
Εθνικό Μετσόβιο Πολυτεχνείο

Σχολή Χημικών Μηχανικών

Τομέας Ανάλυσης, Σχεδιασμού και

Ανάπτυξης Διεργασιών και Συστημάτων

**ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ :**

# ΔΙΕΡΕΥΝΗΣΗ ΜΗ ΓΡΑΜΜΙΚΩΝ ΦΑΙΝΟΜΕΝΩΝ ΜΕ ΧΡΗΣΗ ΜΕΘΟΔΩΝ ΜΗΧΑΝΙΚΗΣ ΜΑΘΗΣΗΣ

ΗΛΙΑΚΗΣ ΕΥΣΤΑΘΙΟΣ

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ: ΚΑΒΟΥΣΑΝΑΚΗΣ ΜΙΧΑΗΛ

ΙΟΥΛΙΟΣ 2023

# Ευχαριστίες

Με αυτή την διπλωματική κλείνει η πενταετής ‘’διαμονή’’ μου στη σχολή Χημικών Μηχανικών του Εθνικού Μετσόβιου Πολυτεχνείου και μου δίνεται το εισιτήριο να συνεχίσω τις σπουδές στο τμήμα Χημικών Μηχανικών του Τεχνολογικού Ινστιτούτου της Μασαχουσέτης, όπου θα εκπονήσω την διδακτορική μου διατριβή.

Μέσα σε αυτή την πενταετία, έμαθα τι σημαίνει να είναι κάποιος μηχανικός, αγάπησα την χημική μηχανική, αντιλήφθηκα τις συνεισφορές της στην εξέλιξη της ανθρωπότητας και συνειδητοποίησα πόσο ευτυχισμένο με κάνει αυτό το αντικείμενο. Χρωστάω ένα μεγάλο ευχαριστώ σε όλους τους καθηγητές που με ζήλο μου μεταλαμπάδευσαν την όρεξη και το μεράκι τους προς τις σπουδές αυτές. Εδώ, θα ήθελα να ευχαριστήσω τους καθηγητές Δώρο Θεοδώρου, Αντώνιο Κοκκόση και Μιχάλη Καβουσανάκη, οι οποίοι αποτέλεσαν και τους υποστηρικτές μου στην προσπάθεια μου να συνεχίσω ακαδημαϊκά στην απέναντι όχθη του Ατλαντικού.

Ιδιαίτερα, θα ήθελα να ευχαριστήσω τον καθηγητή Μιχάλη Καβουσανάκη που με εμπιστεύτηκε στην εκπόνηση αυτής της διπλωματικής. Η μηχανική μάθηση είναι ένα φλέγον ζήτημα, στο οποίο οι χημικοί μηχανικοί κάνουμε τα πρώτα μας βήματα και οι ευκαιρίες για έρευνα είναι πολλές. Υπό την καθοδήγηση και την άμεση βοήθειά του, καταφέρνω σήμερα να παρουσιάζω την συγκεκριμένη εργασία.

Επιπλέον, θα ήθελα να ευχαριστήσω τους συμφοιτητές και φίλους μου, που ήταν παρόντες σε κάθε μου βήμα και χωρίς την ύπαρξή τους, αυτά τα χρόνια θα ήταν άτονα και οι ομαδικές εργασίες μάλλον ακόμα ανεκπλήρωτες. Ανάμεσα στους τόσους υπέροχους ανθρώπους, θα ήθελα να τονίσω τα ονόματα της Βασιλικής Αγγάνη, της Βασιλικής – Μαρίας Κολυνδρίνη και του Αλέξανδρου Γιαννακόπουλου, που από την πρώτη μέρα είμαστε μαζί και ανακαλύψαμε παρέα τη μαγεία της χημικής μηχανικής. Θα είστε για πάντα η οικογένεια που βρήκα στην Αθήνα.

Έπειτα, νιώθω ανεκτίμητη ευγνωμοσύνη για τον κ. Αλέξανδρο Μποτό και την εταιρία ΡΟΥΣΣΑΣ Α.Ε. που από τα πρώτα χρόνια μου με στήριξαν οικονομικά, επαγγελματικά και συναισθηματικά. Χωρίς αυτούς, δεν θα είχα τις δυνάμεις να αγωνιστώ με τόσο πάθος και προσήλωση, όπως έκανα.

Τέλος, θα ήθελα να ευχαριστήσω την αδελφή μου, Μαρία Χριστίνα, τη μητέρα μου, Ευγενία, τον Παππού Τάκη και τη Γιαγιά Στέλλα για την καθημερινή εμψύχωση και την πίστη τους στις δυνάμεις μου.

# Περίληψη

Στη παρούσα διπλωματική εργασία παρουσιάζεται ο συνδυασμός μεθόδων μηχανικής μάθησης και συγκεκριμένα των διεργασιών Gauss (Gaussian processes) και των ρηχών τεχνητών νευρωνικών δικτύων (shallow artificial neural networks), με κλασικές μεθόδους αριθμητικής ανάλυσης (πεπερασμένες διαφορές και πεπερασμένα στοιχεία) με στόχο την εκμάθηση του δεξιού μέλους μιας μερικής διαφορικής εξίσωσης, δηλαδή τον υπολογισμό της χρονικής παραγώγου. Στόχος της εργασίας είναι να αποδείξει ότι διαδικασίες, όπως η εύρεση μονίμων καταστάσεων και πολλαπλότητας λύσεων καθώς και η μελέτη της δυναμικής εξέλιξης σε διάφορους χρόνους, είναι εφικτές ακόμα κι αν είναι άγνωστη στο χρήστη η διατύπωση της διαφορικής εξίσωσης, αρκεί να χρησιμοποιηθεί η συγκεκριμένη προσέγγιση (data-driven approach).

Συγκεκριμένα εξετάστηκε η ίδια διαφορική εξίσωση, το πρόβλημα Bratu σε μία και δύο διαστάσεις, διότι παρουσιάζει ενδιαφέρουσα / μη γραμμική συμπεριφορά ως προς μια παράμετρο που περιέχεται στον ορισμό της. Συγκεκριμένα, ο χώρος λύσεων της εξίσωσης Bratu χαρακτηρίζεται από πολλαπλότητα με διάστημα τιμών παραμέτρων όπου μπορούν να συνυπάρχουν ευσταθείς και ασταθείς λύσεις, και ένα διάστημα τιμών παραμέτρων όπου δεν υπολογίζεται λύση.

Εφαρμόστηκαν διεργασίες Gauss προκειμένου να βρεθούν ποιες από τις υπό εξέταση μεταβλητές επηρεάζουν τη χρονική παράγωγο της εξαρτημένης μεταβλητής τόσο στο μονοδιάστατο όσο και στο δισδιάστατο πρόβλημα. Οι διεργασίες κατάφεραν να εντοπίσουν με επιτυχία τις σημαντικές παραμέτρους με βάση τους συντελεστές που υπολογίζει η Automatic Relevance Determination (ARD) ανάλυση.

Έπειτα, για το μονοδιάστατο πρόβλημα, εκπαιδεύτηκε μοντέλο τύπου προς τα εμπρός τροφοδοσίας νευρωνικού δικτύου (Feed Forward Neural Network) με βάση δεδομένων 6.000 περίπου λύσεις. Για την επαλήθευση της ορθής εκπαίδευσης του νευρωνικού δικτύου, διερευνήθηκε η δυναμική συμπεριφορά επιβάλλοντας διαφορετικές αρχικές συνθήκες από αυτές που χρησιμοποιήθηκαν για παραγωγή δεδομένων. Το εκπαιδευμένο μοντέλο βρέθηκε να παρουσιάζει αποκλίσεις μικρότερες από 1 % ως προς τις λύσεις που προκύπτουν με τη μέθοδο των πεπερασμένων διαφορών. Επίσης, εξετάστηκε η εύρεση μόνιμης κατάστασης στις 3 περιοχές σημασίας (ευσταθή και ασταθή λύση, καθώς και κοντά στην κρίσιμη τιμή της παραμέτρου) με τη μέθοδο Newton - GMRES, που επίσης η απόκλιση από τη λύση είναι μικρότερη του 1%. Τέλος, πραγματοποιήθηκε και παραμετρική ανάλυση συνδυάζοντας τις μεθόδους με εκείνη του Keller (Pseudo – Arc length Continuation), η οποία χρησιμοποιήθηκε για τον εντοπισμό της κρίσιμης παραμέτρου $\lambda_{critical}$ .Σε σχέση με τη μέθοδο των πεπερασμένων στοιχείων (FEM), η διαφορά είναι της τάξης του 0.01%, τονίζοντας την επιτυχία του μοντέλου.

Για το 2D πρόβλημα εκπαιδεύτηκε νευρωνικό δίκτυο ίδιου τύπου με εκείνο του 1D αλλά με περίπου 18000 λύσεις για εκπαίδευση. Δοκιμάστηκε στις ίδιες πρακτικές με την απόδοση του να είναι κοντά στο 98% στο μεγαλύτερο εύρος τιμών της παραμέτρου, αποδεικνύοντας την αξία των νευρωνικών δικτύων στην επίλυση και μελέτη των διαφορικών εξισώσεων.

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: Μερικές διαφορικές εξισώσεις, επιλύτες διαφορικών εξισώσεων, διεργασίες Gauss, τεχνητά νευρωνικά δίκτυα, μεταβατικό πρόβλημα, πρόβλημα μόνιμης κατάστασης, αριθμητική ανάλυση, παραμετρική ανάλυση, εύρεση πολλαπλών λύσεων, πρόβλημα Bratu

# Abstract ("Study Of Nonlinear Phenomena Using Machine Learning Methods")

This thesis presents the co-implementation of machine learning methods, specifically Gaussian processes, and shallow artificial neural networks, with classical methods of numerical analysis (finite differences and finite elements) with the aim of learning the right-hand side of a partial differential equation, i.e., calculating the time derivative. This work purposes to prove that procedures such as finding steady states and multiplicity of solutions as well as studying the dynamic evolution in various times are possible, even if the formulation of the differential equation is unknown to the user, as long as the specific data-driven approach is applied.

Specifically, we examined the Bratu partial differential equation in one and two dimensions, because it presents interesting/non-linear behavior. In particular, one can find a parametric region with solution multiplicity (co-existence of dynamically stable and unstable solutions), and a region where no solution can be calculated.

Gaussian processes were applied in order to find which of the variables under consideration influence the time derivative in both the 1D and 2D problem. The processes are able to successfully identify the important variables based on the coefficients calculated by the Automatic Relevance Determination (ARD) analysis.

Then, for the one-dimensional problem, a Feed Forward Neural Network model was trained on a database of about 6,000 solutions. We tested the validity of our trained network by computing its dynamic behavior imposing different initial conditions than those used for data generation. The trained model was found to show deviations of less than 1 % with respect to the solutions obtained by the finite difference method. Also, it was used for calculating the steady state in three examples (a stable and an unstable solution, as well as one near the critical value of the parameter λ) with the Newton - GMRES method. Once again, the deviations from the FEM-based solution are less than 1%. Finally, a parametric analysis was carried out combining the methods with that of Keller (Pseudo – Arc length Continuation), which was used to identify the critical parameter $\lambda_{critical}$ . The value was compared to the FEM-based critical value and the difference is of the order of 0.01%, highlighting the success of the model.

For the 2D problem, a neural network of the same type as the 1D one but with about 18000 solutions was trained. It was tested in the same practices with its performance close to 98%, over the largest range of parameter λ values, proving the significance of neural networks in solving and studying differential equations.

KEY WORDS: Partial differential equations, ode solvers, Gaussian processes, artificial neural networks, transient problem, steady state problem, numerical analysis, parametric analysis, finding multiple solutions, Bratu problem

# Περιεχόμενα

# Κατάλογος Σχημάτων

# Κατάλογος Πινάκων

# 1. Εισαγωγή

## 1.1. Διαφορικές Εξισώσεις

Στον τομέα των μαθηματικών, οι διαφορικές εξισώσεις διαδραματίζουν κρίσιμο ρόλο στη μοντελοποίηση και την κατανόηση διαφόρων φαινομένων στην φυσική και γενικότερα στις φυσικές επιστήμες. Εξάλλου, μαθηματικό πρότυπο ή μοντέλο ενός φυσικού φαινομένου είναι η διαφορική εξίσωση που το περιγράφει. Έτσι, και στη χημική μηχανική είναι κυρίαρχη η παρουσία τους στην περιγραφή θεμελιωδών εννοιών με σημαντικά παραδείγματα την διαφορική εξίσωση των ισοζυγίων μάζας και ενέργειας ενός συστήματος όπως και την εξίσωση διάδοσης θερμότητας. Συνεπώς, αυτή η ενότητα εμβαθύνει στις θεμελιώδεις έννοιες και τεχνικές που σχετίζονται με τις διαφορικές εξισώσεις, οι οποίες παρέχουν μια σταθερή βάση για την αντιμετώπιση προβλημάτων του πραγματικού κόσμου. Γίνεται μια αναφορά σε βασικούς ορισμούς έως προηγμένες μεθόδους επίλυσης, καθώς αυτός ο κλάδος αποτελεί την θεμέλιο λίθο της διπλωματικής αυτής [1-2].

### 1.1.1. Ορισμοί και είδη διαφορικών εξισώσεων

Μια διαφορική εξίσωση ορίζεται ως μια εξίσωση που συσχετίζει μια άγνωστη συνάρτηση με τις παραγώγους της. Αντιπροσωπεύει μια σχέση μεταξύ μιας συνάρτησης και των ρυθμών μεταβολής της, επιτρέποντάς την περιγραφή της εξέλιξης της συνάρτησης με την πάροδο του χρόνου, αν είναι χρονικά εξαρτώμενη ή με την αλλαγή των συντεταγμένων στο χώρο κ.ο.κ. . Οι διαφορικές εξισώσεις μπορούν να ταξινομηθούν με βάση διάφορους παράγοντες:

**Συνήθεις και Μερικές Διαφορικές εξισώσεις:** Όλες οι διαφορικές εξισώσεις ανήκουν σε μια από αυτές τις δύο κατηγορίες. Αν η συνάρτηση της διαφορικής είναι μιας μεταβλητής και οι παράγωγοι είναι συνήθεις, τότε η διαφορική καλείται συνήθης. Διαφορετικά, αν η συνάρτηση ορίζεται από δύο και άνω ανεξάρτητες μεταβλητές και υπάρχουν μερικές παράγωγοι στην εξίσωση, αυτή καλείται μερική [2].

Για παράδειγμα, η εξίσωση μιας μονόδρομης στοιχειώδους χημικής αντίδρασης δεύτερης τάξης (1.1) είναι συνήθης διαφορική, καθώς η συγκέντρωση (εξαρτημένη μεταβλητή) εξαρτάται μόνο από τον χρόνο (ανεξάρτητη) και έτσι υπάρχει μόνο συνήθης παράγωγος στην εξίσωση.

$$\frac{dc_A}{dt} = kc_A^2, \tag{1.1}$$

όπου: $c_A$: η συγκέντρωση του προϊόντος Α και, $k$:σταθερά της αντίδρασης,

Ενώ η εξίσωση Laplace (1.2) αποτελεί μερική διαφορική εξίσωση, αφού ο λαπλασιανός τελεστής είναι το άθροισμα των δευτέρων μερικών παραγώγων ως προς τις χωρικές συντεταγμένες. Ειδικά, αν οι συντεταγμένες είναι καρτεσιανές σε δύο διαστάσεις., η εξίσωση λαμβάνει τη μορφή:

$$\nabla^2 T = 0\ (2) \rightarrow \frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial^2 y} = 0. \tag{1.2}$$

**Συστήματα διαφορικών εξισώσεων:** Αν μια συνάρτηση είναι αυτή που πρέπει να προσδιορισθεί τότε μια εξίσωση είναι αρκετή. Αν πρέπει να προσδιοριστούν περισσότερες, τότε απαιτείται σύστημα διαφορικών εξισώσεων. Παράδειγμα συστήματος είναι όταν μια αντίδραση αποτελείται από

ενδιάμεσες αντιδράσεις με την παραγωγή ενδιάμεσων προϊόντων και πρέπει να προσδιοριστούν όλες οι συγκεντρώσεις συναρτήσει του χρόνου [4].

**Τάξη**: Η τάξη μιας διαφορικής εξίσωσης καθορίζεται από την υψηλότερης τάξης παράγωγο που υπάρχει στην εξίσωση. Για παράδειγμα, μια διαφορική εξίσωση που περιλαμβάνει μόνο την πρώτη παράγωγο λέγεται ότι είναι πρώτης τάξης, ενώ αν υπάρχει και τρίτη παράγωγος καλείται τρίτης τάξης κ.ο.κ. [3].

**Γραμμικότητα**: Μια διαφορική εξίσωση είναι γραμμική εάν η άγνωστη συνάρτηση και οι παράγωγοί της εμφανίζονται γραμμικά, χωρίς γινόμενα ή μη γραμμικές συναρτήσεις. Διαφορετικά, θεωρείται μη γραμμική [4].

**Ομοιογένεια:** Μια ομοιογενής διαφορική εξίσωση είναι αυτή όπου όλοι οι όροι που περιλαμβάνουν την άγνωστη συνάρτηση και τις παραγώγους της έχουν την ίδια ισχύ, δηλαδή δεν υπάρχει όρος που να μην περιέχει την άγνωστη συνάρτηση σε κάποια μορφή της. Διαφορετικά, είναι μη ομοιογενής [5].

**Συντελεστές**: Οι συντελεστές μιας διαφορικής εξίσωσης είναι οι σταθερές ή οι συναρτήσεις που πολλαπλασιάζουν τις παραγώγους. Αυτοί οι συντελεστές μπορεί να είναι σταθεροί ή μεταβλητοί [1].

Η κατανόηση αυτών των ταξινομήσεων παρέχει ένα πλαίσιο για την ανάλυση και την επίλυση διαφορικών εξισώσεων. Τέλος, αξίζει να αναφερθεί και η έννοια του προβλήματος συνοριακών τιμών.

**Ένα πρόβλημα συνοριακών τιμών** συνίσταται στην εύρεση της λύσης μιας διαφορικής εξίσωσης που ικανοποιεί κατάλληλες συνοριακές συνθήκες, οι οποίες εφαρμόζονται στο σύνορο του πεδίου ορισμού της λύσης [1].

### 1.1.2. Επίλυση διαφορικών εξισώσεων

Η επίλυση διαφορικών εξισώσεων περιλαμβάνει την εύρεση της άγνωστης συνάρτησης που ικανοποιεί την εξίσωση. Διάφορες μέθοδοι και τεχνικές έχουν αναπτυχθεί όλα αυτά τα χρόνια. Σε αυτή την υποενότητα, παρουσιάζονται μερικές κοινές προσεγγίσεις:

**Διαχωρισμός μεταβλητών**: Αυτή η μέθοδος εφαρμόζεται σε συνήθεις διαφορικές εξισώσεις πρώτης τάξης που μπορούν να γραφτούν με τη μορφή $dy/dx = f(x)g(y)$. Διαχωρίζοντας τις μεταβλητές x και y, μπορούν να ολοκληρωθούν οι δύο πλευρές χωριστά και να ληφθεί η λύση [4].

Επίσης, στις μερικές διαφορικές εξισώσεις, η εξαρτημένη μεταβλητή θεωρείται γινόμενο τόσων συναρτήσεων όσων το πλήθος των ανεξάρτητων μεταβλητών, οι οποίες είναι συναρτήσεις μιας μεταβλητής [1].

**Ομογενείς εξισώσεις:** Οι ομογενείς διαφορικές εξισώσεις μπορούν να λυθούν χρησιμοποιώντας μια αντικατάσταση που μετατρέπει την εξίσωση σε χωριστή μορφή. Για παράδειγμα, αντικαθιστώντας το $y = vx$, όπου v είναι μια νέα μεταβλητή, μπορεί να αναχθεί μια ομογενής εξίσωση δεύτερης τάξης σε μια διαχωρίσιμη εξίσωση πρώτης τάξης [2].

**Συντελεστές ολοκλήρωσης:** Ορισμένες γραμμικές διαφορικές εξισώσεις μπορούν να λυθούν χρησιμοποιώντας έναν συντελεστή ολοκλήρωσης. Αυτός ο παράγοντας καθορίζεται πολλαπλασιάζοντας ολόκληρη την εξίσωση για να γίνει ακριβής, επιτρέποντας την ολοκλήρωση και τη λύση [2].

**Δυναμοσειρές και γενικότερα σειρές :** Οι μέθοδοι σειράς χρησιμοποιούνται όταν η διαφορική εξίσωση δεν μπορεί να λυθεί μέσω τυπικών τεχνικών. Υποθέτοντας ότι η λύση μπορεί να εκφραστεί ως σειρά απείρων όρων, μπορεί να λυθεί η εξίσωση όρο προς όρο, προσδιορίζοντας τους συντελεστές που ικανοποιούν την εξίσωση. Χαρακτηριστικά παραδείγματα αποτελούν οι σειρές Fourier και Chebyshev, που βρίσκουν εφαρμογή στην επίλυση μερικών διαφορικών εξισώσεων [3].

Αυτές είναι μόνο μερικές από τις πολλές διαθέσιμες τεχνικές για την επίλυση διαφορικών εξισώσεων και κάθε μέθοδος έχει τα δικά της πλεονεκτήματα και περιορισμούς. Η επιλογή της κατάλληλης μεθόδου εξαρτάται από τα ειδικά χαρακτηριστικά της δεδομένης εξίσωσης.

### 1.1.3. Εφαρμογές Διαφορικών Εξισώσεων

Οι διαφορικές εξισώσεις βρίσκουν εκτεταμένες εφαρμογές σε ένα ευρύ φάσμα επιστημονικών κλάδων. Χρησιμεύουν ως ισχυρά εργαλεία για τη μοντελοποίηση και την πρόβλεψη φαινομένων σε διάφορους τομείς. Μερικές αξιόλογες εφαρμογές περιλαμβάνουν:

**Φυσική:** Οι διαφορικές εξισώσεις χρησιμοποιούνται ευρέως στην κλασική μηχανική, την κβαντομηχανική, τον ηλεκτρομαγνητισμό και τη δυναμική των ρευστών για να περιγράψουν τη συμπεριφορά των φυσικών συστημάτων. Παρέχουν πληροφορίες για την κίνηση των σωματιδίων, τη διάδοση των κυμάτων και τη ροή ρευστού [1].

**Μηχανική:** Οι μηχανικοί βασίζονται σε διαφορικές εξισώσεις για να αναλύσουν και να σχεδιάσουν συστήματα όπως ηλεκτρικά κυκλώματα, συστήματα ελέγχου και δομική μηχανική. Επιτρέπουν την κατανόηση της συμπεριφοράς του συστήματος και τη βελτιστοποίηση των διαδικασιών μηχανικής [2].

**Βιολογία:** Οι διαφορικές εξισώσεις βοηθούν στη μοντελοποίηση βιολογικών συστημάτων όπως η δυναμική του πληθυσμού, οι βιοχημικές αντιδράσεις και τα νευρωνικά δίκτυα. Βοηθούν στην πρόβλεψη της ανάπτυξης και της αλληλεπίδρασης των οργανισμών και παρέχουν πολύτιμες γνώσεις για τις βιολογικές διεργασίες [1].

**Οικονομικά:** Χρησιμοποιούνται διαφορικές εξισώσεις για τη μοντελοποίηση οικονομικών φαινομένων όπως η δυναμική της προσφοράς και της ζήτησης, η οικονομική ανάπτυξη και η συμπεριφορά της χρηματοπιστωτικής αγοράς. Βοηθούν στην κατανόηση πολύπλοκων οικονομικών συστημάτων και στην πραγματοποίηση προβλέψεων [5].

**Χημική Μηχανική:** Όλα τα ισοζύγια (ενέργειας, μάζας, δυνάμεων κ.λπ..) που αποτελούν τον πυρήνα της χημικής μηχανικής, διατυπώνονται στην διαφορική τους μορφή. Έτσι, οι διαφορικές εξισώσεις είναι ισχυρό εργαλείο στα χέρια του χημικού μηχανικού για την μελέτη της ρευστομηχανικής, της βιοχημικής μηχανικής, της μοντελοποίησης αντιδραστήρων και γενικότερα μονάδων λειτουργίας.[1]

Αυτά είναι μόνο μερικά παραδείγματα από τις πολλές εφαρμογές των διαφορικών εξισώσεων. Έχουν εκτεταμένες εφαρμογές σε τομείς όπως η χημεία, η περιβαλλοντική επιστήμη, η επιστήμη των υπολογιστών και άλλες.

## 1.2. Εξίσωση Bratu

Το πρόβλημα Bratu είναι ένα κλασικό πρόβλημα συνοριακών τιμών με εφαρμογές στα μαθηματικά και τη φυσική και λειτουργεί ως βάση για την εξέταση και τον έλεγχο αριθμητικών μεθόδων και αλγορίθμων. Περιλαμβάνει την επίλυση μιας δεύτερης τάξης μη γραμμικής διαφορικής εξίσωσης, η οποία υπόκειται σε συνοριακές συνθήκες [6].

### 1.2.1. Διατύπωση του Προβλήματος

Το πρόβλημα Bratu ορίζεται ως εξής:

$$\Delta u + \lambda e^{u} = 0, u \in \Omega$$

$$u = 0\,, u \in \partial\Omega\,.$$

Ενώ η αναλυτική λύση που προτείνεται από τη βιβλιογραφία [6] για το μονοδιάστατο πρόβλημα είναι η:

$$u(x) = -2\ln\left[\frac{\cosh((x-\frac{1}{2}) * \frac{\theta}{2}))}{\cosh\left(\frac{\theta}{4}\right)}\right],$$

όπου:

$$\theta = \sqrt{2\lambda}\cosh\left(\frac{\theta}{4}\right),$$

$$1 = \sqrt{2\lambda_{critical}}\sinh\left(\frac{\theta}{4}\right) * \frac{1}{4}.$$

Όπου : u: η άγνωστη συνάρτηση, λ: μια παράμετρος, Ω το πεδίο ορισμού (για το μονοδιάστατο πρόβλημα (0,1) και για το δισδιάστατο (0,1)x(0,1)) και ∂Ω το σύνορο (x=0,x=1 στο 1D, x=0,y=0, x=1,y=1 στο 2D).

Παρατηρείται ότι η εξίσωση περιλαμβάνει ένα μη γραμμικό εκθετικό όρο και τον Λαπλασιανό τελεστή. ($\Delta = \nabla^2$ ).

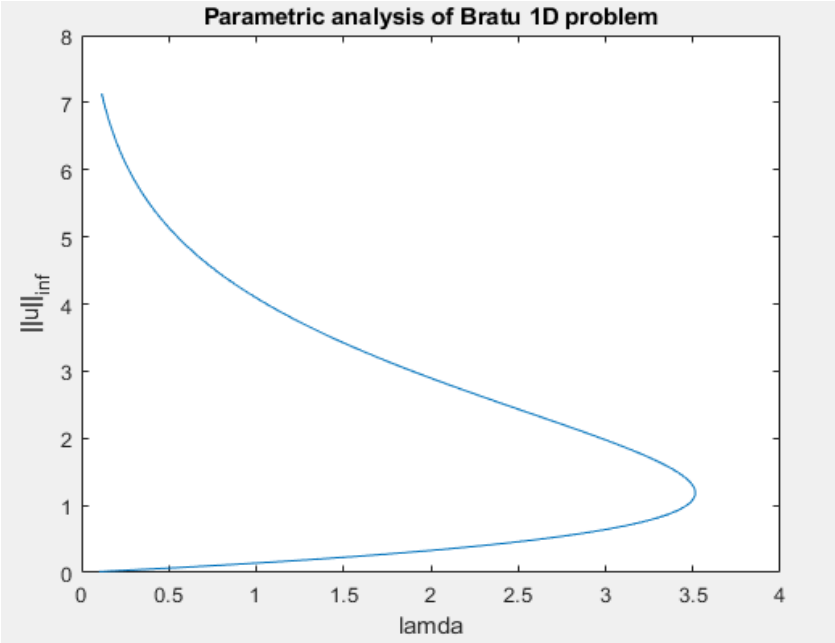
### 1.2.2. Μη γραμμική συμπεριφορά του προβλήματος Bratu

Το πρόβλημα αυτό παρουσιάζει κάποιες πολύ ενδιαφέρουσες μαθηματικές προκλήσεις και ιδιότητες:

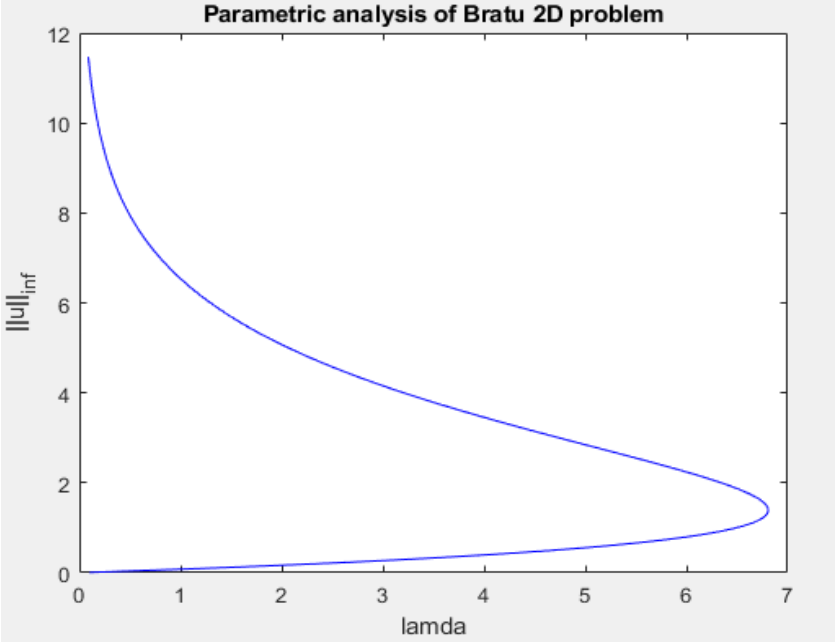
- Η ύπαρξη του εκθετικού, μη-γραμμικού όρου καθιστά πολύ δύσκολη την αναλυτική επίλυση της εξίσωσης, η οποία απαιτεί και αυτή κάποια επαναληπτική μέθοδο για να επιλυθεί [6]. Έτσι, όπως γίνεται άμεσα αντιληπτό, η χρήση αριθμητικών μεθόδων είναι σχεδόν αναπόφευκτη.
- Στην συμπεριφορά της λύσης, φαίνεται ότι η παράμετρος λ παίζει πολύ σημαντικό ρόλο. Για τιμές παραμέτρων που υπερβαίνουν μια κρίσιμη τιμή της παραμέτρου ($\lambda_c$), η εξίσωση δεν έχει λύση. Για λ= $\lambda_c$ , η εξίσωση έχει μια ακριβής (οριακά) ευσταθή λύση. Για λ< $\lambda_c$, η εξίσωση έχει δύο λύσεις, μια ευσταθή και μια ασταθή. Επομένως, σχηματίζονται δύο κλάδοι λύσεων, ένας ευσταθής και ένας ασταθής, οι οποίοι ενώνονται στην κρίσιμη τιμή της παραμέτρου. Και επειδή στο σημείο αυτό κατά την σχεδίαση του γραφήματος της παραμετρικής ανάλυσης

φαίνεται να ‘’διπλώνει’’ η γραφική παράσταση στο κρίσιμο σημείο, αυτό καλείται σημείο μονής δίπλωσης (single-fold point) [7].

Οι ακόλουθες ιδιότητες παρατηρούνται και στα Σχήματα 1-1 και 1-2 , όπου παρουσιάζεται η παραμετρική ανάλυση του μονοδιάστατου και του δισδιάστατου προβλήματος Bratu, αντίστοιχα.




*Σχήμα 1-1. Παραμετρική ανάλυση του μονοδιάστατου προβλήματος Bratu.*




*Σχήμα 1-2. Παραμετρική ανάλυση του δισδιάστατου προβλήματος Bratu.*

### 1.2.3. Αριθμητικές Προσεγγίσεις

Για την επίλυση του προβλήματος έχουν χρησιμοποιηθεί πολλές και διαφορετικές τεχνικές. Σε αυτές περιλαμβάνονται η μέθοδος πεπερασμένων διαφορών, η μέθοδος πεπερασμένων στοιχείων και οι φασματικές μέθοδοι. Όλες οι τεχνικές διακριτοποιούν το πεδίο ορισμού, προσεγγίζουν την λύση, επιλύοντας το σύστημα αλγεβρικών εξισώσεων που προκύπτει. Ωστόσο, επειδή το σύστημα είναι μη γραμμικό, απαιτείται η ενσωμάτωση μιας επαναληπτικής μεθόδου στον αλγόριθμο επίλυσης [8-10].

### 1.2.4. Εφαρμογές του Προβλήματος Bratu

Το πρόβλημα Bratu έχει βρει εφαρμογές σε διάφορους επιστημονικούς και μηχανικούς κλάδους. Μερικές αξιόλογες εφαρμογές περιλαμβάνουν :

**Μη Γραμμική Φυσική**

Το πρόβλημα Bratu χρησιμεύει ως πρωτότυπο για τη μελέτη μη γραμμικών φαινομένων στη φυσική. Παρέχει πληροφορίες για τη συμπεριφορά των μη γραμμικών εξισώσεων, τις μεταβάσεις φάσης και το σχηματισμό προτύπων. Γίνεται λόγος ότι η εξίσωση βρίσκει εφαρμογή στο μοντέλο Chandrasekhar για την έκταση του σύμπαντος [6].

**Επιστήμη των Υλικών**

Στην επιστήμη των υλικών, το πρόβλημα Bratu έχει χρησιμοποιηθεί για τη μοντελοποίηση φαινομένων όπως οι μεταβάσεις φάσης, η διάχυση και οι διαδικασίες αντίδρασης-διάχυσης. Ενσωματώνοντας συγκεκριμένες ιδιότητες υλικού στη διατύπωση του προβλήματος, οι ερευνητές μπορούν να προσομοιώσουν και να αναλύσουν τη συμπεριφορά των υλικών κάτω από διαφορετικές συνθήκες (μεταφορά θερμότητας με ραδιενέργεια, θερμικές αντιδράσεις) [6,11].

**Καύση και Χημική Μηχανική**

Το πρόβλημα Bratu έχει επίσης βρει εφαρμογές στην καύση. Συγκεκριμένα, χρησιμοποιείται για τη μοντελοποίηση της καύσης στερεής βιομάζας, συνδέοντας την εξίσωση αυτή στενά με τη χημική μηχανική [12-13].

## 1.3 Μηχανική Μάθηση

### 1.3.1. Κατηγορίες

Οι τεχνικές μηχανικής μάθησης περιλαμβάνουν ένα ευρύ φάσμα αλγορίθμων και προσεγγίσεων. Αυτές οι τεχνικές, που υποστηρίζονται από ισχυρά μαθηματικά θεμέλια, έχουν φέρει επανάσταση σε διάφορους τομείς, συμπεριλαμβανομένης της επεξεργασίας φυσικής γλώσσας και της ρομποτικής. Κατανοώντας τις αρχές και τις εφαρμογές της μηχανικής μάθησης, οι ερευνητές και οι επαγγελματίες μπορούν να αξιοποιήσουν τη δύναμή της για την επίλυση σύνθετων προβλημάτων του πραγματικού κόσμου.

Η μηχανική μάθηση έχει φέρει επανάσταση στον τομέα της τεχνητής νοημοσύνης δίνοντας τη δυνατότητα στους υπολογιστές να μαθαίνουν από δεδομένα και να λαμβάνουν αποφάσεις χωρίς την

παρέμβαση κάποιου προγραμματιστή. Έτσι, κατά την μελέτη της μηχανικής μάθησης, πρέπει πρώτα να γίνει μια αναφορά στις θεμελιώδεις έννοιες και τις δημοφιλείς τεχνικές που χρησιμοποιούνται σε αυτή.

#### *1.3.1.1. Εποπτευόμενη μάθηση (Supervised Learning)*

Η εποπτευόμενη μάθηση είναι μια ευρέως χρησιμοποιούμενη προσέγγιση στη μηχανική μάθηση, όπου τα μοντέλα εκπαιδεύονται σε δεδομένα με ετικέτα για να κάνουν προβλέψεις ή να ταξινομούν νέες, μη εμφανείς περιπτώσεις. Δύο χαρακτηριστικές τεχνικές στην εποπτευόμενη μάθηση είναι:

Δένδρα αποφάσεων: Τα δέντρα αποφάσεων είναι ιεραρχικές δομές που διαχωρίζουν τα δεδομένα με βάση διαφορετικά χαρακτηριστικά, δημιουργώντας ένα διάγραμμα ροής αποφάσεων που μοιάζει με δέντρο. Κάθε εσωτερικός κόμβος αντιπροσωπεύει μια δοκιμή σε ένα χαρακτηριστικό και κάθε κόμβος φύλλου αντιπροσωπεύει μια ετικέτα κλάσης ή μια πρόβλεψη. Επέκταση αυτής της μεθόδου είναι και τα τυχαία δάση (random forests) [14].

Διανυσματικές μηχανές υποστήριξης (Support vector machine, SVM): Τα SVM στοχεύουν να βρουν ένα βέλτιστο υπερεπίπεδο που διαχωρίζει τα δεδομένα σε διαφορετικές κλάσεις. Μεγιστοποιώντας το περιθώριο μεταξύ των κλάσεων, τα SVM παρέχουν ισχυρές δυνατότητες ταξινόμησης [14].

#### *1.3.1.2. Μη εποπτευόμενη μάθηση (Unsupervised Learning)*

Η μάθηση χωρίς επίβλεψη περιλαμβάνει την εξαγωγή μοτίβων ή δομών από δεδομένα χωρίς ετικέτα. Αυτός ο τύπος μάθησης είναι χρήσιμος όταν δεν υπάρχει προκαθορισμένο αποτέλεσμα που να καθοδηγεί τη μαθησιακή διαδικασία. Δύο δημοφιλείς τεχνικές μάθησης χωρίς επίβλεψη είναι:

Ομαδοποίηση / Clustering: Οι αλγόριθμοι ομαδοποίησης ομαδοποιούν παρόμοια σημεία δεδομένων με βάση τις εγγενείς ομοιότητες ή αποστάσεις τους. Η ομαδοποίηση K-means και η ιεραρχική ομαδοποίηση είναι μέθοδοι που χρησιμοποιούνται ευρέως σε αυτήν την κατηγορία [15].

Μείωση διαστάσεων / Dimensionality Reduction : Οι τεχνικές μείωσης διαστάσεων στοχεύουν στη μείωση του αριθμού των χαρακτηριστικών σε ένα σύνολο δεδομένων, διατηρώντας παράλληλα τις βασικές πληροφορίες. Η ανάλυση κύριου στοιχείου (PCA) και η t-SNE (t-Distributed Stochastic Neighbor Embedding) είναι κοινώς χρησιμοποιούμενες τεχνικές μείωσης διαστάσεων, ενώ αρκετά δημοφιλής είναι και η Χαρτογράφηση Διάχυσης (Diffusion Maps), που είναι μια μη γραμμική τεχνική που χρησιμοποιεί χαρακτηριστικά ευκλείδειας γεωμετρίας σε χώρους τύπου manifold [15].

#### *1.3.1.3. Βαθιά Μάθηση (Deep Learning)*

Η βαθιά μάθηση  εστιάζει στην εκπαίδευση σε βαθιά νευρωνικά δίκτυα με πολλαπλά επίπεδα για την εκμάθηση ιεραρχικών αναπαραστάσεων δεδομένων. Μερικές βασικές τεχνικές στη βαθιά μάθηση είναι:

Συνελικτικά νευρωνικά δίκτυα / Convolutional Neural Networks (CNN): Τα CNN χρησιμοποιούνται κυρίως για εργασίες ταξινόμησης εικόνων. Χρησιμοποιούν συνελικτικά επίπεδα για να μαθαίνουν αυτόματα τοπικά μοτίβα και ιεραρχικές αναπαραστάσεις από δεδομένα εικόνας [16].

Επαναλαμβανόμενα νευρωνικά δίκτυα / Recurrent Neural Networks (RNN): Τα RNN έχουν σχεδιαστεί για να χειρίζονται διαδοχικά δεδομένα, όπως κείμενο ή χρονοσειρές. Έχουν επαναλαμβανόμενες

συνδέσεις που επιτρέπουν στις πληροφορίες να διατηρούνται στα χρονικά βήματα, επιτρέποντάς τους να μοντελοποιούν εξαρτήσεις και να καταγράφουν τη χρονική δυναμική [16].

#### *1.3.1.4 Ενισχυτική Μάθηση (Reinforcement Learning)*

Η ενισχυτική μάθηση περιλαμβάνει την εκπαίδευση ενός πράκτορα (agent), ώστε να αλληλεπιδρά με ένα περιβάλλον και να μαθαίνει τις βέλτιστες ενέργειες μέσω δοκιμής και σφάλματος. Μαθαίνει από τις ανταμοιβές ή τις ποινές που λαμβάνει με βάση τις πράξεις του. Δύο αξιόλογες τεχνικές στην ενισχυτική μάθηση είναι:

Q-Learning: Ο Q-Learning είναι ένας αλγόριθμος ενισχυτικής μάθησης χωρίς μοντέλα που μαθαίνει μια συνάρτηση τιμής δράσης, γνωστή ως Q-values. Εξερευνά το περιβάλλον, ενημερώνει τις τιμές Q με βάση τις παρατηρούμενες ανταμοιβές και σταδιακά συγκλίνει σε μια βέλτιστη πολιτική [17].

Deep Q-Networks (DQN): Το DQN συνδυάζει βαθιά νευρωνικά δίκτυα με Q-Learning, επιτρέποντας πιο πολύπλοκους και υψηλών διαστάσεων χώρους καταστάσεων. Χρησιμοποιεί ένα νευρωνικό δίκτυο για την προσέγγιση των τιμών Q, επιτρέποντας πιο αποτελεσματική και ακριβή μάθηση [17].

### 1.3.2. Νευρωνικά Δίκτυα

Τα νευρωνικά δίκτυα αποτελούν θεμελιώδες δομικό στοιχείο της σύγχρονης μηχανικής μάθησης και της τεχνητής νοημοσύνης. Εμπνευσμένα από τη δομή και τη λειτουργία του ανθρώπινου εγκεφάλου, τα νευρωνικά δίκτυα προσομοιώνουν πολύπλοκα διασυνδεδεμένα δίκτυα τεχνητών νευρώνων για να μάθουν και να κάνουν προβλέψεις. Παρουσιάζονται τα θεμέλια, η αρχιτεκτονική και οι αλγόριθμοι εκπαίδευσης των νευρωνικών δικτύων, επισημαίνοντας την ευελιξία και τις εφαρμογές τους.

#### *1.3.2.1. Θεμελιώσεις Νευρωνικών Δικτύων*

Τα νευρωνικά δίκτυα αποτελούνται από διασυνδεδεμένους τεχνητούς νευρώνες, γνωστούς και ως κόμβους ή μονάδες, οργανωμένους σε επίπεδα. Κάθε νευρώνας λαμβάνει εισόδους, τις επεξεργάζεται χρησιμοποιώντας μια συνάρτηση ενεργοποίησης και παράγει μια έξοδο. Οι βασικές έννοιες στα νευρωνικά δίκτυα περιλαμβάνουν:

1. Δίκτυα τροφοδοσίας: Τα νευρωνικά δίκτυα προώθησης (Feed Forward Neural Networks, FFNN) είναι ο απλούστερος τύπος, όπου οι πληροφορίες ρέουν προς μία κατεύθυνση, από το επίπεδο εισόδου μέσω των κρυφών στρωμάτων στο επίπεδο εξόδου. Είναι αποτελεσματικά σε εργασίες όπως η ταξινόμηση και η παλινδρόμηση [16].

2. Συναρτήσεις ενεργοποίησης: Οι συναρτήσεις ενεργοποίησης εισάγουν τη μη γραμμικότητα στο νευρωνικό δίκτυο, επιτρέποντάς του να μοντελοποιεί περίπλοκες σχέσεις. Οι δημοφιλείς συναρτήσεις ενεργοποίησης περιλαμβάνουν τις συναρτήσεις σιγμοειδούς (sigmoids) , ReLU (Διορθωμένη Γραμμική Μονάδα) και tanh (υπερβολική εφαπτομένη) [16].

#### *1.3.2.2. Αρχιτεκτονικές Νευρωνικών Δικτύων*

Τα νευρωνικά δίκτυα μπορούν να έχουν διάφορες αρχιτεκτονικές, καθεμία κατάλληλη για διαφορετικούς τύπους εργασιών και δεδομένων. Ενδεικτικές είναι οι ακόλουθες αρχιτεκτονικές:

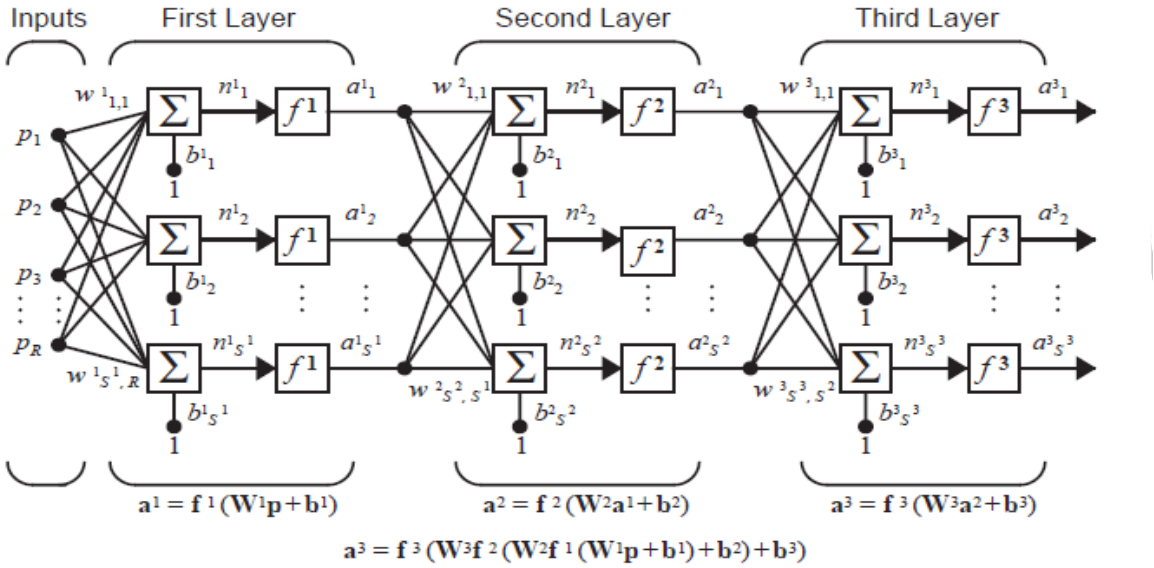
1. Multilayer Perceptron (MLP): Το MLP είναι μια κλασική αρχιτεκτονική νευρωνικών δικτύων προώθησης με πολλαπλά κρυφά στρώματα μεταξύ των επιπέδων εισόδου και εξόδου. Χρησιμοποιείται ευρέως για εργασίες ταξινόμησης και παλινδρόμησης γενικής χρήσης [16].

2. Συνελικτικά νευρωνικά δίκτυα (CNN): Τα CNN έχουν σχεδιαστεί για την επεξεργασία δεδομένων που μοιάζουν με πλέγμα, όπως εικόνες. Χρησιμοποιούν συνελικτικά επίπεδα για να μαθαίνουν αυτόματα τοπικά μοτίβα και ιεραρχικές αναπαραστάσεις, καθιστώντας τα εξαιρετικά αποτελεσματικά σε εργασίες computer vision [16].

#### *1.3.2.3. Εκπαίδευση Νευρωνικών Δικτύων Τύπου FFNN*

Τα νευρωνικά δίκτυα προώθησης είναι μια θεμελιώδης κατηγορία τεχνητών νευρωνικών δικτύων που χρησιμοποιούνται ευρέως σε διάφορους τομείς, συμπεριλαμβανομένης της μηχανικής μάθησης, της αναγνώρισης προτύπων και της βαθιάς μάθησης.

Τα νευρωνικά δίκτυα τροφοδοσίας αποτελούνται από πολλαπλά στρώματα διασυνδεδεμένων κόμβων, γνωστών ως νευρώνες. Οι πληροφορίες ρέουν μέσω του δικτύου από το επίπεδο εισόδου, μέσω ενός ή περισσότερων κρυφών επιπέδων, στο επίπεδο εξόδου, χωρίς καμία σύνδεση ανάδρασης. Κάθε νευρώνας λαμβάνει εισόδους από το προηγούμενο στρώμα, εφαρμόζει μια συνάρτηση ενεργοποίησης και περνά την έξοδο στο επόμενο στρώμα. Τα βάρη και οι βάσεις του δικτύου υπολογίζονται κατά τη διάρκεια της διαδικασίας εκπαίδευσης τους δικτύου. Ένα τυπικό δίκτυο τύπου FFNN παρουσιάζεται στο Σχήμα 1-3.




*Σχήμα 1-3. Τυπική δομή νευρωνικού δικτύου προωθούμενης τροφοδοσίας 3 επιπέδων [18]*

Η πληροφορία κινείται μόνο προς τα εμπρός, περνώντας μέσα από τα κρυφά στρώματα, το σήμα αλλάζει μέσα από τις συναρτήσεις ενεργοποίησης του κάθε επιπέδου και τελικά λαμβάνεται το

επιθυμητό σήμα εξόδου. Εδώ, αναλύεται το διάνυσμα εισόδου p, οι συναρτήσεις ενεργοποίησης κάθε επιπέδου i $f^i$,τα βάρη $\mathbf{w}^i$ και ο πίνακας των βαρών $\mathbf{W}^i$, οι βάσεις $\mathbf{b}^i$, η ποσότητα στην οποία δρα η συνάρτηση ενεργοποίησης $\mathbf{n}^i=\mathbf{W}^{i*}\mathbf{p}^i+\mathbf{b}^i$, $\mathbf{a}^i$: η έξοδος από το επίπεδο i $\mathbf{a}^i= f^i(\mathbf{n}^i)$ [18].

#### 1.3.2.3.1. Μέθοδοι εκπαίδευσης για νευρωνικά δίκτυα τροφοδοσίας

Η εκπαίδευση των νευρωνικών δικτύων τροφοδοσίας προς τα εμπρός περιλαμβάνει προσαρμογή των βαρών και των βάσεων (weights and biases) για την ελαχιστοποίηση ενός προκαθορισμένου κόστους ή συνάρτησης σφάλματος. Έχουν αναπτυχθεί διάφορες μέθοδοι εκπαίδευσης, καθεμία με τα μοναδικά χαρακτηριστικά της και την καταλληλόλητά της για διαφορετικούς τομείς προβλημάτων:

**Κάθοδος κλίσης (Gradient Descent)**

Η Gradient descent είναι ένας ευρέως χρησιμοποιούμενος αλγόριθμος βελτιστοποίησης για την εκπαίδευση νευρωνικών δικτύων feed-forward. Περιλαμβάνει την επαναληπτική προσαρμογή των βαρών και των βάσεων με βάση την κλίση της συνάρτησης σφάλματος σε σχέση με τις παραμέτρους του δικτύου. Οι παραλλαγές του gradient descent περιλαμβάνουν την batch gradient descent, τη στοχαστική gradient descent και την mini-batch descent, καθεμία με διαφορετικούς συμβιβασμούς μεταξύ της ταχύτητας σύγκλισης και της υπολογιστικής απόδοσης, που εξαρτάται από την ποσότητα δεδομένων που χρησιμοποιείται για την εκπαίδευση / αλλαγή των παραμέτρων [18].

**Αλγόριθμος Levenberg-Marquardt Backpropagation (LMBP)**

Πριν την μαθηματική διατύπωση του αλγορίθμου, αξίζει να γίνει αναφορά στον αλγόριθμο backpropagation. Αυτός είναι ένας βασικός αλγόριθμος για τον υπολογισμό της διαβάθμισης σε νευρωνικά δίκτυα feed-forward. Διαδίδει το σφάλμα από το επίπεδο εξόδου στα κρυφά επίπεδα, επιτρέποντας στο δίκτυο να μάθει τις κατάλληλες ενημερώσεις βάρους. Ο αλγόριθμος υπολογίζει τις διαβαθμίσεις χρησιμοποιώντας τον κανόνα της αλυσίδας του λογισμού, επιτρέποντας τον αποτελεσματικό υπολογισμό των ενημερώσεων βάρους στα επίπεδα και είναι ο ακόλουθος [18] :

Έστω p το διάνυσμα μεταβλητών εισόδου, t: το διάνυσμα μεταβλητών εξόδου $\alpha^i$ : το διάνυσμα των αποτελεσμάτων μετά από το i επίπεδο του νευρωνικού, W: ο πίνακας που περιλαμβάνει τα διάφορα βάρη όλων των νευρώνων του νευρωνικού και b: το διάνυσμα των βάσεων όλων των επιπέδων του νευρωνικού. Σε όλη την ανάλυση, θεωρείται ότι το νευρωνικό έχει M επίπεδα / νευρώνες. Με f συμβολίζονται οι συναρτήσεις ενεργοποίησης ενώ με F η συνάρτηση επίδοσης / κόστους, που δεν είναι άλλη από μέσο τετραγωνικό σφάλμα με εξίσωση:

$$F(\boldsymbol{x}) = \sum_{q=1}^{Q}(\boldsymbol{t}_q - \mathbf{a}_q)^T(\boldsymbol{t}_q - \mathbf{a}_q) = \sum_{q=1}^{Q}\boldsymbol{e}_q{}^T\boldsymbol{e}_q = \sum_{q=1}^{Q}\sum_{j=1}^{S^M}(e_{j,q})^2 = \sum_{i=1}^{N}(v_i)^2, \quad (1.3.1)$$

όπου: $e_{j,q}$ είναι το j στοιχείο του σφάλματος για το q ζεύγος τιμής εισόδου και τιμής στόχου.

Το πρώτο βήμα είναι η διάδοση των δεδομένων εισόδου στο δίκτυο:

$$\mathbf{a^0} = \mathbf{p}, \quad (1.3.2)$$

$$\mathbf{a^{m+1}} = \mathbf{f^{m+1}}\left(\mathbf{W^{m+1}a^m} + \mathbf{b^{m+1}}\right) \quad \boldsymbol{για\ m} = \mathbf{0}, \mathbf{1}, \ldots, \boldsymbol{M} - \mathbf{1}, \quad (1.3.3)$$

$$\mathbf{a} = \mathbf{a}^{\mathbf{M}}. \tag{1.3.4}$$

Το επόμενο βήμα είναι ο υπολογισμός των ευαισθησιών προς τα πίσω μέσω του νευρωνικού.

$$\boldsymbol{s}^{\mathbf{M}} = -\mathbf{2}\dot{\boldsymbol{F}}^{\boldsymbol{M}}(\mathbf{n}^{\mathbf{M}})(\boldsymbol{t} - \boldsymbol{a}), \tag{1.3.5}$$

$$\boldsymbol{s}^{\mathbf{m}} = \dot{\boldsymbol{F}}^{\boldsymbol{m}}(\mathbf{n}^{\mathbf{m}})(\boldsymbol{W}^{\boldsymbol{m+1}})^{T}\boldsymbol{s}^{\mathbf{m+1}}, \boldsymbol{\gamma\iota\alpha}\ \boldsymbol{m} = \boldsymbol{M} - \mathbf{1}, \ldots, \mathbf{2}, \mathbf{1}. \tag{1.3.6}$$

Τέλος με $\dot{F}$ συμβολίζεται ο διαγώνιος πίνακας που περιλαμβάνει στην κύρια διαγώνιο του τις πρώτες παραγώγους των συναρτήσεων ενεργοποίησης των νευρώνων του δικτύου ξεκινώντας από το πρώτο επίπεδο και πηγαίνοντας γραμμικά προς το τελευταίο.

Τέλος, τα βάρη και οι βάσεις ενημερώνονται χρησιμοποιώντας τον κανόνα της προσεγγιστικής πιο απότομης κλίσης στο βήμα k (1.3.7-1.3.8) :

$$\mathbf{W}^{\mathbf{m}}(\boldsymbol{k} + \mathbf{1}) = \mathbf{W}^{\mathbf{m}}(\boldsymbol{k}) - \boldsymbol{\alpha}\boldsymbol{s}^{\mathbf{m}}\left(\mathbf{a}^{\mathbf{m-1}}\right)^{\boldsymbol{T}}, \tag{1.3.7}$$

$$\mathbf{b}^{\mathbf{m}}(\boldsymbol{k} + \mathbf{1}) = \mathbf{b}^{\mathbf{m}}(\boldsymbol{k}) - \boldsymbol{\alpha}\boldsymbol{s}^{\mathbf{m}}. \tag{1.3.8}$$

Όπου α: ο ρυθμός μάθησης (learning rate). Υπάρχουν τεχνικές προσδιορισμού του βέλτιστου ρυθμού, μέθοδοι μεταβαλλόμενου ρυθμού( αύξηση ή μείωση του ανάλογα με την πορεία στην ελαχιστοποίηση της συνάρτησης κόστους).

Ο αλγόριθμος Levenberg-Marquardt, ο οποίος φέρει το όνομά του από τους Kenneth Levenberg και Donald Marquardt είναι ένας αλγόριθμος αριστοποίησης που χρησιμοποιείται συχνά για μη γραμμικά προβλήματα ελαχίστων τετραγώνων για προσαρμογή καμπυλών. Συνδυάζει τα πλεονεκτήματα της μεθόδου Gauss-Newton και της μεθόδου πιο απότομης κλίσης για να εκτιμήσει αποτελεσματικά τις παραμέτρους ενός μαθηματικού μοντέλου που θα ταιριάζει καλύτερα στα δεδομένα [20-22].

Ο αλγόριθμος είναι ο ακόλουθος [18]:

1. Αρχικά, παρουσιάζονται όλα τα ορίσματα εισόδου στο δίκτυο και υπολογίζονται οι αντίστοιχες τιμές εξόδου του νευρωνικού (από τις εξισώσεις 1.3.1 και 1.3.2), όπως και τα σφάλματα $\mathbf{e}_{\mathrm{q}}=\boldsymbol{t}_q - \boldsymbol{\alpha}_q^M$. Έπειτα, υπολογίζεται το άθροισμα των τετραγώνων των σφαλμάτων σε όλες τις τιμές εισόδου F(**x**) (χρησιμοποιώντας την εξίσωση 1.3.1).

2. Υπολογίζονται οι ευαισθησίες και οι σχέσεις επανάληψης (recurrence relations) αφού ξεκινήσει ο υπολογισμός από την σχέση 1.3.9.

$$\tilde{\boldsymbol{S}}_{\boldsymbol{q}}^{\boldsymbol{M}} = -\dot{\boldsymbol{F}}^{\boldsymbol{M}}(\boldsymbol{n}_{\boldsymbol{q}}^{\boldsymbol{M}}), \tag{1.3.9}$$

$$\tilde{s}_{i,h}^{m} = \frac{\partial v_h}{\partial n_{i,q}^{m}} = \frac{\partial e_{k,q}}{\partial n_{i,q}^{m}}\ (Ευαισθησία\ Marquardt), \qquad όπου\ h = (q-1)S^M + k \tag{1.3.10}$$

$$\tilde{\boldsymbol{S}}_{\boldsymbol{q}}^{\boldsymbol{m}} = -\dot{\boldsymbol{F}}^{\boldsymbol{m}}(\boldsymbol{n}_{\boldsymbol{q}}^{\boldsymbol{m}})\left(W^{\boldsymbol{m+1}}\right)^{T}\tilde{\boldsymbol{S}}_{\boldsymbol{q}}^{\boldsymbol{m+1}}. \tag{1.3.11}$$

3. Όλοι οι πίνακες ενώνονται για τις ευαισθησίες Marquardt Εφαρμόζοντας την εξίσωση 1.3.12.

$$\tilde{\boldsymbol{S}}^{\boldsymbol{m}} = \left[\tilde{\boldsymbol{S}}_{\boldsymbol{1}}^{\boldsymbol{m}} | \tilde{\boldsymbol{S}}_{\boldsymbol{2}}^{\boldsymbol{m}} | \ldots | \tilde{\boldsymbol{S}}_{\boldsymbol{Q}}^{\boldsymbol{m}}\right]. \tag{1.3.12}$$

Τα στοιχεία του Ιακωβιανού πίνακα υπολογίζονται από τις σχέσεις 1.3.13 και 1.3.14, ενώ ο Ιακωβιανός δίνεται από την 1.3.15.

$$[J]_{h,l} = \frac{\partial v_h}{\partial x_l} = \frac{\partial e_{k,q}}{\partial w_{i,j}^m} = \frac{\partial e_{k,q}}{\partial n_{i,q}^m} x \frac{\partial n_{i,q}^m}{\partial w_{i,j}^m} = \tilde{s}_{i,h}^m x \frac{\partial n_{i,q}^m}{\partial w_{i,j}^m} = \tilde{s}_{i,h}^m \; x \; a_{j,q}^{m-1} \; \gamma\iota\alpha \; \tau o \; \beta\acute{\alpha}\rho o\varsigma \; x_l. \tag{1.3.13}$$

$$[J]_{h,l} = \frac{\partial v_h}{\partial x_l} = \frac{\partial e_{k,q}}{\partial b_i^m} = \frac{\partial e_{k,q}}{\partial n_{i,q}^m} x \frac{\partial n_{i,q}^m}{\partial b_i^m} = \tilde{s}_{i,h}^m x \frac{\partial n_{i,q}^m}{\partial b_i^m} = \tilde{s}_{i,h}^m \;\; \gamma\iota\alpha \; \tau\eta \; \beta\acute{\alpha}\sigma\eta \; (bias) \; x_l. \tag{1.3.14}$$

$$\boldsymbol{J}(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial e_{1,1}}{\partial w_{1,1}^1} & \frac{\partial e_{1,1}}{\partial w_{1,2}^1} & \cdots & \frac{\partial e_{1,1}}{\partial w_{S^1,R}^1} & \frac{\partial e_{1,1}}{\partial b_1^1} & \cdots \\ \frac{\partial e_{2,1}}{\partial w_{1,1}^1} & \frac{\partial e_{2,1}}{\partial w_{1,2}^1} & \cdots & \frac{\partial e_{2,1}}{\partial w_{S^1,R}^1} & \frac{\partial e_{2,1}}{\partial b_1^1} & \cdots \\ \vdots & \vdots & & \vdots & \vdots & \\ \frac{\partial e_{S^M,1}}{\partial w_{1,1}^1} & \frac{\partial e_{S^M,1}}{\partial w_{1,2}^1} & \cdots & \frac{\partial e_{S^M,1}}{\partial w_{S^1,R}^1} & \frac{\partial e_{S^M,1}}{\partial b_1^1} & \cdots \\ \frac{\partial e_{1,2}}{\partial w_{1,1}^1} & \frac{\partial e_{1,2}}{\partial w_{1,2}^1} & \cdots & \frac{\partial e_{1,2}}{\partial w_{S^1,R}^1} & \frac{\partial e_{1,2}}{\partial b_1^1} & \cdots \\ \vdots & \vdots & & \vdots & \vdots & \end{bmatrix}. \tag{1.3.15}$$

4. Επιλύεται η εξίσωση 1.3.16 για να ληφθεί η διαφορά $\Delta\mathbf{x}_k$.

$$\Delta\mathbf{x}_k = -[\mathbf{J}^T(\mathbf{x}_k)\mathbf{J}(\mathbf{x}_k) + \mu_k\mathbf{I}]^{-1}\mathbf{J}^T(\mathbf{x}_k)\mathbf{v}(\mathbf{x}_k)\,. \tag{1.3.16}$$

Όπου:

$$\mathbf{v}^T = [\boldsymbol{v_1}\, \boldsymbol{v_2} \ldots \boldsymbol{v_N}] = \left[\boldsymbol{e_{1,1}}\, \boldsymbol{e_{2,1}} \ldots \boldsymbol{e_{S^M,1}} \boldsymbol{e_{1,2}} \;\ldots \boldsymbol{e_{S^M,Q}}\right], \tag{1.3.17}$$

$$\mathbf{x}^T = [\boldsymbol{x_1}\, \boldsymbol{x_2} \ldots \boldsymbol{x_N}] = \left[\boldsymbol{w_{1,1}^1 w_{1,2}^1} \ldots \boldsymbol{w_{S^1,R}^1 b_1^1} \ldots \boldsymbol{b_{S^1}^1 w_{1,1}^2} \ldots \boldsymbol{b_{S^M}^M}\right], \tag{1.3.18}$$

$$N = Q \times S^M and\; n = S^1(R+1) + S^2(S^1+1) + \cdots + S^M(S^{M-1}+1). \tag{1.3.19}$$

5. Υπολογίζεται ξανά το άθροισμα των τετραγώνων των σφαλμάτων χρησιμοποιώντας το $\mathbf{x}_k + \Delta\mathbf{x}_k$. Αν το σφάλμα είναι μικρότερο από αυτό που υπολογίστηκε στο βήμα 1, τότε το μ διαιρείται με το θ, $\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta\mathbf{x}_k$ και επαναλαμβάνεται το βήμα 1. Στην αντίθετη περίπτωση, πολλαπλασιάζεται το μ με το θ και επαναλαμβάνεται το βήμα 3. (μ,θ ορίζονται από τον χρήστη).

Ο αλγόριθμος τερματίζεται και έχει συγκλίνει όταν η νόρμα της κλίσης είναι μικρότερη από μια προκαθορισμένη τιμή ή όταν το άθροισμα των τετραγώνων είναι μικρότερο από μια συγκεκριμένη τιμή στόχο.

Η κλίση δίνεται από την εξίσωση (1.3.20) :

$$\nabla F(\boldsymbol{x}) = 2\boldsymbol{J}^{\boldsymbol{T}}(\boldsymbol{x})\boldsymbol{v}(\boldsymbol{x}). \qquad (1.3.20)$$

Ο όρος $\mathbf{J}^{\mathbf{T}}(\mathbf{x}_{\mathrm{k}})\mathbf{J}(\mathbf{x}_{\mathrm{k}}) + \mu_{\mathrm{k}}\mathbf{I}$ είναι προσέγγιση του Εσσιανού πίνακα και χρησιμοποιείται έτσι ώστε ο πίνακας να γίνει θετικά ορισμένος (μη αρνητικές ιδιοτιμές), να εξασφαλίζει την αντιστρεψιμότητά του και να υπάρχει λύση στο πρόβλημα ελαχίστων τετραγώνων.

**Τεχνικές κανονικοποίησης (Regularization techniques)**

Για να αποφευχθεί η υπερβολική προσαρμογή και να βελτιωθεί η γενίκευση, κατά τη διάρκεια της εκπαίδευσης εφαρμόζονται συνήθως διάφορες τεχνικές κανονικοποίησης. Οι μέθοδοι κανονικοποίησης, όπως η κανονικοποίηση L1 και L2, επιβάλλουν περιορισμούς στα βάρη του δικτύου, ενισχύοντας την απλότητα και μειώνοντας την πολυπλοκότητα του μοντέλου. Η κανονικοποίηση τύπου dropout αποβάλλει τυχαία ορισμένους νευρώνες κατά τη διάρκεια της εκπαίδευσης, μειώνοντας τις αλληλεξαρτήσεις και προάγοντας την ευρωστία [24].

#### 1.3.2.3.2. Χαρακτηριστικά Νευρωνικών Δικτύων Τροφοδοσίας Προώθησης

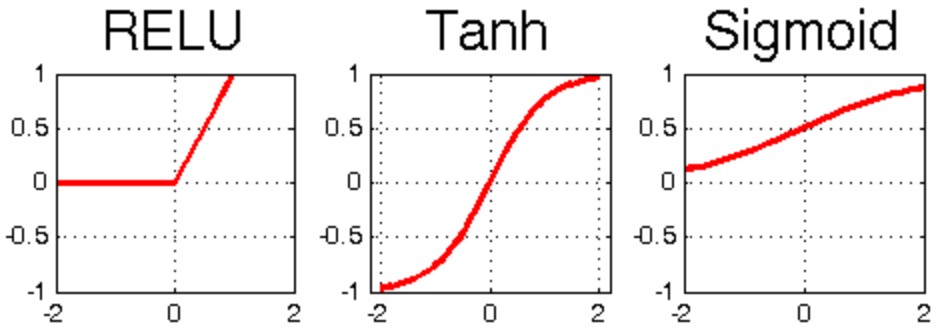
Η κατανόηση των χαρακτηριστικών των νευρωνικών δικτύων feed-forward είναι απαραίτητη για τον αποτελεσματικό σχεδιασμό και τη χρήση τους. Μερικά βασικά χαρακτηριστικά περιλαμβάνουν:

**Καθολική προσέγγιση**

Τα νευρωνικά δίκτυα τροφοδοσίας με επαρκή αριθμό κρυμμένων νευρώνων και κατάλληλες συναρτήσεις ενεργοποίησης έχουν την ικανότητα να προσεγγίζουν οποιαδήποτε συνεχή συνάρτηση, γνωστή ως θεώρημα καθολικής προσέγγισης. Αυτή η ιδιότητα τους επιτρέπει να μοντελοποιούν πολύπλοκες σχέσεις και να καταγράφουν μη γραμμικά μοτίβα σε δεδομένα [25].

**Μη γραμμικότητα και Λειτουργίες ενεργοποίησης**

Η μη γραμμικότητα που εισάγεται από τις συναρτήσεις ενεργοποίησης επιτρέπει στα νευρωνικά δίκτυα τροφοδοσίας να μοντελοποιούν μη γραμμικές σχέσεις. Οι κοινές συναρτήσεις ενεργοποίησης περιλαμβάνουν τη σιγμοειδή, την υπερβολική εφαπτομένη (tanh) και την ReLU και η μορφή τους φαίνεται στο ακόλουθο σχήμα 1-4. Η επιλογή της λειτουργίας ενεργοποίησης μπορεί να επηρεάσει τη σύγκλιση, την εκφραστική ισχύ και την υπολογιστική απόδοση του δικτύου [18].




*Σχήμα 1-4. Γραφική παράσταση των συναρτήσεων ReLU, Tanh και Sigmoid (by Muhammad Hamdan).*

**Βάθος και Πλάτος**

Το βάθος και το πλάτος ενός νευρωνικού δικτύου τροφοδοσίας προς τα εμπρός αναφέρονται στον αριθμό των κρυφών στρωμάτων και νευρώνων, αντίστοιχα. Τα βαθύτερα δίκτυα μπορούν να μάθουν ιεραρχικές αναπαραστάσεις, καταγράφοντας χαρακτηριστικά υψηλού επιπέδου, ενώ τα ευρύτερα δίκτυα μπορούν να συλλάβουν πιο περίπλοκα μοτίβα. Ωστόσο, τα βαθύτερα και ευρύτερα δίκτυα μπορεί επίσης να είναι επιρρεπή σε υπερπροσαρμογή και να απαιτούν περισσότερους υπολογιστικούς πόρους [16].

#### 1.3.2.3.3. Θεωρήσεις και βέλτιστες πρακτικές

Κατά την εργασία με νευρωνικά δίκτυα τροφοδοσίας, θα πρέπει να λαμβάνονται υπόψη διάφορες σκέψεις και βέλτιστες πρακτικές:

**Προεπεξεργασία δεδομένων**

Η σωστή προεπεξεργασία των δεδομένων εισόδου, συμπεριλαμβανομένης της κανονικοποίησης, της κλιμάκωσης και του χειρισμού των τιμών που λείπουν, μπορεί να επηρεάσει σημαντικά την απόδοση και τη σύγκλιση των νευρωνικών δικτύων προώθησης. Τεχνικές προεπεξεργασίας, όπως η κλιμάκωση χαρακτηριστικών και η κωδικοποίηση μίας χρήσης, διασφαλίζουν ότι τα δεδομένα είναι κατάλληλα για εκπαίδευση δικτύου. [16]

**Επιλογή υπερ-παραμέτρων**

Η απόδοση των νευρωνικών δικτύων τροφοδοσίας επηρεάζεται από διάφορες υπερ-παράμετρους, όπως ο ρυθμός εκμάθησης, το μέγεθος παρτίδας, ο αριθμός των κρυφών επιπέδων και οι συναρτήσεις ενεργοποίησης νευρώνων. Η διεξαγωγή πειραμάτων συντονισμού υπερπαραμέτρων, χρησιμοποιώντας τεχνικές όπως η αναζήτηση πλέγματος ή η τυχαία αναζήτηση, βοηθά στον εντοπισμό των βέλτιστων ρυθμίσεων υπερπαραμέτρων για βελτιωμένη απόδοση δικτύου [24].

**Πρόωρη διακοπή**

Η πρόωρη διακοπή, η οποία σταματά τη διαδικασία εκπαίδευσης όταν το σφάλμα επικύρωσης αρχίζει να αυξάνεται, είναι μια άλλη αποτελεσματική προσέγγιση για την αποφυγή υπερβολικής προσαρμογής [23-24].

#### *1.3.2.4 Εφαρμογές Νευρωνικών Δικτύων*

Τα νευρωνικά δίκτυα έχουν φέρει επανάσταση σε διάφορους τομείς, επιδεικνύοντας εξαιρετική απόδοση και ευελιξία. Μερικές αξιόλογες εφαρμογές περιλαμβάνουν:

**1. Computer Vision**: Τα νευρωνικά δίκτυα, ιδιαίτερα τα CNN, έχουν επιτύχει αξιοσημείωτη επιτυχία στην αναγνώριση εικόνων, την ανίχνευση αντικειμένων και τις εργασίες δημιουργίας εικόνας. Ενισχύουν τεχνολογίες όπως αυτοοδηγούμενα αυτοκίνητα, αναγνώριση προσώπου και ανάλυση ιατρικής εικόνας [16].

2**. Επεξεργασία φυσικής γλώσσας (NLP):** Τα επαναλαμβανόμενα νευρωνικά δίκτυα (RNN) και οι παραλλαγές τους έχουν συμβάλει σημαντικά σε εργασίες NLP όπως η μετάφραση γλώσσας, η ανάλυση συναισθημάτων και η δημιουργία κειμένου. Τα μοντέλα νευρωνικών γλωσσών όπως το GPT έχουν επιτύχει επιδόσεις αιχμής στην κατανόηση και τη δημιουργία γλωσσών [16].

**3. Προσέγγιση Συναρτήσεων και Επίλυση Διαφορικών Εξισώσεων:** Ένας κλάδος που αναπτύσσεται ταχύτητα και αναλύεται στην ακόλουθη ξεχωριστή ενότητα [18,25-29].

### 1.3.3. Επίλυση διαφορικών Εξισώσεων με Μηχανική Μάθηση

Οι διαφορικές εξισώσεις είναι θεμελιώδη μαθηματικά εργαλεία για την περιγραφή δυναμικών συστημάτων και φαινομένων. Παραδοσιακά, η επίλυση διαφορικών εξισώσεων βασίζεται σε αναλυτικές τεχνικές ή αριθμητικές μεθόδους. Ωστόσο, η εμφάνιση της μηχανικής μάθησης έχει ανοίξει νέες δυνατότητες για την επίλυση διαφορικών εξισώσεων. Διερευνάται η εφαρμογή των τεχνικών μηχανικής μάθησης στην επίλυση διαφορικών εξισώσεων, επισημαίνοντας τα πλεονεκτήματα, τις προκλήσεις και τις πρόσφατες εξελίξεις.

#### *1.3.3.1. Διαφορικές Εξισώσεις και Μηχανική Μάθηση*

Οι διαφορικές εξισώσεις περιλαμβάνουν την εύρεση της άγνωστης συνάρτησης που ικανοποιεί μια σχέση μεταξύ της συνάρτησης και των παραγώγων της. Οι τεχνικές μηχανικής μάθησης προσφέρουν μια εναλλακτική προσέγγιση για την επίλυση διαφορικών εξισώσεων αξιοποιώντας μεγάλες ποσότητες δεδομένων και υπολογιστική ισχύ. Εκπαιδεύοντας μοντέλα σε δεδομένα, η μηχανική μάθηση μπορεί να μάθει να προσεγγίζει λύσεις σε διαφορικές εξισώσεις και να κάνει προβλέψεις [30-31].

Έχουν προταθεί πολλά και ενδιαφέροντα μοντέλα, όπως εκείνο που χρησιμοποιεί 2 νευρωνικά δίκτυα και αθροίζει τις εξόδους, ώστε να δώσει τη λύση. Σε αυτή την περίπτωση το ένα νευρωνικό εκπαιδεύεται στις συνοριακές συνθήκες και το άλλο στην γενική μορφή της εξίσωσης [28]. Άλλοι προτείνουν την χρήση ενισχυτικής μάθησης που χρησιμοποιεί σαν πολιτική της εκπαίδευσης (policy) την συμπεριφορά της κλίσης (gradient) [29].

#### *1.3.3.2. Physics-Informed Neural Networks (PINNs)*

Τα νευρωνικά δίκτυα με πληροφόρηση φυσικής (Physics Informed Neural Networks, PINN) συνδυάζουν τη βαθιά μάθηση με την υποκείμενη φυσική των διαφορικών εξισώσεων. Τα PINN ενσωματώνουν τη

γνωστή φυσική ως περιορισμό κατά τη διάρκεια της εκπαίδευσης για να διασφαλίσουν ότι η μαθημένη λύση συμμορφώνεται με τις εξισώσεις που διέπουν. Αυτή η προσέγγιση επιτρέπει ακριβείς λύσεις, ενώ μειώνει την ανάγκη για εκτεταμένα δεδομένα [26, 32-33].

#### *1.3.3.3 Data-Driven Approaches*

Οι προσεγγίσεις που βασίζονται σε δεδομένα στοχεύουν να μάθουν άμεσα τη λύση μιας διαφορικής εξίσωσης από τα διαθέσιμα δεδομένα χωρίς να μοντελοποιούν ρητά την υποκείμενη φυσική. Αυτές οι μέθοδοι χρησιμοποιούν νευρωνικά δίκτυα για να προσεγγίσουν το χώρο λύσης και να ελαχιστοποιήσουν την απόκλιση μεταξύ των παρατηρούμενων δεδομένων και των προβλεπόμενων τιμών [25,34].

### 1.3.4. Γκαουσιανές Διεργασίες

Οι διεργασίες Gauss (GPs) είναι ισχυρά πιθανοτικά μοντέλα που παρέχουν ένα ευέλικτο πλαίσιο για τη μοντελοποίηση και την πρόβλεψη πολύπλοκων διανομών δεδομένων. Οι GP προσφέρουν μια μη παραμετρική προσέγγιση, επιτρέποντας ένα ευρύ φάσμα εφαρμογών στην παλινδρόμηση, την ταξινόμηση και πολλά άλλα [35].

Οι διεργασίες Gauss είναι στοχαστικές διαδικασίες όπου οποιοδήποτε πεπερασμένο σύνολο τυχαίων μεταβλητών ακολουθεί μια κοινή πολυμεταβλητή Γκαουσιανή κατανομή. Οι βασικές έννοιες στις διαδικασίες Gauss περιλαμβάνουν:

1. Συνάρτηση συνδιακύμανσης (kernel function): Η συνάρτηση συνδιακύμανσης, γνωστή και ως συνάρτηση πυρήνα, ορίζει την ομοιότητα μεταξύ των σημείων δεδομένων. Καταγράφει την υποκείμενη δομή και κωδικοποιεί τις υποθέσεις σχετικά με τη διανομή δεδομένων [35].

2. Προηγούμενες και Μεταγενέστερες Κατανομές: Οι GP ορίζουν μια προηγούμενη κατανομή έναντι των συναρτήσεων, η οποία μπορεί να ενημερωθεί σε μια μεταγενέστερη κατανομή μόλις ενσωματωθούν τα παρατηρούμενα δεδομένα. Η μεταγενέστερη κατανομή παρέχει έναν ευέλικτο και πιθανολογικό τρόπο για να γίνουν προβλέψεις [35].

Η παλινδρόμηση διαδικασίας Gauss είναι μια τεχνική που χρησιμοποιείται για τη μοντελοποίηση και την πρόβλεψη συναρτήσεων πραγματικών τιμών. Με δεδομένο ένα σύνολο ζευγών εισόδου-εξόδου, δύναται να γίνει η εκπαίδευση στην υποκείμενη συνάρτηση και να παρέχονται πιθανολογικές προβλέψεις σε οποιοδήποτε σημείο εισόδου. Η μεταγενέστερη κατανομή επιτρέπει την εκτίμηση της αβεβαιότητας, επιτρέποντας την ενημερωτική λήψη αποφάσεων.

Η ταξινόμηση διεργασιών Gauss επεκτείνει τις GP για να χειριστούν προβλήματα ταξινόμησης δυαδικών ή πολλαπλών κλάσεων. Με τη μοντελοποίηση των πιθανοτήτων κλάσης χρησιμοποιώντας μια διαδικασία Gaussian, οι GP παρέχουν ένα βασικό πλαίσιο για εργασίες ταξινόμησης. Η μεταγενέστερη κατανομή επιτρέπει πιθανοτικές προβλέψεις και ποσοτικοποίηση της αβεβαιότητας [35].

Οι διεργασίες Gauss μπορούν επίσης να εφαρμοστούν στην ανάλυση χρονοσειρών, όπου ο στόχος είναι η μοντελοποίηση και η πρόβλεψη της συμπεριφοράς μιας ακολουθίας δεδομένων με την πάροδο του

χρόνου. Με την ενσωμάτωση χρονικών εξαρτήσεων στη συνάρτηση συνδιακύμανσης, οι GPs καταγράφουν την υποκείμενη δυναμική και επιτρέπουν την πρόβλεψη με εκτίμηση αβεβαιότητας [36].

Οι διεργασίες Gauss έχουν βρει εφαρμογές σε διάφορους τομείς λόγω της ευελιξίας και της πιθανολογικής φύσης τους. Μερικές αξιόλογες εφαρμογές περιλαμβάνουν:

1.**Bayesian Optimization**: Οι διεργασίες Gauss χρησιμοποιούνται συνήθως στη Bayesian βελτιστοποίηση, όπου μοντελοποιούν την αντικειμενική συνάρτηση και καθοδηγούν την αναζήτηση βέλτιστων λύσεων με τρόπο αρχής και αποδοτικότητας ως προς το δείγμα [37].

2.**Χωρική Μοντελοποίηση**: Οι GP είναι αποτελεσματικοί σε εργασίες χωρικής μοντελοποίησης, όπως η γεωστατιστική, όπου μπορούν να μοντελοποιήσουν τη χωρική συσχέτιση και να παρέχουν προβλέψεις σε μη παρατηρούμενες τοποθεσίες [35].

3.**Εύρεση σημαντικών παράμετρών και όρων μιας εξίσωσης**: Καθώς αντιστοιχεί μια υπερπαράμετρος σε κάθε όρο που χρησιμοποιείται στις γκαουσιανές διαδικασίες ως μεταβλητή εισόδου γίνεται να βρεθεί και το κατά πόσο σημαντική είναι η μεταβλητή αυτή στην εξίσωση. Συγκεκριμένα, μικρή τιμή παραμέτρου υποδηλώνει την σημαντικότητα του όρου στην εξίσωση, ενώ αντίθετα, μεγάλη τιμή σημαίνει ότι ο όρος μπορεί να αγνοηθεί. Δύναται, λοιπόν, ένας πειραματιστής να μάθει τα σημαντικά φαινόμενα που διέπουν το πείραμα του, χρησιμοποιώντας τις GP ως ένα μαύρο κουτί (black box) και χωρίς να χρειάζονται να γνωρίζουν τη φυσική του προβλήματος  μέσω της ARD ανάλυσης (Automatic Relevance Determination ) , η οποία υποστηρίζει ότι οι κλίμακες μήκους συσχέτισης (correlation length scales) στη συνάρτηση συνδιακύμανσης μπορούν να χρησιμοποιηθούν για τη ανάδειξη της συσχέτισης μεταξύ των μεταβλητών εισόδου και να εντοπιστούν έτσι οι ενεργές μεταβλητές, δηλαδή αυτές που ουσιαστικά είναι οι σημαντικές. [38].

Οι διαδικασίες Gauss προσφέρουν ένα ισχυρό πιθανολογικό πλαίσιο για τη μοντελοποίηση και την πρόβλεψη πολύπλοκων κατανομών δεδομένων. Με την ευελιξία, τη μη παραμετρική φύση τους και την ικανότητά τους να παρέχουν εκτιμήσεις αβεβαιότητας, οι GP έχουν βρει διαφορετικές εφαρμογές στην παλινδρόμηση, την ταξινόμηση, την ανάλυση χρονοσειρών και πολλά άλλα. Κατανοώντας τα θεμελιώδη και αξιοποιώντας την πιθανολογική φύση των διαδικασιών Gauss, οι ερευνητές και οι επαγγελματίες μπορούν να αντιμετωπίσουν ένα ευρύ φάσμα προβλημάτων και να λάβουν τεκμηριωμένες αποφάσεις.

**Μαθηματική θεμελίωση Γκαουσιανών Διαδικασιών**

Όπως και οι κανονικές κατανομές σε μία διάσταση, οι διεργασίες Gauss, για να ορίζονται πλήρως απαιτούν δύο στοιχεία, μία μέση τιμή, η οποία συνήθως θεωρείται άγνωστη και λαμβάνεται ίση με το μηδέν καθώς και τον πίνακα συνδιακύμανσης (covariance matrix), ο οποίας εκφράζει την διακύμανση κάθε μεταβλητής στην κύρια διαγώνιό του, ενώ στις υπόλοιπες θέσεις δηλώνει την συνδιακύμανση μεταβλητών που μελετώνται ανά ζευγάρια. Ο πίνακας συνδιακύμανσης, λοιπόν, χρειάζεται μια συνάρτηση πάνω στην οποία θα σχηματιστεί και αποτελεί τον πυρήνα του πίνακα (εξού και η ονομασία kernel function κ). Η πιο συνηθισμένη μορφή της συνάρτησης πυρήνα είναι η radial basis function (RDF), η οποία βασίζεται στην ευκλείδεια απόσταση εντός του χώρου των μεταβλητών εισόδου και αποτελεί την βασική συνάρτηση που χρησιμοποιείται στα προβλήματα παλινδρόμησης με GP. Ο πίνακας σχηματίζεται ως εξής: [38-39]

$$K_{ij} = \kappa(\boldsymbol{x_i}, \boldsymbol{x_j}; \boldsymbol{\vartheta}) = \vartheta_0 \exp\left( -\frac{1}{2} \sum_{l=1}^{k} \frac{(x_{i,l} - x_{j,l})^2}{\vartheta_l} \right). \quad (1.3.21)$$

Στην παραπάνω εξίσωση είναι $\theta = [\theta_0, \theta_1, \dots, \theta_k]^T$ ένα διάνυσμα μήκους (k+1) με k τις διαστάσεις των δεδομένων εισόδου. Η διαδικασία μάθησης έγκειται στον υπολογισμό τιμών του διανύσματος θ έτσι, ώστε να ελαχιστοποιείται ο αρνητικός λογάριθμος της οριακής πιθανότητας (negative log marginal likelihood), η οποία περιγράφεται με την ακόλουθη εξίσωση (1.3.22):

$$\vartheta^* = argmin_\theta\{-logp(\boldsymbol{y}|\boldsymbol{x}, \theta)\} = \frac{1}{2}\boldsymbol{y}^T(K + \sigma^2 I)^{-1}\boldsymbol{y} + \frac{1}{2}log|(K + \sigma^2 I)| + \frac{N}{2}log2\pi, \quad (1.3.22)$$

όπου: N είναι ο αριθμός των σημείων εκπαίδευσης και $\sigma^2$ και I συμβολίζουν την διακύμανση του Γκαουσιανού θορύβου παρατήρησης και τον μοναδίαιο πίνακα NxN, αντίστοιχα.

## 1.4. Αριθμητικές μέθοδοι

### 1.4.1 Newton - GMRES (Γενικευμένο Ελάχιστο Υπόλοιπο)

Η μέθοδος Newton-GMRES αποτελεί υποκατηγορία των μεθόδων Newton-Krylov και χρησιμοποιείται όπως και η μέθοδος Newton-Raphson για επίλυση αλγεβρικής εξίσωσης με μία εξαρτημένη και ανεξάρτητη μεταβλητή**.**

Οι μέθοδοι τύπου Newton χρησιμοποιούνται συνήθως για την αριθμητική λύση συστημάτων μη γραμμικών εξισώσεων

$$\mathbf{F}(\mathbf{u}, \lambda) = 0 . \quad (1.4.1)$$

Η $\mathbf{F} : \mathbf{R^N} \times R \rightarrow \mathbf{R^N}$ είναι ένα διάνυσμα μη γραμμικών συναρτήσεων, το u είναι ένα διάνυσμα μεγέθους N και το $\lambda \in R$ είναι μια παράμετρος συνέχειας. Η μέθοδος Newton επιλύει το (1.4.1) επαναληπτικά ξεκινώντας από μια αρχική εικασία $u^0$ και στο k-ο βήμα προσεγγίζει τη λύση του (1.4.1) με το διάνυσμα

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \boldsymbol{\delta}\mathbf{u}^k, \quad k = 0, 1, \dots \text{ μέχρι τη σύγκλιση.} \quad (1.4.2)$$

Στο (1.4.2), $\boldsymbol{\delta}\mathbf{u}^k$ είναι η λύση του γραμμικού συστήματος εξισώσεων

$$\mathbf{J}(\mathbf{u}^k, \lambda)\,\boldsymbol{\delta}\mathbf{u}^k = -\mathbf{F}(\mathbf{u}^k, \lambda), \quad (1.4.3)$$

όπου $\mathbf{J}(\mathbf{u}^k, \lambda) \equiv \mathbf{F_u}(\mathbf{u}^k, \lambda) \in \mathbf{R}^{N \times N}$ είναι ο Ιακωβιανός πίνακας.

Όταν ο Ιακωβιανός είναι μεγάλος, οι περιορισμοί αποθήκευσης καθώς και η εκμετάλλευση των παράλληλων υπολογισμών απαιτούν την επαναληπτική λύση του γραμμικού συστήματος (1.4.3), οι επαναληπτικές μέθοδοι υποχώρου Krylov χρησιμοποιούνται συνήθως για την εξαγωγή μιας προσεγγιστικής λύσης του (1.4.3). Η μέθοδος τύπου Krylov απαιτεί μόνο το γινόμενο της Ιακωβιανής μήτρας με κάποια διανύσματα και όχι τον ρητό υπολογισμό όλων των στοιχείων της μήτρας [40]. Αυτό είναι απαραίτητο σε περιπτώσεις όπου δεν υπάρχει διαθέσιμη αναλυτική έκφραση για αυτά τα στοιχεία. Η παραλλαγή της μεθόδου GMRES [41], κοινώς γνωστή ως GMRES(m), είναι ο επαναληπτικός επιλύτης τύπου Krylov που προτιμάται στην περίπτωση που το γραμμικό σύστημα (1.4.3) είναι μη

συμμετρικό. Ο συνδυασμός του GMRES(m) με μια τεχνική προ συνθήκης (preconditioning) είναι απαραίτητη για βελτιωμένη σύγκλιση.

**Μαθηματική Θεμελίωση GMRES (Generalized Minimal Residual Method) πάνω στην Newton**

Η GMRES με προσταθεροποίηση (preconditioned) είναι η μέθοδος επιλογής για την επαναληπτική λύση μεγάλων συνόλων αλγεβρικών εξισώσεων με μη συμμετρικούς πίνακες, με βάση την παράλληλη απόδοσή του. Ξεκινώντας από μια αρχική εικασία, $\mathbf{x}_0$, της λύσης του (3), η GMRES χρησιμοποιεί τη μέθοδο Arnoldi, σε συνδυασμό με μια τεχνική ορθογωνοποίησης – η τροποποιημένη μέθοδος Gram–Schmidt χρησιμοποιείται εδώ – για να κατασκευάσει μια ορθοκανονική βάση $\mathbf{V_m} \in \mathbf{R^{N\times m}}$ του m-διάστατου υποχώρου Krylov [40].

$$\mathbf{K_m(J, v) = span}\{v, Jv, J^2v, \ldots, J^{m-1}v\}, \tag{1.4.4}$$

όπου: $v \equiv \frac{r_o}{\|r_0\|_2}$, $r_o \equiv -\mathbf{F} - \mathbf{J}x_o$, $\mathbf{F} \equiv \mathbf{F}(u^k, \lambda)$, $\mathbf{J} \equiv \mathbf{J}(u^k, \lambda)$.

Η νέα προσέγγιση της λύσης είναι:

$$\mathbf{x_m = x_0 + V_m y_m}\ , \tag{1.4.5}$$

όπου $\mathbf{y_m}$ είναι ένα διάνυσμα μεγέθους m και υπολογίζεται από τη λύση του προβλήματος των ελαχίστων τετραγώνων (1.4.6).

$$y_m = \underset{y}{argmin}\|\beta e_1 - \bar{H}_m y_m\|_2. \tag{1.4.6}$$

Στο (1.4.6), $\beta \equiv \|r_0\|_2$, $\mathbf{e_1} = [1, 0, \ldots, 0]^T$ και $\bar{H}_m \in \mathbf{R^{(m+1)\times m}}$ είναι ένας άνω πίνακας Hessenberg, τέτοιος ώστε

$$JV_m = V_{m+1}\bar{H}_m \Rightarrow V_m^T J V_m = H_m\ , \tag{1.4.7}$$

$\mathbf{H_m} \in \mathbf{R^{m\times m}}$ είναι ένας άνω πίνακας Hessenberg που λαμβάνεται από τον $\bar{H}_m$ διαγράφοντας την τελευταία του σειρά. Το πρόβλημα των ελαχίστων τετραγώνων (1.4.6) επιλύεται μετατρέποντας τον $\bar{H}_m$ σε έναν άνω τριγωνικό πίνακα $\mathbf{R_m} \in \mathbf{R^{m\times m}}$ χρησιμοποιώντας περιστροφές επιπέδου.

Οι απαιτήσεις αποθήκευσης και το υπολογιστικό κόστος της μεθόδου Arnoldi αυξάνονται γρήγορα με το m και, επομένως, η παραλλαγή του GMRES – το GMRES(m) – χρησιμοποιείται στην πράξη. Όταν το m φτάσει σε μια συγκεκριμένη προκαθορισμένη τιμή, ο αλγόριθμος επανεκκινείται, χρησιμοποιώντας την τελευταία προσέγγιση $\mathbf{x_m}$ από την (1.4.5) ως νέα αρχική εικασία.

Ο ρυθμός σύγκλισης του GMRES εξαρτάται από τις ιδιοτιμές (μικρότερες κατ' απόλυτη τιμή), που είναι κοντά στο μηδέν, του Ιακωβιανού πίνακα στην (1.4.3) [42] και επιδεινώνεται καθώς οι ιδιοτιμές τείνουν στο μηδέν. Οι ακραίες ιδιοτιμές του Jacobian προσεγγίζονται με τις αντίστοιχες ιδιοτιμές του πίνακα Hessenberg, γνωστές ως τιμές Ritz [40]. Σε κάποιο σημείο της λύσης, η διαδικασία GMRES συγκλίνει με υπεργραμμικό ρυθμό σε αντίθεση με την προηγούμενη γραμμική σύγκλιση. Ωστόσο, το GMRES(m) χάνει πληροφορίες που σχετίζονται με τις μικρότερες τιμές Ritz σε κάθε επανεκκίνηση και πολλές τεχνικές προετοιμασίας έχουν αναπτυχθεί που στοχεύουν στη διατήρηση αυτών των πληροφοριών μέσω της διαδικασίας. Ένας προπαρασκευαστής είναι απαραίτητος για την ενίσχυση του ρυθμού σύγκλισης του GMRES(m), ειδικά κοντά σε ιδιάζοντα σημεία (singular points). Έτσι, το αρχικό γραμμικό σύστημα (1.4.3) πρέπει να μετατραπεί σε ένα ισοδύναμο που έχει καλύτερες ιδιότητες σύγκλισης. Τεχνικές προσταθεροποίησης μπορούν να βρεθούν στην βιβλιογραφία[40,43].

### 1.4.2 Μέθοδος Keller – Pseudo Arc length Continuation

Για την παραμετρική ανάλυση μιας εξίσωσης, μια από τις πιο δημοφιλείς τεχνικές είναι η μέθοδος Keller (Pseudo Arc-Length Continuation) [43-44]. Στην μέθοδο αυτή, η παράμετρος ως προς την οποία γίνεται η ανάλυση, θεωρείται και αυτή ως μεταβλητή του συστήματος και έτσι προκύπτει ένα επαυξημένο σύστημα εξισώσεων που χρειάζεται μια επιπλέον εξίσωση για να μπορεί να επιλυθεί. Η εξίσωση αυτή χρησιμοποιεί το διάνυσμα διεύθυνσης της λύσης και της παραμέτρου ($\dot{\boldsymbol{u}}_{\boldsymbol{0}}, \dot{\lambda}_0$):

$(\mathbf{u_1 - u_0})^* \dot{\boldsymbol{u}}_{\boldsymbol{0}} + (\lambda_1 - \lambda_0)^* \dot{\lambda}_0 - ds = 0,$ (1.4.8)

όπου ds είναι το λεγόμενο ‘’μήκος τόξου’’ και είναι μια μικρή ποσότητα που ορίζεται από τον χρήστη.

Έτσι το σύστημα που καλείται κάποιος να επιλύσει για να γίνει η παραμετρική ανάλυση είναι το ακόλουθο:

Για την λύση ($\mathbf{u_1}, \lambda_1$) της μεθόδου Newton :

$$\begin{pmatrix} \left(\boldsymbol{G}_{\boldsymbol{u}}^{\boldsymbol{1}}\right)^{(\boldsymbol{\nu})} & \left(\boldsymbol{G}_{\lambda}^{1}\right)^{(\nu)} \\ \dot{\boldsymbol{u}}_{\boldsymbol{0}}^{*} & \dot{\lambda}_0 \end{pmatrix} \begin{pmatrix} \Delta \boldsymbol{u}_1^{(\nu)} \\ \Delta \lambda_1^{(\nu)} \end{pmatrix} = \begin{pmatrix} \boldsymbol{G}(\boldsymbol{u}_1^{(\nu)}, \lambda_1^{(\nu)}) \\ (\boldsymbol{u}_1^{(\nu)} - \mathbf{u_0})^* \dot{\boldsymbol{u}}_{\boldsymbol{0}} + (\lambda_1^{(\nu)} - \lambda_0)^* \dot{\lambda}_0 - ds \end{pmatrix}. \quad (1.4.9)$$

Ενώ, το επόμενο διάνυσμα κατεύθυνσης υπολογίζεται από την επίλυση του συστήματος (1.4.10 ή 1.4.11):

$$\begin{pmatrix} \boldsymbol{G}_{\boldsymbol{u}}^{\boldsymbol{1}} & \boldsymbol{G}_{\lambda}^{1} \\ \dot{\boldsymbol{u}}_{\boldsymbol{0}}^{*} & \dot{\lambda}_0 \end{pmatrix} \begin{pmatrix} \dot{\boldsymbol{u}}_{\boldsymbol{1}} \\ \dot{\lambda}_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}. \quad (1.4.10)$$

$$\begin{pmatrix} \dot{\boldsymbol{u}}_{\boldsymbol{1}} \\ \dot{\lambda}_1 \end{pmatrix} = \frac{\begin{pmatrix} \Delta \boldsymbol{u}_1^{(\nu)} \\ \Delta \lambda_1^{(\nu)} \end{pmatrix}}{norm \begin{pmatrix} \Delta \boldsymbol{u}_1^{(\nu)} \\ \Delta \lambda_1^{(\nu)} \end{pmatrix}}. \quad (1.4.11)$$

Επειδή χρησιμοποιείται σε συνδυασμό με την μέθοδο Newton – Raphson ισχύει ότι $(\boldsymbol{G}_{\boldsymbol{u}})^{(\boldsymbol{\nu})} = \boldsymbol{J}^{(\nu)}$.

Η 1.4.11 δεν αποτελεί άλλη από την έκφραση του μοναδιαίου διανύσματος κατεύθυνσης χρησιμοποιώντας την κατά προσέγγιση παραγώγιση.

# 2. Μεθοδολογία

## 2.1. Συνοπτική παρουσίαση της μεθοδολογίας

Στόχος της παρούσας διπλωματικής εργασίας είναι η παρουσίαση ενός αλγορίθμου / μεθοδολογίας στο λογισμικό Matlab (Έκδοση R2021b) , τόσο απλής που να μπορεί να εφαρμοσθεί από κάθε πειραματιστή, ώστε να αποκτήσει γνώση όλων των καταστάσεων του συστήματος που μελετά. Σε αυτές περιλαμβάνονται η γνώση των φυσικών φαινομένων που λαμβάνουν χώρα στο πείραμα (τα οποία εκδηλώνονται με χωρικές παραγώγους, αλλά και την ίδια την τιμή της συνάρτησης), η έγκυρη μελέτη της δυναμικής εξέλιξης του φαινομένου, καθώς επίσης και η εύρεση όλων των μόνιμων καταστάσεων, τόσο ασταθών όσο και ευσταθών [45]. Έτσι, εφαρμόζονται μέθοδοι μηχανικής μάθησης, για την επιλογή παραμέτρων και την ανάπτυξη μοντέλων, που θα προσεγγίζουν το δεξί σκέλος μερικής διαφορικής εξίσωσης και ο συνδυασμός των μεθόδων μηχανικής μάθησης με κλασικές μεθόδους αριθμητικής ανάλυσης για την ενσωμάτωση των συνοριακών συνθηκών. Προς επίτευξη του στόχου αυτού ακολουθείται η μεθοδολογική προσέγγιση που περιγράφεται σε αυτό το κεφάλαιο.

Αρχικά, στην ενότητα 2.2.1 η μονοδιάστατη εξίσωση Bratu χρησιμοποιείται ως το δεξί σκέλος μίας απλής μερικής διαφορικής εξίσωσης ($\frac{\partial u}{\partial t} = Bratu$) για την παραγωγή των απαραιτήτων δεδομένων μέσω προσομοιώσεων στο υπολογιστικό περιβάλλον COMSOL Multiphysics (Έκδοση 5.2, Classkit License), τα οποία αξιοποιούνται εν συνεχεία κατά την εφαρμογή διεργασίας Gauss και την εκπαίδευση νευρωνικών δικτύων [46].

Έπειτα, θεωρώντας άγνωστη την εξίσωση των δεδομένων, εφαρμόζονται οι διεργασίες Gauss ώστε να βρεθούν ποιες μεταβλητές / όροι καθορίζουν το δεξί σκέλος της διαφορικής σαν μια τεχνική μαύρου κουτιού. Έπειτα, κατασκευάζονται μοντέλα νευρωνικού δικτύου τα οποία χρησιμοποιούνται ως μοντέλα παλινδρόμησης, υπολογίζοντας, για κάθε τιμή των επιλεγμένων μεταβλητών εισόδου, την τιμή του αριστερού σκέλους της διαφορικής εξίσωσης, ήτοι, του χρονικού διαφορικού της εξαρτημένης μεταβλητής (∂u/∂t).

Στην ενότητα 2.2.2., παρουσιάζονται αναλυτικά τα βήματα που ακολουθήθηκαν για την εκπαίδευση του νευρωνικού δικτύου που υπολογίζει τη χρονική παράγωγο στο μονοδιάστατο πρόβλημα.

Στην ενότητα 2.2.3., με τα μοντέλα αυτά προσεγγίζεται η λύση της διαφορικής εξίσωσης σε μεταβατική κατάσταση χρησιμοποιώντας ode solvers και μάλιστα σε διαφορετική διακριτοποίηση της χωρικής μεταβλητής (x) και για μεγαλύτερους χρόνους σε σχέση με τα δεδομένα που χρησιμοποιήθηκαν για την εκπαίδευση. Τα αποτελέσματα συγκρίνονται με την ‘πραγματική λύση’, όπως αυτή παράγεται με τη μέθοδο των πεπερασμένων στοιχείων. Για να είναι δυνατή η χρήση εργαλείων συνήθων διαφορικών εξισώσεων στην επίλυση μερικών διαφορικών, χρησιμοποιήθηκε η μέθοδος των γραμμών (method of lines), στην οποία όλες εκτός από μια διάσταση (εδώ ο χρόνος) διακριτοποιούνται, δίνοντας την ευελιξία να επιλυθεί η εξίσωση ως συνήθης.

Στην ενότητα 2.2.4. εξετάζεται το καλύτερο από τα μοντέλα που παρήχθησαν με τη διαδικασία της ενότητας (2.2.1) για την επίλυση του προβλήματος εύρεσης της μόνιμης κατάστασης του συστήματος. Προς τούτο αξιοποιείται μία μέθοδος Newton-Krylov και συγκεκριμένα η Newton-GMRES, η οποία είναι μια μη ακριβής μέθοδος Newton και έχει αναλυθεί στην ενότητα 1.4. Επειδή, το νευρωνικό αδυνατούσε να έχει καλή προσαρμογή στις ασταθείς μόνιμες καταστάσεις, γεγονός που οφείλεται στην

εκπαίδευση του στις μικρές τιμές του ευσταθούς κλάδου, προτείνεται η επανάληψη δύο μόνο πειραμάτων σε τιμές του ασταθούς κλάδου ή πειραμάτων που έχουν μεγάλες τιμές της συνάρτησης σε αρχικές συνθήκες, με αρχική συνθήκη την λύση που προκύπτει από το νευρωνικό. Αυτό οδηγεί στην παραγωγή νέων δεδομένων και επανεκπαίδευση του νευρωνικού δικτύου με αποτέλεσμα την καλύτερη προσαρμογή στα δεδομένα. Γίνεται σύγκριση μεταξύ των λύσεων που προκύπτουν από τη μέθοδο πεπερασμένων στοιχείων τόσο σε διαφορετικές μόνιμες καταστάσεις (ευσταθείς και ασταθείς) και φαίνεται η ακρίβεια του μοντέλου. Τέλος, ακολουθεί παραμετρική ανάλυση, ώστε να βρεθούν όλες οι πιθανές λύσεις. Συγκρίνεται η λύση του νευρωνικού σε συνδυασμό με την Newton GMRES και τη μέθοδο Keller και σε εκείνη που προκύπτει από τον συνδυασμό FEM με τη μέθοδο Keller (Pseudo Arc length Continuation Method).

Τέλος στο κεφάλαιο 2.3 επαναλαμβάνονται τα βήματα των κεφαλαίων 2.2.1. με 2.2.4 για την επίλυση του πιο περίπλοκου 2D προβλήματος Bratu.

Στο κεφάλαιο 2.4 παρουσιάζεται η εργασία που έγινε στο λογισμικό Matlab.

## 2.2. Μεθοδολογία Μονοδιάστατου προβλήματος Bratu

### 2.2.1. Προσομοιώσεις και Γκαουσιανές Διαδικασίες

Η τεχνητή νοημοσύνη και ειδικότερα η μηχανική μάθηση χρειάζεται συνήθως πολλά δεδομένα για την εφαρμογή τους. Τα δεδομένα αυτά μπορεί να είναι πειραματικά ή να προκύπτουν από προσομοιώσεις. Στην διπλωματική αυτή επιλέγεται η χρήση προσομοιώσεων ως πιο γρήγορη στην παραγωγή των ζητούμενων αποτελεσμάτων και πιο εύκολη στην εφαρμογή της υπό μελέτης εξίσωση.

Έτσι, η εξίσωση, που μελετάται στην περίπτωση του μονοδιάστατου προβλήματος, λαμβάνει την ακόλουθη μορφή (2.1):

$$\boldsymbol{f(x, u_x, u_{xx}, \lambda) = \frac{\partial u}{\partial t} = u_{xx} + \lambda e^{u}, x \in (0, 1)}, \quad (2.1)$$

$$\boldsymbol{u(0, t) = u(1, t) = 0\ (συνοριακές\ συνθήκες)}, \quad (2.1)$$

$$\boldsymbol{u(x, 0) = g(x)\ (αρχική\ συνθήκη)}. \quad (2.1)$$

Η εξίσωση 2.1 μπορεί να επιλυθεί με πολλές τεχνικές, όπως είναι οι σειρές Chebyshev, με πεπερασμένες διαφορές και άλλες. Ωστόσο, επιλέγεται να επιλυθεί με τη μέθοδο των πεπερασμένων στοιχείων όπως αυτή εφαρμόζεται στο λογισμικό COMSOL Multiphysics 5.2. και συγκεκριμένα στην κατηγορία του μεταβατικής κατάστασης προβλήματος και γενική μορφή της μερικής διαφορικής εξίσωσης. Έπειτα, προστίθενται οι συνοριακές συνθήκες, οι οποίες είναι τύπου Dirichlet, όπως περιγράφεται και στην ενότητα 1.2.1. Τέλος, πρέπει να ορισθούν και οι αρχικές συνθήκες, δηλαδή οι τιμές της u για κάθε χ την στιγμή 0. Χρησιμοποιήθηκαν διαφορετικές αρχικές συνθήκες, οι οποίες παρουσιάζονται στον Πίνακα 2.1.

*Πίνακας 2-1. Συναρτήσεις των αρχικών συνθήκων που χρησιμοποιήθηκαν για τη συλλογή των δεδομένων για το πρόβλημα Bratu 1D.*

| **Αρχική Συνθήκη / Περιγραφή** | **Εξίσωση** |
|---|---|
| Γκαουσιανή κατανομή (gp1) | $y(x) = \frac{1}{0.15\sqrt{2\pi}} \exp\left(-\frac{(x-0.6)^2}{2*0.15^2}\right)$ |
| Τριγωνικός παλμός (trig_1) | $y(x) = \begin{cases} 0, & x = 0 \\ 2x, & x \in (0,1) \\ 1-2x, & x \in (0,1) \\ 0, & x = 1 \end{cases}$ |
| Ορθογώνιος παλμός (pulse_1) | $y(x) = \begin{cases} 0, & x = 0 \\ 1, & x \in (0,1) \\ 0, & x = 1 \end{cases}$ |
| Κυματικός παλμός / ημίτονο (sin_1) | $y(x) = \sin(2\pi x)$ |
| Συνάρτηση τύπου Sigmoid (exp_sigm_1) | $y(x) = \frac{1}{0.6\sqrt{2\pi}} \exp(-(x-0.5)^2) * \frac{10}{10+\exp(10(x-0.25)}$ |

Σημειώνεται ότι στην περίπτωση των αρχικών συνθηκών pulse_1 και trig_1 πραγματοποιήθηκε λείανση των γωνιών ώστε να εξασφαλίζεται πλήρης παραγωγισιμότητα. Επιπλέον, επιθυμείται η επίλυση της εξίσωσης για πληθώρα συνοριακών συνθηκών και έτσι τα σημεία που συμμετέχουν στις διαδικασίες Gauss και στην εκπαίδευση του νευρωνικού λαμβάνονται μακριά από τα άκρα, ώστε να μην επηρεάζονται από τις συνοριακές συνθήκες.

Επιπλέον, η διακριτοποίηση του χώρου έγινε με κανονικό πλέγμα (16 ισαπέχοντα σημεία), ενώ κάποιες προσομοιώσεις έτρεξαν με εξαιρετικά λεπτό πλέγμα (101 σημεία). Σε κάθε περίπτωση, οι παρατηρήσεις διατηρούνται σε σημεία με συνταγμένες στο διάστημα x ∈ [0.1, 0.9]. Οι χρονικές στιγμές για τις οποίες λήφθηκαν τιμές ανήκουν σε λογαριθμική σειρά και είναι από $10^{-4}$ έως 1 και δίνονται από 10^(-5:1:1). Η επιλογή τιμών της παραμέτρου λ έγινε τυχαία και σκοπό είχε να καλυφθεί όλο το εύρος τιμών.

Τα δεδομένα που εξήχθησαν είχαν τη μορφή τιμών του διανύσματος [x $u_t$ u $u_x$ $u_{xx}$ lamda]. Η διαδικασία παραγωγής δεδομένων από το COMSOL Multiphysics, από τη δημιουργία μοντέλου μέχρι την εξαγωγή δεδομένων, περιγράφεται αναλυτικά στο παράρτημα (6.3). Συνολικά παρήχθησαν 3,192 εξάδες δεδομένων [x $u_t$ u $u_x$ $u_{xx}$ lamda]. Σε τμήματα αυτών των δεδομένων εφαρμόστηκαν διεργασίες Gauss χρησιμοποιώντας ως συνάρτηση kernel τη 'Radial Basis Function' για να γίνει η ARD (Automatic Relevance Determination) ανάλυση. Κρίθηκε δηλαδή σκόπιμο να εφαρμοστούν διεργασίες Gauss σε επιμέρους μικρά τμήματα δεδομένων παρά στο σύνολό τους, λόγω μεγάλου υπολογιστικού κόστους επειδή ο χρόνος εκπαίδευσης διεργασίας Gauss αποτελεί συνάρτηση του κύβου του αριθμού χρησιμοποιούμενων δεδομένων: t = $O(n^3)$ (περιλαμβάνει αντιστροφή τετράγωνου πίνακα διαστάσεων n) καθώς ο αριθμός των χρησιμοποιούμενων δεδομένων αποδείχθηκε αρκετός για την εξαγωγή συμπεράσματος για τις χρήσιμες μεταβλητές.

### 2.2.2. Εκπαίδευση Νευρωνικού Δικτύου

Έπειτα, εκπαιδεύτηκε μοντέλο νευρωνικού δικτύου προκειμένου να χρησιμοποιηθεί ως συνάρτηση προσέγγισης του δεξιού σκέλους της διαφορικής εξίσωσης (2.1) και μελέτης της μεταβατικής κατάστασης, εύρεσης μόνιμων καταστάσεων τόσο στον ευσταθή όσο και τον ασταθή κλάδο αλλά και για την πραγματοποίηση της παραμετρικής ανάλυσης. Για την εκπαίδευση του νευρωνικού χρησιμοποιήθηκε η εξής πρόκληση:

**Μπορεί να εκπαιδευθεί το νευρωνικό μόνο με ένα είδος αρχικής συνθήκης ώστε να δίνει τα ορθά αποτελέσματα;**

Έτσι, χρησιμοποιήθηκε μόνο μια κυματική συνάρτηση: διάφορες παραλλαγές του ημιτόνου (a*sin(πx), a: μια πραγματική σταθερά), ενώ τα πειράματα έτρεξαν και για τυχαίες τιμές της παραμέτρου λ. Έτσι, ο χρόνος λήψης των μετρήσεων, η διακριτοποίηση των σημείων και ο τύπος της συνάρτησης δίνονται στον Πίνακα 2-2.

*Πίνακας 2-2. Συναρτήσεις, χρόνοι και πυκνότητες πλέγματος για παραγωγή δεδομένων στο COMSOL για το πρόβλημα Bratu 1D.*

| Συνάρτηση (Αρχική Συνθήκη) | Αρχικός χρόνος | Τελικός χρόνος | Βήμα | Πυκνότητα πλέγματος |
|---|---|---|---|---|
| sin(πx) | $10^{-3}$ | 10 | exp10(-5:1:1) | Normal |
| sin(πx) | 0.1 | 1 | 0.1:0.1:1 | Extremely Fine |
| -sin(πx) | $10^{-3}$ | 10 | exp10(-5:1:1) | Normal |
| -10*sin(πx) | $10^{-3}$ | 10 | exp10(-5:1:1) | Normal |

Οι χρονικές στιγμές προκύπτουν στις πρώτες περιπτώσεις με λογαριθμική βάση, για να εκπαιδευτεί το νευρωνικό σε μεγάλες κατά απόλυτη τιμή χρονικά δυναμικά, ενώ η προσθήκη της τρίτης περίπτωσης είναι για να δοθεί σημασία και στην ακριβή εκτίμηση της μόνιμης κατάστασης.

Τα χαρακτηριστικά του αρχικού μοντέλου παρατίθενται στον παρακάτω πίνακα 2-3:

*Πίνακας 2-3. Χαρακτηριστικά πρώτου νευρωνικού του προβλήματος Bratu 1D.*

| | |
|---|---|
| Όνομα Νευρωνικού | NetBratu1d1 |
| Αριθμός εσωτερικών κρυφών επιπέδων | 2 |
| Αριθμός νευρώνων (ανά εσωτερικό επίπεδο) | Πρώτο: 4\|δεύτερο: 4 |
| Συνάρτηση μεταφοράς εσωτερικών νευρώνων | Πρώτο: tansig\|δεύτερο: purelin |
| Αλγόριθμος εκπαίδευσης νευρωνικού | Levenberg Marquardt |
| Επιλογή δεδομένων για τα 3 sets | Τυχαίος Τρόπος |
| Συνολικός αριθμός δεδομένων εισόδου | 5601 |
| Αριθμός εποχών μέχρι τερματισμό εκπαίδευσης | 1505 |
| Λόγος τερματισμού εκπαίδευσης | Κριτήριο Εγκυρότητας (Validation Criterion) |
| Τιμή mse κατά τον τερματισμό | 0.0119 |

Έτσι, βρέθηκε η πρώτη εκτίμηση για τη χρονική παράγωγο. Ωστόσο, γίνεται και προσπάθεια βελτίωσης του νευρωνικού. Έτσι, το συγκεκριμένο νευρωνικό χρησιμοποιείται για να γίνει η παραμετρική

ανάλυση που αναφέρεται στην παράγραφο 2.2.4.. Από αυτή την ανάλυση, επιλέγονται 1 λύση (συγκεκριμένα για λ=0.5847) και η $u_0$=10*sin(πx) για λ=0.001. έχοντας αυτές ως αρχική συνθήκη, επαναλαμβάνονται προσομοιώσεις στο COMSOL και λαμβάνονται επιπλέον δεδομένα τα οποία συμπληρώνουν το αρχικό σύνολο δεδομένων. Τα δεδομένα των προσομοιώσεων φαίνονται στον Πίνακα 2-4.

*Πίνακας 2-4. Συναρτήσεις, χρόνοι και πυκνότητες πλέγματος για παραγωγή επιπλέον δεδομένων στο Comsol για το πρόβλημα Bratu 1D.*

| Συνάρτηση (Αρχική Συνθήκη) | Αρχικός χρόνος | Τελικός χρόνος | Βήμα | Πυκνότητα πλέγματος |
|---|---|---|---|---|
| Λύση NetBratu1d1 για λ=0.5874 | $10^{-5}$ | $5.0119*10^{-5}$ | exp10(-5:0.1:-4.3) | Normal |
| $u_0$=10*sin(πx) για λ=0.001 | $10^{-5}$ | $10^{-3}$ | exp10(-5:0.2:-3) | Normal |

Έπειτα, προστίθενται επιπλέον 216 λύσεις (που έπειτα από την εφαρμογή των διεργασιών Gauss της 3.1.1 από εξάδες γίνονται τετράδες) από το πείραμα για λ=0.5847 και 128 τετράδες για λ=0.001. Στη συνέχεια, γίνεται εκ νέου εκπαίδευση νευρωνικού με τα ακόλουθα χαρακτηριστικά.

Οι χρονικές στιγμές προκύπτουν στις πρώτες περιπτώσεις με λογαριθμική βάση, για να εκπαιδευτεί το νευρωνικό σε μεγάλα κατά απόλυτη τιμή χρονικά δυναμικά, ενώ η προσθήκη της τρίτης περίπτωσης είναι για να δοθεί σημασία και στην ακριβή εκτίμηση της μόνιμης κατάστασης.

Τα χαρακτηριστικά του τελικού μοντέλου παρατίθενται στον παρακάτω πίνακα 2-5:

*Πίνακας 2-5. Χαρακτηριστικά τελικού νευρωνικού του προβλήματος Bratu 1D.*

| Όνομα Νευρωνικού | NetBratu1d |
|---|---|
| Αριθμός εσωτερικών κρυφών επιπέδων | 2 |
| Αριθμός νευρώνων (ανά εσωτερικό επίπεδο) | Πρώτο: 5\|δεύτερο: 5 |
| Συνάρτηση μεταφοράς εσωτερικών νευρώνων | Πρώτο: tansig\|δεύτερο: tansig |
| Αλγόριθμος εκπαίδευσης νευρωνικού | Levenberg Marquardt |
| Συνολικός αριθμός δεδομένων εισόδου | 5945 |
| Επιλογή δεδομένων για τα 3 sets | Τυχαίος Τρόπος |
| Αριθμός εποχών μέχρι τερματισμό εκπαίδευσης | 8293 |
| Λόγος τερματισμού εκπαίδευσης | Κριτήριο Εγκυρότητας (Validation Criterion) |
| Τιμή mse κατά τον τερματισμό | 0.0141 |

### 2.2.3. Δυναμική Εξέλιξη του Φαινομένου

Το μοντέλο εφαρμόστηκε για την προσέγγιση λύσεων της (2.1) μέσω ode solver και εξετάστηκε η προσεγγιστική ικανότητά του, για διαφορετικές αρχικές συνθήκες, οι οποίες προσεγγίζουν την λύση της μόνιμης κατάστασης, και σε πιο πολύπλοκες αρχικές συνθήκες. Οι συναρτήσεις που χρησιμοποιήθηκαν ως αρχικές συνθήκες φαίνονται στον Πίνακα 2-6, ενώ αξίζει να τονιστεί ότι η επίλυση του μεταβατικού προβλήματος τόσο στην περίπτωση του νευρωνικού όσο και της λύσης με κλασικές αριθμητικές μεθόδους έγινε με ιδίους κώδικες που βρίσκονται στο παράρτημα 8. Παράλληλα, γίνεται επίλυση με 51 σημεία, που ουσιαστικά είναι τα **υπερτριπλάσια** από τα περισσότερα πειράματά που

χρησιμοποιήθηκαν για την εκπαίδευση του νευρωνικού. Δηλαδή, εξετάζεται και η ικανότητα υπολογισμού ενός πιο πυκνού πεδίου σε σχέση με το αδρομερές αρχικό.

*Πίνακας 2-6. Εξισώσεις αρχικών συνθηκών των προσομοιώσεων επαλήθευσης και χρονικά διαστήματα λήψης τιμών του προβλήματος Bratu 1D.*

| **Αρχική Συνθήκη / Περιγραφή** | **Εξίσωση** | **Τιμή λ** | **Χρόνοι** |
|---|---|---|---|
| Πολυωνυμική συνάρτηση | $y(x) = 0.5x^6 + 1.5\,x^5 - 2.5x^4 - 7.5x^3 + 2x^2 + 6x$ | 1.5 | [ 0.01, 0.1, 0.2, 1] |
| Γκαουσιανή κατανομή | $y(x) = \frac{1}{0.15\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{x-0.5}{0,15}\right)^2\right)$ | 2 | [ 0.01, 0.1, 0.2, 1] |
| Μηδενική Συνάρτηση | $y(x) = 0$ | 3 | [ 0.01, 0.1, 0.2, 1] |

### 2.2.4. Εύρεση Μόνιμης Κατάστασης και Παραμετρική Ανάλυση με Νευρωνικό Δίκτυο και Σύγκριση με Υπολογιστικές Τεχνικές

Παρόλο που αναφέρθηκαν στην 1.4 ενότητα τα θετικά της ύπαρξης κάποιου αλγορίθμου προ συνθήκης, στα πλαίσια της εργασίας δεν χρησιμοποιήθηκε.

Το τεχνητό νευρωνικό δίκτυο , το οποίο εκπαιδεύτηκε με τη διαδικασία της ενότητας 2.2.2., χρησιμοποιήθηκε για την προσέγγιση της μόνιμης κατάστασης του προβλήματος Bratu εφαρμόζοντας την αριθμητική μέθοδο Newton-GMRES και της παραμετρικής ανάλυσης αυτού ως προς την παράμετρο λ συνδυάζοντας την παραπάνω αριθμητική μέθοδο με εκείνη της συνέχειας ψευδο- μήκους τόξου, γνωστή και ως μέθοδος Keller.

## 2.3. Μεθοδολογία Δισδιάστατου προβλήματος Bratu

### 2.3.1. Προσομοιώσεις και Γκαουσιανές Διαδικασίες στο πρόβλημα Bratu 2D

Η εξίσωση, που μελετάται στην περίπτωση του δισδιάστατου προβλήματος, λαμβάνει την ακόλουθη μορφή (2.2):

$$\boldsymbol{f\left(u, u_{xx}, u_{yy}, \lambda\right) = \frac{\partial u}{\partial t} = \Delta u + \lambda e^{u} = u_{xx} + u_{yy} + \lambda e^{u}, x \in (0,1), y \in (0,1),} \quad (2.2)$$

$$\boldsymbol{u(x,0,t) = u(x,1,t) = u(0,y,t) = u(1,y,t) = 0\ (συνοριακές\ συνθήκες),} \quad (2.2)$$

$$\boldsymbol{u(x,y,0) = g(x,y)\ (αρχική\ συνθήκη).} \quad (2.2)$$

Η εξίσωση 2.2 μπορεί να επιλυθεί με πολλές τεχνικές, όπως είναι οι σειρές Chebyshev, με πεπερασμένες διαφορές και άλλες. Ωστόσο, επιλέγεται να επιλυθεί με τη μέθοδο των πεπερασμένων στοιχείων όπως αυτή εφαρμόζεται στο λογισμικό COMSOL Multiphysics 5.2. και συγκεκριμένα στην κατηγορία του μεταβατικής κατάστασης προβλήματος και γενική μορφή της μερικής διαφορικής εξίσωσης. Η διακριτοποίηση του χωρίου έγινε σε τετράγωνα στοιχεία και όχι σε τρίγωνα για να συμφωνεί με τα αποτελέσματα των κωδίκων του παραρτήματος 6. Έπειτα, προστίθενται οι συνοριακές

συνθήκες, οι οποίες είναι τύπου Dirichlet, όπως περιγράφεται και στην ενότητα 1.2.1.. Τέλος, ορίζονται οι αρχικές συνθήκες, δηλαδή οι τιμές της u για κάθε την στιγμή 0. Χρησιμοποιήθηκαν διαφορετικές αρχικές συνθήκες, οι οποίες παρουσιάζονται στον Πίνακα 2-7.

*Πίνακας 2-7. Συναρτήσεις των αρχικών συνθήκων που χρησιμοποιήθηκαν για τη συλλογή των δεδομένων για το δισδιάστατο πρόβλημα Bratu*

| **Αρχική Συνθήκη / Περιγραφή** | **Εξίσωση** |
|---|---|
| Κυματικός παλμός I (sin_sin_1) | $z(x) = 2 * \sin(\pi x) * \sin(\pi y)$ |
| Κυματικός παλμός II (sin_sin_2) | $z(x) = -\sin(\pi x) * \sin(\pi y)$ |
| Κυματικός παλμός III (sin_sin_3) | $z(x) = 4 * \sin(\pi x) * \sin(\pi y)$ |
| Άπειρο Παραβολοειδές I ( inf_paraboloid_1) | $z(x) = 4 * (x^2 + y^2)$ |
| Υπερβολικό παραβολοειδές (hyper_ par_1) | $z(x) = 4 * x * y$ |
| Μεικτή 1 (mix_1) | $z(x) = 4 - x^2 + y^2 * \sin(\pi xy)$ |

Επιθυμείται η επίλυση της εξίσωσης για πληθώρα συνοριακών συνθηκών και έτσι τα σημεία που συμμετέχουν στις διαδικασίες Gauss και στην εκπαίδευση του νευρωνικού λαμβάνονται μακριά από τα άκρα, ώστε να μην επηρεάζονται από τις συνοριακές συνθήκες

Επιπλέον, η διακριτοποίηση του χώρου έγινε με κανονικό πλέγμα (17x17), ωστόσο δεδομένα λήφθηκαν σε κάθε περίπτωση για ζεύγη τιμών που βρίσκονται στο τετραγωνικό χωρίο (x,y) ∈ [0.1, 0.9]x [0.1,0.9] με βήμα 0.05 για κάθε μια από τις συντεταγμένες, με αποτέλεσμα να λαμβάνονται 289 δεδομένα για κάθε προσομοίωση και στιγμή. Οι χρονικές στιγμές για τις οποίες λήφθηκαν τιμές ανήκουν σε λογαριθμική σειρά και είναι από $10^{-5}$ έως 1 (για τις πρώτες τρεις συναρτήσεις) και δίνονται από 10^(-5:0.5:0), ενώ για τις 3 τελευταίες ανήκουν από $10^{-4}$ έως 10 και δίνονται από 10^(-4:0.5:1).

Τα δεδομένα που εξήχθησαν είχαν τη μορφή τιμών του διανύσματος [x y $u_t$ u $u_x$ $u_y$ $u_{xx}$ $u_{yy}$ $u_x$ $u_x$ lamda]. Η διαδικασία παραγωγής δεδομένων από το COMSOL Multiphysics, από τη δημιουργία μοντέλου μέχρι την εξαγωγή δεδομένων, περιγράφεται αναλυτικά στο παράρτημα (6.3). Συνολικά παρήχθησαν 123,981 δεκάδες δεδομένων [x y $u_t$ u $u_x$ $u_y$ $u_{xx}$ $u_{yy}$ $u_x$ $u_x$ lamda]. Η μεγάλη αύξηση στην ποσότητα των δεδομένων, δικαιολογείται από την πολυπλοκότητα της δισδιάστατης γεωμετρίας (επιλογή της ίδιας ποσότητας δεδομένων από τον κάθε άξονα με εκείνη του 1D προβλήματος, οδηγεί στην συλλογή του τετραγώνου του αριθμού αυτού, από την ύπαρξη μιας επιπλέον σημαντικής μεταβλητής ($u_{yy}$) και από τον περίπου διπλασιασμό τιμών της παραμέτρου λ, οπότε και υπάρχει μεγαλύτερο εύρος που πρέπει να καλυφθεί. Σε τμήματα αυτών των δεδομένων εφαρμόστηκαν διεργασίες Gauss χρησιμοποιώντας ως συνάρτηση kernel τη 'Radial Basis Function'. Για την επιλογή των τμημάτων που εφαρμόστηκαν οι GP ισχύουν τα ίδια με την παράγραφο 2.2.1.

### 2.3.2. Εκπαίδευση Νευρωνικού Δικτύου 2D και Δυναμική Εξέλιξη του Φαινομένου

Έπειτα, εκπαιδεύτηκε μοντέλο νευρωνικού δικτύου χρησιμοποιώντας τα δεδομένα που παρήχθησαν με την προηγούμενη διαδικασία προκειμένου να χρησιμοποιηθεί ως συνάρτηση προσέγγισης του δεξιού σκέλους της διαφορικής εξίσωσης (2.2) και μελέτης της μεταβατικής κατάστασης, εύρεσης μόνιμων καταστάσεων τόσο στον ευσταθή όσο και τον ασταθή κλάδο αλλά και για την πραγματοποίηση της παραμετρικής ανάλυσης. Σε αυτή την περίπτωση δεν υπολογίσθηκαν άλλα δεδομένα αλλά χρησιμοποιήθηκε τμήμα των αρχικών (Πίνακας 2-8). Αυτό έγινε με την εξής διαδικασία:

Αρχικά, εκπαιδεύτηκε νευρωνικό με δεδομένα από μεγάλους χρόνους (που προσεγγίζουν την μόνιμη κατάσταση, ήτοι τις μικρές τιμές συνάρτησης του ευσταθούς κλάδου). Αυτό είχε σαν αποτέλεσμα, το νευρωνικό να εκτιμάει ακριβώς τις μικρές τιμές. Έπειτα, η επανεκπαίδευση (re-training) έγινε με δεδομένα από τμήματα (batches) που ανήκουν σε προηγούμενες χρονικές στιγμές, που βρίσκονται στην δυναμική φάση του φαινομένου.

Το τμήματα ήταν μοιρασμένα σε υποομάδες των 20,000 λύσεων , ενώ ιδιαίτερα χρησιμοποιήθηκε το τμήμα αρχικών δεδομένων του απείρου παραβολοειδούς, διότι αυτό περιλαμβάνει μεγάλες τιμές u που θα επιτρέψουν στο νευρωνικό να προσεγγίσει τον ασταθή κλάδο.

*Πίνακας 2-8. Συναρτήσεις, χρόνοι και πυκνότητες πλέγματος για παραγωγή δεδομένων στο COMSOL για το πρόβλημα Bratu 2D.*

| Συνάρτηση (Αρχική Συνθήκη) | Αρχικός χρόνος | Τελικός χρόνος | Βήμα | Πυκνότητα πλέγματος |
|---|---|---|---|---|
| $z(x) = 2 * \sin(\pi x) * \sin(\pi y)$ | $10^{-5}$ | 1 | exp10(-5:0.5:0) | Normal |
| $z(x) = -\sin(\pi x) * \sin(\pi y)$ | $10^{-5}$ | 1 | exp10(-5:0.5:0) | Normal |
| $z(x) = 4 * \sin(\pi x) * \sin(\pi y)$ | $10^{-5}$ | 1 | exp10(-5:0.5:0) | Normal |
| $z(x) = 4 * (x^2 + y^2)$ | $10^{-4}$ | 10 | exp10(-4:0.5:1) | Normal |
| $z(x) = 4 * x * y$ | $10^{-4}$ | 10 | exp10(-4:0.5:1) | Normal |
| $z(x) = 4 - x^2 + y^2 * \sin(\pi xy)$ | $10^{-4}$ | 10 | exp10(-4:0.5:1) | Normal |

Τα χαρακτηριστικά του τελικού μοντέλου παρατίθενται στον Πίνακα 2-9:

*Πίνακας 2-9. Χαρακτηριστικά νευρωνικού του προβλήματος Bratu 2D.*

| | |
|---|---|
| Όνομα Νευρωνικού | NetBratu2D |
| Αριθμός εσωτερικών κρυφών επιπέδων | 2 |
| Αριθμός νευρώνων (ανά εσωτερικό επίπεδο) | Πρώτο: 10\|δεύτερο: 10 |
| Συνάρτηση μεταφοράς εσωτερικών νευρώνων | Πρώτο: tansig\|δεύτερο: tansig |
| Αλγόριθμος εκπαίδευσης νευρωνικού | Levenberg Marquardt |
| Επιλογή δεδομένων για τα 3 sets | Τυχαίος Τρόπος |
| Συνολικός αριθμός δεδομένων εισόδου (σε όλη την εκπαίδευση) | 18368 |
| Λόγος τερματισμού εκπαίδευσης | Κριτήριο Εγκυρότητας (Validation Criterion) |
| Τιμή mse κατά τον τερματισμό | 0.0508 |

Έτσι, βρέθηκε η πρώτη εκτίμηση για τη χρονική παράγωγο, η οποία σε αυτή την περίπτωση κρίθηκε ικανοποιητική και δεν απαιτήθηκε επιπλέον εκπαίδευση. Αξίζει να τονιστεί πόσο σημαντική είναι η διαχείριση των δεδομένων, καθώς ικανοποιητικά αποτελέσματα είναι δυνατόν να εξαχθούν με χρήση μόνο του 15% του συνολικού αρχείου.

### 2.3.3 Δυναμική Εξέλιξη του Φαινομένου

Το μοντέλο εφαρμόστηκε για την προσέγγιση λύσεων της (2.2) μέσω ode solver και εξετάστηκε η προσεγγιστική ικανότητά του, για διαφορετικές αρχικές συνθήκες, οι οποίες προσεγγίζουν την λύση της μόνιμης κατάστασης, και σε πιο πολύπλοκες αρχικές συνθήκες. Η συνάρτηση που χρησιμοποιήθηκε ως αρχική συνθήκη φαίνεται στον Πίνακα 2-10, ενώ αξίζει να τονιστεί ότι η επίλυση του μεταβατικού προβλήματος τόσο στην περίπτωση του νευρωνικού όσο και της λύσης με κλασικές αριθμητικές μεθόδους έγινε με ίδιους κώδικες που βρίσκονται στο παράρτημα 6.5.2.

*Πίνακας 2-10. Εξισώση αρχικών συνθηκών των προσομοιώσεων επαλήθευσης και χρονικά διαστήματα λήψης τιμών του προβλήματος Bratu 2D.*

| **Αρχική Συνθήκη / Περιγραφή** | **Εξίσωση** | **Τιμή λ** | **Χρόνοι** |
|---|---|---|---|
| Ημιτονοειδής συνάρτηση | $z = \cos(\pi x) * \sin(\pi y)$ | 3.6 | [0.1, 1] |

Η συνάρτηση αυτή θεωρείται ικανή να παρουσιάσει την ικανότητα του νευρωνικού καθώς περιέχει τόσο αρνητικές, όσο και θετικές τιμές, ενώ η τιμή της παραμέτρου οδηγεί σε μέγιστη τιμή μικρότερη του 1, οπότε δείχνει και θετική και αρνητική δυναμική. Επομένως, χρησιμοποιείται σαν πρότυπη συνάρτηση.

Η διακριτοποίηση έγινε σε τετράγωνα στοιχεία, που αντιστοιχεί σε 21 ισαπέχοντα σημεία στον άξονα x και 21 ισαπέχοντα σημεία στον άξονα y. Αυτό οδήγησε σε 441 κόμβους, από τους οποίους στους 80 εφαρμόζονται οι συνοριακές συνθήκες και στους υπόλοιπους 361 λύνονται οι εξισώσεις.

### 2.3.4 Εύρεση Μόνιμης Κατάστασης και Παραμετρική Ανάλυση με Νευρωνικό Δίκτυο και Σύγκριση με Υπολογιστικές Τεχνικές

Παρόλο που αναφέρθηκαν στην 1.4 ενότητα τα θετικά της ύπαρξης κάποιου αλγορίθμου προσταθεροποίησης, στο πλαίσιο της εργασίας δεν χρησιμοποιήθηκε.

Το τεχνητό νευρωνικό δίκτυο , το οποίο εκπαιδεύτηκε με τη διαδικασία της ενότητας 2.3.2., χρησιμοποιήθηκε για την προσέγγιση της μόνιμης κατάστασης του προβλήματος Bratu εφαρμόζοντας την αριθμητική μέθοδο Newton-GMRES και της παραμετρικής ανάλυσης αυτού ως προς την παράμετρο λ συνδυάζοντας την παραπάνω αριθμητική μέθοδο με εκείνη της συνέχειας ψευδο- μήκους τόξου, γνωστή και ως μέθοδος Keller.

Έγινε η ίδια διακριτοποίηση με εκείνη στην παράγραφο 2.3.3 για το χώρο της λύσης.

## 2.4. Εφαρμογή Μεθοδολογίας Σε Περιβάλλον Matlab

### 2.4.1. Εφαρμογή Διεργασιών Gauss

Για την εφαρμογή των Γκαουσιανών διαδικασιών χρησιμοποιείται ο κώδικας του παραρτήματος 6.3. Με το συγκεκριμένο κώδικα επιλέγονται από τα διανύσματα δεδομένων x, u, $u_t$, $u_x$ και $u_{xx}$ στην περίπτωση του 1D προβλήματος. Στην περίπτωση του 2D, τα διανύσματα δεδομένων είναι x, y, u, $u_x$, $u_y$, $u_{xx}$, $u_{yy}$, $u_{yx}$ και $u_{xy}$. Αξίζει να σημειωθεί ότι αυτά έχουν τη μορφή στήλης. Τα αποτελέσματα αυτά εκτυπώνονται από το λογισμικό COMSOL σε αρχείο Excel και έπειτα εισάγονται στο Matlab μέσω της επιλογής import data στο εικονίδιο variable που υπάρχει στην γραμμή εργασιών.

Έπειτα, εισάγονται με την μορφή αριθμητικού πίνακα, αφού λαμβάνονται δεδομένα για τις ίδιες χρονικές στιγμές σε κάθε προσομοίωση και επιλέγεται ο αποκλεισμός των σειρών που περιέχουν μη αριθμητικά δεδομένα, ώστε να μην υπάρχουν προβλήματα από τις περιγραφές στο αρχείο excel. Έπειτα, ο πίνακας διαιρείται σε μικρότερους πίνακες, αφού κάθε στήλη αντιστοιχεί σε τιμές μιας μεταβλητής (ανά 5 γραμμές ίδια μεταβλητή στο 1D, ενώ ανά 9 στο 2D) και τέλος μέσω της εντολής reshape, κάθε μεταβλητή αποτυπώνεται με ένα διάνυσμα στήλης και στην αντίστοιχη θέση κάθε διανύσματος υπάρχει η αντίστοιχη τιμή από τη προσομοίωση. Έπειτα, σχηματίζεται ο πίνακας mat_1 των μεταβλητών x, u, $u_x$ και $u_{xx}$ στο 1D πρόβλημα και ο πίνακας των μεταβλητών x, y, u, $u_x$, $u_y$, $u_{xx}$, $u_{yy}$, $u_{yx}$ και $u_{xy}$ στο 2D πρόβλημα , που θεωρείται ότι επηρεάζουν τη μεταβλητή $u_t$

Η διαδικασία Gauss πραγματοποιείται μέσω της συνάρτησης fitrgp, η οποία παράγει ένα μοντέλο παλινδρόμησης, χρησιμοποιώντας τα δεδομένα του πίνακα mat_1 με συνάρτηση πυρήνα (kernel) την ARD. Οι παράμετροι της συνάρτησης kernel του μοντέλου (πλην της τελευταίας) δίνουν την επίδραση κάθε μεταβλητής επί της μεταβλητής $u_t$. Μεγαλύτερες τιμές αντιστοιχούν σε μικρότερη συνεισφορά, καθώς σημαίνει ότι υπάρχει πολλή μεγάλη μεταβλητότητα της υπό εξεταζόμενης μεταβλητής σε σύγκριση με την μεταβλητή που είναι επιθυμητή η εύρεση της εξάρτησης από τις υπόλοιπες.

### 2.4.2. Εκπαίδευση Νευρωνικών Δικτύων

Το Παράρτημα 6.4 περιλαμβάνει τον κώδικα που εφαρμόσθηκε για την εκπαίδευση όλων των νευρωνικών δικτύων που παρήχθησαν στο πλαίσιο της συγκεκριμένης εργασίας, τόσο για το μονοδιάστατο όσο και για το δισδιάστατο πρόβλημα Bratu. Ο κώδικας δημιουργήθηκε αυτόματα χρησιμοποιώντας την εφαρμογή Neural Net Fitting της εργαλειοθήκης Neural Network Toolbox (έκδοση 11.1) του Matlab και κατόπιν έγιναν σε αυτόν οι απαραίτητες τροποποιήσεις. Στον κώδικα ορίζονται:

I. Τα δεδομένα εισόδου και εξόδου που θα χρησιμοποιηθούν για την εκπαίδευση

Τα δεδομένα αυτά ορίζονται οι πίνακες **X** και **T**, όπου:

**X**: Ο πίνακας μεταβλητών εισόδου. Αυτός μπορεί να αποτελείται από μία γραμμή (διάνυσμα γραμμής) ή να αποτελείται από πίνακα με 2 γραμμές και άνω. Γενικά, με το Matlab μπορεί να εκπαιδευθεί νευρωνικό δίκτυο με πολλές μεταβλητές εισόδου, αρκεί αυτές να έχουν τη μορφή διανύσματος γραμμής στον πίνακα μεταβλητών εισόδου.

**T**: ισχύουν τα αντίστοιχα με το x, μόνο που εδώ πρόκειται για τις μεταβλητές εξόδου.

II. Η μέθοδος εκπαίδευσης

Ως μέθοδος εκπαίδευσης χρησιμοποιείται η trainlm, η οποία εφαρμόζει τον αλγόριθμο εκπαίδευσης Levenberg-Marquardt backpropagation. Ο αλγόριθμος περιγράφεται αναλυτικά στον ενότητα 1.3.2.3. Ο συγκεκριμένος αλγόριθμος έχει το πλεονέκτημα ότι αποτελεί το γρηγορότερο της εργαλειοθήκης Neural Network Toolbox, για προβλήματα προσέγγισης συνάρτησης, με μέχρι μερικές εκατοντάδες συντελεστών βαρύτητας. Στο πλαίσιο της συγκεκριμένης εργασίας, ο αριθμός αυτός πληροί το κριτήριο των μερικών εκατοντάδων.

Επίσης, δοκιμάστηκε η εκπαίδευση και με τη μέθοδο απότομης κλίσης με επιτάχυνση και προσαρμόσιμο ρυθμό μάθησης (gradient descent with momentum and adaptive learning rate backpropagation), αλλά τα αποτελέσματα ήταν αρκετά αποθαρρυντικά ώστε να αναφερθεί ξεχωριστά σε άλλο κεφάλαιο.

III. Η αρχιτεκτονική του νευρωνικού δικτύου, δηλαδή ο αριθμός των εσωτερικών επιπέδων (hidden layers) και ο αριθμός των νευρώνων κάθε επιπέδου.

Κάθε επίπεδο συμβολίζεται με το γράμμα Η (από το hidden) και έναν αριθμό που υποδηλώνει ποιο κρυφό επίπεδο είναι. Τέλος, στην παραπάνω μεταβλητή ανατίθεται ένας αριθμός που είναι ο αριθμός των κόμβων / νευρώνων του επιπέδου.

IV. Η συνάρτηση προεπεξεργασίας (pre-processing) των σημάτων εισόδου και η συνάρτηση μετεπεξεργασίας (post-processing) των σημάτων εξόδου.

Αν δεν υπάρξει επεξεργασία των σημάτων εισόδου, τότε είναι πιθανό το ενδεχόμενο να εμφανιστούν πολύ μικρές κλίσεις κατά την εκπαίδευση του νευρωνικού, οδηγώντας σε περισσότερες επαναλήψεις και μεγαλύτερο υπολογιστικό χρόνο. Η αποφυγή τέτοιων φαινομένων, μπορεί να βοηθηθεί από την προεξεργασία των δεδομένων εισόδου, ώστε να κανονικοποιηθούν οι παρατηρήσεις μέσα σε ένα συγκεκριμένο εύρος, και την μετεπεξεργασία των σημάτων εξόδου, ώστε το αποτέλεσμα να γυρίσει στην τάξη μεγέθους των σημάτων εισόδου. Η συνάρτηση mapminmax πραγματοποιεί την προεπεξεργασία των σημάτων εισόδου, κανονικοποιώντας τα δεδομένα στο διάστημα [-1 1]. Έπειτα υλοποιεί και την μετεπεξεργασία, ώστε τα σήματα εξόδου να έχουν κλίμακα σύμφωνη με τα δεδομένα εισόδου.

V. Ο τρόπος και το μέγεθος της διαίρεσης των δεδομένων εισόδου σε ομάδες.

Τα δεδομένα που χρησιμοποιούνται για εκπαίδευση χωρίζονται σε τρεις ομάδες (sets):

Στην πρώτη ομάδα (training set) περιέχονται τα δεδομένα με τα οποία πραγματοποιείται η εκπαίδευση του νευρωνικού δικτύου.

Η δεύτερη ομάδα (validation set) χρησιμοποιείται για να τερματίζει την εκπαίδευση του νευρωνικού δικτύου όταν το σφάλμα επαλήθευσης δεν μειώνεται περαιτέρω.

Η τρίτη ομάδα (test set) μπορεί να περιέχει δεδομένα που για την εκπαίδευση θεωρούνται άγνωστα. Δεν επηρεάζουν την εκπαιδευτική διαδικασία, αλλά χρησιμοποιούνται για τον έλεγχο της ικανότητας γενίκευσης του δικτύου.

Συνήθης πρακτική και default επιλογή του Matlab είναι τα δεδομένα να χωρίζονται σε ποσοστά 70-15-15% (training-validation-test ) πρακτική που ακολουθήθηκε και σε αυτή την εργασία.

VI. Η συνάρτηση σφάλματος.

Εν προκειμένω χρησιμοποιείται η συνάρτηση mean squared error (mse) / μέσο τετραγωνικό σφάλμα, όπως περιγράφεται στη θεωρία.

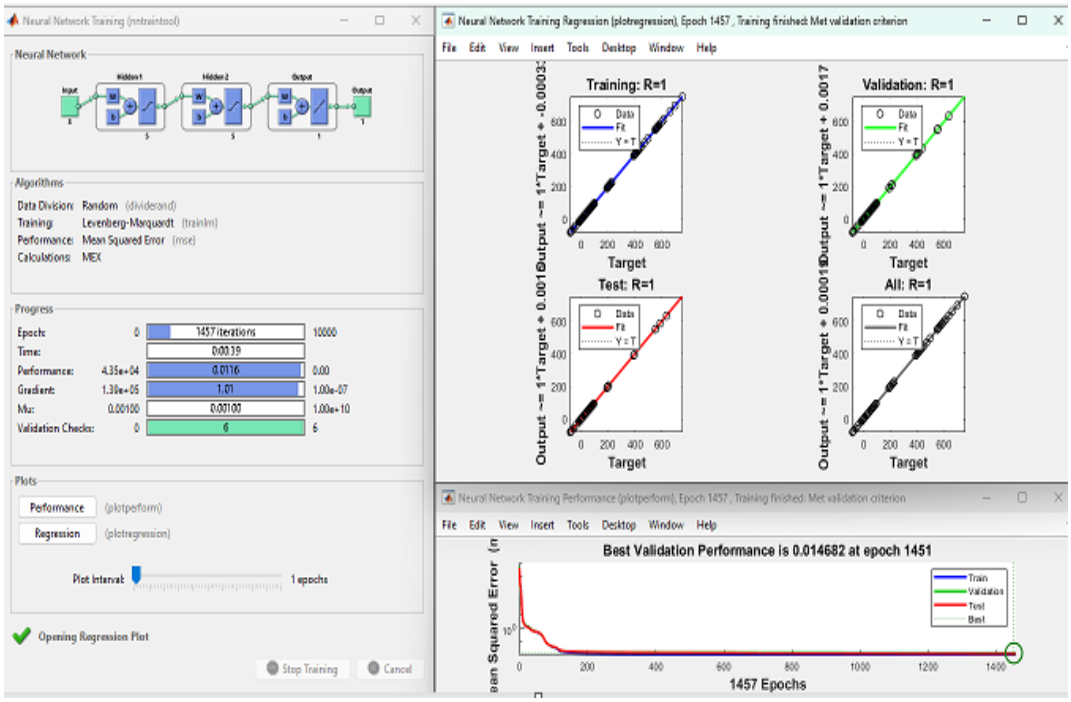
VII. Οι συναρτήσεις ενεργοποίησης κάθε εσωτερικού επιπέδου.

Σε κάθε νευρωνικό δίκτυο που εκπαιδεύτηκε σε αυτή την εργασία χρησιμοποιήθηκε η σιγμοειδής συνάρτηση tansig, δηλαδή η συνάρτηση υπερβολικής εφαπτομένης, αλλά και η συνάρτηση purelin, η οποία είναι ουσιαστικά η γραμμική συνάρτηση. Οι συναρτήσεις δίνονται από τις ακόλουθες σχέσεις (2.2 - 2.3).:

$$tansig(x) = \tanh(x) = \frac{2}{1 - e^{-2x}} - 1 \tag{2.2}$$

$$purelin(x) = x \tag{2.3}$$

Η εκπαίδευση πραγματοποιείται με τη συνάρτηση train χρησιμοποιώντας τις παραπάνω επιλογές. Η συνάρτηση ανοίγει επίσης διεπιφάνεια επαφής (GUI) η οποία παρέχει πληροφορίες (όπως ο αριθμός επανάληψης/εποχής (epoch), η τιμή του σφάλματος (performance) καθώς και κάποιες παράμετροι του αλγορίθμου Levenberg-Marquardt (gradient, mu)) και έχει τη δυνατότητα εμφάνισης επιλεγμένων διαγραμμάτων (η επιλογή πραγματοποιείται στον κώδικα με χρήση των συναρτήσεων plotperform και plotregression). Η διεπιφάνεια εκπαίδευσης και τα διαγράμματα παρουσιάζονται στο Σχήμα 2-1.




*Σχήμα 2-1. (Αριστερά) Διεπιφάνεια εκπαίδευσης νευρωνικού δικτύου. (Πάνω δεξιά) Διάγραμμα παλινδρόμησης χωρισμένο σε τέσσερα τμήματα (Training, Validation, Test και αθροιστικό). (Κάτω δεξιά) Διάγραμμα τετραγωνικού σφάλματος για τις τρεις ομάδες δεδομένων.*

### 2.4.3. Επίλυση Μερικής Διαφορικής Εξίσωσης Με Χρήση Μοντέλου Νευρωνικού Δικτύου Και ode solver

Στο Παράρτημα 6.5 παρατίθεται ο κώδικας που υλοποιεί την επίλυση διαφορικής εξίσωσης μέσω εκπαιδευμένου μοντέλου νευρωνικού δικτύου. Το μοντέλο υπολογίζει της τιμές ∂u/∂t της εξίσωσης (2.1, 2.2) λαμβάνοντας ως σήματα εισόδου τιμές των μεταβλητών u και $u_{xx}$ και της παραμέτρου λ. στην περίπτωση του 1D, και των u, $u_{xx}$, $u_{yy}$, λ για το 2D. Σε αυτούς ορίζονται οι αρχικές συνθήκες ($u_0$), το διάστημα της ανεξάρτητης μεταβλητής x, y στο οποίο πραγματοποιείται επίλυση μέσω ode solver καθώς και το χρονικό διάστημα επίλυσης. Ως ode solver χρησιμοποιήθηκε η συνάρτηση ode15s του Matlab, η οποία δέχεται ως ορίσματα τη συνάρτηση υπολογισμού του χρονικού διαφορικού εκφρασμένη ως function handle (f = @(t,y)), το χρονικό διάστημα επίλυσης και τις αρχικές συνθήκες. Η συνάρτηση παράγει ένα πίνακα [t,y], με την πρώτη στήλη να περιέχει τις χρονικές στιγμές στις οποίες απαιτείται εξαγωγή δεδομένων και τόσες επιπλέον στήλες, όσες τα σημεία του διαστήματος x, οι οποίες περιέχουν τις υπολογισμένες τιμές του u.

### 2.4.4. Εύρεση Μόνιμης Κατάστασης Και Παραμετρική Ανάλυση

Η εύρεση της μόνιμης κατάστασης στο σύστημα που εξετάζεται πραγματοποιείται στο Matlab χρησιμοποιώντας τους κώδικες που επισυνάπτονται στο παράρτημα 6.6. Ο πρώτος κώδικας καλεί τη συνάρτηση gmres η οποία δέχεται ως ορίσματα μία αρχική πρόβλεψη της λύσης $x_o$, τη συνάρτηση υπολογισμού του residual (-Res) καθώς και μία συνάρτηση atv (εκφρασμένη ως function handle) η οποία υπολογίζει το γινόμενο του Ιακωβιανού πίνακα με διάνυσμα x. Η συνάρτηση αυτή δέχεται επίσης τρεις παραμέτρους που καθορίζουν την ακρίβεια (εκφρασμένη ως μείωση του αρχικού residual, χρησιμοποιήθηκε η τιμή $10^{-3}$), το μέγιστο αριθμό επαναλήψεων (ίσος με τον αριθμό των σημείων που γίνεται η διακριτοποίηση του πεδίου ορισμού της εξίσωσης) και την ύπαρξη και το είδος μεθόδου ορθογωνοποίησης. Πιο συγκεκριμένα, οι κώδικες που χρησιμοποιούνται είναι οι ακόλουθοι:

gmres.m : είναι ο βασικός κώδικας της gmres. Η gmres χρησιμοποιεί και το βοηθητικό function, givapp.

Ο gmres καλεί τη συνάρτηση atv.m που πραγματοποιεί την πράξη A*x, με A τον πίνακα του γραμμικού συστήματος που επιθυμείται η λύση του και x ένα input vector.

dirder.m: Η atv καλεί τη συνάρτηση dirder η οποία υπολογίζει το directional derivative, δηλαδή κάνει την πράξη J*w, όπου J είναι ο Ιακωβιανός πίνακας.

Resi_par.m: υπολογίζει τα residuals του προβλήματός. Η παραλλαγή Resi_par_2D υπολογίζει τα προβλήματα του 2D προβλήματος.

LC_comp.m: είναι το βασικό script για την επίλυση του μη γραμμικού προβλήματος. Επιλύεται το Bratu με νευρωνικά. Γενικά, το LC_comp φορτώνει το νευρωνικό δίκτυο, μια αρχική εκτίμηση της λύσης (sol1). Η λύση στο τελευταίο στοιχείο της έχει την παράμετρο (δηλαδή τα πρώτα στοιχεία είναι το solution vector και το τελευταίο η τιμή της παραμέτρου).

Οι κώδικες των συναρτήσεων gmres, atv καθώς και givapp και dirder που αξιοποιούνται εσωτερικά της συνάρτησης gmres ελήφθησαν από τη βιβλιογραφία [43]. Η συνάρτηση gmres παράγει για κάθε εξωτερική επανάληψη ένα διάνυσμα dx, τη μεταβολή δηλαδή του διανύσματος της συνάρτησης (J*δu =

-R) και ο κώδικας σταματά όταν έχει επιτευχθεί η επιθυμητή ακρίβεια (υπάρχουν επίσης περιπτώσεις απόκλισης (στο άπειρο) ή μη σύγκλισης (μη μείωση του σφάλματος κάτω από ένα όριο λόγω ταλάντωσης) στις οποίες ο χρήστης πρέπει να σταματήσει την εκτέλεση κώδικα).

Τέλος, για την πραγματοποίηση της παραμετρικής ανάλυσης, χρησιμοποιείται ο ίδιος κώδικας, μόνο που πλέον το διάνυσμα της λύσης έχει μια παραπάνω διάσταση, εκείνη της παραμέτρου λάμδα (λ). Έτσι, χρησιμοποιούνται οι ίδιοι κώδικες, που τώρα περιλαμβάνουν και την εξίσωση της μεθόδου Keller. Αξίζει να τονιστεί, ότι η παράγωγος σε κάθε νέο σημείο υπολογίζεται με τη μέθοδο των πεπερασμένων διαφορών, καθώς βρέθηκε ότι επιτυγχάνεται καλύτερη ευστάθεια στο σύστημα για οποιοδήποτε ds δοθεί για παραμετρική ανάλυση.

Σε όλες τις μεθόδους ως κριτήριο σύγκλισης χρησιμοποιείται η διαφορά των λύσεων στην Newton-Raphson να είναι μικρότερη του $10^{-6}$.

# 3. Παρουσίαση και σχολιασμών αποτελεσμάτων

## 3.1. Αποτελέσματα Μονοδιάστατου προβλήματος Bratu

### 3.1.1. Γκαουσιανές Διαδικασίες (Gaussian Processes)

Τα αποτελέσματα των Γκαουσιανών διαδικασιών παρουσιάζονται στον Πίνακα 3-1.

*Πίνακας 3-1. Αποτελέσματα ARD ανάλυσης διεργασιών Gauss του προβλήματος Bratu 1D.*

| Μεταβλητή | Τιμή Παραμέτρου (ARD) |
|---|---|
| **u** | **2.7e-03** |
| $u_x$ | 7.7e+07 |
| **$u_{xx}$** | **8.1e+01** |
| **λ** | **4.5e+01** |
| x (χωρική συντεταγμένη) | 1.9e+07 |

Παρατηρείται εύκολα ότι η μέθοδος δίνει σαν σημαντικές μεταβλητές την τιμής της συνάρτησης, την παράμετρο λ και τη δεύτερη χωρική παράγωγο. Αν και υπάρχει και διαφορά στην τάξη μεγέθους των $u_{xx}$ και της παραμέτρου λ (αυτές έχουν την ίδια) από την τιμή της συνάρτησης u (4 τάξεις κάτω) δεν θεωρείται τόσο μεγάλη η διαφορά καθώς πρόκειται για μικρές σχετικά τιμές. Αντίθετα, οι μεταβλητές $u_{xx}$ και x παρουσιάζουν πολύ μεγάλη τιμή που διαφέρει τουλάχιστον 6 τάξεις μεγέθους από τις επιλεγμένες σημαντικές και για αυτό τον λόγο θεωρούνται αμελητέες. Πρέπει, λοιπόν, να τονιστεί η αξία της παραπάνω μεθόδου: χωρίς να γνωρίζει την φυσική του προβλήματος, κατάφερε να εντοπίσει τις μεταβλητές που βρίσκονται στο δεξί μέρος της διαφορικής εξίσωσης, βασιζόμενο μόνο σε πειραματικά δεδομένα (δεδομένα προσομοιώσεων).

Άρα, με οδηγό τις Gaussian Processes, προχωράει κανείς στην εκπαίδευση νευρωνικού δικτύου έχοντας τη γνώση ότι μπορεί να χρησιμοποιήσει σαν μεταβλητές εισόδου μόνο τις 3 (u, $u_{xx}$, λ) από τις 5 (u, $u_x$, $u_{xx}$, λ, x) μεταβλητές, περιορίζοντας έτσι κατά πολύ (40%) το σύνολο των δεδομένων που χρειάζονται για την εκπαίδευση του νευρωνικού δικτύου.

Αξίζει να σημειωθεί ότι έγινε χρήση μόλις των 500 πρώτων παρατηρήσεων, γεγονός που αντιστοιχεί στις πρώτες τιμές που λαμβάνει ο πειραματιστής αν ακολουθηθεί η λογική που παρουσιάστηκε κατά την πειραματική διαδικασία. Στις τιμές αυτές περιλαμβάνονται πειράματα από όλες τις αρχικές συνθήκες και διαφορετικές τιμές της παραμέτρου λάμδα.

Επίσης, παρατηρήθηκε ότι το σύστημα είναι πολύ ευαίσθητο στην προσθήκη επιπλέον σημείων / παρατηρήσεων στην ομάδα τιμών με την οποία γίνεται η εκπαίδευση των διαδικασιών Gauss. Αυτό έχει ως αποτέλεσμα την αλλαγή των άνωθι τιμών, οι οποίες βρίσκονται όλες στην ίδια τάξη μεγέθους, γεγονός που αποτρέπει την εύρεση των σημαντικών μεταβλητών του προβλήματος και έτσι την

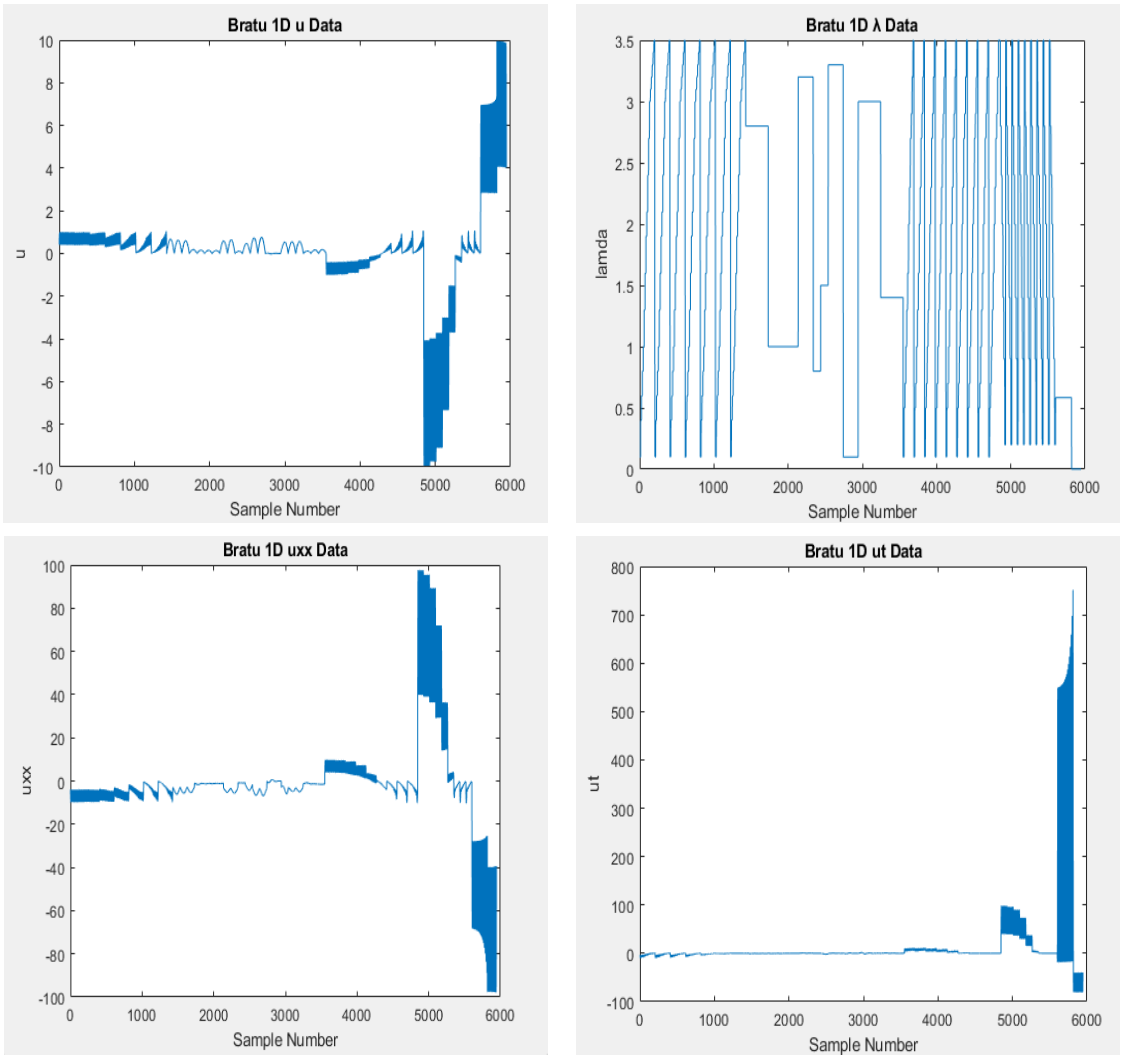
κατανόηση της φυσικής που το διέπει. Αποδίδεται στο γεγονός ότι γίνεται προσθήκη μικρότερων τιμών της χρονικής παραγώγου και έτσι δυσκολεύει την εύρεση των σημαντικών μεταβλητών.

Έτσι, το συμπέρασμα το οποίο παράγεται από την προηγούμενη συλλογιστική πορεία είναι το ακόλουθο :

Για να μπορέσει κανείς να βρει τις σημαντικές παραμέτρους που διέπουν μια μερική διαφορική εξίσωση πρέπει να κινηθεί στην περιοχή των έντονων δυναμικών φαινομένων, σε περιοχές όπου η χρονική παράγωγος λαμβάνει συγκριτικά μεγάλες τιμές, το οποίο συνήθως συμβαίνει σε αρχικούς μικρούς χρόνους.

### 3.1.2. Εκπαίδευση Νευρωνικού Δικτύου και Δυναμική Εξέλιξη του Φαινομένου

Αξίζει να παρουσιαστεί σε αυτό το σημείο το σύνολο των δεδομένων που χρησιμοποιήθηκαν για την εκπαίδευση του νευρωνικού. Αυτές παρουσιάζονται στο Σχήμα 3-1.




*Σχήμα 3-1. (α)-(δ).Πάνω αριστερά (α): δεδομένα για τη μεταβλητή εισόδου u, πάνω δεξιά (β): δεδομένα για τη μεταβλητή εισόδου λ, κάτω αριστερά (γ): δεδομένα για τη μεταβλητή εισόδου $u_{xx}$ και κάτω δεξιά (δ): δεδομένα για τη μεταβλητή εξόδου $u_t$ που χρησιμοποιήθηκαν για την εκπαίδευση του νευρωνικού του προβλήματος Bratu 1D.*

Όπως γίνεται αντιληπτό, το σύστημα έχει εκπαιδευτεί σε σχετικά χαμηλές τιμές των μεταβλητών, ενώ η μεταβλητή λ παίρνει αρκετές τιμές μέσα στο σύνολο τιμών της. Οι τελικές 600 περίπου τιμές, όπου η τιμή των λύσεων είναι πολύ μεγάλη, είναι αυτές που οδηγούν στην απόλυτη ταύτιση της λύσης της μηχανικής μάθησης με εκείνη της μεθόδου των πεπερασμένων στοιχείων. Αυτό συμβαίνει γιατί ευρύνεται κυρίως το πεδίο τιμών της μεταβλητής u.

Ακολουθώντας τα βήματα που περιγράφονται στο κεφάλαιο 2.2.3., λαμβάνονται τα ακόλουθα διαγράμματα για κάθε μία από τις 3 αρχικές συνθήκες με διαφορετική τιμή παραμέτρου λ. Η επιλογή του νευρωνικού γίνεται μετά την ανανέωση της βάσης δεδομένων (dataset) από προσομοιώσεις με αρχική συνθήκη τις λύσεις του νευρωνικού στον ασταθή κλάδο. Η επιλογή επιβεβαιώνεται με τα διαγράμματα 3-8 και 3-9 στην ενότητα 3.1.4.

Επιπλέον, για την αξιολόγηση, εκτός από τον οπτικό έλεγχο, χρησιμοποιούνται και πιο φορμαλιστικά μαθηματικά μεγέθη, τα οποία είναι η τετραγωνική ρίζα του μέσου τετραγωνικού σφάλματος (root mean squared error) και μέσο ποσοστιαίο απόλυτο σφάλμα( mean absolute percentage error).

Η εξίσωση του rMSE και MAPE είναι:

$$rMSE = \sqrt{\frac{\sum_{i=2}^{i=total-1}\left(u_{NN_i} - u_{comp_i}\right)^2}{total\ points - 2}}, \tag{3.1}$$

$$MAPE = \sum_{i=2}^{i=total-1} \left|\frac{u_{NN_i} - u_{comp_i}}{u_{comp_i}}\right| * \frac{100}{total\ points - 2}, \tag{3.2}$$

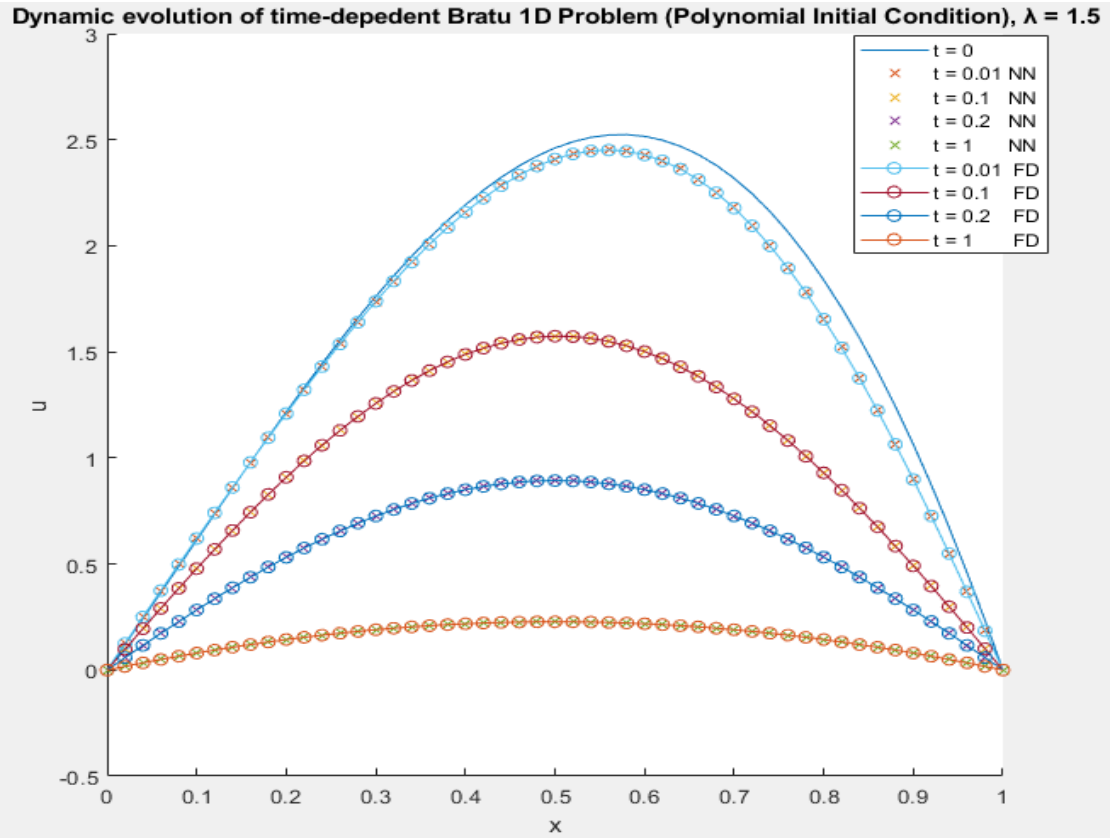
όπου: mape (mean relative absolute error): μέσο σχετικό απόλυτο σφάλμα

$u_{NN_i}$: η τιμή της συνάρτησης στο σημείο i, όπως αυτό υπολογίζεται από τη χρήση του νευρωνικού δικτύου.

$u_{comp_i}$: η τιμή της συνάρτησης στο σημείο i, όπως αυτό υπολογίζεται από την υπολογιστική μέθοδο των πεπερασμένων διαφορών/ πεπερασμένων στοιχείων.

Το πρώτο και το τελευταίο σημείο κρατούνται εκτός, αφού αποτελούν συνοριακή συνθήκη και δεν συμμετέχουν στον υπολογισμό του σφάλματος.

a. Έτσι, παρουσιάζονται οι εξής περιπτώσεις δυναμικών προσομοιώσεων για τις οποίες υπολογίζονται και τα αναφερθέντα μεγέθη για την ακρίβεια των νευρωνικών δικτύων, που θα βοηθήσει στην αξιολόγησή τους (πόσο επιτυχής είναι η εκπαίδευση του νευρωνικού). Συγκεκριμένα οι περιπτώσεις αυτές αναφέρονται στη χρήση διαφορετικών αρχικών συνθηκών: Πολυωνυμική Συνάρτηση 6ου βαθμού, Γκαουσιανή Κατανομή, και Μηδενική Συνάρτηση (Σχήματα 3-2 με 3-4)

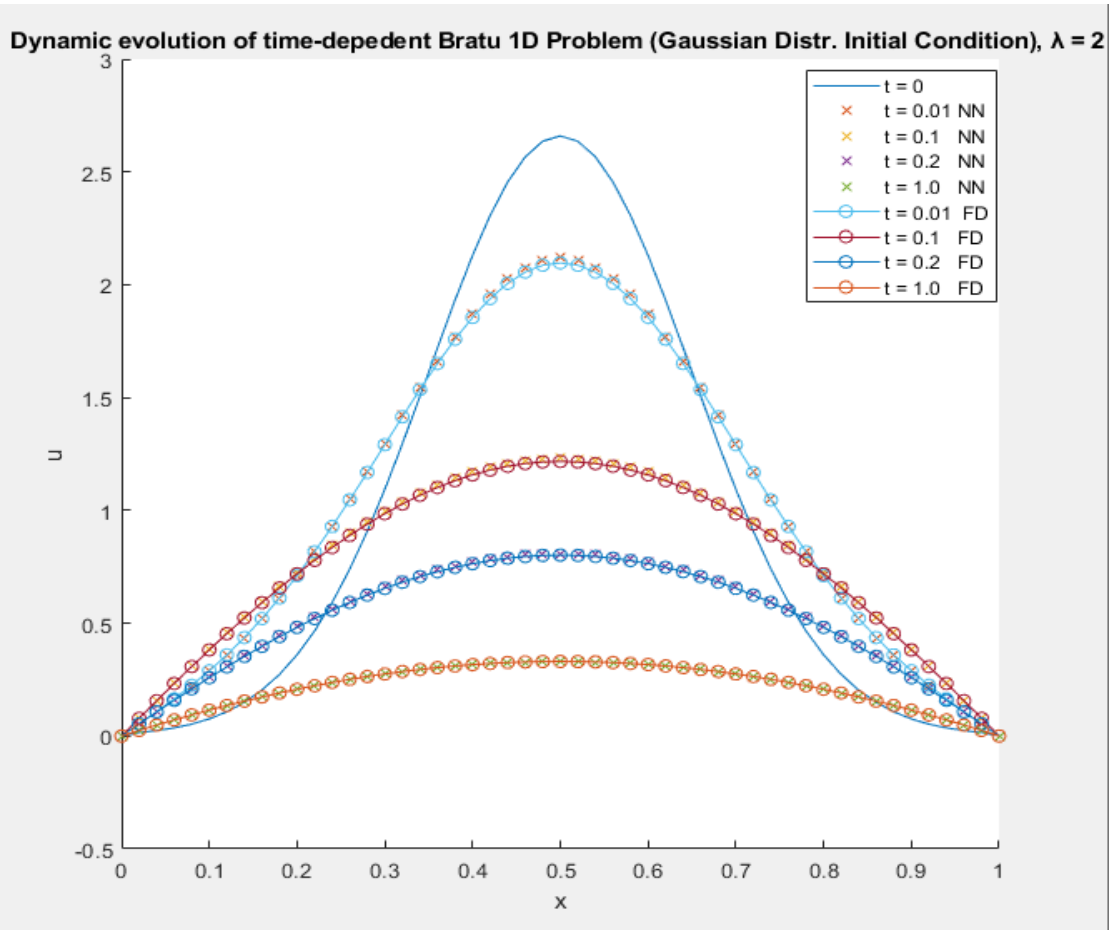



*Σχήμα 3-2. Στιγμιότυπα λύσης του προβλήματος Bratu 1D για λ=1.5 σε χρόνους 0.01, 0.1, 0.2, 1 όταν χρησιμοποιείται πολυωνυμική (6ου βαθμού) αρχική συνθήκη.*

*Πίνακας 3-2. Σύγκριση των μεθόδων (νευρωνικού δικτύου και πεπερασμένων διαφορών) για πολυωνυμική (6ου βαθμού) αρχική συνθήκη του προβλήματος Bratu 1D.*

| Time (t) | rMSE | MAPE |
|---|---|---|
| 0.0100 | 0.0076 | 0.0856 |
| 0.1000 | 0.0067 | 0.0916 |
| 0.2000 | 0.0023 | 0.0589 |
| 1 | 0.0025 | 0.1961 |

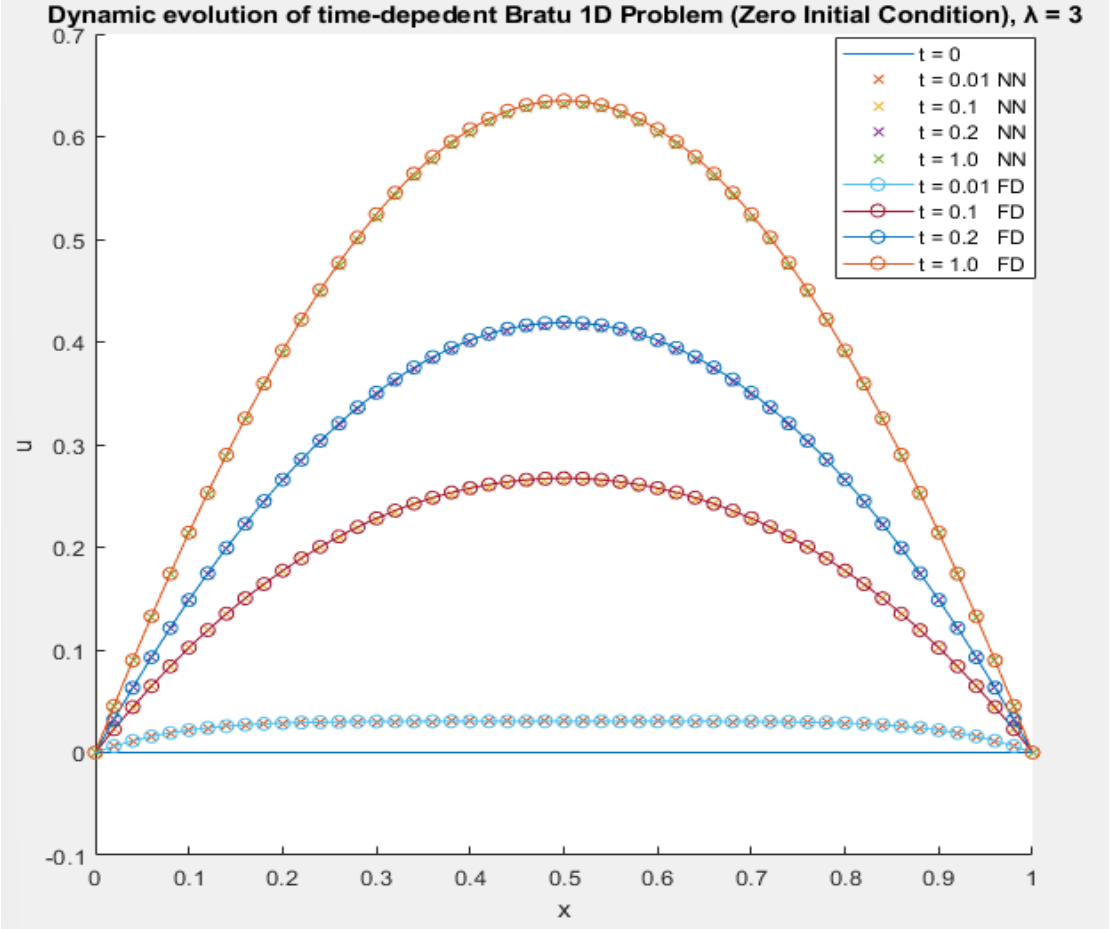
b.




*Σχήμα 3-3. Στιγμιότυπα λύσης του προβλήματος Bratu 1D για λ=2 σε χρόνους 0.01, 0.1, 0.2, 1 και γκαουσιανή κατανομή ως αρχική συνθήκη.*

*Πίνακας 3-3. Σύγκριση των μεθόδων (νευρωνικού δικτύου και πεπερασμένων διαφορών) για γκαουσιανή αρχική συνθήκη του προβλήματος Bratu 1D.*

| Time (t) | rMSE | MAPE |
|---|---|---|
| 0.0100 | 0.0444 | 0.4397 |
| 0.1000 | 0.0554 | 0.9735 |
| 0.2000 | 0.0328 | 0.8771 |
| 1 | 0.0039 | 0.2200 |




*Σχήμα 3-4. Στιγμιότυπα λύσης του προβλήματος Bratu 1D για λ=3.0 σε χρόνους 0.01, 0.1, 0.2, 1 και μηδενική αρχική συνθήκη.*

*Πίνακας 3-4. Σύγκριση των μεθόδων (νευρωνικού δικτύου και πεπερασμένων διαφορών) για μηδενική αρχική συνθήκη του προβλήματος Bratu 1D.*

| Time (t) | rMSE | MAPE |
|---|---|---|
| 0.0100 | 0.0009 | 0.4281 |
| 0.1000 | 0.0066 | 0.4623 |
| 0.2000 | 0.0086 | 0.4027 |
| 1 | 0.0160 | 0.5202 |

Παρατηρείται ότι και στις 3 περιπτώσεις το μέσο ποσοστιαίο σφάλμα είναι πολύ μικρό και τα σημεία έχουν πολύ καλή προσαρμογή σε διαφορετικές τιμές λ. Μάλιστα, το σφάλμα είναι μικρότερο του 1%. Αυτό σημαίνει ότι το νευρωνικό μπορεί να χρησιμοποιηθεί με ασφάλεια για την πρόβλεψη της δυναμικής ενός συστήματος που διέπει η εξίσωση Bratu. Επιπλέον, ενθαρρυντικό στοιχείο αποτελεί και το μέσο τετραγωνικό σφάλμα. Παρατηρείται ότι η τιμή του είναι περίπου 4 τάξεις μεγέθους μικρότερη από την μέγιστη τιμή της συνάρτησης σε κάθε περίπτωση, γεγονός που επιβεβαιώνει την επιτυχή εκπαίδευση του νευρωνικού. Το μεγαλύτερο σφάλμα παρατηρείται στο Σχήμα 3-3, το οποίο περιέχει τις πιο απομακρυσμένες τιμές από την μόνιμη κατάσταση που υπολογίζεται για λ=2. Αυτό, μπορεί να εξηγηθεί ως εξής: οι τιμές που λαμβάνει το χρονικό διαφορικό είναι μεγαλύτερες ή βρίσκονται στα όρια του πεδίου τιμών που χρησιμοποιήθηκαν για την εκπαίδευση του δυναμικού.

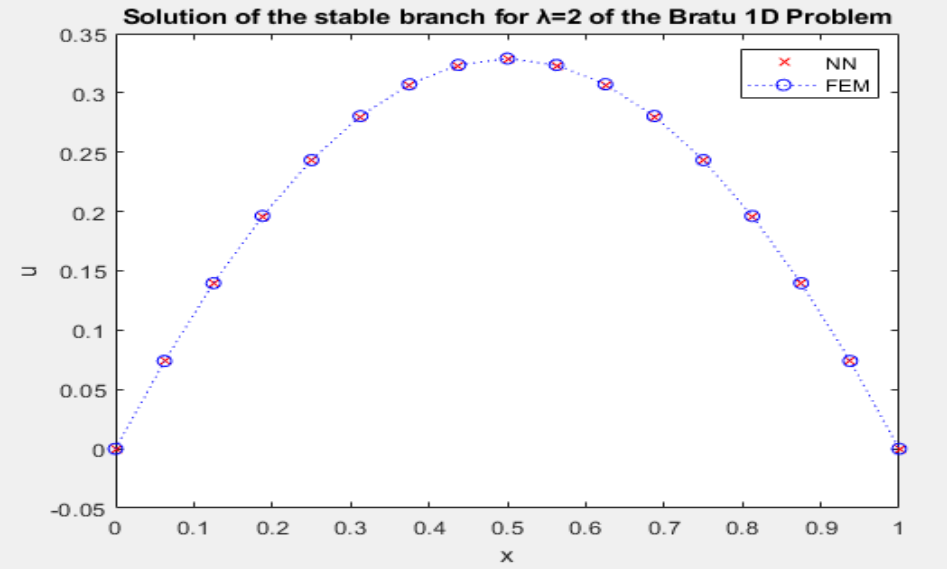
Οι τρεις περιπτώσεις δεν επιλέχτηκαν τυχαία. Στην πρώτη περίπτωση υπάρχουν μόνο αρνητικές χρονικές παράγωγοι, στην τρίτη περίπτωση μόνο θετικές, ενώ στη δεύτερη συνδυασμός αρνητικών και θετικών, ώστε να οδηγηθεί το σύστημα σε μόνιμη κατάσταση, που είναι πολύ κοντά στη λύση για τη στιγμή t=1. Αυτό έγινε για να διαπιστωθεί αν το σύστημα είναι αξιόπιστο σε κάθε είδος μεταβολής.

### 3.1.3. Εύρεση Μόνιμης Κατάστασης και Σύγκριση με Μέθοδο Πεπερασμένων Στοιχείων

Στην ενότητα αυτή παρουσιάζονται τα αποτελέσματα για τρεις ξεχωριστές περιπτώσεις μόνιμης κατάστασης (Διακριτοποίηση του χωρίου σε 17 ισαπέχοντα σημεία):

1. **Ευσταθής λύση μόνιμης κατάστασης (λ=2).**

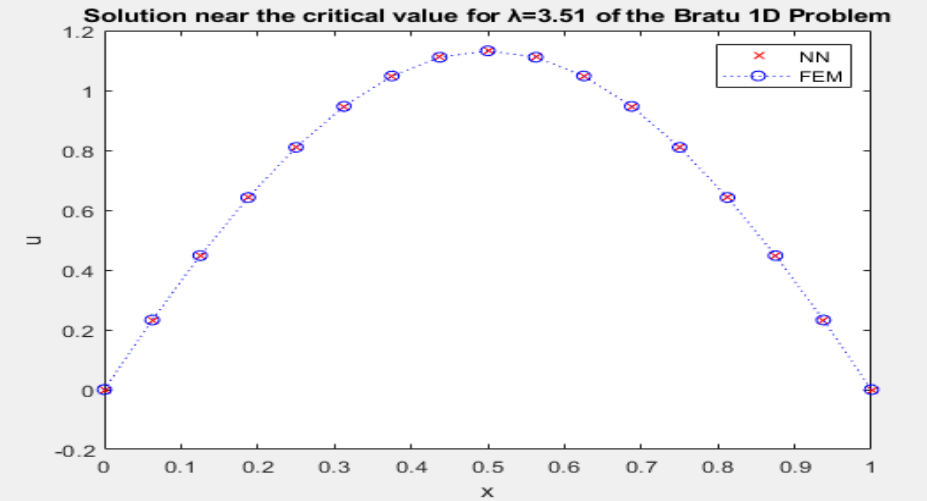
Οι αρχικές συνθήκες είναι κοντά στο μηδέν (η μηδενική στην περίπτωση των FEM κι η **u=0.01** για την Newton GMRES) έτσι ώστε να οδηγηθεί στον ευσταθή κλάδο. Η σύγκριση που προκύπτει από τις 2 μεθόδους παρουσιάζεται στο Σχήμα 3-5.




*Σχήμα 3-5. Ευσταθής λύση μόνιμης κατάστασης του προβλήματος Bratu 1D για λ=2 με το νευρωνικό δίκτυο (κόκκινα χ) και τη μέθοδο των πεπερασμένων στοιχείων (διακεκομμένη μπλε γραμμή με κύκλους) .*

2. **Λύση πολύ κοντά στο κρίσιμο σημείο (λ=3.51) που ανήκει στον ευσταθή κλάδο.**

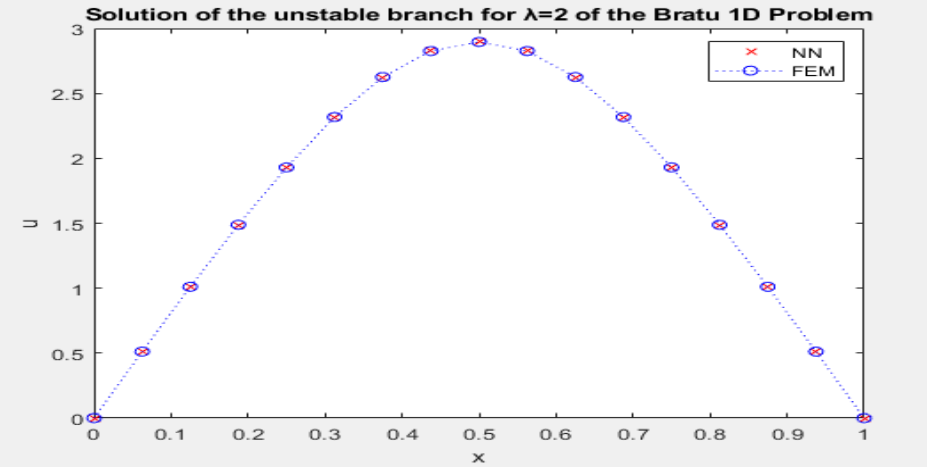
Οι αρχικές συνθήκες είναι κοντά στο μηδέν (η μηδενική στην περίπτωση των FEM κι η **u=0.01** για την Newton GMRES) έτσι ώστε να οδηγηθεί στον ευσταθή κλάδο. Η σύγκριση που προκύπτει από τις 2 μεθόδους παρουσιάζεται στο σχήμα 3-6.




*Σχήμα 3-6. Λύση μόνιμης κατάστασης του προβλήματος Bratu 1D κοντά στο κρίσιμο σημείο για λ=3.51 με το νευρωνικό δίκτυο (κόκκινα χ) και τη μέθοδο των πεπερασμένων στοιχείων (διακεκομμένη μπλε γραμμή με κύκλους) .*

### 3. Ασταθής λύση μόνιμης κατάστασης (λ=2).

Οι αρχικές συνθήκες είναι κατάλληλες (η u=4*sin(πx) στην περίπτωση των FEM κι η **u=3** για την Newton GMRES) έτσι ώστε να οδηγηθεί στον ασταθή κλάδο. Η σύγκριση που προκύπτει από τις 2 μεθόδους παρουσιάζεται στο σχήμα 3-7.




*Σχήμα 3-7. Ασταθής λύση μόνιμης κατάστασης του προβλήματος Bratu 1D για λ=2 με το νευρωνικό δίκτυο (κόκκινα χ) και τη μέθοδο των πεπερασμένων στοιχείων (διακεκομμένη μπλε γραμμή με κύκλους)*

Στον ακόλουθο πίνακα 3-5, παρουσιάζονται τα αποτελέσματα των μαθηματικών δεικτών που χρησιμοποιούνται για την αξιολόγηση του νευρωνικού.

*Πίνακας 3-5. Μαθηματική σύγκριση των 2 μεθόδων (NN-Newton-GMRES με FEM) για το πρόβλημα Bratu 1D στην εύρεση μόνιμης κατάστασης.*

| Τιμή Παραμέτρου/ Χώρος λύσης | rMSE | MAPE |
|---|---|---|
| λ=2 / ευσταθής κλάδος | 0.0015 | 0.1474 |
| λ=3.51 / ~ κρίσιμο σημείο | 6.6348e-04 | 0.0228 |
| λ=2 / ασταθής κλάδος | 0.0132 | 0.2513 |

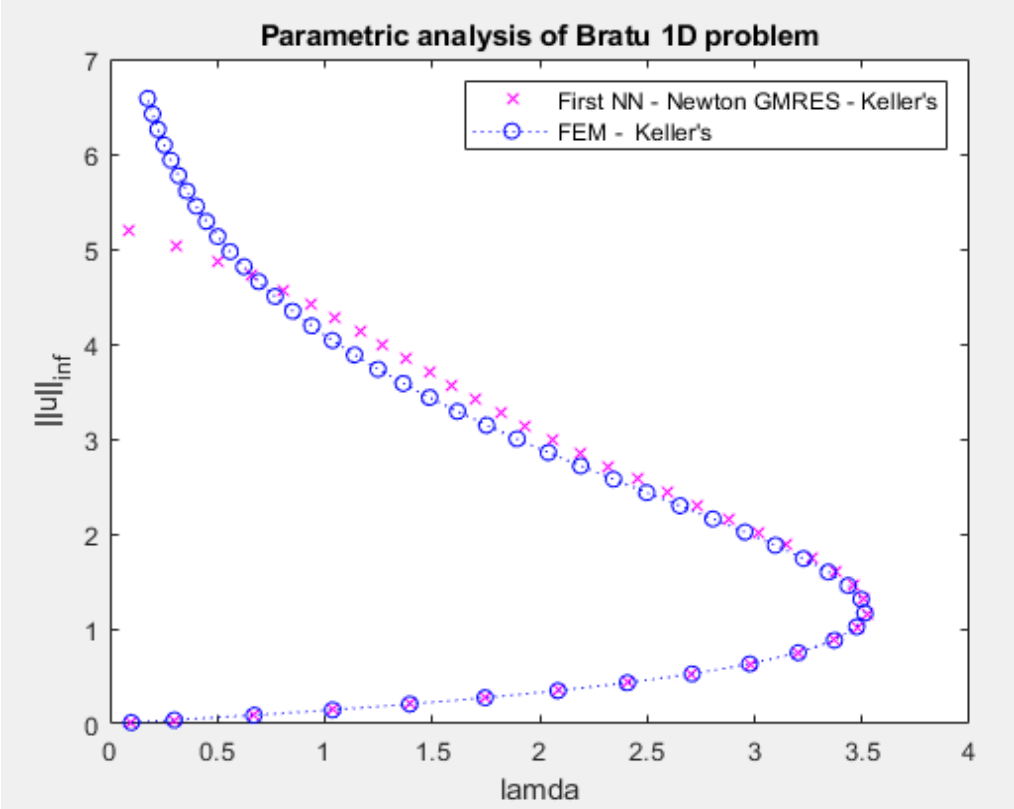
Είναι φανερό ότι υπάρχει πολύ καλή συμφωνία ανάμεσα στις 2 τεχνικές, καθώς η μέγιστη μέση διαφορά των δύο μεθόδων είναι περίπου 0.25%, ενώ και η τετραγωνική ρίζα του μέσου τετραγωνικού σφάλματος έχει πολύ μικρή τιμή και στις 3 περιπτώσεις. Επομένως, υπάρχει εμπιστοσύνη στην χρήση του νευρωνικού και σε αυτή την περίπτωση.

Αξιοσημείωτο είναι επίσης το γεγονός ότι η διακριτοποίηση δεν επηρεάζει το νευρωνικό σε αυτά τα παραδείγματα, καθώς η απόδοση του είναι παρόμοια τόσο στην περίπτωση του αδρού πλέγματος (17 σημεία στην εύρεση μόνιμης κατάστασης) και του πιο πυκνού πλέγματος (51 σημεία στην δυναμική μελέτη του φαινομένου).

### 3.1.4. Παραμετρική Ανάλυση με Νευρωνικό Δίκτυο και Σύγκριση με Υπολογιστικές Τεχνικές

Στην ενότητα αυτή παρουσιάζεται η παραμετρική ανάλυση που έγινε με τη χρήση του νευρωνικού δικτύου για τον υπολογισμό των υπολοίπων στην μέθοδο Newton GMRES και την μέθοδο Keller και συγκρίνεται με τη λύση που προκύπτει από τον συνδυασμό των πεπερασμένων στοιχείων με την ίδια μέθοδο παραμετρικής ανάλυσης. Η διακριτοποίηση περιλαμβάνει το χωρισμό του χωρίου σε 17 ισαπέχοντα σημεία (μείωση υπολογιστικού χρόνου, ενώ παράλληλα δεν χάνονται πληροφορίες για τη λήψη αποτελεσμάτων) ενώ το μήκος τόξου ds που χρησιμοποιείται είναι ίσο με 0.4 για την παραμετρική ανάλυση και 0.1 για την σύγκριση της κρίσιμης τιμής της παραμέτρου ($\lambda_{critical}$) μεταξύ των δύο μεθόδων. Τα αρχικά σημεία για την εκκίνηση του αλγορίθμου λαμβάνονται στις τιμές $\lambda_1$=0.1 και $\lambda_2$=0.3, ώστε να καλυφθεί μεγάλο εύρος των τιμών των παραμέτρων. Στο διάγραμμα εκφράζεται η άπειρη νόρμα της λύσης **u** (δηλαδή η μέγιστη τιμή της) ως προς την παράμετρο λ. Χρησιμοποιείται η άπειρη νόρμα ως η πιο ενδεικτική για την αλλαγή που συμβαίνει στις λύσεις, αλλάζοντας την λ.

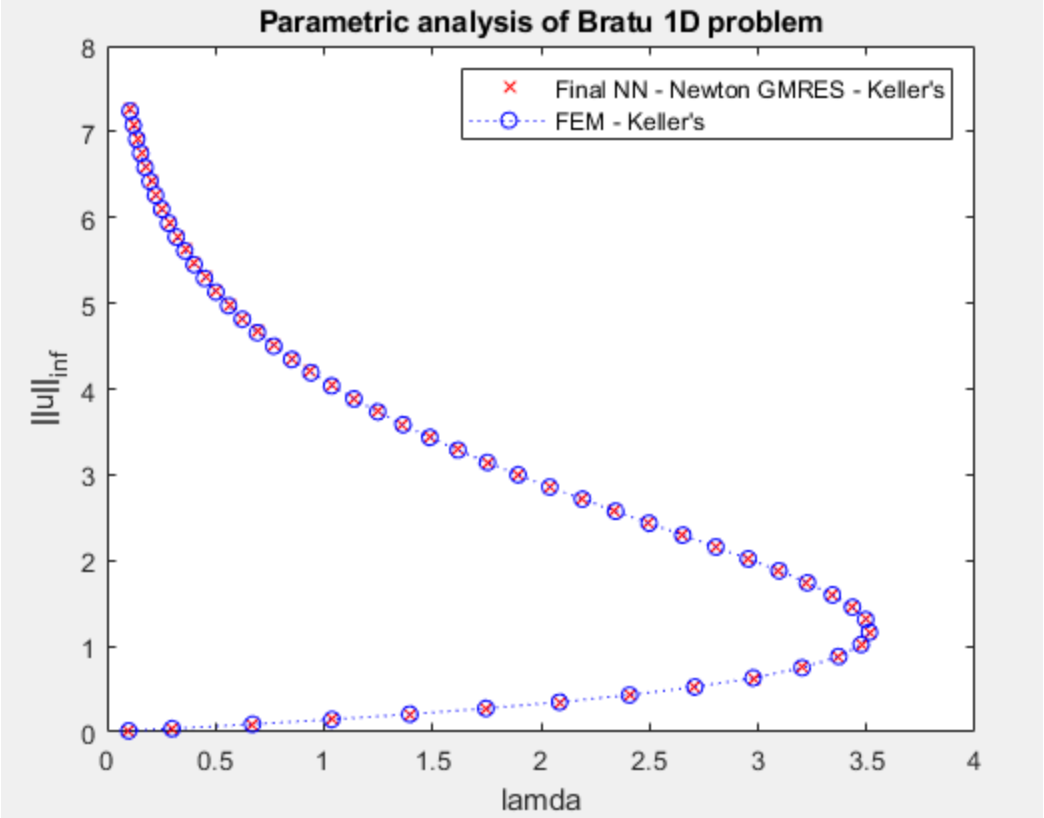
Αρχικά, παρουσιάζεται και η λύση του πρώτου εκπαιδευμένου νευρωνικού, πριν την επανεκπαίδευση αυτού με πειράματα που προκύπτουν από την παραμετρική ανάλυση με χρήση αυτού (Σχήμα 3-8).




*Σχήμα 3-8. Παραμετρική ανάλυση του μονοδιάστατου προβλήματος Bratu 1D ως προς λ με χρήση του πρώτου νευρωνικού δικτύου (πριν την ενημέρωση της βάσης δεδομένων) και της μεθόδου πεπερασμένων στοιχείων.*

Όπως διαπιστώνεται εύκολα, το νευρωνικό δίνει ικανοποιητικές εκτιμήσεις για τον ευσταθή κλάδο, κάτι που δικαιολογείται από το γεγονός της εκπαίδευσής του σε τιμές που βρίσκονται σε αυτό το εύρος. Ωστόσο, καθώς πλησιάζει το κρίσιμο σημείο, οι λύσεις έχουν σημαντικό σφάλμα, το οποίο μεγαλώνει όλο και περισσότερο καθώς οι λύσεις λαμβάνουν πλέον μεγαλύτερες τιμές.

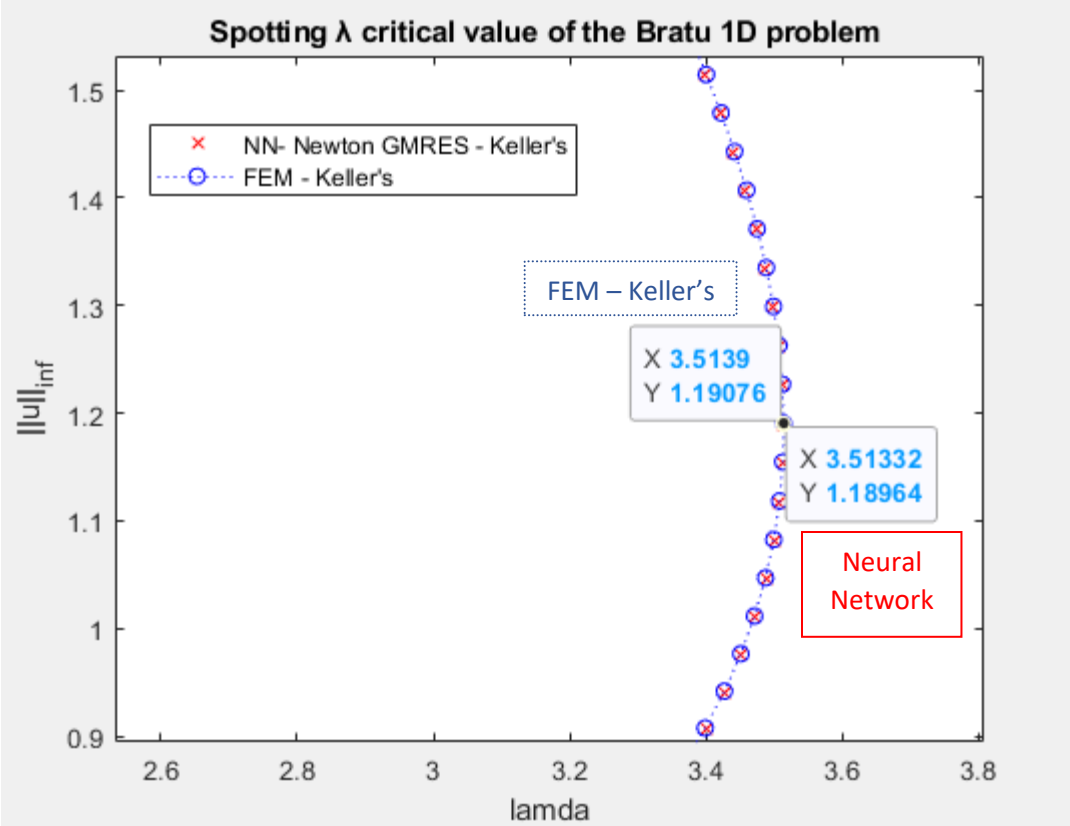
Έτσι, ακολουθώντας την διαδικασία της ενότητας 2.1.2, εκπαιδεύτηκε νευρωνικό σε ενημερωμένη βάση που δίνει την παραμετρική ανάλυση που παρουσιάζεται στο Σχήμα 3-9.




*Σχήμα 3-9. Παραμετρική ανάλυση του μονοδιάστατου προβλήματος Bratu ως προς λ με χρήση του τελικού νευρωνικού δικτύου (μετά την ενημέρωση της βάσης δεδομένων) και της μεθόδου πεπερασμένων στοιχείων.*

Παρατηρείται μια τέλεια συμφωνία ανάμεσα στις λύσεις που υπολογίζουν οι 2 μέθοδοι. Επειδή οι λύσεις που υπολογίζονται μπορεί να βρίσκονται σε ελαφρώς διαφορετικές τιμές λ (μια απόκλιση μικρότερη του 1%) , δεν χρησιμοποιούνται μαθηματικά μεγέθη για την επιβεβαίωση του αποτελέσματος. Αυτή τεκμηριώνεται από τα δεδομένα του πίνακα 3-5 για συγκεκριμένες τιμές του λ και θεωρείται ενδεικτικός της ακρίβειας της μεθόδου με το νευρωνικό δίκτυο.

Ωστόσο, η παραμετρική ανάλυση χρησιμοποιείται για να βρεθεί η τιμή $\lambda_{crit}$. Μειώνοντας το βήμα ds και εστιάζοντας στο σημείο όπου γίνεται η στροφή στο σύνολο των λύσεων, εντοπίζονται οι επιθυμητές τιμές. Αυτές φαίνονται στο σχήμα 3-10 και στον πίνακα 3-6 δίνεται και κάποια στοιχειώδης στατιστική σύγκριση μεταξύ των μεθόδων.




*Σχήμα 3-10 Εντοπίζοντας την κρίσιμη τιμή της παραμέτρου και την άπειρη νόρμα της συνάρτησης u για το πρόβλημα Bratu 1D με χρήση NN και FEM .*

Από τα δεδομένα Σχήματος 3-10 ισχύει :

*Πίνακας 3-6. Δεδομένα για τη κρίσιμη τιμή της παραμέτρου λ και στατιστική σύγκριση NN με FEM για το πρόβλημα Bratu 1D.*

| Method | $\lambda_{crit}$ | $\|u\|_\infty$ | Error (%) in $\lambda_{crit}$ | Error (%) in $\|u\|_\infty$ |
|---|---|---|---|---|
| FEM – Keller's | 3.51390 | 1.19076 | - | - |
| NN- Newton GMRES - Keller's | 3.51332 | 1.18964 | -0.0165 | -0.0941 |

Το σφάλμα στον υπολογισμό της κρίσιμης παραμέτρου και της άπειρης νόρμας στο κρίσιμο λ παρατηρείται ότι είναι εντελώς αμελητέο (<<1%).

Εν κατακλείδι, μπορεί με βεβαιότητα να υποστηριχθεί ότι το νευρωνικό δίκτυο μπορεί να χρησιμοποιηθεί με ασφάλεια για την μελέτη προβλήματος Bratu 1D.

## 3.2. Αποτελέσματα Δισδιάστατου προβλήματος Bratu

### 3.2.1. Γκαουσιανές Διαδικασίες

Τα αποτελέσματα των Γκαουσιανών διαδικασιών παρουσιάζονται στον ακόλουθο πίνακα 3-7.

*Πίνακας 3-7. Αποτελέσματα ARD ανάλυσης διεργασιών Gauss του προβλήματος Bratu 2D.*

| Μεταβλητή | Τιμή Παραμέτρου (ARD) |
|---|---|
| x | 4.2e+07 |
| y | 5.8e+05 |
| **u** | **2.0e+00** |
| $u_x$ | 4.8e+07 |
| $u_{yy}$ | 3.5e+07 |
| **$u_{xx}$** | **2.6e+04** |
| **$u_{yy}$** | **2.4e+04** |
| $u_{xy}$ | 1.9e+09 |
| $u_{yx}$ | 1.9e+09 |
| **lamda** | **2.1e+01** |

Παρατηρείται εύκολα ότι η μέθοδος δίνει σαν σημαντικές μεταβλητές την τιμή της συνάρτησης, την παράμετρο λ και τις δεύτερες χωρικές παραγώγους. Αν και υπάρχει και διαφορά στην τάξη μεγέθους των $u_{xx}$, $u_{yy}$ (αυτές έχουν την ίδια), της παραμέτρου λ από την τιμή της συνάρτησης u (3 τάξεις κάτω) δεν θεωρείται τόσο μεγάλη η διαφορά. Αντίθετα, οι υπόλοιπες μεταβλητές παρουσιάζουν πολύ μεγάλη τιμή που διαφέρει τουλάχιστον 5 τάξεις μεγέθους από τις επιλεγμένες σημαντικές και για αυτό τον λόγο θεωρούνται αμελητέες. Πρέπει, λοιπόν, να τονιστεί η αξία της παραπάνω μεθόδου: χωρίς να γνωρίζει την φυσική του προβλήματος, κατάφερε να εντοπίσει τις μεταβλητές που βρίσκονται στο δεξί μέρος της διαφορικής εξίσωσης, βασιζόμενο μόνο σε πειραματικά δεδομένα (δεδομένα προσομοιώσεων).

Άρα, με οδηγό τις Gaussian Processes, προχωράει κανείς στην εκπαίδευση νευρωνικού δικτύου έχοντας τη γνώση ότι μπορεί να χρησιμοποιήσει σαν μεταβλητές εισόδου μόνο τις 4 (u, $u_{xx}$, $u_{yy}$, λ) από τις 10 (x, y, u, $u_x$, $u_y$, $u_{xx}$, $u_{yy}$, $u_{xy}$, $u_{yx}$, λ) μεταβλητές, περιορίζοντας έτσι κατά πολύ (60%) το σύνολο των δεδομένων που χρειάζονται για την εκπαίδευση του νευρωνικού δικτύου.

Αξίζει να σημειωθεί ότι έγινε χρήση μόλις των 1500 πρώτων παρατηρήσεων, γεγονός που αντιστοιχεί στις πρώτες τιμές που λαμβάνει ο πειραματιστής αν ακολουθηθεί η λογική που παρουσιάστηκε κατά την πειραματική διαδικασία. Στις τιμές αυτές περιλαμβάνονται πειράματα από αρκετές αρχικές συνθήκες και διαφορετικές τιμές της παραμέτρου λ.

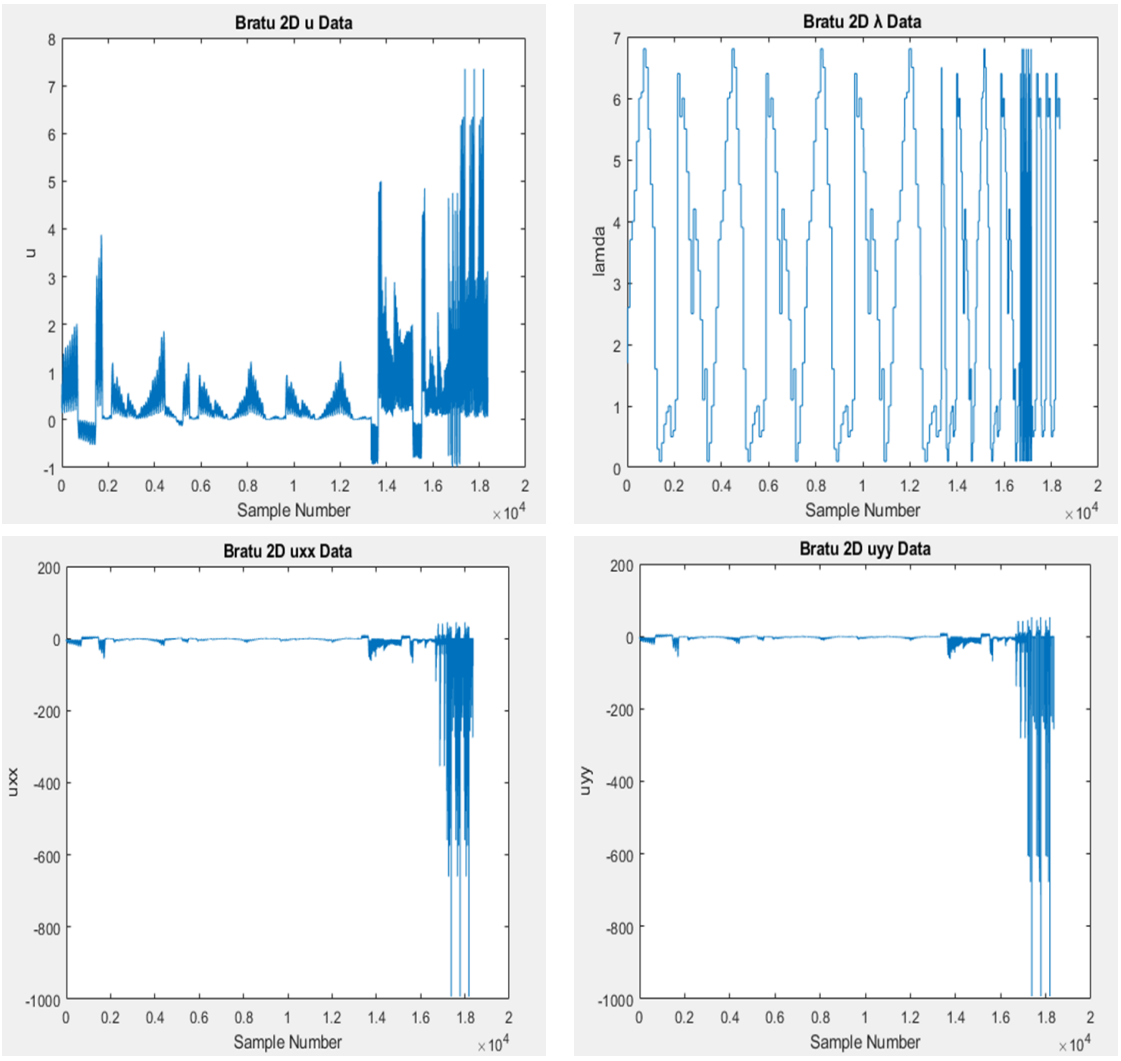
Επίσης, παρατηρήθηκε ότι το σύστημα είναι πολύ ευαίσθητο στην προσθήκη επιπλέον σημείων / παρατηρήσεων στην ομάδα τιμών με την οποία γίνεται η εκπαίδευση των διαδικασιών Gauss. Αυτό έχει ως αποτέλεσμα την αλλαγή των άνωθι τιμών, οι οποίες οδηγούνται όλες στην ίδια τάξη μεγέθους, γεγονός που αποτρέπει την εύρεση των σημαντικών μεταβλητών του προβλήματος και έτσι την κατανόηση της φυσικής που το διέπει και αποδίδεται στην προσθήκη δεδομένων όπου η χρονική παράγωγος παίρνει μικρότερες τιμές. Όσο μικρότερες, τόσο μεγαλύτερη η πιθανότητα αποτυχίας.

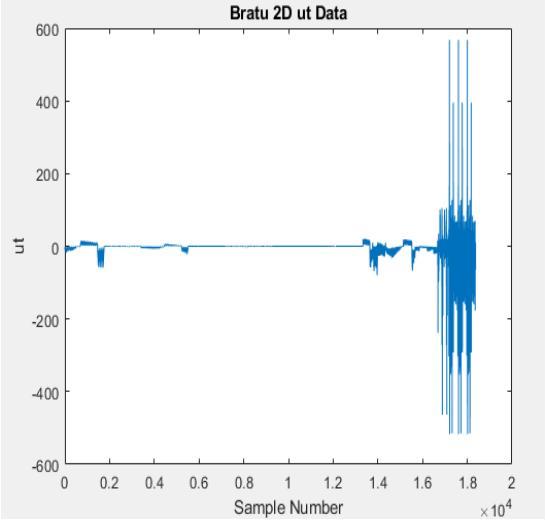
Επιπλέον, επιβεβαιώνεται το συμπέρασμα της παραγράφου 3.1.2 περί της συλλογής δεδομένων σε μικρούς χρόνους και κοντά στη περιοχή των έντονων δυναμικών φαινομένων, επιβεβαιώνοντας ότι είναι η πλέον κατάλληλη για την αποτελεσματική εύρεση των σημαντικών όρων του δεξιού μέλους της εξίσωσης.

Τέλος, η διαφορετική τιμή των x και y οφείλεται στο γεγονός της διαφορετικής αρχειοθέτησης των τιμών. Ανατίθενται όλες οι τιμές του x σε κάποιο y, οπότε μέχρι το 1500$^{ο}$ σημείο δεν έχουν λάβει τις ίδιες τιμές οι χωρικές μεταβλητές.

### 3.2.2. Εκπαίδευση Νευρωνικού Δικτύου και Δυναμική Εξέλιξη του Φαινομένου

Αξίζει να παρουσιαστεί σε αυτό το σημείο το σύνολο των δεδομένων που χρησιμοποιήθηκαν για την εκπαίδευση του νευρωνικού. Αυτά παρουσιάζονται στο Σχήμα 3-11.

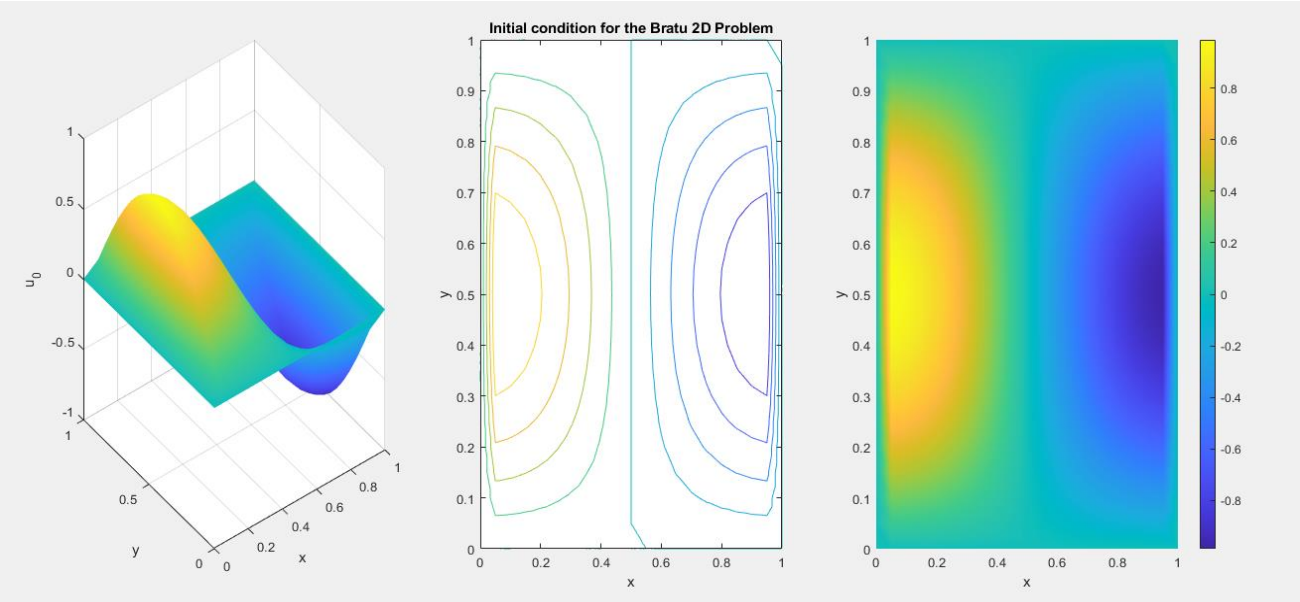





*Σχήμα 3-11 (α)-(ε). Πάνω αριστερά (α): δεδομένα για τη μεταβλητή εισόδου u, πάνω δεξιά (β): δεδομένα για τη μεταβλητή εισόδου λ, μέση αριστερά (γ): δεδομένα για τη μεταβλητή εισόδου $u_{xx}$, μέση δεξιά (δ): δεδομένα για τη μεταβλητή εισόδου $u_{yy}$ και κάτω κέντρο (ε): δεδομένα για τη μεταβλητή εξόδου $u_t$ που χρησιμοποιήθηκαν για την εκπαίδευση του νευρωνικού του προβλήματος Bratu 2D.*

Όπως γίνεται αισθητό, το σύστημα έχει εκπαιδευτεί σε σχετικά χαμηλές τιμές των μεταβλητών, ενώ η μεταβλητή λ παίρνει αρκετές τιμές μέσα στο σύνολο τιμών της. Οι λίγες μετρήσεις που παίρνουν σχετικά μεγάλες τιμές είναι αυτές που επιτρέπουν στο νευρωνικό να βρίσκει τον ασταθή κλάδο χωρίς να χάνει σημαντικά την ακρίβεια στην εύρεση του ευσταθούς.

Ακολουθώντας τα βήματα που περιγράφονται στο κεφάλαιο 2.3.3., λαμβάνονται τα ακόλουθα διαγράμματα για την αρχική συνθήκη με την τιμή της παραμέτρου λάμδα να ορίζεται στο 3.6 και παρουσιάζονται στο σχήμα 3-12.




*Σχήμα 3-12. Παρουσίαση της αρχικής συνθήκης $u_0$=cos(πx)sin(πy) στο Ω και u=0 στο ∂Ω. Απεικόνιση σε 3D, με τη μορφή ισοϋψών και προβολής στον x-y άξονα. (Από δεξιά προς τα αριστερά)*

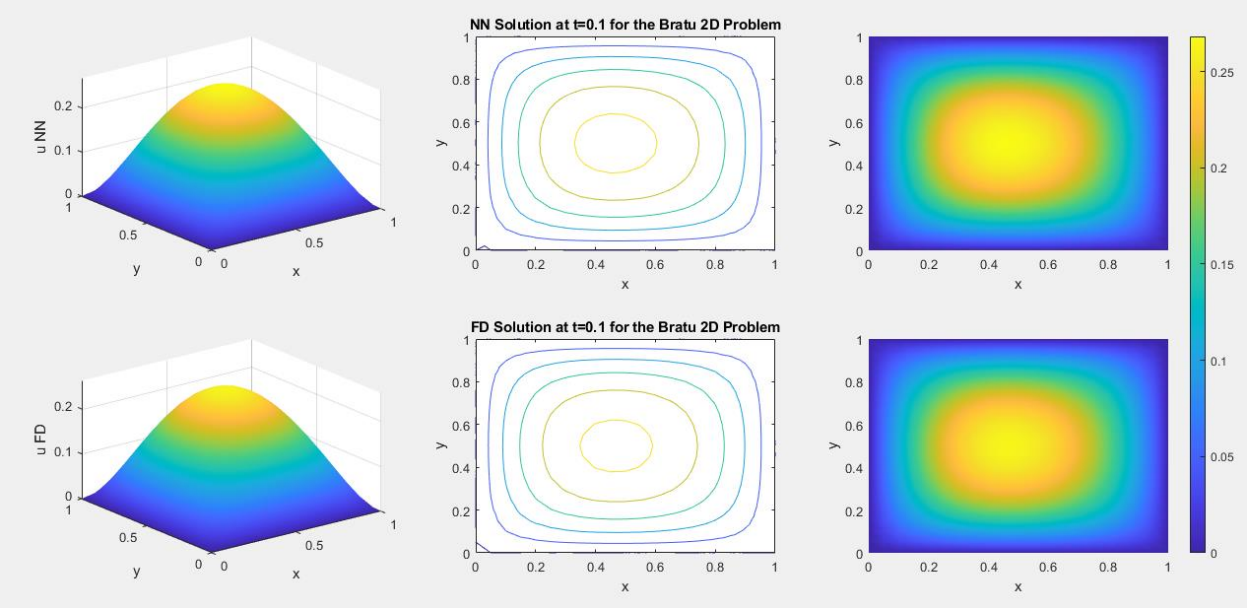
Επιπλέον, για την αξιολόγηση, εκτός από τον οπτικό έλεγχο, όπως και στο μονοδιάστατο πρόβλημα, χρησιμοποιούνται και πιο φορμαλιστικά μαθητικά μεγέθη, τα οποία είναι η τετραγωνική ρίζα του μέσου τετραγωνικού σφάλματος (root mean squared error) (3.1) και μέσο ποσοστιαίο απόλυτο σφάλμα( mean absolute percentage error) (3.2).

Τα σημεία των συνόρων (τα οποία είναι στο πλήθος $2n_x+ 2n_y – 4$) κρατούνται εκτός, αφού αποτελούν συνοριακή συνθήκη και δεν συμμετέχουν στον υπολογισμό του σφάλματος.

Έτσι, παρουσιάζονται οι εξής περιπτώσεις (σχήματα 3-13 έως 3-16) για τις οποίες υπολογίζονται και τα αναφερθέντα ποσοτικά εργαλεία για να συζητηθεί κατά πόσο επιτυχής είναι η εκπαίδευση του νευρωνικού.

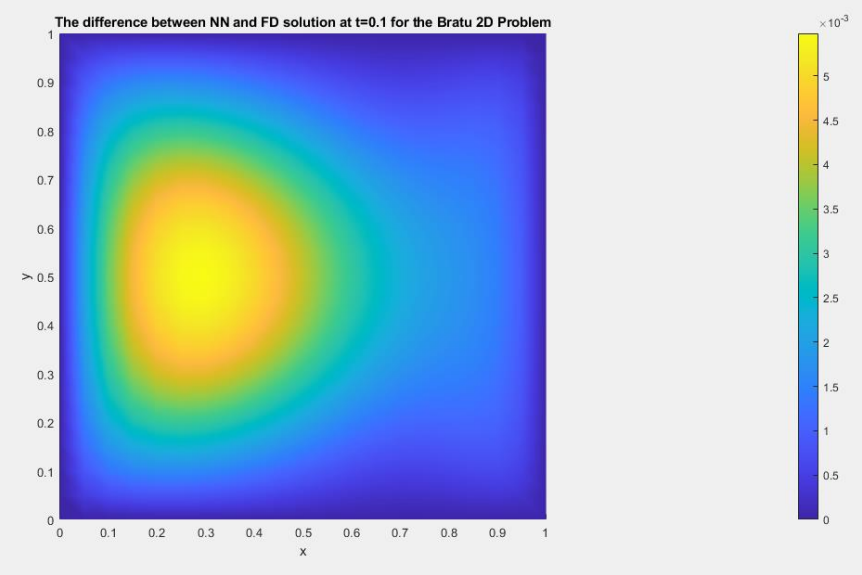
Όπως είναι φανερό επιλέγεται η συγκεκριμένη αρχική συνθήκη γιατί περιλαμβάνει τόσο θετικές όσο και αρνητικές τιμές, ενώ η τιμή λ=3.6 οδηγεί σε λύση με μέγιστο χαμηλότερο της μονάδας. Οπότε εξετάζεται το νευρωνικό αν μπορεί να προβλέψει τις «αντίθετες» χρονικές παραγώγους που χρειάζονται για να φθάσει το σύστημα στην μόνιμη κατάσταση ($u_t$=0).

- **Χρονική στιγμή t=0.1**




*Σχήμα 3-13. Παρουσίαση της λύσης για τη ΜΔΕ με $u_0$=cos(πx)sin(πy) στο Ω και u=0 στο ∂Ω στο χρόνο t=0.1. Απεικόνιση σε 3D, με τη μορφή ισοϋψών και προβολής στον x-y άξονα. (Από δεξιά προς τα αριστερά). Στην πάνω σειρά δίνεται η λύση του νευρωνικού (NN), ενώ στην κάτω η λύση με τη μέθοδο των πεπερασμένων διαφορών (FD). (λ=3.6 και np=441).*

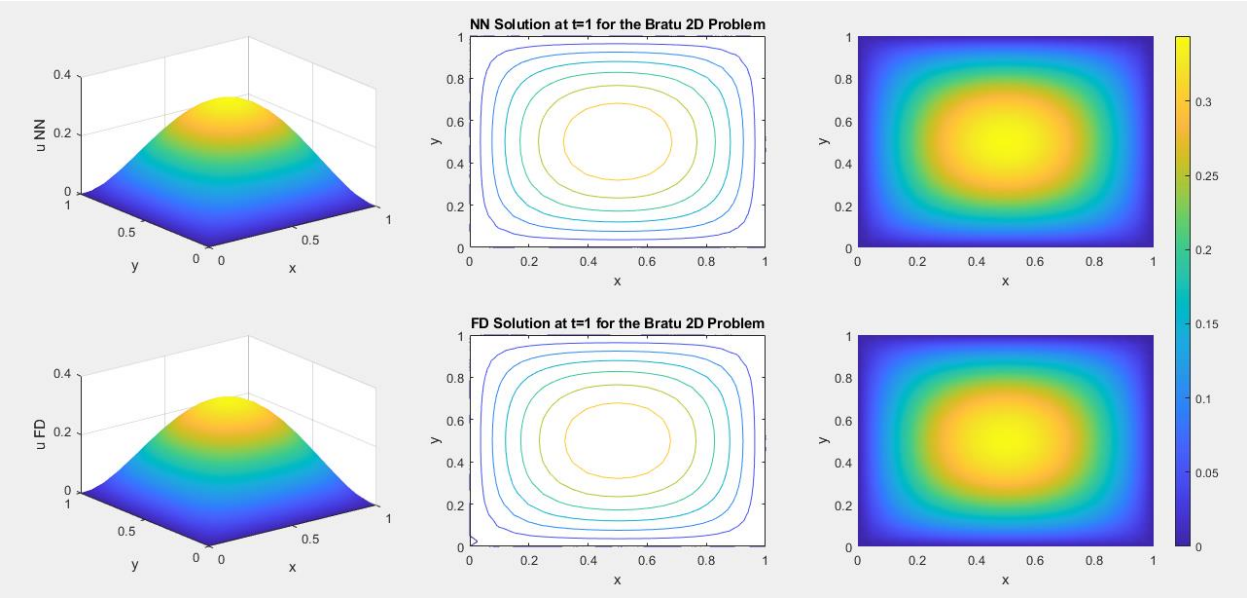
Κατά τη σύγκριση των διαγραμμάτων, προκύπτει ότι υπάρχει συμφωνία ανάμεσα στις 2 μεθόδους. Η 3D γεωμετρία και η προβολή στον δισδιάστατο χώρο έχουν τον ίδιο χρωματισμό σε όλα τα σημεία, ενώ οι ισοϋψείς έχουν το ίδιο σχήμα και σχηματίζονται στις ίδιες θέσεις. Η συμφωνία ανάμεσα στα αποτελέσματα επιβεβαιώνεται και από το Σχήμα 3-14 και τα μαθηματικά μεγέθη του Πίνακα 3-8.




*Σχήμα 3-14. Διαφορά ανάμεσα στη λύση με τη χρήση του νευρωνικού και εκείνη των μεθόδων των πεπερασμένων διαφορών του Δυναμικού Προβλήματος Bratu 2D για λ=3.6 στο χρόνο t= 0.1 και np=441.*

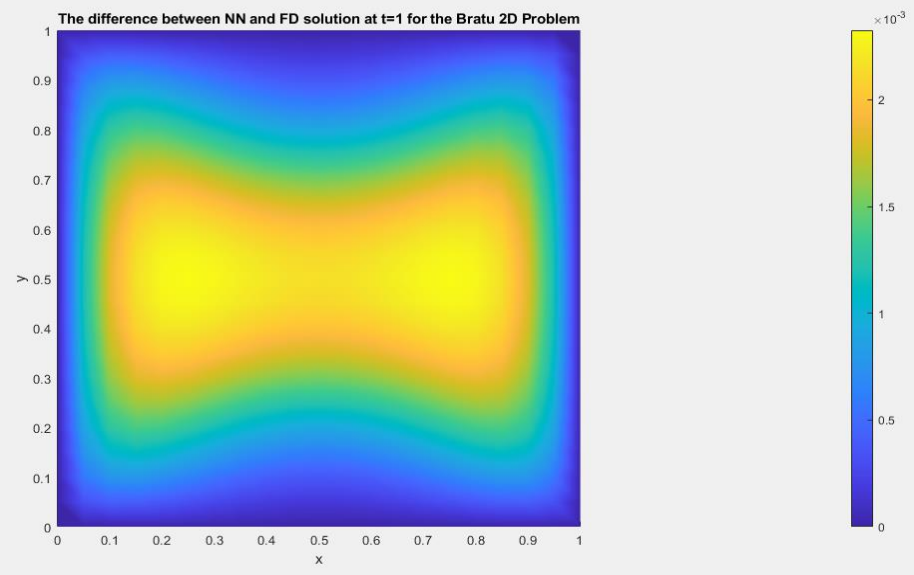
Όπως γίνεται φανερό, οι μεγαλύτερες διαφορές στις τιμές παρουσιάζονται στα «δύσκολα σημεία» της αρχικής συνθήκης, δηλαδή εκείνα που είναι πιο απομακρυσμένα από την αρχική συνθήκη. Πρόκειται για τα σημεία του θετικού λόφου, ενώ παρατηρείται μια γενικότερη υπερεκτίμηση της λύσης. Ωστόσο, αξίζει να τονιστεί ότι οι διαφορές είναι μία με δύο τάξεις μεγέθους κάτω από εκείνες των λύσεων, που αντιστοιχεί σε τοπικό σφάλμα μικρότερο του 5%.

- **Χρονική στιγμή t=1**




*Σχήμα 3-15. Παρουσίαση της λύσης για τη ΜΔΕ με $u_0$=cos(πx)sin(πy) στο Ω και u=0 στο ∂Ω στο χρόνο t=1. Απεικόνιση σε 3D, με τη μορφή ισοϋψών και προβολής στον x-y άξονα. (Από δεξιά προς τα αριστερά). Στην πάνω σειρά δίνεται η λύση του νευρωνικού (NN), ενώ στην κάτω η λύση με τη μέθοδο των πεπερασμένων διαφορών (FD). (λ=3.6 και np=441).*

Είναι φανερό ότι το σύστημα έχει προσεγγίσει αρκετά την μόνιμη κατάσταση. Ειδικά από τις ισοϋψείς αποκαλύπτεται η μεταφορά του μεγίστου προς το κέντρο του πεδίου ορισμού της εξίσωσης. Οι 2 λύσεις βρίσκονται πολύ κοντά και σε αυτή την περίπτωση, με τα αποτελέσματα να τα επιβεβαιώνουν τόσο το Σχήμα 3-16 όσο και ο Πίνακας 3-8.




*Σχήμα 3-16. Διαφορά ανάμεσα στη λύση με τη χρήση του νευρωνικού και εκείνη των μεθόδων των πεπερασμένων διαφορών του Δυναμικού Προβλήματος Bratu 2D με λ=3.6 στο χρόνο t= 1 και np=441 .*

Από το σχήμα 3-16 γίνεται αντιληπτό ότι το νευρωνικό υπερεκτιμά την λύση του προβλήματος σε σύγκριση με τη μέθοδο των πεπερασμένων διαφορών. Ωστόσο, οι μεγαλύτερες τιμές εντοπίζονται στο κέντρο του σχήματος, όπου η συνάρτηση λαμβάνει επίσης μεγαλύτερες τιμές και έτσι προκύπτει πάλι μικρό σχετικά σφάλμα.

*Πίνακας 3-8. Σύγκριση των μεθόδων (νευρωνικού δικτύου και πεπερασμένων διαφορών) για κυματική αρχική συνθήκη του προβλήματος Bratu 2D.*

| Time (t) | rMSE | MAPE |
|---|---|---|
| 0.1 | 0.0423 | 1.6247 |
| 1 | 0.0256 | 0.8815 |

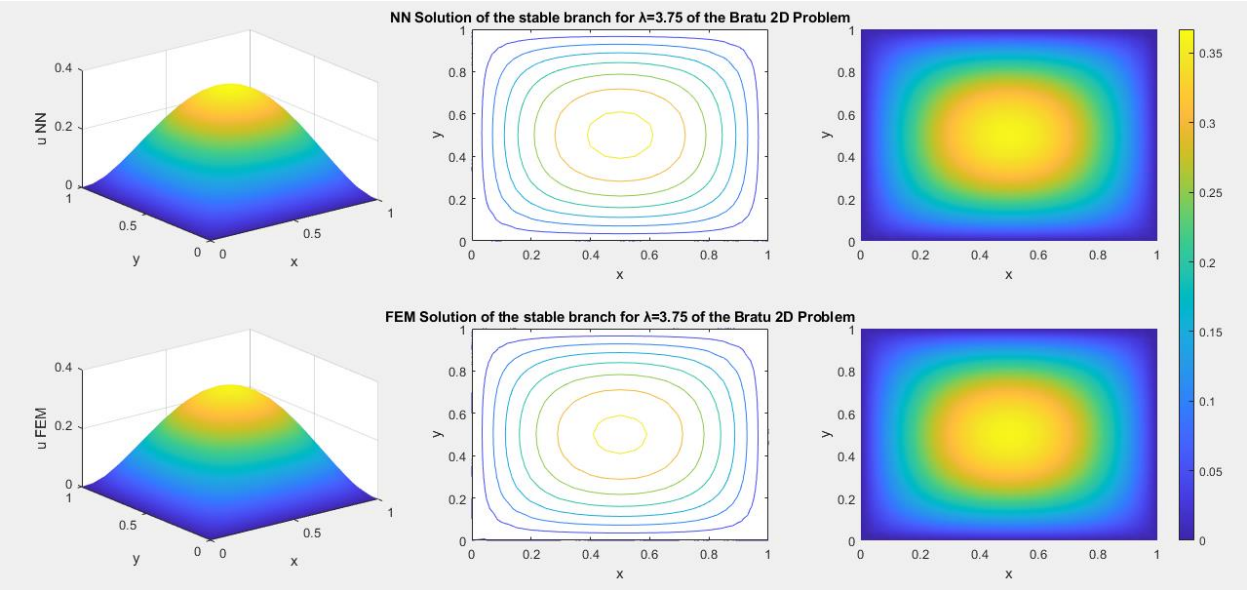
Παρατηρείται ότι και στους 2 χρόνους, το μέσο ποσοστιαίο σφάλμα είναι πολύ μικρό και τα σημεία έχουν πολύ καλή προσαρμογή. Μάλιστα, το σφάλμα είναι της τάξεως του 2%. Αυτό σημαίνει ότι το νευρωνικό μπορεί να χρησιμοποιηθεί με ασφάλεια για την πρόβλεψη της δυναμικής ενός συστήματος που διέπει η εξίσωση Bratu σε αυτό το εύρος χρόνου και σε αντίστοιχες αρχικές συνθήκες. Επιπλέον, ενθαρρυντικό στοιχείο αποτελεί και το μέσο τετραγωνικό σφάλμα. Παρατηρείται ότι η τιμή του είναι αρκετά μικρότερη από την μέγιστη τιμή της συνάρτησης σε κάθε περίπτωση, γεγονός που επιβεβαιώνει την επιτυχή εκπαίδευση του νευρωνικού.

### 3.2.3. Εύρεση Μόνιμης Κατάστασης και Σύγκριση με Μέθοδο Πεπερασμένων Στοιχείων

Στην ενότητα αυτή παρουσιάζονται τα αποτελέσματα για δύο ξεχωριστές περιπτώσεις μόνιμης κατάστασης (Διακριτοποίηση του χωρίου σε 21x21 ισαπέχοντα σημεία):

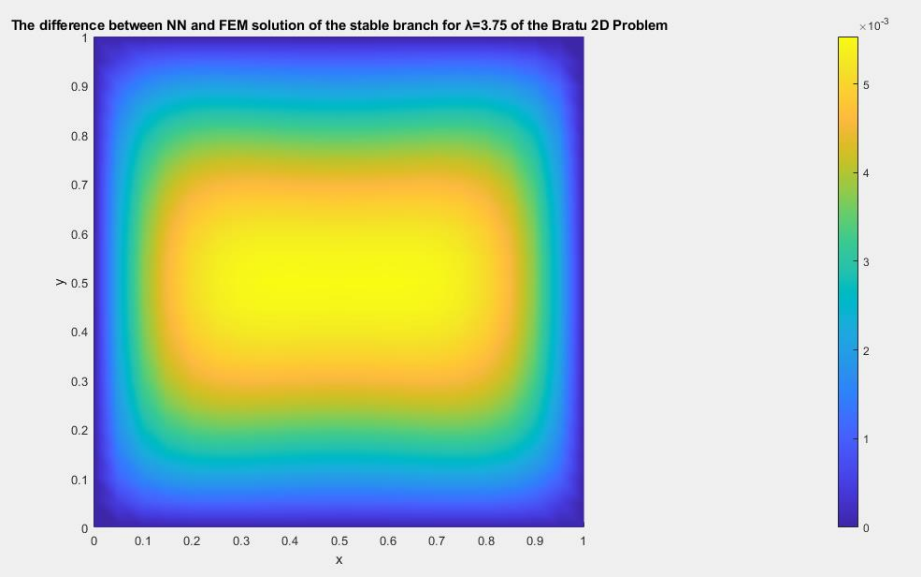
1. **Ευσταθής λύση μόνιμης κατάστασης (λ=3.75).**

Η αρχική εκτίμηση της λύσης για την επαναληπτική διαδικασία είναι μεταξύ του μηδενός και της μονάδας (η μηδενική στην περίπτωση των FEM κι η **u=sin(πx)sin(πy)** για την Newton GMRES) έτσι ώστε να οδηγηθεί στον ευσταθή κλάδο. Η σύγκριση που προκύπτει από τις 2 μεθόδους παρουσιάζεται στα Σχήματα 3-17 και 3-18.




*Σχήμα 3-17. Λύση μόνιμης κατάστασης στον ευσταθή κλάδο για λ=3.75. Απεικόνιση σε 3D, με τη μορφή ισοϋψών και προβολής στον x-y άξονα. (Από δεξιά προς τα αριστερά). Στην πάνω σειρά δίνεται η λύση του νευρωνικού (NN), ενώ στην κάτω η λύση με τη μέθοδο των πεπερασμένων στοιχείων (FEM). (np=441).*

Όπως προκύπτει από τα διαγράμματα, οι λύσεις είναι πολύ κοντά μεταξύ τους. Δίνουν τον ίδιο αριθμό ισοϋψών στις ίδιες θέσεις, ενώ τα μοτίβα των χρωματισμών, δηλαδή οι τιμές που υπολογίζονται ως λύση είναι πανομοιότυπα. Αυτή η αξιολόγηση επιβεβαιώνεται από το σχήμα 3-18, όπως επίσης και από τον πίνακα 3-9.

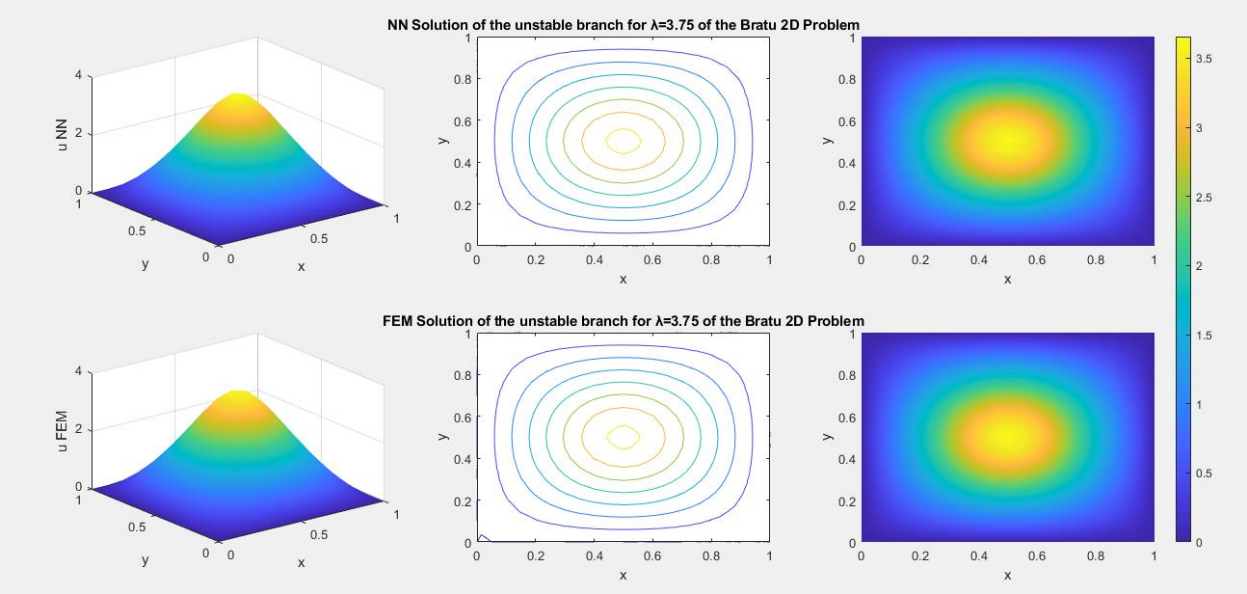



*Σχήμα 3-18. Διαφορά ανάμεσα στη λύση με τη χρήση του νευρωνικού και εκείνη των μεθόδων των πεπερασμένων στοιχείων στην εύρεση μόνιμης κατάστασης στον ευσταθή κλάδο του Προβλήματος Bratu 2D με λ=3.75 και np=441 .*

Παρατηρείται η συμπεριφορά της λύσης του μεταβατικού προβλήματος για t=1. Ισχύουν ακριβώς οι ίδιες παρατηρήσεις. Ικανοποιητική ακρίβεια καθώς οι διαφορές έχουν μικρή τιμή σε σχέση με αυτές που λαμβάνει η συνάρτηση.

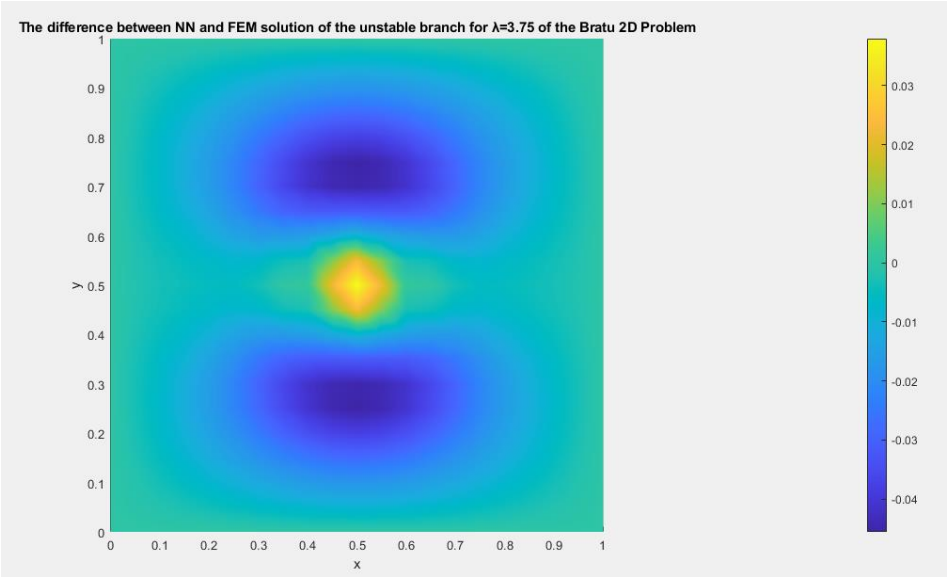
### 2. Ασταθής λύση μόνιμης κατάστασης (λ=3.75).

Η αρχική εκτίμηση επιλέγεται κατάλληλα (η u=3.5*sin(πx)*sin(πy) και στην περίπτωση των FEM και για την Newton GMRES) έτσι ώστε η επαναληπτική διαδικασία εύρεσης μόνιμης κατάστασης να οδηγηθεί στον ασταθή κλάδο. Η σύγκριση που προκύπτει από τις 2 μεθόδους παρουσιάζεται στα σχήματα 3-19 και 3-20. Βέβαια, αυτός είναι ένας τρόπος που δεν δουλεύει γενικά. Δηλαδή για να βρει κάποιος ασταθείς λύσεις με αυτόν τον τρόπο, θα πρέπει να ξέρει ότι υπάρχουν και ποιες είναι περίπου. Ο συστηματικός τρόπος είναι μέσω της παραμετρικής ανάλυσης που επιτρέπει τον υπολογισμό όλων των λύσεων.




*Σχήμα 3-19. Παρουσίαση της λύσης για τη μόνιμη κατάσταση στον ασταθή κλάδο με λ=3.75. Απεικόνιση σε 3D, με τη μορφή ισοϋψών και προβολής στον x-y άξονα. (Από δεξιά προς τα αριστερά). Στην πάνω σειρά δίνεται η λύση του νευρωνικού (NN), ενώ στην κάτω η λύση με τη μέθοδο των πεπερασμένων στοιχείων (FEM). (np=441).*

Και σε αυτή την περίπτωση, παρατηρείται συμφωνία μεταξύ των δύο τεχνικών (εκπαιδευμένου νευρωνικού και FEM). Αυτή επιβεβαιώνεται από το Σχήμα 3-20 όπου παρατηρείται ότι για το μεγαλύτερο τμήμα του χωρίου, η διαφορά ανάμεσα στις δύο λύσεις είναι κοντά στο μηδέν. Επιπλέον, οι μεγαλύτερες διαφορές που εμφανίζονται στο κεντρικό τμήμα, έχουν τιμές που είναι αμελητέες συγκριτικά με εκείνες που παίρνει η συνάρτηση / λύση σε εκείνα τα σημεία. Η θέση αυτή αποδεικνύεται από τα δεδομένα του Πίνακα 3-9.




*Σχήμα 3-20. Διαφορά ανάμεσα στη λύση με τη χρήση του νευρωνικού και εκείνη των μεθόδων των πεπερασμένων στοιχείων στην εύρεση μόνιμης κατάστασης στον ασταθή κλάδο του Προβλήματος Bratu 2D με λ=3.75 και np=441 .*

Στον Πίνακα 3-9, παρουσιάζονται τα αποτελέσματα των μαθηματικών δεικτών που χρησιμοποιούνται για την αξιολόγηση του νευρωνικού.

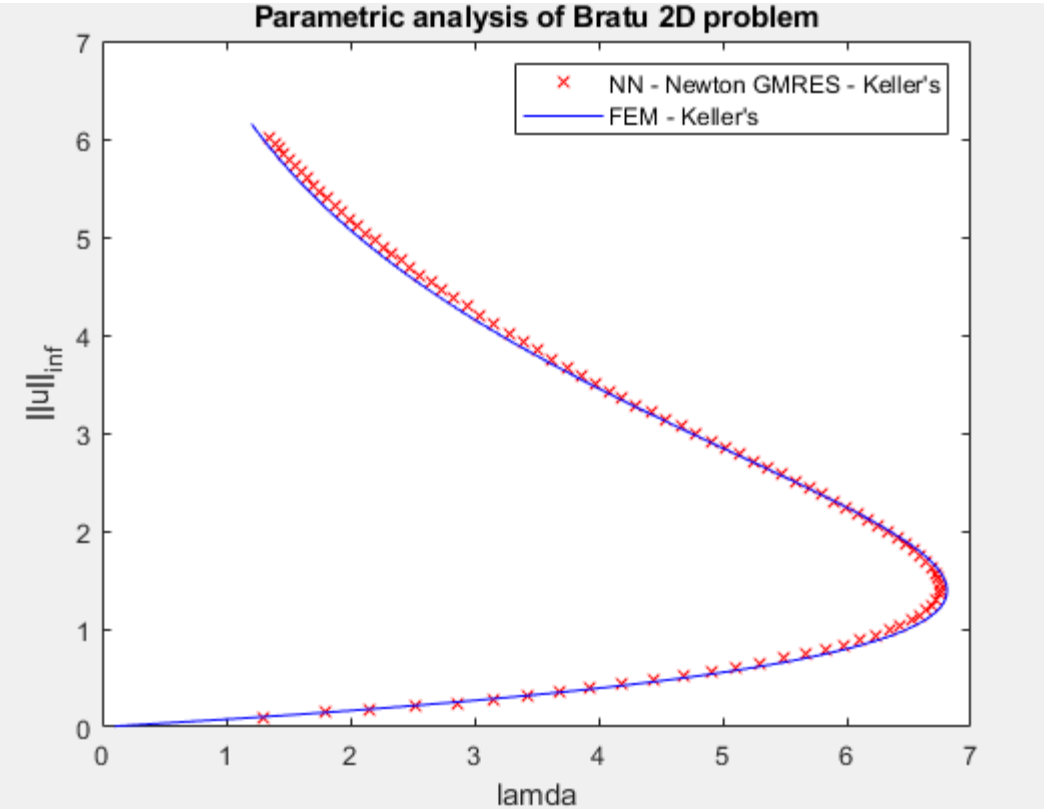
*Πίνακας 3-9. Μαθηματική σύγκριση των 2 μεθόδων (NN-Newton-GMRES με FEM) για το πρόβλημα Bratu 2D στην εύρεση μόνιμης κατάστασης.*

| Τιμή Παραμέτρου/ Χώρος λύσης | rMSE | MAPE |
|---|---|---|
| λ=3.75 / ευσταθής κλάδος | 0.0652 | 2.0212 |
| λ=3.75 / ασταθής κλάδος | 0.2822 | 1.3107 |

Είναι φανερό ότι υπάρχει πολύ καλή συμφωνία ανάμεσα στις 2 τεχνικές, καθώς η μέγιστη μέση διαφορά των δύο μεθόδων είναι περίπου 2%, ενώ και η τετραγωνική ρίζα του μέσου τετραγωνικού σφάλματος έχει πολύ μικρή τιμή και στις 2 περιπτώσεις. Επομένως, υπάρχει εμπιστοσύνη στην χρήση του νευρωνικού και σε αυτή την περίπτωση.

### 3.2.4. Παραμετρική Ανάλυση με Νευρωνικό Δίκτυο και Σύγκριση με λύσεις από πεπερασμένα στοιχεία.

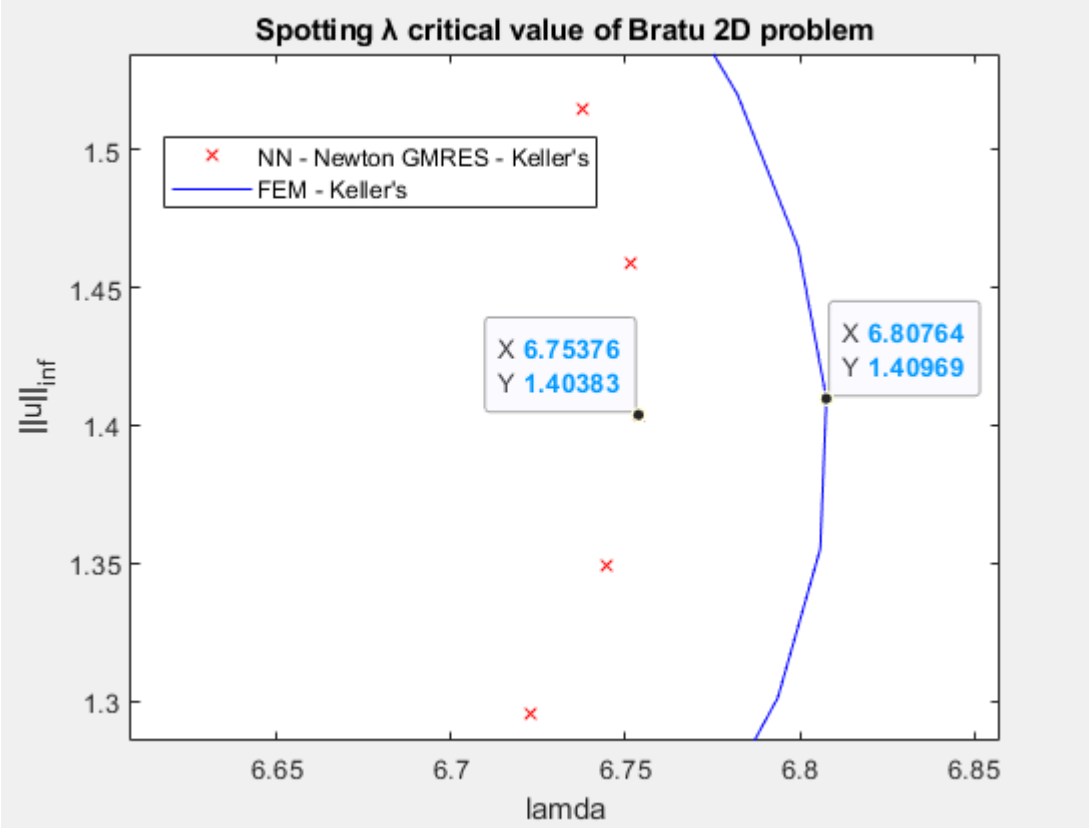
Στην ενότητα αυτή παρουσιάζεται η παραμετρική ανάλυση που έγινε με τη χρήση του νευρωνικού δικτύου για τον υπολογισμό των υπολοίπων (residuals) στην μέθοδο Newton GMRES και την μέθοδο Keller και συγκρίνεται με τη λύση που προκύπτει από τον συνδυασμό των πεπερασμένων στοιχείων με την ίδια μέθοδο παραμετρικής ανάλυσης. Η διακριτοποίηση περιλαμβάνει το χωρισμό του χωρίου σε 21x21 ισαπέχοντα σημεία (μείωση υπολογιστικού χρόνου, ενώ παράλληλα δεν χάνονται πληροφορίες για τη λήψη αποτελεσμάτων) ενώ το μήκος τόξου ds που χρησιμοποιείται είναι ίσο με 0.5 για την παραμετρική ανάλυση και για την σύγκριση της κρίσιμης τιμής της παραμέτρου ($\lambda_{critical}$) μεταξύ των δύο μεθόδων. Τα αρχικά σημεία για την εκκίνηση του αλγορίθμου λαμβάνονται στις τιμές $\lambda_1$=1.3 και $\lambda_2$=1.8, ώστε να καλυφθεί μεγάλο εύρος των τιμών των παραμέτρων, και αποτελούν το κάτω όριο για το οποίο προκύπτουν έγκυρα δεδομένα για το νευρωνικό. Στο διάγραμμα εκφράζεται η άπειρη νόρμα της λύσης **u** (δηλαδή η μέγιστη τιμή της) ως προς την παράμετρο λ. Χρησιμοποιείται η άπειρη νόρμα ως η πιο ενδεικτική για την αλλαγή που συμβαίνει στις λύσεις, αλλάζοντας την λ. Η λύση που προκύπτει παρουσιάζεται στο σχήμα 3-21.




*Σχήμα 3-21. Παραμετρική ανάλυση του δισδιάστατου προβλήματος Bratu ως προς λ με χρήση του νευρωνικού δικτύου και της μεθόδου πεπερασμένων στοιχείων.*

Παρατηρείται μια καλή συμφωνία ανάμεσα στις λύσεις που υπολογίζουν οι 2 μέθοδοι. Επειδή οι λύσεις που υπολογίζονται μπορεί να βρίσκονται σε ελαφρώς διαφορετικές τιμές λ, δεν χρησιμοποιούνται μαθηματικά μεγέθη για την επιβεβαίωση του αποτελέσματος. Αυτή τεκμηριώνεται από τα δεδομένα του πίνακα 3-9 για συγκεκριμένες τιμές του λ και θεωρείται ενδεικτικός της ακρίβειας της μεθόδου με το νευρωνικό δίκτυο. Όπως φαίνεται από το σχήμα 3-21, η λύση έχει πολύ μικρή απόκλιση από εκείνη των FEM και έτσι θεωρείται ικανοποιητική η χρήση του για την αρχική εκτίμηση της λύσης. Αξίζει να τονιστεί ότι το πεδίο που λειτουργεί ικανοποιητικά το νευρωνικό είναι για λ μεγαλύτερο ή ίσο του 1.3.

Έπειτα, η παραμετρική ανάλυση χρησιμοποιείται για να βρεθεί η τιμή $\lambda_{crit}$. Εστιάζοντας στο σημείο όπου γίνεται η στροφή στο σύνολο των λύσεων, εντοπίζονται οι επιθυμητές τιμές. Αυτές φαίνονται στο Σχήμα 3-22 και στον Πίνακα 3-10 δίνεται και κάποια στοιχειώδης στατιστική σύγκριση μεταξύ των μεθόδων.




*Σχήμα 3-22 Εντοπίζοντας την κρίσιμη τιμή της παραμέτρου και την άπειρη νόρμα της συνάρτησης u για το πρόβλημα Bratu 2d με χρήση NN και FEM.*

Από τα δεδομένα του Σχήματος 3-22 ισχύει :

*Πίνακας 3-10. Δεδομένα για τη κρίσιμη τιμή της παραμέτρου λ και στατιστική σύγκριση NN με FEM για το Πρόβλημα Bratu 2D.*

| Method | $\lambda_{crit}$ | $\|u\|_\infty$ | Error (%) in $\lambda_{crit}$ | Error (%) in $\|u\|_\infty$ |
|---|---|---|---|---|
| FEM – Keller's | 6.80764 | 1.40969 | - | - |
| NN- Newton GMRES - Keller's | 6.75376 | 1.40383 | -0.791 | -0.416 |

Το σφάλμα στον υπολογισμό της κρίσιμης παραμέτρου και της άπειρης νόρμας στο κρίσιμο λ παρατηρείται ότι είναι αμελητέο (<1%) με μια μικρή υποεκτίμηση αυτού.

Εν κατακλείδι, μπορεί με βεβαιότητα να υποστηριχθεί ότι το νευρωνικό δίκτυο μπορεί να χρησιμοποιηθεί για την μελέτη προβλήματος Bratu 2D στο εύρος τιμών [1.3 $\lambda_{critical}$] και να δώσει αξιόπιστα αποτελέσματα.

# 4. Συμπεράσματα και προτάσεις

## 4.1. Συμπεράσματα

### 4.1.1. Διεργασίες Gauss

Οι διεργασίες Gauss και ιδιαίτερα η ανάλυση ARD είναι μια πολύ χρήσιμη μέθοδος προσδιορισμού και εκτίμησης της σημαντικότητας διαφόρων μεταβλητών κατά τη μοντελοποίηση ενός φαινομένου. Ωστόσο, είναι πολύ ευαίσθητη στην ποιότητα των δεδομένων που χρησιμοποιούνται, καθώς δίνει ικανοποιητικά αποτελέσματα υπό προϋποθέσεις. Στην συγκεκριμένη εργασία, χρησιμοποιώντας μόνο τις αρχικές μετρήσεις από τα πειράματα, οι συντελεστές που αντιστοιχούν στις λιγότερο σημαντικές μεταβλητές ήταν τουλάχιστον 6 τάξεις μεγέθους μεγαλύτερες από εκείνες των συντελεστών που όντως ορίζουν την εξίσωση και για αυτό το λόγο η εφαρμογή τους θεωρείται επιτυχημένη.

### 4.1.2. Εκπαίδευση Νευρωνικών

Έγινε προσπάθεια η εκπαίδευση να γίνει με τη χρήση όσο ήταν δυνατόν λιγότερων στρωμάτων και νευρώνων με αποτέλεσμα να εκπαιδευτούν πολύ ρηχά νευρωνικά δίκτυα με μικρές συλλογές δεδομένων (περίπου 6000 στην περίπτωση του 1D, ενώ στην περίπτωση του 2D από μια βάση περίπου 120 χιλιάδων λύσεων, χρησιμοποιήθηκαν μόλις τα περίπου 18 χιλιάδες, που αντιστοιχίσει σε ένα ποσοστό της τάξης του 15%). Επίσης, από τη σύγκριση των διάφορων τεχνικών, εκείνη των Levenberg – Marquardt οδήγησε στα καλύτερα αποτελέσματα και έτσι προτείνεται η χρήση αυτού για την εκπαίδευση νευρωνικών που στοχεύουν στην εκτίμηση συνάρτησης / επίλυση διαφορικής εξίσωσης.

Επιπλέον, ο χρόνος εκπαίδευσης των νευρωνικών διήρκησε το πολύ μερικά λεπτά δείχνοντας την αξία του μικρού νευρωνικού δικτύου, καθώς θα μπορούσε η εκπαίδευση να διαρκεί ώρες στην περίπτωση ενός deep neural network και να εμφανιστούν φαινόμενα overfitting.

Τέλος, διαπιστώθηκε ότι το νευρωνικό λειτουργεί επιτυχημένα στην περίπτωση που οι τιμές στις οποίες υπολογίζεται η χρονική παράγωγος περιέχονται στο διάστημα τιμών που χρησιμοποιήθηκε για την εκπαίδευση του. Έτσι, προτείνεται η διενέργεια λίγων πειραμάτων, τα οποία καλύπτουν ένα μεγάλο εύρος, προκειμένου το νευρωνικό να έχει τα καλύτερα αποτελέσματα.

### 4.1.3. Bratu 1D

Διαπιστώθηκε ότι το νευρωνικό δίκτυο αντικαθιστά με επιτυχία το δεξί μέλος της διαφορικής εξίσωσης Bratu 1D. Αυτό επιβεβαιώνεται με τη σύγκριση των αποτελεσμάτων σε τρεις διαφορετικές αρχικές συνθήκες από αυτές που χρησιμοποιήθηκαν για την εκπαίδευση του νευρωνικού και σε χρόνους που κάλυπταν τρεις τάξεις μεγέθους με μέγιστη διαφορά μικρότερη του 1% από τη λύση που υπολογίζει η μέθοδος των πεπερασμένων διαφορών.

Εδώ, αξίζει να τονιστεί ότι για την εκπαίδευση χρησιμοποιήθηκαν μόνο συναρτήσεις ημιτονοειδούς μορφής, που ωστόσο κάλυπταν μεγάλο εύρος τιμών, αλλά και λύσεις από την εκπαίδευση

προηγούμενου νευρωνικού. Έτσι, δεδομένης της μορφής της εξίσωσης δεν απαιτείται η κατάστρωση πολλών πειραμάτων / προσομοιώσεων με διαφορετικές και εξωτικές αρχικές συνθήκες. Αρκούν στοιχειώδεις συναρτήσεις με την προϋπόθεση να επιτρέπουν στο νευρωνικό να λαμβάνει τιμές από ένα σχετικά μεγάλο σύνολο.

Πέρα από τα αξιοσημείωτα αποτελέσματα στην εκτίμηση της δυναμικής συμπεριφοράς, το νευρωνικό είναι ικανό να βρίσκει το σύνολο των μονίμων καταστάσεων, τόσο ευσταθών όσο και ασταθών. Αυτό φαίνεται από τις προσομοιώσεις που έγιναν και από τη σύγκριση αυτών με τη μέθοδο των υπολοίπων Galerkin, που θεωρείται πλέον έγκυρη. Οι διαφορές ήταν πάλι μικρότερες του 1% και στις 3 συνθήκες (ευσταθής λύση, λύση κοντά στην κρίσιμη τιμή της παραμέτρου, ασταθής λύση).

Επίσης, μη σημαντική διαπιστώθηκε και η διακριτοποίηση του πλέγματος στις συνθήκες των προσομοιώσεων (μέχρι και 3 φορές πιο πυκνό πλέγμα). Αν και η εκπαίδευση του νευρωνικού έγινε κυρίως σε δεδομένα αραιού πλέγματος με λίγα δεδομένα από πολύ πυκνό πλέγμα, δεν χάνει την διακριτική του ικανότητα ακόμα και στην περίπτωση που χρησιμοποιούνται τρεις φορές πιο πυκνά πλέγματα.

Τέλος, κατά την παραμετρική ανάλυση, η λύση του νευρωνικού παρουσιάζει ακριβώς την ίδια συμπεριφορά με εκείνη της μεθόδου των πεπερασμένων στοιχείων, ενώ η διαφορά στην εκτίμηση του $\lambda_{critical}$ είναι της τάξης του 0.01% που αποδεικνύει την ικανότητα του να προσεγγίζει την αναλυτική εξίσωση.

### 4.1.4. Bratu 2D

Διαπιστώθηκε ότι το νευρωνικό δίκτυο προσεγγίζει με επιτυχία το δεξί μέλος της διαφορικής εξίσωσης Bratu 2D. Αυτό επιβεβαιώνεται με τη σύγκριση των αποτελεσμάτων με μία διαφορετική αρχική συνθήκη από αυτές που χρησιμοποιήθηκαν για την εκπαίδευση του νευρωνικού και σε χρόνους που προσεγγίζουν την μόνιμη κατάσταση με μέγιστη περίπου 2% από τη λύση που υπολογίζει η μέθοδος των πεπερασμένων διαφορών.

Εδώ, αξίζει να τονιστεί ότι για την εκπαίδευση χρησιμοποιήθηκαν διάφορες συναρτήσεις που κάλυπταν μεγάλο εύρος τιμών. Επιβεβαιώνεται ότι αρκούν στοιχειώδεις συναρτήσεις για την εκπαίδευση του νευρωνικού, με την προϋπόθεση να του επιτρέπουν να λαμβάνει τιμές από ένα σχετικά μεγάλο σύνολο. Ωστόσο, η συμπερίληψη μεγάλου εύρους στις τιμές που μπορεί να λάβει το νευρωνικό αυξάνει το σφάλμα στην εκτίμηση των μικρότερων ευσταθών λύσεων με αποτέλεσμα να είναι έγκυρο για συγκεκριμένο εύρος τιμών της παραμέτρου.

Πέρα από τα αξιοσημείωτα αποτελέσματα στην εκτίμηση της δυναμικής συμπεριφοράς, το νευρωνικό είναι ικανό να βρίσκει το σύνολο των μονίμων καταστάσεων, τόσο ευσταθών όσο και ασταθών. Αυτό φαίνεται από τις προσομοιώσεις που έγιναν και από τη σύγκριση αυτών με τη μέθοδο των υπολοίπων Galerkin, που θεωρείται πλέον έγκυρη. Οι διαφορές ήταν πάλι περίπου στο 2% και στις 2 συνθήκες (ευσταθής λύση, ασταθής λύση).

Τέλος, κατά την παραμετρική ανάλυση, η λύση του νευρωνικού ακολουθεί την ίδια συμπεριφορά με εκείνη της μεθόδου των πεπερασμένων στοιχείων, ενώ η διαφορά στην εκτίμηση του $\lambda_{critical}$ είναι της τάξης του 1% που αποδεικνύει την ικανότητα του να προσεγγίζει την αναλυτική εξίσωση.

Συμπερασματικά, η σύζευξη του νευρωνικού δικτύου με τις υπολογιστικές μεθόδους δίνει νέες ευκαιρίες στην αντιμετώπιση φυσικών προβλημάτων ως καθαρά μαθηματικών και την ανάλυση όλων των λύσεων τους, χωρίς να υπάρχει η ανάγκη κατανόησης των φαινομένων που λαμβάνουν χώρα και εφαρμογής των νόμων που τα διέπουν.

#### 4.1.5. Μεθοδολογία

Πρόκειται για μια μεθοδολογία η οποία επιτρέπει την ανίχνευση από δεδομένα των κυρίαρχων μηχανισμών (διάχυση, συναγωγή) και βέβαια από λίγες σχετικά μετρήσεις την εύρεση / προσέγγιση της εξίσωσης. Αυτό γίνεται γιατί οι μηχανισμοί εκδηλώνονται με το δεξί μέλος της ΜΔΕ.

Από λίγες χρονικές στιγμές είναι δυνατή η εκτίμηση της εξίσωσης και με την εφαρμογή αριθμητικών τεχνικών καθίσταται πραγματοποιήσιμος ο υπολογισμός λύσεων μόνιμης κατάστασης σε πολύ λιγότερο χρόνο και η ανίχνευση λύσεων σε χώρους που δεν έχουν διερευνηθεί.

Δεν χρειάζεται πλέον να είναι γνωστή η διαφορική που διέπει το φαινόμενο που μελετάται. Η μεθοδολογία αυτή είναι τύπου black or gray box. Επιτρέπει την πλήρη εξερεύνηση όλων των χώρων μονίμων καταστάσεων, καθώς επίσης και της δυναμικής συμπεριφοράς της διαφορικής εξίσωσης, γνωρίζοντας λίγους σημαντικούς όρους της ή ακόμα και τίποτα για αυτή, αρκεί να υπάρχουν πειραματικά δεδομένα ή αποτελέσματα προσομοιώσεων.

### 4.2. Προτάσεις

#### 4.2.1. Επανεκπαίδευση Νευρωνικού Δικτύου για το πρόβλημα Bratu 2D

Προτείνεται η συνέχεια στην εκπαίδευση των τιμών του νευρωνικού ώστε να καλυφθούν και οι αρχικές τιμές της παραμέτρου και να υπάρξει πιο επιτυχημένη ταύτιση με τη λύση των Πεπερασμένων στοιχείων.

#### 4.2.2. Χρήση διαφορετικών συναρτήσεων πυρήνα (kernel function) στις Διεργασίες Gauss

Εκτός από την κλασική συνάρτηση που είναι η radial basis function kernel (τετραγωνικός εκθετικός πυρήνας) μπορούν να χρησιμοποιηθούν και άλλα είδη και να εξεταστεί η απόδοση τους στην εύρεση των σημαντικών μεταβλητών (όπως τον εκθετικό ημιτονοειδή τετραγωνικό πυρήνα, τον  σταθερό και τον λογικό τετραγωνικό ή και συνδυασμό όλων των προηγούμενων.

#### 4.2.3. Χρήση άλλων μεθόδων για την επίτευξη της διαστασιολογικής μείωσης

Θα μπορούσαν να χρησιμοποιηθούν κι άλλες τεχνικές για την μείωση της τάξης των διαστάσεων (μέσο την επιλογή λιγότερων μεταβλητών). Αυτό μπορεί να γίνει με τη χρήση της Χαρτογράφησης Διάχυσης (Diffusion Maps) ή με την Ανάλυση Κύριων Συνιστωσών (Principal Component Analysis).

### 4.2.4. Σύγκριση με άλλες τεχνικές μηχανικής μάθησης

Εκτός από το νευρωνικό δίκτυο (FFNN) θα μπορούσαν να χρησιμοποιηθούν και άλλες τεχνικές μηχανικής μάθησης, όπως είναι οι διεργασίες Gauss που εδώ χρησιμοποιήθηκαν μόνο για την εύρεση των σημαντικότερων μεταβλητών. Επιπλέον, θα μπορούσε να γίνει σύγκριση και με τα νευρωνικά δίκτυα με πληροφόρηση Φυσικής (PINN) και να γίνει σύγκριση της πιο απλής δομής με την πιο περίπλοκη των τελευταίων.

### 4.2.5. Μελέτη πιο περίπλοκων και σύνθετων προβλημάτων

Προτείνεται η χρήση των μεθόδων για την αντιμετώπιση πιο περίπλοκων φαινομένων, όπως είναι το μονοδιάστατο μοντέλο Poiseuille ροής σε Oldroyd-B ρευστό που εμφανίζει και περιοδικότητα στις λύσεις του (ύπαρξη Hopf Point).

Επίσης, προτείνεται να εφαρμοσθεί αυτή η μεθοδολογία και με δεδομένα πειραματικά, εκτός από αυτά προσομοιώσεων.

# 5. Βιβλιογραφία


[1] Boyce, W.E.; DiPrima, R.C. *Elementary Differential Equations and Boundary Value Problems*, 10th ed.; Wiley, 2012.

[2] Πολυράκης, Ι.Α. *Συνήθεις Διαφορικές Εξισώσεις*, 2η έκδοση; Ιδία Έκδοση, 1992.

[3] Nagle, R.K.; Saff, E.B.; Snider, A.D. *Fundamentals of Differential Equations*, 8th ed.; Pearson Education, 2011.

[4] Tenenbaum, M.; Pollard, H. *Ordinary Differential Equations* ; Dover Publications, 2012.

[5] Simmons, G. F. *Differential Equations with Applications and Historical Notes*, 3rd ed.; Springer, 2012.

[6] Mohsen, A. A simple solution of the Bratu problem. *CAMWA* **2014**, *67* (1), 26-33. DOI: 10.1016/j.camwa.2013.10.003

[7] Dickson, K.I.; Kelley, C.T.; Ipsen, I.C.F.; Kevrekidis, I.G. Condition Estimates for Pseudo-Arclength Continuation. *SINUM* **2007,** *45* (1), 263–276. DOI: 10.1137/060654384

[8] Brenner, S.C.; Scott, L.R. *The Mathematical Theory of Finite Element Methods*, 3rd ed.; Springer, 2008.

[9] LeVeque, R.J. *Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-State and Time-Dependent Problems*; SIAM, 2007.

[10] Boyd, J.P. *Chebyshev and Fourier Spectral Methods*, 2nd ed.; Dover Publications, 2000.

[11] Wan, Y.Q.; Guo, Q.; Pan, N. Thermo-electro-hydrodynamic model for electro spinning process. *Int. J. Nonlinear Sci. Numer. Simul.* **2004**, *5* (1), 5–8. DOI: 10.1515/IJNSNS.2004.5.1.5

[12] Hosseini, V.R.; Mehrizi, A.A.; Gungor A.; Afrouzi H.H. Application of a physics-informed neural network to solve the steady-state Bratu equation arising from solid biofuel combustion theory. *Fuel* **2023**, *332* (2), 125908. DOI: 10.1016/j.fuel.2022.125908

[13] Jalilian, R. Non-polynomial spline method for solving Bratu's problem. *CPC* **2010**, *181* (11), 1868-1872. DOI: 10.1016/j.cpc.2010.08.004.

[14] Hastie, T.; Tibshirani, R.; Friedman, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* , 2nd ed.; Springer, 2009.

[15] Bishop, C.M. *Pattern Recognition and Machine Learning*; Springer, 2006.

[16] Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press, 2016.

[17] Sutton, R.S.; Barto, A.G.; *Reinforcement Learning*, 2nd ed.; MIT Press, 2018.

[18] Hagan, M.T.; Demuth, H.B.; Hudson Beale, M.; De Jesus, O. *Neural Network Design*, 2nd ed.; Martin Hagan, 2014.

[19] Levenberg K. A Method for the Solution of Certain Non-Linear Problems in Least Squares. *Quart. Appl. Math.* **1944**, *2* (2), 164-168. DOI: 10.1090/qam/10666

[20] Marquardt, D.W. An Algorithm for Least-Squares Estimation of Nonlinear Parameters. *J. Soc. Indust. Appl. Math.* **1963**, *11* (2), 431-441. DOI: 10.1137/0111030

[21] Press, W.H. ; Teukolsky, S.A.; Vetterling W.T.; Flannery B.P. *Numerical Recipes: The Art of Scientific Computing*, 3rd ed.; Cambridge University Press, 2007.

[22] Nocedal, J.; Wright, S. J. *Numerical Optimization*, 2nd Ed.; Springer, 2006.

[23] Bishop, C.M. *Neural Networks for Pattern Recognition*; Oxford University Press,1995.

[24] Haykin, S. *Neural Networks and Learning Machines*, 3rd ed.; Prentice Hall, 2009.

[25] Sirignano, J.; Spiliopoulos, K. DGM: A deep learning algorithm for solving partial differential equations. *J. Comput. Phys.* **2018**, *375*, 1339-1364. DOI: 10.1016/j.jcp.2018.08.029

[26] Blechschmidt, J.; Ernst, G.E. Three ways to solve partial differential equations with neural networks -A review. *GAMM- Mitteilungen* **2021**, *44* (2), e202100006. DOI: 10.1002/gamm.202100006

[27] Brunton, S. L.; Kutz, J. N. Machine Learning for Partial Differential Equations. *arXiv (Machine Learning, Computer Science),* March 30, 2023, 2303.17078, Ver. 1. https://arxiv.org/pdf/2303.17078.pdf (accessed 2023-06-22)

[28] Xu, K.; Shi, B.; Yin, S. *Deep Learning for Partial Differential Equations (PDEs)*; Stanford CS230,2018. https://cs230.stanford.edu/projects_spring_2018/reports/8287640.pdf (accessed 2023-06-22)

[29] Han, J.; Jentzen, A.; Weinan, E. Solving high-dimensional partial differential equations using deep learning. *PNAS* **2018**, *115* (34), 8505-8510. DOI: 10.1073/pnas.1718942115

[30] Raissi, M.; Karniadakis G.E. Hidden Physics Models: Deep Learning of Nonlinear Partial Differential Equations. *J. Comput. Phys.* **2018**, *357*, 125-141. DOI: 10.1016/j.jcp.2017.11.039

[31] Rackauckas, C.; Ma, Y.; Martensen, J.; Warner, C.; Zubov, K.; Supekar, R.; Skinner, D.; Ramadhan, A.; Edelman, A. Universal Differential Equations for Scientific Machine Learning. *arXiv (Machine Learning, Computer Science*), January 2, 2021, 2001.04385, Ver. 4, https://arxiv.org/ pdf/ 2001.04385.pdf (accessed 2023-06-22)

[32] Raissi, M.; Perdikaris, P.; Karniadakis G.E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. . *J. Comput. Phys*. **2018**, *378*, 686-707. DOI: 10.1016/j.jcp.2018.10.045

[33] Raissi, M.; Yazdani, A.; Karniadakis G.E. Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science* **2020**, *367* (6481),1026-1030. DOI: 10.1126/science.aaw4741

[34] Berg J.; Nyström, K. Data-driven discovery of PDEs in complex datasets. J. Comput. Phys. **2019**, *384* (4), 686-707. DOI: 10.1016/j.jcp.2019.01.036

[35] Rasmussen, C.E.; Williams, C.K.I. *Gaussian Processes for Machine Learning*; MIT Press, 2016.

[36] Roberts, S.; Osborne, M.; Ebden, M.; Reece, S.; Gibson, N.; Aigrain, S. Gaussian Processes for Time-Series Modelling. Philos. Trans. Royal Soc. A **2013,** *371*, 20110550. DOI: 10.1098/rsta.2011.0550

[37] Snoek, J.; Larochelle, H.; Adams, R.P.. Practical Bayesian Optimization of Machine Learning Algorithms. https://arxiv.org/pdf/1206.2944.pdf *(Machine Learning*), August 29, 2012, 1206.2944, Ver. 4, (accessed 2023-06-22)

[38] Raissi, M.; Perdikaris, P.; Karniadakis G. E., Machine learning of linear differential equations using Gaussian processes. *J. Comput. Phys.* **2017**, *348*, 683–693. DOI: 10.1016/j.jcp.2017.07.050

[39] Lee, S.; Kooshkbaghi, M.; Spiliotis, K.; Siettos, C. I.; Kevrekidis, I.G. Coarse-scale PDEs from fine-scale observations via machine learning. *Chaos* **2020**, *30*, 013141. DOI: 10.1063/1.5126869

[40] Pashos, G.; Kavousanakis, M.E.; Spyropoulos, A.N.; Palyvos J.A.; Boudouvis, A.G. Simultaneous solution of large-scale linear systems and eigenvalue problems with a parallel GMRES method. *J. Comput. Appl. Math* **2009**, *227* (1), 196-205. DOI: 10.1016/j.cam.2008.07.012.

[41] Saad, Y.; Schultz M.H. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems, *SIAM J. Sci. Stat. Comp.* **1986,** *7*, 856–869. DOI: 10.1137/0907058

[42] Joubert, W. On the convergence behavior of the restarted GMRES algorithm for solving nonsymmetric linear systems. *Numer. Linear Algebra Appl.* **1994**, *1* (5), 427–447. DOI: 10.1002/ nla.1680010502

[43] Kelley, C.T. *Solving Nonlinear Equations with Newton's Method*; SIAM,2003

[44] Jokar, H.; Vatankhah, R. ; Mahzoon, M., Nonlinear vibration analysis of horizontal axis wind turbine blades using a modified pseudo arc-length continuation method. *Eng. Struct*. **2021**, *247*, 113103, DOI: 10.1016/j.engstrct.2021.113103.

[45] https://www.mathworks.com/products/matlab.html (accessed 2023-06-22)

[46] http://www.comsol.com/. (accessed 2023-06-22)

# 6. Παράρτημα

## 6.1. Επίλυση μονοδιάστατου και δισδιάστατου προβλήματος Bratu με τη μέθοδο πεπερασμένων στοιχείων (FEM) / υπολοίπων Galerkin (Finite Elements Method / Galerkin Residuals)

### 6.1.1. Παρουσίαση Προβλήματος και Κατάστρωση Υπολοίπων Galerkin 2D

Η εξίσωση Bratu περιγράφεται από το ακόλουθο ζεύγος εξισώσεων, τη διαφορική εξίσωση (6.1α) και τις συνοριακές συνθήκες (6.1β):

$$\nabla^2 u(x,y) + \lambda * \exp(u) = 0, \qquad (x,y) \in \Omega\,, \qquad \Omega = (0,1)x(0,1)\,, \tag{6.1α}$$

$$u = 0, (x,y) \in \partial\Omega, \qquad \partial\Omega = \{x,y\ \in R, \qquad x = 0 \text{ ή } x = 1\,, y = 0 \text{ ή } y = 1\,\}. \tag{6.1β}$$

Όπου u: συνάρτηση των x,y (μπορεί να εκφράζει θερμοκρασία ή συγκέντρωση κάποιου αντιδρώντος σε μια εξίσωση λειτουργίας ενός αντιδραστήρα) και λ: παράμετρος που υπάρχει στην εξίσωση και θα μπορούσε να ειπωθεί ότι εξαρτάται από τις συνθήκες του πειράματος και είναι ανεξάρτητη μεταβλητή.

Για την εύρεση των υπολοίπων Galerkin ακολουθήθηκε η εξής πορεία επίλυσης (6.2-6.9):

$$R_i = \iint_D (\nabla^2 u - f)\varphi^i \, dx\, dy = 0 \Rightarrow R_i = \iint_D \nabla^2 u * \varphi^i \, dx\, dy - \iint_D f * \varphi^i \, dx\, dy = 0. \tag{6.2}$$

Χρησιμοποιώντας τον κανόνα αλυσίδας προκύπτει:

$$R_i = \iint_D \nabla(\varphi^i \nabla u)\, dx\, dy - \iint_D \nabla\varphi^i \nabla u \, dx\, dy - \iint_D f * \varphi^i \, dx\, dy = 0, \tag{6.3}$$

$$R_i = \iint_D \frac{\partial}{\partial x}\left(\varphi^i \frac{\partial u}{\partial x}\right) + \frac{\partial}{\partial y}\left(\varphi^i \frac{\partial u}{\partial y}\right) dx\, dy - \iint_D \frac{\partial \varphi^i}{\partial x} * \frac{\partial u}{\partial x} dx\, dy - \iint_D \frac{\partial \varphi^i}{\partial y} * \frac{\partial u}{\partial y} dx\, dy - \iint_D f * \varphi^i \, dx\, dy = 0\,. \tag{6.4}$$

Θέτοντας :

$$I_{xi} = \iint_D \frac{\partial \varphi^i}{\partial x} * \frac{\partial u}{\partial x} dx\, dy, \tag{6.5}$$

$$I_{yi} = \iint_D \frac{\partial \varphi^i}{\partial y} * \frac{\partial u}{\partial y} dx\, dy\,, \tag{6.6}$$

$$\kappa\alpha\iota \quad I_{fi} = \iint_D f * \varphi^i \, dx \, dy. \tag{6.7}$$

Και με τη χρήση του θεωρήματος Green προκύπτει:

$$\iint_D \left(\frac{\partial P}{\partial x} + \frac{\partial Q}{\partial y}\right) dx \, dy = \oint_{\partial D} P dy - Q dx, \tag{$\Theta.Green$, 6.8}$$

$$R_i = \oint_{\partial D} \varphi^i \frac{\partial u}{\partial x} dy - \varphi^i \frac{\partial u}{\partial y} dx - I_{xi} - I_{yi} - I_{fi} = 0. \tag{6.9}$$

Και τελικά χωρίζοντας το σύνορο ∂D στις στοιχειώδεις 4 πλευρές του (πρόκειται για τετράγωνο με κορυφές τις 0ABC, με 0 την κάτω αριστερή κορυφή και ακολουθώντας αντιωρολόγια φορά για την ονομασία των άλλων) καταλήγει κανείς στον εξής τύπο (6.10):

$$R_i = -\int_0^A \varphi^i \frac{\partial u}{\partial y} dx + \int_A^B \varphi^i \frac{\partial u}{\partial x} dy - \int_B^C \varphi^i \frac{\partial u}{\partial y} dx + \int_C^0 \varphi^i \frac{\partial u}{\partial x} dy - I_{xi} - I_{yi} - I_{fi} = 0. \tag{6.10}$$

Ισχύει ότι τα άνωθι σημεία έχουν τις εξής συντεταγμένες : 0(0,0), A(1,0), B(1,1), C(0,1)

Για όλους τους κόμβους ισχύει ότι f=λ*exp(u), άρα και

$$\mathrm{I}_{\mathrm{fi}} = -\iint_D \lambda * \exp(\mathrm{u}) * \varphi^i \, dx \, dy. \tag{6.7′}$$

Οπότε η εξίσωση γίνεται (8.11) :

$$R_i = -\int_0^1 \varphi^i \frac{\partial u}{\partial y} dx + \int_0^1 \varphi^i \frac{\partial u}{\partial x} dy - \int_1^0 \varphi^i \frac{\partial u}{\partial y} dx + \int_1^0 \varphi^i \frac{\partial u}{\partial x} dy - I_{xi} - I_{yi} - I_{fi} = 0. \tag{6.11}$$

Τώρα η παραπάνω γενική εξίσωση δύναται να χρησιμοποιηθεί για την εύρεση των υπολοίπων όλων των κόμβων.

Έτσι, αντικαθιστώντας στις παραπάνω εξισώσεις και τη σχέση (6.12) προκύπτει η (6.13) :

$$u = \sum_j u_j * \varphi^j \,, \tag{6.12}$$

$$R_i = -\int_0^1 \varphi^i \frac{\partial u}{\partial y} dx + \int_0^1 \varphi^i \frac{\partial u}{\partial x} dy - \int_1^0 \varphi^i \frac{\partial u}{\partial y} dx + \int_1^0 \varphi^i \frac{\partial u}{\partial x} dy + \sum_j u_j * \left( - \int_0^1 \int_0^1 \frac{\partial \varphi^i}{\partial x} * \frac{\partial \varphi^j}{\partial x} dx \, dy - \int_0^1 \int_0^1 \frac{\partial \varphi^i}{\partial y} * \frac{\partial \varphi^j}{\partial y} dx \, dy \right)$$
$$+ \int_0^1 \int_0^1 \lambda * \exp\left( \sum_j u_j * \varphi^j \right) * \varphi^i \, dx \, dy = 0 \,. \tag{6.13}$$

Παρατηρείται, λοιπόν, ότι από την παραπάνω σχέση (6.13) προκύπτει ένα σύστημα μη γραμμικών εξισώσεων ($R_i = 0$, i=1,..,nn, όπου nn:number of nodes, αριθμός κόμβων), οπότε χρειάζεται η χρήση κάποιας επαναληπτικής μεθόδου ώστε να λυθεί η εξίσωση. Η σχέση που επιλέγεται είναι η Newton-Raphson για μη γραμμικά προβλήματα.

Η σχέση που πρέπει να επιλυθεί είναι (σχέση (6.14)) :

$$\boldsymbol{J}^{(k)}\delta\boldsymbol{u}^{(k+1)} = \boldsymbol{R}^{(k)}\,, \tag{6.14}$$

όπου, $\delta\boldsymbol{u}^{(k+1)} = \boldsymbol{u}^{(k+1)} - \boldsymbol{u}^{(k)}$. Ο πίνακας $\boldsymbol{J}$ είναι η ιακωβιανή του συστήματος και τα στοιχεία τους είναι οι μερικές παράγωγοι των υπολοίπων Galerkin ως προς τους κομβικούς αγνώστους, δηλαδή (6.15)

$$J_{ij} = \frac{\partial Ri}{\partial uj}. \tag{6.15}$$

Ο ακέραιος k είναι ο μετρητής των επαναλήψεων (k = 0, 1, 2, …) . Έτσι, σε αναπτυγμένη μορφή ο πίνακας γράφεται :

$$\begin{bmatrix} \frac{\partial R_1^{(k)}}{\partial u_1^{(k)}} & \frac{\partial R_1^{(k)}}{\partial u_2^{(k)}} & \cdots & \frac{\partial R_1^{(k)}}{\partial u_N^{(k)}} \\ \frac{\partial R_2^{(k)}}{\partial u_1^{(k)}} & \frac{\partial R_2^{(k)}}{\partial u_2^{(k)}} & \cdots & \frac{\partial R_2^{(k)}}{\partial u_N^{(k)}} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\partial R_N^{(k)}}{\partial u_1^{(k)}} & \frac{\partial R_N^{(k)}}{\partial u_2^{(k)}} & \cdots & \frac{\partial R_N^{(k)}}{\partial u_N^{(k)}} \end{bmatrix} \begin{bmatrix} u_1^{(k+1)} - u_1^{(k)} \\ u_2^{(k+1)} - u_2^{(k)} \\ \vdots \\ u_N^{(k+1)} - u_N^{(k)} \end{bmatrix} = - \begin{bmatrix} R_1^{(k)} \\ R_2^{(k)} \\ \vdots \\ R_N^{(k)} \end{bmatrix}.$$

Αναλύοντας για καθένα από τα i, δηλαδή για καθένα από τα στοιχεία του διαστήματος προκύπτει η εξής ανάλυση(6.16$^{\alpha}$-6.16$^{\beta}$, 6.17$^{\alpha}$-6.17$^{\beta}$):

**Για όλους τους εσωτερικούς κόμβους**, τα ολοκληρώματα των συνόρων μηδενίζονται . όποτε για αυτούς ισχύει (Περίπτωση I):

$$\mathrm{R_i} - \mathrm{I_{xi}} - \mathrm{I_{yi}} - \mathrm{I_{fi}} = 0$$

Και στην εξίσωση επίλυσης , το ij-στοιχείο της Ιακωβιανής στην k-επανάληψη θα είναι (6.16$^{\alpha}$-6.16$^{\beta}$) :

$$R_i^{(k)} = \sum_j u_j^{(k)} * \left( -\int_0^1 \int_0^1 \frac{\partial \varphi^i}{\partial x} * \frac{\partial \varphi^j}{\partial x} dx\, dy - \int_0^1 \int_0^1 \frac{\partial \varphi^i}{\partial y} * \frac{\partial \varphi^j}{\partial y} dx\, dy \right) + \int_0^1 \int_0^1 \lambda * \exp\left( \sum_j u_j^{(k)} * \varphi^j \right) * \varphi^i\, dx\, dy = 0, \qquad (6.16^{\alpha})$$

$$J_{ij}^{(k)} = \frac{\partial R_i^{(k)}}{\partial u_j} = -\int_0^1 \int_0^1 \frac{\partial \varphi^i}{\partial x} * \frac{\partial \varphi^j}{\partial x} dx\, dy - \int_0^1 \int_0^1 \frac{\partial \varphi^i}{\partial y} * \frac{\partial \varphi^j}{\partial y} dx\, dy + \int_0^1 \int_0^1 \lambda * \exp(u_i^k)\, \varphi^j * \varphi^i\, dx\, dy. \qquad (6.16^{\beta})$$

**Για τους κόμβους των συνόρων** θα ισχύει η $6.17^{\alpha}$-$6.17^{\beta}$ (Περίπτωση ΙΙ):

u=0 (Συνθήκη Dirichlet)

$$R_i^{(k)} = u_i^{(k)} - u_0 = 0\,, \qquad (6.17^{\alpha})$$

Και στην εξίσωση επίλυσης θα είναι :

$$J_{ij}^{(k)} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}. \qquad (6.17^{\beta})$$

### 6.1.2. Σχόλια Επίλυσης

- Ως αρχική εκτίμηση λαμβάνεται η μηδενική **$u_0$=0**. Παρατηρείται ότι όταν η αρχική εκτίμηση είναι της μορφής **$u_0$=a*sin(pi*x)*sin(pi*y)** , τότε ανάλογα με την τιμή της μεταβλητής a, ο αλγόριθμος συγκλίνει είτε στον ευσταθή είτε στον ασταθή κλάδο.
- Κριτήριο σύγκλισης ορίζεται ο κατά προσέγγιση μηδενισμός της διαφοράς δύο διαδοχικών προσεγγίσεων, δηλαδή (6.18-6.19):
  $\delta \boldsymbol{u}^{(k+1)} = \boldsymbol{u}^{(k+1)} - \boldsymbol{u}^{(k)}$, (6.18)

  $$\left\| \delta \boldsymbol{u}^{(k+1)} \right\| = \sqrt{\sum_{j=1}^{nn} u_j^{(k+1)} - u_j^{(k)}} < \varepsilon. \qquad (6.19)$$

  Όπου ε : μικρή προκαθορισμένη τιμή η οποία λαμβάνεται ίση με $10^{-6}$ στις προσομοιώσεις.
- Για την περίπτωση της μονοδιάστατης εξίσωσης, ισχύουν ακριβώς τα ίδια υπόλοιπα κι οι ίδιες εξισώσεις μόνο που μηδενίζεται ο όρος $I_{yi}$ , τα ολοκληρώματα από διπλά γίνονται μονά και ισχύει ότι u=u(x). Παρατηρείται ότι όταν η αρχική εκτίμηση είναι της μορφής **$u_0$=a*sin(pi*x)** , τότε ανάλογα με την τιμή της μεταβλητής a, ο αλγόριθμος συγκλίνει είτε στον ευσταθή είτε στον ασταθή κλάδο.
- Όλοι οι κώδικες των πεπερασμένων στοιχείων έχουν βασιστεί στους κώδικες που αναπτύσσονται στο μάθημα επιλογής «Υπολογιστική Ανάλυση Φαινομένων Μεταφοράς» του $6^{ου}$ εξαμήνου της Σχολής Χημικών Μηχανικών του ΕΜΠ.

Για ευκολία και ακρίβεια, τα ολοκληρώματα υπολογίζονται με τη βοήθεια των σημείων Gauss (κανόνες ολοκλήρωσης Gauss) και χρησιμοποιείται ισοπαραμετρική απεικόνιση, ενώ σαν συναρτήσεις βάσεις επιλέγονται δευτεροβάθμια πολυώνυμα.

### 6.1.3. Κώδικας Επίλυσης Bratu 1D

```
function[c]=NewtonBratu(nez,el)
% Computational solution with FEM method (Galerkin residual,
% Second order polynomial basis function) of the Bratu problem in 1D.
% uxx  + lamda*exp(u) = 0 , D
% u = 0 , δD
r1_new=[];Jo=[];nop=[];azpt=[];ph=[];phd=[];

ngl=zeros(3,1);
phz=zeros(3,1);
nnz=2*nez+1;

calc=1;
L=1;
z0=0;

maxloop=2000;
discr();
yo = zeros(1,nnz) ;

%x=linspace(0,1,nnz);
%yo=4*sin(pi*x);

c_old=yo';
deltaz = (L-z0)/nnz ;
JodcR();

for iter=calc:maxloop
    r1=r1_new' ;
    delta_c = Jo\(-r1);
    c = c_old+delta_c;
    c_old=c;
    ea=norm(delta_c,2);
    JodcR();
    if ea<=10^(-6), break; end
end

%eig(Jo)% finds the eigenvalus for stability analysis
%c=c';
%plot(azpt,c,'r')

function discr()
        %***  define values of parameters

        %***  x-coordinates
        zfirst=0.;
        zlast=L;
        deltaz=(zlast-zfirst)/nez;
        azpt(1)=zfirst;

        for i=2:nnz
              azpt(i)=azpt(i-1)+deltaz/2.;
        end

        %***  nodal numbering ***
        for i=1:nez
            nop(i,1)=1+2*(i-1);
            nop(i,2)= nop(i,1)+1;
            nop(i,3)= nop(i,2)+1;
        end
        %************************
    end
```

```
function JodcR () %make the tables
  for i=1:nnz
      r1_new(i)=0;
            for j=1:nnz
                Jo(i,j)=0.;
            end
  end
      for nell=1:nez
          JoRfind(nell);
      end

      for i=1:nnz
          Jo(1,i)=0;
          Jo(end,i)=0;
      end
        Jo(1,1)     = 1;
        Jo(end,end) = 1;
        r1_new(1)   = c_old(1);
        r1_new(end) = c_old(end);
end

function tsfun(z)

        q=z;
        ph(1)=2.*q^2-3.*q+1.;
        ph(2)=-4.*q^2+4.*q;
        ph(3)=2.*q^2-q;
        phd(1)=4.*q-3.;
        phd(2)=-8.*q+4.;
        phd(3)=4.*q-1.;

end

function JoRfind(nell)

        w  = [0.27777777777778, 0.444444444444, 0.27777777777778];
        gp = [0.1127016654    , 0.5           , 0.8872983346    ];


        for i=1:3
            ngl(i)=nop(nell,i);

        end


        for j=1:3 %Loop over the qauss points

            tsfun(gp(j));

            z=0.;
            z1=0.;

            for n=1:3
                z=z+azpt(ngl(n))*ph(n);
                z1=z1+azpt(ngl(n))*phd(n);
            end

            for i=1:3
                phz(i)=phd(i)/z1;

            end
```

```
            c_old_gp=0;
            dc_old_gp=0;

            for gpoints=1:3
              c_old_gp=c_old_gp+c_old(ngl(gpoints))*ph(gpoints);
              dc_old_gp=dc_old_gp+c_old(ngl(gpoints))*phz(gpoints);
            end

            for m=1:3

                m1=ngl(m);
                     r1_new(m1)= r1_new(m1)...
                    -w(j)*phz(m)*dc_old_gp*z1...
                    +w(j)*el*exp(c_old_gp)*ph(m)*z1 ;

                for n=1:3

                    n1=ngl(n);

                    Jo(m1,n1)=Jo(m1,n1) ...
                     -w(j)*z1*phz(m)*phz(n)            ...
                     +w(j)*el*ph(m)*ph(n)*exp(c_old_gp)*z1;


                 end   % loop for n


           end       % loop for


       end % end of the  loop over the gauss points

 end %JoRfind

end
```

### 6.1.4. Κώδικας Επίλυσης Bratu 2D

```
function [xpt, ypt, u] = Bratu2D(nex,ney,el)
% 2-dimensional Bratu problem.
% Second order polynomial finite element basis functions.

%Global variables
calc=1;
nnx = 2*nex+1; % node number in x-axis.
nny = 2*ney+1; % node number in y-axis.
ne  = nex*ney; % total element number.
np  = nnx*nny; % total node number.
u = []; r1_new=[];Jo=[];
xorigin = []; yorigin = []; xlast = []; ylast = [];
deltax = []; deltay = []; xpt = []; ypt = [];
nop = [];
ncod = [];
w = []; gp =[];
phi = []; phic = []; phie = [];
maxloop=30;
L=1.;  % L,w defines the domain of the problem [0,1]x[0,1]
```

```
w=1.;
u_old=zeros(np,1);
u=u_old;

xydiscr() %function that is used for the discretization of the domain
nodnumb() %function that numbers the node
xycoord() % *** (x,y) coordinates of each node

%fprintf('nex=%d, ney=%d, ne=%d, np=%d\n',nex,ney,ne,np)

% prepare for essential boundary conditions
ncod  =  zeros(np,1);
%bc   = zeros(np,1);

%ncod(1:nny) = 1; there is no need to imply new bc, because these nodes are
%bc(1:nny)   = 0; treated like inner nodes, as shown in residuals

ncod(1:1:nny)           =  1;
%bc(1:1:nny)            =  0;

ncod(nny:nny:np)        =  1;
%bc(nny:nny:np)         =  0;

ncod(1:nny:np-nny+1)    =  1;
%bc(1:nny:np-nny+1)     =  0;

ncod(np-nny+1:1:np)     =  1;
%bc(np-nny+1:1:np)      =  0;


JodcR();


for iter=calc:maxloop
    r1=r1_new';
    delta_u = Jo\-(r1);
    u = u_old+delta_u;
    u_old=u;
    ea=norm(delta_u,2);
    %disp(ea)
    JodcR();
    if ea <=10^(-6), break; end
end


    function xydiscr()
        xorigin = 0.;
        yorigin = 0.;
        xlast   = L; %the new length
        ylast   = w; %the new width
        deltax = (xlast-xorigin)/nex;
        deltay = (ylast-yorigin)/ney;
    end

    function nodnumb()
    % *** nodal numbering
      nel=0;
        for i=1:nex
            for j=1:ney
                nel=nel+1;
                for k=1:3
                    l=3*k-2;
                    nop(nel,l)=nny*(2*i+k-3)+2*j-1;
```

```
                nop(nel,l+1)=nop(nel,l)+1;
                nop(nel,l+2)=nop(nel,l)+2;
            end
        end
    end
end

function xycoord()
% *** (x,y) coordinates of each node
    xpt=zeros(np,1);
    ypt=zeros(np,1);
    xpt(1)=xorigin;
    ypt(1)=yorigin;
    for i=1:nnx
        nnode=(i-1)*nny+1;
        xpt(nnode)=xpt(1)+(i-1)*deltax/2.;
        ypt(nnode)=ypt(1);
        for j=2:nny
            xpt(nnode+j-1)=xpt(nnode);
            ypt(nnode+j-1)=ypt(nnode)+(j-1)*deltay/2.;
        end
    end
end

function JodcR () %make the tables

    for nj=1:np
        r1_new(nj)=0;
        for j=1:np
            Jo(nj,j)=0;
        end
    end

    for nell=1:ne
        JoRfind(nell);
    end

    for hj=1:np

         if(ncod(hj)==1)

            r1_new(hj)=u(hj);

            Jo(hj,1:np)=0;
            Jo(hj,hj)=1;

        end
    end

end

function tsfun(x,y)
    l1  =@(c)  2.*c^2-3.*c+1.;
    l2  =@(c) -4.*c^2+4.*c;
    l3  =@(c)  2.*c^2-c;
    dl1 =@(c)  4.*c-3.;
    dl2 =@(c) -8.*c+4.;
    dl3 =@(c)  4.*c-1.;

    phi(1)=l1(x)*l1(y);
    phi(2)=l1(x)*l2(y);
    phi(3)=l1(x)*l3(y);
    phi(4)=l2(x)*l1(y);
```

```
        phi(5)=l2(x)*l2(y);
        phi(6)=l2(x)*l3(y);
        phi(7)=l3(x)*l1(y);
        phi(8)=l3(x)*l2(y);
        phi(9)=l3(x)*l3(y);
        phic(1)=l1(y)*dl1(x);
        phic(2)=l2(y)*dl1(x);
        phic(3)=l3(y)*dl1(x);
        phic(4)=l1(y)*dl2(x);
        phic(5)=l2(y)*dl2(x);
        phic(6)=l3(y)*dl2(x);
        phic(7)=l1(y)*dl3(x);
        phic(8)=l2(y)*dl3(x);
        phic(9)=l3(y)*dl3(x);
        phie(1)=l1(x)*dl1(y);
        phie(2)=l1(x)*dl2(y);
        phie(3)=l1(x)*dl3(y);
        phie(4)=l2(x)*dl1(y);
        phie(5)=l2(x)*dl2(y);
        phie(6)=l2(x)*dl3(y);
        phie(7)=l3(x)*dl1(y);
        phie(8)=l3(x)*dl2(y);
        phie(9)=l3(x)*dl3(y);
    end

    function JoRfind(nell)

       tphx=zeros(9,1); tphy=zeros(9,1);

        w  = [0.27777777777778, 0.444444444444, 0.27777777777778];
        gp = [0.1127016654    , 0.5           , 0.8872983346    ];

        ngl = nop(nell,1:9);

        for j=1:3 %LOOP j

            for k=1:3 %LOOP k

                tsfun(gp(j),gp(k))

                % *** isoparametric transformation
                x1=0;x2=0;y1=0;y2=0;x=0;y=0;
                for n=1:9

                    x=x+xpt(ngl(n))*phi(n);
                    y=y+ypt(ngl(n))*phi(n);

                    x1=x1+xpt(ngl(n))*phic(n);
                    x2=x2+xpt(ngl(n))*phie(n);
                    y1=y1+ypt(ngl(n))*phic(n);
                    y2=y2+ypt(ngl(n))*phie(n);
                end
                dett=x1*y2-x2*y1;
                for i=1:9
                    tphx(i)=(y2*phic(i)-y1*phie(i))/dett;
                    tphy(i)=(x1*phie(i)-x2*phic(i))/dett;
                end
            u_old_gp=0;
            du_old_gpx=0;
            du_old_gpy=0;

            for gpoints=1:9
              u_old_gp=u_old_gp+u_old(ngl(gpoints))*phi(gpoints);
```

```
              du_old_gpx=du_old_gpx+u_old(ngl(gpoints))*tphx(gpoints);
              du_old_gpy=du_old_gpy+u_old(ngl(gpoints))*tphy(gpoints);
            end

                % *** residuals
                for l=1:9
                    r1_new(ngl(l))=r1_new(ngl(l))...
                                   +w(j)*w(k)*dett*el*exp(u_old_gp)*phi(l)...
                                   -w(j)*w(k)*dett*du_old_gpx*tphx(l)...
                                   -w(j)*w(k)*dett*du_old_gpy*tphy(l);
                    for m=1:9
                        Jo(ngl(l),ngl(m)) = Jo(ngl(l),ngl(m)) ...
                                            -w(j)*w(k)*dett*(tphx(l)*tphx(m))...
                                            -w(j)*w(k)*dett*(tphy(l)*tphy(m))...

+w(j)*w(k)*dett*el*exp(u_old_gp)*phi(l)*phi(m);
                    end %inner nodes' residuals remain the same
                end
            end %LOOP k
        end %LOOP j
    end
end
```

## 6.2. Σύζευξη Μεθόδου Πεπερασμένων Στοιχείων (FEM) με Μέθοδο Συνέχειας Ψεύδο - Μήκους Τόξου (Pseudo-Arc length Continuation) για παραμετρική ανάλυση της εξίσωσης Bratu 1D και 2D.

### 6.2.1. Κατάστρωση των εξισώσεων του επαυξημένου συστήματος

Για την παραμετρική ανάλυση, χρησιμοποιήθηκε η μέθοδος Keller (Pseudo Arc-Length Continuation).

Χρησιμοποιήθηκαν οι εξισώσεις 1.4.8-1.4.11 για τον σχηματισμό του επαυξημένου πίνακα και την εύρεση του διανύσματος κατεύθυνσης. Η σχέση που δίνει την παραμετρική παραγώγιση των υπολοίπων που χρειάζεται στον επαυξημένο πίνακα δίνεται από τη σχέση 6.20

$$(\boldsymbol{G_\lambda})^{(\nu)} = \frac{\partial \boldsymbol{R}_i^{(\nu)}}{\partial \boldsymbol{\lambda}} = + \int_0^1 \int_0^1 \exp(u_i^\nu)\, dx\, dy. \tag{6.20}$$

### 6.2.2. Κώδικας Bratu 1D (FEM x Keller's Method)

```
function [XR, po, Jaug, Xdot] = ArcBratu(nez,point,ds)
% Parametric continuation with FEM method (Galerkin residual,
% Second order polynomial basis function) of the Bratu problem in 1D.
% uxx  + lamda*exp(u) = 0 , D
% u = 0 , δD

r1_new=[];Jo=[];nop=[];azpt=[];ph=[];phd=[];

nnz=2*nez+1; %total points

es=10^(-6); % Newton Raphson error
b = zeros(1,point); % % This variable is used to store the
% infinite norm value of the solution.
```

```
%Method_____________________________________________________________________
na = 2*nez+2; % number of independent variables + parameter

x1 = NewtonBratu(nez,0.1);           % Newton Raphson for the first point
x2 = NewtonBratu(nez,0.3);           % Newton Raphson for the second point

x1(end+1) = 0.1; % lamda(1)
x2(end+1) = 0.3; % lamda(2)

s1 = 0;
s2 = s1 + norm(x2-x1);             % Arc length evaluation

Xdot = (x2-x1)./(s2-s1);           % Derivative vector with respect to s

Xold = x2;                         % Preparation for Keller's Method

XR = zeros(na,point);              % Pre allocation of the Matrix(solutions)
XR(:,1) = x1;
XR(:,2) = x2;
b(1,1) = norm(x1(1:end-1),inf);
b(1,2) = norm(x2(1:end-1),inf);
po=zeros(point,nnz-2)';

L=1;
discr();
ngl=zeros(3,1);
phz=zeros(3,1);

for k=3:point

    Xprevious = Xold;
    Xold = Xold + ds*Xdot;
    el=Xold(end);


    calc=1;

    maxloop=200;

    c_old = Xold(1:end-1,1);



    JodcR();

    c_old = Xold;

    for iter=calc:maxloop
        r1=r1_new' ;
        Jaug = [Jo; Xdot.'];
        delta_c = Jaug\(-[r1;dot((Xold-Xprevious),Xdot)-ds]);
        c = c_old+delta_c;
        c_old = c;
        ea=norm(delta_c,2);
        el=c(end);
        c_old = c(1:end-1);
        JodcR();
        %Xdot = Jaug\[zeros(na-1,1);1];
        Jaug = [Jo; Xdot.'];
        c_old = c;
        if ea<=es, break; end
```

```
     end

    XR(:,k) = c_old;
    b(1,k)=norm(c_old(1:end-1),inf);
    Xold=c_old;
    J=Jaug(2:end-2,2:end-2);
    %po(:,1)=0;
    %po(:,2)=0;
    po(:,k)=eig(J);
    Xdot=(XR(:,k)-XR(:,k-1))/norm(XR(:,k)-XR(:,k-1));
    %Xdot=Jaug\[zeros(na-1,1);1];
end

plot(XR(end,:),b,'b:o')

title('Parametric analysis of Bratu 1D problem')
xlabel('lamda')
ylabel('||u||_i_n_f')

function discr()
        %***  define values of parameters

        %***  x-coordinates
        zfirst=0.;
        zlast=L;
        deltaz=(zlast-zfirst)/nez;
        azpt(1)=zfirst;

        for i=2:nnz
              azpt(i)=azpt(i-1)+deltaz/2.;
        end

        %***  nodal numbering ***
        for i=1:nez
            nop(i,1)=1+2*(i-1);
            nop(i,2)= nop(i,1)+1;
            nop(i,3)= nop(i,2)+1;
        end
        %************************
    end

function JodcR () %make the tables
  for i=1:nnz
      r1_new(i)=0.;
            for j=1:nnz+1
                Jo(i,j)=0.;
            end
  end
      for nell=1:nez
          JoRfind(nell);
      end

      for i=1:nnz+1 %implementation of Dirichlet bcs
          Jo(1,i)=0;
          Jo(end,i)=0;
      end
        Jo(1,1) = 1.;
        Jo(end,end-1) = 1.;
        r1_new(1) = c_old(1);
        r1_new(end) = c_old(end);
end
```

```
function tsfun(z)

        q=z;
        ph(1)=2.*q^2-3.*q+1.;
        ph(2)=-4.*q^2+4.*q;
        ph(3)=2.*q^2-q;
        phd(1)=4.*q-3.;
        phd(2)=-8.*q+4.;
        phd(3)=4.*q-1.;

end

function JoRfind(nell)

        w  = [0.27777777777778, 0.444444444444, 0.27777777777778];
        gp = [0.1127016654    , 0.5            , 0.8872983346    ];


        for i=1:3
            ngl(i)=nop(nell,i);

        end


        for j=1:3 %Loop over the qauss points

            tsfun(gp(j));

            z=0.;
            z1=0.;

            for ngp=1:3
                z=z+azpt(ngl(ngp))*ph(ngp);
                z1=z1+azpt(ngl(ngp))*phd(ngp);
            end

            for i=1:3
                phz(i)=phd(i)/z1;

            end


            c_old_gp=0;
            dc_old_gp=0;

            for gpoints=1:3
              c_old_gp=c_old_gp+c_old(ngl(gpoints))*ph(gpoints);
              dc_old_gp=dc_old_gp+c_old(ngl(gpoints))*phz(gpoints);
            end

            for m=1:3

                m1=ngl(m);
                    r1_new(m1)= r1_new(m1)...
                    -w(j)*phz(m)*dc_old_gp*z1...
                    +w(j)*el*exp(c_old_gp)*ph(m)*z1 ;

                    Jo(m1,end)=Jo(m1,end)+ w(j)*exp(c_old_gp)*ph(m)*z1;

                for ni=1:3

                    n1=ngl(ni);
```

```
                    Jo(m1,n1)=Jo(m1,n1) ...
                     -w(j)*z1*phz(m)*phz(ni)            ...
                     +w(j)*el*ph(m)*ph(ni)*exp(c_old_gp)*z1;


                end   % loop for n


            end       % loop for


        end % end of the  loop over the gauss points

end %JoRfind

end
```

### 6.2.3. Κώδικας Bratu 2D (FEM x Keller's Method)

```
function [xpt, ypt, u, XR, po]=ArcBratu2D(nex,ney,point,ds)
% Parametric continuation with FEM method (Galerkin residual,
% Second order polynomial basis function) of the Bratu problem in 2D.
% uxx + uyy + lamda*exp(u) = 0 , D
% u = 0 , δD

nnx = 2*nex+1; % node number in x-axis.
nny = 2*ney+1; % node number in y-axis.
ne  = nex*ney; % total element number.
np  = nnx*nny; % total node number.
u = []; r1_new=[];Jo=[]; % Declaration of the variable used by the sub-routines
xorigin = []; yorigin = []; xlast = []; ylast = []; % Demanded by Matlab
deltax = []; deltay = []; xpt = []; ypt = [];
nop = [];
ncod = [];
w = []; gp =[];
phi = []; phic = []; phie = [];
L=1.;  % L,w defines the domain of the problem [0,1]x[0,1]
w=1.;

es=10^-6; % Newton-Raphson error

% prepare for essential boundary conditions
ncod  =  zeros(np,1); % variable used for the implementation of the bc
%bc   = zeros(np,1);

xydiscr() %function that is used for the discretization of the domain
nodnumb() %function that numbers the node
xycoord() % *** (x,y) coordinates of each node

%ncod(1:nny) = 1; there is no need to imply new bc, because these nodes are
%bc(1:nny)   = 0; treated like inner nodes, as shown in residuals

ncod(1:1:nny)           =  1;
%bc(1:1:nny)            =  0;

ncod(nny:nny:np)        =  1;
%bc(nny:nny:np)         =  0;

ncod(1:nny:np-nny+1)    =  1;
%bc(1:nny:np-nny+1)     =  0;
```

```
ncod(np-nny+1:1:np)     =  1;
%bc(np-nny+1:1:np)      =  0;
ncod(end+1)=0;
%Method_____________________________________________________________________

b = zeros(1,point); % This variable is used to store the
% infinite norm value of the solution.

na = np+1; % number of independent variables + parameter


[xpt, ypt, x1] = Bratu2D(nex,ney,0.01); % Newton Raphson for the first point
[xpt, ypt, x2] = Bratu2D(nex,ney,0.06); % Newton Raphson for the second point

x1(end+1) = 0.01;
x2(end+1) = 0.06;

s1 = 0;
s2 = s1 + norm(x2-x1);              % Arc length evaluation

Xdot = (x2-x1)./(s2-s1);            % Derivative vector with respect to s

Xold = x2;                          % Preparation for Keller's Method

XR = zeros(na,point);               % Pre allocation of the Matrix(solutions)
XR(:,1) = x1;
XR(:,2) = x2;
b(1,1) = norm(x1(1:end-1),inf);
b(1,2) = norm(x2(1:end-1),inf);
po=zeros(point,np-2)';

for g=3:point
    %disp(g)
    Xprevious = Xold;
    Xold = Xold + ds*Xdot;
    el=Xold(end);

    calc=1;
    L=1.;  % the new parameters
    w=1.;

   maxloop=200;


   u_old = Xold(1:end-1,1);


   JodcR();
   u_old = Xold;
    for iter=calc:maxloop
        r1=r1_new' ;
        Jaug = [Jo; Xdot.'];
        delta_u = Jaug\(-[r1;dot((Xold-Xprevious),Xdot)-ds]);
        u = u_old+delta_u;
        u_old = u;
        ea=norm(delta_u,2)
        el=u(end);
        u_old = u(1:end-1);
        JodcR();
        %Xdot = Jaug\[zeros(na-1,1);1];
        Jaug = [Jo; Xdot.'];
        u_old = u;
```

```
        if ea<=es, break; end

     end

    XR(:,g) = u_old;
    b(1,g)=norm(u_old(1:end-1),inf);
    Xold=u_old;
    J=Jaug(2:end-2,2:end-2);% the original Jacobian of the system.
    %po(:,1)=0;
    %po(:,2)=0;
    po(:,g)=eig(J);%evaluates the eigenvalues. Beware of the bcs.
    Xdot=(XR(:,g)-XR(:,g-1))/norm(XR(:,g)-XR(:,g-1));
end
plot(XR(end,:),b,'b:o')
title('Parametric analysis of Bratu 2D problem')
xlabel('lamda')
ylabel('||u||_i_n_f')

    function xydiscr()
        xorigin = 0.;
        yorigin = 0.;
        xlast   = L; %the new length
        ylast   = w; %the new width
        deltax = (xlast-xorigin)/nex;
        deltay = (ylast-yorigin)/ney;
    end

    function nodnumb()
    % *** nodal numbering
      nel=0;
        for i=1:nex
            for j=1:ney
                nel=nel+1;
                for k=1:3
                    l=3*k-2;
                    nop(nel,l)=nny*(2*i+k-3)+2*j-1;
                    nop(nel,l+1)=nop(nel,l)+1;
                    nop(nel,l+2)=nop(nel,l)+2;
                end
            end
        end
    end

    function xycoord()
    % *** (x,y) coordinates of each node
        xpt=zeros(np,1);
        ypt=zeros(np,1);
        %xpt(1)=xorigin;
        %ypt(1)=yorigin;
        xpt(1)=0; % xorigin is 0.
        ypt(1)=0; % yorigin is 0.
        for i=1:nnx
            nnode=(i-1)*nny+1;
            xpt(nnode)=xpt(1)+(i-1)*deltax/2.;
            ypt(nnode)=ypt(1);
            for j=2:nny
                xpt(nnode+j-1)=xpt(nnode);
                ypt(nnode+j-1)=ypt(nnode)+(j-1)*deltay/2.;
            end
        end
    end

    function JodcR () %make the tables
```

```
        for nj=1:np % prepare the residual array and the Jacobian Matrix
            r1_new(nj)=0;
            for j=1:np+1
                Jo(nj,j)=0;
            end
        end

        for nell=1:ne
            JoRfind(nell);
        end

        for hj=1:np+1 %implementation of the Dirichlet bc

             if(ncod(hj)==1)

                r1_new(hj)=u_old(hj);

                Jo(hj,1:np)=0;
                Jo(hj,hj)=1;

            end
        end

    end

    function tsfun(x,y)
        l1  =@(c)  2.*c^2-3.*c+1.;
        l2  =@(c) -4.*c^2+4.*c;
        l3  =@(c)  2.*c^2-c;
        dl1 =@(c)  4.*c-3.;
        dl2 =@(c) -8.*c+4.;
        dl3 =@(c)  4.*c-1.;

        phi(1)=l1(x)*l1(y);
        phi(2)=l1(x)*l2(y);
        phi(3)=l1(x)*l3(y);
        phi(4)=l2(x)*l1(y);
        phi(5)=l2(x)*l2(y);
        phi(6)=l2(x)*l3(y);
        phi(7)=l3(x)*l1(y);
        phi(8)=l3(x)*l2(y);
        phi(9)=l3(x)*l3(y);
        phic(1)=l1(y)*dl1(x);
        phic(2)=l2(y)*dl1(x);
        phic(3)=l3(y)*dl1(x);
        phic(4)=l1(y)*dl2(x);
        phic(5)=l2(y)*dl2(x);
        phic(6)=l3(y)*dl2(x);
        phic(7)=l1(y)*dl3(x);
        phic(8)=l2(y)*dl3(x);
        phic(9)=l3(y)*dl3(x);
        phie(1)=l1(x)*dl1(y);
        phie(2)=l1(x)*dl2(y);
        phie(3)=l1(x)*dl3(y);
        phie(4)=l2(x)*dl1(y);
        phie(5)=l2(x)*dl2(y);
        phie(6)=l2(x)*dl3(y);
        phie(7)=l3(x)*dl1(y);
        phie(8)=l3(x)*dl2(y);
        phie(9)=l3(x)*dl3(y);
    end
```

```
function JoRfind(nell)

   tphx=zeros(9,1); tphy=zeros(9,1);

    w  = [0.27777777777778, 0.444444444444, 0.27777777777778];
    gp = [0.1127016654    , 0.5           , 0.8872983346    ];

    ngl = nop(nell,1:9);

    for j=1:3 %LOOP j

        for k=1:3 %LOOP k

            tsfun(gp(j),gp(k))

            % *** isoparametric transformation
            x1=0;x2=0;y1=0;y2=0;x=0;y=0;
            for n=1:9

                x=x+xpt(ngl(n))*phi(n);
                y=y+ypt(ngl(n))*phi(n);

                x1=x1+xpt(ngl(n))*phic(n);
                x2=x2+xpt(ngl(n))*phie(n);
                y1=y1+ypt(ngl(n))*phic(n);
                y2=y2+ypt(ngl(n))*phie(n);
            end
            dett=x1*y2-x2*y1;
            for i=1:9
                tphx(i)=(y2*phic(i)-y1*phie(i))/dett;
                tphy(i)=(x1*phie(i)-x2*phic(i))/dett;
            end
        u_old_gp=0;
        du_old_gpx=0;
        du_old_gpy=0;

        for gpoints=1:9
          u_old_gp=u_old_gp+u_old(ngl(gpoints))*phi(gpoints);
          du_old_gpx=du_old_gpx+u_old(ngl(gpoints))*tphx(gpoints);
          du_old_gpy=du_old_gpy+u_old(ngl(gpoints))*tphy(gpoints);
        end

            % *** residuals
            for l=1:9
                r1_new(ngl(l))=r1_new(ngl(l))...
                              +w(j)*w(k)*dett*el*exp(u_old_gp)*phi(l)...
                              -w(j)*w(k)*dett*du_old_gpx*tphx(l)...
                              -w(j)*w(k)*dett*du_old_gpy*tphy(l);

                Jo(ngl(l),end)=Jo(ngl(l),end)+w(j)*w(k)*dett*exp(u_old_gp)*phi(l);



                for m=1:9
                    Jo(ngl(l),ngl(m)) = Jo(ngl(l),ngl(m)) ...
                                        -w(j)*w(k)*dett*(tphx(l)*tphx(m))...
                                        -w(j)*w(k)*dett*(tphy(l)*tphy(m))...

                                      +w(j)*w(k)*dett*el*exp(u_old_gp)*phi(l)*phi(m);
                end %inner nodes' residuals remain the same
            end
        end %LOOP k
    end %LOOP j
```

```
    end
end
```

## 6.3. Κώδικας Γκαουσιανών Διεργασιών

```
%the same code is used for both the 1D and 2D Bratu Problem. In the current mode, it
evaluates the parameters for the 2D problem.(For the 1D just use the variables
u,ux,uxx,lamda

u=u(~isnan(u)); %only necessary if there is NaN in the dataset

ut=ut(~isnan(ut));

ux=ux(~isnan(ux));

uxx=uxx(~isnan(uxx));

lamda=lamda(~isnan(lamda));

%k=600; training_data_1D

k=1:1500;%training_data_2D

%k=randperm(length(u),10000);

u1=u(k);
ut1=ut(k);
uxy1=uxy(k);
uyx1=uxy(k);
uxx1=uxx(k);
uyy1=uyy(k);
ux1=ux(k);
uy1=uy(k);

lamda1=lamda(k);
mat_11=[u1 ux1 uy1 uxx1 uyy1 uyx1 uxy1 lamda1];

mdlk = fitrgp(mat_11,ut1,'KernelFunction','ardsquaredexponential');

sigmaK = mdlk.KernelInformation.KernelParameters(1:end-1,1)
el_K = log(sigmaK)
```

## 6.4. Κώδικας Εκπαίδευσης Νευρωνικών Δικτύων

Κοινός κώδικας και για το 1D και το 2D. Ο αρχικός κώδικας παράγεται από το Matlab και οι απαραίτητες αλλαγές προστίθενται σε αυτόν.

```
% Solve an Input-Output Fitting problem with a Neural Network
% Script generated by Neural Fitting app
% Created 08-Mar-2023 22:22:54
%
% This script assumes these variables are defined:
%
% for 1D  mat1 [u ux lamda]      - input data.
```

```
% for 2D  mat1 [u uxx uyy lamda] - input data.
% ut - target data.


x = [u uxx uyy lamda]';
t = ut';

% Choose a Training Function
% For a list of all training functions type: help nntrain
% 'trainlm' is usually fastest.
% 'trainbr' takes longer but may be better for challenging problems.
% 'trainscg' uses less memory. Suitable in low memory situations.
trainFcn = 'trainlm';  % Levenberg-Marquardt backpropagation.

% Create a Fitting Network
H1 = 10;
H2 = 10;
net = fitnet([H1 H2],trainFcn);

% Setup Division of Data for Training, Validation, Testing
%net.divideFcn = 'dividerand';
%net.divideMode = 'sample';
net.divideParam.trainRatio = 70/100;
net.divideParam.valRatio = 15/100;
net.divideParam.testRatio = 15/100;

% Choose Input and Output Pre/Post-Processing Functions
net.input.processFcns = {'removeconstantrows','mapminmax'};
net.output.processFcns = {'removeconstantrows','mapminmax'};

%Choose a Perfomance Function
net.performFcn = 'mse'; %the cost function is the mean squared error.


% Choose Plot Functions
net.plotFcns = {'plotperform','plotregression'};
net.layers{1}.transferFcn = 'tansig';
net.layers{2}.transferFcn = 'purelin';


%Training parameters.

%net.trainParam.min_grad=1e-5; %minimum gradient. Under this value,the training stops.
%net.trainParam.show=10; %Number of epochs after the plots are performed.
%net.trainParam.lr=1; %Learning rate,
%net.trainParam.lr_inc=1.05; % Increasing learning rate value.
%net.trainParam.lr_dec=0.7; % Decreasing learning rate value.
net.trainParam.epochs=10000; % Number of epochs.
%net.trainParam.goal=0.0; % MSE goal value.
%net.trainParam.max_fail=6; % if the MSE of validation set does not decrease after 6
epochs, the training stops.

% Train the Network
[net,tr] = train(net,x,t);

% Test the Network
y = net(x);
e = gsubtract(t,y);
performance = perform(net,t,y);

% View the Network
%view(net)
```

```
% Plots
% Uncomment these lines to enable various plots.
%figure, plotperform(tr)
%figure, plottrainstate(tr)
%figure, ploterrhist(e)
%figure, plotregression(t,y)
%figure, plotfit(net,x,t)
```

## 6.5 Κώδικας Μελέτης Δυναμικής Εξέλιξης Φαινομένου

### 6.5.1. Μονοδιάστατο Πρόβλημα (1D)

```
%This script evaluates the dynamic response of a system governed by the time-
%dependent Bratu equation (ut=uxx + λ*exp(u), 0<x<1 && u(0)=u(1)=0), using
% classical computational methods (finite differences +ode15) and neural
% network approach. Finally, it plots the two solution and compares them
% with the mean squared error (mse) value and the mean absolute relative
% error

clear;clc;
hold on
xpt=51; % the number of nodes of the domain
el=3; %value of the parameter lamda
x=linspace(0,1,xpt); %discretization of the domain

dx=x(2)-x(1);

ts=[0 0.01 0.1 0.2 1]; %times when the values of the function are evaluated


% Three different initial methods to check how the network-based method responses

%u0 =0.5*x.*(x-1).*(x+1).*(x+3).*(x+2).*(x-2); %lamda=1.5

%u0 =exp(-1/2*((x-0.5)/0.15).^2)/(0.15*sqrt(2*pi)); %lamda=2

u0=zeros(xpt,1);%lamda=3


%%implementation of the bcs
u0(1)=0;
u0(xpt)=0;
plot(x,u0)
uxx=zeros(xpt,1);

for i=2:xpt-1
    uxx(i)= (u0(i+1) +u0(i-1)-2*u0(i))/dx^2; %evaluation of the initial second order
derivative
end

f=@(t,u) UtNet(u',dx,el,xpt); % NN Solution
[t,u] = ode15s(f,ts,u0);
plot(x,u(2:end,:)','x')


g=@(t,y) timeBratu(y,xpt,el); % finite differences solution
[t1,w] = ode15s(g,ts,u0);
plot(x,w(2:end,:)','o','LineStyle','-')
legend
xlabel('x')
ylabel('u')
title('Dynamic evolution of time-depedent Bratu 1D Problem ')
```

```
% Perfomance function (we do not consider the bc points)
mse=sum(sqrt((u(:,2:end-1)'-w(:,2:end-1)').^2/(xpt-2))); % root mean square error
mape=sum(abs((u(:,2:end-1)'-w(:,2:end-1)')./u(:,2:end-1)'))*100/(xpt-2);
%  mean absolute percentage error

function dydt = UtNet(u,dx,el,xpt)
load netBratu1D.mat
uxx(1)=0;
uxx(xpt)=0;
    for m=2:(xpt-1)
    uxx(m) = (u(m+1)+u(m-1)-2*u(m))/(dx.^2);
    end

lamda=el*ones(1,xpt-2);
trsp=[u(2:xpt-1);uxx(2:xpt-1);lamda];
dydt(1)=0;
dydt(xpt)=0;
dydt(2:xpt-1)= net(trsp);
dydt=dydt';
end
function dudt= timeBratu(u,xpt,l)

x=linspace(0,1,xpt);

dx=x(2)-x(1);

dudt(1)=0;
dudt(xpt)=0;

for i=2:xpt-1
    dudt(i)= (u(i+1) +u(i-1)-2*u(i))/dx^2 + l*exp(u(i));
end

dudt=dudt';
end
```

### 6.5.2. Δισδιάστατο Πρόβλημα (2D)

```
%This script evaluates the dynamic response of a system governed by the time-
%dependent Bratu 2D equation (ut=uxx + uyy + λ*exp(u), 0<x<1, 0<y<1 &&
% u(x,0)=u(x,1)=u(0,y)=u(1,y)=0), using classical computational methods (finite
differences
%+ode15) and a  neural network approach. Finally, it plots the
% two solution and compares them with the root mean squared error (rmse)
% value and the mean absolute relative error

clear;clc;
hold on
xpt=21; % the number of nodes of x points
ypt=21; % the number of nodes of y points

el=4; %value of the parameter lamda
x=linspace(0,1,xpt); %discretization of the domain
y=linspace(0,1,ypt); %discretization of the domain

dx=x(2)-x(1);
dy=y(2)-y(1);

np=xpt*ypt;
nny=ypt;
[X, Y] = meshgrid(x,y);
```

```
X=reshape(X,[],1);
Y=reshape(Y,[],1);


ts=[0 0.1 1 ];  %times when the values of the function are evaluated


% 2 different initial methods to check how the network-based method responses
u0=cos(pi*X).*sin(pi*Y);


u0(1:1:nny)             =  0;
u0(nny:nny:np)          =  0;
u0(1:nny:np-nny+1)      =  0;
u0((np-nny+1):1:np)     =  0;


f=@(t,u) UtNet2D(u,xpt,ypt,el); % NN Solution
[t,u] = ode15s(f,ts,u0);



g=@(t,u) tB2D(u,xpt,ypt,el); % finite differences solution
[t1,w] = ode15s(g,ts,u0);


% Perfomance function (we do not consider the bc points)
a=1:np;
b=1:1:nny;
c=nny:nny:np;
d=1:nny:np-nny+1;
e=(np-nny+1):1:np;
tot=[b c d e];
a(tot)=[];

rmse=sum(sqrt((u(:,a)'-w(:,a)').^2/length(a))); % root mean square error
mape=sum(abs((u(:,a)'-w(:,a)')./u(:,a)'))*100/length(a);
%  mean absolute percentage error


[xi, yi] =
meshgrid(linspace(min(X),max(X),length(X)),linspace(min(Y),max(Y),length(Y)));

for i=2:3


zi = griddata(X,Y,u(i,:)',xi,yi);
wi = griddata(X,Y,w(i,:)',xi,yi);

figure(i) % creating the figures

tiledlayout(2,3);


if max(max(zi))>max(max(wi))
    maxi=max(max(zi));
else
    maxi=max(max(wi));
end

if min(min(zi))<min(min(wi))
    mini=min(min(zi));
else
```

```
    mini=min(min(wi));
end

% Tile 1
nexttile
surf(xi,yi,zi)
shading interp
caxis([mini maxi])
xlabel('x')
ylabel('y')
zlabel('u NN')

% Tile 2
nexttile
contour(xi,yi,zi)
caxis([mini maxi])
xlabel('x')
ylabel('y')
title(['NN Solution at t=',num2str(ts(i)),' for the Bratu 2D Problem'])


% Tile 3
nexttile
surf(xi,yi,zi)
caxis([mini maxi])
shading interp
xlabel('x')
ylabel('y')


% Tile 4
nexttile
surf(xi,yi,wi)
shading interp
caxis([mini maxi])
xlabel('x')
ylabel('y')
zlabel('u FD')


% Tile 5
nexttile
contour(xi,yi,wi)
caxis([mini maxi])
xlabel('x')
ylabel('y')
title(['FD Solution at t=',num2str(ts(i)),' for the Bratu 2D Problem'])

% Tile 6
nexttile
surf(xi,yi,wi)
caxis([mini maxi])
shading interp
xlabel('x')
ylabel('y')


cb = colorbar();
cb.Layout.Tile = 'east';


figure(3*i)
surf(xi,yi,zi-wi)
```

```
caxis([mini maxi])
shading interp
xlabel('x')
ylabel('y')
title(['The difference between NN and FD solution at t=',num2str(ts(i)),' for the
Bratu 2D Problem'])

end
function dudt= tB2D(u,xpt,ypt,l)

x=linspace(0,1,xpt);
y=linspace(0,1,ypt);

dx=x(2)-x(1);
dy=y(2)-y(1);

dudt(1)=0;
dudt(xpt*ypt)=0;

np=xpt*ypt;

nny=ypt;
 for i = (nny+1):(np-nny)

dudt(i)= (u(i+nny)-2*u(i)+u(i-nny))/dx^2+(u(i+1)-2*u(i)+u(i-1))/dy^2+l*exp(u(i));
 end

dudt(1:1:nny)             =  0;
dudt(nny:nny:np)          =  0;
dudt(1:nny:np-nny+1)      =  0;
dudt((np-nny+1):1:np)     =  0;

dudt=dudt';
end


function dudt = UtNet2D(u, xpt, ypt, el)

load NetBratu2D

nnbest = net;

x=linspace(0,1,xpt);
dx=x(2)-x(1);

y=linspace(0,1,ypt);
dy=y(2)-y(1);

np=xpt*ypt;
nny=ypt;
dudt=zeros(1,np);
X=zeros(4,np);

X(1,:)= u;
X(4,:)= el;


nny=ypt;

 i = (nny+1):(np-nny);

    X(2,i) = (u(i+nny)-2*u(i)+u(i-nny))/dx^2;
    X(3,i) = (u(i+1)-2*u(i)+u(i-1))/dy^2;
```

```
dudt(1:end)=nnbest(X);
dudt(1:1:nny)              =  0;
dudt(nny:nny:np)           =  0;
dudt(1:nny:np-nny+1)       =  0;
dudt((np-nny+1):1:np)      =  0;



dudt=dudt';

end
```

## 6.6. Κώδικας μεθόδου GMRES

### 6.6.1. Βασικός κώδικας επίλυσης (1D)

```
clear;clc;

load netBratu1D.mat %loading the neural network

global x Res fn0  xpt  x1 x2 ds xso nnbest

nnbest=net;
o=54; %number of points/solutions in the parametric analysis

b=zeros(1,o); %stores the maximum value of the function for the
% lamda value of all solutions

a=17; % number of nodes

po=[];%  solution matrix

ds=0.4;% defined by the user. The Euclidean distance
% between the evaluated solutions.

xpt=linspace(0,1,a); %discretization of the domain

load x11d %load the first solution (λ=0.1)
x1=x11d;
b(1)=max(x1(1:end-1));

load x21d %load the second solution (λ=0.3)
x2=x21d;
b(2)=max(x2(1:end-1));

po(:,1) = x1; % create the solution matrix
po(:,2) = x2;

s1 = 0;
s2 = s1 + norm(x2-x1);             % Arc length evaluation
xso = (x2-x1)./(s2-s1);

dx=xpt(2)-xpt(1);


% Start Newton Raphson iterations
params=[1e-3 length(xpt) 3]; %GMRES tolerance, dimension of Krylov Subspace,
```

```
%orthonormalization technique
tol = 1e-6; % the Newton iteration tolerance
k=2;

% Implement a modified arc length method. Change the value ds according
% to convergence

flag1=0;

np=length(x);

for i=3:o
    disp(i)

    if flag1==0
    k=k+1;
    x2=po(:,k-1);

    else
        flag1=0;
    end

    x=x2+xso*ds;
    Res=resi_par(x);



    ametr=0;

    while 1

        kmax=np;
        %Res=zeros(np,1);
        fn0=resi_par(x);
        Res=fn0;

        % call GMRES for the solution of linearized problem
        [dx,error,total_iters] = gmres (zeros(np,1), -Res, @atv, params );
        err=max(abs(dx));
        ametr=ametr+1;

        Newton_error = num2str(err);
        disp(Newton_error); %the error of the Newton-Raphson iteration

        x=x+dx; %update the solution

        if err <= tol
            break
        end


        if ametr>15
            flag1=1;
            break;
        end



Resi=zeros(np,1);

    end

    if flag1==0
```

```
        po(:,k) = x;
        b(k)=max(po(1:end-1,k));
        xso=(po(:,k)-po(:,k-1))/norm(po(:,k)-po(:,k-1));
        fn0=resi_par(x);

        %ds=1.1*ds; %Used if the acceleration of the method is wanted

    else
        ds=ds/2; % Reduce the distance to help find solution near singular points
    end

end

plot(po(end,:),b,'rx','MarkerSize',8) % plots the parametric analysis

legend('Newton GMRES')
xlabel('lamda')
ylabel('|u|_i_n_f')

%save sol2
%save solupback246 'x'

hold on

y=ArcBratu(8,o,ds); %the FEM solution to use it for visual comparison.

hold off
```

Η resi_par σχηματίζει τα υπόλοιπα (residuals) που χρησιμοποιούνται στην Newton-GMRES για το 1D πρόβλημα.

```
function y=resi_par(w)
global x2 ds xso x nnbest xpt net1 net2

%x0=w;
y=zeros(length(w),1);
np=length(xpt);
dx=xpt(2)-xpt(1);


X(1,:)=w(1:end-1)';

i=2:np-1;

%y(i) = (w(i+1)+w(i-1)-2*w(i))/dx^2 +w(end)*exp(w(i));

X(2,i)=(w(i+1)+w(i-1)-2*w(i))/dx^2;

X(2,1)=0; X(2,end)=0;


X(3,:)=w(end);

y(1:end-1)=nnbest(X); y(1)=w(1); y(end-1)=w(end-1);
%y(1:end-1)=(net1(X)+net2(X))/2; y(1)=w(1); y(end-1)=w(end-1);

%y(end)=(w-x2)'*(w-x2)-ds^2;
y(end)=xso'*(w-x2)-ds;
```

### 6.6.2. Βασικός κώδικας επίλυσης (2D)

```
clear;clc;

tic

load NetBratu2D.mat
global x Res fn0  xpt  x1 x2 ds xso ypt dy dxd nnbest ay

nnbest=net;
o=96;

b=zeros(1,o);

ax=21;

ay=21;

po=[];

ds=0.5;

xdiscr=linspace(0,1,ax);

ydiscr=linspace(0,1,ay);

dxd = xdiscr(2) - xdiscr(1);

dy = ydiscr(2) - ydiscr(1);

[X Y]=meshgrid(xdiscr, ydiscr);

xpt=reshape(X,[],1);

ypt=reshape(Y,[],1);


load x12d %load the first solution (λ=1.3)
x1=x12d;
b(1)=max(x1(1:end-1));P(1)=x1(end);

load x22d %load the second solution (λ=1.8)
x2=x22d;
b(2)=max(x2(1:end-1));P(2)=x2(end);



po(:,1) = x1;
po(:,2) = x2;
s1 = 0;
s2 = s1 + norm(x2-x1);              % Arc length evaluation
xso = (x2-x1)./(s2-s1);


% Start Newton Raphson iterations
params=[1e-3 length(xpt) 3];
```

```
tol = 1e-6; % the Newton iteration tolerance
k=2
flag1=0;




for i=3:o
    if flag1==0
    k=k+1;
    x2=po(:,k-1);
    else
        flag1=0;
    end
x=x2+xso*ds;
Res=resi_par_2D(x);
np=length(x);

ametr=0;
while 1
    kmax=np;
    %Res=zeros(np,1);
    fn0=resi_par_2D(x);
    Res=fn0;


    % call GMRES for the solution of linearized problem
    [dx,error,total_iters] = gmres (zeros(np,1), -Res, @atv2D, params );
    err=max(abs(dx));
    ametr=ametr+1;

    Newton_error = num2str(err)

    x=x+dx;

    if err <= tol
        break
    end

    end
    if ametr>60
        flag1=1;
        break
    end
    Resi=zeros(np,1);
    %Resi=resi_par(x)-x(1:np); params(1)=min(1e-2,0.9*norm(Resi)^2/norm(Res)^2);
    %Res=Resi;


if flag1==0
po(:,k) = x;
b(k)=max(po(1:end-1,k)); P(k)=x(end);
xso=(po(:,k)-po(:,k-1))/norm(po(:,k)-po(:,k-1));
fn0=resi_par_2D(x);

else
ds=ds/2;
end
end
plot(po(end,:),b,'rx')
toc
legend('Newton GMRES')
xlabel('lamda')
```

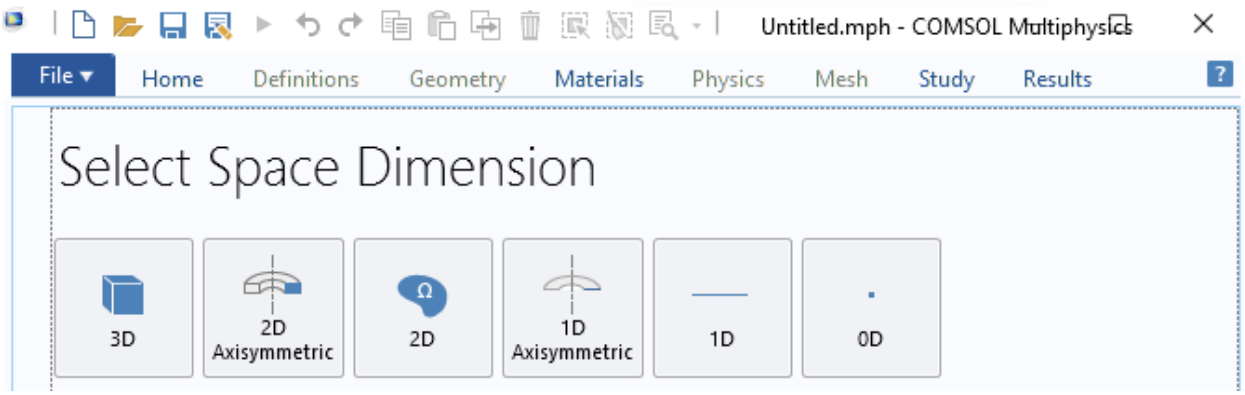
```
ylabel('|u|_i_n_f')
%save sol2
%save solupback246 'x'
hold on
ArcBratu2D((ax-1)/2,(ay-1)/2,o-6,ds);
hold off
```

Η resi_par_2D σχηματίζει τα υπόλοιπα (residuals) που χρησιμοποιούνται στην Newton-GMRES για το 2D πρόβλημα.

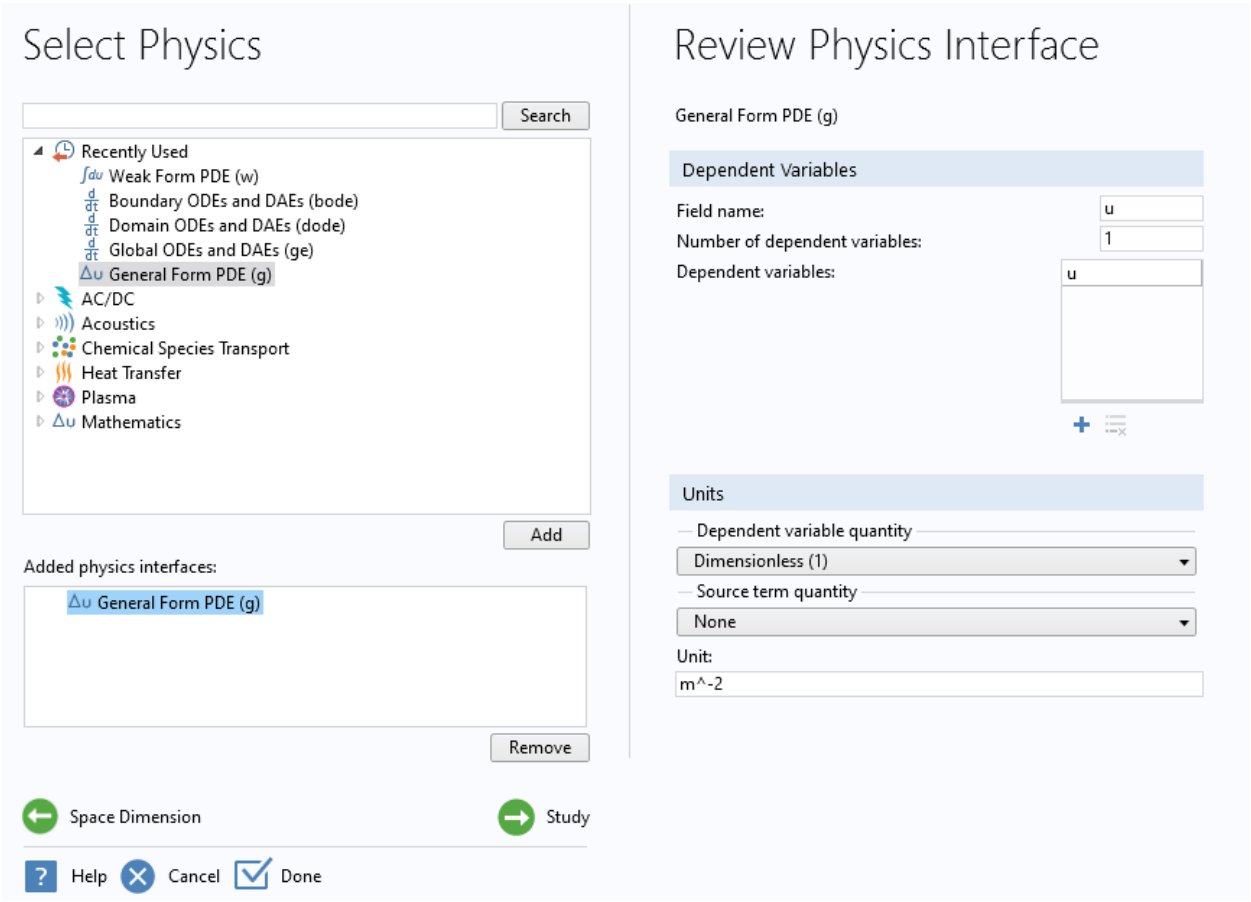
```
function y=resi_par_2D(w)
global x2 ds xso x  xpt dy dxd ay nnbest
%x0=w;

y=zeros(length(w),1);


np=length(xpt);
X=zeros(4,length(xpt));
X(1,:)= w(1:end-1)';
X(4,:)= w(end);

nny=ay;
 i = (nny+1):(np-nny);

   % y(i)= (w(i+nny)-2*w(i)+w(i-nny))/dxd^2+(w(i+1)-2*w(i)+w(i-
1))/dy^2+w(end)*exp(w(i));

    X(2,i) = (w(i+nny)-2*w(i)+w(i-nny))/dxd^2;
    X(3,i) = (w(i+1)-2*w(i)+w(i-1))/dy^2;


y(1:end-1)=nnbest(X);
y(1:1:nny)             =  w(1:1:nny)     ;
y(nny:nny:np)          =  w(nny:nny:np)  ;
y(1:nny:np-nny+1)      =  w(1:nny:np-nny+1);
y((np-nny+1):1:np)     =  w((np-nny+1):1:np);


y(end)=xso'*(w-x2)-ds;
```

## 6.7. Λογισμικό COMSOL Multiphysics Edition 5.2.

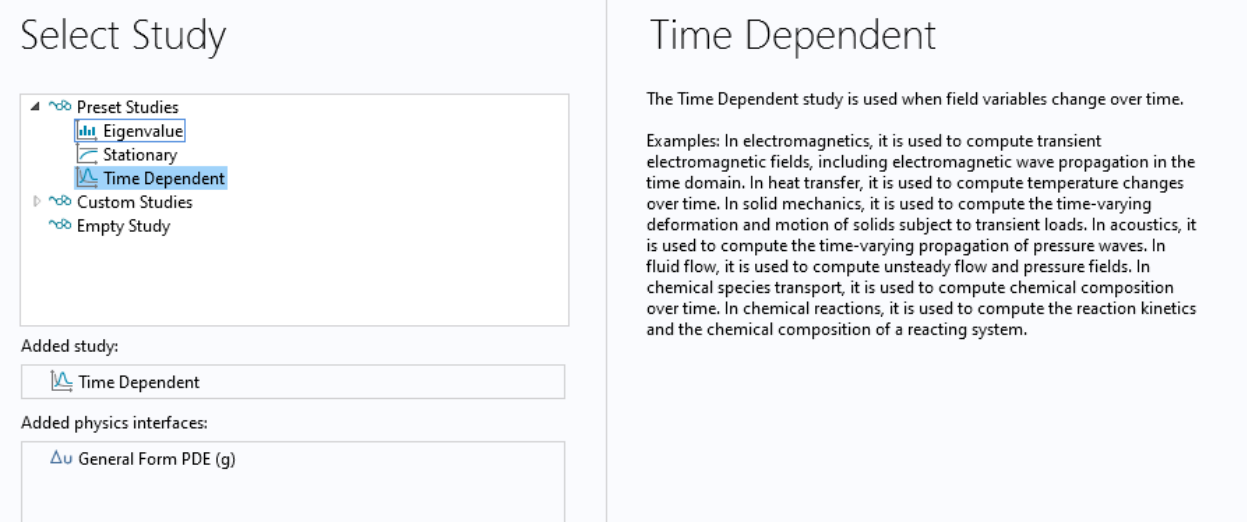
Στην παρακάτω εικόνα συνοψίζεται η δημιουργία μοντέλου μέσω του model wizard του COMSOL.




Αρχικά, επιλέγεται ο χώρος των διαστάσεων. Έτσι, στην περίπτωση της μιας διάστασης επιλέγεται η περίπτωση 1D, ενώ στις 2 διαστάσεις επιλέγεται το εικονίδιο 2D (αν και η λύση είναι συμμετρική ως προς την ευθεία x=0.5, θεωρείται ότι δεν είναι γνωστό).




Έπειτα, επιλέγεται η «φυσική» που ακολουθεί το σύστημα που επιθυμείται η μελέτη του. Έτσι, σε αυτή την περίπτωση επιλέγεται η Γενική Μορφή ΜΔΕ (General From PDE),αφού πρόκειται για την επίλυση Μερικής Διαφορικής Εξίσωσης.

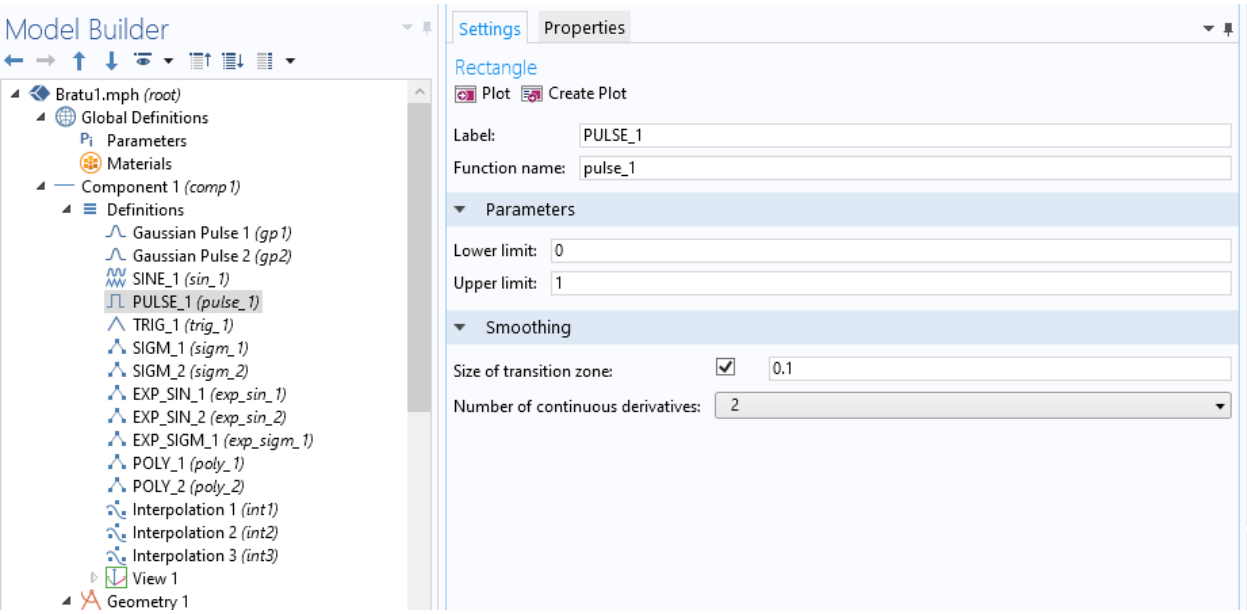



Τέλος, επιλέγεται αν το πρόβλημα είναι πρόβλημα ιδιοτιμών (ανάλυση ευστάθειας, Eigenvalue) ή εξαρτάται από το χρόνο (Time Dependent) ή χρησιμοποιείται για την εύρεση της μόνιμης κατάστασης (stationary). Σε αυτή την εργασία, το ενδιαφέρον βρίσκεται στο χρονικά μεταβαλλόμενο πρόβλημα.

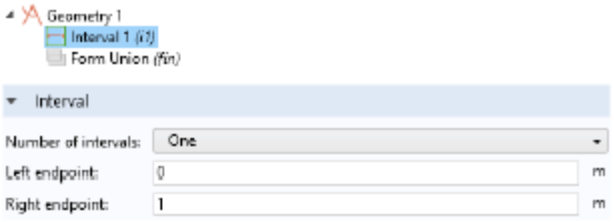
Ακολουθεί η περιγραφή της συλλογής δεδομένων από τις προσομοιώσεις για το πρόβλημα 1D. Η διαδικασία είναι ακριβώς ίδια και για το 2D πρόβλημα.

① Ορισμός Αρχικών συνθηκών

Στο πεδίο ‘Definitions’ εισάγονται οι εξισώσεις που περιγράφουν τις αρχικές συνθήκες που χρησιμοποιούνται .

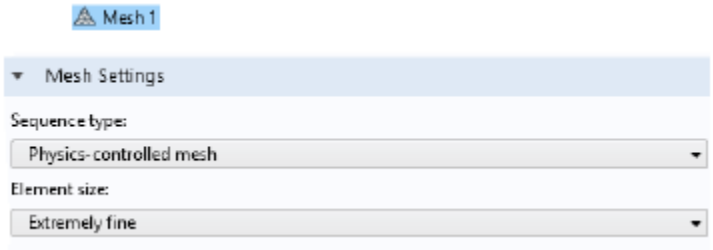



② Εισαγωγή διαστήματος επίλυσης




Πρόκειται για τα άκρα γραμμής (1D) ή για τετράγωνο ακμής α=1 (2D).

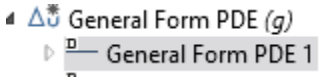
③. Σχηματισμός πλέγματος (mesh)

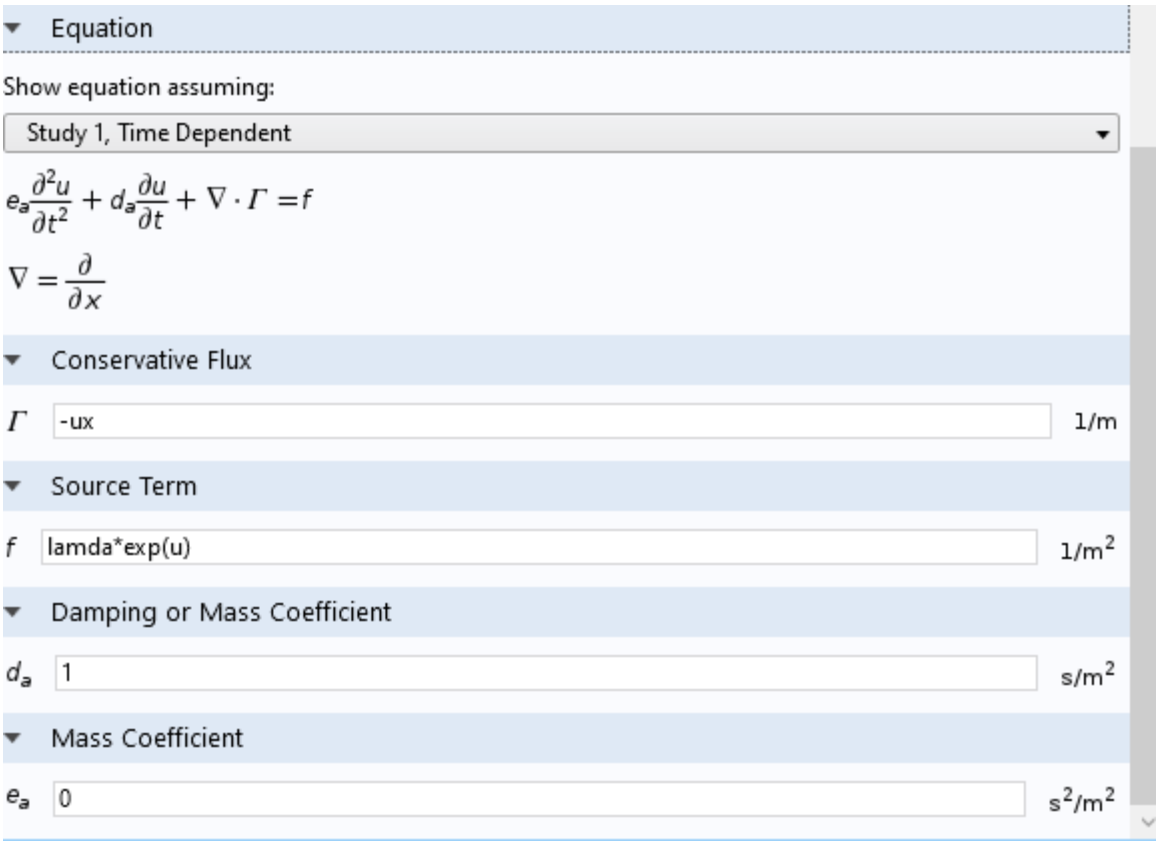



Όσο πιο πυκνό το πλέγμα, τόσο περισσότερα σημεία χρησιμοποιούνται, γεγονός που αυξάνει την ακρίβεια αλλά και το υπολογιστικό κόστος. Επειδή η εξίσωση είναι σχετικά απλή, ένα κανονικό πλέγμα (16 σημεία) επαρκεί για την λήψη των απαραίτητων δεδομένων.

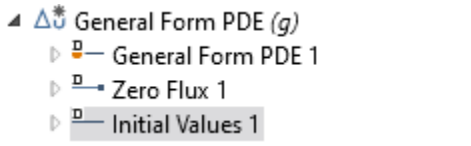
④. Εισαγωγή Εξίσωσης

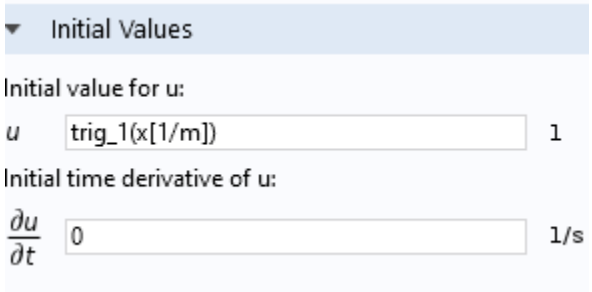
Με βάση την εξίσωση που υποθέτει το λογισμικό, σχηματίζεται η εξίσωση Bratu.





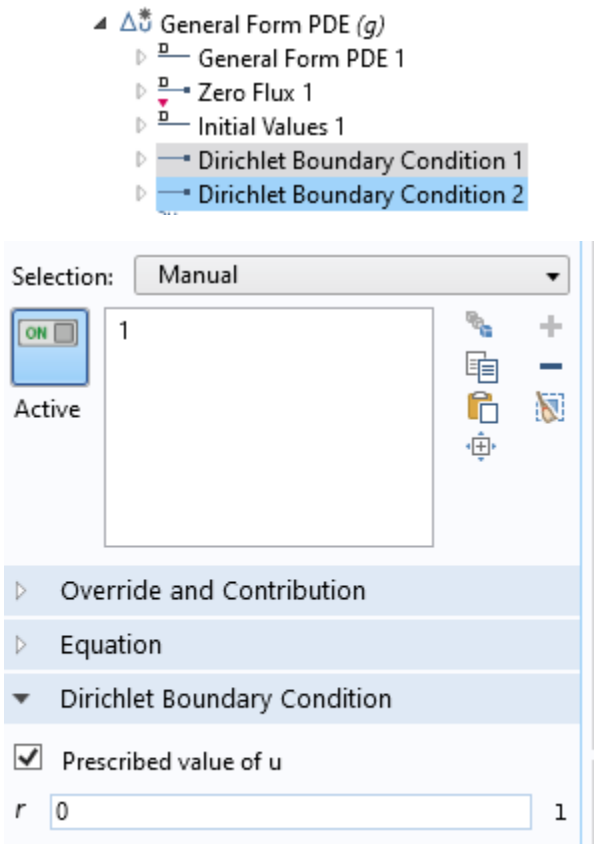

⑤. Επιλογή Αρχικής Συνθήκης



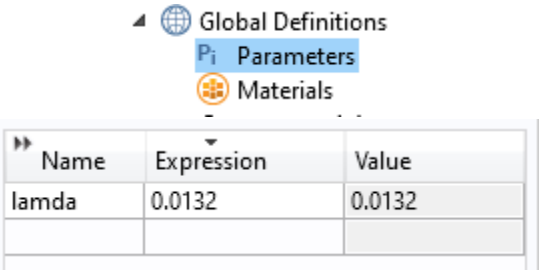

Επιλέγεται μία από τις αρχικές συνθήκες που έχουν δημιουργηθεί.

⑥. Εισαγωγή Συνοριακών Συνθηκών

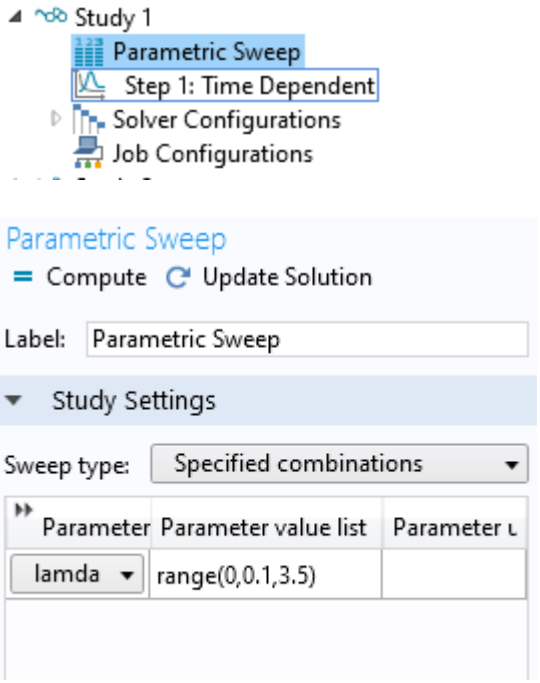



Προστίθενται οι συνοριακές συνθήκες στα σύνορα του χωρίου (Dirichlet).

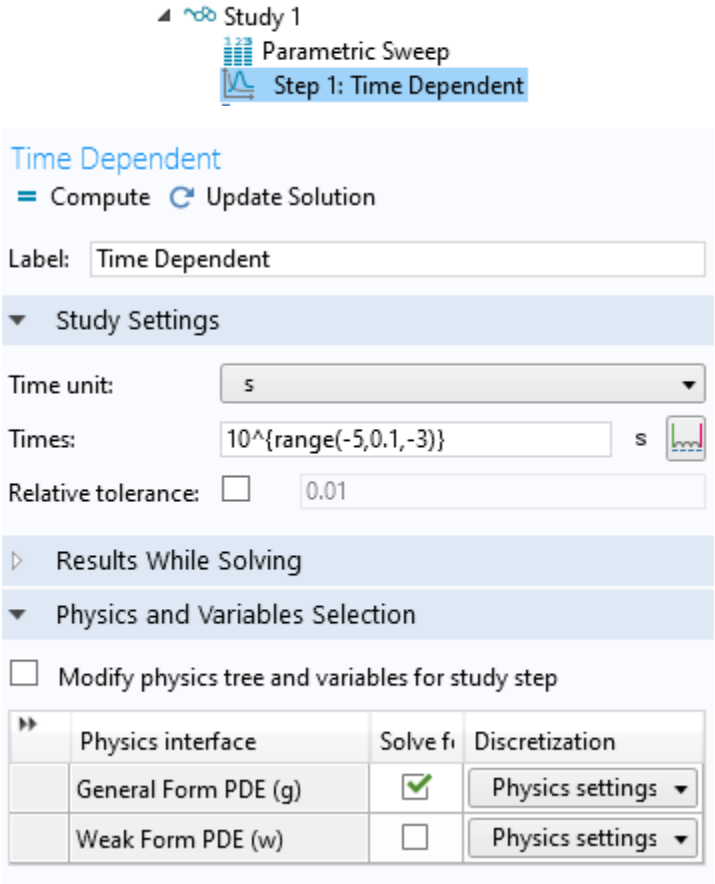
⑦. Παραμετρική Ανάλυση




Αφού η λ οριστεί ως παράμετρος και της δοθεί μια αρχική τιμή, προστίθεται στη μελέτη (study) η παραμετρική σάρωση (parametric sweep), ώστε να ληφθούν δεδομένα για πολλά διαφορετικά λ για την ίδια αρχική συνθήκη.

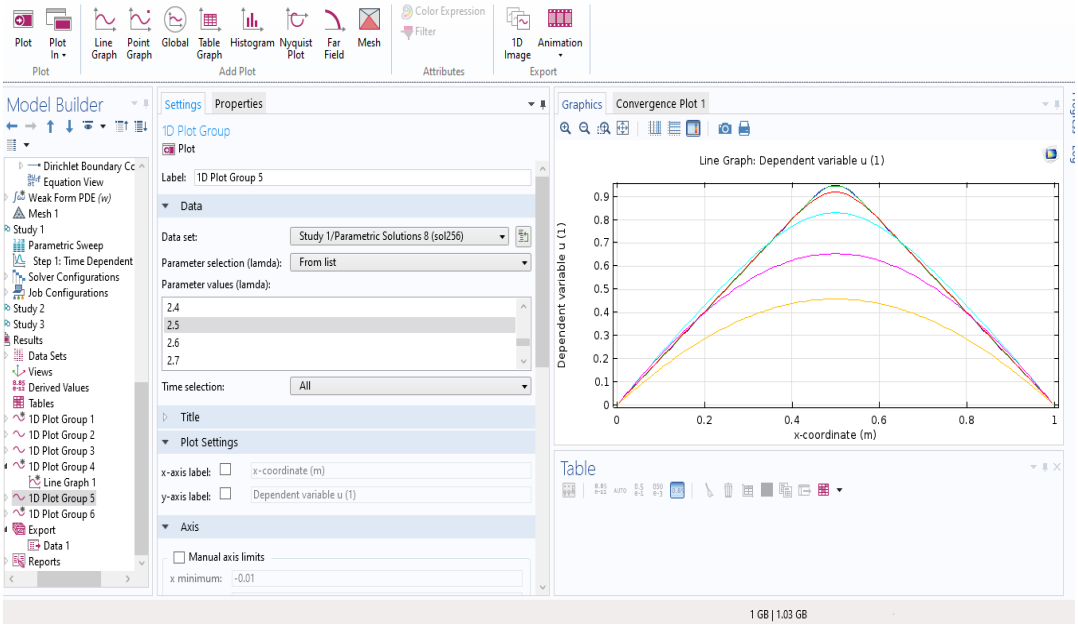



⑧. Επιλογή χρονικών στιγμών για εξαγωγή δεδομένων



Χρησιμοποιώντας την εντολή range, εισάγονται οι χρονικές στιγμές στις οποίες απαιτείται καταγραφή των δεδομένων για εξαγωγή. Όπως και στην περίπτωση της ode15s του Matlab, η επίλυση δεν λαμβάνει υπόψιν τους χρόνους αυτούς, αλλά τους επιστρέφει ως αποτέλεσμα γραμμικής παρεμβολής από εσωτερικά υπολογισμένες στιγμές.

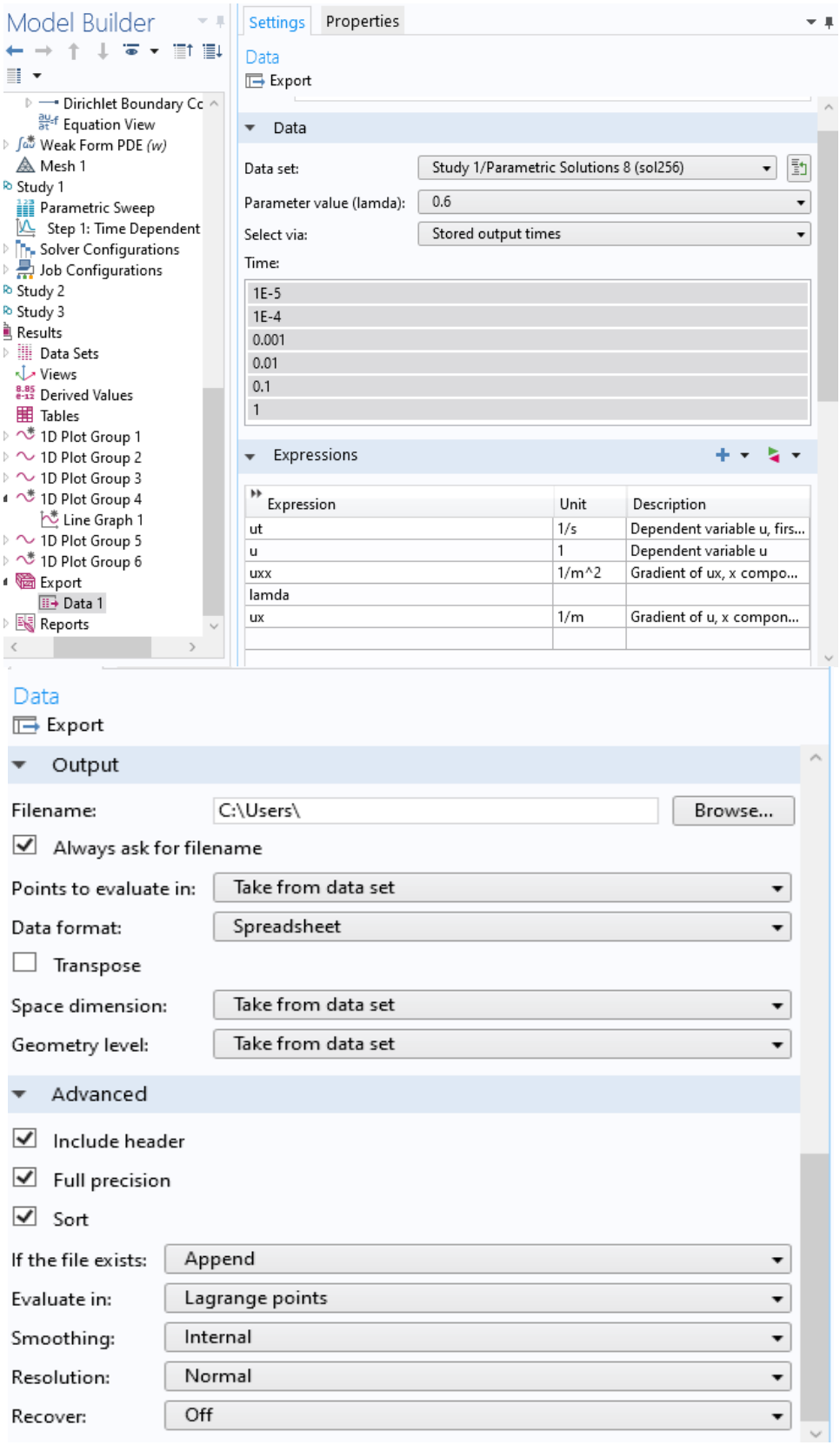
⑨. Έλεγχος Αποτελεσμάτων




Αφού λυθεί η εξίσωση, στην περιοχή 1D plot group ή 2D plot group, παρουσιάζεται η λύση του λογισμικού για κάθε χρονική στιγμή που έχει ζητηθεί. Εκεί διαπιστώνεται αν η λύση έχει νόημα.

⑩. Εξαγωγή Αποτελεσμάτων

Τέλος, από τη διεπιφάνεια που παρουσιάζεται στην επομένη σελίδα, πραγματοποιείται η λήψη των δεδομένων σε αρχείο τύπου .csv στο φάκελο που ορίζεται. Ορίζεται να μπαίνουν όλα τα δεδομένα στο ίδιο φύλλο και έπειτα γίνεται έλεγχος στο Excel για να διαπιστωθεί ότι η εργασία αυτή έγινε με επιτυχία. Έπειτα, ακολουθούνται οι διαδικασίες που περιγράφονται στο κεφάλαιο 4.


Model Builder
Dirichlet Boundary Cc
Equation View
Weak Form PDE (w)
Mesh 1
Study 1
Parametric Sweep
Step 1: Time Dependent
Solver Configurations
Job Configurations
Study 2
Study 3
Results
Data Sets
Views
Derived Values
Tables
1D Plot Group 1
1D Plot Group 2
1D Plot Group 3
1D Plot Group 4
Line Graph 1
1D Plot Group 5
1D Plot Group 6
Export
Data 1
Reports
Settings
Properties
Data
Export
Data
Data set:
Study 1/Parametric Solutions 8 (sol256)
Parameter value (lamda):
0.6
Select via:
Stored output times
Time:
1E-5
1E-4
0.001
0.01
0.1
1
Expressions
Expression
Unit
Description
ut
1/s
Dependent variable u, firs...
u
1
Dependent variable u
uxx
1/m^2
Gradient of ux, x compo...
lamda
ux
1/m
Gradient of u, x compon...
Data
Export
Output
Filename:
C:\Users\
Browse...
Always ask for filename
Points to evaluate in:
Take from data set
Data format:
Spreadsheet
Transpose
Space dimension:
Take from data set
Geometry level:
Take from data set
Advanced
Include header
Full precision
Sort
If the file exists:
Append
Evaluate in:
Lagrange points
Smoothing:
Internal
Resolution:
Normal
Recover:
Off