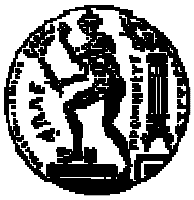


ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ & ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μελέτη του Λάθους Πεπερασμένης Ακρίβειας σε Σημαντικούς Αλγορίθμους

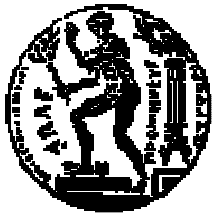
ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΠΑΠΑΓΕΩΡΓΙΟΥ ΜΑΡΙΑΣ

Επιβλέπων : Παπαοδυσσεύς Κώστας
Καθηγητής Ε.Μ.Π.

Αθήνα, Μάρτιος 2011



ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ &
ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μελέτη του Λάθους Πεπερασμένης Ακρίβειας σε Σημαντικούς Αλγορίθμους

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΠΑΠΑΓΕΩΡΓΙΟΥ ΜΑΡΙΑΣ

Επιβλέπων : Παπαοδυσσεύς Κώστας
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την .....

.....	.....	.....
Παπαοδυσσεύς Κώστας	Ηλίας Κουκούτσης	Μιχαήλ Θεολόγου
Αναπληρωτής	Επίκουρος	Καθηγητής Ε.Μ.Π.
Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	

Αθήνα, Μάρτιος 2011

.....

ΠΑΠΑΓΕΩΡΓΙΟΥ ΜΑΡΙΑ

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Στην παρούσα διπλωματική εργασία μελετάται η γένεση και διάδοση του αριθμητικού λάθους πεπερασμένης ακρίβειας, δηλαδή, του αριθμητικού λάθους που οφείλεται στο γεγονός ότι όλες οι παραστάσεις των αριθμών και οι μεταξύ τους πράξεις γίνονται στον υπολογιστή με περασμένο αριθμό ψηφίων. Αρχικά, πραγματοποιήθηκε μελέτη και κατανόηση μίας νέας σχετικής μεθοδολογίας που έχει αναπτύξει ο επιβλέπων καθηγητής και η ερευνητική του ομάδα. Εν συνεχεία, αυτή η μεθοδολογία εφαρμόστηκε για πρώτη φορά στη μελέτη της γένεσης και διάδοσης του λάθους πεπερασμένης ακρίβειας στους πολύ σημαντικούς αλγορίθμους υπολογισμού των Zernike Moments. Συγκεκριμένα, μελετήθηκαν οι πέντε δημοφιλέστεροι αναδρομικοί αλγόριθμοι υπολογισμού αυτών των ποσοτήτων. Διαπιστώθηκε ότι η μέθοδος έδινε τον ακριβή αριθμό λανθασμένων ψηφίων με τον οποίον κάθε ποσότητα του εκάστοτε αλγορίθμου υπολογιζόταν. Η μελέτη που πραγματοποιήσαμε κατέδειξε επίσης, ότι μόνο συγκεκριμένοι τύποι σε κάθε αλγόριθμο είναι οι κύριες και καθοριστικές πηγές γένεσης του λάθους πεπερασμένης ακρίβειας. Διεφάνη επίσης και κατενοήθη, γιατί τα αποτελέσματα των αλγορίθμων αυτών καθίστανται εντελώς αναξιόπιστα από μία τάξη αναδρομής και άνω. Τέλος, έγινε συγκριτική μελέτη της συμπεριφοράς των τριών κυριοτέρων και πλέον εξελιγμένων μορφών των αλγορίθμων υπολογισμού των Zernike Moments. Αυτή η σύγκριση υπέδειξε ότι ο πλέον ανθεκτικός σε βάθος τάξης αναδρομών αλγόριθμος είναι ο Q-recursive,ενώ οι αλγόριθμοι Kintner και Prata εμφάνιζαν σαφώς αυξημένη μέση τιμή λάθους σε όλες τις αναδρομές. Η παρούσα εργασία υποδεικνύει και το δρόμο για τη διόρθωση αυτού του αριθμητικού λάθους ή και τη σταθεροποίηση κάποιου αλγορίθμου Zernike Moments, γεγονός που έχει μεγάλη θεωρητική και πρακτική αξία.

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ:

Λάθος πεπερασμένης ακρίβειας, Ζερνίκε Μόμεντς, αναδρομικοί αλγόριθμοι, λάθος αποκοπής, μέθοδος q-recursive, αλγόριθμος Kintner, αλγόριθμος Prata

Abstract

In the present dissertation, a study of the generation and propagation of the finite precision error in iterative algorithms is undertaken. The finite precision error is due to the fact that all numbers in a contemporary computing machine are represented with a limited number of digits; in addition, all operations are executed with the same limitation. First, a bibliographical research was undertaken, mainly in connection with a new methodology for the study the finite precision error, which has been developed by the tutoring professor and his research team. Next, this methodology has been for the first time applied to the study of the generation and propagation of the finite precision error in the very important class of algorithms that compute the Zernike Moments. In fact, the five more popular iterative algorithms for the computation of these moments has been studied. It has been shown that the employed method offered the exact number of erroneous digits with which each quantity of these algorithms was computed in every iteration. The perform study and the related analysis manifest that only a limited number of formulae in each algorithm are decisively responsible for the generation of the finite precision error. The performed experiments and the related analysis also clarified why these algorithms fail after a certain number of iterations. In the dissertation, it has been also undertaken a comparative study of the finite precision error in the three more advanced versions iterative algorithms for the Zernike Moments computation. This comparison has shown that the more robust algorithm is the q-recursive one. The other two schemes, namely the Kintner algorithm and the Prata one, presented a higher mean value of erroneous digits practically in all iterations. Finally, this dissertation may open the way for remedying this finite precision error, so as to develop an algorithm that remains robust and stable for a large number of iterations.

KEY WORDS:
Finite precision error, Zernike Moments, round-off error in iterative algorithms, Q-Recursive, Kintner, Prata

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον καθηγητή, κ. Κώστα Παπαοδυσσέα για την συνεχή καθοδήγηση, την υποστήριξη και την πολύτιμη βοήθειά του, καθώς και τον κ. Παναγιώτη Ρουσόπουλο και όλη την ερευνητική ομάδα του κ. Παπαοδυσσέα για τη βοήθειά τους όποτε χρειάστηκε.
Τους ανθρώπους της ζωής μου.
Την οικογένειά μου...Ευτυχώς δεν άλλαξα πορεία!

Πίνακας Περιεχομένων

Α.1. Το πρόβλημα του λάθους πεπερασμένης ακρίβειας - Παραδείγματα

Α.1.1. Εισαγωγή.

Η πεπερασμένη ακρίβεια με την οποία πραγματοποιούνται οι πράξεις στον υπολογιστή ενδέχεται να εισάγει μια ποσότητα λάθους στο αποτέλεσμα. Όταν το αποτέλεσμα αυτό χρησιμοποιείται σε επόμενες πράξεις όπως π.χ. συμβαίνει σε αναδομικές διαδικασίες, τότε ενδέχεται να συσσωρευτεί και τελικά να δώσει εσφαλμένα ανεπιθύμητα αποτελέσματα που αποκλίνουν από τα αναμενόμενα ορθά κατά πολύ. Αυτό κάνει τα όποια αποτελέσματα του υπολογιστή αμφιβόλου ορθότητας και στόχος της εργασίας αυτής είναι ακριβώς η παρατήρηση του λάθους αυτού στις διάφορες πράξεις(πρόσθεση, αφαίρεση, πολλαπλασιασμό και διαίρεση), η εισαγωγή μεθόδων ακριβούς υπολογισμού του λάθους και διόρθωσης και τελικά η εφαρμογή των ανωτέρω σε σημαντικούς αλγορίθμους.

Α.1.2. Παραδείγματα.

Παράδειγμα Α.1.2.α.

Συχνά, χρειάζεται να κάνουμε υπολογισμούς με αριθμούς αποτελούμενους από πολλά δεκαδικά ψηφία που περιέχουν χρήσιμη πληροφορία στα τελευταία δεκαδικά τους ψηφία. Άλλες φορές πρόκειται για ιδιαίτερα μεγάλους αριθμούς και άλλες φορές για ιδιαιτέρως μικρούς. Ο χειρισμός τέτοιων ποσοτήτων από τον υπολογιστή ή γενικότερα από ένα σύστημα που χρησιμοποιεί πεπερασμένο αριθμό ψηφίων για τις πράξεις του, οδηγεί σε αστοχίες και σφάλματα. Υπάρχει τρόπος να αντιμετωπιστούν ορισμένα από αυτά τα προβλήματα για να αποφύγουμε να χάσουμε πληροφορία. Αυτή η απώλεια πληροφορίας και ο τρόπος παράκαμψής της φαίνεται στο παράδειγμα που ακολουθεί.

Μέσα από τη σχέση $\chi^2 - \psi^2 = (\chi+\psi)\cdot(\chi-\psi)$ θα κατανοήσουμε τι είναι τελικά πρακτικά το λάθος πεπερασμένης ακρίβειας και με ποια μορφή υπεισέρχεται στους υπολογισμούς μας. Πιο συγκεκριμένα, έστω :

$eps_1=10^{-19}$
$eps_2=2*10^{-19}$
$x=1+ eps_1$
$y=1+ eps_2$

Εκτελέσαμε τις ακόλουθες πράξεις στο MATLAB και λάβαμε τα παρακάτω αποτελέσματα:

$z_1= x^2 – y^2 = 0$
$z_2= (x+y)\cdot(eps_1- eps_2) = -2.0000e-019$
$z_3= 2\cdot eps_1+ eps_1^2-2\cdot eps_2- eps_2^2 = -2.0000e-019$

Όπως βλέπουμε, αν και τα αποτελέσματα έπρεπε να είναι ίδια, τελικά το πρώτο είναι διαφορετικό. Η πεπερασμένη ακρίβεια με την οποία πραγματοποιεί στοιχειώδεις πράξεις το MATLAB, είναι αντίστοιχη αυτής ενός υπολογιστή που εκτελεί πράξεις με 4 δεκαδικά ψηφία. Κατά συνέπεια, αν ο αριθμός μας όπως συνέβη με τους x και y, αποτελείται από τμήματα με διαφορετικές τάξεις μεγέθους οι οποίες διαφέρουν και κατά πολύ(θα προσδιορίσουμε αργότερα ποια είναι τα όρια αυτής της διαφοράς), τότε επειδή οι πράξεις θα πρέπει ταυτόχρονα να χειριστούν αριθμούς πολυψήφιους, μέρος της πληροφορίας μας θα χαθεί. Ο λόγος που στις παραστάσεις z_2 και z_3 δε χάθηκε η πληροφορία της διαφοράς μεταξύ των δύο αριθμών οφείλεται στο γεγονός ότι κάθε επιμέρους πράξη που πραγματοποιούμε χειρίζεται κάθε φορά αριθμούς χωρίς τμήματα με τάξεις μεγέθους που να διαφέρουν κατά πολύ. Δηλαδή, αντιμετωπίσαμε το παρατηρηθέν στην πρώτη περίπτωση σφάλμα, εκτελώντας ισοδύναμες διαδικασίες με όρους ολιγοψήφιους, ώστε να μη χάνεται πληροφορία του εκάστοτε αριθμού. Αυτό είναι ένα απλό παράδειγμα στο οποίο το λάθος πεπερασμένης ακρίβειας φαίνεται ξεκάθαρα.

Παράδειγμα Α.1.2.β.

Ανθρωπομορφιστικά το σύμβολο $\sqrt{x}$ χρησιμοποιείται για να περιγράψει με μοναδικό τρόπο τη θετική ρίζα της εξίσωσης $y^2=x$. Ο αριθμητικός όμως υπολογισμός αυτής της ρίζας υφίσταται πρακτικά πάντα f.p.e., λόγω ακριβώς του πεπερασμένου μήκους λέξης με το οποίο γίνονται οι πράξεις. Σε αρκετές εφαρμογές αυτό το f.p.e. δεν επηρεάζει τις πρακτικές εκτιμήσεις που θέλουμε να πραγματοποιήσουμε. Για παράδειγμα, εάν η εξαγωγή της τετραγωνικής ρίζας χρησιμοποιηθεί για να φτιαχτεί μια στοιχειώδης συσκευή καθημερινής χρήσεως που επιδέχεται κατασκευαστικής ανοχής, τότε είναι ήσσονος ή και καθόλου σημασίας το γεγονός ότι από τα 15 ψηφία με τα οποία παριστούμε την τετραγωνική ρίζα, τα 13 πρώτα είναι σωστά και τα δύο τελευταία είναι λάθος.

Υπάρχουν όμως περιπτώσεις, που αυτό το f.p.e. παίζει σημαντικό ρόλο και μπορεί να δημιουργήσει σοβαρότατα προβλήματα. Αυτό εμφανίζεται εντονότερα όταν οι πράξεις στις οποίες υπεισέρχεται ο υπολογισμός της τετραγωνικής ρίζας είναι αναδρομικές. Σημειωτέον ότι η απαίτηση όλα τα δεκαδικά ψηφία να είναι ορθά, δεν αφορά μόνον κατασκευές μεγάλης ακριβείας. Αντιθέτως, υπάρχουν περιπτώσεις που ακόμα κι αν η επίτευξη μεγάλης ακριβείας δεν απαιτείται, εν τούτοις τα αποτελέσματα ενός μικρού f.p.e. μπορεί να είναι καταστροφικά. Ας θεωρήσουμε για παράδειγμα, ένα οποιοδήποτε project στο οποίον πρέπει να υπολογίζουμε ιδιοτιμές ή και να επιλύω μεγάλο πλήθος γραμμικών συστημάτων. Ας υποθέσουμε, επομένως, ότι σε μία τέτοια εφαρμογή απαιτείτο να ελεγχθεί ως προς την αναστρεψιμότητα ο πίνακας

$$A = \begin{bmatrix} \sqrt{2} & \sqrt{6} \\ 1 & \sqrt{3} \end{bmatrix}$$

Το αν ένας πίνακας είναι αναστρέψιμος ή μη είναι πολύ συχνά ένα καίριο πρόβλημα σε πολλές εφαρμογές της επιστήμης των μηχανικών. Εάν δε, ληφθεί λανθασμένη απόφαση περί της αναστρεψιμότητας ενός ή περισσοτέρων πινάκων, τότε αυτό μπορεί να έχει πολύ δυσάρεστες ή και καταστροφικές συνέπειες στην όλη εξέλιξη του project. Είναι προφανές ότι ο πίνακας Α γραμμένος σε συμβολική μορφή δεν είναι αναστρέψιμος, διότι η ορίζουσά του είναι $\sqrt{2}\cdot\sqrt{3}-1\cdot\sqrt{6}=0$.

Για να καταδείξουμε το ρόλο του f.p.e. θα υπολογίσουμε τις τετραγωνικές ρίζες στο MATLAB, το οποίο έχει πολύ ικανοποιητικό για τις περισσότερες πρακτικές εφαρμογές μήκος λέξης, ίσο με 16 δεκαδικά ψηφία. Τότε,

$$\sqrt{2} = 1.1414213562373095$$
$$\sqrt{3} = 1.732050807568877$$
$$\sqrt{6} = 2.449489742783178$$

και οπότε, η ορίζουσα του A είναι det(A)= $4.440892098500626 \cdot 10^{-16}$, η οποία είναι διάφορη του μηδενός από αυστηρής μαθηματικής απόψεως και άρα ο A αντιστρεπτός. Αντίστοιχα ισχύουν και για ορίζουσες μητρών μεγαλυτέρου βαθμού.

Για παράδειγμα, υπάρχει μία κλάση πινάκων mxn οι οποίοι παράγονται από μονοδιάστατους πίνακες που λέγονται και διανύσματα. Συγκεκριμένα, έστω ο μονοδιάστατος πίνακας $x=[x_1 ,x_2 ,x_3]$ και ο 3x3 πίνακας που προκύπτει από τη σχέση πολλαπλασιασμού πινάκων

$$\beta = xTx = \begin{matrix} x_1 \cdot x_1 & x_2 \cdot x_1 & x_3 \cdot x_1 \\ x_1 \cdot x_2 & x_2 \cdot x_2 & x_3 \cdot x_2 \\ x_1 \cdot x_3 & x_2 \cdot x_3 & x_3 \cdot x_3 \end{matrix}$$

Με ευθύγραμμες πράξεις προκύπτει ότι κάθε πίνακας β mxn αυτής της μορφής, που προκύπτει δηλαδή από πολλαπλασιασμό πινάκων διαστάσεων nx1 και 1xn, τότε αυτός έχει ορίζουσα 0.

Ας εξετάσουμε και ας αναδείξουμε το ρόλο του f.p.e. με ένα απλό παράδειγμα που ανήκει στην προαναφερθείσα κατηγορία πινάκων. Πράγματι, έστω ο 1x3 πίνακας

$$x=[\pi/2 \quad e^{1.2} \quad \cos(\sqrt{3})]$$

Τότε,

$$\beta = xTx = \begin{matrix} 2.4674 & 5.2152 & -0.2522 \\ 5.2152 & 11.0232 & -0.5331 \end{matrix}$$

-0.2522 -0.5331 0.0258.

Αλλά τότε, οι πράξεις που εκτελεί μία οποιαδήποτε γλώσσα προγραμματισμού για να υπολογίσει την ορίζουσα του β, εισάγει ένα λάθος πεπερασμένης ακρίβειας σχεδόν πάντα. Εν προκειμένω, στο MATLAB η ορίζουσα του β λαμβάνει την τιμή det(β)= $-1.6071\cdot10^{-32}$ η οποία είναι από αυστηρά μαθηματικής απόψεως διάφορη του μηδενός, γεγονός που έρχεται σε αντίθεση με τη θεωρία, με απρόβλεπτες πρακτικές συνέπειες.

Στο σημείο αυτό, θα θέλαμε να κάνουμε μια παρατήρηση, η οποία μπορεί να επιλύσει πολλά πρακτικά προβλήματα. Πράγματι, επειδή κατά τον υπολογισμό της ορίζουσας, τελικά γίνονται αφαιρέσεις ποσοτήτων που διαφέρουν μόνο στα τελευταία - ελάχιστα σημαντικά - δεκαδικά ψηφία τους, τότε μπορούμε να δώσουμε έναν εναλλακτικό ορισμό του μηδενός που ισχύει σε οποιοδήποτε υπολογιστικό σύστημα. Στη συγκεκριμένη εφαρμογή που γίνεται στο MATLAB και επειδή αυτό κάνει πράξεις 14 ή 15 δεκαδικά ψηφία το πολύ, τότε μπορούμε να ορίσουμε ένα άνω φράγμα ανοχής, το οποίο ορίζει ένα διάστημα τιμών που υποκαθιστά το θεωρητικά αποδεκτό μηδέν. Εν προκειμένω, ορίζουμε ότι μία ποσότητα z είναι 0 αν και μόνο αν $|z|\leq10^{-14}$. Η επιλογή του εκθέτη έγινε ακριβώς διότι η τελική πράξη ήταν αφαίρεση και επιπλέον διότι κάτω από 15 ψηφία τα αποτελέσματα του MATLAB είναι αναξιόπιστα. Σύμφωνα με αυτόν τον ορισμό του μηδενός για τη συγκεκριμένη εφαρμογή και το συγκεκριμένο υπολογιστικό εργαλείο, η ορίζουσα του πίνακα β είναι 0. Η γενική περίπτωση του ορισμού του μηδενός για όλους τους συνδυασμούς των βασικών πράξεων και για τα κυριότερα υπολογιστικά συστήματα, θα παρουσιαστεί σε μελλοντική δημοσίευση.

Παράδειγμα Α.1.2.γ.

Πρακτικά, στο πρώτο παράδειγμα περιγράψαμε πώς πραγματοποιείται απώλεια πληροφορίας. Δηλαδή, όπως είδαμε στους δύο αριθμούς x, y που περιέχουν πληροφορία 19 ψηφία μετά το ακέραιο μέρος τους ($1+eps_1=10^{-19}$ και $1+eps_2=2*10^{-19}$), όταν πήγαμε να τους αντιμετωπίσουμε σαν αριθμούς αυτούσιους και να τους υψώσουμε στο τετράγωνο, εφόσον το MATLAB είχε ακρίβεια 14 δεκαδικών ψηφίων,οτιδήποτε βρισκόταν μετά από αυτά τα ψηφία το αγνόησε. Έτσι οι δύο αυτοί

αριθμοί υψωμένοι στη δεύτερη δύναμη φάνηκαν τελικά ίδιοι, γι'αυτό αφαιρώντας τους πήραμε την τιμή 0.

Πιο συγκεκριμένα, ας θεωρήσουμε έναν υπολογιστή που πραγματοποιεί πράξεις με 7 δεκαδικά ψηφία για το float και 14 για το double. Θα παραθέσουμε πίνακα με παραδείγματα για την απωλεσθείσα κάθε φορά πληροφορία σε διάφορους αριθμούς όταν εκτελούμε πράξεις με 7 και αντιστοίχως με 14 δεκαδικά ψηφία. Η τελευταία στήλη δίνει τα ψηφία της διαφοράς μεταξύ των αναπαραστάσεων των αριθμών με 7 και 14 δεκαδικά ψηφία. (Οι αριθμοί που προκύπτουν αποτελούνται από 7 ψηφία).

ΠΡΟΣΟΧΗ:

Θεωρώ πως κατά τη μετατροπή ενός αριθμού γίνεται στρογγυλοποίηση και όχι απλό truncation. Επίσης, στο MATLAB χρησιμοποιούμε format long e, το οποίο πρακτικά μας εξασφαλίζει αντιμετώπιση των αριθμών με τουλάχιστον 14 δεκαδικά ψηφία.

A.	**Αρχική μορφή αριθμού**	**7 δεκαδικά ψηφία**	**14 δεκαδικά ψηφία**	**N14-N7**
1ος	7.23564782739526	7.2356478	7.23564782739526	$2.739526 \cdot 10^{-8}$
2ος	1.25863529328412	1.2586352	1.25863521328412	$1.328412 \cdot 10^{-8}$
Αθροισμα		8.4942830	8.49428312067938	$1.2067938 \cdot 10^{-7}$
Αφαίρεση		5.9770126	5.97701253411114	$-6.5888860 \cdot 10^{-8}$
Πολλαπλασιασμός		9.1070410	9.10704114648241	$1.4648241 \cdot 10^{-7}$
Διαίρεση		1.7394921	1.73949208600056	$-1.3999440 \cdot 10^{-8}$

Βλέπουμε λοιπόν στον παραπάνω πίνακα την απώλεια πληροφορίας λόγω πεπερασμένης ακρίβειας στους υπολογισμούς μας. Παρακάτω παραθέτουμε πίνακα στον οποίο το MATLAB πραγματοποίησε τις πράξεις αφαίρεσης μεταξύ του N14 και του N7, δίνοντάς μας αριθμό αποτελούμενο από 14 δεκαδικά ψηφία, έναντι των 7 που πήραμε πραγματοποιώντας τις πράξεις μας οι ίδιοι αυστηρά στα 7 τελευταία ψηφία στα οποία διέφεραν.

Α.	Αρχική μορφή αριθμού	7 δεκαδικά ψηφία	14 δεκαδικά ψηφία	N14-N7	Αποτελεσμα πραξης στο MATLAB
1ος	7.23564782739526	7.2356478	7.23564782739526	$2.739526 \cdot 10^{-8}$	$2.739526028250339 \cdot 10^{-8}$
2ος	1.25863529328412	1.2586352	1.25863521328412	$1.328412 \cdot 10^{-8}$	$1.32841200262139 \cdot 10^{-8}$
Αθροισμα		8.4942830	8.49428312067938	$1.2067938 \cdot 10^{-7}$	$1.206793811547868 \cdot 10^{-7}$
Αφαίρεση		5.9770126	5.97701253411114	$-6.5888860 \cdot 10^{-8}$	$-6.588885970160163 \cdot 10^{-8}$
Πολλαπλασιασμός		9.1070410	9.10704114648241	$1.4648241 \cdot 10^{-7}$	$1.464824102015427 \cdot 10^{-7}$
Διαίρεση		1.7394921	1.73949208600056	$-1.3999440 \cdot 10^{-8}$	$-1.399944005164855 \cdot 10^{-8}$

Βλέπουμε λοιπόν πως εκτός από το σφάλμα που υπεισέρχεται στους υπολογισμούς μας λόγω πεπερασμένης ακρίβειας, στρογγυλοποίησης, truncation κλπ παρατηρείται κι άλλη ποσότητα σφάλματος στους υπολογισμούς μας. Αυτή η ποσότητα είναι ιδιατέρως σημαντική, παρ'όλο που η τάξη μεγέθους είναι αρκετά μικρότερη από τις ποσότητες των όρων που συμμετέχουν στις πράξεις μας. Η εισαγωγή αυτών των ψηφίων γίνεται μετά το shift των ψηφίων ώστε να προκύψει δεκαδικός αριθμός σε κανονική μορφή. Πρόκειται για εσφαλμένα νούμερα τα οποία εισάγει ο υπολογιστής μας ώστε να καλύψει τα 14(ή 15) ψηφία του υπολογιστικού μας συστήματος. Για να παρατηρήσουμε το εσφαλμένο αποτέλεσμα θα το συγκρίνουμε(αφαίρεση από το ορθό) ώστε να έχουμε μία ποσοτική εικόνα της συνεισφοράς του shift στο εκάστοτε λάθος αποτέλεσμα.

Α.	Αρχική μορφή αριθμού	N14-N7 (A)	Αποτελεσμα πραξης στο MATLAB (B)	|B|-|A|
1ος	7.23564782739526	$2.7395260 \cdot 10^{-8}$	$2.739526028250339 \cdot 10^{-8}$	$2.825033895703655 \cdot 10^{-16}$
2ος	1.25863529328412	$1.3284120 \cdot 10^{-8}$	$1.32841200262139 \cdot 10^{-8}$	$2.621389993961998 \cdot 10^{-16}$
Αθροισμα		$1.2067938 \cdot 10^{-7}$	$1.206793811547868 \cdot 10^{-7}$	$1.15478679951763 \cdot 10^{-15}$
Αφαίρεση		$-6.5888860 \cdot 10^{-8}$	$-6.588885970160163 \cdot 10^{-8}$	$-2.983983637960819 \cdot 10^{-16}$
Πολλαπλασιασμός		$1.4648241 \cdot 10^{-7}$	$1.464824102015427 \cdot 10^{-7}$	$2.015426936452514 \cdot 10^{-16}$
Διαίρεση		$-1.3999440 \cdot 10^{-8}$	$-1.399944005164855 \cdot 10^{-8}$	$5.16485498813779 \cdot 10^{-17}$

Όπως φαίνεται στον ανωτέρω πίνακα είναι το σφάλμα αρκετές τάξεις μεγέθους μικρότερο από τις συμμετέχουσες αρχικά ποσότητες όμως σε πολλές περιπτώσεις ακόμα και αυτό το σφάλμα λόγω της ακρίβειας που απαιτείται είναι ανεπιθύμητο και επιδρά ιδιαιτέρως αρνητικά στην αντίστοιχη εφαρμογή.

Παράδειγμα Α.1.2.δ.: Αλγόριθμος FAEST

Ο FAEST 5p ήταν η ταχύτερη μέθοδος προσδιορισμού αναδρομικά των ελαχίστων τετραγώνων. Η μέθοδος FAEST εισήχθη και βελτιώθηκε από τον Καραγιάννη [3]. Όλες οι μέθοδοι αυτές αντιμετωπίζουν σοβαρά αριθμητικά σφάλματα λόγω του λάθους πεπερασμένης ακρίβειας που υπεισέρχεται στους υπολογισμούς και πιο συγκεκριμένα μετά από ένα περιορισμένο αριθμό αναδρομών αποτυγχάνουν, δηλαδή, δίνουν εντελώς λανθασμένα αποτελέσματα. Όσον αφορά τον FAEST-5p σύμφωνα με πειράματα, υποφέρει κυρίως από το πεπερασμένο μήκος λέξης και αποτυγχάνει λοιπόν ταχύτερα. Αντιμετωπίζοντας τα εσφαλμένα αποτελέσματα στον FAEST σαν αποτέλεσμα της επίδρασης του λάθους πεπερασμένης ακρίβειας σε 4 μόνο λειτουργίες κατάφεραν οι [6] να προβλέψουν το ποσό του παραγόμενου λάθους και κυρίως να προτείνουν τρόπους διόρθωσης των ήδη υπαρχόντων αλγορίθμων και συγγραφής καινούριων.

Ο FAEST είναι αρκετά ασταθής, από την άποψη ότι μετά από ένα πολύ περιορισμένο αριθμό επαναλήψεων αποτυγχάνει. Από διάφορες παραμέτρους εξαρτάται ο αριθμός των επαναλήψεων μετά από τον οποίο ο αλγόριθμός μας δίνει αποτελέσματα λανθασμένα. Οι παράμετροι αυτές είναι: ο παράγοντας λ, οι αρχικές συνθήκες, η ταχύτητα της σύγκλισης, το «SNR», το χρησιμοποιούμενο αριθμητικό σύστημα, κλπ. Ωστόσο, με χρήση λ=0.97, SNR στο διάστημα (10-100db), και το αριθμητικό σύστημα που ορίζεται από το IEEE, τότε μεγάλο πλήθος εφαρμογών δείχνει ότι ο FAEST-5p αποτυγχάνει συνήθως μέσα στο διάστημα [1000,1200] όταν το σήμα εισόδου είναι λευκός θόυβος, ή μια περιοδική συνάρτηση.

Στο τέλος της εργασίας αυτής παραθέτουμε παράρτημα(APPENDIX A) στο οποίο περιγράφεται ο FAEST-5p.

Οι κυριότερες πηγές αριθμητικού λάθους για τον FAEST είναι:

1. Ο τύπος (A.5) που μας δίνει τον υπολογισμό του $w_{m+1(n+1)}$.
2. Οι τύποι (A.9) και (A.10) σαν ομάδα που υπολογίζουν το $\alpha_{m(n+1)}$.
3. Ο τύπος (A.13) για τον υπολογισμό του $b_{m(n+1)}$. Στην πραγματικότητα, ο (A.13) που υπολογίζει το $b_{m(n+1)}$ είναι μια βασική πηγή f.p.e., ενώ ο τύπος

(Α.8) που υπολογίζει το $w_{m(n+1)}$ είναι ένας μεταδότης του αριθμητικού λάθους.

4. Οι τρεις τύποι (Α.7), (Α.11) και (Α.12) για τον FAEST που χρησιμοποιούνται για τον υπολογισμό του $a^b_{m(n+1)}$.

Πιο συγκεκριμένα:

Α.1.2.δ.1. Το f.p.e. που παράγεται κατά τον υπολογισμό των $w_{m+1(n+1)}$

Με βάση την ιδιαίτερη φύση της εισόδου x(n+1) του λευκού θορύβου, ισχύει ότι στο δεξί τμήμα του τύπου (Α.5), οι δύο παράγοντες:

$$w_{j,m(n)} \quad \kappa\alpha\iota \quad -a_{j,m(n)} \cdot \frac{e^f_{m(n+1)}}{\lambda \cdot \alpha^f_{m(n+1)}}$$

Είναι στατιστικά αντιθέτου προσήμου και έχουν έναν αριθμό κοινών ψηφίων υπό τον ορισμό που δώσαμε. Έτσι, όσο ισχύουν οι συνθήκες αυτές η ποσότητα $w_{j,m+1(n+1)}$ φορτώνεται με κ=p-τ επιπρόσθετα δ.ψ.λ., όπου το τ είναι ο εκθέτης του $w_{j,m+1(n+1)}$ και p ο μέγιστος των εκθετών του $w_{j,m(n)} \quad \kappa\alpha\iota \quad -a_{j,m(n)} \cdot \frac{e^f_{m(n+1)}}{\lambda \cdot \alpha^f_{m(n+1)}}$. Γι'αυτό το λόγο, σε κάθε επανάληψη ένα επιπλέον ποσό λάθους κβαντισμού παράγεται και συσσωρεύεται στο $w_{m+1(n+1)}$, που μεταδίδεται στο κέρδος τάξης m του Kalman, $w_{m+1(n+1)}$ και τότε μέσω των τύπων (Α.4), (Α,5), (Α.13) και (Α.16) σε όλες τις ποσότητες του αλγορίθμου, και ιδιαίτερα στους συντελεστές του φίλτρου, $c_{m(n)}$.

Έτσι, μπορεί κανείς να προσδιορίσει με βάση τα παραπάνω τον αριθμό των δ.ψ.λ. με το οποίο όλες οι ποσότητες του αλγορίθμου αυτού υπολογίζονται, και οφείλονται μονάχα στις λειτουργίες που συνδέονται με τον υπολογισμό του $w_{m+1(n+1)}$.

Α.1.2.δ.2. Το f.p.e. που παράγεται κατά τον υπολογισμό των $a_{m(n+1)}$ και $b_{m(n+1)}$.

Ο υπολογισμός του $a_{m(n+1)}$ ίσως δι καιολογήσει τον όρο «κύρια πηγή f.p.e.». Ας υπολογιστεί, ο $a_{m(n+1)}$ μέσα από τη σχέση: $\alpha_{m(n+1)} = 1 - x^T_{m(n+1)} \cdot w_{m(n+1)}$. Εφόσον ο

πίνακας του συστήματος $\mathbf{R_{m(n)}}$, και για αυτό το λόγο και ο ανάστροφός του ο $\mathbf{R_{m(n)}}^{-1}$ είναι θετικά ορισμένοι για κάθε n, ισχύει ότι η ανισότητα $1<\alpha_{m(n+1)}$ πρέπει να ισχύει. Αυτό υπονοεί ότι οι ποσότητες 1 και $-x^T_{m(n+1)} \cdot w_{m(n+1)}$ έχουν το ίδιο πρόσημο. Τότε αυτό που πραγματικά εφαμόζεται στην ανωτέρω δοεθείσα εξίσωση είναι το $\alpha_{m(n+1)} = 1 + |x^T_{m(n+1)} \cdot w_{m(n+1)}|$ που δεν εισάγει λάθος λόγω πεπερασμένου μήκους λέξης.

Σε αντίθεση, ας υποθέσουμε ότι το $a_{m(n+1)}$ υπολογίζεται από τον τύπο:

$$\alpha_{m(n+1)} = \alpha_{m(n)} + e^b_{m(n+1)} \cdot \delta_{m(n+1)} + \frac{(e^f_{m(n+1)})^2}{\lambda \cdot \alpha^f_{m(n)}}$$

Που στην πραγματικότητα χρησιμοποιείται στο FAEST-5p και είναι ένας απλός συνδυασμός των τύπων (A.9) και (A.10). Τότε, εφόσον ο πίνακας συστήματος $\mathbf{R_{m(n)}}$ είναι θετικά ορισμένος για κάθε n, είναι προφανές ότι και οι δύο όροι $\alpha_{m(n)}$ και $\frac{(e^f_{m(n+1)})^2}{\lambda \cdot \alpha^f_{m(n)}}$ είναι θετικοί και η πρόσθεσή τους δεν παράγει κάποιο f.p.e. Αν, ωστόσο,οι ποσότητες $e^b_{m(n+1)} \cdot \delta_{m(n+1)}$ ήταν πάντα θετικές επίσης, τότε το $a_{m(n+1)}$ θα υπολογιζόταν μέσω του τύπου $\alpha_{m(n+1)} = \alpha_{m(n)} + e^b_{m(n+1)} \cdot \delta_{m(n+1)} + \frac{(e^f_{m(n+1)})^2}{\lambda \cdot \alpha^f_{m(n)}}$, χωρίς f.p.e., και επιπρόσθετα, υποθετικά τότε το $a_{m(n+1)}$ θα ήταν μία αυστηρά αυξανόμενη συνάρτηση του n. Στην πράξη βέβαια η ποσότητα $e^b_{m(n+1)} \cdot \delta_{m(n+1)}$ είναι συχνά αρνητική κάνοντας το $a_{m(n+1)}$ να ταλαντώνεται. Από την ταλάντωση αυτή ένα ποσό f.p.e. παράγεται. Το μήκος αυτού του λάθους μεταφέρεται μέσω των τύπων (A.2), (A.11) και (A.15) στον αλγόριθμο και πολλαπλασιάζεται στις επόμενες επαναλήψεις.

Κατά ανάλογο τρόπο, εξ'αιτίας της φύσης του σήματος εισόδου $x_{m(n+1)}$, εφοσον πραγματοποιούνται προϋποθέσεις που θα περιγράψουμε παρακάτω, μία ποσότητα λάθους παράγεται από τον τύπο (A.13) που παράγει το $b_{m(n+1)}$ και μέσω του (A.8) που υπολογίζει το $w_{m(n+1)}$ μεταφέρεται.

Α.1.2.δ.3. Το f.p.e. που παράγεται κατά τον υπολογισμό του $a^b_{m(n+1)}$.

Οι τρεις τύποι (Α.7), (Α.11)και (Α.12) που οδηγούν στον υπολογισμό του $a^b_{m(n+1)}$, αποτελούν βασική πηγή παραγωγής f.p.e. κατά τρόπο όμως διαφορετικό από τους προαναφερθέντες.

Στην πραγματικότητα αυτοί οι τρεις τύποι είναι ισοδύναμοι με τον εξής τύπο για τον υπολογισμό του $a^b_{m(n+1)}$:

$$\alpha^b_{m(n+1)} = \lambda \cdot \alpha^b_{m(n)} \left(1 + \frac{\lambda \cdot \alpha^b_{m(n)} (\delta_{m(n+1)})^2}{\alpha_{m(n+1)}} \right)$$

Εξ'αιτίας των απαιτήσεων για θετικά ορισμένο πίνακα συστήματος και με προϋπόθεση το λευκό θόρυβο σαν είσοδο η ποσότητα:

$$J_{m(n+1)} = 1 + \frac{\lambda \cdot \alpha^b_{m(n)} \cdot (\delta_{m(n+1)})^2}{\alpha_{m(n+!)}}$$

στην πράξη βρίσκεται συνέχεια εντός του διαστήματος: (1, 1.3) και η μέση τιμή του $J_{m(n+1)}$ βρίσκεται εντός του διαστήματος: (1.02, 1.09). Τώρα, εξ'αιτίας των περιορισμών στην τιμή του $J_{m(n+1)}$ και την απαίτηση για το θετικά ορισμένο $\mathbf{R_{m(n+1)}}$, και επειδή ο παράγιντας λ παραμένει κοντά στο 1^- (τυπικές τιμές γύρω στο 0.97, 0.98, 0.99 κλπ) από την εξίσωση για το $a^b_{m(n+1)}$ παραμένει πάντα κοντά στο 1^+. Γι' αυτό υπολογίζοντας το $a^b_{m(n+1)}$ μπορεί κανείς βασιζόμενος στις συνθήκες που ισχύουν κατά τον πολαπλασιασμό για το f.p.e. και τα δ.ψ.λ. πως ο πολλαπλασιασμός $\lambda a^b_{m(n)}$ σέβεται τον αριθμό των δ.ψ.λ. του $a^b_{m(n)}$ ή τα αυξάνει κατά ένα, ενώ είναι σχεδόν απίθανο να τα μειώσει κατά ένα.

Στη συνέχεια αν θεωρήσουμε πως ο πολλαπλασιασμός $\lambda a^b_{m(n)}$ εκτελείται με αποτέλεσμα το $O_{m(n+1)}=\lambda \cdot a^b_{m(n)}$ και η ποσότητα $J_{m(n)}$ υπολογίζεται από τον τύπο που δόθηκε ανωτέρω, τότε αν εκτελεστεί ο τελικός πολλαπλασιασμός $O_{m(n+1)} \cdot J_{m(n)}$, ώστε να υπολογιστεί το $a^b_{m(n+1)}$. Έτσι, εφόσον ισχύουν οι προαναφερθείσες προϋποθέσεις και όπως θα δούμε παρακάτω μέσα από τις συνθήκες που ισχύουν κατά τον

πολλαπλασιασμό όσον αφορά το f.p.e. και τα δ.ψ.λ. τα λανθασμένα ψηφία με τα οποία θα υπολογιστεί το $a^{b}_{m(n+1)}$ θα αυξηθούνε κατά 1.

Άρα με δύο τρόπους μπορούν τα δ.ψ.λ. να αυξηθούν κατά ένα:

Α. Μέσω του πολλαπλασιασμού $\lambda \cdot a^{b}_{m(n)}$ και

Β. Μέσω του πολλαπλασιασμού $O_{m(n+1)} \cdot J_{m(n)}$.

Και στις δύο περιπτώσεις ο ουσιαστικός λόγος της αύξησης των δ.ψ.λ. διαφαίνεται στις συνθήκες που θα περιγράψουμε παρακάτω για τον πολλαπλασιασμό. Προφανώς όσο το λ είναι μικρότερο τόσο συχνότερα τα δ.ψ.λ. θα αυξάνονται κατά ένα, εφόσον, όσο μικρότερο το λ, τόσο πιθανότερο είναι να ισχύει η σχέση:

$10-\varepsilon<man(\lambda) \cdot man(a^{b}_{m(n)})<10$ και άρα να ισχύουν οι προϋποθέσεις που αναφέραμε.

Α.2. Αυστηρός ορισμός του λάθους πεπερασμένης ακρίβειας

Το λάθος πεπερασμένης ακρίβειας(f.p.e.) θα μελετηθεί στο δεκαδικό σύστημα παρ'όλο που οι προτάσεις και οι ορισμοί μας ισχύουν ανεξαρτήτως μετρικού συστήματος. Το δεκαδικό προτιμήθηκε για λόγους ευκολίας και επειδή σίγουρα είναι πιο οικείο και πιο κατανοητό στον εκάστοτε αναγνώστη.

Α.2.1.Ορισμοί.

1. Θεώρησε δύο αριθμούς n_1 και n_2, γραμμένους σε κανονική εκθετική μορφή, και έχοντας τον ίδιο αριθμό n δεκαδικών ψηφίων στη mantissa. Δηλαδή,

$$n_1= d_1.d_2d_3\ldots d_n \text{ x } 10^{\tau} \text{ , } n_2= \delta_1.\delta_2\delta_3...\delta_n \text{ x } 10^{\rho}$$

όπου $\tau \geq \rho$

Τότε οι δύο αυτοί αριθμοί διαφέρουν κατά K δεκαδικά ψηφία αν και μόνο αν:

$$||n_1|-|n_2||=d\cdot 10^{\tau-(n-k)},\ 1\leq d<10$$

Α.2.2. Παρατηρήσεις.

- Για κάθε ποσότητα a εκφρασμένη σε κανονική εκθετική μορφή συμβολίζουμε man(a) για τη mantissa του αριθμού αυτού, και E(a) για τον εκθέτη του αριθμού.
- η συντομογραφία δ.ψ.λ σημαίνει δεκαδικά ψηφία λανθασμένα
- η συντομογραφία f.p.e. σημαίνει λάθος πεπερασμένης ακρίβειας(finite precision error)

2. Αν υποθέσουμε πως γράφονται όλες οι ποσότητες σε κανονική εκθετική μορφή με n δεκαδικά ψηφία στη mantissa. Ας υποθέσουμε ότι η σωστή τιμή μιας τυχαίας ποσότητας a είναι a_c αν θεωρήσουμε πως όλοι οι υπολογισμοί έγιναν με άπειρη ακρίβεια. Τότε μπορεί κανείς να ορίσει πως το a έχει

υπολογιστεί με ακριβώς τα τελευταία λ ψηφία λανθασμένα, αν και μόνο αν το a διαφέρει από το a_c κατά λ ψηφία με βάση τον ορισμό1.

Α.3. Το λάθος πεπερασμένης ακρίβειας στις θεμελιώδεις πράξεις

Α.3.1. Προτάσεις.

Α.3.1.1. Πρόταση 1.

Ας υποθέσουμε πως όλες οι συμμετέχουσες ποσότητες υπολογίζονται με πεπερασμένη ακρίβεια στα n δεκαδικά ψηφία της mantissa και ας θεωρήσουμε ότι κάθε ποσότητα υπολογίζεται μέσα από τη σχέση:

$$z_n = x_n \cdot y_n$$

Θεωρούμε πως, εξ'αιτίας προηγούμενων πεπερασμένης ακρίβειας υπολογισμών, η ποσότητα x υπολογίζεται με ακριβώς τα τελευταία λ δ.ψ.λ., ενώ το y έχει υπολογιστεί με μ το πλήθος δ.ψ.λ., όπου μ≤λ, χωρίς βλάβη της γενικότητας. Στην περίπτωση αυτή ισχύουν τα ακόλουθα:

i. **Αν $|man(x_n) \cdot man(y_n)| \geq 10$**

Τότε το z υπολογίζεται με ακριβώς τα τελευταία λ δ.ψ.λ. με πιθανότητα ίση προς P_m, και με λ-1 δ.ψ.λ. με πιθανότητα $1-P_m$ όπου, αν $x_n=x_{n,c} + e_{n,x}$ και $y_n=y_{n,c} + e_{n,y,}$ τότε P_m είναι η πιθανότητα η έκφραση:

$$E_m=\{man(x_{n,c})\cdot man(e_y)\cdot 10^{\mu-\lambda} + man(y_{n,c})\cdot man(e_x) + man(e_x)\cdot man(e_y)\cdot 10^{\mu-n}\}$$

να ανήκει στο διάστημα [10,100). Στην πραγματικότητα υπάρχει πολύ μικρή πιθανότητα το z να υπολογιστεί με ακριβώς τα τελευταία λ+1 δ.ψ.λ. η οποία μπορεί ωστόσο να θεωρηθεί ασήμαντη. Σε κάθε περίπτωση όλες αυτές οι πιθανότητες μπορούν σαφώς να υπολογιστούν με ένα κατάλληλο λογισμικό αν οι στατιστικές διακυμάνσεις των λαθών $e_{n,x}$ $e_{n,y}$ είναι γνωστές.

ii. **Αν $|man(x_n) \cdot man(y_n)|<10$**

Τότε το z υπολογίζεται με ακριβώς τα τελευταία λ δ.ψ.λ. με πιθανότητα Q_m ή λ+1 λ.δ.ψ. με πιθανότητα 1- Q_m όπου Q_m είναι η πιθανότητα η έκφραση:

$$E_m=\{man(x_{n,c})\cdot man(e_y)\cdot 10^{\mu-\lambda} + man(y_{n,c})\cdot man(e_x) + man(e_x)\cdot man(e_y)\cdot 10^{\mu-n}\}$$

να ανήκει στο διάστημα [10,100). Ας σημειώσουμε ότι υπάρχει μία μικρή πιθανότητα ο z να υπολογιστεί με τα τελευταία λ+2 δ.ψ.λ. που πρακτικά μπορεί να θεωρηθεί ασήμαντη. Και πάλι με κατάλληλο λογισμικό μπορεί να υπολογιστεί αυτή η πιθανότητα.

Για να έχουμε και μια ιδέα για το αν η στατιστική διακύμανση των προστιθέμενων πληθυσμών ψηφίων είναι ομοιόμορφη, τότε $P_m\approx0,5$ και $Q_m\approx0,5$.

iii. Αν x_{2n} και y_{2n} είναι οι τιμές των δύο όρων όταν ένα πεπερασμένο μήκος λέξης 2n χρησιμοποιείται. Τότε, δεδομένου ότι τα $e_{n,x}$, $e_{n,y}$, $e_{n,2x}$ και $e_{n,2y}$ είναι τυχαίες μεταβλητές με την ίδια συνάρτηση διακύμανσης τότε τα λάθη $e_{n,xy}$ $e_{2n,xy}$, που παράγονται όταν πολλαπλασιάζουμε mantissa μήκους n και 2n,έχουν τις ίδιες στατιστικές ιδιότητες.

Α.3.1.2. Πρόταση 2.

Ας υποθέσουμε πως όλες οι συμμετέχουσες ποσότητες υπολογίζονται με πεπερασμένη ακρίβεια στα n δεκαδικά ψηφία της mantissa και ας θεωρήσουμε επίσης ότι κάθε ποσότητα z υπολογίζεται ως το άθροισμα δύο ποσοτήτων x,y όπου $x*y\geq0$, κα για παράδειγμα έχουμε:

$$z= x + y$$

όπου $x=x_1,x_2\ldots x_n \times10^{\tau}$, $y=y_1,y_2 \ldots y_n \times10^{\rho}$

Ας υποθέσουμε επίσης χωρίς βλάβη της γενικότητας ότι ο εκθέτης του y είναι κατά ν μικρότερος από τον εκθέτη του x. Δηλαδή:

$$v=E(x)-E(y)=\tau-\rho\geq 0.$$

Επίσης,ας υποθέσουμε ότι το y έχει τα τελευταία μ δ.ψ.λ. ενώ το x έχει τα τελευταία λ. Τότε, ισχύει το ακόλουθο:

i. αν μ-ν>0 τότε ο z υπολογίζεται με $\kappa=\max\{\lambda,\mu-\nu\}$ δ.ψ.λ. με μια συγκεκριμένη πιθανότητα P_{a1}, ή με $\kappa=\max\{\lambda,\mu-\nu\}+1$ δ.ψ.λ. με πιθανότητα $1-P_{a1}$.

ii. Αν μ-ν≤0 ο z υπολογίζεται με λ δ.ψ.λ. και με δεδομένη πιθανότητα P_{a2}, ή με λ+1 δ.ψ.λ. και πιθανότητα ίση προς $1-P_{a2}$.

iii. Αν x_{2n} και y_{2n} είναι οι τιμές των δύο όροι όταν ένα πεπερασμένο μήκος λέξης 2n χρησιμοποιείται. Τότε, δεδομένου ότι τα $e_{n,x}$, $e_{n,y}$, $e_{n,2x}$ και $e_{n,2y}$ είναι τυχαίες μεταβλητές με την ίδια συνάρτηση διακύμανσης τότε τα λάθη $e_{n,xy}$ $e_{2n,xy}$, που παράγονται όταν αθροίζουμε mantissa μήκους n και 2n,έχουν τις ίδιες στατιστικές ιδιότητες.

Α.3.1.3. Πρόταση 3.

Ας υποθέσουμε πως όλες οι συμμετέχουσες ποσότητες υπολογίζονται με πεπερασμένη ακρίβεια στα n δεκαδικά ψηφία της mantissa και ας θεωρήσουμε επίσης ότι κάθε ποσότητα υπολογίζεται από τη φόρμουλα:

$$k=\frac{\beta}{\alpha}$$

Ας υποθέσουμε ότι εξ'αιτίας των προηγούμενων πεπερασμένης ακρίβειας υπολογισμών, η ποσότητα β έχει υπολογιστεί με ακριβώς τα τελέυταια λ δ.ψ.λ., ενώ η α με περισσότερα από λ δ.ψ.λ.. Τότε ισχύουν τα εξής:

i. **Αν** $\left|\frac{man(\beta)}{man(a)}\right| \geq 1$, τότε το k υπολογίζεται με ακριβώς τα τελευταία λ δ.ψ.λ. με πιθανότητα ίση προς P_{d1}, ή με λ-1 δ.ψ.λ. με πιθανότητα ίση προς $1-P_{d1}$. Πιο συγκεκριμένα, υπάρχει πολύ μικρή πιθανότητα ο k να προσδιοριστεί με τα τελευταία λ+1 δ.ψ.λ. οπότε και η πιθανότητα αυτή θεωρείται ασήμαντη.

ii. **Αν** $\left|\frac{man(\beta)}{man(a)}\right| < 1$, τότε το k υπολογίζεται με ακριβώς τα τελευταία λ δ.ψ.λ.
με πιθανότητα ίση προς P_{d2} ή τα λ+1 δ.ψ.λ. με πιθανότητα ίση προς $1-P_{d2}$.
Και πάλι για την ακρίβεια, υπάρχει μια ασήμαντη πιθανότητα για υπολογισμό του k με λ+2 δ.ψ.λ..
Και παλι με κατάλληλο λογισμικό μπορεί να υπολογιστεί αυτή η πιθανότητα.

iii. Αν β_{2n} και α_{2n} είναι οι τιμές των δύο όροι όταν ένα πεπερασμένο μήκος λέξης 2n χρησιμοποιείται. Τότε, δεδομένου ότι τα $e_{n,\beta}$, $e_{n,\alpha}$, $e_{n,2\beta}$ και $e_{n,2\alpha}$ είναι τυχαίες μεταβλητές με την ίδια συνάρτηση διακύμανσης τότε τα λάθη $e_{n,\beta\alpha}$ $e_{2n,\beta\alpha}$,που παράγονται όταν διαιρούμε mantissa μήκους n και 2n,έχουν τις ίδιες στατιστικές ιδιότητες.

Α.3.1.4. Πρόταση 4.

Ας υποθέσουμε πως όλες οι συμμετέχουσες ποσότητες υπολογίζονται με πεπερασμένη ακρίβεια στα n δεκαδικά ψηφία της mantissa και ας θεωρήσουμε επίσης ότι κάθε ποσότητα υπολογίζεται από τη φόρμουλα:

$$w= t - r \text{ , } t\ r>0$$

όπου $t=t_1,t_2\ldots t_n \times 10^{\rho}$, $w=w_1,w_2\ldots w_n \times 10^{\delta}$
Ας υποθέσουμε ότι εξ'αιτίας των υπολογισμών πεπερασμένης ακρίβειας η ποσότητα υψηλότερης τάξης έχει υπολογιστεί με ακριβώς τα τελευταία λ δ.ψ.λ., ενώ ο άλλος

όρος με εκθέτη μικρότερο ή ίσο έχει υπολογιστεί με ακριβώς τα τελευταία μ δ.ψ.λ.
(Να σημειωθεί πως στην περίπτωση ίσων εκθετών,δηλαδή αριθμών ίδιας τάξεως τότε με τον όρο «ποσότητα υψηλότερης τάξης» αναφερόμαστε στο μεγαλύτερο εκ των δύο αριθμών, ενώ για ίσους αριθμούς τυχαία προκύπτει η επιλογή αυτού που θα ονομάσουμε υψηλότερη τάξης εκ των δύο.)

Τότε κανείς παρατηρεί τις εξής περιπτώσεις:

i. **λ≥μ**, για παράδειγμα η ποσότητα υψηλότερης τάξης έχει τον ίδιο ή μεγαλύτερο αριθμό δ.ψ.λ.. Στην περίπτωση αυτή, εφόσον: $\delta \le \max\{\tau,\rho\} \Leftrightarrow E(w) \le \max\{E(t),E(r)\}$, τότε ο w υπολογίζεται είτε με τα τελευταία (λ+d) δ.ψ.λ. με πιθανότητα ίση προς P_{s1} είτε με τα τελευταία (λ+d+1)δ.ψ.λ. με πιθανότητα ίση προς 1- P_{s1},όπου: $d = \max\{\tau,\rho\} - \delta$,

λ<μ, για παράδειγμα η ποσότητα υψηλότερης τάξης έχει το μικρότερο αριθμό δ.ψ.λ. και σε αυτή την περίπτωση εφόσον $\delta \le \max\{\tau,\rho\} \leftrightarrow E(w) \le \max\{E(t),E(r)\}$

ii. **a.** **Αν λ≥μ-man{τ,ρ}+min{τ,ρ}**τότε ο wυπολογίζεται με τα τελευταία (λ+d) δ.ψ.λ. με πιθανότητα P_{s2} ή με τα τελευταία (λ+d+1) δ.ψ.λ. με πιθανότητα ίση προς 1- P_{s2}.

b. **Αν λ<μ-max{τ,ρ}+min{τ,ρ}** τότε ο w υπολογίζεται με τα τελευταία (d+e) δ.ψ.λ. με πιθανότητα ίση προς P_{s3} ή με τα τελευταία (d+e+1) με πιθανότητα ίση προς 1- P_{s3}, όπου θεωρώ ότι e=μ-max{τ,ρ}-δ.

Δηλαδή, η αφαίρεση δε μπορεί ποτέ να μειώσει το λάθος πεπερασμένης ακρίβειας αλλά σε αντίθεση με όλες τις υπόλοιπες διαδικασίες έχει τη δυνατότητα να οδηγεί αυθαίρετα σε έντονα φαινόμενα λάθους πεπερασμένης ακρίβειας σύμφωνα με τον αριθμό κατά τον οποίο ο εκθέτης του αποτελέσματος «βυθίζεται» με βάση πάντα το μεγαλύτερο εκ των δύο συμμετεχόντων εκθετών. Και πάλι με κατάλληλο λογισμικό μπορούν να προσδιοριστούν οι πιθανότητες P_{s1},P_{s2},P_{s3}.

iii. Αν t_{2n} και r_{2n} είναι οι τιμές των δύο όροι όταν ένα πεπερασμένο μήκος λέξης 2n χρησιμοποιείται. Τότε, δεδομένου ότι τα $e_{n,t}$, $e_{n,r}$, $e_{n,2t}$ και $e_{n,2r}$ είναι τυχαίες μεταβλητές με την ίδια συνάρτηση διακύμανσης τότε τα λάθη $e_{n,tr}$ $e_{2n,tr}$, που παράγονται όταν αφαιρούμε mantissa μήκους n και 2n,έχουν τις ίδιες στατιστικές ιδιότητες.

Α.3.2. Αποδείξεις.

Στο σημείο αυτό και με τη βοήθεια εφαρμογών θα αποδείξουμε και θα παρατηρήσουμε τα δ.ψ.λ. στις διάφορες στοιχειώδεις πράξεις που πραγματοποιούνται στον υπολογιστή.

Α.3.2.1. Πολλαπλασιασμός.

Έστω x_c ο άπειρης ακρίβειας αριθμός. Για παράδειγμα, έστω:

$$x_c = \frac{3}{7} = \overline{0.428571}$$

και $x_{n,c}$ ο αριθμός που προκύπτει αν από τον x_c κρατήσω αυστηρά τα n-ψηφία. (Τα ψηφία αυτά μπορεί να προκύψουν είτε από truncation-αυστηρή αποκοπή των ψηφίων από το n+1 και μετά, είτε από στρογγύλευση στο n-οστό δεκαδικό ψηφίο.)
Άρα έχουμε ότι,

$$x_{n,c}=x_c - \text{διόρθωση} \cdot 10^{-(n+1)}$$

όπου διόρθωση = $\delta_{n,c}$,δηλαδή η mantissa της διόρθωσης
Άρα, $man(x_{n,c})=m_{n,c}$
x_n=οποιαδήποτε παράσταση του αριθμού x_c σε n-δεκαδικά ψηφία, παράσταση που έχει φορτωθεί με f.p.e. Άρα,έστω ότι x_n : έχει λ δ.ψ.λ. .
Ορισμός για λ δεκαδικά ψηφία: $x_n=x_{nc}+d_n\ 10^{\tau-(n-\lambda)}$
όπου τ: ο max εκθέτης του αριθμού

ΕΦΑΡΜΟΓΗ:

Έστω x_{nc}=0.428571428571428571,

τ=-1,

n=18+1=(6+6+6)+1=19 ψηφία

(Το +1 οφείλεται στο shift που κάνουμε κατά ένα ώστε να τον αριθμό στην κανονική μορφή δεκαδικών αριθμών όπου η mantissa οφείλει να έχει τιμή στο διάστημα[1,10).)

Αλλάζω στο $x_{n,c}$ τα τελευταία 4 δεκαδικά ψηφία και προκύπτει το x_n.

Δηλαδή,

x_n=0.4285714285714<u>26328</u>

Άρα αφαιρώντας: $x_n - x_{nc} = -2.243 \cdot 10^{-15}$

Όπου ο εκθέτης της διαφοράς προκύπτει από τη σχέση:

τ-(n-λ)=-1-(18-4)

Όμοια για έναν άλλο αριθμό y, όπου η άπειρης ακρίβειας μορφή του y είναι ο y_c και

$$y_{nc} = y_c - \varepsilon_{nc} \cdot 10^{-(n+1)}$$

Όπου ο y_c έχει μ-δεκαδικά ψηφία και εκθέτη ρ

$$\text{Άρα } y_n = y_{nc} + f_n \cdot 10^{\rho-(n-\mu)}$$

Πολλαπλασιάζουμε x_n και y_n σε υπολογιστικό περιβάλλον με n- δεκαδικά ψηφία και άρα έχουμε:

$$z_n = x_n \cdot y_n$$

<u>*-Ερώτημα:*</u> Με πόσα δεκαδικά ψηφία λανθασμένα υπολογίστηκε ο z_n;

<u>*-Απάντηση:*</u> Αρχικά για την άπειρης ακρίβειας μορφή του αριθμού έχουμε:

$$z_c = x_c \cdot y_c$$

Και άρα $z_{nc} = x_c \cdot y_c - \zeta_{nc} \cdot 10^{-(n+1)}$

Και θα έχουμε λοιπόν:

$$
\begin{aligned}
x_n \cdot y_n &= (x_{nc} + d_n \cdot 10^{\tau-(n-\lambda)}) \cdot (y_{nc} + f_n \cdot 10^{\rho-(n-\mu)}) = \\
&= (x_c - \delta_{nc} \cdot 10^{-(n+1)} + d_n \cdot 10^{\tau-(n-\lambda)}) \cdot (y_c - \varepsilon_{nc} \cdot 10^{-(n+1)} + f_n \cdot 10^{\rho-(n-\mu)} = \\
&= x_c \cdot [y_c - \varepsilon_{nc} \cdot 10^{-(n+1)} + f_n \cdot 10^{\rho-(n-\mu)} - \\
&- \delta_{nc} \cdot 10^{-(n+1)} \cdot [y_c - \varepsilon_{nc} \cdot 10^{-(n+1)} + f_n \cdot 10^{\rho-(n-\mu)}] + \\
&+ d_n \cdot 10^{\tau-(n-\lambda)} \cdot [y_c - \varepsilon_{nc} \cdot 10^{-(n+1)} + f_n \cdot 10^{\rho-(n-\mu)}]
\end{aligned}
$$

Εάν ο αλγόριθμος δεν έχει καταστραφεί, τότε το γινόμενο:

$$- \delta_{nc} \cdot 10^{-(n+1)} \cdot [- \varepsilon_{nc} \cdot 10^{-(n+1)} + f_n \cdot 10^{\rho-(n-\mu)}]$$

είναι αμελητέο. Από τον όρο :

$$d_n \cdot 10^{\tau-(n-\lambda)} \cdot [y_c - \varepsilon_{nc} \cdot 10^{-(n+1)} + f_n \cdot 10^{\rho-(n-\mu)}]$$

μπορούμε να αγνοήσω μόνο το γινόμενο $d_n \cdot 10^{\tau-(n-\lambda)} \cdot \varepsilon_{nc} \cdot 10^{-(n+1)}$. Όσον αφορά τα υπόλοιπα δεν ξέρω τι κάνουν.

Άρα για το $z_n = x_n \cdot y_n$

$$
\begin{aligned}
&= x_c \cdot y_c - x_c \cdot \varepsilon_{nc} \cdot 10^{-(n+1)} + x_c \cdot f_n \cdot 10^{\rho-(n-\mu)} - y_c \cdot \delta_{nc} \cdot 10^{-(n+1)} + \\
&+ d_n \cdot y_c \cdot 10^{\tau-(n-\lambda)} + d_n \cdot f_n \cdot 10^{\tau-(n-\lambda)+\rho-(n-\mu)} = \\
&= x_c \cdot y_c - 10^{-(n+1)} \cdot [x_c \cdot \varepsilon_{nc} + y_c \cdot \delta_{nc}] + x_c \cdot f_n \cdot 10^{\rho-(n-\mu)} + \\
&+ d_n \cdot y_c \cdot 10^{\tau-(n-\lambda)} + d_n \cdot f_n \cdot 10^{\tau-(n-\lambda)+\rho-(n-\mu)}
\end{aligned}
$$

Έτσι,συγκρίνοντας τους εκθέτες των ανωτέρω όρων καταλήγουμε στο ότι ο $2^{ος}$ όρος αμελείται.

Α.3.2.2. Διαίρεση.

Κρατώντας τις ίδιες τιμές για τα x_c και y_c με την περίπτωση του πολλαπλασιασμού θα προσδιορίσουμε την πηγή λάθους

$$\frac{x_n}{y_n} = \frac{x_{nc} + d_n \cdot 10^{\tau-(n-\lambda)}}{y_{nc} + f_n \cdot 10^{\rho-(n-\mu)}} = \frac{(x_{nc} + d_n \cdot 10^{\tau-(n-\lambda)}) \cdot (y_{nc} - f_n \cdot 10^{\rho-(n-\mu)})}{y_{nc}{}^2 - (f_n \cdot 10^{\rho-(n-\mu)})^2} \Rightarrow$$

$$\frac{x_n}{y_n} = \frac{(x_n + d_n \cdot 10^{\tau-(n-\lambda)}) \cdot (y_{nc} - f_n \cdot 10^{\rho-(n-\mu)})}{y_n{}^2}$$

Προκύπτει δηλαδή ένα γινόμενο το οποίο με ανάλογο τρόπο μπορεί να αντιμετωπιστεί όπως και το γινόμενο που περιγράψαμε ανωτέρω, υπολογίζοντας το λάθος πεπερασμένης ακρίβειας όπως δείξαμε.

Α.4. Διαπίστωση πως η διεύσδυση του λάθους είναι πρακτικά ανεξάρτητη του μήκους λέξης (±2 δεκαδικά ψηφία)

Α.4.1 Πρόταση.

Ο αριθμός των δ.ψ.λ. που προφανώς βρίσκονται στο τέλος κάθε πεπερασμένης παράστασης αριθμού είναι στατιστικά ανεξάρτητος από το μήκος λέξης , δηλαδή, το πλήθος των ψηφίων με το οποίο αναπαριστώ τον αριθμό. Αυτή η ιδιότητα έχει σαν αποτέλεσμα το εξής: έστω ο ίδιος αριθμός x_c παριστάμενος με n δεκαδικά ψηφία, δηλαδή $x_{n,c}$ και με m δεκαδικά ψηφία, δηλαδή $x_{m,c}$ όπου m>n. Έστω λ η μέση τιμή των προβλεπόμενων δ.ψ.λ. στο $x_{n,c}$. Τότε, και η μέση τιμή των $x_{m,c}$ είναι ίδια, εφόσον έχουν τις ίδιες στατιστικές ιδιότητες. Επιπλέον, η μέγιστη διαφορά των πραγματικών δ.ψ.λ. είναι κάποιο πολλαπλάσιο της διασποράς σ. Είναι απολύτως εύλογο να υποθέσουμε ότι ο αριθμός των παραγομένων δ.ψ.λ. ακολουθεί γεωμετρική κατανομή (σχέση P_m, $1-P_m$) οπότε και για μεγάλο πλήθος πειραμάτων κατά προσέγγιση κανονική. Άρα, με πιθανότητα περίπου 99% προκύπτει ότι τόσο στο $x_{n,c}$ όσο και στο $x_{m,c}$ τα δ.ψ.λ. θα είναι λ±2.

Α.4.2. Παράδειγμα.

Ας υποθέσουμε ότι n=16 και m=32. Ας υποθέσουμε επίσης ότι κάνουμε κ-στο πλήθος πολλαπλασιασμούς διαδοχικά. Από την Πρόταση 1 για τον πολλαπλασιασμό προκύπτει η μέση τιμή των δ.ψ.λ.. Πράγματι, έστω ότι από τους κ-πολλαπλασιασμούς στους κ_1 το γινόμενο των mantissa ήταν ≥ 10 και στους $\kappa_2=\kappa-\kappa_1$ <10. Τότε, έχουμε μέση τιμή ανακούφισης ίση με $-(1-P_m)\cdot\kappa_1$, όπου το πρόσημο οφείλεται στην ανακούφιση που επέρχεται και μέση τιμή χειροτέρευσης ίση με
$(1-Q_m)\cdot\kappa_2$.Άρα έχουμε για τη μέση τιμή των ψηφίων μετά από κ-πολλαπλασιασμούς:

$$-(1-P_m)\cdot\kappa_1+(1-Q_m)\cdot\kappa_2$$

-Ερώτηση: Πώς βλέπουμε όμως πως είναι ανεξάρτητη του μήκους λέξης η μόλυνση;

Παράδειγμα: Έστω ότι η μέση τιμή των δ.ψ.λ. είναι 6 και για μήκος λέξης 16 και για μήκος λέξης 32 ψηφία. Επιπλέον, στο 99% των περιπτώσεων τα πραγματικά λάθος ψηφία θα είναι και στα 16 καιστα 32 : 6+2. Αν λοιπόν κρατήσω στα 16 τα πρώτα: 16-(6±2)=8 και στα 32 τα 32-(6+2)= 24 τότε είμαστε βέβαιοι με πιθανότητα 99% ότι τα 8 ψηφία στα 16 είναι «καθαρά» και τα 24 στα 32 είναι επίσης σωστά.

Το άνω όριο του λάθους είναι τα 6+2=8 ψηφία, για αυτό και χρησιμοποιώ το +2 στους υπολογισμούς μας, για να συμπεριλάβω και τη χειρότερη των περιπτώσεων. Άρα, συγκρίνοντας τα 16 με τα 24 ψηφία, μπορούμε με βεβαιότητα να ξέρω από ποιο σημείο και μετά στα 16 ξεκινάει η μόλυνση, ποια είναι τα δ.ψ.λ. στον αριθμό των n-δεκαδικών ψηφίων και ποιο είναι το αναμενόμενο αποτέλεσμά μας.

ΥΠΟΣΗΜΕΙΩΣΗ: Προϋπόθεση εφαρμογής είναι το να είναι το 2n-λ+2 πάντα μεγαλύτερο του n. Δηλαδή, $2n-\lambda+2>n \rightarrow n>\lambda-2$

Α.5. Πρακτικός υπολογισμός του f.p.e. σε συνεχόμενη ακολουθία βασικών πράξεων (περιγραφή αλγορίθμων, flow charts)

Στο σημείο αυτο θα παραθέσουμε αρχικά την ανάλυση των προγραμμάτων και της λογικής που ακολουθήσαμε ώστε να μελετήσουμε το f.p.e. και στη συνέχεια θα οργανώσουμε τα βήματά μας σε δομημένα flow charts για καλύτερη κατανόησή τους.

Α.5.1. Πολλαπλασιασμός.

Αρχικοποίηση:

Αρχικά, δίνουμε τιμές στα a1, fa1, b1, fb1 έτσι ώστε τα fa1 και fb1 να είναι οι float μορφές των a1 και b1. Στη συνέχεια, και εντός της main του προγράμματος, ορίζουμε τα gin kai fgin. Τα gin και fgin προκύπτουν ως τα γινόμενα των a1, b1 και fa1, fb1 αντίστοιχα, και τελικά το fgin αποτελεί τη float εκδοχή του gin .(Προσοχή. Δεν προέκυψε το fgin από truncation ή στρογγύλευση του double gin, αλλά από διαίρεση των float όρων.)

Περιγραφή:

Αρχικά, καλούμε τη συνάρτηση epistrofh_double_mantissa_ektheth και epistrofh_float_mantissa_ektheth με τους double και float a1 και fa1 ώστε να απομονώσουμε τη mantissa και τον εκθέτη των δύο αριθμών.

Στο σημείο αυτό αρχίζει ένα for loop που τελειώνει μετά το πέρας του πλήθους των αναδομών που ορίζουμε στο πρόγραμμά μας ως MEGISTO.

Έπειτα, καταχωρούμε σε πίνακες μήκους MEGISTO τις τιμές των a1, fa1, gin, fgin με στόχο τη διατήρηση της πληροφορίας μας σε κάθε αναδρομή, ώστε να μπορούμε δηλαδή, να μεταβάλουμε αν θέλουμε τις τιμές τους χωρίς να χάνουμε κάθε φορά την προηγούμενη τιμή και να συμπεριλάβουμε κάθε φορά το όποιο σφάλμα υπεισέρχεται στους όρους των πράξεων.

Έπειτα, καλούμε τη συνάρτηση θεωρητικής πρόβλεψης (theorhtikh_problepsh_ginomenou) με είσοδο τα a1, fa1, gin, fgin ώστε να προβλέψουμε το αναμενόμενο σφάλμα όπως το έχουμε αντιληφθεί θεωρητικά και να επιβεβαιώσουμε στο τέλος την ορθότητα της διαδικασίας μας. Απομονώνουμε επίσης και τη mantissa του αναμενόμενου σφάλματος και κρατάμε την τιμή της σε ένα πίνακα όπου θα καταχωρούμε τις mantissa που προκύπτουν από τη θεωρητική πρόβλεψη.

Επαναπροσδιορίζουμε στη συνέχεια τις νέες τιμές των gin και fgin πολλαπλασιάζοντας τα παλιά με a1 και fa1 αντίστοιχα. Τις νέες τιμές γινομένων που προέκυψαν, τόσο τη double όσο και τη float μορφή τα αποθηκεύουμε σε κατάλληλους νέους πίνακες για να διευκολύνουμε την αναφορά στα σημεία αυτά καθώς και την παρατήρηση.

Στη συνέχεια, καλούμε τη συνάρτηση epistrofh_double_mantissa_ektheth και epistrofh_float_mantissa_ektheth με τη double και float μορφή του gin(δηλαδή gin και fgin που προέκυψαν από τον τελευταίο πολλαπλασιασμό με a1 και fa1). Από τις συναρτήσεις αυτές, σε δείκτη απομονώνουμε τις τιμές των εκθετών για τη double και float μορφή του αριθμού μας ενώ επίσης από τη float μορφή κρατάμε τη mantissa σαν αριθμό σε πίνακα για να τη χρησιμοποιήσουμε στη συνέχεια. Πιο συγκεκριμένα, προσδιορίζουμε τη διαφορά(diafora) ανάμεσα στα fgin και στη float εκδοχή του gin. Με τη βοήθεια του diafora θα προσδιορίσουμε στη συνέχεια το πλήθος των λαθος ψηφίων. Βρίσκουμε επίσης και το μέγιστο εκθέτη μεταξύ της double και float μορφής του αριθμού(στόχος είναι να συμπεριλάβουμε τη διαφορά στους εκθέτες στην περίπτωση που π.χ. έχουμε τους αριθμούς 0.9999999 και 1.00000000000001). Κρατάμε λοιπόν το μέγιστο εκθέτη, και στη συνέχεια τον εισάγουμε μαζί με τη diafora σαν ορίσματα στη συνάρτηση pinax_lathos_pshfiwn,συνάρτηση που θα μας επιστρέψει το πλήθος λάθος ψηφίων στον αριθμό μας.

Συμπεριλαμβάνουμε στον αλγόριθμό μας και ένα if loop με το οποίο αυτοπροστατευόμαστε από το overload κάθε φορά που ο εκθέτης μας πέφτει κάτω από -30. Πολλαπλασιάζουμε δηλαδή, κάθε φορά με 10^{60} ώστε να μην έχουμε το φόβο του overload και να μπορούμε να παρατηρούμε τις επαναλήψεις του αλγορίθμου μας αντιμετωπίζοντας την αδυναμία του συστήματός μας να μας καλύψει την ανάγκη μας για μεγάλο εύρος τιμών στο οποίο και θα κινηθούμε.

Στη συνέχεια, κατασκευάζουμε πίνακα diafora_check και κάθε στοιχείο αποτελείται από τη διαφορά της mantissa (αριθμός) της float μορφής του double αριθμού αν αφαιρέσουμε τη mantissa του float αριθμού. Με είσοδο το τρέχον στοιχείο του diafora_check καλούμε τη συνάρτηση epistrofh_float_mantissa_ektheth και παίρνουμε στην έξοδο τα στοιχεία της(δηλαδή mantissa,εκθέτη κλπ).

Με στόχο να παρατηρήσουμε πού έχουμε ανακούφιση,τα σημεία δηλαδή όπου το γινόμενο της προηγούμενης mantissa επί τη mantissa του a1 είναι μεγαλύτερο του 10, φτιάχνω ένα if loop το οποίο πραγματοποιεί τον ανωτέρω έλεγχο και μας κρατάει σε πίνακα (checked_gin) τα στοιχεια του πίνακα diafora_check στα οποία επιβεβαιώθηκε η συνθήκη της ανακούφισης. Αμέσως μετά από τον έλεγχο της ανακούφισης, με ένα if loop ελέγχουμε τα σημεία στα οποία έχουμε μείωση του πλήθους των λάθος ψηφίων. Αντιστοίχως, κρατάμε τα σημεία αυτά σε ένα νέο πίνακα(reverse_checked).

Εδώ ολοκληρώνονται οι αναδρομές, λόγω ολοκλήρωσης του for loop μας.

Στο σημείο αυτό και αφού έχει ολοκληρωθεί το for loop για τις επαναληψεις που έχουμε επιλεξει να πραγματοποιήσουμε, φτιάχνω αρχεία με στόχο την παρατήρηση των σημείων ανακούφισης και τον έλεγχο της αντίστροφης διαδικασίας, ώστε να είμαστε σίγουροι για τις εκάστοτε πηγές λάθους ή διόρθωσης των ψηφίων του αριθμού μας.

Έτσι, τυπώνουμε αρχικά σε ένα αρχείο τα στοιχεια του πίνακα checked και στη συνέχεια ένα άλλο αρχείο αποτελούμενο από τα στοιχεία του reverse_checked ώστε να ελεγξουμε ότι πράγματι τα ίδια στοιχεία εμφανίζονται και άρα η αντίληψή

μας πάνω στο θέμα ήταν όντως σωστή. Τέλος, το πρόγραμμά μας αυτό αναλαμβάνει να τυπώσει σε ένα τρίτο αρχείο συγκεντρωμένα τη float μορφή του a1, τη double μορφή του, τη mantissa του θεωρητικά αναμενόμενου αποτελέσματος, την τιμή του gin και του fgin πριν τον εκάστοτε πολλαπλασιασμό με το a1 και fa1, καθώς και τις αντίστοιχες τιμές μετά τον πολλαπλασιασμό και τέλος το πλήθος των λάθος ψηφίων κάθε επανάληψης.

Παραθέτουμε παραδείγματα στη συνέχεια πινάκων συγκεντρωτικών στους οποίους φαίνονται για συγκεκριμένους αριθμούς η πορεία των αυξομειώσεων στο πλήθος λάθος ψηφίων, καθώς και τα σημεία στα οποία επαληθεύεται η συνθήκη ανακούφισης και κατά συνέπεια περιμένουμε κάποια μείωση των λανθασμένων ψηφίων, ώστε να μπορέσει ο αναγνώστης να ελέγξει την ορθότητα του προγράμματος.

Πράγματι, υλοποιήσαμε τον αλγόριθμο που περιγράψαμε παραπάνω λεπτομερώς. Όσον αφορά τις δηλώσεις μας και την αρχικοποίηση μεταβλητών είχαμε τα εξής:

MEGISTO = 30000(L)
fa1=1.001098845000000000
a1=1.001098845000000000
b1=7.3335354678e-024
fb1=7.3335354678e-024

Παρακάτω παραθέτουμε τον πίνακα που από τη μία δίνει τα σημεία στα οποία επαληθεύουμε θεωρητικά ότι θα έπρεπε να υπάρξει ανακούφιση, δηλαδή, σημεία στα οποία το γινόμενο της προηγούμενης mantissa επί τη mantissa του α1 θα είναι μεγαλύτερο του 10 και κατά συνέπεια θα μειώσει κατά ένα το πλήθος των δ.ψ.λ. και θα αναγκάσει ένα shift προς τα δεξιά κατά μία θέση ώστε να πάρει ο αριθμός μας την κανονική μορφή δεκαδικού. Δίπλα στα στοιχεία αυτά έχουμε παραθέσει και την τιμή της διαφοράς της float εκδοχής του double αριθμού, αν αφαιρέσουμε το float αριθμό. Επίσης, παραθέτουμε τα σημεία στα οποία παρατηρήθηκε ανακούφιση με βάση το πλήθος των δ.ψ.λ. . Δηλαδή, τα σημεία στα οποία το πλήθος των δ.ψ.λ. μειώθηκε

κατά 1. Βλέπουμε πως τα σημεία ταυτίζονται και πως ο αλγόριθμός μας δεν αστοχεί πουθενά.

ΕΛΕΓΧΟΣ ΣΥΝΘΗΚΗΣ ΑΝΑΚΟΥΦΙΣΗΣ (ορθή διαδικασία)		ΕΛΕΓΧΟΣ ΣΗΜΕΙΩΝ ΜΕΙΩΣΗΣ Δ.Ψ.Λ. (ανάστροφη διαδικασία)	
Diafora_check	**Σημείο ανακούφισης**	**Diafora_check**	**Σημεία μείωσης του πλήθους λάθος ψηφίων**
-7.0333481e-006	281	-7.0571899e-005	281
-6.1273575e-005	2377	-6.1225891e-004	2377
-1.1646748e-004	4474	-1.1634827e-003	4474
-1.7178059e-004	6571	-1.7166138e-003	6571
-2.2745132e-004	8667	-2.2716522e-003	8667
-2.8192997e-004	10764	-2.8142929e-003	10764
8.9991865e+000	12860	-3.3569336e-003	12860
-3.9088726e-004	14957	-3.9052963e-003	14957
8.9968929e+000	17053	-4.4612885e-003	17053
-5.0055981e-004	19150	-5.0010681e-003	19150
8.9946003e+000	21246	-5.5437088e-003	21246
8.9983931e+000	23343	-6.1025620e-003	23343
-6.6673756e-004	25440	-6.6604614e-003	25440
8.9960995e+000	27536	-7.2097778e-003	27536
-7.7784061e-004	29633	-7.7714920e-003	29633

Συναρτήσεις.

- **Epistrofh_float_mantissa_ektheth**

Στόχος αυτής της συνάρτησης είναι να επιστρέψει σε δείκτες την τιμή της mantissa και του εκθέτη του αριθμού μας. Ο αριθμός μας είναι float.

Αρχικά με τη βοήθεια της sprintf μεταφέρουμε τον αριθμό μας σε έναν πίνακα και στη συνέχεια απομονώνουμε τη mantissa από τον εκθέτη μας. Αυτή η διαδικασία υλοποιείται με ένα loop που διαβάζει τα στοιχεία του πίνακά μας μέχρι να βρει το e, οπότε και κρατάει τη θέση του e. Τα στοιχεία μέχρι και πριν το e τα καταχωρεί σε ένα νέο πίνακα, τον πίνακα της mantissa, και τα στοιχεία από το e και μετά τα χησιμοποιεί ώστε να κατασκευάσει πίνακα και για τον εκθέτη.

Παρατήρηση: Για να διαβάσουμε τα στοιχεία του πίνακα, εφόσον πρόκειται για αριθμό float αρχικά, τα όρια που θα χρησιμοποιήσω στο for loop μας είναι προσαρμοσμένα στη μορφή του αριθμού μας, δηλαδή από 0 έως 14. Επίσης, στην

sprintf που χρησιμοποιήσαμε χρησιμοποιήσαμε format 10.7 και πάλι λαμβάνοντας υπόψη τη μορφή του αριθμού μας.

Στη συνέχεια, χρησιμοποιώντας τα στοιχεία των πινάκων και με χρήση της atof και της atoi μετατρέπω τον char πίνακα της mantissa και του εκθέτη μας σε float αριθμό και ακέραιο αριθμό αντίστοιχα. Τέλος, καρτάω σε ένα δείκτη τον ακέραιο αριθμό της mantissa και του εκθέτη του αριθμού μας (*num_mantissa_num, *num_ektheths_num) ώστε να μπορούμε τελικά να τα μεταφέρουμε στο πρόγραμμά μας.

- **Epistrofh_double_mantissa_ektheth**

Χρησιμοποιείται διαδικασία αντίστοιχη με αυτή της προηγούμενης συνάρτησης, αλλάζοντας τα μεγέθη των πινάκων μας ώστε να αντιστοιχούν σε double μορφή αριθμού καθώς επίσης και τα όρια στα for loops ώστε να μπορούμε να πραγματοποιήσουμε έλεγχο κατά μήκος ολόκληρου του πίνακά μας. Δηλαδή, το loop μας ξεκινάει από 0 έως 24. Και το format μας διαφοροποιείται στο sprintf ώστε να ανταποκρίνεται στη double μορφή του αριθμού μας. Διαφορά επίσης έχουμε και στη μετατροπή του πίνακα mantissa σε double αριθμό, όπου αντί για χρήση της συνάρτησης atof που χρησιμοποιήσαμε τη συνάρτηση our_atod όπου πρακτικά μετατρέπουμε έναν πίνακα char σε double αριθμό. Δηλαδή, στόχος και πάλι αυτής της συνάρτησης είναι να επιστρέψει σε δείκτες την τιμή της mantissa και του εκθέτη του αριθμού μας.

Αρχικά με τη βοήθεια της sprintf μεταφέρω τον αριθμό μας σε έναν πίνακα και στη συνέχεια απομονώνουμε τη mantissa από τον εκθέτη μας. Αυτή γίνεται με ένα loop που διαβάζει τα στοιχεία του πίνακά μας μέχρι να βρει το e, οπότε και κρατάει τη θέση του e και τα στοιχεία μέχρι και πριν αυτό τα καταχωρεί σε ένα νέο πίνακα, τον πίνακα της mantissa, και τα στοιχεία από το e και μετά τα χρησιμοποιεί ώστε να κατασκευάσει πίνακα χαρακτήρων και για τον εκθέτη. Στη συνέχεια, χρησιμοποιώντας τα στοιχεία των πινάκων και με χρήση της our_atod και της atoi μετατρέπουμε τον char πίνακα της mantissa και του εκθέτη μας σε double αριθμό και ακέραιο αριθμό αντίστοιχα. Τέλος, κρατάμε σε ένα δείκτη τον ακέραιο αριθμό της mantissa και του εκθέτη του αριθμού μας (*num_mantissa_num,

*num_ektheths_num) ώστε να μπορούμε τελικά να τα μεταφέρουμε στο πρόγραμμά μας.

- **Euresh_plhthous_lathos_pshfiwn**

Η συγκεκριμένη συνάρτηση αναλαμβάνει να επιστρέψει στην κύρια το πλήθος των λάθος ψηφίων του αριθμού που δέχθηκε σαν είσοδο.

Εκτελούμε τον αλγόριθμό μας με double και έπειτα με float με εισόδους τις αποκομμένες εκδοχές του double αριθμού. Για την ίδια ποσότητα p έχουμε μια παράσταση float έστω fp και double έστω dp. Κρατάμε τα 7 πρώτα ψηφία της double μορφής, δηλαδή του dp, και τα ονομάζουμε fdp. Εφαρμόζουμε τον ορισμό της βύθισης στα fp και fdp. Αν τα fdp είναι καθαρά τότε η σύγκριση των fdp με τα fp μέσω του λήμματος προσφέρει τον αρκιβή αριθμό δ.ψ.λ. του float.

-Ερώτηση: Πώς ξέρουμε ότι τα 7 πρώτα ψηφία του fdp είναι καθαρά;

Έχουμε θεμελιώσει ότι ο βαθμός μόλυνσης είναι ανεξάρτητος του μήκους λέξης (±2ψηφία). Αν το fp υπολογιστεί με μέχρι 5 δ.ψ.λ. από τα 7 τότε το dp θα έχει υπολογιστεί με το πολύ 5+2=7 δ.ψ.λ. (τα τελευταία ψηφία). Αυτό σημαίνει πως τα 7 πρώτα είναι καθαρά. Άρα η προσέγγιση είναι συνεπής.

Στην πράξη ακόμα και με 6 δ.ψ.λ. πάλι ο fdp υπολογίζεται με 8 δ.ψ.λ. το πολύ, άρα έχει τα 6 πρώτα καθαρά και άρα η σύγκριση fp και fdp εξακολουθεί να ισχύει. Αυτό παρακάμπεται εύκολα σε ένα σύστημα που κάνει πράξεις με απεριόριστο αριθμό ψηφίων στη mantissa αφήνοντας το ρόλο του float να τον παίζει ένας αριθμός n ψηφίων και το ρόλο του double ένας αριθμός 2·n+3 ψηφίων. Διότι τότε θα είμαστε σίγουροι ότι τα n πρώτα ψηφία του μεγάλου αριθμού θα είναι καθαρά όσο ο μικρός αριθμός δεν είναι κατεστραμμένος.

Πιο συγκεκριμένα, με είσοδο το τρέχον στοιχείο της diafora_check[i] στο πρόγραμμά μας, ελέγχουμε αν ο αριθμός μας είναι 0 ώστε να μην προχωρήσουμε στην αναζήτηση λάθος ψηφίων. Αν δεν ειναι μηδενικός ο αριθμός μας, τότε προχωράμε στην εύρεση των λάθος ψηφίων.

Καλούμε τη συνάρτηση epistrofh_float_mantissa_ektheth με είσοδο τον αριθμό μας ώστε να κρατήσουμε τη mantissa και τον εκθέτη μας σαν αριθμούς. Στη συνέχεια, υπολογίζουμε τη βύθιση, ως τη διαφορά μεταξύ του μέγιστου εκθέτη που επίσης σαν είσοδο από το κύριο πρόγραμμα εισάγουμε στη συνάρτησή μας και του εκθέτη που μας έδωσε η κλήση της epistrofh_float_mantissa_ektheth. Αν το λάθος μας μπαίνει σε βάθος μεγαλύτερο από το $7^{ο}$ δεκαδικό ψηφίο, τότε η βύθισή μας είναι μεγαλύτερη από 7 και άρα θεωρούμε πως δεν έχουμε δ.ψ.λ. Από την άλλη, αν το λάθος είναι εντός των ορίων του float αριθμού δηλαδή η βύθιση είναι μέχρι 7, τότε τα λάθος ψηφία είναι ίσα προς 8-βύθιση. Επιστρέφουμε τέλος, το πλήθος των λάθος ψηφίων στο πρόγραμμα που μας κάλεσε.

Παρατήρηση: Τι θα πει πως η βύθισή μας είναι μεγαλύτερη του 7;

Στο σημείο αυτό, θα θέλαμε να τονίσουμε ότι η μελέτη του λάθους πεπερασμένης ακρίβειας έγινε στο δεκαδικό σύστημα, διότι, αυτό είναι πολύ πιο προσιτό στον ανθρώπινο τρόπο σκέψης. Στις σημερινές όμως υπολογιστικές μηχανές, η παράσταση των αριθμών είναι δυαδική. Αυτό συνεπάγεται τη μη ύπαρξη σαφούς αντιστοιχίας μεταξύ του μήκους λέξης σε δυαδική και δεκαδική μορφή. Για παράδειγμα, στη δυαδική παράσταση απλής ακριβείας (παράσταση float για τη γλώσσα C)διατίθενται 32 bits, από τα οποία 24 χρησιμοποιούνται για τη mantissa και 8 για τον εκθέτη. Τα 24 bits της mantissa δεν αντιστοιχούν πάντα σε καθορισμένο αριθμό δεκαδικών ψηφίων. Αντιθέτως, άλλες φορές αντιστοιχούν σε 7 δεκαδικά ψηφία, άλλες σε 8 και ενδεχομένως άλλες σε 9, ανάλογα με την ακριβή τιμή του αριθμού, το αν είναι προσημασμένος κ.λ.π.. Επομένως, εμείς λαμβάνουμε σα βάση του δεκαδικού συστήματος για την παράσταση float το n=7. και αυτό χρησιμοποιούμε στον τύπο που δίνει τη βύθιση άρα και τον ακριβή αριθμό των δ.ψ.λ.. Εάν η βύθιση είναι μεγαλύτερη από 7, αυτό προφανώς σημαίνει ότι ο συγκεκριμένος αριθμός ήταν δυνατόν να παρασταθεί με 8 ή 9 δ.ψ.. Τότε, εύλογα θεωρούμε ότι ο αριθμός παράγεται χωρίς κανένα από τα 7 ψηφία του λανθασμένα.

- **Theorhtikh_problepsh_ginomenou**

Στόχος αυτής της συνάρτησης είναι το να προβλέψουμε θεωρητικά την τιμή της mantissa της διαφοράς. Ο τύπος που χρησιμοποιήσαμε μπλέκει πρακτικά διαφορά μεταξύ float αριθμού και float μορφής του double αριθμού για τους δύο αριθμούς που πολλαπλασιάζουμε με τους ίδιους τους αριθμούς. Οι διαφορές προκύπτουν αν από τους float άφαιρέσουμε το float κομμάτι των double. Οι αριθμοί που χρησιμοποιώ μπορούν να είναι είτε οι ίδιοι οι float, είτε οι double αριθμοί, είτε οι float μορφές των double αριθμών, με θετικότερα απότελέσματα στην τελευταία περίπτωση.

Στο τέλος επιστρέφουμε σε έναν δείκτη τη θεωρητική μας διαφορά στο κύριο πρόγραμμα.

Α.5.2. Διαίρεση.

Αρχικοποίηση:

Αρχικά, δίνουμε τιμές στα a1, fa1, b1, fb1 έτσι ώστε τα fa1 και fb1 να είναι οι float μορφές των a1 και b1 τα οποία τα έχουμε ορίσει αρχικά στις δηλώσεις μας να είναι τύπου double. Στη συνέχεια, και εντός της main του προγράμματος ορίζουμε τα phl και fphl προκύπτουν ως τα πηλίκα των b1, fb1 προς τα a1, fa1 αντίστοιχα, και τελικά το fphl αποτελεί τη float εκδοχή του phl.(Προσοχή. Δεν προέκυψε το fphl από truncation ή στρογγύλευση του double phl, αλλά από διαίρεση των float όρων.)

Περιγραφή:

Αρχικά, καλούμε τη συνάρτηση epistrofh_double_mantissa_ektheth και epistrofh_float_mantissa_ektheth με τους double και float a1 και fa1 ώστε να απομονώσουμε τη mantissa και τον εκθέτη των δύο αριθμών.

Στο σημείο αυτό αρχίζει ένα for loop που τελειώνει μετά το πέρας του πλήθους των αναδρομών που ορίζουμε στο πρόγραμμά μας ως MEGISTO.

Έπειτα, καταχωρούμε σε πίνακες μήκους MEGISTO τις τιμές των a1, fa1, fphl, phl με στόχο τη διατήρηση της πληροφορίας μας σε κάθε αναδρομή, ώστε να μπορούμε δηλαδή, να μεταβάλουμε αν θέλουμε τις τιμές τους χωρίς να χάνουμε κάθε φορά την προηγούμενη τιμή και να συμπεριλάβουμε κάθε φορά το όποιο σφάλμα υπεισέρχεται στους όρους των πράξεων. Τους πινακες αυτούς θα σχρησιμοποιήσουμε και στο τέλος για να τυπώσουμε σε κάποιο αρχείο τις τιμές του π.χ. a1 και να κάνουμε ευκολότερη την παρατήρησή μας. Οι πίνακες στους οποίους αποθηκεύουμε τα ανωτέρω στοιχεία είναι οι pinax_a1, pinax_fa1, old_fphl, old_phl.

Επαναϋπολογίζουμε τις τιμές των phl, fphl, όπως προκύπτουν αν εκ νέου διαιρέσουμε τα στοιχεία μας με a1, fa1 αντίστοιχα και τα αποτελέσματά μας τα καταχωρούμε στα τρέχοντα στοιχεία των πινάκων new_phl, new_fphl, σε καινούριες δηλαδή, μεταβλητές οι οποίες περιέχουν τη νέα τιμή του πηλίκου για το συγκεκριμένο στάδιο, είτε σε float είτε σε double μορφή.

Παρατήρηση: Ονομάζουμε τον αριθμό που προκύπτει από την εντολή (float)double_number, float μορφή του double_number.

Σε μία νέα μεταβλητή, την elegxos_theor_a καταχωρούμε τη διαφορά μεταξύ float αριθμού fa1 και float μορφής του a1 και αν ο αριθμός αυτός είναι ίσος με το 0, καλούμε τη συνάρτηση θεωρητικής πρόβλεψης (theorhtikh_problepsh_phlikou) με είσοδο τα phl, fphl ώστε να προβλέψουμε το αναμενόμενο σφάλμα όπως το έχουμε αντιληφθεί θεωρητικά και να επιβεβαιώσουμε μετά το πέρας της εκτέλεσης του προγράμματός μας, την ορθότητα της διαδικασίας μας. Απομονώνουμε επίσης και τη mantissa του αναμενόμενου σφάλματος και κρατάμε την τιμή της σε ένα πίνακα όπου θα καταχωρούμε τις mantissa που προκύπτουν από τη θεωρητική πρόβλεψη. Στην περίπτωση που η μεταβλητή elegxos_theor_a έχει τιμή διάφορη του 0, καλούμε τη συνάρτηση theorhtikh_problepsh_phlikou_diorthemenou με εισόδους τα old_phl,old_fphl,a1,fa1 και στόχους αντίστοιχους με αυτούς της προγούμενης συνάρτησης θεωρητικής πρόβλεψης. Ουσιαστικά, με αυτή την περιπτωσιολογία όταν η διαφορά είναι μηδενική γλιτώνουμε κάποιες επιπλέον πράξεις και η διαδικασία μας επιταχύνεται.

Στη συνέχεια, καλούμε τη συνάρτηση epistrofh_double_mantissa_ektheth και epistrofh_float_mantissa_ektheth με τους double και float αριθμούς phl και fphl (πρόκειται για τις νεότερες τιμές των phl και fphl μετά την τελευταία διαίρεση με τα a1 και fa1). Από τις συναρτήσεις αυτές, σε δείκτη απομονώνουμε τις τιμές των εκθετών για τη double και float μορφή του αριθμού μας ενώ επίσης από τη float μορφή κρατάμε τη mantissa σαν αριθμό σε πίνακα για να τη χρησιμοποιήσουμε στη συνέχεια. Πιο συγκεκριμένα, προσδιορίζουμε τη διαφορά(diafora) ανάμεσα στα fphl και στη float εκδοχή του phl. Με τη βοήθεια του diafora θα προσδιορίσουμε στη συνέχεια το πλήθος λαθος ψηφίων του fphl. Βρίσκουμε επίσης και το μέγιστο εκθέτη μεταξύ της double και float μορφής του αριθμού(στόχος είναι να συμπεριλάβουμε τη διαφορά στους εκθέτες στην περίπτωση που π.χ. έχουμε τους αριθμούς 0.9999999 και 1.0000001). Κρατάμε λοιπόν το μέγιστο εκθέτη, και στη συνέχεια τον εισάγουμε μαζί με τη diafora σαν ορίσματα στη συνάρτηση euresh_plhthous_lathos_pshfiwn, συνάρτηση που θα μας επιστρέψει το πλήθος λάθος ψηφίων στον αριθμό μας και θα το αποθηκεύσει σε ένα πίνακα, τον pinax_lathos_pshfiwn.

Παρατήρηση: Χρησιμοποιήσαμε και πάλι τη μέθοδο αυτοπροστασίας από το overload που αναφέραμε στην περίπτωση του πολλαπλασιασμού.

Στη συνέχεια, κατασκευάζουμε πίνακα diafora_check και κάθε στοιχείο αποτελείται από τη διαφορά της mantissa (αριθμός) της float μορφής του double αριθμού αν αφαιρέσουμε τη mantissa του float αριθμού. Με είσοδο το τρέχον στοιχείο του diafora_check καλούμε τη συνάρτηση epistrofh_float_mantissa_ektheth και παίρνουμε στην έξοδο τα στοιχεία της(δηλαδή mantissa,εκθέτη κλπ)

Για να παρατηρήσουμε πού έχουμε επιδείνωση του λάθους, τα σημεία δηλαδή όπου το πηλίκο της diafora_check προς το a1 ειναι μικρότερο της μονάδας(κατά απόλυτη τιμή!), φτιάχνουμε ένα if loop το οποίο πραγματοποιεί τον ανωτέρω έλεγχο και μας κρατάει σε πίνακα (checked) τα στοιχεια του πίνακα diafora_check στα οποία επιβεβαιώθηκε η συνθήκη της επιδείνωσης. Αμέσως μετά από τον έλεγχο της επιδείνωσης, με ένα if loop ελέγχουμε τα σημεία στα οποία έχουμε αύξηση του πλήθους των λάθος ψηφίων. Αντιστοίχως, κρατάμε τα σημεία αυτά σε ένα νέο πίνακα(reverse_checked).

Εδώ ολοκληρώνονται οι αναδρομές, λόγω ολοκλήρωσης του for loop μας.

Στο σημείο αυτό, και αφού έχει ολοκληρωθεί το for loop για τις επαναληψεις που έχουμε επιλεξει να πραγματοποιήσουμε, φτιάχνουμε αρχεία με στόχο την παρατήρηση των σημείων επιδείνωσης και τον έλεγχο της αντίστροφης διαδικασίας, ώστε να είμαστε σίγουροι για τις εκάστοτε πηγές λάθους ή διόρθωσης των ψηφίων του αριθμού μας.

Έτσι, τυπώνουμε αρχικά σε ένα αρχείο τα στοιχεια του πίνακα checked και στη συνέχεια ένα άλλο αρχείο αποτελούμενο από τα στοιχεία του reverse_checked ώστε να ελεγξουμε ότι πράγματι τα ίδια στοιχεία εμφανίζονται και άρα η αντίληψή μας πάνω στο θέμα ήταν όντως σωστή. Τέλος, το πρόγραμμά μας αυτό αναλαμβάνει να τυπώσει σε ένα τρίτο αρχείο συγκεντρωμένα τη float μορφή του a1, τη double μορφή του, τη mantissa του θεωρητικά αναμενόμενου αποτελέσματος, την τιμή του phl και του fphl πριν την εκάστοτε διαίρεση με το a1 και fa1, καθώς και τις αντίστοιχες τιμές μετά τη διαίρεση και τέλος το πλήθος των λάθος ψηφίων κάθε επανάληψης.

Παραθέτουμε παραδείγματα στη συνέχεια πινάκων συγκεντρωτικών στους οποίους φαίνονται για συγκεκριμένους αριθμούς η πορεία των αυξομειώσεων στο πλήθος λάθος ψηφίων, καθώς και τα σημεία στα οποία επαληθεύεται η συνθήκη επιδείνωσης και κατά συνέπεια περιμένουμε κάποια αύξηση των λανθασμένων ψηφίων, ώστε να μπορέσει ο αναγνώστης να ελέγξει την ορθότητα του προγράμματος.

Πράγματι, υλοποιήσαμε τον αλγόριθμο που περιγράψαμε παραπάνω λεπτομερώς. Όσον αφορά τις δηλώσεις μας και την αρχικοποίηση μεταβλητών είχαμε τα εξής:

MEGISTO = 7000(L)
fa1=1.0123119 ,fb1=1.9935354678e+030
a1=1.0123119, b1=1.9935354678e+030

ΕΛΕΓΧΟΣ ΣΥΝΘΗΚΗΣ ΕΠΙΔΕΙΝΩΣΗΣ (ορθή διαδικασία)		ΕΛΕΓΧΟΣ ΣΗΜΕΙΩΝ ΑΥΞΗΣΗΣ Δ.Ψ.Λ. (ανάστροφη διαδικασία)	
Diafora_check	**Σημείο επιδείνωσης**	**Diafora_check**	**Σημεία αύξησης του πλήθους λάθος ψηφίων**
1.0013580e-005	227		
1.0013580e-005	229	1.0013580e-005	229
1.0061264e-004	471	1.0061264e-004	471
1.0061264e-004	691	1.0061264e-004	691
1.0037422e-004	900	1.0037422e-004	900
1.0037422e-004	1105	1.0037422e-004	1105
1.0108948e-004	1307	1.0108948e-004	1307
1.0085106e-004	1507	1.0085106e-004	1507
1.0097027e-004	1705	1.0097027e-004	1705
1.0108948e-004	1902		
1.0013580e-004	1903	1.0013580e-004	1903
1.0120869e-004	2098	1.0120869e-004	2098
1.0085106e-004	2294	1.0085106e-004	2294
1.0085106e-004	2489	1.0013580e-004	2489
1.0013580e-004	2683	1.0085106e-004	2683
1.0085106e-004	2877		
1.0042191e-003	3071	1.0042191e-003	3071
1.0099411e-003	3264	1.0099411e-003	3264
1.0099411e-003	3457	1.0099411e-003	3457
1.0118484e-003	3649	1.0118484e-003	3649
1.0032654e-003	3842	1.0032654e-003	3842
1.0046959e-003	4034	1.0046959e-003	4034
1.0027885e-003	4226	1.0027885e-003	4226
1.0004044e-003	4418	1.0004044e-003	4418
1.0113716e-003	4609	1.0113716e-003	4609
1.0027885e-003	4801	1.0027885e-003	4801
1.0056496e-003	4992	1.0056496e-003	4992
1.0113716e-003	5183	1.0113716e-003	5183
1.0037422e-003	5375	1.0037422e-003	5375
1.0066032e-003	5566	1.0066032e-003	5566
1.0066032e-003	5757	1.0066032e-003	5757
1.0089874e-003	5948	1.0089874e-003	5948
1.0066032e-003	6139	1.0066032e-003	6139
1.0018349e-003	6330	1.0018349e-003	6330
1.0089874e-003	6520	1.0089874e-003	6520
1.0118484e-003	6710	1.0118484e-003	6710
1.0042191e-003	6901	1.0042191e-003	6901

Παραπάνω παραθέσαμε τον πίνακα που από τη μία δίνει τα σημεία στα οποία επαληθεύουμε θεωρητικά ότι θα έπρεπε να υπάρξει επιδείνωση, δηλαδή, σημεία στα οποία το επόμενο στοιχείο της diafora_check θα είναι μικρότερο της μονάδας και κατά συνέπεια θα αυξήσει κατά ένα το πλήθος των δ.ψ.λ. και θα αναγκάσει ένα shift προς τα αριστερά κατά μία θέση ώστε να πάρει ο αριθμός μας την κανονική μορφή δεκαδικού. Δίπλα στα στοιχεία αυτά έχουμε παραθέσει και την τιμή της διαφοράς της float εκδοχής του double αριθμού, αν αφαιρέσουμε το float αριθμό. Επίσης, παραθέτουμε τα σημεία στα οποία παρατηρήθηκε επιδείνωση με βάση το πλήθος των δ.ψ.λ. . Δηλαδή, τα σημεία στα οποία το πλήθος των δ.ψ.λ. αυξήθηκε κατά 1. Βλέπουμε πως τα σημεία ταυτίζονται και πως ο αλγόριθμός μας αστοχεί σε πολύ λίγα στοιχεία. Έχουμε ήδη όμως αναφέρει, από τον ορισμό των δ.ψ.λ. και τον τρόπο υπολογισμού αυτών πως το πλήθος των δ.ψ.λ. έχει ανοχή ± 2 ψηφία. Έτσι, σε ορισμένα σημεία και φυσικά λαμβάνοντας και πάλι υπόψη μας το γεγονός ότι στις πράξεις μας όπως εδώ κιόλας αποδείξαμε υπεισέρχεται κάποιο σφάλμα, τα σημεία στα οποία η συνθήκη μας αστοχεί είναι ελάχιστα και βρίσκονται εντός του στατιστικού εύρους ανοχής. Καταφεύγοντας μάλιστα στα αρχεία εξόδου της συνάρτησής μας στα οποία τυπώνεται το πλήθος των δ.ψ.λ. καθώς και οι τιμές των συμμετεχουσών ποσοτήτων, στα σημεία αυτά δεν υπάρχει στην πραγματικότητα επιδείνωση. Είναι προφανές άλλωστε πως τα σημεία αυτά είναι ιδιαιτέρως κοντά στα σημεία στα οποία τελικά πράγματι συμβαίνει μεταβολή και οριακά παρατηρείται κάποιο στοιχειώδες ασήμαντο σφάλμα.

Συναρτήσεις.

- **Epistrofh_float_mantissa_ektheth**

Στόχος αυτής της συνάρτησης είναι να επιστρέψει σε δείκτες την τιμή της mantissa και του εκθέτη του αριθμού μας. Ο αριθμός μας είναι float.

Αρχικά με τη βοήθεια της sprintf μεταφέρουμε τον αριθμό μας σε έναν πίνακα και στη συνέχεια απομονώνουμε τη mantissa από τον εκθέτη μας. Αυτή η διαδικασία υλοποιείται με ένα loop που διαβάζει τα στοιχεία του πίνακά μας μέχρι να βρει το e,

οπότε και κρατάει τη θέση του e. Τα στοιχεία μέχρι και πριν το e τα καταχωρεί σε ένα νέο πίνακα, τον πίνακα της mantissa, και τα στοιχεία από το e και μετά τα χησιμοποιεί ώστε να κατασκευάσει πίνακα και για τον εκθέτη.

Παρατήρηση: Για να διαβάσουμε τα στοιχεία του πίνακα, εφόσον πρόκειται για αριθμό float αρχικά, τα όρια που θα χρησιμοποιήσουμε στο for loop μας είναι προσαρμοσμένα στη μορφή του αριθμού μας,δηλαδή από 0 έως 14. Επίσης, στην sprintf που χρησιμοποιήσαμε χρησιμοποιήσαμε format 10.7 και πάλι λαμβάνοντας υπόψη τη μορφή του αριθμού μας.

Στη συνέχεια, χρησιμοποιώντας τα στοιχεία των πινάκων και με χρήση της atof και της atoi μετατρέπουμε τον char πίνακα της mantissa και του εκθέτη μας σε float αριθμό και ακέραιο αριθμό αντίστοιχα. Τέλος, κρατάμε σε ένα δείκτη τον ακέραιο αριθμό της mantissa και του εκθέτη του αριθμού μας (*num_mantissa_num, *num_ektheths_num) ώστε να μπορούμε τελικά να τα μεταφέρουμε στο πρόγραμμά μας.

- **Epistrofh_double_mantissa_ektheth**

Στόχος αυτής της συνάρτησης είναι να επιστρέψει σε δείκτες την τιμή της mantissa και του εκθέτη του αριθμού μας. Ο αριθμός μας είναι double.

Χρησιμοποιείται διαδικασία αντίστοιχη με αυτή της προηγούμενης συνάρτησης, αλλάζοντας τα μεγέθη των πινάκων μας ώστε να αντιστοιχούν σε double μορφή αριθμού καθώς επίσης και τα όρια στα for loops ώστε να μπορούμε να πραγματοποιώ έλεγχο κατά μήκος ολόκληρου του πίνακά μας. Δηλαδή, το loop μας ξεκινάει από 0 έως 24. Και το format μας διαφοροποιείται στο sprintf ώστε να ανταποκρίνεται στη double μορφή του αριθμού μας. Διαφορά επίσης έχουμε και στη μετατροπή του πίνακα mantissa σε double αριθμό, όπου αντί για χρήση της συνάρτησης atof που χρησιμοποιήσαμε τη συνάρτηση our_atod όπου πρακτικά μετατρέπουμε έναν πίνακα char σε double αριθμό.

Δηλαδή, στόχος και πάλι αυτής της συνάρτησης είναι να επιστρέψει σε δείκτες την τιμή της mantissa και του εκθέτη του αριθμού μας.

Αρχικά με τη βοήθεια της sprintf μεταφέρουμε τον αριθμό μας σε έναν πίνακα και στη συνέχεια απομονώνουμε τη mantissa από τον εκθέτη μας. Αυτή γίνεται με ένα loop που διαβάζει τα στοιχεία του πίνακά μας μέχρι να βρει το e, οπότε και κρατάει τη θέση του e και τα στοιχεία μέχρι και πριν αυτό τα καταχωρεί σε ένα νέο πίνακα, τον πίνακα της mantissa, και τα στοιχεία από το e και μετά τα χησιμοποιεί ώστε να κατασκευάσει πίνακα χαρακτήρων και για τον εκθέτη. Στη συνέχεια, χρησιμοποιώντας τα στοιχεία των πινάκων και με χρήση της our_atod και της atoi μετατρέπουμε τον char πίνακα της mantissa και του εκθέτη μας σε double αριθμό και ακέραιο αριθμό αντίστοιχα. Τέλος, καρτάω σε ένα δείκτη τον ακέραιο αριθμό της mantissa και του εκθέτη του αριθμού μας (*num_mantissa_num, *num_ektheths_num) ώστε να μπορούμε τελικά να τα μεταφέρουμε στο πρόγραμμά μας.

- **Euresh_plhthous_lathos_pshfiwn**

Η συγκεκριμένη συνάρτηση αναλαμβάνει να επιστρέψει στην κύρια το πλήθος των λάθος ψηφίων του αριθμού που δέχθηκε σαν είσοδο.

Εκτελούμε τον αλγόριθμό μας με double και έπειτα με float με εισόδους τις αποκομμένες εκδοχές του double αριθμού. Για την ίδια ποσότητα p έχουμε μια παράσταση float έστω fp και double έστω dp. Κρατάμε τα 7 πρώτα ψηφία της double μορφής, δηλαδή του dp, και τα ονομάζουμε fdp. Εφαρμόζουμε τον ορισμό της βύθισης στα fp και fdp. Αν τα fdp είναι καθαρά τότε η σύγκριση των fdp με τα fp μέσω του λήμματος προσφέρει τον αρκιβή αριθμό δ.ψ.λ. του float.

Στο κομμάτι περιγραφής του πολλαπλασιασμού περιγράφουμε πώς γνωρίζουμε πως τα 7 πρώτα ψηφία ενός αριθμού είναι καθαρά.

Με είσοδο το τρέχον στοιχείο της diafora_check[i] στο πρόγραμμά μας, ελέγχουμε αρχικά αν ο αριθμός μας είναι 0 ώστε να μην προχωρήσουμε στην αναζήτηση λάθος ψηφίων και να επιστρέψουμε στη main την τιμή 0 στο πλήθος των λάθος ψηφίων, εφόσον ο αριθμός μας πρακτικά δεν έχει μολυνθεί. Αν δεν ειναι μηδενικός ο αριθμός, τότε προχωράμε στη συνάρτησή μας.

Καλούμε τη συνάρτηση epistrofh_float_mantissa_ektheth με είσοδο τον αριθμό μας ώστε να απομονώσουμε τη mantissa και τον εκθέτη του. Στη συνέχεια, υπολογίζουμε τη βύθιση, ως τη διαφορά μεταξύ του μέγιστου εκθέτη και του εκθέτη που μας έδωσε η κλήση της epistrofh_float_mantissa_ektheth. Αν το λάθος μας μπαίνει σε βάθος μεγαλύτερο από το $7^{ο}$ δεκαδικό ψηφίο, τότε η βύθισή μας είναι μεγαλύτερη από 7 και άρα θεωρούμε πως δεν έχουμε λανθασμένα ψηφία.. Από την άλλη, αν το λάθος είναι εντός των ορίων του float αριθμού δηλαδή η βύθιση είναι μέχρι 7, τότε τα λάθος ψηφία είναι ίσα προς 8-βύθιση. Επιστρέφουμε τέλος, το πλήθος των λάθος ψηφίων στο πρόγραμμα που μας κάλεσε.

- **Theorhtikh_problepsh_phlikou**

Ο στόχος αυτής της συνάρτησης είναι να προβλέψουμε θεωρητικά την τιμή της mantissa της διαφοράς, όταν η διαφορά μεταξύ των a και fa είναι μηδενική. Λεπτομέρειες για τη συνάρτηση δίνουμε παρακάτω, στην περιγραφή της συνάρτησης theorhtikh_problepsh_phlikou_diorthwmenou, αφού η συνάρτηση αυτή αποτελεί απλοποιημένη μορφή της τελευταίας.

- **Theorhtikh_problepsh_phlikou_diorthwmenou**

Ο στόχος μας μέσω αυτής της συνάρτησης είναι να προβλέψουμε θεωρητικά την τιμή της mantissa της διαφοράς, όταν η διαφορά μεταξύ των a και fa είναι μη μηδενική.

Ο τύπος που χρησιμοποιήσαμε μπλέκει πρακτικά τη διαφορά μεταξύ float αριθμού και float μορφής του double αριθμού για τους δύο αριθμούς που διαιρούμε με τους ίδιους τους θεωρητικά πλήρεις αριθμούς. Οι διαφορές προκύπτουν αν από τους float άφαιρέσουμε το float κομμάτι των double. Οι αριθμοί που χρησιμοποιώ μπορούν να είναι είτε οι ίδιοι οι float, είτε οι double αριθμοί, είτε οι float μορφές των double αριθμών, με θετικότερα απότελέσματα στην τελευταία περίπτωση.

Α.5.3. Αφαίρεση.

Με την αφαίρεση δεν ασχοληθήκαμε ιδιαίτερα στο να περιγράψουμε διαδικασία εύρεσης του πλήθους λάθος ψηφίων, παρατήρησης των αναδρομών, διαδικασίας αναγέννησης του σφάλματος κλπ. διότι πρόκειται για μια πράξη πιο κατανοητή σε σχέση με τις πράξεις του πολλαπλασιασμού και της διαίρεσης. Ουσιαστικά μία πράξη αφαίρεσης μας βοηθάει να αντιληφθούμε εύκολα πώς υπεισέρχεται το f.p.e. σε αυτή. Θα μιλήσουμε για την αφαίρεση με ένα παράδειγμα το οποίο σίγουρα θα διαλευκάνει τη μόλυνση του αποτελέσματος αυτής(διαφοράς) με λανθασμένα ψηφία χωρίς να χρειαστούν πίνακες και πολύπλοκες σκέψεις. Για παράδειγμα ας θεωρήσουμε τις εισόδους στο πρόγραμμά μας:

a1=1.000987654321, b1=0.999999999999
fa1=1.000987654321, fb1=0.999999999999

Ορίζουμε μια διαφορά float και μία double μορφής έστω fdiaf και ddiaf αντίστοιχα, κάθε μία από τις οποίες προκύπτει αν από τον α1 αφαιρέσω τον b1 στην αντίστοιχη μορφή τους. Τυπώνοντας αυτές τις δύο διαφορές, παρατηρούμε πως τα αποτελέσματά μας διαφέρουν και μάλιστα αρκετά. Δηλαδή,

ddiaf = 9.876543e-004
fdiaf = 9.876490e-004

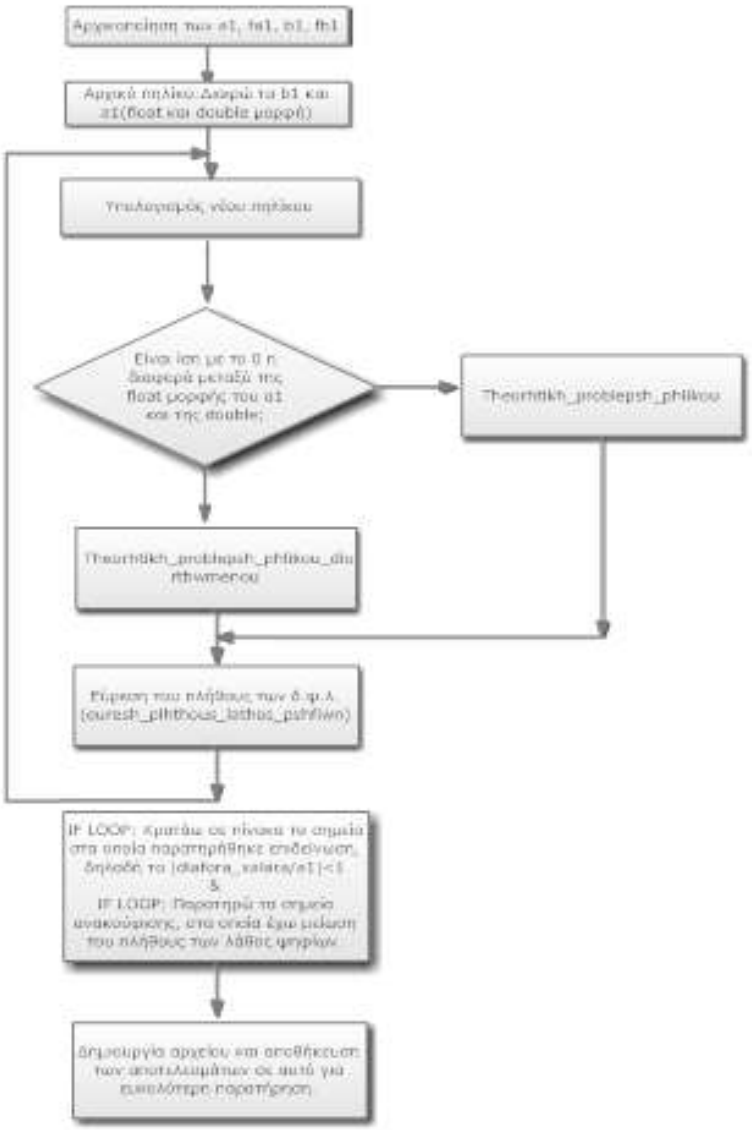
Είναι προφανές, δηλαδή, πως η αφαίρεση ως πράξη εισάγει έντονα φαινόμενα f.p.e. στα αποτελέσματά μας. Οι δύο αριθμοι που τυπώσαμε διαφέρουν κατά 2 δεκαδικά ψηφία (τα τελευταία) πρακτικά, παρ'όλο που μια πρώτη παρατήρηση δείχνει πως διαφέρουν κατά 3 (543-490=53). Αυτό είναι απολύτως συμβατό με τον ορισμό του υπολογισμού λάθος δεκαδικών ψηφίων που δόθηκε στο Α.2., ο οποίος άλλωστε προσφέρει τον ακριβή αριθμό δ.ψ.λ. που παράγονται κατά την αφαίρεση. Σημειώνεται ότι η αφαίρεση είναι η μοναδική πράξη η οποία μπορεί να παράξει οσονδήποτε μεγάλο αριθμό δ.ψ.λ.. Μπορεί ακόμα και να καταστρέψει έναν οποιοδήποτε αλγόριθμο σε μία αναδρομή. Αυτό συμβαίνει στην περίπτωση που οι δύο όροι της αφαίρεσης έχουν κατ' ουσίαν την ίδια τάξη μεγέθους και τα πρώτα

δεκαδικά ψηφία κοινά. Όσο μεγαλύτερο το πλήθος των συνεχόμενων κοινών ψηφίων, τόσο μεγαλύτερο το πλήθος των δ.ψ.λ. .

A.5.4. Flow charts

Παραθέτουμε εδώ flow charts για να διευκολύνουμε την κατανόηση της συλλογιστικής που ακολουθήσαμε στην κατασκευή των προγραμμάτων μας.

Flow Chart Αλγορίθμου Γινομένου



Flow Chart Αλγορίθμου Πηλίκου

B. Zernike Moments

B.1. Γενικά για τα Zernike Moments (ZMs):

Αν και έχει περάσει αρκετός καιρός από τότε που ο [2] εισήγαγε τα ορθογωνικά σημεία στην επεξεργασία εικόνας, το επιστημονικό ενδιαφέρον στη χρήση σημείων στη μηχανική ακόμα αυξάνεται. Αυτό οφείλεται στο μοναδικό τους τρόπο να περιγράφουν ένα σήμα, γι'αυτό χρησιμοποιούνται σα διακριτικά χαρακτηριστικά στις εφαρμογές αναπαράστασης εικόνας και αναγνώρισης προτύπων.

Τελευταία οι επιστήμονες συγκεντρώνουν την προσοχή τους σε ερευνητικά πεδία σχετικά με τα ορθογωνικά σημεία. Πολλοί ερευνητές ανά τον κόσμο παλεύουν να αναπτύξουν γρήγορους αλγόριθμους για να εοιταχύνουν το χρόνο υπολογισμού των ορθογωνικών σημείων και να κάνουν τις εγφαρμογές του hardware εύκολες. Η πλειοψηφία των γρήγορων αλγορίθμων που έχουν παρουσιαστεί μέχρι σήμερα, κάνουν χρήση αναδρομικών εξισώσεων που επιτρέπουν τον υπολογισμό πυρήνων σημείων υψηλής τάξης, κάνοντας χρήση πυρήνων χαμηλότερων τάξεων. Αυτή η μεθοδολογία είναι ακριβής, ειδικά όταν ένα ολόκληρο σύνολο σημείων είναι απαραίτητο να υπολογιστεί.

Επιπρόσθετα από τότε που τα ορθογωνικά σημεία έχουν σα συναρτήσεις βάσης ορθογωνικά πολυώνυμα, τα οποία συνθέτουν μία ορθογωνική βάση, είναι πολύ χρήσιμα για να αναπαραστήσουν κατά μοναδικό τρόπο πρότυπα στην αναγνώριση προτύπων. Για αυτές τις εφαρμογές κάποιες ιδιαιτέρως ελπιδοφόρες προσπάθειες [23 - 24] έγιναν για να ενσωματώσουν την αμεταβλητότητα σε κλίμακα, περιστροφή, και «μετάφραση» στον υπολογισμό των ορθογωνικών σημείων.

Πρόσφατα, τα μειονεκτήματα των ορθογωνικών σημείων που έχουν ορθογωνικά πολυώνυμα σαν συναρτήσεις βάσης μελετήθηκαν και πιο αριθμητικά ακριβή ορθογωνικά σημεία, με διακριτά σημεία [18 - 20] χρησιμοποιήθηκαν σαν εναλλακτικες για τα συνεχή. Ωστόσο, αν και μια σύντομη μελέτη πάνω στην ακρίβεια των ορθογωνικών σημείων σε σχέση με προσεγγιστικά λάθη, εξαιτίας της μετατροπής

από το συνεχή χρόνο στο διακριτό [17] και την ακρίβεια των σημείων στις μετατροπές κλίμακας, περιστροφής και «μετάφρασης» [27 - 28] έγινε, εξέχουσα σημασία έχουν τα σφάλματα τα οφειλόμενα στην πεπερασμένη ακρίβεια ειδικά στους γρήγορους αναδρομικούς αλγορίθμους, εφόσον ένα σφάλμα σε κάποιο βήμα του αλγορίθμου μπορεί να συσσωρευτεί από τη μία επανάληψη στην άλλη, καταλήγοντας σε αποτελέσματα αναξιόπιστα.

Τα Τα Zernike Moments(ZMs) είναι η πιο ευρέως χρησιμοποιούμενη οικογένεια ορθογωνικών σημείων, εξαιτίας

Α. Των ιδιοτήτων τους να είναι αμετάβλητα στην τυχαία περιστροφή του αντικειμένου που πεγιγράφουν και

Β. Στο ότι εξασφαλίζουν τέλεια ανασύνθεση της εικόνας από τα σημεία της.

Επίσης, τα ZMs αποτελούν ιδιαιτέρως χρήσιμα εργαλεία στον τομέα της αναγνώρισης προτύπων και στην ανάλυση εικόνας εξαιτίας:

Α. Της ορθογωνιότητάς τους και

Β. Της αμεταβλητότητάς τους με την περιστροφή.

Βέβαια, ο ευθύς υπολογισμός τους είναι πολύ «ακριβός» και χρονοβόρος περιορίζοντας τη χρήση τους ειδικά σε υψηλές τάξεις.

Η εισαγωγή στα ZMs έγινε από τον Teague [2], ο οποίος χρησιμοποίησε ένα σύνολο σύνθετων πολυωνύμων που σχηματίζουν ένα ολόκληρο ορθογωνικό σύνολο στο εσωτερικό του μοναδιαίου κύκλου για τον οποίο ισχύει:

$$x^2 + y^2 = 1$$

Η μορφή των πολυωνύμων αυτών είναι:

$$V_{pq}(x,y) = V_{pq}(r,\theta) = R_{pq}(r) \cdot \exp(jq\theta)$$

Όπου p: μη αρνητικός ακέραιος

q: θετικός και αρνητικός ακέραιος που υπακούουν στους περιορισμούς:

$$p-|q| : \text{περιττός} \quad \text{και} \quad |q| \le p$$

r: μήκος του διανύσματος από την αρχή $(\bar{x}, \bar{y})$ έως το pixel (x, y) και

θ: το τόξο μεταξύ του διανύσματος r και του άξονα των x υπό την ανθωρολογιακή φορά.

Τα $R_{pq}(r,\theta)$ είναι τα ακτινικά πολυώνυμα Zernike στις (r,θ) πολικές συντεταγμένες που ορίζονται ως:

$$R_{pq}(r) = \sum_{s=0}^{\frac{p-|q|}{2}} (-1)^s \cdot \frac{(p-s)!}{s!\cdot\left(\frac{p+|q|}{2}-s\right)!\cdot\left(\frac{p-|q|}{2}-s\right)!} \cdot r^{p-2s}$$

Προσοχή: Ισχύει η σχέση: $R_{p,-q}(r)=R_{pq}(r)$.

Η αρχή στην οποία υπακούουν τα ορθογωνικά πολυώνυμα είναι η αρχή της ορθογωνιότητας, σύμφωνα με την οποία:

$$\iint_{x^2+y^2\le 1} V_{nm}^*(x,y)\cdot V_{pq}(X,y)dx\cdot dy = \frac{\pi}{n+1}\cdot\delta_{np}\cdot\delta_{mq}$$

Όπου $\delta_{\alpha\beta}=1$ όταν $\alpha=\beta$ και

$\delta_{\alpha\beta}=0$ σε καθε άλλη περίπτωση.(Το $\delta_{\alpha\beta}$ είναι το σύμβολο Kronecker)

Το ZM τάξης p με q επαναλήψεις για μια συνεχή συνάρτηση εικόνας f(x,y) που εξαφανίζεται έξω από το μοναδιαίο κύκλο:

$$Z_{pq} = \frac{p+1}{\pi}\cdot \iint_{x^2+y^2\le 1} f(x,y)\cdot V_{pq}^*(r,\theta)\cdot r\cdot dr\cdot d\theta$$

Για μια ψηφιακή εικόνα, τα ολοκληρώματα αντικαθίστανται από αθροίσματα [9 - 10] ώστε να μας δώσουν:

$$Z_{pq} = \frac{p+1}{\pi}\cdot \sum_x \sum_y f(x,y)\cdot V_{pq}^*(r,\theta)\ ,\ \ \text{όπου}\ \ x^2+y^2\le 1$$

Ας υποθέσουμε ότι κάποιος ξέρει όλα τα σημεία Zpq της f(x,y) μέχρι μια μέγιστη τάξη p_{max}. Είναι επιθυμητό να ανακατασκευάσουμε μία διακριτή συνάρτηση $\hat{f}(x,y)$ τα σημεία της οποίας ταιριάζουν ακριβώς με εκείνα της f(x,y) μέχρι μια συγκεκριμένη μέγιστη τάξη p_{max}. Τα ZMs είναι οι συντελεστές που προκύπτουν από την επέκταση μιας εικόνας σε ορθογωνικά πολυώνυμα Zernike όπως δείχνει η ακόλουθη εξίσωση ανασύνθεσης:

$$\hat{f}(x,y) = \sum_{p=0}^{p_{\max}} \sum_{q} Z_{pq} \cdot V_{pq}(r,\theta)$$

με τους περιορισμούς που αναφέραμε παραπάνω για το q. Να σημειωθεί επίσης ότι καθώς το p_{max} τείνει στο άπειρο, τόσο το $\hat{f}(x,y)$ θα πλησιάζει το $f(x,y)$. Η συγκεκριμένη μέθοδος προσδιορισμού των ZMs είναι γνωστή ως Ευθεία μέθοδος. Είναι προφανές πως οι τύποι που παραθέσαμε περιέχουν υπολογισμούς πολλών παραγοντικών, λειτουργίες που καταναλώνουν πολύ χρόνο υπολογισμών. Γι'αυτό και ξεκινήσε να διευρύνεται η χρήση αναδρομικών μεθόδων για τα ακτινικά πολυώνυμα. Στη συνέχεια θα παραθέσουμε τους ευρύτερα χρησιμοποιούμενους αλγορίθμους.

Η χρονική πολυπλοκότητα των ZMs όλων των τάξεων που είναι μικρότερες της pmax, όπου p_{max} η μέγιστη τάξη των σημείων, είναι $O(N^2 \ {p_{max}}^3)$. Η χρονική πολυπλοκότητα ενός ακτινικού πολυωνύμου είναι $O(p_{max})$ και για αυτό η χρονική πολυπλοκότητα των σημείων Zernike είναι για όλα τα $0 \le p \le p_{max}$ και $|q| \le p$ είναι $O(N^2 \ {p_{max}}^2 \times p_{max}) = O(N^2 \ {p_{max}}^3)$. Αυτή η τάξη είναι πολύ μεγάλη όταν τα N και p_{max} είναι πολύ μεγάλα. Γίνονται προσπάθειες να μειωθεί η χρονική πολυπλοκότητα και να κατέβει στο O(1). Τα πολυώνυμα Zernike αποτελούνται από δύο κομμάτια: τα ακτινικά πολυώνυμα $R_{pq}(r)$ και την έκφραση $e^{jq\theta}$. Εκτεταμμένη έρευνα έχει πραγματοποιηθεί τα τελευταία χρόνια για βελτίωση τη χρονικής πολυπλοκότητας των ακτινικών αυτών πολυωνύμων. Παρακάτω παραθέτουμε τις πιο συχνά χρησιμοποιούμενες αναδρομικές μεθόδους.

B.1.A.i. Αλγόριθμος Kintner

Πρώτος ο Kintner μελέτησε τις ιδιότητες των πολυωνύμων Zernike και εισήγαγε μία επαναλαμβανόμενη σχέση [1]. Στη συνέχεια ο αναδρομικός αλγόριθμος για τον υπολογισμό των πολυωνύμων Zernike που προτάθηκε από τον Kintner περιγράφεται:

Αλγόριθμος:

- p=q → Ευθεία μέθοδος και χρήση της εξίσωσης:

$$R_{pq}(r) = \sum_{s=0}^{\frac{p-|q|}{2}} (-1)^s \cdot \frac{(p-s)!}{s!\cdot\left(\frac{p+|q|}{2}-s\right)!\cdot\left(\frac{p-|q|}{2}-s\right)!} \cdot r^{p-2s}$$

- p-q=2 → (όμοια)Ευθεία μέθοδος και χρήση της εξίσωσης

$$R_{pq}(r) = \sum_{s=0}^{\frac{p-|q|}{2}} (-1)^s \cdot \frac{(p-s)!}{s!\cdot\left(\frac{p+|q|}{2}-s\right)!\cdot\left(\frac{p-|q|}{2}-s\right)!} \cdot r^{p-2s}$$

- σε κάθε άλλη περίπτωση →

$$R_{pq}(r) = \frac{(K_2 \cdot r^2 + K_3) \cdot R_{p-2,q}(r) + K_4 \cdot R_{p-4,q}(r)}{K_1}$$

όπου $K_1 = \frac{(p+q)\cdot(p-q)\cdot(p-2)}{2}$,

$K_2 = 2\cdot p\cdot(p-1)\cdot(p-2)$,

$K_3 = -q^2\cdot(p-1) - p\cdot(p-1)\cdot(p-2)$,

$$K_4 = \frac{-p \cdot (p+q-2) \cdot (p-q-2)}{2}$$

Όπως φαίνεται στις ανωτέρω εξισώσεις, ο αλγόριθμος του Kintner δεν μπορεί να εφαρμοστεί σε περιπτώσεις όπου p=q ή p-q=2, οπότε και χρησιμοποιείται στις περιπτώσεις αυτές η ευθεία μέθοδος. Αυτός ο αλγόριθμος είναι ταχύτερος από την ευθεία μέθοδο αλλά και αυτός περιλαμβάνει παραγοντικούς υπολογισμούς που αντιστοιχούν σε αρκετό υπολογιστικό χρόνο.

Ας σημειωθεί ότι ο δείκτης q είναι σταθερός και ο δείκτης p ποικίλει και παίρνει τιμές p = q + 4, q + 6,…, p_{max}. Αυτή η μορφή αναδρομής στην οποία τα πολυώνυμα υψηλότερης τάξης p υπολογίζονται για ένα συγκεκριμένο δείκτη αναδρομής q έχει ένα σαφές πλεονέλτημα στην ταχύτητα του υπολογισμού των ZMs επειδή μπορούμε να χρησιμοποιήσουμε την ίδια τιμή της έκφρασης $e^{-jq\theta}$ για ένα δεδομένο q αλλά διαφορετικές τιμές του p.

Η τάξη της χρονικής πολυπλοκότητας της μεθόδου του Kintner για τον υπολογισμό των πολυωνύμων όλων των τάξεων και επαναλήψεων είναι $O(p_{max}^2)$. Εφόσον η ευθεία μέθοδος χρησιμοποιείται για να υπολογίσει τα $R_{qq}(r)$, $R_{q+2,q}(r)$ για ένα δεδομένο δείκτη επανάληψης q, η μέθοδος είναι αργή.

B.1.A.ii. Τροποποιημένη Μέθοδος Kintner

Ο Chong [23] τροποποίησε τη μέθοδο Kintner που αποφεύγει την ευθεία εξίσωση για τον υπολογισμό των ακτινικών πολυωνύμων $R_{qq}(r)$ και $R_{q+2,p}(r)$ για ένα δεδομένο q. Πιο συγκεκριμένα, όσον αφορά την τροποποιημένη μέθοδο Kintner για q= 0, 1, 2,…, p_{max}:

- $R_{qq}(r) = r^q$, για q = 0, 1, 2,..., p_{max}
- $R_{p,q}(r) = p \cdot R_{pp}(r) - (p-1) \cdot R_{p-2,p-2}(r)$, για q = 0, 1, 2, …, p_{max}-2 , (p = q + 2)

- $R_{pq}(r) = \dfrac{(K_2 \cdot r^2 + K_3) \cdot R_{p-2,q}(r) + K_4 \cdot R_{p-4,q}(r)}{K_1}$, για p =q+4, q+6,…,p_{max}

Με τις ίδιες τιμές για τους συντελεστές K που είχαμε στον αλγόριθμο Kintner που αναφέραμε παραπάνω. Η χρονική πολυπλοκότητα της τροποποιημένης μεθόδου Kintner για τον υπολογισμό όλων των πολυωνύμων, κάθε τάξης και κάθε επανάληψης μέχρι την pmax είναι $O(p_{max}^2)$. Εφόσον δεν περιλαμβάνει τον υπολογισμό ευθέων εξισώσεων, είναι ταχύτερη από την αρχική μέθοδο Kintner.

Β.1.Β. Μέθοδος Q-recursive

Ο Chong εισήγαγε έναν γρήγορο αλγόριθμο για να υπολογίζει όλα τα πολυώνυμα διαφόρων τιμών δείκτη επανάληψης q που δε βασίζεται σε πολυώνυμα χαμηλότερων τάξεων. Η μέθοδος αυτή είναι η:

- $R_{pp}(r) = r^p$, για p = 0, 1, 2,..., p_{max}
- $R_{pq}(r) = p \cdot R_{pp}(r) - (p-1) \cdot R_{p-2,p-2}(r),\ p = 2,3,..., p_{\max},\ (q = p-2)$
- $R_{pq}(r) = K_1 \cdot R_{p,q+4}(r) + \left(K_2 + \dfrac{K_3}{r^2} \right) \cdot R_{p,q+2}(r), p = 4,5,..., p_{\max}, (q = p-4, p-6,...,0 \,\acute{\eta}\, 1)$

όπου

$$K_1 = \frac{q \cdot (q-1)}{2} - q \cdot K_2 + \frac{K_3 \cdot (p+q+2) \cdot (p-q)}{8}$$

$$K_2 = \frac{K_3 \cdot (p-q+2) \cdot (p+q)}{4 \cdot (q-1)} + (q-2) \text{ και}$$

$$K_3 = \frac{-4 \cdot (q-2) \cdot (q-3)}{(p+q-2) \cdot (p-q+4)}$$

Το πλεονέκτημα της μεθόδου αυτής είναι το ότι όταν τα σημεία μόνο μιας συγκεκριμένης τάξης p με διάφορες τιμές του δείκτη επανάληψης q απαιτούνται, τα πολυώνυμα χαμηλότερης τάξης δε χρειάζονται, πράγμα που δεν εφαρμόζεται στις υπόλοιπες μεθόδους που αναφέραμε μέχρι στιγμής.

B.1.C. Αλγόριθμος Prata

Ο Αλγόριθμος Prata χρησιμοποιεί πολυώνυμα χαμηλότερης τάξης $R_{p-1,q-1}(r)$ και $R_{p-2,q}(r)$ για να παράγει ένα πολυώνυμο $R_{pq}(r)$, υψηλότερης τάξης.Ωστόσο χρησιμοποιεί την ευθεία μέθοδο για τον υπολογισμό του $R_{pq}(r)$ για q=0 και p≠q και τη σχέση $R_{pq}(r)=r^p$ για p=q. Πιο συγκεκριμένα ο αλγόριθμος δίνεται ως εξής:

- $R_{qq}(r) = r^q$, για q = 0, 1, 2,..., p_{max}
- $R_{pq}(r) = K_1 \cdot r \cdot R_{p-1,q-1}(r) + K_2 \cdot R_{p-2,q}(r),\ \forall\ p\ και\ q, p \neq q,\ q \neq 0$

Όπου $K_1 = \frac{2 \cdot p}{p+q}, \quad K_2 = -\frac{p-q}{p+q}$

Η συμβολή της ευθείας εξίσωσης στον υπολογισμό του $R_{p0}(r)$, όπου p≠0, η τάξη της χρονικής πολυπλοκότητας της μεθόδου Prata των πολυωνύμων για όλες τις τάξεις είναι $O(p_{max}^3)$.

B.1.D. Τροποποιημένη Μέθοδος Prata

Συγκρίνοντας όλες τις επαναλαμβανόμενες σχέσεις, είναι εμφανές πως η επαναληπτική σχέση Prata περιέχει το μικρότερο αριθμό ενεργειών αν παραβλέψουμε τις ενέργειες που απαιτούνται για τον υπολογισμό του $R_{p0}(r)$. Η μέθοδος Prata περιλαμβάνει μόλις μία πρόσθεση και τρεις πολλαπλασιασμούς σε σχέση με τις δύο προσθέσεις και τους πέντε πολλαπλασιασμούς της Kintner και της τροποποιημένης Kintner και δύο πρσθέσεις και τέσσερις πολλαπλασιασμούς της q-recursive. Επίσης,

ο προσδιορισμός των συντελεστών K1 και K2 της Prata περιλαμβάνει πολύ μικρότερο αριθμό υπολογισμών σε σύγκριση με τον αριθμό των ενεργειών που περιλαμβάνουν οι άλλες τρεις μέθοδοι. Ο λόγος που περιλαμβάνει την ευθεία εξίσωση για τον υπολογισμό των πουλωνύμων $R_{p0}(r)$ είναι ότι η επαναλαμβανόμενη σχέση περιέχει τα πολυώνυμα $R_{p-1,-1}(r)$ για τον υπολογισμό των πολυωνύμων $R_{p0}(r)$, p=2, 4, …, p_{max}. Εφόσον ξέρουμε ότι $R_{p,q}(r)= R_{p,-q}(r)$ χρησιμοποιούμε την ιδιότητα των ακτινικών πολυωνύμων για να αποφύγουμε τον ευθύ υπολογισμό των πολυωνύμων $R_{p0}(r)$, p=2, 4, …, p_{max}. Έχουν προταθεί δύο φόρμες για τη μέθοδο Prata. [34]

$1^{η}$.

- $R_{pp}(r) = r^p$, για p = 0, 1, 2,..., p_{max}
- $R_{q+p,q}(r) = K_1 \cdot r \cdot R_{p+q-1,|q-1|}(r) + K_2 \cdot R_{p+q-2,q}(r), \quad p = 2,4,..., p_{\max}, \ (q = 0,1,2,..., p_{\max} - p)$

Με τις τιμές των K_1 και K_2 να παραμένουν ίδιες με τις αναφερθείσες στην απλή μέθοδο Prata.

$2^{η}$.

- $R_{pp}(r) = r^p$, για p = 0, 1, 2,..., p_{max}
- $R_{pq}(r) = p \cdot R_{pp}(r) - (p-1) \cdot R_{p-2,p-2}(r), \ p = 0,1,2,..., p_{\max} - 2, \ (p = q+2)$
- $R_{q+p,q}(r) = K_1 \cdot r \cdot R_{p+q-1,|q-1|}(r) + K_2 \cdot R_{p+q-2,q}(r), \quad p = 4,6,..., p_{\max}, \ (q = 0,1,2,..., p_{\max} - p)$

Η $2^{η}$ φόρμα είναι υπολογιστικά πιο αποδοτική.

Η χρονική πολυπλοκότητα της τροποποιημένης μεθόδου Prata είναι $O(p_{max}^2)$ για όλα τα πολυώνυμα για τάξεις μικρότερες ή ίσες της p_{max} σε σύγκριση με την πολυπλοκότητα $O(p_{max}^3)$ της μεθόδου Prata.

Β.1.Ε. Προτεινόμενη Ταχεία Μέθοδος Kintner

Η τροποποιημένη επαναλαμβανόμενη σχέση Kintner μπορεί να γίνει ταχύτερη υπολογιστικά επαναπροσδιορίζοντας του συντελεστές K_2, K_3 και K_4 στην επεναλαμβανόμενη σχέση. Αυτή αναφέρεται σαν την ταχεία μέθοδο Kintner.

$$R_{pq}(r) = (K_2^{'} \cdot r^2 + K_3^{'}) \cdot R_{p-2,q}(r) + K_4^{'} \cdot R_{p-4,q}(r), \qquad p = q+4, q+6, \ldots, p_{\max}$$

Όσον αφορά τους συντελεστές στην περίπτωση αυτή, έχω:

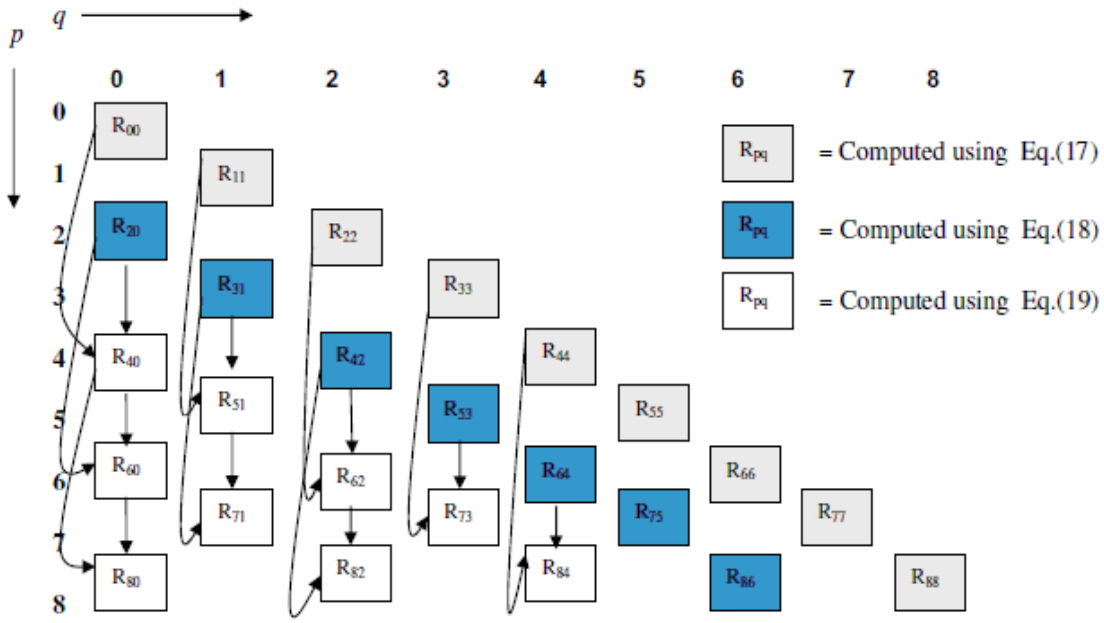
$$K_2^{'} = K_2 / K_1 , \qquad K_3^{'} = K_3 / K_1 , \qquad K_4^{'} = K_4 / K_1$$

Β.2. Περιγραφή αλγορίθμων για υλοποίηση των ZMs (Kintner's, Prata's q-recursive)

Στο σημείο αυτό θα παραθέσουμε μία περιγραφή της υλοποίησης των διαφόρων αλγορίθμων που υλοποιούν τα ZMs και με τη βοήθεια της θεωρίας και των μεθόδων που αναπτύξαμε παραπάνω θα προχωρήσουμε στην παρατήρηση των δ.ψ.λ. σε κάθε σημείο της υλοποίησής μας. Δηλαδή, πώς παρατηρήσαμε τα δ.ψ.λ. καθώς κινούμασταν μεταξύ των δεικτών τάξης και επανάληψης κάθε συντελεστή Zernike μέχρι να φτάσουμε στο p_{max} το οποίο απαιτούνταν κάθε φορά. Θα παραθέσουμε και εποπτικούς πίνακες [34] οι οποίοι δείχνουν ποιους συντελεστές Zernike χρειαζόμαστε κάθε φορά για να προσδιορίσουμε το ZM που επιθυμούμε.

Β.2.Α. Αναδρομική Τροποποιημένη Μέθοδος Kintner & Ταχεία Μέθοδος Kintner

Όπως φαίνεται στο Σχήμα Β.1. για την τροποποιημένη αναδρομική μέθοδο Kintner, για τον προσδιορισμό κάθε συντελεστή Zernike χρειάζεται να χρησιμοποιήσουμε στοιχεία της ίδιας επανάληψης(δείκτης q), με τους δύο αμέσως μεγαλύτερους δείκτες p, δηλαδή με τις δύο αμέσως μεγαλύτερες τάξεις. Για παράδειγμα για τον προσδιορισμό του συντελεστή R_{62}, θα χρειαστώ τα στοιχεία R_{22} και R_{42}. Τα δύο αυτά στοιχεία ανήκουν στην ίδια επανάληψη αλλά σε διαφορετικές τάξεις. Αυτό γίνεται σαφέστερο στο σχήμα που παραθέτουμε παρακάτω.



Σχήμα Β.1. Η πορεία της αναδρομικής μεθόδου Kintner

Όπως φαίνεται και στο σχήμα τα στοιχεία της εξωτερικής διαγωνίου δε χρησιμοποιούν κάποια άλλα στοιχεία για να προσδιοριστούν, πράγμα που γίνεται κατανοητό αν κανείς παρατηρήσει και τη μέθοδο Kintner όπως την παραθέσαμε παραπάνω. Πιο συγκεκριμένα, από τον τύπο:

$$R_{qq}(r) = r^q, \text{ για } q = 0, 1, 2,..., p_{max}$$

τα στοιχεία της διαγωνίου για να προσδιοριστούν χρειάζονται μόνο την τιμή της ακτίνας r και την τιμή της επανάληψης. Τα στοιχεία της εσωτερικής διαγωνίου, στα οποία η τιμή του δείκτη q της επανάληψης είναι κατά 2 μικρότερος του δείκτη p της τάξης υπολογίζονται από τον τύπο :

$$R_{p,q}(r) = p \cdot R_{pp}(r) - (p-1) \cdot R_{p-2,p-2}(r) \text{ , για } q = 0, 1, 2, \ldots, p_{max}-2 \text{ , } (p = q + 2)$$

που δείχνει πως για να προσδιορίσουμε τους συντελεστές αυτούς χρειαζόμαστε τα στοιχεία της εξωτερικής διαγωνίου τα οποία έχουν σαν δείκτες (p=q εφόσον ανήκουν στην εξωτερική διαγώνιο) την τιμή της τάξης και της αμέσως προηγούμενης τάξης. Δηλαδή, τους R_{pp} και τους $R_{p-2,p-2}$. Για τον προσδιορισμό των υπολοίπων στοιχείων

θα κάνουμε χρήση του παρακάτω τύπου μέσω του οποίου επιβεβαιώνεται αυτό που φαίνεται και στο σχήμα που παραθέτουμε, ότι δηλαδή, θα χρειαστούμε κάποιους συντελεστές, τους K_1, K_2, K_3 και K_4 καθώς επίσης και τις τιμές των συντελεστών Zernike για τη συγκεκριμένη τιμή του δείκτη επενάληψης και για κατά 2 και κατά 4 μικρότερο συντελεστή τάξης, δηλαδή, για p'=p-2 και p''=p-4, τους δύο αμέσως προηγούμενους συντελεστές με ίδια επανάληψη, δηλαδή. Αυτό διαφαίνεται από τον τύπο:

$$R_{pq}(r) = \frac{(K_2 \cdot r^2 + K_3) \cdot R_{p-2,q}(r) + K_4 \cdot R_{p-4,q}(r)}{K_1} \text{ , για p =q+4, q+6,…,}p_{max}$$

καθώς και από το σχήμα όπου κάθε εσωτερικό στοιχείο χρησιμοποιεί τα αμέσως 2 προηγούμενα στοιχεία στον κατακόρυφο άξονα.

Στο σημείο αυτό και με βάση τη θεωρία που περιγράψαμε στην προηγούμενη ενότητα καθώς και τα όσα αναφέραμε μόλις θα περιγράψουμε τον αλγόριθμο που κατασκευάσαμε για τη μέθοδο Kintner.

Β.2.Α.i. Περιγραφή Αλγορίθμου

main function:

Καταστρώνουμε αρχικά τον τριγωνικό πίνακα με τους συντελεστές των ZMs που θα μας χρειαστούν. Το δοθέν σχήμα μας βοηθάει να αντιληφθούμε πώς θα είναι πιο εύκολο να ξεκινήσουμε τους υπολογισμούς μας ώστε να μην παρουσιαστεί καμιά έλλειψη στον προσδιορισμό του εκάστοτε συντελεστή Zernike.

Πιο συγκεκριμένα, είναι προφανές πως αρχικά θα προσδιορίσουμε τα στοιχεία της εξωτερικής διαγωνίου τα οποία δεν προαπαιτούν τη γνώση κάποιων πληροφοριών πέρα από την τιμή του δείκτη επανάληψης και την τιμή του r την οποία όμως έχουμε στις δηλώσεις μας ως global καθορίσει . Οπότε με ένα for loop καλούμε τη συνάρτηση Zernike_sunarthsh_d, με εισόδους τις τιμές του δείκτη επανάληψης και

τάξης οι οποίοι ταυτίζονται και είναι ίσοι με q, ενώ ζητάμε να μας επιστρέψει τους συντελεστές Zernike, σε float και double μορφή για κάθε στοιχείο της διαγωνίου. Μετά το πέρας του for loop αυτού έχουμε συμπληρωμένα τα διαγώνια στοιχεία μας και σταδιακά θα προχωρήσουμε και στα υπόλοιπα.

Έπειτα θα «σκαννάρουμε» ολόκληρο τον πίνακα συντελεστών Zernike από την εσωτερική διαγώνιο και μετά λαμβάνοντας φυσικά υπόψην μας τα απαραίτητα κάθε φορά στοιχεία για τον εκάστοτε προσδιορισμό. Η φορά που θα ακολουθήσουμε θα είναι από πάνω προς τα κάτω και από τα δεξιά προς τα αριστερά, πράγμα που μας εξασφαλίζουν τα δύο φωλιασμένα for loops που ακολουθούν, κατά τα οποία ξεκινώντας από την τάξη 2 και αυξάνοντας κατά ένα την τάξη και ξεκινώντας από το κατά δύο μικρότερο στην επανάληψη από το δείκτη τάξης στοιχείο και μειώνοντας κατά 2 το συντελεστή μας προχωράμε οριζόντια κατά μήκος μιας γραμμής του πίνακα. Εξασφαλίζουμε με τον τρόπο αυτό την εκτέλεση πράξεων με στοιχεία γνωστά.

Τέλος, πάλι μέσω ενός διπλού φωλιασμένου for loop τυπώνουμε τα στοιχεία του πίνακά μας από το 0 έως το στοιχείο μέγιστης τάξης δίνοντας και πληροφορίες για τα λανθασμένα ψηφία (δ.ψ.λ.) κατά τη float και double εκτέλεση των πράξεων(εννοούμε την εκτέλεση πράξεων με αριθμούς float και double αντιστοίχως).

B.2.A.ii. Περιγραφή Συναρτήσεων

Zernike_sunarthsh_d function

Η συνάρτηση αυτή δέχεται σαν εισόδους τους δείκτες για την τάξη και την επανάληψη (p και q αντίστοιχα) και επιστρέφει σε πίνακες τις τιμές των πολυωνύμων R για float και double μορφή. Όπως είδαμε για τα R χρειαζόμαστε μόνο τις τιμές τάξης, επανάληψης και ακτίνας r και εφόσον αυτά τα έχουμε και είναι γνωστά είμαστε έτοιμοι να τα υπολογίσουμε.

Αρχικά, για τον double αριθμό r(ας τον ονομάσουμε dr για να είναι σαφέστερη η αναφορά σε αυτόν κάθε φορά) θα κρατήσουμε και τη float μορφή του σε μια νέα μεταβλητή την fr(μέσω της εντολής fr=(float)dr). Επίσης, θα ονομάσουμε d τη διαφορά μεταξύ των p και q, μέσω της οποίας θα εκτελείται το αντίστοχο κάθε φορά με την τιμή αυτής τμήμα εντολών στα πλαίσια μιας εντολής switch. Προφανώς, η switch θα δίνει στη d τη δυνατότητα να πάρει τιμή ίση με 0, με 2 ή κάποια άλλη, τηρώντας βέβαια πάντα τα p και q τους περιορισμούς από τον ορισμό των ZMs.

→Για d=0, δηλαδή όταν p=q, η συνάρτησή μας θα υπολογίζει το R_{qq} όπως αναφέρθηκε στην περιγραφή του αλγορίθμου του Kintner μέσω της σχέσης r^q. Βέβαια, είναι ευκολότερο να κάνουμε τους υπολογισμούς μας μέσω της σχέσης

$$R_{qq}(r) = r^q = e^{q\cdot\log(r)}.$$

Πρακτικά το σκεπτικό μας είναι σε κάθε βήμα μας κατά την εκτέλεση των πράξεων να ελέγχουμε την πορεία των λανθασμένων ψηφίων μας. Δηλαδή, πότε αυξάνονται, πότε μειώνονται, πότε δεν παρατηρείται μεταβολή και να αποδίδουμε κάθε φορά σε κάθε πράξη την αντίστοιχη μεταβολή ή μη. Όπως θα φανεί και στη συνέχεια πολλές φορές κρίνουμε ασήμαντη την επίδραση ορισμένων πράξεων στη μεταβολή των δ.ψ.λ. γι'αυτό και επιλέγουμε να μην ασχοληθούμε με τον προσδιορισμό τους στο συγκεκριμένο σημείο. Στα υπόλοιπα σημεία που η εμπειρία μας και η ενασχόλησή μας με τους μηχανισμούς δημιουργίας του f.p.e. μαρτυρούν την ενδεχόμενη μόλυνση των αριθμών μας με δ.ψ.λ. καλούμε τη συνάρτηση εύρεσης του πλήθους των λανθασμένων ψηφίων, τα οποία και στο τέλος καταχωρούμε σε έναν πίνακα ώστε να μπορέσουμε σε περίπτωση που θέλουμε να τα παρατηρήσουμε. Τέτοια σημεία, είναι τα σημεία στα οποία πραγματοποιούνται πράξεις μεταξύ δεκαδικών αριθμών, λογαρίθμηση δεκαδικών ή ύψωση σε εκθέτη. Για τον υπολογισμό του R_{qq} με τον τύπο που δείξαμε παραπάνω τεμαχίσαμε το αποτέλεσμα σε διάφορα στάδια ώστε να μπορούμε να παρατηρούμε ταυτόχρονα το σφάλμα. Στην τελική πράξη, που δίνει τις τιμές του R_{qq} (σε float και double) αποθηκεύουμε και σε μια καινούρια μεταβλητή τα λάθος ψηφία καθώς επίσης και τις τιμές του R ώστε να τα τυπώσω και να τα επιστρέψω τελικά στο πρόγραμμά μου. Αφού εκτελέσω τις πράξεις μου αυτές μέσω της break βγαίνω από τη συνάρτηση αυτή.

→ για d=2, το σκεπτικό είναι παρόμοιο και πάλι και κάνω τις πράξεις μου σε διάφορα βήματα σταδιακά. Θεωρούμε και πάλι ότι το f.p.e. μολύνει τα αποτελέσματά μας όταν υπεισέρχονται πράξεις με δεκαδικούς αριθμούς και όχι σε γινόμενα π.χ. ακεραίων ή πράξεις αυστηρά ακεραίων. Μόλις προσδιορίσουμε το R_{pq} στην περίπτωση αυτή και πάλι επιστρέφουμε τη double και float τιμή αυτου σε πίνακες, τυπώνουμε το πλήθος των δ.ψ.λ. και πάλι βγαίνουμε απότομα από τη συνάρτησή μας μέσω μιας break.

→ σε κάθε άλλη περίπτωση τιμής του d, αρχικά υπολογίζω τις τιμές των συντελεστών K, στις οποίες προφανώς θεωρούμε πως δεν υπάρχει f.p.e. τέτοιο ώστε να επηρεάσει ουσιαστικά τα αποτελέσματά μας μολύνοντάς τα με σφάλμα. Προχωράμε πάλι σε βήματα υπολογίζοντας κάθε φορά το υπεισερχόμενο σφάλμα στις πράξεις μας, χρησιμοποιώντας και τους συντελεστές K και όταν τελικά υπολογίσουμε το R σε float και double μορφή πάλι προσδιορίζουμε το λάθος, καταχωρούμε τα κάθος ψηφία στον αντίστοιχο πίνακα και βγαίνουμε μέσω μιας break από τη συνάρτησή μας.

Epistrofh_plhthous_dde function

Προφανώς αυτή η συνάρτηση θα αναλάβει να προσδιορίσει τα δ.ψ.λ. και να τα επιστρέψει στο πρόγραμμά μας σε έναν πίνακα ώστε να κάνει εύκολη την παρατήρησή τους.

Οι είσοδοι της συνάρτησης αυτής είναι οι δύο αριθμοί double και float καθώς και ο ακέραιος αριθμός του μεγέθους των float για το σύστημά μας και οι έξοδοι είναι μέσω δεικτών η mantissa και ο εκθέτης της διαφοράς.

Για τις δύο εισόδους μας αρχικά καλούμε τις συναρτήσεις epistrofh_float_mantissa_ektheth και epistrofh_double_mantissa_ektheth, συναρτήσεις παρόμοιες με αυτές που χρησιμοποιήσαμε στο Α μέρος της εργασίας μας οι οποίες επιστρέφουν τη mantissa και τον εκθέτη των αριθμών που τους εισάγουμε. Κρατάμε τη διαφορά του float από τη float εκδοχή του double αριθμού σε μια νέα μεταβλητή(diafora) και στη συνέχεια βρίσκουμε και το μέγιστο εκθέτη των

δύο αριθμών(θέλοντας να συμπεριλάβουμε και τις περιπτώσεις όπου ο ένας εκ των δύο έχει π.χ. την τιμή 0.9999999 και ο άλλος την τιμή 1.00000000000001). Αν η διαφορά των δύο αριθμών είναι 0 τότε επιστρέφουμε την τιμή 0 στα λάθος ψηφία . Αν η τιμή της diafora είναι διάφορη του 0 τότε καλούμε την epistrofh_float_mantissa με είσοδο τη diafora των δύο αριθμών, ώστε να απομονώσουμε εκθέτη και mantissa της diafora και προσδιορίζουμε τη βύθισή σαν τη διαφορά του μέγιστου εκθέτη από τον εκθέτη της diafora. Αν η βύθιση είναι μεγαλύτερη από την τιμή του αριθμού n, που αντιστοιχεί στο πλήθος των δεκαδικών ψηφίων ενός float, σημαίνει πως πρακτικά δεν επηρεάζεται το αποτέλεσμά μας από το f.p.e. και επιστρέφει μηδενικά λανθασμένα ψηφία. Αν η βύθιση είναι μικρότερη, τα λάθος ψηφία προσδιορίζονται από τη σχέση: λάθος_ψηφία=n+1-βύθιση. Και τέλος επιστρέφω τα λάθος ψηφία που προέκυψαν από τη σχέση αυτή.

Epistrofh_float_mantissa_ektheth function

Στόχος αυτής της συνάρτησης είναι να επιστρέψει σε δείκτες την τιμή της mantissa και του εκθέτη του αριθμού μας. Ο αριθμός μας είναι float.

Αρχικά με τη βοήθεια της sprintf μεταφέρω τον αριθμό μου σε έναν πίνακα και στη συνέχεια απομονώνω τη mantissa από τον εκθέτη μου. Αυτή η διαδικασία υλοποιείται με ένα loop που διαβάζει τα στοιχεία του πίνακά μου μέχρι να βρει το e, οπότε και κρατάει τη θέση του e. Τα στοιχεία μέχρι και πριν το e τα καταχωρεί σε ένα νέο πίνακα, τον πίνακα της mantissa, και τα στοιχεία από το e και μετά τα χησιμοποιεί ώστε να κατασκευάσει πίνακα και για τον εκθέτη.

Παρατήρηση: Για να διαβάσουμε τα στοιχεία του πίνακα, εφόσον πρόκειται για αριθμό float αρχικά, τα όρια που θα χρησιμοποιήσω στο for loop μου είναι προσαρμοσμένα στη μορφή του αριθμού μου,δηλαδή από 0 έως 14. Επίσης, στην sprintf που χρησιμοποιήσαμε χρησιμοποιήσαμε format 10.7 και πάλι λαμβάνοντας υπόψη τη μορφή του αριθμού μας.

Στη συνέχεια, χρησιμοποιώντας τα στοιχεία των πινάκων και με χρήση της atof και της atoi μετατρέπω τον char πίνακα της mantissa και του εκθέτη μου σε float αριθμό και ακέραιο αριθμό αντίστοιχα. Τέλος, κρατάω σε ένα δείκτη τον ακέραιο

αριθμό της mantissa και του εκθέτη του αριθμού μου (*num_mantissa_num, *num_ektheths_num) ώστε να μπορώ τελικά να τα μεταφέρω στο πρόγραμμά μου.

Epistrofh_double_mantissa_ektheth function

Στόχος αυτής της συνάρτησης είναι να επιστρέψει σε δείκτες την τιμή της mantissa και του εκθέτη του αριθμού μας. Ο αριθμός μας είναι double.

Χρησιμοποιείται διαδικασία αντίστοιχη με αυτή της προηγούμενης συνάρτησης, αλλάζοντας τα μεγέθη των πινάκων μου ώστε να αντιστοιχούν σε double μορφή αριθμού καθώς επίσης και τα όρια στα for loops ώστε να μπορώ να πραγματοποιώ έλεγχο κατά μήκος ολόκληρου του πίνακά μου. Δηλαδή, το loop μου ξεκινάει από 0 έως 24. Και το format μου διαφοροποιείται στο sprintf ώστε να ανταποκρίνεται στη double μορφή του αριθμού μου. Διαφορά επίσης έχω και στη μετατροπή του πίνακα mantissa σε double αριθμό, όπου αντί για χρήση της συνάρτησης atof που χρησιμοποιήσαμε τη συνάρτηση our_atod όπου πρακτικά μετατρέπω έναν πίνακα char σε double αριθμό.

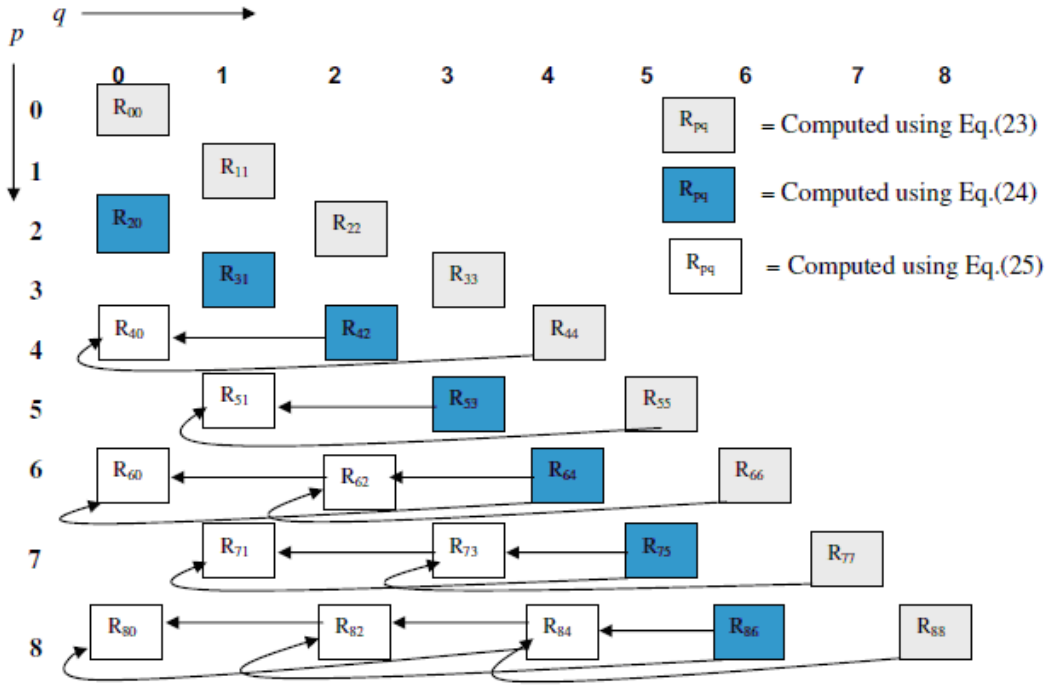
Δηλαδή, στόχος και πάλι αυτής της συνάρτησης είναι να επιστρέψει σε δείκτες την τιμή της mantissa και του εκθέτη του αριθμού μας.

Αρχικά με τη βοήθεια της sprintf μεταφέρω τον αριθμό μου σε έναν πίνακα και στη συνέχεια απομονώνω τη mantissa από τον εκθέτη μου. Αυτή γίνεται με ένα loop που διαβάζει τα στοιχεία του πίνακά μου μέχρι να βρει το e, οπότε και κρατάει τη θέση του e και τα στοιχεία μέχρι και πριν αυτό τα καταχωρεί σε ένα νέο πίνακα, τον πίνακα της mantissa, και τα στοιχεία από το e και μετά τα χησιμοποιεί ώστε να κατασκευάσει πίνακα χαρακτήρων και για τον εκθέτη. Στη συνέχεια, χρησιμοποιώντας τα στοιχεία των πινάκων και με χρήση της our_atod και της atoi μετατρέπω τον char πίνακα της mantissa και του εκθέτη μου σε double αριθμό και ακέραιο αριθμό αντίστοιχα. Τέλος, καρτάω σε ένα δείκτη τον ακέραιο αριθμό της mantissa και του εκθέτη του αριθμού μου (*num_mantissa_num, *num_ektheths_num) ώστε να μπορώ τελικά να τα μεταφέρω στο πρόγραμμά μου.

Όσον αφορά την ταχεία τροποποιημένη μέθοδο Kintner, τόσο το σχήμα όσο και η διαδικασία και οι συναρτήσεις ακολουθούν την ίδια φιλοσοφία και πορεία με μόνη διαφορά τον υπολογισμό των συντελεστών K, οι οποίοι όπως δείξαμε στο θεωρητικό μέρος (B.1) προκύπτουν από την διαίρεση των συντελεστών K_2, K_3, K_4 με το K_1 εξ'αρχής, γεγονός το οποίο επιδρά στις πράξεις μας και πράγματι δίνει ταχύτερα αποτελέσματα από την τροποποιημένη μέθοδο Kintner.

B.2.B. Q-recursive Μέθοδος

Όπως φαίνεται στο Σχήμα B.2. για την q-recursive μέθοδο, για τον προσδιορισμό κάθε συντελεστή Zernike χρειάζεται να χρησιμοποιήσουμε στοιχεία της ίδιας τάξης(δείκτης p), με τους δύο αμέσως μεγαλύτερους δείκτες q, δηλαδή με τις δύο αμέσως μεγαλύτερες επαναλήψεις. Για παράδειγμα για τον προσδιορισμό του συντελεστή R_{62}, θα χρειαστώ τα στοιχεία R_{66} και R_{64}. Τα δύο αυτά στοιχεία ανήκουν στην ίδια τάξη αλλά σε διαφορετικές επαναλήψεις. Αυτό γίνεται σαφέστερο στο σχήμα που παραθέτουμε παρακάτω.



Σχήμα B.2. Η πορεία της μεθόδου q-recursive

Όπως φαίνεται και στο σχήμα τα στοιχεία της εξωτερικής διαγωνίου δε χρησιμοποιούν κάποια άλλα στοιχεία για να προσδιοριστούν, πράγμα που γίνεται κατανοητό αν κανείς παρατηρήσει και τη μέθοδο q-recursive όπως την παραθέσαμε παραπάνω. Πιο συγκεκριμένα, από τον τύπο:

$$R_{pp}(r) = r^p, \text{ για } p = 0, 1, 2,..., p_{max}$$

τα στοιχεία της διαγωνίου για να προσδιοριστούν χρειάζονται μόνο την τιμή της ακτίνας r και την τιμή της τάξης. Τα στοιχεία της εσωτερικής διαγωνίου, στα οποία η τιμή του δείκτη q της επανάληψης είναι κατά 2 μικρότερος του δείκτη p της τάξης υπολογίζονται από τον τύπο :

$$R_{pq}(r) = p \cdot R_{pp}(r) - (p-1) \cdot R_{p-2,q-2}(r),\ p = 2,3,..., p_{\max},\ (q = p-2)$$

που δείχνει πως για να προσδιορίσουμε τους συντελεστές αυτούς χρειαζόμαστε τα στοιχεία της εξωτερικής διαγωνίου τα οποία έχουν σαν δείκτες (p=q εφόσον ανήκουν στην εξωτερική διαγώνιο) την τιμή της τάξης και της αμέσως προηγούμενης τάξης. Δηλαδή, τους R_{pp} και τους $R_{p-2,p-2}$. Για τον προσδιορισμό των υπολοίπων στοιχείων θα κάνουμε χρήση του παρακάτω τύπου μέσω του οποίου επιβεβαιώνεται αυτό που φαίνεται και στο σχήμα που παραθέτουμε, ότι δηλαδή, θα χρειαστούμε κάποιους συντελεστές, τους K_1, K_2, και K_3 καθώς επίσης και τις τιμές των συντελεστών Zernike για τη συγκεκριμένη τιμή του δείκτη τάξης και για κατά 2 και κατά 4 μικρότερο συντελεστή επανάληψης, δηλαδή, για q'=q-2 και q''=q-4, τους δύο αμέσως προηγούμενους συντελεστές με ίδια τάξη, δηλαδή. Αυτό διαφαίνεται από τον τύπο:

$$R_{pq}(r) = K_1 \cdot R_{p,q+4}(r) + \left(K_2 + \frac{K_3}{r^2} \right) \cdot R_{p,q+2}(r),\ p = 4,5,..., p_{\max},\ (q = p-4, p-6,...,0 \acute{\eta} 1)$$

καθώς και από το σχήμα όπου κάθε εσωτερικό στοιχείο χρησιμοποιεί τα αμέσως 2 προηγούμενα στοιχεία στον οριζόντιο άξονα.

Στο σημείο αυτό και με βάση τη θεωρία που περιγράψαμε παραπάνω καθώς και τα όσα αναφέραμε μόλις θα περιγράψουμε τον αλγόριθμο που κατασκευάσαμε για τη μέθοδο q-recursive.

B.2.B.i. Περιγραφή Αλγορίθμου

Το σκεπτικό μας είναι παρόμοιο με αυτό της μεθόδου Kintner και για αυτό δε θα αναλωθούμε στην εκ νέου υπερλεπτομερή περιγραφή των κοινών συναρτήσεων στις μεθόδους αυτές. Το ίδιο ισχύει και για τη μέθοδο Prata που θα περιγράψουμε παρακάτω.

main function:

Καταστρώνουμε αρχικά τον τριγωνικό πίνακα με τους συντελεστές των ZMs που θα μας χρειαστούν. Το δοθέν σχήμα μας βοηθάει να αντιληφθούμε πώς θα είναι πιο εύκολο να ξεκινήσουμε τους υπολογισμούς μας ώστε να μην παρουσιαστεί καμιά έλλειψη στον προσδιορισμό του εκάστοτε συντελεστή Zernike και να μη χρειάζεται να φτιάξουμε σύνθετα for loops.

Πιο συγκεκριμένα, είναι προφανές πως αρχικά θα προσδιορίσουμε τα στοιχεία της εξωτερικής διαγωνίου τα οποία δεν προαπαιτούν τη γνώση κάποιων πληροφοριών πέρα από την τιμή του δείκτη τάξης και την τιμή του r την οποία όμως έχουμε στις δηλώσεις μας ως global καθορίσει . Οπότε με ένα for loop καλούμε τη συνάρτηση Zernike_sunarthsh_d, με εισόδους τις τιμές του δείκτη επανάληψης και τάξης οι οποίοι ταυτίζονται και είναι ίσοι με p, ενώ ζητάμε να μας επιστρέψει τους συντελεστές Zernike, σε float και double μορφή για κάθε στοιχείο της διαγωνίου. Μετά το πέρας του for loop αυτού έχουμε συμπληρωμένα τα διαγώνια στοιχεία μας και σταδιακά θα προχωρήσουμε και στα υπόλοιπα.

Έπειτα θα «σκαννάρουμε» ολόκληρο τον πίνακα συντελεστών Zernike από την εσωτερική διαγώνιο και μετά λαμβάνοντας φυσικά υπόψην μας τα απαραίτητα κάθε φορά στοιχεία για τον εκάστοτε προσδιορισμό. Η φορά που θα ακολουθήσουμε

θα είναι από πάνω προς τα κάτω και από τα δεξιά προς τα αριστερά, πράγμα που μας εξασφαλίζουν τα δύο φωλιασμένα for loops που ακολουθούν, κατά τα οποία ξεκινώντας από την τάξη 2 και αυξάνοντας κατά ένα την τάξη και ξεκινώντας από το κατά δύο μικρότερο στην επανάληψη από το δείκτη τάξης στοιχείο και μειώνοντας κατά 2 το συντελεστή μας προχωράμε οριζόντια κατά μήκος μιας γραμμής του πίνακα συμπληρώνουμε τον πίνακά μας. Εξασφαλίζουμε με τον τρόπο αυτό την εκτέλεση πράξεων με στοιχεία γνωστά, άρα την ορθότητα των τιμών των συντελεστών των πολυωνύμων Zernike .

Τέλος, πάλι μέσω ενός διπλού φωλιασμένου for loop τυπώνουμε τα στοιχεία του πίνακά μας από το 0 έως το στοιχείο μέγιστης τάξης δίνοντας και πληροφορίες για τα λανθασμένα ψηφία (δ.ψ.λ.) κατά τη float και double εκτέλεση των πράξεων(εννοούμε την εκτέλεση πράξεων με αριθμούς float και double αντιστοίχως).

Β.2.Β.ii. Περιγραφή Συναρτήσεων

Zernike_sunarthsh_d function

Η συνάρτηση αυτή δέχεται σαν εισόδους τους δείκτες για την τάξη και την επανάληψη (p και q αντίστοιχα) και επιστρέφει σε πίνακες τις τιμές των πολυωνύμων R για float και double μορφή. Όπως είδαμε για τα R χρειαζόμαστε μόνο τις τιμές τάξης, επανάληψης και ακτίνας r και εφόσον αυτά τα έχουμε και είναι γνωστά είμαστε έτοιμοι να τα υπολογίσουμε.

Αρχικά, για τον double αριθμό r(ας τον ονομάσουμε dr για να είναι σαφέστερη η αναφορά σε αυτόν κάθε φορά) θα κρατήσουμε και τη float μορφή του σε μια νέα μεταβλητή την fr(μέσω της εντολής fr=(float)dr). Επίσης, θα ονομάσουμε d τη διαφορά μεταξύ των p και q, μέσω της οποίας θα εκτελείται το αντίστοχο κάθε φορά με την τιμή αυτής τμήμα εντολών στα πλαίσια μιας εντολής switch. Προφανώς, η switch θα δίνει στη d τη δυνατότητα να πάρει τιμή ίση με 0, με 2 ή κάποια άλλη, τηρώντας βέβαια πάντα τα p και q τους περιορισμούς από τον ορισμό των ZMs.

→Για d=0, δηλαδή όταν p=q, η συνάρτησή μας θα υπολογίζει το R_{pp} όπως αναφέρθηκε στην περιγραφή της μεθόδου q-recursive μέσω της σχέσης r^p. Βέβαια, είναι ευκολότερο να κάνουμε τους υπολογισμούς μας μέσω της σχέσης

$$R_{qq}(r) = r^q = e^{q \cdot \log(r)}.$$

Πρακτικά, το σκεπτικό μας είναι σε κάθε βήμα μας κατά την εκτέλεση των πράξεων να ελέγχουμε την πορεία των λανθασμένων ψηφίων μας. Δηλαδή, πότε αυξάνονται, πότε μειώνονται, πότε δεν παρατηρείται μεταβολή και να αποδίδουμε κάθε φορά σε κάθε πράξη την αντίστοιχη μεταβολή ή μη. Πολλές φορές, και έχοντας ήδη δουλέψει πάνω στο f.p.e. και στους μηχανισμούς δημιουργίας ή διάδοσής του, κρίνουμε ασήμαντη την επίδραση ορισμένων πράξεων στη μεταβολή των δ.ψ.λ. γι'αυτό και επιλέγουμε να μην ασχοληθούμε με τον προσδιορισμό τους σε κάποια σημεία. Στα υπόλοιπα σημεία που η εμπειρία μας και η ενασχόλησή μας με τους μηχανισμούς δημιουργίας του f.p.e. μαρτυρούν την ενδεχόμενη μόλυνση των αριθμών μας με δ.ψ.λ. καλούμε τη συνάρτηση εύρεσης του πλήθους των λανθασμένων ψηφίων, τα οποία και στο τέλος καταχωρούμε σε έναν πίνακα ώστε να μπορέσουμε σε περίπτωση που θέλουμε να τα παρατηρήσουμε. Τέτοια σημεία, είναι τα σημεία στα οποία πραγματοποιούνται πράξεις μεταξύ δεκαδικών αριθμών, λογαρίθμηση δεκαδικών ή ύψωση σε εκθέτη. Για τον υπολογισμό του R_{pp} με τον τύπο που δείξαμε παραπάνω τεμαχίσαμε το αποτέλεσμα σε διάφορα στάδια ώστε να μπορούμε να παρατηρούμε ταυτόχρονα το σφάλμα. Στην τελική πράξη, που δίνει τις τιμές του R_{pp} (σε float και double) αποθηκεύουμε και σε μια καινούρια μεταβλητή τα λάθος ψηφία καθώς επίσης και τις τιμές του R ώστε να τα τυπώσω και να τα επιστρέψω τελικά στο πρόγραμμά μου. Αφού εκτελέσω τις πράξεις μου αυτές μέσω της break βγαίνω από τη συνάρτηση αυτή.

→ για d=2, το σκεπτικό είναι παρόμοιο και πάλι και κάνω τις πράξεις μου σε διάφορα βήματα σταδιακά. Θεωρούμε και πάλι ότι το f.p.e. μολύνει τα αποτελέσματά μας όταν υπεισέρχονται πράξεις με δεκαδικούς αριθμούς και όχι σε γινόμενα π.χ. ακεραίων ή πράξεις αυστηρά ακεραίων. Μόλις προσδιορίσουμε το R_{pq} στην περίπτωση αυτή και πάλι επιστρέφουμε τη double και float τιμή αυτου σε πίνακες,

τυπώνουμε το πλήθος των δ.ψ.λ. και πάλι βγαίνουμε απότομα από τη συνάρτησή μας μέσω μιας break.

→ σε κάθε άλλη περίπτωση τιμής του d, αρχικά υπολογίζω τις τιμές των συντελεστών K, στις οποίες προφανώς θεωρούμε πως δεν υπάρχει f.p.e. τέτοιο ώστε να επηρεάσει ουσιαστικά τα αποτελέσματά μας μολύνοντάς τα με σφάλμα. Προχωράμε πάλι σε βήματα υπολογίζοντας κάθε φορά το υπεισερχόμενο σφάλμα στις πράξεις μας, χρησιμοποιώντας και τους συντελεστές K και όταν τελικά υπολογίσουμε το R σε float και double μορφή πάλι προσδιορίζουμε το λάθος, καταχωρούμε τα κάθος ψηφία στον αντίστοιχο πίνακα και βγαίνουμε μέσω μιας break από τη συνάρτησή μας.

Epistrofh_plhthous_dde function

Προφανώς αυτή η συνάρτηση θα αναλάβει να προσδιορίσει τα δ.ψ.λ. και να τα επιστρέψει στο πρόγραμμά μας σε έναν πίνακα ώστε να κάνει εύκολη την παρατήρησή τους.

Οι είσοδοι της συνάρτησης αυτής είναι οι δύο αριθμοί double και float καθώς και ο ακέραιος αριθμός του μεγέθους των float για το σύστημά μας και οι έξοδοι είναι μέσω δεικτών η mantissa και ο εκθέτης της διαφοράς.

Συνολικά το σκεπτικό μας είναι το εξής: Για την ίδια ποσότητα p έχουμε μια παράσταση float έστω fp και double έστω dp. Κρατάμε τα 7 πρώτα ψηφία της double μορφής, δηλαδή του dp, και τα ονομάζουμε fdp. Η ποσότητα αυτή προκύπτει από την εντολή: fdp=(float)dp. Εφαρμόζουμε τον ορισμό της βύθισης στα fp και fdp. Αν τα fdp είναι καθαρά τότε η σύγκριση των fdp με τα fp μέσω του λήμματος προσφέρει τον αρκιβή αριθμό δ.ψ.λ. του float. Υπάρχει το ενδεχόμενο η διαφορά τους να είναι μηδενική και να μην έχουμε δ.ψ.λ. και να επιστρέψουμε την τιμή 0 στο κυρίως πρόγραμμά μας, το ενδεχόμενο η βύθιση να περνάει πέρα από τα όρια του float και να επιστρέφουμε και πάλι την τιμή 0 στο κυρίως πρόγραμμά μας και το ενδεχόμενο η βύθιση να ανήκει στα όρια του float και να τη χρησιμοποιούμε ώστε να προσδιορίσουμε τα λανθασμένα ψηφία μέσω της σχέσης λάθος_ψηφία=n+1-βύθιση. Και τέλος επιστρέφουμε τα λάθος ψηφία που προέκυψαν από τη σχέση αυτή.

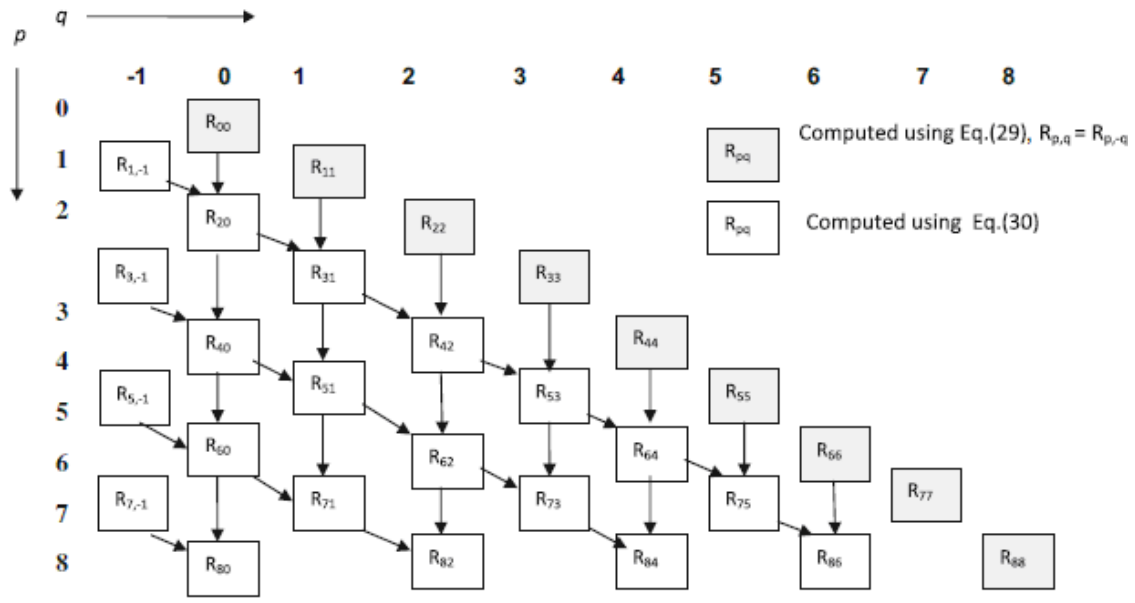
Ο τρόπος για να ξέρουμε πως τα 7 πρώτα ψηφία του fdp είναι καθαρά περιγράφεται σε αντίστοιχο χωρίο (Ερώτηση) στη συνάρτση εύρεσης πλήθους λάθος ψηφίων του μέρους Α.

Epistrofh_float_mantissa_ektheth function, Epistrofh_double_mantissa_ektheth function

Αυτές οι δύο συναρτήσεις ουσιατικά αναλαμβάνουν να παραλάβουν έναν αριθμό που τους εισάγουμε σαν είσοδο και να απομονώσουν τον εκθέτη και τη mantissa του. Ο ένας αναλαμβάνει float και ο άλλος double αριθμούς να επεξεργαστεί. Επειδή έχουμε ήδη περιγράψει σε προηγούμενους αλγορίθμους τη λειτουργία αυτών των συναρτήσεων τόσο εδώ όσο και στον αλγόριθμο του Prata θα αποφύγουμε να αναλωθούμε εκ νέου σε περιγραφή.

Β.2.C. Τροποποιημένη Μέθοδος Prata

Όπως φαίνεται στο Σχήμα Β.3. για την τροποποιημένη μέθοδο Prata, για τον προσδιορισμό κάθε συντελεστή Zernike χρειάζεται να χρησιμοποιήσουμε δύο στοιχεία. Το ένα εξ'αυτών είναι της ίδιας επανάληψης (ίδιος δείκτης q) και της αμέσως προηγούμενης τάξης(δηλαδή τιμής δείκτη τάξης κατά 2 μικρότερος) και το άλλο να είναι το αμέσως προηγούμενο στη διαγώνιο στην οποία ανήκει ο συντελεστής αυτός Zernike. Δηλαδή το στοιχείο εκείνο για το οποίο η τιμή του δείκτη επανάληψης και τάξης να είναι κατά ένα μικρότερος από αυτούς. Για παράδειγμα για τον προσδιορισμό του συντελεστή R_{62}, θα χρειαστώ τα στοιχεία R_{42} και R_{51}. Αυτό γίνεται σαφέστερο στο σχήμα που παραθέτουμε παρακάτω.



Σχήμα Β.3. Η πορεία της τροποποιημένης μεθόδου Prata

Επιλέξαμε να υλοποιήσουμε και να παραθέσουμε τη δεύτερη μορφή της μεθόδου Prata που είχε προταθεί [34] ακριβώς επειδή, είναι όπως αναφέρθηκε αποδοτικότερη υπολογιστικά από την πρώτη.

Όπως φαίνεται και στο σχήμα τα στοιχεία της εξωτερικής διαγωνίου δε χρησιμοποιούν κάποια άλλα στοιχεία για να προσδιοριστούν, πράγμα που γίνεται κατανοητό αν κανείς παρατηρήσει και τη μέθοδο q-recursive όπως την παραθέσαμε παραπάνω. Πιο συγκεκριμένα, από τον τύπο:

$$R_{pp}(r) = r^p, \text{ για } p = 0, 1, 2, ..., p_{max}$$

τα στοιχεία της διαγωνίου για να προσδιοριστούν χρειάζονται μόνο την τιμή της ακτίνας r και την τιμή της τάξης. Τα στοιχεία της εσωτερικής διαγωνίου, στα οποία η τιμή του δείκτη q της επανάληψης είναι κατά 2 μικρότερος του δείκτη p της τάξης υπολογίζονται από τον τύπο :

$$R_{pq}(r) = p \cdot R_{pp}(r) - (p-1) \cdot R_{p-2,p-2}(r), \; p = 0,1,2,..., p_{\max} - 2, \; (p = q+2)$$

που δείχνει πως για να προσδιορίσουμε τους συντελεστές αυτούς χρειαζόμαστε τα στοιχεία της εξωτερικής διαγωνίου τα οποία έχουν σαν δείκτες (p=q εφόσον ανήκουν στην εξωτερική διαγώνιο) την τιμή της τάξης και της αμέσως προηγούμενης τάξης. Δηλαδή, τους R_{pp} και τους $R_{p-2,p-2}$. Για τον προσδιορισμό των υπολοίπων στοιχείων θα κάνουμε χρήση του παρακάτω τύπου μέσω του οποίου επιβεβαιώνεται αυτό που φαίνεται και στο σχήμα που παραθέτουμε, ότι δηλαδή, θα χρειαστούμε κάποιους συντελεστές, τους K_1 και K_2 καθώς επίσης και τις τιμές των συντελεστών Zernike $R_{p+q-1,|q-1|}$ και $R_{p+q-2,q}$, δηλαδή, για p'=p+q-1 και q'=|q-1| και p''=p+q-2 και q''=q. Αυτό διαφαίνεται από τον τύπο:

$$R_{q+p,q}(r) = K_1 \cdot r \cdot R_{p+q-1,|q-1|}(r) + K_2 \cdot R_{p+q-2,q}(r),\ p = 4,6,..., p_{\max}, (q = 0,1,2,..., p_{\max} - p)$$

Στο σημείο αυτό και με βάση τη θεωρία που περιγράψαμε στην προηγούμενη ενότητα καθώς και τα όσα αναφέραμε μόλις θα περιγράψουμε τον αλγόριθμο που κατασκευάσαμε για τη μέθοδο Prata.

B.2.C.i. Περιγραφή Αλγορίθμου

main function:

Καταστρώνουμε αρχικά τον τριγωνικό πίνακα με τους συντελεστές των ZMs που θα μας χρειαστούν. Το δοθέν σχήμα μας βοηθάει να αντιληφθούμε πώς θα είναι πιο εύκολο να ξεκινήσουμε τους υπολογισμούς μας ώστε να μην παρουσιαστεί καμιά έλλειψη στον προσδιορισμό του εκάστοτε συντελεστή Zernike και να μη χρειάζεται να φτιάξουμε σύνθετα for loops.

Πιο συγκεκριμένα, είναι προφανές πως αρχικά θα προσδιορίσουμε τα στοιχεία της εξωτερικής διαγωνίου τα οποία δεν προαπαιτούν τη γνώση κάποιων πληροφοριών πέρα από την τιμή του δείκτη τάξης και την τιμή του r την οποία όμως έχουμε στις δηλώσεις μας ως global καθορίσει . Οπότε με ένα for loop καλούμε τη συνάρτηση Zernike_sunarthsh_d, με εισόδους τις τιμές του δείκτη επανάληψης και τάξης οι

οποίοι ταυτίζονται και είναι ίσοι με p, ενώ ζητάμε να μας επιστρέψει τους συντελεστές Zernike, σε float και double μορφή για κάθε στοιχείο της διαγωνίου. Μετά το πέρας του for loop αυτού έχουμε συμπληρωμένα τα διαγώνια στοιχεία μας και σταδιακά θα προχωρήσουμε και στα υπόλοιπα.

Έπειτα θα «σκαννάρουμε» ολόκληρο τον πίνακα συντελεστών Zernike από την εσωτερική διαγώνιο και μετά λαμβάνοντας φυσικά υπόψην μας τα απαραίτητα κάθε φορά στοιχεία για τον εκάστοτε προσδιορισμό. Η φορά που θα ακολουθήσουμε θα είναι από πάνω προς τα κάτω και από τα δεξιά προς τα αριστερά, πράγμα που μας εξασφαλίζουν τα δύο φωλιασμένα for loops που ακολουθούν, κατά τα οποία ξεκινώντας από την τάξη 2 και αυξάνοντας κατά ένα την τάξη και ξεκινώντας από το κατά δύο μικρότερο στην επανάληψη από το δείκτη τάξης στοιχείο και μειώνοντας κατά 2 το συντελεστή μας προχωράμε οριζόντια κατά μήκος μιας γραμμής του πίνακα συμπληρώνουμε τον πίνακά μας. Εξασφαλίζουμε με τον τρόπο αυτό την εκτέλεση πράξεων με στοιχεία γνωστά, άρα την ορθότητα των τιμών των συντελεστών των πολυωνύμων Zernike .

Τέλος, πάλι μέσω ενός διπλού φωλιασμένου for loop τυπώνουμε τα στοιχεία του πίνακά μας από το 0 έως το στοιχείο μέγιστης τάξης δίνοντας και πληροφορίες για τα λανθασμένα ψηφία (δ.ψ.λ.) κατά τη float και double εκτέλεση των πράξεων(εννοούμε την εκτέλεση πράξεων με αριθμούς float και double αντιστοίχως).

B.2.C.ii. Περιγραφή Συναρτήσεων

Zernike_sunarthsh_d function

Η συνάρτηση αυτή δέχεται σαν εισόδους τους δείκτες για την τάξη και την επανάληψη (p και q αντίστοιχα) και επιστρέφει σε πίνακες τις τιμές των πολυωνύμων R για float και double μορφή. Όπως είδαμε για τα R χρειαζόμαστε μόνο τις τιμές τάξης, επανάληψης και ακτίνας r και εφόσον αυτά τα έχουμε και είναι γνωστά είμαστε έτοιμοι να τα υπολογίσουμε.

Αρχικά, για τον double αριθμό r(ας τον ονομάσουμε dr για να είναι σαφέστερη η αναφορά σε αυτόν κάθε φορά) θα κρατήσουμε και τη float μορφή του σε μια νέα μεταβλητή την fr(μέσω της εντολής fr=(float)dr). Επίσης, θα ονομάσουμε d τη διαφορά μεταξύ των p και q, μέσω της οποίας θα εκτελείται το αντίστοχο κάθε φορά με την τιμή αυτής τμήμα εντολών στα πλαίσια μιας εντολής switch. Προφανώς, η switch θα δίνει στη d τη δυνατότητα να πάρει τιμή ίση με 0, με 2 ή κάποια άλλη, τηρώντας βέβαια πάντα τα p και q τους περιορισμούς από τον ορισμό των ZMs.

→Για d=0, δηλαδή όταν p=q, η συνάρτησή μας θα υπολογίζει το R_{pp} όπως αναφέρθηκε στην περιγραφή της μεθόδου q-recursive μέσω της σχέσης r^p. Βέβαια, είναι ευκολότερο να κάνουμε τους υπολογισμούς μας μέσω της σχέσης

$$R_{qq}(r) = r^q = e^{q\cdot\log(r)}.$$

Πρακτικά, το σκεπτικό μας είναι σε κάθε βήμα μας κατά την εκτέλεση των πράξεων να ελέγχουμε την πορεία των λανθασμένων ψηφίων μας. Πολλές φορές, και έχοντας ήδη δουλέψει πάνω στο f.p.e. και στους μηχανισμούς δημιουργίας ή διάδοσής του, κρίνουμε ασήμαντη την επίδραση ορισμένων πράξεων στη μεταβολή των δ.ψ.λ. γι'αυτό και επιλέγουμε να μην ασχοληθούμε με τον προσδιορισμό τους σε κάποια σημεία. Στα υπόλοιπα σημεία, καλούμε τη συνάρτηση εύρεσης του πλήθους των λανθασμένων ψηφίων, τα οποία και στο τέλος καταχωρούμε σε έναν πίνακα ώστε να μπορέσουμε σε περίπτωση που θέλουμε να τα παρατηρήσουμε Για τον υπολογισμό του R_{pp} με τον τύπο που δείξαμε παραπάνω τεμαχίσαμε το αποτέλεσμα σε διάφορα στάδια ώστε να μπορούμε να παρατηρούμε ταυτόχρονα το σφάλμα. Στην τελική πράξη, που δίνει τις τιμές του R_{pp} (σε float και double) αποθηκεύουμε και σε μια καινούρια μεταβλητή τα λάθος ψηφία καθώς επίσης και τις τιμές του R ώστε να τα τυπώσω και να τα επιστρέψω τελικά στο πρόγραμμά μου. Αφού εκτελέσω τις πράξεις μου αυτές μέσω της break βγαίνω από τη συνάρτηση αυτή.

→ για d=2, το σκεπτικό είναι παρόμοιο και πάλι και κάνω τις πράξεις μου σε διάφορα βήματα σταδιακά. Θεωρούμε και πάλι ότι το f.p.e. μολύνει τα αποτελέσματά μας όταν υπεισέρχονται πράξεις με δεκαδικούς αριθμούς και όχι σε γινόμενα π.χ. ακεραίων ή πράξεις αυστηρά ακεραίων. Μόλις προσδιορίσουμε το R_{pq} στην

περίπτωση αυτή και πάλι επιστρέφουμε τη double και float τιμή αυτου σε πίνακες, τυπώνουμε το πλήθος των δ.ψ.λ. και πάλι βγαίνουμε απότομα από τη συνάρτησή μας μέσω μιας break.

→ σε κάθε άλλη περίπτωση τιμής του d, αρχικά υπολογίζω τις τιμές των συντελεστών K, στις οποίες προφανώς θεωρούμε πως δεν υπάρχει f.p.e. τέτοιο ώστε να επηρεάσει ουσιαστικά τα αποτελέσματά μας μολύνοντάς τα με σφάλμα. Προχωράμε πάλι σε βήματα υπολογίζοντας κάθε φορά το υπεισερχόμενο σφάλμα στις πράξεις μας, χρησιμοποιώντας και τους συντελεστές K και όταν τελικά υπολογίσουμε το R σε float και double μορφή πάλι προσδιορίζουμε το λάθος, καταχωρούμε τα κάθος ψηφία στον αντίστοιχο πίνακα και βγαίνουμε μέσω μιας break από τη συνάρτησή μας.

Epistrofh_plhthous_dde function

Κατά παρόμοιο τρόπο και σκεπτικό με τις προαναρθείσες συναρτήσεις εύρεσης του πλήθους των λάθος ψηφίων, αυτή η συνάρτηση θα αναλάβει να προσδιορίσει τα δ.ψ.λ. και να τα επιστρέψει στο πρόγραμμά μας σε έναν πίνακα ώστε να κάνει εύκολη την παρατήρησή τους.

Οι είσοδοι της συνάρτησης αυτής είναι οι δύο αριθμοί double και float καθώς και ο ακέραιος αριθμός του μεγέθους των float για το σύστημά μας και οι έξοδοι είναι μέσω δεικτών η mantissa και ο εκθέτης της διαφοράς.

Συνολικά το σκεπτικό μας είναι το εξής: Για την ίδια ποσότητα p έχουμε μια παράσταση float έστω fp και double έστω dp. Κρατάμε τα 7 πρώτα ψηφία της double μορφής, δηλαδή του dp, και τα ονομάζουμε fdp. Η ποσότητα αυτή προκύπτει από την εντολή: fdp=(float)dp. Εφαρμόζουμε τον ορισμό της βύθισης στα fp και fdp. Αν τα fdp είναι καθαρά τότε η σύγκριση των fdp με τα fp μέσω του λήμματος προσφέρει τον αρκιβή αριθμό δ.ψ.λ. του float. Υπάρχει το ενδεχόμενο η διαφορά τους να είναι μηδενική και να μην έχουμε δ.ψ.λ. και να επιστρέψουμε την τιμή 0 στο κυρίως πρόγραμμά μας, το ενδεχόμενο η βύθιση να περνάει πέρα από τα όρια του float και να επιστρέφουμε και πάλι την τιμή 0 στο κυρίως πρόγραμμά μας και το ενδεχόμενο η βύθιση να ανήκει στα όρια του float και να τη χρησιμοποιούμε ώστε να

προσδιορίσουμε τα λανθασμένα ψηφία μέσω της σχέσης λάθος_ψηφία=n+1-βύθιση. Και τέλος επιστρέφουμε τα λάθος ψηφία που προέκυψαν από τη σχέση αυτή.

Ο τρόπος για να ξέρουμε πως τα 7 πρώτα ψηφία του fdp είναι καθαρά περιγράφεται σε αντίστοιχο χωρίο (Ερώτηση) στη συνάρτση εύρεσης πλήθους λάθος ψηφίων του μέρους Α.

Epistrofh_float_mantissa_ektheth function, Epistrofh_double_mantissa_ektheth function

Αυτές οι δύο συναρτήσεις ουσιατικά αναλαμβάνουν να παραλάβουν έναν αριθμό που τους εισάγουμε σαν είσοδο και να απομονώσουν τον εκθέτη και τη mantissa του. Ο ένας αναλαμβάνει float και ο άλλος double αριθμούς να επεξεργαστεί. Επειδή έχουμε ήδη περιγράψει σε προηγούμενους αλγορίθμους τη λειτουργία αυτών των συναρτήσεων και όπως και στη μέθοδο q-recursive θα αποφύγουμε να αναλωθούμε εκ νέου σε περιγραφή.

Β.3. Συγκεντρωτικοί Πίνακες Δ.Ψ.Λ.

Στο σημείο αυτό, και έχοντας περιγράψει τους τρεις αλγορίθμους μας για την υλοποίηση των ZMs θα παραθέσουμε για τάξη 30 τους πίνακες των συντελεστών των πολυωνύμων Zernike που πήραμε καθώς και το υπεισερχόμενο σφάλμα.

Β.3.Α. Συγκεντρωτικός Πίνακας Αλγορίθμου Kintner

Λανθασμένα Δεκαδικά ψηφία Συντελεστών πολυωνύμων Zernike (p,q,λανθασμένα ψηφία)							
0 0 0	10 10 1	14 6 3	18 18 2	20 0 3	23 11 4	25 3 2	28 28 2
1 1 0	10 8 3	14 4 3	18 16 3	21 21 2	23 9 4	25 1 1	28 26 4
2 2 0	10 6 3	14 2 2	18 14 3	21 19 3	23 7 2	26 26 2	28 24 4
2 0 0	10 4 3	14 0 1	18 12 3	21 17 3	23 5 2	26 24 3	28 22 4
3 3 0	10 2 2	15 15 2	18 10 3	21 15 3	23 3 2	26 22 4	28 20 4
3 1 2	10 0 2	15 13 3	18 8 3	21 13 3	23 1 2	26 20 3	28 18 3
4 4 1	11 11 1	15 11 3	18 6 3	21 11 4	24 24 2	26 18 4	28 16 3
4 2 2	11 9 3	15 9 3	18 4 3	21 9 4	24 22 3	26 16 4	28 14 3
4 0 1	11 7 3	15 7 3	18 2 2	21 7 3	24 20 3	26 14 3	28 12 3
5 5 1	11 5 2	15 5 2	18 0 2	21 5 3	24 18 3	26 12 3	28 10 3
5 3 2	11 3 2	15 3 2	19 19 2	21 3 3	24 16 4	26 10 3	28 8 2
5 1 2	11 1 2	15 1 2	19 17 3	21 1 3	24 14 5	26 8 2	28 6 1
6 6 1	12 12 2	16 16 2	19 15 3	22 22 2	24 12 4	26 6 2	28 4 1
6 4 2	12 10 3	16 14 3	19 13 3	22 20 3	24 10 3	26 4 2	28 2 2
6 2 2	12 8 3	16 12 3	19 11 3	22 18 3	24 8 3	26 2 2	28 0 2
6 0 2	12 6 3	16 10 3	19 9 3	22 16 3	24 6 2	26 0 2	29 29 2
7 7 1	12 4 3	16 8 3	19 7 3	22 14 3	24 4 2	27 27 2	29 27 4
7 5 2	12 2 2	16 6 3	19 5 3	22 12 4	24 2 2	27 25 3	29 25 4
7 3 2	12 0 1	16 4 3	19 3 3	22 10 5	24 0 2	27 23 4	29 23 5
7 1 2	13 13 2	16 2 2	19 1 3	22 8 4	25 25 2	27 21 4	29 21 5
8 8 1	13 11 3	16 0 2	20 20 2	22 6 3	25 23 4	27 19 5	29 19 3
8 6 3	13 9 3	17 17 2	20 18 3	22 4 2	25 21 4	27 17 4	29 17 3
8 4 3	13 7 3	17 15 3	20 16 3	22 2 2	25 19 3	27 15 3	29 15 3
8 2 2	13 5 2	17 13 3	20 14 3	22 0 2	25 17 4	27 13 3	29 13 3
8 0 2	13 3 2	17 11 3	20 12 3	23 23 2	25 15 4	27 11 3	29 11 2
9 9 1	13 1 2	17 9 3	20 10 3	23 21 3	25 13 3	27 9 3	29 9 2
9 7 2	14 14 2	17 7 3	20 8 4	23 19 3	25 11 3	27 7 2	29 7 0
9 5 2	14 12 3	17 5 2	20 6 4	23 17 3	25 9 3	27 5 1	29 5 0
9 3 2	14 10 3	17 3 2	20 4 5	23 15 3	25 7 2	27 3 2	29 3 1
9 1 2	14 8 3	17 1 2	20 2 3	23 13 4	25 5 2	27 1 0	29 1 1

Β.3.Β. Συγκεντρωτικός Πίνακας Μεθόδου Q-recursive

Λανθασμένα Δεκαδικά ψηφία Συντελεστών πολυωνύμων Zernike (p,q,λανθασμένα ψηφία)							
0 0 0	10 10 0	14 6 2	18 18 1	20 0 4	23 11 4	25 3 2	28 28 1
1 1 0	10 8 1	14 4 2	18 16 2	21 21 1	23 9 4	25 1 2	28 26 1
2 2 0	10 6 2	14 2 2	18 14 1	21 19 1	23 7 3	26 26 1	28 24 2
2 0 0	10 4 2	14 0 2	18 12 2	21 17 0	23 5 3	26 24 2	28 22 2
3 3 0	10 2 2	15 15 1	18 10 2	21 15 2	23 3 3	26 22 2	28 20 3
3 1 1	10 0 2	15 13 0	18 8 2	21 13 2	23 1 3	26 20 3	28 18 2
4 4 0	11 11 0	15 11 2	18 6 3	21 11 3	24 24 1	26 18 3	28 16 2
4 2 1	11 9 1	15 9 2	18 4 3	21 9 3	24 22 2	26 16 3	28 14 2
4 0 1	11 7 1	15 7 1	18 2 3	21 7 3	24 20 2	26 14 2	28 12 2
5 5 0	11 5 1	15 5 2	18 0 3	21 5 3	24 18 1	26 12 2	28 10 3
5 3 1	11 3 1	15 3 2	19 19 1	21 3 2	24 16 3	26 10 2	28 8 3
5 1 1	11 1 1	15 1 2	19 17 2	21 1 2	24 14 4	26 8 2	28 6 3
6 6 0	12 12 1	16 16 1	19 15 2	22 22 1	24 12 3	26 6 2	28 4 3
6 4 0	12 10 0	16 14 1	19 13 2	22 20 1	24 10 2	26 4 2	28 2 3
6 2 1	12 8 2	16 12 2	19 11 2	22 18 2	24 8 2	26 2 2	28 0 3
6 0 1	12 6 2	16 10 2	19 9 2	22 16 3	24 6 2	26 0 2	29 29 1
7 7 0	12 4 2	16 8 2	19 7 2	22 14 3	24 4 2	27 27 1	29 27 1
7 5 0	12 2 2	16 6 2	19 5 3	22 12 4	24 2 2	27 25 1	29 25 2
7 3 1	12 0 2	16 4 2	19 3 3	22 10 5	24 0 2	27 23 2	29 23 3
7 1 1	13 13 1	16 2 2	19 1 3	22 8 4	25 25 1	27 21 3	29 21 4
8 8 0	13 11 1	16 0 2	20 20 1	22 6 4	25 23 0	27 19 5	29 19 3
8 6 1	13 9 2	17 17 1	20 18 1	22 4 3	25 21 2	27 17 4	29 17 3
8 4 0	13 7 2	17 15 1	20 16 2	22 2 3	25 19 2	27 15 3	29 15 3
8 2 1	13 5 2	17 13 1	20 14 3	22 0 3	25 17 3	27 13 3	29 13 3
8 0 1	13 3 2	17 11 1	20 12 3	23 23 1	25 15 3	27 11 3	29 11 3
9 9 0	13 1 2	17 9 1	20 10 3	23 21 2	25 13 2	27 9 3	29 9 3
9 7 0	14 14 1	17 7 2	20 8 4	23 19 3	25 11 2	27 7 3	29 7 3
9 5 0	14 12 1	17 5 2	20 6 4	23 17 3	25 9 2	27 5 3	29 5 3
9 3 1	14 10 2	17 3 2	20 4 6	23 15 3	25 7 2	27 3 3	29 3 3
9 1 1	14 8 2	17 1 2	20 2 4	23 13 4	25 5 2	27 1 3	29 1 3

Β.3.C. Συγκεντρωτικός Πίνακας Αλγορίθμου Prata

Λανθασμένα Δεκαδικά ψηφία Συντελεστών πολυωνύμων Zernike (p,q,λανθασμένα ψηφία)							
0 0 0	10 10 1	14 6 2	18 18 2	20 0 2	23 11 5	25 3 2	28 28 2
1 1 0	10 8 2	14 4 3	18 16 3	21 21 2	23 9 5	25 1 2	28 26 2
2 2 0	10 6 2	14 2 2	18 14 3	21 19 2	23 7 4	26 26 2	28 24 3
2 0 0	10 4 2	14 0 2	18 12 2	21 17 3	23 5 3	26 24 3	28 22 3
3 3 0	10 2 2	15 15 2	18 10 3	21 15 3	23 3 2	26 22 3	28 20 4
3 1 2	10 0 2	15 13 0	18 8 3	21 13 3	23 1 1	26 20 3	28 18 4
4 4 1	11 11 1	15 11 3	18 6 3	21 11 5	24 24 2	26 18 4	28 16 5
4 2 2	11 9 2	15 9 2	18 4 3	21 9 5	24 22 3	26 16 5	28 14 5
4 0 2	11 7 2	15 7 2	18 2 1	21 7 5	24 20 3	26 14 5	28 12 5
5 5 1	11 5 2	15 5 3	18 0 2	21 5 4	24 18 3	26 12 5	28 10 5
5 3 2	11 3 2	15 3 1	19 19 2	21 3 3	24 16 4	26 10 5	28 8 5
5 1 2	11 1 2	15 1 2	19 17 3	21 1 0	24 14 6	26 8 4	28 6 4
6 6 1	12 12 2	16 16 2	19 15 3	22 22 2	24 12 5	26 6 4	28 4 3
6 4 0	12 10 1	16 14 2	19 13 2	22 20 2	24 10 4	26 4 3	28 2 2
6 2 2	12 8 2	16 12 3	19 11 3	22 18 3	24 8 4	26 2 2	28 0 2
6 0 2	12 6 2	16 10 2	19 9 3	22 16 3	24 6 4	26 0 2	29 29 2
7 7 1	12 4 2	16 8 3	19 7 3	22 14 3	24 4 3	27 27 2	29 27 2
7 5 1	12 2 2	16 6 3	19 5 4	22 12 5	24 2 1	27 25 2	29 25 3
7 3 2	12 0 2	16 4 2	19 3 3	22 10 6	24 0 1	27 23 3	29 23 4
7 1 2	13 13 2	16 2 2	19 1 2	22 8 5	25 25 2	27 21 3	29 21 4
8 8 1	13 11 2	16 0 2	20 20 2	22 6 5	25 23 1	27 19 5	29 19 5
8 6 2	13 9 2	17 17 2	20 18 2	22 4 3	25 21 3	27 17 5	29 17 5
8 4 2	13 7 2	17 15 2	20 16 3	22 2 1	25 19 3	27 15 5	29 15 5
8 2 2	13 5 1	17 13 3	20 14 2	22 0 1	25 17 4	27 13 5	29 13 5
8 0 2	13 3 2	17 11 1	20 12 3	23 23 2	25 15 5	27 11 5	29 11 5
9 9 1	13 1 2	17 9 3	20 10 4	23 21 3	25 13 5	27 9 5	29 9 5
9 7 1	14 14 2	17 7 3	20 8 5	23 19 3	25 11 5	27 7 4	29 7 4
9 5 2	14 12 2	17 5 3	20 6 4	23 17 3	25 9 4	27 5 3	29 5 3
9 3 2	14 10 3	17 3 2	20 4 6	23 15 3	25 7 4	27 3 2	29 3 2
9 1 2	14 8 2	17 1 2	20 2 2	23 13 5	25 5 3	27 1 2	29 1 2

Β.4. Συμπεράσματα.

Β.4.Α. Αλγόριθμος Kintner

Τα πειράματα που πραγματοποιήσαμε στον αλγόριθμο των συντελεστών Zernike με βάση τη μέθοδο Kintner, η κύρια πηγή του λάθους είναι ο αναδρομικός τύπος που παράγει τους συντελεστές:

$$R_{pq}(r) = \frac{(K_2 \cdot r^2 + K_3) \cdot R_{p-2,q}(r) + K_4 \cdot R_{p-4,q}(r)}{K_1} \text{ , για p =q+4, q+6,…,p}_{\max}$$

που πρακτικά υπολογίζει τα εσωτερικά στοιχεία του πίνακα των ZMs, δηλαδή τα στοιχεία μετά τις 2 διαγωνίους. Προσδιορίσαμε το σφάλμα στους επιμέρους όρους αυτού του αθροίσματος και παρατηρήσαμε πως το υπεισερχόμενο σφάλμα είναι μικρότερο του συνολικού που προκύπτει από το συντελεστή αυτό συνολικά. Έτσι, μπορεί οι δύο αυτοί όροι να έχουν ο καθένας τους και μηδενικό ακόμα σφάλμα και όμως το άθροισμά τους να δίνει π.χ. 2 δ.ψ.λ. Η ερμηνεία αυτού ανάγεται στο πρόσημο των δύο προσθετέων που είναι αντίθετο και κατά συνέπεια, σύμφωνα και με την $4^{η}$ πρόταση (Α.3.1.4) για την αφαίρεση. Στατιστικά συχνά δηλαδή, συντρέχουν οι συνθήκες για την πρόταση αυτή. Πιο συγκεκριμένα, ο παραγόμενος αριθμός δ.ψ.λ. είναι τόσο μεγαλύτερος όσο και η βύθιση του υπολογιζόμενου ZMs σε σχέση με τον κοινό εκθέτη των δύο προσθετέων.

Αυτό το παραγόμενο λάθος πεπερασμένης ακρίβειας μεταδίδεται στις επόμενες αναδρομές ακριβώς επειδή αφορά αυτούς τους συντελεστές Zernike τους ιδίους. Προφανώς δε μεταδίδεται μέσω αυτών. Σποραδικά, οι προσθέσεις και οι πολλαπλασιασμοί που συμμετέχουν στον αλγόριθμο, δημιουργούν κάποια μείωση του λάθους, αλλά όχι σε βαθμό ικανό να υπερνικήσει τη συστηματικότητα με την οποία παράγεται το λάθος στην αναδρομή αυτήν.

Β.4.Β. Μέθοδος Q-recursive

Τα εκτελεσθέντα πειράματα υποδεικνύουν ότι στον αλγόριθμο των συντελέστων Zernike με βάση την q-recursive μέθοδο, η κύρια πηγή λάθους είναι κατά κανόνα ο αναδρομικός τύπος που παράγει τους συντελεστές $R_{p,q-4}$, δηλαδή η σχέση:

$$R_{pq}(r) = K_1 \cdot R_{p,q+4}(r) + \left(K_2 + \frac{K_3}{r^2} \right) \cdot R_{p,q+2}(r),\ p = 4,5,...,p_{\max},\ (q = p-4, p-6,...,0 \dot{\eta} 1)$$

που πρακτικά υπολογίζει τα εσωτερικά στοιχεία του πίνακα των ZMs, δηλαδή τα στοιχεία μετά τις 2 διαγωνίους. Ο λόγος που συμβαίνει αυτό είναι ότι συστηματικά, από στατιστικής απόψεως οι δύο προσθετέοι

$$K_1 \cdot R_{p,q+4}(r) \qquad \text{και} \qquad \left(K_2 + \frac{K_3}{r^2} \right) \cdot R_{p,q+2}(r)$$

εισέρχονται στην αναδρομή έχοντας έχοντας αντίθετο πρόσημο και την ίδια τάξη μεγέθους. Το γεγονός αυτό σημαίνει ότι στατιστικά συχνά και συστηματικά συντρέχουν οι συνθήκες της $4^{\eta\varsigma}$ πρότασης (Α.3.1.4) για την αφαίρεση. Όπως αναφέρεται στην πρόταση αυτή, ο παραγόμενος αριθμός λάθος δεκαδικών ψηφίων είναι τόσο μεγαλύτερος όσο και η βύθιση του υπολογιζόμενου ZMs σε σχέση με τον κοινό εκθέτη των δύο προσθετέων. Και όπως αναφέραμε και προηγουμένως στον αλγόριθμο Kintner, όντως υπάρχουν μέσω άλλων πράξεων τάσεις για μείωση του πλήθους των δ.ψ.λ. οι οποίες όμως δεν καταφέρνουν να υπερνικήσουντη συστηματικότητα με την οποία παράγεται το σφάλμα στο συγκεκριμένο υπολογισμό.

Β.4.C. Αλγόριθμος Prata

Τέλος, τα πειράματα επί του αλγορίθμου υλοποίησης των συντελεστών Zernike, Prata, έδειξαν πως βασική πηγή λάθους είναι και πάλι ο τύπος

$$R_{q+p,q}(r) = K_1 \cdot r \cdot R_{p+q-1,|q-1|}(r) + K_2 \cdot R_{p+q-2,q}(r),\ p = 4,6,..., p_{\max}, (q = 0,1,2,..., p_{\max} - p)$$

Βέβαια, παρατηρώντας τους επιμέρους όρους αυτών των συντελεστών παρατηρήσαμε πως ιδιαίτερα έντονη ήταν η συμβολή του δεύτερου όρου στο αποτέλεσμά μας, αφού ως επί το πλείστον το σφάλμα του όρου αυτού ταυτιζόταν με το σφάλμα του συνολικού αποτελέσματος. Στην προσπάθειά μας να ερμηνεύσουμε αυτό καταφεύγουμε στην πρόταση (Α.3.1.2) για την πρόσθεση. Αυτό, σημαίνει πως πρακτικά προσθέτοντας τους δύο όρους (ομόσημους) ουσιατικά το σφάλμα παραμένει αυτό του μεγαλυτέρου. Βέβαια, πρακτικά το σφάλμα αυτό ενδέχεται να έχει την ίδια τιμή ±1 ψηφίο, εφόσον όπως είδαμε και στην εφαρμογή της αφαίρεσης(αναφερόμαστε αυστηρά στο αποτέλεσμα και όχι στη διαδικασία) αν και το αποτέλεσμα φαινόταν να εχει τρία δ.ψ.λ. τελικά εφαρμόζοντας πράξεις είχε ένα λιγότερο.

Παρατήρηση: Στο τέλος της εργασίας παραθέτουμε APPENDIX B με συγκεντρωτικό πίνακα για τα δ.ψ.λ. στις τρεις μεθόδους που αναλύσαμε και υλοποιήσαμε, με στόχο να παρατηρηθεί εύκολα (μέσω χρωματικής αναπαράστασης) το σφάλμα και να μπορεί εύκολα να δει ο αναγνώστης ποια εκ των μεθόδων και σε ποια τάξη p είναι πιο συμφέρουσα.

C. Appendix

APPENDIX A:

Ο FAEST-5p αλγόριθμος

1. Ποσότητες διαθέσιμες στο χρόνο n: $\alpha_{m(n)}$, $b_{m(n)}$, $c_{m(n)}$, $x_{m(n)}$, $w_{m(n)}$, $a^f_{m(n)}$, $a^b_{m(n)}$, $a_{m(n)}$

2. Νέα πληροφορία: x(n+1), z(n+1)

3. Επικαιροποίηση του διανύσματος κέρδους

$$e^f_{m(n+1)} = x_{m(n+1)} + \alpha^T_{m(n)} \cdot x_{m(n)} \quad \text{(A.1)}$$

$$\varepsilon^f_{m(n+1)} = \varepsilon^f_{m(n+1)} / \alpha_{m(n)} \quad \text{(A.2)}$$

$$a_{m(n+1)} = a_{m(n)} + w_{m(n)} \cdot \varepsilon^f_{m(n+1)} \quad \text{(A.3)}$$

$$\alpha^f_{m(n+1)} = \alpha^f_{m(n)} + e^f_{m(n+1)} \cdot \varepsilon^f_{m(n+1)} \quad \text{(A.4)}$$

$$w_{m+1(n+1)} = \begin{bmatrix} 0 \\ w_{m(n)} \end{bmatrix} - e^f_{m(n+1)} / \alpha^f_{m(n)} \begin{bmatrix} 1 \\ \alpha_{m(n)} \end{bmatrix} \quad \text{(A.5)}$$

$$w_{m+1(n+1)} = \begin{bmatrix} d_{m(n+1)} \\ \delta_{m(n+1)} \end{bmatrix} \quad \text{(A.6)}$$

$$e^b_{m(n+1)} = -\delta_{m(n+1)} \cdot \alpha^b_{m(n)} \quad \text{(A.7)}$$

$$w_{m(n+1)} = d_{m(n+1)} - \delta_{m(n+1)} \cdot b_{m(n)} \quad \text{(A.8)}$$

$$\alpha_{m+1(n+1)} = \alpha_{m(n)} + e^f_{m(n+1)} \cdot e^f_{m(n+1)} / \alpha^f_{m(n)} \quad \text{(A.9)}$$

$$\alpha_{m(n+1)} = \alpha_{m+1(n+1)} + \delta_{m(n+1)} \cdot e^b_{m(n+1)} \quad \text{(A.10)}$$

$$\varepsilon^b_{m(n+1)} = e^b_{m(n+1)} / \alpha_{m(n+1)} \quad \text{(A.11)}$$

$$\alpha^b_{m(n+1)} = \alpha^b_{m(n)} + e^b_{m(n+1)} \cdot \varepsilon^b_{m(n+1)} \quad \text{(A.12)}$$

$$b_{m(n+1)} = b_{m(n)} + w_{m(n+1)} \cdot \varepsilon^b_{m(n+1)} \quad \text{(A.13)}$$

4. Επικαιροποίηση του φίλτρου ελαχίστων τετραγώνων πεπερασμένης κρουστικής απόκρισης

$$e_{m(n+1)} = z_{m(n+1)} + c^T_{m(n)} \cdot x_{m(n+1)} \quad \text{(A.14)}$$

$$\varepsilon_{m(n+1)} = e_{m(n+1)} / \alpha_{m(n+1)} \tag{A.15}$$

$$c_{m(n+1)} = c_{m(n)} + w_{m(n+1)} \cdot \varepsilon_{m(n+1)} \tag{A.16}$$

APPENDIX B.

kintner	q-recursive	prata
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
2	1	2
1	0	1
2	1	2
1	1	2
1	0	1
2	1	2
2	1	2
1	0	1
2	0	0
2	1	2
2	1	2
1	0	1
2	0	1
2	1	2
2	1	2
1	0	1
3	1	2
3	0	2
2	1	2
2	1	2
1	0	1
2	0	1
2	0	2
2	1	2
2	1	2
1	0	1
3	1	2
3	2	2
3	2	2
2	2	2
2	2	2
1	0	1
3	1	2
3	1	2
2	1	2
2	1	2
2	1	2
2	1	2
3	0	1
3	2	2
3	2	2
3	2	2
3	2	2

3	2	2
2	1	2
2	1	2
2	2	2
3	2	2
3	2	1
3	2	2
3	2	2
3	1	2
2	1	2
2	2	3
2	2	2
3	2	2
3	2	3
2	2	2
1	2	2
2	1	2
3	0	0
3	2	3
3	2	2
3	1	2
2	2	3
2	2	1
2	2	2
2	1	2
3	1	2
3	2	3
3	2	2
3	2	3
3	2	3
3	2	2
2	2	2
2	2	2
2	1	2
3	1	2
3	1	3
3	1	1
3	1	3
3	2	3
2	2	3
2	2	2
2	2	2
2	1	2
3	2	3
3	1	3
3	2	2
3	2	3
3	2	3
3	3	3
3	3	3
2	3	1
2	3	2
2	1	2
3	2	3
3	2	3

3	2	2
3	2	3
3	2	3
3	2	3
3	3	4
3	3	3
3	3	2
2	1	2
3	1	2
3	2	3
3	3	2
3	3	3
3	3	4
4	4	5
4	4	4
5	6	6
3	4	2
3	4	2
2	1	2
3	1	2
3	0	3
3	2	3
3	2	3
4	3	5
4	3	5
3	3	5
3	3	4
3	2	3
3	2	0
2	1	2
3	1	2
3	2	3
3	3	3
3	3	3
4	4	5
5	5	6
4	4	5
3	4	5
2	3	3
2	3	1
2	3	1
2	1	2
3	2	3
3	3	3
3	3	3
3	3	3
4	4	5
4	4	5
4	4	5
2	3	4
2	3	3
2	3	2
2	3	1
2	1	2
3	2	3

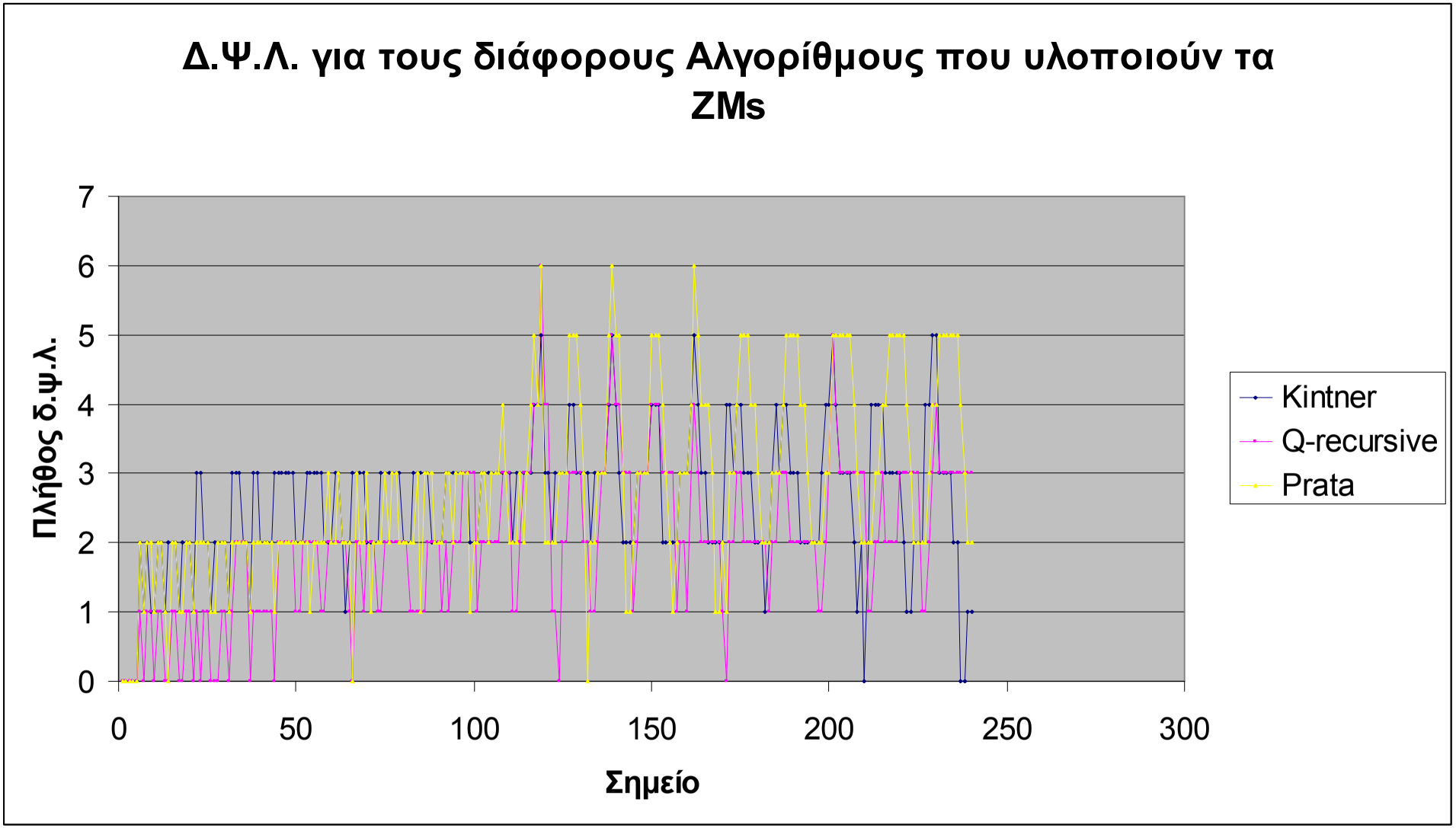
3	2	3
3	1	3
4	3	4
5	4	6
4	3	5
3	2	4
3	2	4
2	2	4
2	2	3
2	2	1
2	2	1
2	1	2
4	0	1
4	2	3
3	2	3
4	3	4
4	3	5
3	2	5
3	2	5
3	2	4
2	2	4
2	2	3
2	2	2
1	2	2
2	1	2
3	2	3
4	2	3
3	3	3
4	3	4
4	3	5
3	2	5
3	2	5
3	2	5
2	2	4
2	2	4
2	2	3
2	2	2
2	2	2
2	1	2
3	1	2
4	2	3
4	3	3
5	5	5
4	4	5
3	3	5
3	3	5
3	3	5
3	3	5
2	3	4
1	3	3
2	3	2
0	3	2
2	1	2
4	1	2
4	**2**	**3**

4	3	4
3	2	4
3	2	5
3	2	5
3	2	5
3	3	5
2	3	5
1	3	4
1	3	3
2	3	2
2	3	2
2	1	2
4	1	2
4	2	3
5	3	4
5	4	4
3	3	5
3	3	5
3	3	5
3	3	5
2	3	5
2	3	5
0	3	4
0	3	3
1	3	2
1	3	2

Όσον αφορά τις μέσες τιμές σφάλματος, αυτές για τις τρεις προαναφερθείσες μεθόδους είναι:

2,595833	1,991667	2,754167

Πρακτικά δηλαδή, ο αλγόριθμος Prata έχει τη μεγαλύτερη μέση τιμή σφάλματος στο διάστημα μέχρι την 30στή τάξη. Επίσης, στο διάγραμμα που παραθέτουμε παρακάτω παρατηρούμε πως μέχρι και την τάξη 19, ο αλγόριθμος Kintner σχεδόν κυριαρχεί στο μέγιστο αριθμό δ.ψ.λ., ενώ ο Prata ακολουθεί ώντας ο πιο ασταθης αριθμός αφού παρατηρούνται έντονες μεταβολές στο πλήθος των δ.ψ.λ.. Από το σημείο εκείνο και μετά ο Prata υπερέχει σε πλήθος δ.ψ.λ. ακολουθούμενος από τον Kintner. Η μέθοδος q-recursive είναι η ας πούμε λιγότερο προσβεβλημμένη από το f.p.e. από τις τρεις αυτές μεθόδους.


Δ.Ψ.Λ. για τους διάφορους Αλγορίθμους που υλοποιούν τα ZMs
Πλήθος δ.ψ.λ.
0
1
2
3
4
5
6
7
0
50
100
150
200
250
300
Σημείο
Kintner
Q-recursive
Prata

D. Βιβλιογραφία

[1] E.C. Kintner, A recursive relation for calculating the Zernike polynomials, Opt. Acta 23 (6) (1976) 499–500.

[2] M. Teague, Image analysis via the general theory of moments, J. Opt. Soc. Am. 70 (8) (1980) 920–930.

[3] G. Carayannis, D. Manolakis and N. Kalouptsidis, "A fast sequential algorithm for least squares filtering and prediction", IEEE Trans. on ASSP vol. 31, (Dec. 1983), pp. 1394-1402.

[4] S. Ljung and L. Ljung: Error propagation properties of recursive least squares adaptation algorithms. Proc. 9th IFAC World Congress, Budapest, pp X-70-74, (July 1984).

[5] J.M. Cio and T. Kailath, \Windowed Fast Transversal Filters Adaptive Algorithms with Normalization," IEEE Transactions on ASSP, Vol. 33, No. 3, (June 1985), pp. 607-625.

[6] P.Fabre and C. Gueguen, Improvement of the fast recursive least-squares algorithms via normalization: A comparative study, IEEE Trans. On ASSP 34(2)(1986) 296 - 308

[7] J-L. Botto, G.V. Moustakides, Stabilizing the fast Kalman algorithms, IEEE Transactions on Acoustic, Speech and Signal Processing, vol. ASSP-37, no. 9, pp. 1342-1348, (Sept. 1989)

[8] A. Prata, W.V.T. Rusch, Algorithm for computation of Zernike polynomials expansion coefficients, Appl. Opt. 28 (1989) 749–754.

[9] A. Khotanzad, J.-H. Lu, Classification of invariant image representations using a neural network, IEEE Trans. Acoust., Speech Sign.Process. 38 (6) (1990) 1028–1038.

[10] A. Khotanzad, Y.H. Hong, Invariant image recognition by Zernike moments, IEEE Trans. Pattern Anal. Machine Intell. PAMI-12 (5) (1990) 489–497.

[11] C. Papaodysseus, E. Koukoutsis, C. Vassilatos, Accurate Prediction of the Numerical Error Generated in the Forward Linear Prediction Algorithms, ISMM Microcomputer Applic., Vol. 12, No 1, (1993), pp. 1-6.

[12] C. Papaodysseus, E. Koukoutsis, C. Triantafyllou, Error Sources and Error Propagation in the Levinson-Durbin algorithm, IEEE Transactions on Signal Processing, Vol. 41, No 4, (April 1993), pp. 1635-1651.

[13] C. Papaodysseus, D. Gorgoyannis, E. Koukoutsis, P. Roussopoulos, "A very robust, fast, parallelizable adaptive least squares algorithm with excellent tracking abilities," icassp, vol. 6, pp.385-388, Acoustics, Speech, and Signal Processing, 1994. ICASSP-94 Vol 6., 1994 IEEE International Conference on, (1994)

[14] C. Papaodysseus, E. Koukoutsis, C. Vassilatos, Error Propagation and Methods of Error Correction in the LS FIR Filtering and the l-step ahead Prediction Algorithms, IEEE Transactions on Signal Processing, (May 1994), Vol. 42, No 5, pp. 1097-1108.

[15]Papaodysseus, C., Koukoutsis E., Halkias C.C., Stavrakakis G., Exact Analysis of the Finite Precision Error Generation and Propagation in the FAEST and the Fast Transversal Algorithm. A General Methodology for Developing Robust RLS Schemes, Journal of Mathematics and Computers in Simulation, (January 1997), 44, pp. 29-41, Elsevier Science.

[16]Papaodysseus, C., Koukoutsis E., Halkias C.C., Roussopoulos G., A Parallelizable, Robust, Recursive Least Squares Algorithm for Adaptive Filtering, International Journal of Parallel Algorithms of Computer Mathematics, Vol. 67, (1998), pp. 275-292.

[17] S.X. Liao, M. Pawlak, On the accuracy of Zernike moments for image analysis, IEEE PAMI 20 (12) (1998) 1358–1364.

[18] R. Mukundan, S.H. Ong, P.A. Lee, Image analysis by Tchebichef moments, IEEE Trans. Image Process. 10 (9) (2001) 1357–1364.

[19] R. Mukundan, S.H. Ong, P.A. Lee, Discrete vs. continuous orthogonal moments for image analysis, in: Proceedings of International
Conference on Imaging Science Systems and Technology, vol. 1, 2001, pp. 23–29.

[20] J. Haddadnia, M. Ahmadi, K. Faez, An efficient method for recognition of human faces using higher orders pseudo Zernike momentinvariant, in: Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition (FGR'02), (2002).

[21] P.T. Yap, P. Raveendran, S.H. Ong, Image analysis by Krawtchouk moments, IEEE Trans. Image Process. 12 (11) (2003) 1367–1377.

[22] C. Papaodysseus, C. Alexiou, Th. Panagopoulos, G. Roussopoulos, D. Kravaritis: A Novel Methodology for Studying and Remedying Finite Precision Error, Stochastic Environmental Research and Risk Assessment, Vol. 17, Issue 1-2, pp. 1-19, (2003).

[23] C.-W Chong, R. Paramesran, R. Mukundan, A comparative analysis of algorithms for fast computation of Zernike moments, Pattern Recognition 36 (2003) 731–742.

[24]Chee-Way Chong, P. Raveendran, R. Mukundan: The scale invariants of pseudo-Zernike moments. Pattern Anal. Appl. 6(3): 176-184 (2003) 3

[25]Chee-Way Chong, P. Raveendran, R. Mukundan: A comparative analysis of algorithms for fast computation of Zernike moments. Pattern Recognition 36(3): 731-742 (2003) 2

[26]Chee-Way Chong, P. Raveendran, R. Mukundan: Translation invariants of Zernike moments. Pattern Recognition 36(8): 1765-1773 (2003)

[27] S. Rodtook, S.S. Makhanov, Numerical experiments on the accuracy of rotation moment invariants, Image Vis. Comput. 23 (2005) 577–586.

[28] N.K. Kamila, S. Mahapatra, S. Nanda, Invariance image analysis using modified Zernike moments, Pattern Recogn. Lett. 26 (2005) 747–753.

[29] G. A. Papakostas, Y. S. Boutalis, C. N. Papaodysseus and D. K. Fragoulis, Numerical Error Analysis in Zernike Moments Computation, Image and Vision Computing, Vol. 24, No 9, pp. 960-969, (2006).

[30] C. -Y. Wee, R. Paramesran, Efficient computation of radial moment functions using symmetry property, Pattern Recognition 39 (2006) 2036–2046.

[31] S.-K. Hwang, W.-Y. Kim, A novel approach to the fast computation of Zernike moments, Pattern Recognition 39 (2006) 2065–2076.

[32] G.A. Papakostas, Y.S. Boutalis, D.A. Karras, B.G. Mertzios, A new class of Zernike moments for computer vision applications, Inf. Sci. 177 (13) (2007) 2802–2819.

[33] G.A. Papakostas, Numerical stability of fast computation algorithms of Zernike moments, Appl. Math. Comput. 195 (2008) 326–345.

[34] Chandan Singh, Ekta Walia: Fast and numerically stable methods for the computation of Zernike moments. Pattern Recognition 43(7): 2497-2506 (2010)

Χρήσιμες πληροφορίες βρήκαμε:

[a] Digital image reconstruction by using Zernike moments. A. Padilla-Vivanco, A. Martinez-Ramirez² and F. Granados-Agustin,[1]Instituto Nacional de Astrofisica, Puebla. Mexico,²Facultad de Ciencias de la Computacisn. Ciudad Universitaria, Edificio,Puebla. Mexico

[b] Zernike Polynomials. Zernike for the web. James C. Wyant
http://www.optics.arizona.edu/jcwyant/zernikes/zernikepolynomials.htm
[c] Accurate and efficient computation of high order zernike moments. Gholam Reza Amayeh, Ali Erol, George Bebis, and Mircea Nicolescu
Computer Vision Laboratory, University of Nevada, Reno