ΑΝΑΠΤΥΞΗ ΕΡΓΑΛΕΙΩΝ CAD ΓΙΑ ΣΧΕΔΙΑΣΗ ΟΛΟΚΛΗΡΩΜΕΝΩΝ ΚΥΚΛΩΜΑΤΩΝ – ΗΡΥ 608 / 419

Πρώτο Σετ Ασκήσεων - Εαρινό Εξάμηνο 2023
Προθεσμία: Παρασκευή 10/3/23, υποβολή on-line στο eClass

ΑΣΚΗΣΗ 1η Έκδοση 1.1 (με διορθώσεις 2/3)
Η ΜΕΘΟΔΟΣ NEWTON – RAPHSON

Έχουμε διδαχθεί στο μάθημα την μέθοδο Newton – Raphson (και την συναφή μέθοδο της τέμνουσας - secant). Στο σετ αυτό θα την υλοποιήσετε και θα μελετήσετε θέματα σύγκλισης, αριθμού επαναλήψεων, αλλά και αριθμητικής προσέγγισης (μέθοδος εφαπτομένης ή/και μέθοδος τέμνουσας) στην πρώτη παράγωγο σε σχέση με αναλυτική προσέγγιση. Για να κρατήσουμε απλό το πρόβλημα, θα θεωρήσουμε ότι η συνάρτησή μας είναι ένα πολυώνυμο έως $5^{ου}$ βαθμού, το αποδεκτό σφάλμα $e= 10^{-3}$ (0,1%) και οι πράξεις θα γίνουν με αριθμητική κινητής υποδιαστολής απλής ακρίβειας (float, όχι double).

Σε σχέση με το σφάλμα, όπως είπαμε στο μάθημα υπάρχουν εναλλακτικές προσεγγίσεις, και αναφέραμε ότι μία προσέγγιση είναι ότι αποδεχόμαστε το x_k αν $| x_k – x_{k-1} | < e$ ενώ η άλλη είναι $| f(x_k) | < e$. Μπορείτε να χρησιμοποιήσετε όποιο από τους δύο ορισμούς θέλετε (αλλά να το γράφετε σε σχόλια στον κώδικα και στην αναφορά), με την κατανόηση ότι υπάρχουν και άλλοι τρόποι να οριστεί το σφάλμα.

Θυμίζουμε ότι $x_{n+1} = x_n – [f(x_n) / f\,'(x_n)]$ – στο μάθημα εξηγήσαμε και πως προκύπτει.

Η είσοδος (μπορει να υπάρχει κάποιο Prompt στον χρήστη) θα είναι ο βαθμός του πολυωνύμου, και σε ξεχωριστή γραμμή οι συντελεστές κάθε δύναμης της μεταβλητής. Π.χ. η είσοδος

5
8 3 6 2 0 12

σημαίνει $8x^5 + 3x^4 + 6x^3 + 2x^2 + 0x + 12 = 0$

Εσείς πρέπει να βρείτε κάποια ρίζα, αλλά για να κρατήσουμε το πρόβλημα ρεαλιστικό μπορείτε να φροντίσετε να υπάρχει μόνο μία ρίζα (ενθαρρύνεστε όμως να πειραματιστείτε και με άλλα πολυώνυμα). Αποφύγετε όμως τετριμμένες εκδοχές του πολυωνύμου όπως μία πενταπλή ρίζα, κάτι που συμβαίνει π.χ. στο $(x – 5)^5$. Αυτό που ζητάμε είναι εύκολο αν ο βαθμός του πολυωνύμου είναι περιττός και μεταθέσετε ολόκληρο το πολυώνυμο κατά τον Y άξονα ώστε να τον τέμνει μόνο μία φορά. Η παραδοχή αυτή είναι ρεαλιστική γιατί εν γένει στα εργαλεία CAD μία είναι η ρίζα που μας ενδιαφέρει – δηλαδή το σημείο ισορροπίας του συστήματος (η τάση στο παράδειγμα που περιγράψαμε στο μάθημα), όπως η τάση σε κάποιο κόμβο όπου το αλγεβρικό άθροισμα των ρευμάτων από τον Νόμο του Kirchhoff είναι 0.

Η έξοδος του προγράμματός σας πρέπει να είναι κατ’ αρχήν το πολυώνυμο, καθώς και τα στοιχεία ανά επανάληψη, δηλ. από το ποιο είναι το x_0 (που επιλέγει το πρόγραμμά σας) και σε κάθε αριθμό επανάληψης ποιο είναι το κάθε x_k μέχρι να συγκλίνει το πρόγραμμα, βάζοντας ένα άνω όριο, π.χ. 25 επαναλήψεις, όπου αν δεν συγκλίνει το πρόγραμμα να τερματίζει (κάτι που θα πρέπει να αναφέρεται στην έξοδο, αν δηλαδή δεν τερμάτισε μετά από 25 επαναλήψεις). Στο τέλος του προγράμματος πρέπει να βγαίνουν τα στατιστικά όπως: αριθμός επαναλήψεων,

αριθμός προσθέσεων/αφαιρέσεων, αριθμός πολλαπλασιασμών, αριθμός διαιρέσεων. Ο κανόνας του Horner μπορεί να σας είναι χρήσιμος στον κώδικα.

Για να καταλάβουμε καλύτερα πως η προσέγγιση επηρρεάζει τα αποτελέσματα, λύσετε το πρόβλημα με δύο διαφορετικούς τρόπους (ουσιαστικά είναι ένας κώδικας, με την παράγωγο υλοποιημένη με δύο διαφορετικούς τρόπους):

(α) η παράγωγος της συνάρτησης υπολογίζεται αναλυτικά από το πρόγραμμά σας, και,
(β) η παράγωγος της συνάρτησης υπολογίζεται αριθμητικά. Αυτό σημαίνει ότι είτε διαλέγετε ένα μικρό δ και υπολογίζετε dy/dx σαν Δy/Δx και επομένως:
$x_{n+1} = x_n - f(x_n) / \{[(f(x_n+\delta) - f(x_n)] / \delta \}$ – αυτή είναι η μέθοδος της εφαπτομένης, ή, προσεγγίζετε το x_{n+1} με τον τύπο
$x_{n+1} = x_n - f(x_n) / \{[(f(x_n) - f(x_{n-1})] / [(x_n - x_{n-1})] \}$ – αυτή είναι η μέθοδος της τέμνουσας (secant). Στην μέθοδο της τέμνουσας ξεκινάμε με (αυθαίρετες αλλά προσεκτικά επιλεγμένες) τιμές για το x_0 και x_1 αφού χρειάζονται δύο προηγούμενες τιμές των x, f(x) για να βρούμε την επόμενη.

Ενδεχόμενα να ανακοινωθεί κάποιο dataset εκτός από το τι θα δοκιμάσετε μόνοι σας, ώστε να μπορούμε να κάνουμε συγκρίσεις πράξεων, επαναλήψεων, κλπ.

Πρέπει να λύσετε το πρόβλημα σε μία γλώσσα όπως C (να είναι procedural, structured, και τουλάχιστον το ίδιο το πρόγραμμα να είναι strongly typed ακόμη και αν η γλώσσα δεν είναι – η C δεν είναι strongly typed) για να έχετε μεγαλύτερο έλεγχο των πράξεων, και όχι απλά την παραγωγή αποτελέσματος. Υποβάλετε στο eClass το πρόγραμμα (κώδικα πηγής και εκτελέσιμο), καθώς και αναφορά, με σαφή ανάλυση για το πως το δ επηρρεάζει την σύγκλιση σε σχέση με την παράγωγο υπολογισμένη αναλυτικά, και τα datasets που χρησιμοποιήσατε. Προσοχή: οι κώδικες πρέπει να έχουν καλά σχόλια και η βασική δομή να αναφέρεται και στην αναφορά (όχι μόνο τα αποτελέσματα).

Επειδή ρωτήθηκα στο μάθημα, η Rust είναι απολύτως κατάλληλη για τις εργασίες του μαθήματος, αλλά κατά την εξέταση μπορεί να χρειαστεί να έρθετε με κάποιο laptop ή να βρούμε μηχάνημα με το κατάλληλο λειτουργικό και compiler. Είναι πάντως γλώσσα με τα επιθυμητά χαρακτηριστικά, και μάλιστα είναι καλύτερη από την C με την έννοια ότι είναι strongly typed και statically typed, έχει δε τα επιθυμητά χαρακτηριστικά από procedural languages παρότι υποστηρίζει και χαρακτηριστικά functional programming.

ΚΑΛΗ ΕΠΙΤΥΧΙΑ!!!