^1_{0,D}(\Omega)$, where as we can see $V_h \subset V$.

Lemma 4.5.2. *(Orthogonality property)*

If T_h is a local refinement of T_H, such that $V_H \subseteq V_h$ and we have $u \in H^1_{g,D}(\Omega)$ the weak solution of (4.8) and $u_h \in V_{T_h,g,D}$, $u_H \in V_{T_H,g,D}$ the solutions of the finite element problems described for $V_{T_h,g,D}$ and $V_{T_H,g,D}$ respective in (4.9), then we have the following

$$\|\nabla(u-u_H)\|^2_{L^2(\Omega)} = \|\nabla(u-u_h)\|^2_{L^2(\Omega)} + \|\nabla(u_h-u_H)\|^2_{L^2(\Omega)}. \tag{4.18}$$

Proof. We have $u-u_H \in V$, $u-u_h \in V$ and because $V_H \subseteq V_h$ we also have $u_h-u_H \in V_h$, then we can write the following

$$\|\nabla(u-u_H)\|^2_{L^2(\Omega)} = \|\nabla(u-u_h+u_h-u_H)\|^2_{L^2(\Omega)} = \|\nabla(u-u_h)\|^2_{L^2(\Omega)} + 2a(u-u_h,u_h-u_H) + \|\nabla(u_h-u_H)\|^2_{L^2(\Omega)}$$

and because of Galerkin orthogonality property on V_h (Appendix 5.7), we take as $v_h = u_h - u_H \in V_h$ and we have $a(u-u_h,u_h-u_H) = a(u-u_h,v_h) = 0$, so

$$\|\nabla(u-u_H)\|^2_{L^2(\Omega)} = \|\nabla(u-u_h)\|^2_{L^2(\Omega)} + \|\nabla(u_h-u_H)\|^2_{L^2(\Omega)}.$$

□

Lemma 4.5.3. *(Estimator reduction)*

Based on Corollary 3.4 in Quasi-Optimal Convergence Rate for an Adaptive Finite Element Method [6], we have the following.

For a triangulation T_n and $T_{M_n} \subset T_n$ a subset of triangles on T_n, which we mark for refinement, we define $T_{n+1} := REFINE(T_n, T_{M_n})$ the new triangulation from the refinement of T_n on the triangles in the T_{M_n}.

Also if $\Lambda_1 := \dfrac{(d+1)\overline{\Lambda_1}^2}{\sqrt{c_1}}$ and $\lambda := 1-2^{-b/d} > 0$, where $\overline{\Lambda_1}$ is defined in Proposition 3.3 in Quasi-Optimal Convergence Rate for an Adaptive Finite Element Method [6], c_1 the constant from (b) in Lax - Milgram Theorem (Appendix 5.2), $d \geq 2$ the dimension and b the minimum amount of times each element of T_{M_n} needs to bisected, then, for all $v_n \in V_{T_n,g,D}$ and $v_{n+1} \in V_{T_{n+1},g,D}$ and any $\delta > 0$, we have

$$e^2(v_{n+1},T_{n+1}) \leq (1+\delta)\{e^2(v_n,T_n) - \lambda e^2(v_n,T_{M_n})\} + (1+\delta^{-1})\Lambda_1 e^2(v_0,T_0)\|\nabla(v_{n+1}-v_n)\|^2_{L^2(\Omega)}. \tag{4.19}$$

Now we use those lemmas to prove the following theorem which is the key for the proof of the convergence of Adaptive FEM to the weak solution.

Theorem 4.5.4. *(Contraction property)*

Based on Theorem 4.5.4 in Quasi-Optimal Convergence Rate for an Adaptive Finite Element Method [6], we have the following.

Let $\theta_n \in (0,1]$, the marking parameter of AFEM with $0 < \theta_{min} \leq \theta_n \leq 1$ for each n and θ_{min} a constant number, and let $\{T_n, V_n, u_n\}_{n\geq 0}$ be the sequence of meshes, finite element spaces (as defined before) and discrete solutions (produced by the AFEM for the finite element problem (4.9) in each T_n).

Then there exists $\gamma > 0$ and $0 < a_n < 1$, depending solely on the shape-regularity of T_0, b (the minimum amount of times an element is bisected at a triangulation step) and $0 < \theta_n \leq 1$, such that

$$\|\nabla(u - u_{n+1})\|^2_{L^2(\Omega)} + \gamma e^2_{n+1} \leq a_n^2(\|\nabla(u - u_n)\|^2_{L^2(\Omega)} + \gamma e^2_n), \tag{4.20}$$

where $u \in H^1_{g,D}(\Omega)$ the weak solution of (4.8).

Proof. From Lemma 4.5.2 (orthogonality property) we have

$$\|\nabla(u - u_{n+1})\|^2_{L^2(\Omega)} = \|\nabla(u - u_n)\|^2_{L^2(\Omega)} - \|\nabla(u_{n+1} - u_n)\|^2_{L^2(\Omega)} \tag{4.21}$$

and from Lemma 4.5.3 (estimator reduction), for any $\delta > 0$, we have

$$e^2(u_{n+1}, T_{n+1}) \leq (1+\delta)\{e^2(u_n, T_n) - \lambda e^2(u_n, T_{M_n})\} + (1+\delta^{-1})\Lambda_1 e^2(u_0, T_0)\|\nabla(u_{n+1} - u_n)\|^2_{L^2(\Omega)}, \tag{4.22}$$

which from (4.15) and (4.16) can be written as

$$e^2_{n+1} \leq (1+\delta)\{e^2_n - \lambda e^2_n(T_{M_n})\} + (1+\delta^{-1})\Lambda_1 e^2_0\|\nabla(u_{n+1} - u_n)\|^2_{L^2(\Omega)}. \tag{4.23}$$

Next, we multiply (4.23) with $\gamma := \dfrac{1}{(1+\delta^{-1})\Lambda_1 e_0^2}$ and add it to (4.21), so we get

$$\begin{aligned}\|\nabla(u - u_{n+1})\|^2_{L^2(\Omega)} + \gamma e^2_{n+1} \leq \|\nabla(u - u_n)\|^2_{L^2(\Omega)} - \|\nabla(u_{n+1} - u_n)\|^2_{L^2(\Omega)} \\ +\gamma(1+\delta)\{e^2_n - \lambda e^2_n(T_{M_n})\} + \gamma(1+\delta^{-1})\Lambda_1 e^2_0\|\nabla(u_{n+1} - u_n)\|^2_{L^2(\Omega)},\end{aligned}$$

which becomes

$$\|\nabla(u - u_{n+1})\|^2_{L^2(\Omega)} + \gamma e^2_{n+1} \leq \|\nabla(u - u_n)\|^2_{L^2(\Omega)} + \gamma(1+\delta)e^2_n - \gamma(1+\delta)\lambda e^2_n(T_{M_n}).$$

Now using the (4.14) we have

$$\|\nabla(u - u_{n+1})\|^2_{L^2(\Omega)} + \gamma e^2_{n+1} \leq \|\nabla(u - u_n)\|^2_{L^2(\Omega)} + \gamma(1+\delta)e^2_n - \gamma(1+\delta)\lambda\theta_n^2 e^2_n$$

and be rewrite it with any $\beta \in (0,1)$, we have

$$\|\nabla(u-u_{n+1})\|^2_{L^2(\Omega)}+\gamma e^2_{n+1} \leq \|\nabla(u-u_n)\|^2_{L^2(\Omega)}+\gamma(1+\delta)e^2_n-\beta\gamma(1+\delta)\lambda\theta_n^2 e^2_n-(1-\beta)\gamma(1+\delta)\lambda\theta_n^2 e^2_n.$$

Simplify further we have

$$\|\nabla(u-u_{n+1})\|^2_{L^2(\Omega)}+\gamma e^2_{n+1}\leq\|\nabla(u-u_n)\|^2_{L^2(\Omega)}-\beta\gamma(1+\delta)\lambda\theta_n^2e_n^2+\gamma(1+\delta)(1-(1-\beta)\lambda\theta_n^2)e_n^2$$

and, using $\gamma(1+\delta)=\frac{\delta}{\Lambda_1 e_0^2}$, we get

$$\|\nabla(u-u_{n+1})\|^2_{L^2(\Omega)}+\gamma e^2_{n+1}\leq\|\nabla(u-u_n)\|^2_{L^2(\Omega)}-\beta\frac{\lambda\theta_n^2}{\Lambda_1 e_0^2}\delta e_n^2+\gamma(1+\delta)(1-(1-\beta)\lambda\theta_n^2)e_n^2.$$

Finally applying (4.17), we have

$$\|\nabla(u-u_{n+1})\|^2_{L^2(\Omega)}+\gamma e^2_{n+1}\leq\|\nabla(u-u_n)\|^2_{L^2(\Omega)}-\beta\frac{\lambda\theta_n^2}{C_1\Lambda_1 e_0^2}\delta\|\nabla(u-u_n)\|^2_{L^2(\Omega)}+\gamma(1+\delta)(1-(1-\beta)\lambda\theta_n^2)e_n^2,$$

or

$$\|\nabla(u-u_{n+1})\|^2_{L^2(\Omega)}+\gamma e^2_{n+1}\leq a^2_{n_1}(\delta,\beta)\|\nabla(u-u_n)\|^2_{L^2(\Omega)}+\gamma a^2_{n_2}(\delta,\beta)e_n^2, \tag{4.24}$$

where

$$a^2_{n_1}(\delta,\beta):=1-\beta\frac{\lambda\theta_n^2}{C_1\Lambda_1 e_0^2}\delta,$$
$$a^2_{n_2}(\delta,\beta):=(1+\delta)(1-(1-\beta)\lambda\theta_n^2).$$

So for $a_n:=\sqrt{max\{a^2_{n_1},a^2_{n_2}\}}>0$ we have

$$\|\nabla(u-u_{n+1})\|^2_{L^2(\Omega)}+\gamma e^2_{n+1}\leq a_n^2(\|\nabla(u-u_n)\|^2_{L^2(\Omega)}+\gamma e_n^2) \tag{4.25}$$

and because $0<\theta_{min}\leq\theta_n\leq 1$ for each n, then by choosing $\delta>0$ small enough, we can have $a_n^2<1$, which proves the theorem. □

Finally we can now show the convergence the discrete solutions u_n of Adaptive FEM to the weak solution u of our weak problem.

Theorem 4.5.5. ***Adaptive FEM convergence to the weak solution***

If $u\in H^1_{g,D}(\Omega)$ the weak solution of (4.8), $u_n\in V_{T_n,g,D}$ the discrete solution of the finite element problem (4.9) for T_n, the n_{th} triangulation of Ω, and let $\theta_n\in(0,1]$, the marking parameter of AFEM with $0<\theta_{min}\leq\theta_n$ for each n and θ_{min} a constant number, then we have that

$$\lim_{n\to\infty}\|u-u_n\|_{H^1_{0,D}(\Omega)}=0. \tag{4.26}$$

From the criterion of Adaptive FEM Algorithm (subsection 4.4.3) and its first case we take as $\theta_{min}=\theta_n=errorPercentage$, while for its second case we take as $\theta_{min}=errorPercentageBoundary$.

Proof. From the contraction property (Theorem 4.5.4) we can see that because $0<\theta_{min}\leq\theta_n\leq 1\ \forall n\in N$, the a_n^2 stays between $(0,1)\ \forall n\in N$ and as $n\to\infty$, we have that

$$\lim_{n\to\infty}\left(\|\nabla(u-u_n)\|^2_{L^2(\Omega)}+\gamma e_n^2\right)=0.$$

Also because $\|\nabla(u-u_n)\|^2_{L^2(\Omega)} \geq 0$ and $\gamma e_n^2 \geq 0$, we have

$$\lim_{n\to\infty} \|\nabla(u-u_n)\|_{L^2(\Omega)} = 0,$$

which as we said would be the same as

$$\lim_{n\to\infty} \|u-u_n\|_{H^1_{0,D}(\Omega)} = 0. \tag{4.27}$$

□

4.6 Asymptotic analysis and comparison of regular and Adaptive FEM algorithms

4.6.1 Structure of Algorithms

In this section, we compare the regular and the Adaptive FEM algorithms based on their computational complexity functions (C). For both of them C is described from the following structure

$$C = At(NT_0) \ + \sum_{i=1}^{fn} Bt(NT_i), \tag{4.28}$$

where:

- NT_0 is the number of triangles (elements), at the initial refinement, before we enter the WHILE loop and it is constant based on initial configuration, which can be set the same for both algorithms.
- NT_i is the number of triangles (elements) we have at each iteration of WHILE loop. Per algorithm this will differ at each i, due to how the triangles are created at each step.
- $At(NT_0)$ is the function that describes the computational complexity for the initial number of triangles we want to compute before we enter the WHILE loop. $At(NT_0) = O(NT_0)$ (Big O asymptotic notation) for both algorithms.
- $Bt(NT_i)$ is the function that describes the computational complexity for the number of triangles we have at each iteration. $Bt(NT_i) = O(NT_i)$ (Big O asymptotic notation), so for both algorithms the main difference here is on how NT_i is created.
- fn is the number of iterations. For both algorithms fn is dependent to *errorBoundary* and in the case of Adaptive algorithm is also at *errorPercentage* for criterion 1 or *errorPercentageBoundary* for criterion 2. So fn will differ per algorithm to $fn_{regular}$ and $fn_{adaptive}$.

As we can see the actual difference of regular and Adaptive algorithms, for the same PDE problem, is how we mark the triangles for refinement (MARK TRIANGLES step) and it is what gives the difference at their computational complexity.

For Adaptive algorithm we take two variables as input at the beginning, *errorBoundary* and *errorPercentage* if we use Criterion 1 or *errorPercentageBoundary* if we use criterion 2.

- Criterion 1: *AFEM(errorBoundary, errorPercentage)*
- Criterion 2: *AFEM(errorBoundary, errorPercentageBoundary)*

For regular algorithm the variable *errorBoundary* is the only input we need, so we have *RFEM(errorBoundary)*. In reality regular is the same as Adaptive if we set:

- Criterion 1:

$$\begin{aligned} &\textit{RFEM(errorBoundary)} = \\ &\textit{AFEM(errorBoundary, 1)} \end{aligned}$$

- Criterion 2:

$$\begin{aligned} &\textit{RFEM(errorBoundary)} = \\ &\textit{AFEM(errorBoundary, 1)} \end{aligned}$$

4.6.2 Asymptotic analysis of regular FEM algorithm

First we take a look at how NT_{i+1}, the number of triangles at step $i+1$, is produced from NT_i, the number of triangles at the previous step i.

On the regular FEM algorithm at each step, we take **every** triangle on our mesh (all NT_i of them) and refine them (divide them to new triangles). The way we do that, may differs from implementation to implementation, but usually each triangle is replaced for a constant number of triangles, lets say $k+1$ with $k \in N$ and $k \geq 1$. So we have

$$NT_{i+1} = (k+1) \cdot NT_i = (k+1)^{i+1} \cdot NT_0,$$

which means

$$NT_i = (k+1)^i \cdot NT_0.$$

Then from (4.28) the computational complexity function C for regular FEM algorithm becomes the following

$$\begin{aligned} C_{regular} &= At(NT_0) + \sum_{i=1}^{fn} Bt(NT_i) = O\left(NT_0\right) + \sum_{i=1}^{fn} O\left(NT_i\right) = O\left(NT_0 + \sum_{i=1}^{fn} NT_i\right) \\ &= O\left(\sum_{i=0}^{fn} NT_i\right) = O\left(\sum_{i=0}^{fn} \left((k+1)^i \cdot NT_0\right)\right) = O((k+1)^{fn} \cdot NT_0) \end{aligned}$$

and because NT_0 is constant, we have

$$C_{regular} = O((k+1)^{fn}), \tag{4.29}$$

where fn is the function

$$fn = fn_{regular}(errorBoundary). \tag{4.30}$$

4.6.3 Asymptotic analysis of AFEM algorithm

Same as before we try to find out how NT_{i+1} is produced at each step from NT_i, but the difference of Adaptive from the regular FEM algorithm is that at each iteration, based on the MARK TRIANGLES step, we choose only **some** of the triangles of our mesh to refine (divide them to new triangles).

So what we have is the following

$$NT_{i+1} = NT_i + m(i+1) + s(i+1). \tag{4.31}$$

That means that number NT_{i+1} is the number of previous NT_i triangles plus two new quantities, $m(i+1)$ which is the number related to new triangles created from marked triangles and $s(i+1)$, the number related to new triangles created from side triangles. Below we define them more accurately.

- Definition of $m(i+1)$:

First we define $markedTriangles(i+1)$ as the number of marked triangles at step $i+1$. Each of those marked triangles at the refinement is replaced for a constant number of triangles, lets say $k+1$ and $k \in N$, $k \geq 1$, while k changes per implementation of the program. So $m(i+1)$ becomes

$$\begin{aligned} m(i+1) &= -markedTriangles(i+1) + (k+1) \cdot markedTriangles(i+1) \\ &= k \cdot markedTriangles(i+1), \end{aligned} \tag{4.32}$$

as we can see we have also a subtraction here and the reason for that, is because on (4.31) we replace the marked triangles ($markedTriangles(i+1)$) from the NT_i triangles, with the newly created triangles ($(k+1) \cdot markedTriangles(i+1)$).

- Definition of $s(i+1)$:

First we define $sideTriangles(i+1)$, as the number of side triangles on the marked ones, but without being marked themselves. We need them because at each new refinement, nodes are created on the edges of every marked triangle during it's division (not necessary at all the edges of a triangle) and some of those edges that get a new node, may also belong to side triangles that are not marked. Then, on those edges the new node we created will not be part of an angle for the side triangle.

In order not to produce hanging nodes of triangles, each node created on an edge belonging to two triangles, must refine those two triangles in order to become itself a vertex of triangles inside both of them. So the refine mesh function of our program will create triangles inside the marked ones with a predefined way as described before and will also create some extra triangles inside the side triangles. Each of these side triangles does not have necessarily the same $k+1$ factor as marked triangles when they are replaced with the new ones, but probably a smaller one, lets say $c+1$ with $c \in N$ and $1 \leq c \leq k$. So $s(i+1)$ becomes

$$\begin{aligned} s(i+1) &= -sideTriangles(i+1) + (c+1) \cdot sideTriangles(i+1) \\ &= c \cdot sideTriangles(i+1). \end{aligned} \tag{4.33}$$

The same remark we made for $m(i+1)$ about subtraction, applies also here.

So now coming back to (4.31) and using (4.32), (4.33) we have

$$NT_{i+1} = NT_i + k \cdot markedTriangles(i+1) + c \cdot sideTriangles(i+1),$$

which becomes

$$\begin{aligned} NT_{i+1} &\leq NT_i + k \cdot markedTriangles(i+1) + k \cdot sideTriangles(i+1) \\ &\leq NT_i + k \cdot (markedTriangles(i+1) + sideTriangles(i+1)), \end{aligned} \tag{4.34}$$

because $c \leq k$. Those $markedTriangles(i+1)$, $sideTriangles(i+1)$ are chosen based on our criterion and has as a result, the quantity $markedTriangles(i+1)+sideTriangles(i+1)$ we take for refinement to be a percentage of our NT_i from the previous step i. We denote this percentage $l(i+1) \in R$, with $0 < l(i+1) \leq 1$ for $(i = 0, 1, \cdots fn-1)$ and then $markedTriangles(i+1) + sideTriangles(i+1)$ can be written as

$$markedTriangles(i+1) + sideTriangles(i+1) = l(i+1) \cdot NT_i. \tag{4.35}$$

So now from (4.34), we have the following

$$NT_{i+1} = O(NT_i + k \cdot (markedTriangles(i+1) + sideTriangles(i+1))),$$

which because of (4.35) becomes

$$\begin{aligned} &NT_{i+1} = O(NT_i + k \cdot l(i+1) \cdot NT_i) = O((k \cdot l(i+1) + 1) \cdot NT_i) \\ &= O((k \cdot l(i+1) + 1) \cdot (k \cdot l(i) + 1) \cdot NT_{i-1}) = O\left(\left(\prod_{j=1}^{i+1}(k \cdot l(j) + 1)\right) \cdot NT_0\right) \end{aligned}$$

and so we have

$$NT_i = O\left(\left(\prod_{j=1}^{i}(k \cdot l(j) + 1)\right) \cdot NT_0\right).$$

Finally if we set as $l(0) = 0$, nothing changes much and we can have

$$NT_i = O\left(\left(\prod_{j=0}^{i}(k \cdot l(j) + 1)\right) \cdot NT_0\right).$$

Coming back to the calculation of the computational complexity function C for AFEM algorithm, we have from (4.28) that

$$\begin{aligned} C_{adaptive} &= At(NT_0) + \sum_{i=1}^{fn} Bt(NT_i) = O\left(NT_0\right) + \sum_{i=1}^{fn} O\left(NT_i\right) \\ &= O\left(NT_0 + \sum_{i=1}^{fn} NT_i\right) = O\left(NT_0 + \sum_{i=1}^{fn}\left(\prod_{j=0}^{i}(k \cdot l(j) + 1)\right) \cdot NT_0\right) \\ &= O\left(\sum_{i=0}^{fn}\left(\prod_{j=0}^{i}(k \cdot l(j) + 1)\right) \cdot NT_0\right) = O\left(\left(\prod_{j=0}^{fn}(k \cdot l(j) + 1)\right) \cdot NT_0\right) \end{aligned}$$

and because NT_0 is constant we have

$$C_{adaptive} = O\left(\prod_{j=0}^{fn}(k \cdot l(j) + 1)\right), \tag{4.36}$$

where for Criterion 1, fn is the function

$$fn = fn_{adaptive}(errorBoundary,\ errorPercentage) \tag{4.37}$$

and for Criterion 2 is

$$fn = fn_{adaptive}(errorBoundary,\ errorPercentageBoundary). \tag{4.38}$$

4.6.4 Comparison of regular and Adaptive FEM algorithms

What we actually see is that both computational complexity functions $C_{regular}$ and $C_{adaptive}$ are asymptotically equal with the number of triangles each algorithm creates and their formulas are

$$C_{regular} = O((k+1)^{fn_{regular}}),$$
$$C_{adaptive} = O\left(\prod_{j=0}^{fn_{adaptive}}(k \cdot l(j) + 1)\right);$$

when we talk for $C_{adaptive}$ we refer to both Criteria in Section 4.4.3.

Now let's take a closer look on the latter. We know that $k \in N$ and since $0 < l(i) \leq 1$, we have

$$1 < k \cdot l(i) + 1 \leq k + 1. \tag{4.39}$$

A note here is that the *refinemesh* function in MATLAB has $k = 3$.

Another thing we know is that for the same $errorBoundary$ we have $fn_{regular} \leq fn_{adaptive}$. Indeed, supposing that $fn_{adaptive} < fn_{regular}$, this would mean that adaptive algorithm has run $fn_{adaptive}$ triangulations and found a solution. We know from Section 4.4.4 that the regular algorithm creates a uniform mesh on the domain, while adaptive may have regions with different density between them. So the regular algorithm at $fn_{adaptive}$ number of triangulations should have already stopped, because compared to the adaptive it would have reached the same density or more in every region and it would have found the required or lower error. So $fn_{adaptive} < fn_{regular}$ can't be true.

Finally although $fn_{regular} \leq fn_{adaptive}$ for the same $errorBoundary$, if we take $errorBoundary \to 0$ then

$$fn_{adaptive} \to \infty \text{ and } fn_{regular} \to \infty.$$

As f_n gets closer to infinity we see that because of (4.39) the $C_{regular}$ quantity may grow faster than $C_{adaptive}$. So for really small $errorBoundary$ we expect $C_{adaptive} < C_{regular}$.

4.6.5 Comparison for the two cases of Adaptive FEM Algorithm

Finally for the Adaptive FEM algorithm we discussed about two cases, based on the two Criteria on Section 4.4.3. Often for really small *errorBoundary* the second criterion can give us a faster algorithm from the first, by giving a boost on the number of triangles we create on the first mesh refinements and delays the refinement process as the number of triangles gets larger.

The likely reason behind this, is that the percentage of error we use at each iteration, would be in some sense proportional to the number of triangles we choose for refinement and hence the number of triangles which will be created on this iteration. Large percentage of error means a lot more triangles on the end, compared with a small percentage of error.

So for Criterion 1 throughout the duration of the algorithm, the percentage of error at each iteration to determine the triangles that needs refinement, is constant. If we choose a large enough percentage, then at each iteration the algorithm produces a lot of triangles, maybe a lot more than we need. If we choose a small enough then we will need a lot iterations, because our progress will be small. We remind that for large percentage close to 1 the Adaptive FEM becomes more like the regular FEM.

On the other hand, Criterion 2 works in the following way. Instead of having the percentage of error to be a constant number, it changes at each iteration. In the beginning, and for the first refinements, we choose larger percentages of error to determine the triangles for refinements, because the computational cost is still small. However, in the process we reduce the percentage of error at each iteration, as the number of triangles increases: a large percentage can boost unnecessary their numbers very quickly and so the computational cost.

So for Adaptive FEM, Criterion 2, due to its adaptive nature, can give us faster results than Criterion 1.

4.7 An Example

As an example to apply the theory above we will use the following Poisson's problem:

$$-\Delta u = 1 \; in \; \Omega \subseteq R^2 \; with \; u : \Omega \mapsto R,$$
$$u = 0 \; on \; \partial\Omega,$$

where $\Omega = (-1,1)^2 \backslash ((0,1) \times (-1,0))$ an L-shaped domain. To find an approximation solution $u \in V_T \cap H_0^1(\Omega)$ for the respective finite element problem, we use both the Adaptive and regular FEM algorithms and compare them.

To have a better understanding of the behavior of those algorithms, we show their application for different cases of *errorBoundary*. For each case we apply both the Adaptive and regular FEM algorithms. For the Adaptive, we use the two iterations we presented before, one for Criterion 1 with *errorPercentage* and one for Criterion 2 with *errorPercentageBoundary*. In the plots below, we refer to the following variables as:

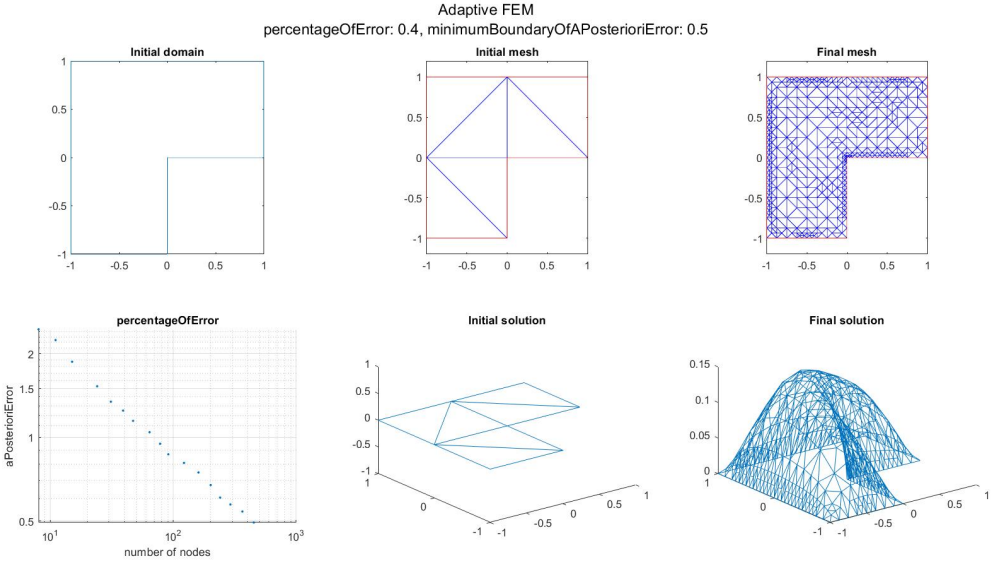
$$error \longrightarrow aPosterioriError,$$
$$errorBoundary \longrightarrow minimumBoundaryOfAPosterioriError,$$
$$errorPercentage \longrightarrow percentageOfError,$$
$$errorPercentageBoundary \longrightarrow adaptivePercentageOfError.$$

4.7.1 Case for minimumBoundaryOfAPosterioriError = 0.5

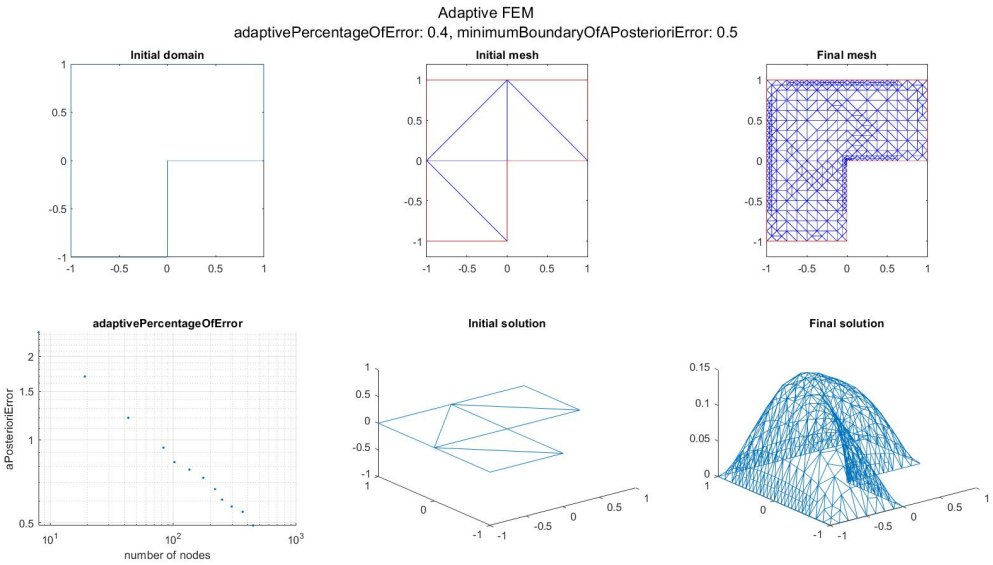
Adaptive FEM

We compute for different *percentageOfError* and *adaptivePercentageOfError*.

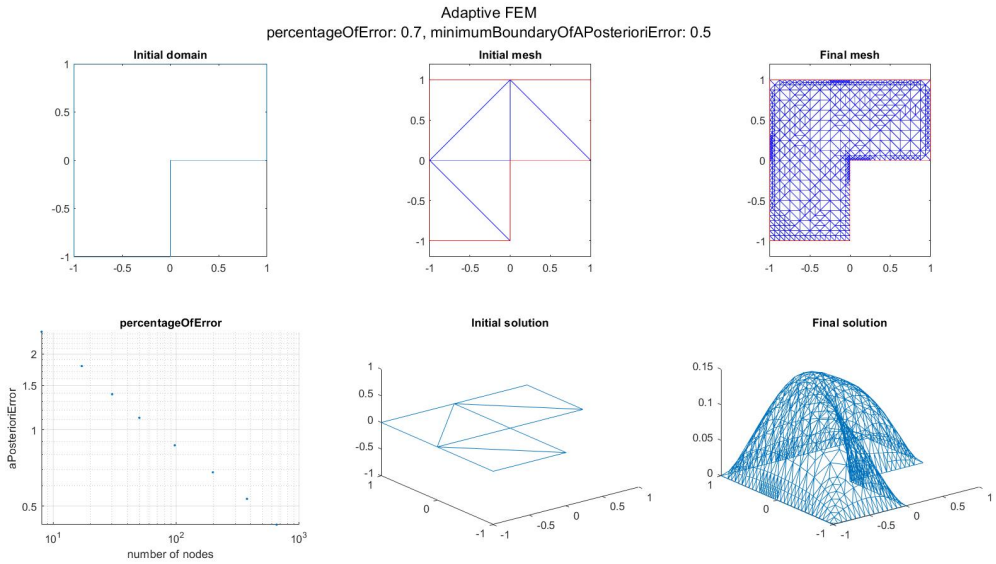
• Criterion 1: $percentageOfError = 0.4$



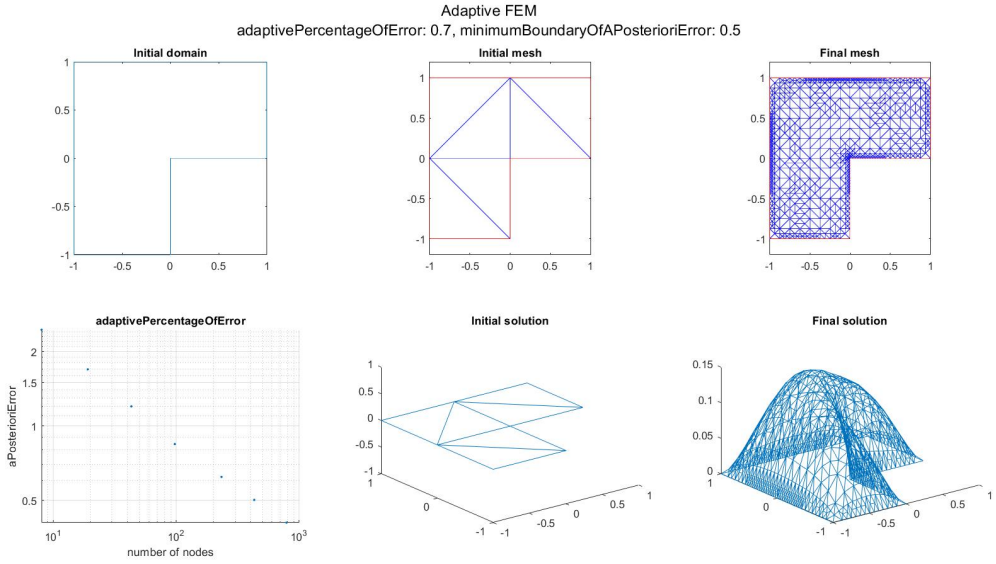
• Criterion 2: $adaptivePercentageOfError = 0.4$



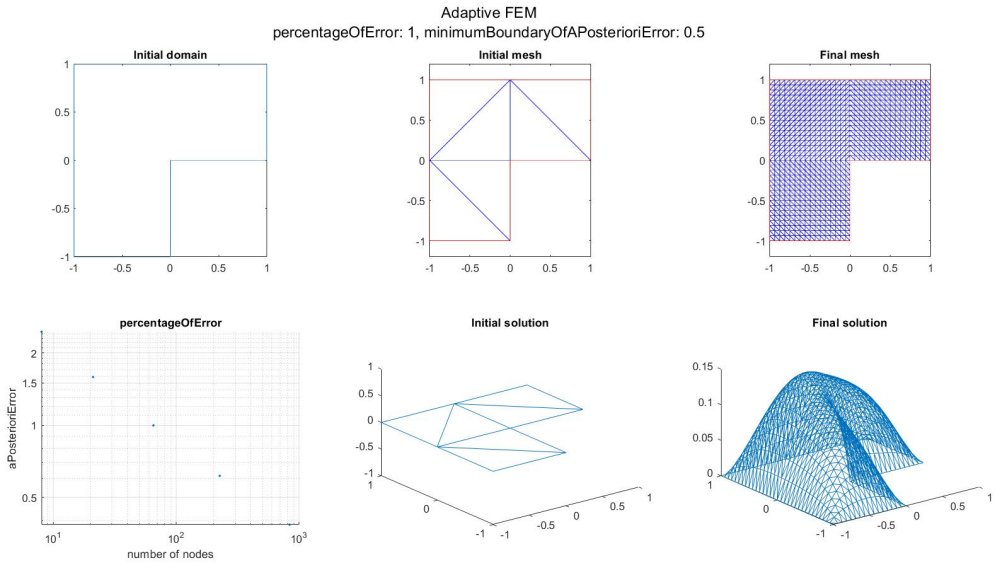
• Criterion 1: $percentageOfError = 0.7$



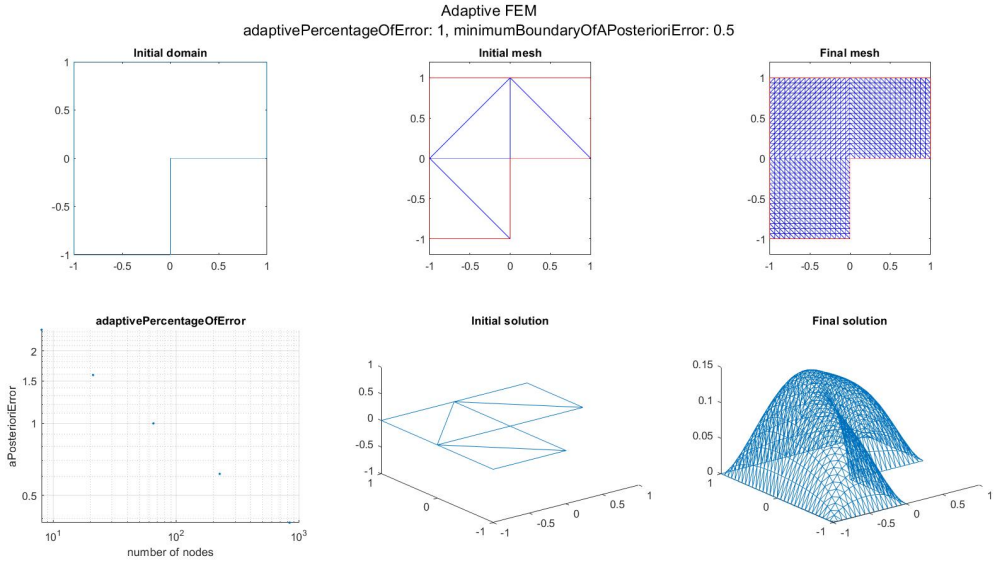
- Criterion 2: $adaptivePercentageOfError = 0.7$



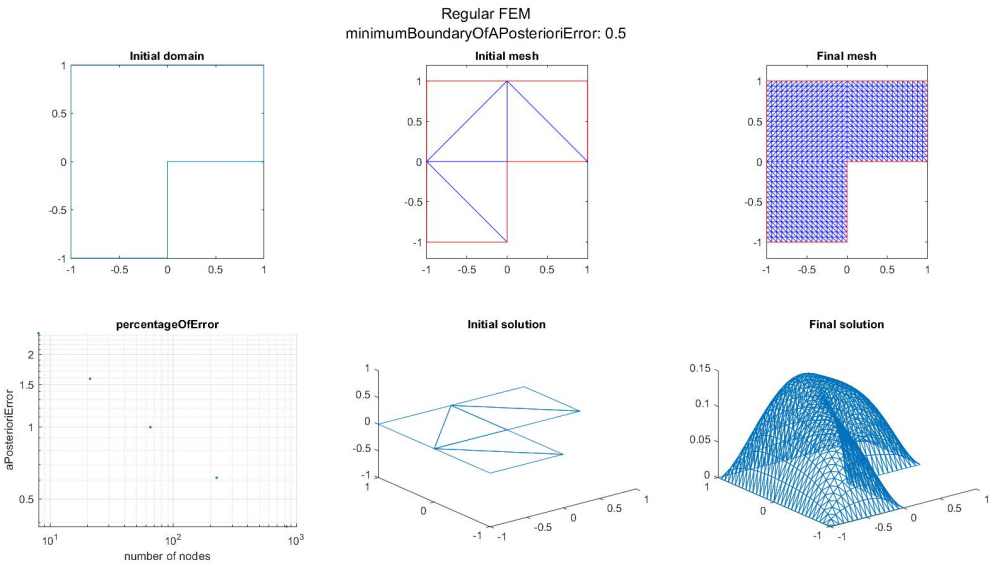
- Criterion 1: $percentageOfError = 1$



- Criterion 2: $adaptivePercentageOfError = 1$



Regular FEM

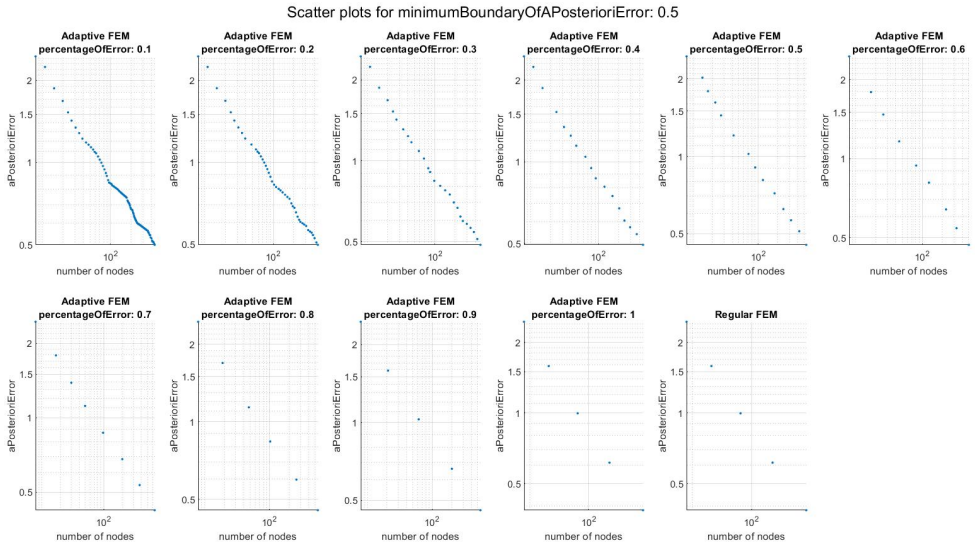


From the plots above we observe some of the things we described on subsection 4.6.5:

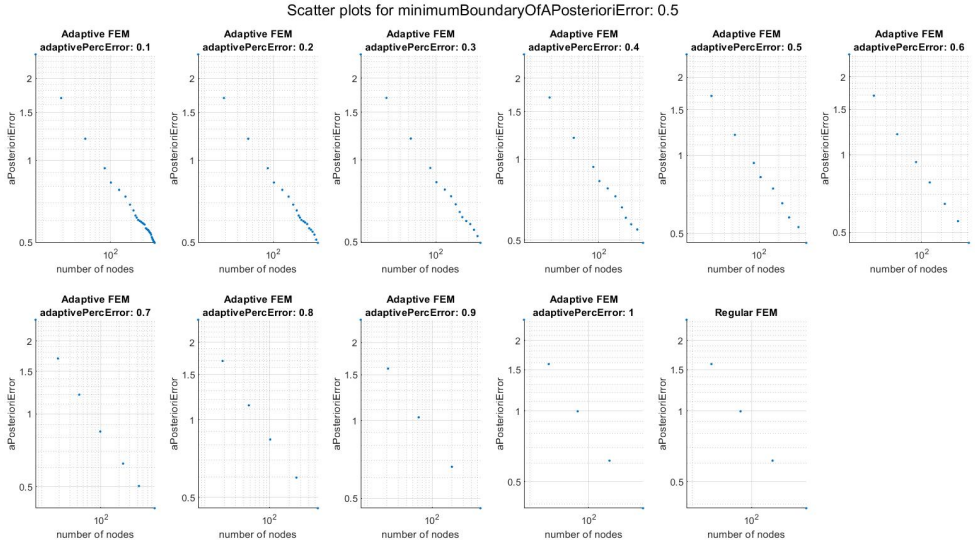
- The meshes of Adaptive FEM examples are not uniform, unlike the Regular's mesh.
- The number of points on the scatter plots for the ***aPosterioriError vs number of nodes***, shows how many time the program will run until it reaches the desired *aPosteriorriError*. We can see that for Adaptive FEM Criterion 2 will run fewer times than Criterion 1.
- As the *percentageOfError* or *adaptivePercentageOfError* approaches 1, Adaptive FEM becomes more like regular FEM and when they reach 1, they become identical.

Scatter Log/Log plots for Adaptive and regular FEM

Below we have some more examples in scatter log/log plots (***aPosterioriError vs number of nodes***) for Adaptive FEM and regular FEM, in order to see how the aPosterioriError decreases as the nodes increase. For Adaptive FEM we will look at both criteria.
• Adaptive FEM with Criterion 1 and regural FEM



• Adaptive FEM with Criterion 2 and regural FEM



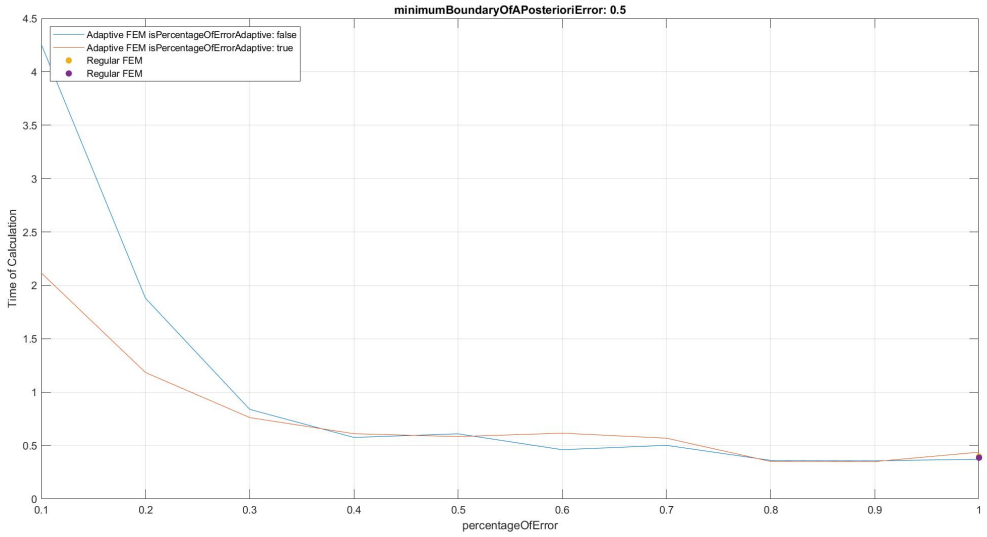
So these plots give us a better look of how the algorithms behave when changing the *percentageOfError* or *adaptivePercentageOfError* and again confirm the points from the theory we noticed with the previous charts.

Also, we can see that the final number of nodes created in each case differs. This number of nodes is proportional to the number of triangles created at the end and, as we have shown in Section 4.6, for small *minimumBoundaryOfAPosterioriError* it can give us a picture for the computational cost of the algorithms.

Computational time comparison plot of Adaptive and regular FEM

Below is the plot of the computational time in seconds as the *percentageOfError* changes for Adaptive FEM with Criterion 1, Adaptive FEM with Criterion 2 and regular FEM.

For Criterion 1 ($isPercentageOfErrorAdaptive = false$) we use as usual *percentageOfError*, while for Criterion 2 ($isPercentageOfErrorAdaptive = true$) with *percentageOfError* we mean *adaptivePercentageOfError* and for regular FEM is just the same as when we take $percentageOfError = adaptivePercentageOfError = 1$



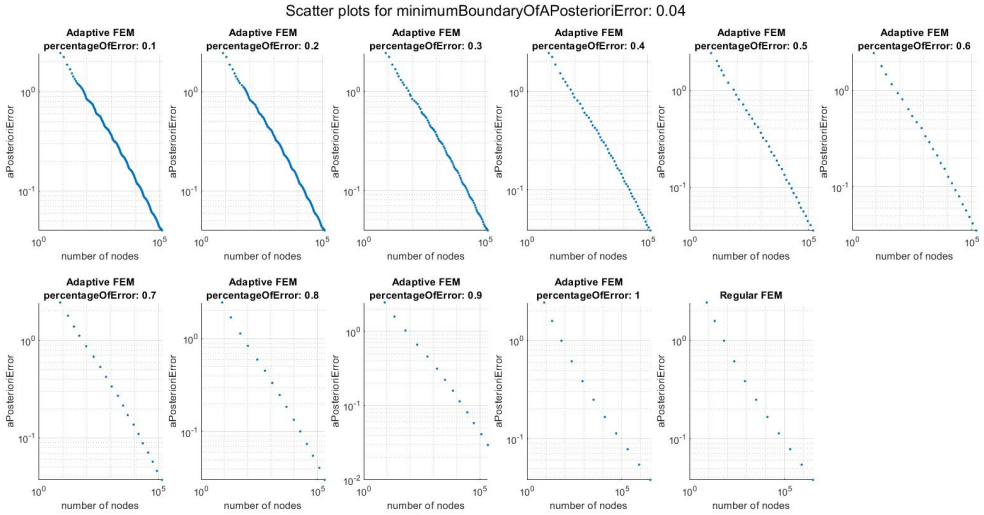
As we can see Adaptive FEM for Criterion 2 is faster than Adaptive FEM for Criterion 1 (mostly because of the fewer iterations). Also regular FEM for most of the *percentageOfError* seems to be faster from any Adaptive, this is in contrast to what we saw on section 4.6, but this happens because we are in the case of $minimumBoundaryOfAPosterioriError = 0.5$ and although the number of final nodes for regular FEM is greater than Adaptive's FEM, here the computational cost is already small at each iteration, so the number of iterations are what's more important at this stage. As *minimumBoundaryOfAPosterioriError* gets smaller this will begin to change and the number of final nodes would be the main drive for the computational cost, so the regular FEM will become the slowest.

So to see that, we have below some of the results for a much smaller case for *minimumBoundaryOfAPosterioriError*.

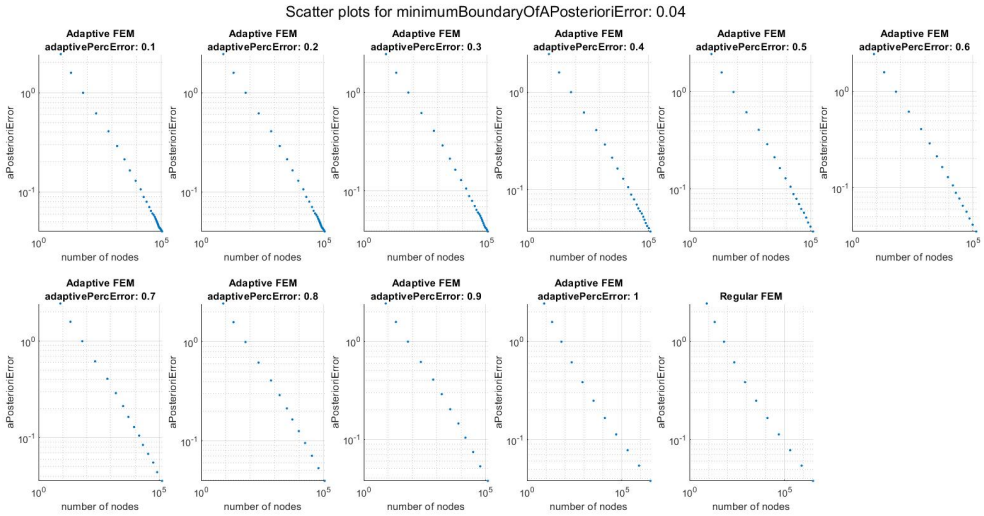
4.7.2 Case for minimumBoundaryOfAPosterioriError = 0.04

Scatter Log/Log plots for Adaptive and regular FEM

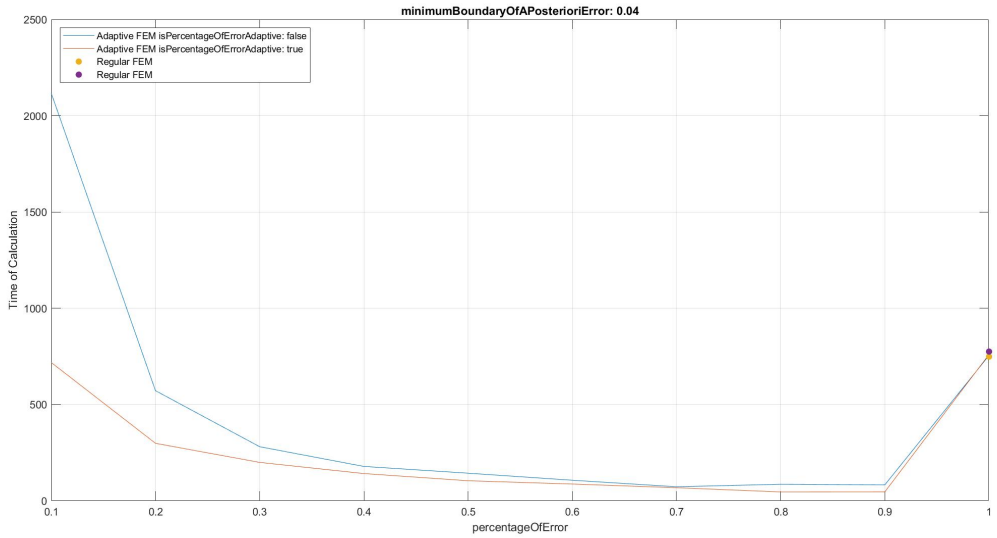
- Adaptive FEM with Criterion 1 and regural FEM



- Adaptive FEM with Criterion 2 and regural FEM



Computational time comparison plot of Adaptive and regular FEM



Finally we see here that for a smaller case of *minimumBoundaryOfAPosterioriError* the Adaptive FEM becomes faster from the regular FEM for most of the *percentageOfErrors*.

Chapter 5

Appendix

5.1 Gauss - Green Theorem

Suppose $\Omega \subseteq R^n,\ u_i \in C^1(\overline{\Omega})$ then

$$\int_\Omega \frac{\partial u_i}{\partial x_i} dx \ = \ \int_{d\Omega} u_i n_i dS \ \ for\ i = 1, 2, \cdots, n$$

and if we take $\vec{u} = (u_1, u_2, u_3, \cdots, u_n) \in R^n$ and $\vec{n} = (n_1, n_2, n_3, \cdots, n_n) \in R^n$ a unit normal vector to $\partial\Omega$, then we have

$$\int_\Omega \nabla \cdot \vec{u} dx \ = \ \int_{\partial\Omega} \vec{u} \cdot \vec{n} dS.$$

For reference see the Appendix C.2 in *Partial Differential Equations* [7] and the divergence theorem on page 97, in *Computational Methods for Partial Differential Equations* [2].

5.2 Lax - Milgram Theorem

Suppose that V is a real Hilbert space equipped with norm $\| \cdot \|_V$. Let $a(\cdot, \cdot)$ be a bilinear functional on $V \times V$ and $l(\cdot)$ be a linear functional on V such that:

(a) $\exists c_0 > 0 : \forall v \in V \ \ a(v, v) \geq c_0 \|v\|_V^2$

(b) $\exists c_1 \geq 0 : \forall v, w \in V \ \ |a(w, v)| \leq c_1 \|w\|_V \|v\|_V$

(c) $\exists c_2 \geq 0 : \forall v \in V \ \ |l(v)| \leq c_2 \|v\|_V$

Then a unique $u \in V$ exists such that $a(u, v) = l(v) \ \ \forall v \in V$.

For reference see the Theorem 1 in *Lecture Notes on Finite Element Methods for Partial Differential Equations* [1].

5.3 Poincaré - Friedrichs inequality

Suppose that Ω is a bounded open set in R^n (with a sufficiently smooth boundary $d\Omega$) and let $u \in H^1_{0,D}(\Omega)$ then there exists a constant $c_*(\Omega) \geq 0$ independent of u, such that

$$\int_\Omega |u(x)|^2 dx \leq c_* \sum_{i=1}^{n} \int_\Omega |\frac{\partial u}{\partial x_i}(x)|^2 dx.$$

For reference see the Lemma 2 in *Lecture Notes on Finite Element Methods for Partial Differential Equations* [1].

5.4 Cauchy-Schwarz inequality

For all $u, v \in V$ with V an inner product space, with $(\cdot, \cdot)$ the inner product and $\|v\| = |(v, v)|^{1/2}$ the norm produced from it, we have that

$$|(u, v)|^2 \leq (u, u) \cdot (v, v),$$

which means

$$|(u, v)| \leq \|u\| \|v\|.$$

For example, if we take $V = L_2(\Omega)$ and $u, v \in L_2(\Omega)$, we have $|(u, v)_{L_2(\Omega)}| \leq \|u\|_{L_2(\Omega)} \|v\|_{L_2(\Omega)}$.

For reference see the Lemma 1 in *Lecture Notes on Finite Element Methods for Partial Differential Equations* [1] and the Appendix B.2 in *Partial Differential Equations* [7].

5.5 Continuity of trace function on boundary $\partial\Omega$ for $v \in H^1(\Omega)$

The trace function $T_r(v)$ on boundary $\partial\Omega$ of Ω, for $v \in H^1(\Omega)$ is defined as $T_r(v) = v_{|\partial\Omega}$ with $T_r : H^1(\Omega) \longrightarrow L_2(\partial\Omega)$. The trace function $T_r(v)$ is a continuous linear operator, which means that for every $v \in H^1(\Omega)$ there is $c \geq 0 \in R$, such as

$$\|v_{|\partial\Omega}\|_{L_2(\partial\Omega)} \leq c \cdot \|v\|_{H^1(\Omega)}.$$

For reference see the section 5.5, Theorem 1 (Trace Theorem) in *Partial Differential Equations* [7].

5.6 Continuity of trace function on Neumann boundary Γ_N for $v \in H^1_{0,D}(\Omega)$

We define the function $T_{r_{\Gamma_N}}(v) = v_{|\Gamma_N}$, with $T_{r_{\Gamma_N}} : H^1_{0,D}(\Omega) \longrightarrow L_2(\Gamma_N)$, where Γ_N is the Neumann boundary defined on (1.3) of our PDE problem. We call $T_{r_{\Gamma_N}}(v)$ the trace function on Γ_N, for $v \in H^1_{0,D}(\Omega)$ and we prove that it is a continuous linear function.

Proof. First we show the linearity of $T_{r_{\Gamma_N}}$. For each $u, w \in H^1_{0,D}(\Omega)$ and $\lambda, \mu \in R$ we have

$$\begin{aligned} T_{r_{\Gamma_N}}(\lambda u + \mu w) &= (\lambda u + \mu w)_{|\Gamma_N} = (\lambda u)_{|\Gamma_N} + (\mu w)_{|\Gamma_N} \\ &= \lambda u_{|\Gamma_N} + \mu w_{|\Gamma_N} = \lambda T_{r_{\Gamma_N}}(u) + \mu T_{r_{\Gamma_N}}(w). \end{aligned}$$

Then we show the continuity. For $v \in H^1_{0,D}(\Omega) \subseteq H^1(\Omega)$ with $\partial\Omega = \Gamma_D \cup \Gamma_N$, we have

$$\|v_{|\partial\Omega}\|_{L_2(\partial\Omega)} = \left(\int_{\partial\Omega} v_{|\partial\Omega}(x)^2 dx\right)^{1/2} = \left(\int_{\Gamma_D} v_{|\partial\Omega}(x)^2 dx + \int_{\Gamma_N} v_{|\partial\Omega}(x)^2 dx\right)^{1/2},$$

which because of $v_{|\Gamma_D} = 0$ becomes

$$\|v_{|\partial\Omega}\|_{L_2(\partial\Omega)} = \left(\int_{\Gamma_N} v_{|\partial\Omega}(x)^2 dx\right)^{1/2} = \left(\int_{\Gamma_N} v_{|\Gamma_N}(x)^2 dx\right)^{1/2} = \|v_{|\Gamma_N}\|_{L_2(\Gamma_N)} = \|v\|_{L_2(\Gamma_N)}.$$

From continuity of trace function on boundary $\partial\Omega$ for $v \in H^1(\Omega)$ (Appendix 5.5), there is $c \geq 0 \in R$ such as $\|v_{|\partial\Omega}\|_{L_2(\partial\Omega)} \leq c \cdot \|v\|_{H^1(\Omega)}$, so

$$\|v\|_{L_2(\Gamma_N)} \leq c \cdot \|v\|_{H^1(\Omega)}$$

and because $v \in H^1_{0,D}(\Omega) \subseteq H^1(\Omega)$, we have

$$\|v\|_{L_2(\Gamma_N)} \leq c \cdot \|v\|_{H^1_{0,D}(\Omega)},$$

so $T_{r_{\Gamma_N}}(v)$ is a continuous operator. □

5.7 Galerkin orthogality

Lemma 5.7.1. *If T_h is a triangulation of Ω, $u \in H^1_{0,D}(\Omega)$ the weak solution of (1.25) and $u_h \in T_h \cap H^1_{0,D}(\Omega)$ the solution of the finite element problem (2.7), then for every $v_h \in T_h \cap H^1_{0,D}(\Omega)$ we have*

$$a(u - u_h, v_h) = 0$$

Proof. If T_h is a triangulation of Ω and since $T_h \cap H^1_{0,D}(\Omega) \subset H^1_{0,D}(\Omega)$, then for every $v_h \in T_h \cap H^1_{0,D}(\Omega) \subset H^1_{0,D}(\Omega)$, we have from (1.25) that

$$a(u, v_h) = l(v_h)$$

and from (2.7) that

$$a(u_h, v_h) = l(v_h).$$

So by subtracting them we have

$$a(u - u_h, v_h) = a(u, v_h) - a(u_h, v_h) = 0.$$

□

Lemma 5.7.2. *We can see easily that the same property is also valid, if T_h is a triangulation of Ω, $u \in H^1_{g,D}(\Omega)$ the weak solution of (4.8) and $u_h \in V_{T_h,g,D}$ the solution of the finite element problem (4.9), then for every $v_h \in T_h \cap H^1_{0,D}(\Omega)$ we have*

$$a(u - u_h, v_h) = 0.$$

5.8 Existence of interpolator of Clément Theorem

If $v \in H^1(\Omega)$ and let's take a triangulation T of Ω and a linear space $H_n \subseteq H^1(\Omega)$ with a finite basis defined by triangulation T on Ω, then there exists $v_n \in H_n$ and $c \geq 0 \in R$, such as

$$\sum_{t \in T} h_t^{-2} \|v - v_n\|^2_{L^2(t)} + \sum_{e \in S(T)} h_e^{-1} \|v - v_n\|^2_{L^2(e)} \leq c \|\nabla v\|^2_{L^2(\Omega)},$$

with $v_n \in H_n$ called interpolator of of Clément.

5.9 Implementation of the a-posteriori error estimate

In order to be able to calculate the a-posteriori error bound (3.8) in our computer implementation of the algorithms used at chapter 4, we need to be more precise about how to calculate the error estimator per triangle $e_{tr}(u_n, f, t)$ defined in (3.7).

At first we write u_n as $u_n(x) = \sum_{i=1}^{N} u_i \phi_i(x)$, where N is the number of all nodes in the triangulation T, $\phi_i \in V_T$ belongs to the basis of linear space V_T, as described in lemma 2.1.2 and $(u_1, u_2, \cdots, u_N)$ is the vector solution we looked for the finite element approximation.

Also when the u_n is defined at each $t \in T$, it can be written as $u_{n|t} = \sum_{k=1}^{3} u_{i_k} \phi_{i_k}(x)$, where $x \in t$ and with i_k for $k = 1, 2, 3$ we mean the number of nodes for the specific triangle t, while in all other $i = 1, \cdots, N$ except from i_k the $\phi_i(x) = 0$ for $x \in t$. Using that we have that $\nabla u_{n|t} = \sum_{k=1}^{3} u_{i_k} \nabla \phi_{i_k}(x)$ and because $\phi_{i_k}(x)$ is a linear function at each t, we have that $\nabla \phi_{i_k}(x)$ is a constant vector, which implies that $\Delta u_{n|t} = 0$.

So from (3.7) we have

$$e_{tr}(u_n, f, t) = \left(h_t^2 \|f + \Delta u_n\|^2_{L^2(t)} + \sum_{e \in dt} h_e \|L(u_n, e)\|^2_{L^2(e)} \right)^{1/2},$$

which because $\Delta u_{n|t} = 0$, it can be written as

$$e_{tr}(u_n, f, t) = \left(h_t^2 \|f\|^2_{L^2(t)} + \sum_{e \in dt} h_e \|L(u_n, e)\|^2_{L^2(e)} \right)^{1/2}$$

and now only the $L(u_n, e)$ needs more analysis, as all the other info we need to calculate $e_{tr}(u_n, f, t)$, seems to be provided.

As defined before in Chapter 3, we have $L(u_n, e) = [\nabla u_n]_e \ \forall e \in S(T) \backslash \Gamma_N$ and $L(u_n, e) = (\nabla u_n \cdot \vec{n} - g_N)_{|e} \ \forall e \in \Gamma_N$ and on an edge e that belongs between t and t' ($e = dt \cap dt'$), we have that $[\nabla u_n]_e = \nabla u_{n|t_{|e}} \cdot \vec{n_{t_e}} + \nabla u_{n|t'_{|e}} \cdot \vec{n_{t'_e}}$. Also keep in mind that if e belong to the boundary Γ_D, then depending on which side of e is outside the triangulation T, we take $\nabla u_{n|t_{|e}}$ or $\nabla u_{n|t'_{|e}}$ as zero on e. So now using $u_{n|t} = \sum_{k=1}^{3} u_{i_k} \phi_{i_k}(x)$, we have

$$[\nabla u_n]_e = (\sum_{j=1}^{3} u_{i_{t_j}} \nabla \phi_{i_{t_j}})_{|e} \cdot \vec{n_{t_e}} + (\sum_{j=1}^{3} u_{i_{t'_j}} \nabla \phi_{i_{t'_j}})_{|e} \cdot \vec{n_{t'_e}}.$$

What remains now is to find a way we can compute on an edge e in the triangle t the $\nabla \phi_{i_{t_j}}$, $\vec{n_{t_e}}$ used above, the same process will be true on an edge e for the triangle t'.

5.9.1 Calculation of $\nabla\phi_{i_{t_j}}$ for an edge e on a triangle t

First, we find $\phi_{i_{t_j}}$ for each $j = 1, 2, 3$ on the triangle t, where j is a reference to the nodes of t. To be more precise we take as $t_1 = (x_1, y_1)$, $t_2 = (x_2, y_2)$, $t_3 = (x_3, y_3)$, the nodes of triangle t. Now because each $\phi_{i_{t_j}}$ (for $j = 1, 2, 3$) is linear and creates a plane on t, we can find its equation with the following way.

For each $j = 1, 2, 3$ we take three points on the plane of $\phi_{i_{t_j}}$, with each corresponding to a node on t and we write them as

$$P = (x_1, y_1, \phi_{i_{t_j}}(x_1, y_1))$$
$$Q = (x_2, y_2, \phi_{i_{t_j}}(x_2, y_2))$$
$$S = (x_3, y_3, \phi_{i_{t_j}}(x_3, y_3)).$$

For the case of $j = 1$, we have

$$\phi_{i_{t_1}}(x_1, y_1) = 1$$
$$\phi_{i_{t_1}}(x_2, y_2) = 0$$
$$\phi_{i_{t_1}}(x_3, y_3) = 0,$$

while for $j = 2$, we have

$$\phi_{i_{t_2}}(x_1, y_1) = 0$$
$$\phi_{i_{t_2}}(x_2, y_2) = 1$$
$$\phi_{i_{t_2}}(x_3, y_3) = 0$$

and finally for $j = 3$, we have

$$\phi_{i_{t_3}}(x_1, y_1) = 0$$
$$\phi_{i_{t_3}}(x_2, y_2) = 0$$
$$\phi_{i_{t_3}}(x_3, y_3) = 1.$$

Now we can take PQ and PS, as the vectors lying on the plane created by $\phi_{i_{t_j}}$, with

$$PQ = (x_2 - x_1, y_2 - y_1, \phi_{i_{t_j}}(x_2, y_2) - \phi_{i_{t_j}}(x_1, y_1))$$
$$PS = (x_3 - x_1, y_3 - y_1, \phi_{i_{t_j}}(x_3, y_3) - \phi_{i_{t_j}}(x_1, y_1))$$

and use them to find a normal vector N on that plane, described as $N = PQ \times PS = (l_1, l_2, l_3)$. So now the equation of the plane would be

$$l_3 \cdot (z - \phi_{i_{t_j}}(x_1, y_1)) + l_1 \cdot (x - x_1) + l_2 \cdot (y - y_1) = 0,$$

which solving for z and replacing it with $\phi_{i_{t_j}}(x, y)$ becomes

$$\phi_{i_{t_j}}(x, y) = -\frac{l_1}{l_3}(x - x_1) - \frac{l_2}{l_3}(y - y_1) + \phi_{i_{t_j}}(x_1, y_1)$$

and so we have

$$\nabla\phi_{i_{t_j}}(x,y) = (-\frac{l_1}{l_3}, -\frac{l_2}{l_3}).$$

So it seems $\nabla\phi_{i_{t_j}}(x,y)$ is a constant vector on the triangle t and so if we restrict it on the edge e.

5.9.2 Calculation of $\vec{n_{t_e}}$ for an edge e on a triangle t

We need to find the outward pointing unit normal vector $\vec{n_{t_e}}$ for an edge e on the triangle t. To do that we take as $t = \overset{\triangle}{ACB}$, where $A = (x_1, y_1)$, $B = (x_2, y_2)$, $C = (x_3, y_3)$, then we take as $e = AB$ and finally as $\vec{n_t} = PM$, the line segment with $P = (x, y)$ and $M = (\frac{x_1+x_2}{2}, \frac{y_1+y_2}{2})$ the mean of AB.

Also because we want MP to be an outward pointing unit normal vector the following properties must also be satisfied:

1. $MP \perp AB$
2. $|MP| = 1$
3. $P = (x, y)$ is from the opposite side of AB than $C = (x_3, y_3)$

So from property 1. we have

$$(x - \frac{x_1+x_2}{2}, y - \frac{y_1+y_2}{2}) \cdot (x_2 - x_1, y_2 - y_1) = 0,$$

which can be simplified further to

$$(x_2 - x_1)x - (x_2 - x_1)\frac{x_1+x_2}{2} + (y_2 - y_1)y - (y_2 - y_1)\frac{y_1+y_2}{2} = 0$$

and then if $x_2 - x_1 \neq 0$, we have

$$x = -\frac{y_2 - y_1}{x_2 - x_1}y + \frac{y_2 - y_1}{x_2 - x_1}\frac{y_1+y_2}{2} + \frac{x_1+x_2}{2}, \tag{5.1}$$

else if $x_2 - x_1 = 0$, we have $y_2 - y_1 \neq 0$ (because if not the A and B will be the same points) and so the equation becomes

$$(y_2 - y_1)y - (y_2 - y_1)\frac{y_1+y_2}{2} = 0,$$

which means

$$y = \frac{y_1+y_2}{2}. \tag{5.2}$$

Also from property 2. where $|PM| = 1$, we have

$$\left(x - \frac{x_1+x_2}{2}\right)^2 + \left(y - \frac{y_1+y_2}{2}\right)^2 = 1$$

and using that we have the following cases.

• **Case A**: If $x_2 - x_1 = 0$, then from (5.1) we have $x = -\dfrac{y_2 - y_1}{x_2 - x_1}y + \dfrac{y_2 - y_1}{x_2 - x_1}\dfrac{y_1 + y_2}{2} + \dfrac{x_1 + x_2}{2}$ and using $|MP| = 1$ from property 1. we have

$$\left(-\frac{y_2 - y_1}{x_2 - x_1}y + \frac{y_2 - y_1}{x_2 - x_1}\frac{y_1 + y_2}{2} + \frac{x_1 + x_2}{2} - \frac{x_1 + x_2}{2}\right)^2 + \left(y - \frac{y_1 + y_2}{2}\right)^2 = 1,$$

which becomes

$$\left(\frac{y_2 - y_1}{x_2 - x_1}\right)^2 \left(-y + \frac{y_1 + y_2}{2}\right)^2 + \left(y - \frac{y_1 + y_2}{2}\right)^2 = 1,$$

then simplifying further we have

$$\left(\left(\frac{y_2 - y_1}{x_2 - x_1}\right)^2 + 1\right)\left(y - \frac{y_1 + y_2}{2}\right)^2 = 1$$

and so we have

$$\left(y - \frac{y_1 + y_2}{2}\right)^2 = \frac{1}{1 + \left(\dfrac{y_2 - y_1}{x_2 - x_1}\right)^2},$$

which finally becomes

$$|y - \frac{y_1 + y_2}{2}| = \left(\frac{1}{1 + \left(\dfrac{y_2 - y_1}{x_2 - x_1}\right)^2}\right)^{1/2}.$$

So now if $y - \dfrac{y_1 + y_2}{2} \geq 0$, then

$$y - \frac{y_1 + y_2}{2} = \left(\frac{1}{1 + \left(\dfrac{y_2 - y_1}{x_2 - x_1}\right)^2}\right)^{1/2},$$

which means

$$y = \left(\frac{1}{1 + \left(\dfrac{y_2 - y_1}{x_2 - x_1}\right)^2}\right)^{1/2} + \frac{y_1 + y_2}{2}, \tag{5.3}$$

while if $y - \dfrac{y_1 + y_2}{2} < 0$, we have

$$y - \frac{y_1 + y_2}{2} = -\left(\frac{1}{1 + \left(\dfrac{y_2 - y_1}{x_2 - x_1}\right)^2}\right)^{1/2}$$

and so

$$y = -\left(\frac{1}{1+\left(\frac{y_2-y_1}{x_2-x_1}\right)^2}\right)^{1/2} + \frac{y_1+y_2}{2}. \tag{5.4}$$

So from (5.1) and (5.3) we have the point $P_1 = (x, y)$ and from (5.1) and (5.4) we have the point $P_2 = (x, y)$. Now we use the property 3. in order to conclude which of the P_1 and P_2 will be our $P(x, y)$, to do that we look which of the two is further away from C, that means which of the $\|P_1C\|$ or $\|P_2C\|$ is larger.

• **Case B**: If $x_2 - x_1 = 0$, then from (5.2) we have $y = \frac{y_1+y_2}{2}$ and using $|PM| = 1$ from property 2. we have

$$\left(x - \frac{x_1+x_2}{2}\right)^2 + \left(\frac{y_1+y_2}{2} - \frac{y_1+y_2}{2}\right)^2 = 1$$

and so

$$|x - \frac{x_1+x_2}{2}| = 1,$$

which means

$$x = 1 + \frac{x_1+x_2}{2} \text{ or } x = -1 + \frac{x_1+x_2}{2}.$$

So one case is $P_1 = (x, y) = (1 + \frac{x_1+x_2}{2}, \frac{y_1+y_2}{2})$ and the other case $P_2 = (x, y) = (-1 + \frac{x_1+x_2}{2}, \frac{y_1+y_2}{2})$. Now we do the same as before, we use the property 3. in order to conclude which of the P_1 and P_2 will be our $P(x, y)$.

Finally using case A or case B, we can find $\vec{n_t} = MP = (x - \frac{x_1+x_2}{2}, y - \frac{y_1+y_2}{2})$ and now we have most of the info we need to calculate the error.

Bibliography

[1] Endre Süli, Lecture Notes on Finite Element Methods for Partial Differential Equations, Mathematical Institute University of Oxford, December 2012.

[2] Manolis Georgoulis, Computational Methods for Partial Differential Equations, Department of Mathematics University of Leicester, January 2009.

[3] Long Chen, Programming Of Finite Element Methods in Matlab.

[4] Ricardo H. Nochetto, Adaptive Finite Element Methods For Elliptic PDE.

[5] L. Ridgway Scott and Shangyou Zhang, Finite Element Interpolation of Nonsmooth Functions Satisfying Boundary Conditions, Mathematics of Computation Volume 54, Number 190, April 1990, Pages 483-493.

[6] J. Manuel Cascon, Christian Kreuzer, Ricardo H. Nochetto, Kunibert G. Siebert, Quasi-Optimal Convergence Rate for an Adaptive Finite Element Method, SIAM Journal on Numerical Analysis, Vol. 46, No. 5, pp 2524 - 2550, Year 2008.

[7] Lawrence C. Evans, Partial Differential Equations, Volume 19