ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΣΕΡΡΩΝ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΕΠΙΚΟΙΝΩΝΙΩΝ

ΘΕΜΑ

«Ανάπτυξη γραφικού περιβάλλοντος σε Matlab για συσταδοποίηση δεδομένων μέσω των ιεραρχικών αλγορίθμων απλού συνδέσμου, πλήρους συνδέσμου, μέσου συνδέσμου και παρουσίαση των αποτελεσμάτων σε δενδρογράμματα.»

Σπουδαστής

Λεωνίδας Ανδρεάδης

Επιβλέπων

Δρ. Τσιμπίρης Αλκιβιάδης

ΣΕΡΡΕΣ 2009

Περιεχόμενα

Εισαγωγή

Η εφαρμογή που υλοποιήθηκε στα πλαίσια της παρούσας πτυχιακής εργασίας αλλά και η μελέτη που θα ακολουθήσει ασχολούνται με το αντικείμενο της συσταδοποίησης και ειδικότερα με τους ιεραρχικούς αλγορίθμους απλού, πλήρους και μέσου δεσμού αλλά και επεκτείνεται σε θέματα που αφορούν τη συσταδοποίηση, όπως τα μέτρα απόστασης και τύπους δεδομένων που συναντώνται. Η εφαρμογή συγκεκριμένα αποτελείται από δύο φόρμες, η μία είναι υπεύθυνη για το φόρτωμα και την επιλογή των δεδομένων από μία βάση και η άλλη ασχολείται με την συσταδοποίηση, την ανάλυση των διαφόρων πτυχών της και την αποθήκευση των επεξεργασμένων δεδομένων στη βάση. Οι βάσεις δεδομένων έχουν καθοριστικό ρόλο στην διαχείριση δεδομένων αφού έχουν σχεδιαστεί για να προσφέρουν ένα οργανωμένο μηχανισμό για την αποθήκευση, διαχείριση και ανάκτηση πληροφοριών. Από τις πληροφορίες που υπάρχουν λοιπόν στη βάση μπορούμε να εξάγουμε διάφορες “κρυφές” πληροφορίες, η διαδικασία αυτή λέγεται εξόρυξη δεδομένων (Data Mining) . Η εξόρυξη δεδομένων χρησιμοποιείται ευρέως σε τομείς όπως στην έρευνα της αγοράς και των καταναλωτών, στην ανάλυση προϊόντων, στην ανάλυση προσφοράς και ζήτησης, το ηλεκτρονικό εμπόριο και άλλα. Με την συσταδοποίηση μπορούμε να επεξεργαστούμε τα δεδομένα που έχουμε και να ανακαλύψουμε τις ομοιότητες και τις μεταξύ τους σχέσεις. Συσταδοποίηση είναι ο διαχωρισμός ενός συνόλου αντικειμένων σε μικρότερα υποσύνολα που έχουν νόημα, ακόμα μπορεί να κατανοηθεί ως η ομαδοποίηση των διαφόρων αντικειμένων. Τα δεδομένα που ανήκουν στην ίδια ομάδα θα είναι πιο όμοια μεταξύ τους από ότι με τα υπόλοιπα αντικείμενα. Έτσι μπορούμε να πούμε ότι θυσιάζουμε την κατανόηση του κάθε επιμέρους αντικειμένου για χάριν της απλότητας και της καλύτερης κατανόησης. Η μελέτη που ακολουθεί αρχικά πραγματεύεται την έννοια της συσταδοποίησης,τις εφαρμογές της, υπάρχουσες έρευνες πάνω στο αντικείμενο και έπειτα επεκτείνεται σε διάφορα μέτρα απόστασης που χρησιμοποιούνται στη συσταδοποίηση, τύπους δεδομένων που συναντώνται αλλά και αναλυτική ή αναφορική ανάλυση των διαφόρων ιεραρχικών μεθόδων συσταδοποίησης. Έπειτα ακολουθεί μία αναφορά στο πρόγραμμα Matlab, τις βάσεις δεδομένων και το ODBC αλλά και των βασικών συναρτήσεων που προσφέρει το Matlab για την επίτευξη της συσταδοποίησης. Τέλος γίνεται ανάλυση της λειτουργίας της εφαρμογής που υλοποιήθηκε στα πλαίσια της πτυχιακής εργασίας και του κώδικα.

1. Αλγόριθμοι Συσταδοποίησης

1. Η έννοια της συσταδοποίησης

1.1 Εισαγωγή στην έννοια της συσταδοποίησης

Ζούμε σε ένα κόσμο γεμάτο πληροφορία. Κάθε μέρα οι άνθρωποι ανά τον κόσμο πρέπει να επεξεργαστούν πληροφορία που προέρχεται από διάφορες παρατηρήσεις και πηγές. Τα δεδομένα περιγράφουν τα χαρακτηριστικά ενός ζωντανού οργανισμού, απεικονίζουν τις ιδιότητες ενός φυσικού φαινομένου, συγκεφαλαιώνουν τα αποτελέσματα μίας επιστημονικής έρευνας και καταγράφουν τις διάφορες δυναμικές ενός μηχανικού συστήματος. Κυρίως όμως, τα δεδομένα παρέχουν μία βάση για περεταίρω ανάλυση, συλλογισμό και λήψη αποφάσεων για την κατανόηση όλων των ειδών αντικειμένων και φαινομένων. Μία σημαντική δραστηριότητα στην ανάλυση των δεδομένων είναι να διαχωρίσουμε αυτά τα δεδομένα σε κατηγορίες ή ομάδες, αλλιώς αποκαλούμενα ως συστάδες (clusters). Τα δεδομένα τα οποία βρίσκονται στην ίδια ομάδα ή κατηγορία θα πρέπει να έχουν κάποια ομοιότητα βασισμένη σε κριτήρια. Έτσι λοιπόν, σαν μία από τις πιο βασικές δραστηριότητες του ανθρώπινου είδους η κατηγοριοποίηση παίζει ένα βασικό και αναντικατάστατο ρόλο στην μακρά ιστορία της εξέλιξης του. Προκειμένου να καταλάβουν ένα νέο αντικείμενο ή κάποιο νέο φαινόμενο, οι άνθρωποι προσπαθούν να αναγνωρίσουν χαρακτηριστικά και περεταίρω να συγκρίνουν αυτά τα χαρακτηριστικά με τα χαρακτηριστικά γνωστών αντικειμένων κατά την ομοιότητα ή μη ομοιότητά τους με βάση κάποιους κανόνες ή μεθόδους. Για παράδειγμα, όλα τα φυσικά στοιχεία μπορούν να διαχωριστούν σε τρείς ομάδες : Ζώα, φυτά και μέταλλα. Σύμφωνα με τη βιολογική ταξινόμηση, τα ζώα μπορούν να ταξινομηθούν σε κατηγορίες όπως βασίλειο, είδος, φύλο, οικογένεια, γένος και άλλα. Έτσι λοιπόν έχουμε ζώα με ονόματα όπως τίγρης, λιοντάρι, λύκος, σκύλος, άλογο, γάτα και τα λοιπά. Με αυτή την κατηγοριοποίηση μπορούμε να συνάξουμε συμπεράσματα για ένα αντικείμενο ανάλογα με την κατηγορία που ανήκει. Έτσι λοιπόν, αν δούμε ένα άλογο, ξέρουμε ότι καλπάζει γρήγορα χωρίς να χρειαστεί να το δούμε στην πραγματικότητα. Η συσταδοποίηση είναι ένα σημαντικό εργαλείο για την εξόρυξη δεδομένων αφού μπορεί να αναγνωρίσει μοντέλα ή τάσεις χωρίς την ύπαρξη κάποιας πληροφορίας προερχόμενης από παρατήρηση. Μπορεί σε γενικές γραμμές να καθοριστεί ως η διαδικασία κατά την οποία χωρίζουμε ένα σύνολο αντικειμένων σε συστάδες, κάθε μια από τις οποίες αντιπροσωπεύει ένα υπο-σύνολο που έχει νόημα. Τα αντικείμενα αυτά μπορεί να είναι εγγραφές μίας βάσης δεδομένων, σημεία γραφήματος, λέξεις, εικόνες, ή οποιαδήποτε συλλογή δεδομένων όπου τα

μέλη της περιγράφονται από ένα σύνολο χαρακτηριστικών ή καθορισμένων σχέσεων.

1.2 Επιβλεπόμενη και μη επιβλεπόμενη συσταδοποίηση

Έχουμε λοιπόν την επιβλεπόμενη και μη επιβλεπόμενη συσταδοποίηση. Στην επιβλεπόμενη συσταδοποίηση θεωρούμε πως τα δεδομένα μας είναι κατηγοριοποιημένα. Ο στόχος της επιβλεπόμενης συσταδοποίησης είναι να αναγνωρίσει έννοιες κατηγορίες συστάδων που έχουν μεγάλη πιθανότητα πυκνότητας. Στην μη επιβλεπόμενη συσταδοποίηση δεν έχουμε διαθέσιμες κάποιες ετικέτες δεδομένων. Ο στόχος της συσταδοποίησης σε αυτή την περίπτωση είναι να χωρίσει τα πεπερασμένα και αταξινόμητα δεδομένα σε φυσικά σύνολα. Η ανάγκη για μη επιβλεπόμενη συσταδοποίηση προέρχεται λοιπόν από την ανάγκη της εξερεύνησης της άγνωστης φύσης των δεδομένων που συλλέξαμε με λίγη ή χωρίς καμία πληροφορία.Η παραδοσιακή αντίθεση μεταξύ της επιβλεπόμενης και μη συσταδοποίησης είναι ότι στην μία περίπτωση της επιβλεπόμενης θέτουμε μία μεταβλητή την οποία καθορίζουμε ως σημαντική για την εκτέλεση ενώ στην μη επιβλεπόμενη δεν υπάρχει κάποια καθοδήγηση, αλλά ερευνούμε τη δομή. Ας πάρουμε για παράδειγμα τη διάγνωση κάποιας αρρώστιας και τη θεραπεία σε κλινικές. Για κάποια συγκεκριμένη αρρώστια μπορεί να υπάρχουν διάφορες άγνωστες υποκατηγορίες που να δείχνουν παρόμοιες μορφολογικές συμπεριφορές αλλά να αντιδρούν διαφορετικά στην ίδια θεραπεία. Σε αυτή την περίπτωση η συσταδοποίηση δεδομένων γονιδιακής έκφρασης που μετρούν τη δραστηριότητα των γονιδίων μπορεί να παρέχει μία υποσχόμενη μέθοδο για την αποκάλυψη υποκατηγοριών και έτσι στο καθορισμό ανάλογων θεραπειών. Ακόμα κάποιες φορές, η διαδικασία σήμανσης των δεδομένων μπορεί να είναι εξαιρετικά δαπανηρή και χρονοβόρα, γεγονός το οποίο καθιστά τη συσταδοποίηση ως μία καλή επιλογή θεωρώντας την πιθανότατα μεγάλη εξοικονόμηση σε κόστος και σε χρόνο. Επιπλέον η συσταδοποίηση παρέχει μία συνοπτική αναπαράσταση των δεδομένων κάτι το οποίο είναι χρήσιμο στην ανάλυση δεδομένων μεγάλης κλίμακας.

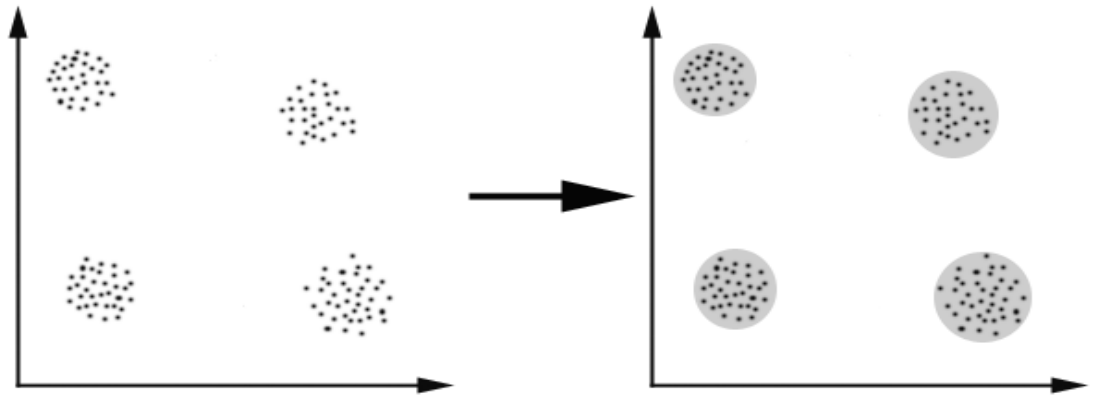
1.3 Οι στόχοι και ο ορισμός της συσταδοποίησης

Οι στόχοι της συσταδοποίησης μπορούν να συνοψιστούν στις παρακάτω τέσσερις κύριες πτυχές :

1. Την ανάπτυξη μίας ταξινόμησης.
2. Τη διερεύνηση χρήσιμων εννοιολογικών συστημάτων για την ομαδοποίηση των οντοτήτων.

3. Την εξαγωγή συμπερασμάτων μέσω της διερεύνησης των δεδομένων.
4. Την δοκιμή των συμπερασμάτων ή την απόπειρα να καθορίσουμε αν πρότυπα τα οποία έχουν καθοριστεί από άλλες διαδικασίες υπάρχουν όντος στα δεδομένα.

Η μη προβλεπόμενη συσταδοποίηση είναι μία υποκειμενική διαδικασία που αποκλείει την απόλυτη κρίση ως προς τη σχετική αποτελεσματικότητα όλων των μεθόδων συσταδοποίησης. Στην συσταδοποίηση μία ομάδα αντικειμένων χωρίζεται σε ένα σύνολο περισσότερο ή μη ομογενών συνόλων με βάση ενός συχνά υποκειμενικά επιλεγμένου μέτρου ομοιότητας, (για παράδειγμα, επιλογή ενός τέτοιου μέτρου με κριτήριο την δημιουργία ενδιαφερόντων συστάδων.) τέτοιο ώστε η ομοιότητα των αντικειμένων που ανήκουν σε ένα σύνολο να είναι μεγαλύτερη από την ομοιότητα αντικειμένων που ανήκουν σε διαφορετικά σύνολα. Επίσης, διαφορετικός αλγόριθμος συσταδοποίησης ή κριτήριο ακόμα και για τον ίδιο



αλγόριθμο αλλά με διαφορετική επιλογή των παραμέτρων μπορεί να επιφέρουν εντελώς διαφορετικά αποτελέσματα. Για παράδειγμα, οι άνθρωποι μπορούν να κατηγοριοποιηθούν με βάση την εθνικότητα, την περιοχή, την ηλικία, κοινωνικό-οικονομική κατάσταση, την παιδεία, την καριέρα, τα χόμπι, το ύψος, το βάρος, το αγαπημένο φαγητό, τον τρόπο ντυσίματος και πάει λέγοντας. Προφανώς, διαφορετικά κριτήρια στη συσταδοποίηση μπορεί να αντιστοιχίσουν ένα συγκεκριμένο πρόσωπο σε πολύ διαφορετικές ομάδες και συνεπώς να παράγουν διαφορετικά σύνολα. Ωστόσο, δεν υπάρχει απολύτως κανένας τρόπος να καθοριστεί ποιό κριτήριο είναι το καλύτερο. Είναι γεγονός ότι κάθε κριτήριο έχει τη δική του χρησιμότητα ανάλογα με την περίπτωση, παρόλο που κάποια έχουν πιο ευρεία χρησιμότητα από κάποια άλλα. Η συσταδοποίηση είναι η ανάθεση αντικειμένων σε ομάδες (αποκαλούμενες συστάδες) έτσι ώστε τα αντικείμενα από την ίδια συστάδα να είναι πιο όμοια το ένα με το άλλο από τα αντικείμενα που ανήκουν σε διαφορετικές συστάδες. Ο στόχος της συσταδοποίησης είναι να καθορίσει τη σωστή ομαδοποίηση μεταξύ ενός συνόλου αταξινόμητων

αντικειμένων. Αλλά με ποίο τρόπο μπορούμε να αποφασίσουμε τί αποτελεί καλή συσταδοποίηση; Μπορεί να αποδειχθεί πώς δεν υπάρχει το απόλυτα σωστό κριτήριο που θα είναι ανεξάρτητο από τον τελικό στόχο της συσταδοποίησης. Επομένως, είναι εκείνος ο οποίος εκτελεί τη συσταδοποίηση ο οποίος πρέπει να θέση το κριτήριο, με τέτοιο τρόπο ώστε το αποτέλεσμα της συσταδοποίησης να ταιριάζει στις ανάγκες του.

1.4 Εφαρμογές της συσταδοποίησης

Η συσταδοποίηση έχει πολλές πιθανές εφαρμογές και έχει χρησιμοποιηθεί σε διάφορους τομείς όπως :

- Μηχανική : (υπολογιστική νοημοσύνη, μηχανικής μάθησης, αναγνώριση προτύπων, μηχανολογία, ηλεκτρολογία) Τυπικές εφαρμογές αποτελούν η φωνητική αναγνώριση, η ανάλυση σήματος ραντάρ, η συμπίεση πληροφορίας και αφαίρεση θορύβου.

- Επιστήμες Υπολογιστών : Εδώ βλέπουμε όλο και περισσότερες εφαρμογές της συσταδοποίησης, σποραδική ανάλυση της βάσης δεδομένων, ανάκτηση πληροφοριών, κατάτμηση των εικόνων, εξόρυξη διαδικτύου.

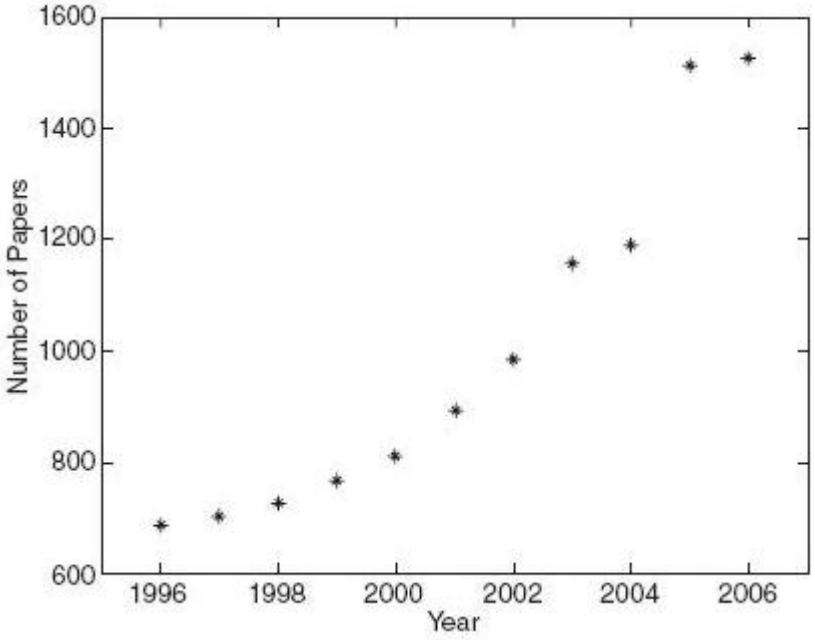
- Ιατρικές και ζωοντολογικές επιστήμες : (Γενετική, βιολογία, μικροβιολογία, παλαιοντολογία, φυσιατρική, παθολογία) Αυτά τα πεδία αποτελούν την κυρίως εφαρμογή της συσταδοποίησης όταν αυτή ήταν ακόμα σε πρώιμο στάδιο και θα συνεχίσει πιθανόν να είναι ένα από τα κύρια πεδία εφαρμογής αλγορίθμων συσταδοποίησης. Σημαντικές εφαρμογές αποτελούν η αναγνώριση της λειτουργίας γονιδίων και πρωτεϊνών, διάγνωση ασθενειών και θεραπεία και άλλα.
- Μάρκετινγκ: Βρίσκοντας ομάδες πελατών με παρόμοια συμπεριφορά δεδομένης μιας μεγάλης βάσης δεδομένων στοιχείων πελατών που περιέχει τις ιδιότητες και την αγοραστική συμπεριφορά τους στο παρελθόν.
- Βιβλιοθήκες: Διάταξη βιβλίων.
- Μελέτες σεισμού: Συσταδοποίηση των παρατηρηθέντων επίκεντρων σεισμού για τον προσδιορισμό των επικίνδυνων ζωνών.
- Στην Αστρονομία και τις γεωλογικές επιστήμες : (γεωγραφία, γεωλογία, τηλεανίχνευση) Η συσταδοποίηση σε αυτή την περίπτωση μπορεί να

χρησιμοποιηθεί γα την ταξινόμηση αστεριών και πλανητών, διερεύνηση σχηματισμών της Γής, κατάτμηση περιοχών και πόλεων, μελέτη ποταμών και βουνών.

- Κοινωνικές επιστήμες : (κοινωνιολογία, αρχαιολογία, ανθρωπολογία, εκπαίδευση) Ενδιαφέρουσες εφαρμογές αποτελούν η ανάλυση μοτίβων συμπεριφοράς, η αναγνώριση σχέσης μεταξύ των διαφόρων πολιτισμών, στην εξελικτική ιστορία των γλωσσών, ανάλυση των κοινωνικών σχέσεων, στην αρχαιολογική ανασκαφή και στην ταξινόμηση κειμηλίων και την μελέτη της εγκληματικής συμπεριφοράς.

1.5 Έρευνες επάνω στο αντικείμενο της συσταδοποίησης

Η συσταδοποίηση έχει μακρά ιστορία, που χρονολογείται από την εποχή του Αριστοτέλη. Υπάρχει μία τεράστια και διαρκώς αυξανόμενη βιβλιογραφία που αφορά την συσταδοποίηση από μία πληθώρα επιστημονικών κλάδων. Η Web of Science που αποτελεί μία υπηρεσία ευρετηρίασης βιβλιογραφικών αναφορών επιστημονικής αρθρογραφίας (www.isinet.com/products/citation/wos/) μόνο εμφανίζει πάνω από 12.000 περιοδικά και έγγραφα διασκέψεων να χρησιμοποιούν τη λέξη συσταδοποίηση μέσα στον τίτλο, στις λέξεις κλειδιά και στην περίληψη την περασμένη δεκαετία. Αυτά τα έγγραφα προέρχονται από πάνω από 3.000 δημοσιευμένα περιοδικά σε περισσότερες από 200 κατηγορίες. Αναλυτική παρουσίαση φαίνεται παρακάτω.



Εικόνα 1 . Ο αριθμός των εγγράφων πάνω στην συσταδοποίηση από το 1996 έως το 2006

Από αυτά τα χιλιάδες έγγραφα πάνω στην συσταδοποίηση, μερικά από τα έγγραφα διαφόρων ερευνών που αξίζουν ιδιαίτερη προσοχή από μια στατιστική άποψη αναγνώρισης μοτίβων, Jain et al. (1999) μελέτησε ένα μεγάλο αριθμό αλγορίθμων συσταδοποίησης μαζί με τις εφαρμογές τους στην κατάτμηση εικόνων, αναγνώριση αντικειμένων και χαρακτήρων, στην ανάκτηση πληροφοριών, και την εξόρυξη δεδομένων. Hansen και Jaumard (1997) ερευνούν προβλήματα συσταδοποίησης πάνω σε ένα μαθηματικό υπόβαθρο. Kolatch (2001) και He (1999) διαπραγματεύονται τις εφαρμογές της συσταδοποίησης στην μελέτη σποραδικών βάσεων δεδομένων και στην ανάκτηση πληροφοριών. Αντίστοιχα ο Berkhin (2001) επεκτείνει το θέμα στο πεδίο της εξόρυξης δεδομένων. Murtagh (1983) εκθέτει τις εξελίξεις στους αλγόριθμους ιεραρχικής συσταδοποίησης. Baraldi και Blonda (1999) ανακεφαλαιώνουν τους αλγόριθμους δεδομένων χρησιμοποιώντας το μαλακό αναγνωριστικό πρότυπο. Liao (2005) επικεντρώνεται στην συσταδοποίηση χρονοσειρών. Gordon (1998) και Halkidi et al. (2002) δίνουν έμφαση σε ένα συγκεκριμένο τομέα περί της επικύρωσης των συστάδων. Μία από τις πιο πρόσφατες και περιεκτικές μελέτες πάνω στους αλγόριθμους συσταδοποίησης αποτελεί αυτή των Xu και Wunsch (2005). Ο Kettenring (2006) , διαπραγματεύεται πρακτικά θέματα της συσταδοποίησης. Περισσότερα έγγραφα ερευνών μπορούν επίσης να βρεθούν από τους Baraldi και Schenato (1999) , Bock (1996), Dubes (1993), Fasulo (1999) , Jain et al. (2000) ,Zhao και Karypis (2005). Επιπλέον, συγκριτική έρευνα πάνω στους αλγορίθμους συσταδοποίησης έχει γίνει και αναφερθεί στη λογοτεχνία. Για παράδειγμα, ο Rauber et al. (2000) παρουσιάζει εμπειρικά αποτελέσματα για πέντε αλγορίθμους συσταδοποίησης, συμπεριλαμβανομένων των συσσωρευτικών ιεραρχικών μεθόδων, μεθόδων

Bayesian, αυτοδιοργάνωση χαρτών χαρακτηριστικών, αυξανόμενων ιεραρχικών αυτοδιοργανωμένων χαρτών, παραγωγική τοπογραφική χαρτογράφηση. Ο Wei et al. επικεντρώνεται στη σύγκριση γρήγορων αλγορίθμων για μεγάλες βάσεις δεδομένων. Εφαρμογές και τις αξιολογήσεις των διαφόρων αλγορίθμων για την ανάλυση δεδομένων γονιδιακής έκφρασης πειραμάτων DNA μικροστοιχείων περιγράφονται από τον Jiang et al. (2004), Madeira και Oliveira (2004), Shamir και Sharan (2002) και Tibshirani et al. (1999). Πειραματικές αξιολογήσεις αλγόριθμων συσταδοποίησης εγγράφων, βασισμένων επάνω στους ιεραρχικούς αλγορίθμους και σε αυτόν K-μέσων (K-means) , συνοψίζονται από τον Steinbach et al. (2000). Εκτός από τα έγγραφα που έχουν εκδοθεί στη λογοτεχνία υπάρχουν επίσης διάφορα σημαντικά βιβλία αφιερωμένα στη συσταδοποίηση. Πρόωρα περιεκτικά βιβλία περιλαμβάνουν αυτά των Aldenderfer και Blashfield (1984) . Anderberg (1973), Duran και Ondell (1974), Hartigan (1975) , Romesburg (1984) και Spath (1980). Οι Jain και Dubes (1988) παρέχουν μία εξαιρετική εισαγωγή και διαπραγμάτευση των κυρίων ιεραρχικών και τμηματικών αλγορίθμων συσταδοποίησης και άλλα σχετικά θέματα πάνω στο αντικείμενο, όπως η αναπαράσταση των δεδομένων και η επικύρωση των συστάδων. Ο Everitt et al. (2001) παρουσίασε την συσταδοποίηση σε αναγνώσιμη μορφή συνδυασμένη με πολλές πρακτικές εφαρμογές. Οι Kaufman και Rousseeuw (1990) ενώ περιγράφουν τις ιδιότητες των αλγορίθμων συσταδοποίησης, παρέχουν επίσης και οδηγίες για τη χρήση προγραμμάτων που παρέχουν τα μέσα για την εκτέλεση τέτοιων αλγορίθμων. Ένα άλλο βιβλίο που διαπραγματεύεται εκτενώς ο θέμα της συσταδοποίησης και είναι γραμμένο από τους Gordon (1999), McLachlan και Peel (2000) μελετά τη συσταδοποίηση πεπερασμένων μοντέλων προβλημάτων, ενώ ο Hoppner et all. (1999) ερευνά τη μέθοδο fuzzy clustering, ο Duda et al. (2001) συμβάλλει με ένα κεφάλαιο πάνω στη συσταδοποίηση με το εξαιρετικό βιβλίο πάνω στην αναγνώριση μοτίβων. Περισσότερα βιβλία που περιλαμβάνουν θέματα σχετικά με την συσταδοποίηση είναι των Bishop (1995) , Cherassky και Mulier (1998), Haykin (1999) και Theodoridis και Koutroumbas (2006). Συσταδοποίηση αποτελεί η τοποθέτηση αντικειμένων σε ομάδες (που ονομάζονται συστάδες, «*clusters»),* έτσι ώστε τα αντικείμενα που ανήκουν στην ίδια ομάδα να είναι ποιό όμοια μεταξύ τους από κάθε άλλο από τα αντικείμενα που ανήκουν σε διαφορετικές συστάδες. Συχνά η ομοιότητα αυτή αξιολογείται σύμφωνα με κάποιο μέτρο. Η συσταδοποίηση είναι μια κοινή τεχνική για την στατιστική ανάλυση δεδομένων ,που χρησιμοποιείται σε ένα μεγάλο εύρος τομέων.

2. Τύποι Δεδομένων

Οι αλγόριθμοι συσταδοποίησης είναι κατά πολύ εφαπτόμενοι σε σχέση με τα δεδομένα τα οποία είναι προς επεξεργασία, επομένως η κλίμακα κατανόησης, η εξομάλυνση και η εγγύτητα των δεδομένων είναι πολύ σημαντικά στην ερμηνεία

των αποτελεσμάτων των αλγορίθμων συσταδοποίησης. Ο όρος τύπος δεδομένων αναφέρεται στον βαθμό κβαντισμού των δεδομένων. Κάποιο από τα δεδομένα μπορεί να είναι δυαδικό, διακριτό ή συνεχές. Ένα δυαδικό δεδομένο έχει ακριβώς δύο πιθανές τιμές, όπως για παράδειγμα αληθές ή ψευδές. Ένα διακριτό δεδομένο έχει ένα πεπερασμένο αριθμό τιμών που μπορεί να πάρει, επομένως τα δυαδικά αποτελούν μία ειδική περίπτωση των διακριτών δεδομένων. Οι κλίμακες δεδομένων που δείχνουν τη σχετική σημασία των αριθμών είναι επίσης ένα σημαντικό στοιχείο για την διαδικασία της συσταδοποίησης. Οι κλίμακες δεδομένων μπορούν να χωριστούν σε ποιοτικές κλίμακες και ποσοτικές κλίμακες. Οι ποιοτικές κλίμακες περιλαμβάνουν τις ονομαστικές κλίμακες και τις τακτικές κλίμακες, ενώ οι ποσοτικές κλίμακες περιλαμβάνουν τις κλίμακες διαστήματος και τις κλίμακες αναλογίας.

2.1 Δεδομένα Κατηγορίας (Categorical Data)

Τα δεδομένα αυτά επίσης αναφέρονται και ως ονομαστικά, τα οποία χρησιμοποιούνται για ονόματα για μάρκες αυτοκινήτων και υποκαταστήματα τραπεζών. Αφού θεωρούμε τα σύνολα δεδομένων με ένα πεπερασμένο πλήθος δεδομένων ένα ονομαστικό χαρακτηριστικό μπορεί να έχει μόνο ένα πεπερασμένο αριθμό τιμών και έτσι επομένως να αποτελεί μία ειδική περίπτωση του διακριτού τύπου.

Records	Values
x_1	{A, A, A, A, B, B}
x_2	{A, A, A, A, C, D}
x_3	{A, A, A, A, D, C}
x_4	{B, B, C, C, D, C}
x_5	{B, B, D, D, C, D}

Δείγμα δεδομένων κατηγορίας.

2.2 Δυαδικά δεδομένα (Binary Data)

Ένα δυαδικό χαρακτηριστικό μπορεί να πάρει ακριβώς δύο τιμές, όπως ψευδές ή αληθές. Τα δυαδικά δεδομένα μπορούν να χωριστούν περεταίρω σε δύο κατηγορίες, συμμετρικά και ασύμμετρα. Στην περίπτωση των συμμετρικών δυαδικών δεδομένων, κάθε μία από τις δύο πιθανές τιμές είναι το ίδιο σημαντική, για παράδειγμα , «αρσενικό-θηλυκό» . Στην περίπτωση των ασύμμετρων μία τιμή έχει περισσότερη σημασία από την άλλη. Για παράδειγμα, το «Ναί» να δείχνει την

ύπαρξη μίας συγκεκριμένης ιδιότητας ενώ το «Όχι» να υποδηλώνει απλά την απουσία της.

2.3 Δεδομένα Συναλλαγών (Transaction Data)

Δεδομένου ενός συνόλου τιμών I= {I_1,I_2,....,I_m}, μία συναλλαγή είναι ένα υποσύνολο του I. Σύνολο δεδομένων συναλλαγής είναι ένα σύνολο συναλλαγών, το οποίο είναι:

$$D = \{t_i : t_i \subseteq I, \quad i = 1, 2, \ldots, n\}$$

Οι συναλλαγές μπορούν να αναπαρασταθούν από δυαδικά διανύσματα, στα οποία κάθε παρουσία εισόδου δηλώνει την παρουσία ή την απουσία του αντίστοιχου αντικειμένου. Για παράδειγμα, μπορούμε να αναπαραστήσουμε μία συναλλαγή t_i με ένα δυαδικό διάνυσμα (b_{i1},b_{i2},.....,b_{im}) όπου :

$$b_{ij} = 1 \text{ εάν } I_j \in t_i \text{ και } b_{ij} = 0 \text{ εάν } I_j \notin t_i$$

Από αυτή την οπτική γωνία, τα δεδομένα συναλλαγών μπορούν να θεωρηθούν ως μία ειδική περίπτωση των δυαδικών δεδομένων. Το ποιό απλό παράδειγμα δεδομένων συναλλαγών είναι τα Market-Basket δεδομένα, η οποία ανάλυση τους είναι μια τεχνική που βασίζεται στην μοντελοποίηση της θεωρίας ότι αν κάποιος αγοράσει κάποια συγκεκριμένη ομάδα ειδών, θα είναι περισσότερο (ή λιγότερο) πιθανό να αγοράσει μια άλλη ομάδα ειδών. Για παράδειγμα, αν κάποιος είναι σε ένα μπαρ και παρήγγειλε μία μπύρα αλλά δεν πήρε κάποιο μεζεδάκι είναι πιο πιθανόν να πάρει πατατάκια σε σχέση με κάποιον που δεν παρήγγειλε μπύρα. Το σύνολο των αντικειμένων που αγοράζει ο πελάτης αναφέρεται ως itemset**,** η ανάλυση market-basket επιδιώκει να βρει τις σχέσεις μεταξύ των αγορών. Συνήθως η σχέση θα είναι υπό τη μορφή ενός κανόνα: EAN (μπύρα, όχι μεζεδάκι ΤΟΤΕ)) {πατατάκια} .Σε ένα σύνολο Market-basket δεδομένων λοιπόν, μία συναλλαγή περιέχει ένα υποσύνολο από το συνολικό αριθμό των αντικειμένων διαθέσιμα προς πώληση. Για παράδειγμα τα ακόλουθα αποτελούν δύο συναλλαγές. [μήλο,πορτοκάλι] , [μήλο,αυγό,γάλα,ψάρι]. Γενικά, πολλές συναλλαγές αποτελούνται από αραιός κατανεμημένα αντικείμενα. Για παράδειγμα, ένας καταναλωτής μπορεί να αγοράσει μόνο μερικά από τα χιλιάδες προϊόντα ενός καταστήματος. Για συναλλαγές οι οποίες έγιναν για αραιός κατανεμημένα αντικείμενα, η ομοιότητα ανά ζευγάρια δεν είναι αναγκαία ούτε ικανή για να κρίνουμε αν οι συναλλαγές που περιέχει μία συστάδα είναι παρόμοιες.

2.4 Χρονοσειρές (Time Series)

Οι χρονοσειρές αποτελούν την πιο απλή μορφή των χρονικών δεδομένων. Επακριβώς, μία χρονοσειρά είναι μία ακολουθία από πραγματικούς αριθμούς που αναπαριστούν τις τιμές μία πραγματικής μεταβλητής σε ίσα χρονικά διαστήματα.

Για παράδειγμα η κινητικότητα της τιμής των μετοχών, η θερμοκρασία για κάποια δεδομένη τοποθεσία, και ο όγκος των πωλήσεων σε σχέση με τον χρόνο αποτελούν χρονοσειρές. Μία χρονοσειρά είναι διακριτή εάν η μεταβλητή μπορεί να καθοριστεί για ένα πεπερασμένο σύνολο χρονικών σημείων. Οι περισσότερες χρονοσειρές που συναντώνται στην συσταδοποίηση είναι πεπερασμένες χρονοσειρές. Όταν μία μεταβλητή ορίζεται για όλα τα σημεία του χρόνου, τότε αυτή η χρονοσειρά αποκαλείται συνεχής. Γενικά, μία χρονοσειρά μπορεί να θεωρηθεί ως μία μίξη των παρακάτω τεσσάρων συνιστωσών :

1. Μια τάση, μόδα δηλαδή, μια μακροπρόθεσμη κίνηση.
2. Διακυμάνσεις της τάσης, μόδας μεγαλύτερης ή μικρότερης συχνότητας.
3. Της εποχιακής συνιστώσας.
4. Του υπολοίπου ή ενός τυχαίου γεγονότος.

Αυτοί οι τύποι δεδομένων αποτελούν κάποιους βασικούς τύπους δεδομένων που συναντώνται στην συσταδοποίηση, όμως υπάρχουν ακόμα διάφοροι άλλοι τύποι δεδομένων όπως τα δεδομένα εικόνων.

3. Μέτρα Απόστασης και Πίνακας Γειτνίασης

3.1 Μέτρο Απόστασης (Distance Measure)

Ένα σημαντικό βήμα για κάθε συσταδοποίηση αποτελεί η επιλογή ενός μέτρου, το οποίο θα καθορίσει τον τρόπο με τον οποίο θα υπολογιστεί η ομοιότητα των δύο στοιχείων. Αυτό θα επηρεάσει τη διαμόρφωση των συστάδων, αφού ορισμένα στοιχεία μπορεί να είναι κοντά το ένα στο άλλο σύμφωνα με μία απόσταση και πιο μακριά, σύμφωνα με μια άλλη. Για παράδειγμα, σε ένα δυσδιάστατο χώρο, η απόσταση μεταξύ του σημείου A (x=1,y=0) και της αρχής των αξόνων (x=0,y=0) είναι πάντα 1 σύμφωνα με τους συνήθεις κανόνες, , αλλά η απόσταση μεταξύ του σημείου A (x=1,y=0) και της αρχής των αξόνων (x=0,y=0) μπορεί να είναι 2, $\sqrt[3]{2}$ ή 1 ανάλογα με άλλους κανόνες απόστασης.

3.2 Πίνακας Γειτνίασης (Proximity Matrix)

Ο πίνακας γειτνίασης είναι ένας πίνακας που περιέχει τους ανά ζεύγη δείκτες εγγύτητας ενός συνόλου δεδομένων. Συνήθως οι πίνακες γειτνίασης είναι συμμετρικοί. Έχουμε μεταξύ αυτών τους πίνακες ομοιότητας ή ανομοιότητας.

Δεδομένου ενός συνόλου δεδομένων D = $\{x_1,x_2,.....,x_n\}$, κάθε αντικείμενο του οποίου περιγράφεται από ένα δ-διαστάσεων χαρακτηριστικό διάνυσμα, ο πίνακας αποστάσεως για το σύνολο δεδομένων D καθορίζεται ως :

$$Mdist(D) = \begin{pmatrix} 0 & d_{12} & \dots & d_{1n} \\ d_{21} & 0 & \dots & d_{2n} \\ . & . & \dots & . \\ . & . & \dots & . \\ d_{n1} & d_{n2} & \dots & 0 \end{pmatrix}$$

Όπου $d_{ij} = d(x_i,x_j)$ σε σχέση με κάποια συνάρτηση απόστασης d(-,-).

Ο πίνακας αποστάσεως αποτελεί ένα παράδειγμα πίνακα γειτνίασης.

3.3 Συνήθη Μετρά Αποστάσεων

α. Ευκλείδεια Απόσταση (Euclidean Distance)

Η ευκλείδεια απόσταση είναι πιθανόν η πιο συχνά χρησιμοποιούμενη για αριθμητικά δεδομένα. Για δύο σημεία x και y σε ένα δ-διάστατο χώρο, η ευκλείδεια απόσταση μεταξύ τους δίνεται από την σχέση :

$$d_{euc}(x,y) = \left[\sum_{j=1}^{d}(x_j - y_j)^2\right]^{\frac{1}{2}}$$

Όπου το x_j και y_j είναι τιμές του j_{th} στοιχείου των x και y, αντίστοιχα.

Η τετραγωνισμένη ευκλείδεια απόσταση δίνεται από τον τύπο :

$$d_{euc}(x,y) = d_{euc}(x,y)^2 = \sum_{j=1}^{d}(x_j - y_j)^2$$

Η τετραγωνισμένη απόσταση δεν είναι στην πραγματικότητα απόσταση.

β. Απόσταση Manhattan (Manhattan Distance)

Η απόσταση Manhattan ονομάζεται επίσης και “City block Distance” και ορίζεται ως το άθροισμα των αποστάσεων όλων των στοιχείων. Δηλαδή, για δύο σημεία x και y σε ένα δ-διάστατο χώρο, η απόσταση Manhattan μεταξύ τους δίνεται από τον τύπο.

$$d_{man}(x,y) = \sum_{j=1}^{d} |x_j - y_j|$$

γ. Μέγιστη Απόσταση ή Απόσταση Chebyshev (Maximum Distance, Chebyshev Distance)

Η μέγιστη απόσταση ορίζεται ως η μέγιστη τιμή της απόστασης των στοιχείων. Δηλαδή, για δύο σημεία x και y σε ένα δ-διάστατο χώρο, η μέγιστη απόσταση μεταξύ τους δίνεται από τον τύπο :

$$d_{max(x,y)} = \max_{1 \le j \le d} |x_j - y_j|$$

Είναι επίσης γνωστή και ως απόσταση σκακιέρας αφού σε ένα παιχνίδι σκακιού, ο ελάχιστος αριθμός κινήσεων που χρειάζεται ο βασιλιάς για να μετακινηθεί από ένα τετράγωνο σε ένα άλλο ισούται με την απόσταση Chebyshev μεταξύ των κέντρων τους εάν τα τετράγωνα έχουν μήκος πλευράς ένα με άξονες ευθυγραμμισμένους στις άκρες της σκακιέρας.

δ. Απόσταση Minkowski (Minkowsi Distance)

Η ευκλείδεια απόσταση, η απόσταση Manhattan και η μέγιστη απόσταση αποτελούν τρεις συγκεκριμένες περιπτώσεις της απόστασης Minkowski η οποία ορίζεται ως :

$$d_{min(x,y)} = \left(\sum_{j=1}^{d} |x_j - y_j|^r \right)^{\frac{1}{r}}, \qquad r \geq 1$$

Το r ονομάζεται τάξη για την παραπάνω απόσταση Minkowski. Ας σημειωθεί ότι εάν πάρουμε r = 2, 1 και ∞ , θα πάρουμε την ευκλείδεια απόσταση, την απόσταση Manhattan και την μέγιστη απόσταση αντίστοιχα. Εάν το σύνολο των δεδομένων

έχει συμπαγής ή απομονωμένες συστάδες, τότε η απόσταση Minkowski λειτουργεί καλά, ειδάλλως ένα στοιχείο μεγάλης κλίμακας τείνει να επικρατεί μεταξύ των άλλων. Για να το αποφύγουμε, πρέπει να κανονικοποιήσουμε τα στοιχεία ή να χρησιμοποιήσουμε σταθμισμένα συστήματα.

4. Ιεραρχικοί Αλγόριθμοι Συσταδοποίησης (Hierarchical Clustering Algorithms)

4.1 Συσσωρευτική Ιεραρχική Συσταδοποίηση (Agglomerative Hierarchical Clustering)

Αυτή η περίπτωση ιεραρχικής συσταδοποίησης ξεκινά με κάθε αντικείμενο να τοποθετείται σε μία ξεχωριστή συστάδα. Έπειτα ο αλγόριθμος συγχωνεύει σε κάθε βήμα το κοντινότερο ζεύγος συστάδων (ή αλλιώς και σημείων διότι στην αρχή κάθε σημείο αποτελεί και μία ξεχωριστή συστάδα) ανάλογα με κάποιο κριτήριο ομοιότητας μέχρι όλα τα δεδομένα τα γίνουν μία συστάδα. Υπάρχουν μερικά μειονεκτήματα όσον αφορά την συσσωρευτική ιεραρχική συσταδοποίηση όπως :

- Τα σημεία τα οποία ομαδοποιήθηκαν λανθασμένα στα αρχικά στάδια της συσταδοποίησης δεν μπορούν να ανακατανεμηθούν
- Η χρήση διαφορετικών μέτρων ομοιότητας για τον καθορισμό της ομοιότητας μεταξύ των συστάδων μπορεί να οδηγήσει σε διαφορετικά αποτελέσματα.

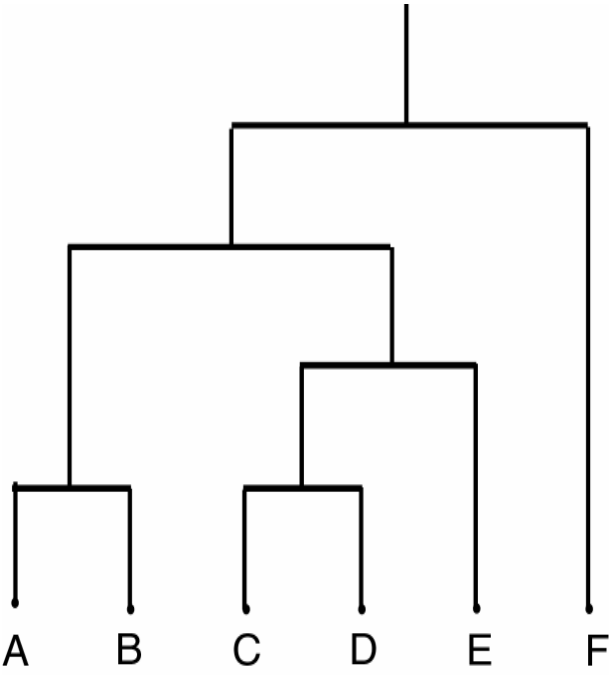
4.2 Διαιρετική Ιεραρχική Συσταδοποίηση (Divisive Hierarchical Clustering)

Εάν αντιμετωπίσουμε την συσσωρευτική ιεραρχική συσταδοποίηση σαν μία από βάση έως κορυφή (bottom-up) μέθοδο, τότε η διαιρετική ιεραρχική συσταδοποίηση μπορεί να θεωρηθεί σαν μία από κορυφή έως βάση αντιμετώπιση (top-down). Η διαιρετική ιεραρχική συσταδοποίηση ξεκινά με όλα τα αντικείμενα σε μία συστάδα και επαναλαμβάνει τη διαδικασία διαχωρισμού μεγάλων συστάδων σε μικρότερα

κομμάτια. Η διαιρετική ιεραρχική συσταδοποίηση έχει κάποια μειονεκτήματα όπως και η συσσωρευτική ιεραρχική συσταδοποίηση.

4.3 Δενδρόγραμμα (Dendrogram)

Η ιεραρχική συσταδοποίηση χτίζει (agglomerative) ή διαιρεί (divisive) μία ιεραρχία από συστάδες. Αυτή η ιεραρχία συνήθως αναπαριστάται με ένα σχήμα το οποίο ονομάζεται δενδρόγραμμα, με όλα τα στοιχεία χωριστά στο ένα άκρο και μία συστάδα να περιέχει όλα τα στοιχεία στο άλλο. Οι συσσωρευτικοί αλγόριθμοι ξεκινούν από τα φύλλα αυτού του δένδρου ενώ οι διαιρετικοί από τη ρίζα. Εάν κόψουμε αυτό το δένδρο σε ένα συγκεκριμένο ύψος θα μας δώσει συσταδοποίηση συγκεκριμένης ακρίβειας.

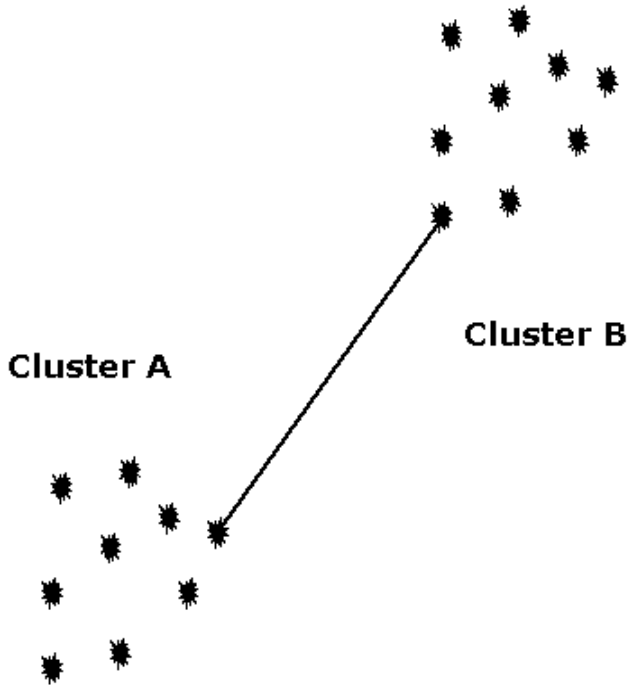


Εικόνα 2. Στην παραπάνω εικόνα φαίνεται ένα τυπικό δενδρόγραμμα.

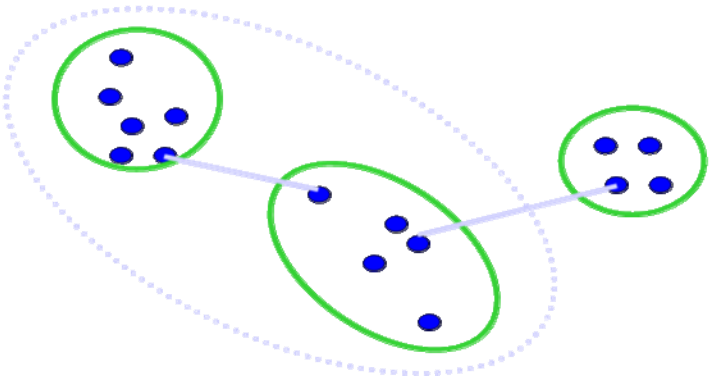
- Βλέπουμε πως στο ένα άκρο υπάρχουν όλα τα σημεία {A,B,C,D,E,F} και στην κορυφή τελικά όλα τα σημεία συγχωνεύονται σε μία συστάδα.
- Κάθε επίπεδο του δενδρογράμματος δείχνει τις συστάδες για αυτό το επίπεδο.
    - Φύλλο – Οι διάφορες συστάδες
    - Ρίζα – Μία συστάδα.
- Μια συστάδα σε ένα επίπεδο προκύπτει από την ένωση των συστάδων στα προηγούμενα επίπεδα.

4.4 Συσταδοποίηση Απλού Δεσμού (Single Link Method)

Η συσταδοποίηση απλού δεσμού είναι πιθανότατα η πιο γνωστή από τις ιεραρχικές μεθόδους και λειτουργεί συγχωνεύοντας, σε κάθε στάδιο, τα δύο πιο όμοια αντικείμενα, που δεν είναι ακόμη στην ίδια συστάδα. Το όνομα «απλού δεσμού» αναφέρεται στην συγχώνευση ενός ζεύγους συστάδων με κριτήριο την κοντινότερη απόσταση μεταξύ τους. Έτσι σε κάθε βήμα συγχωνεύουμε τις δύο συστάδες των οποίων τα μέλη έχουν τη μικρότερη απόσταση.



Εικόνα 3. Μέθοδος Απλού Δεσμού (Single Link)



Εικόνα 4. Παράδειγμα απλού δεσμού. (Single Link)

Παράδειγμα (Μέθοδος Απλού Δεσμού)

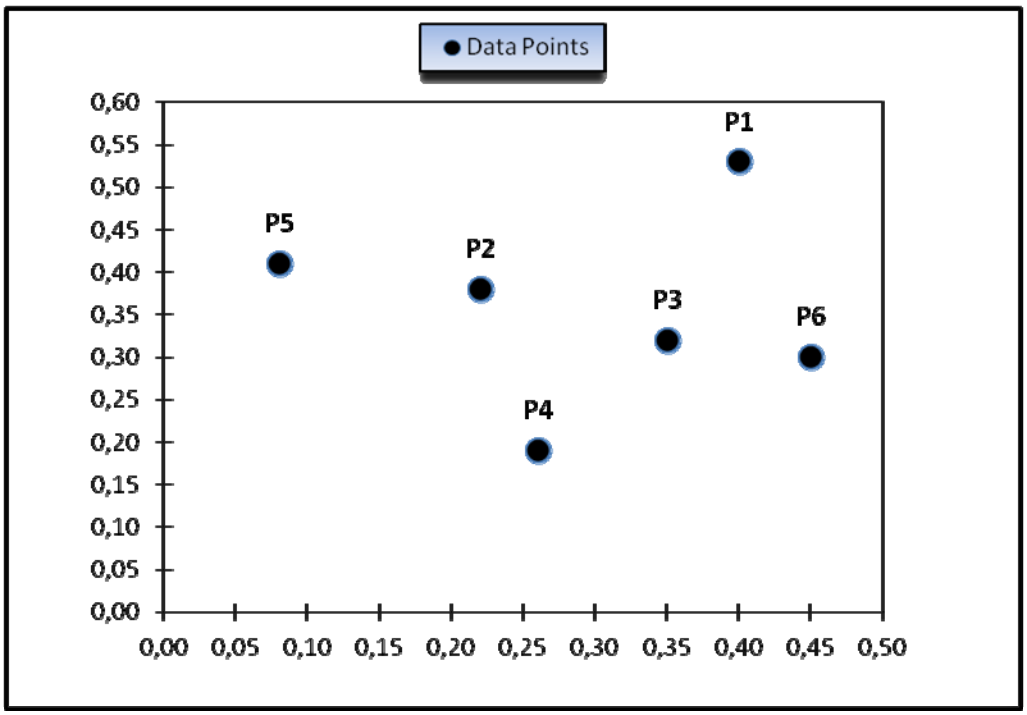
Πρόβλημα: Ας υποθέσουμε ότι η βάση δεδομένων D προς επεξεργασία δίνεται από τον πίνακα παρακάτω. Ακολουθώντας την μέθοδο απλού δεσμού να βρούμε τις συστάδες στην βάση δεδομένων D χρησιμοποιώντας ως μέτρο την Ευκλείδεια απόσταση.

Τα δεδομένα της βάσης D

	X	**Y**
p1	0.40	0.53
p2	0.22	0.38
p3	0.35	0.32
p4	0.26	0.19
p5	0.08	0.41
p6	0.45	0.30

Λύση:

Βήμα 1. Κάνουμε την γραφική παράσταση των σημείων στον ν-διάστατο χώρο (όπου ν είναι ο αριθμός των χαρακτηριστικών), στην περίπτωσή μας είναι 2, το x και y. Επομένως κάνουμε την γραφική παράσταση των σημείων p1, p2, … p6 στον δυσδιάστατο χώρο:



Βήμα 2. Υπολογίζουμε την απόσταση κάθε σημείου από όλα τα άλλα χρησιμοποιώντας τον τύπο της ευκλείδειας απόστασης και τοποθετούμε τα νούμερα στον πίνακα αποστάσεως.

Ανακαλούμε ότι ο τύπος της ευκλείδειας απόστασης μεταξύ δύο σημείων i και j είναι:

$$d(i,j) = \sqrt{|x_{i1} - x_{j1}|^2 + |x_{i2} - x_{j2}|^2 + \ldots + |x_{ip} - x_{jp}|^2}$$

όπου x_{i1} είναι η τιμή του χαρακτηριστικού 1 για i και x_{j1} είναι η τιμή του χαρακτηριστικού 1 για j, και τα λοιπά, για όσα χαρακτηριστικά έχουμε.

Στην περίπτωσή μας, έχουμε δύο χαρακτηριστικά. Επομένως η ευκλείδεια απόσταση μεταξύ των σημείων μας p1 και p2,τα οποία έχουν χαρακτηριστικά x και y υπολογίζεται ως έχει παρακάτω:

$$d(p1,p2) = \sqrt{|x_{p1} - x_{p2}|^2 + |y_{p1} - y_{p2}|^2}$$

$$= \sqrt{|0.40 - 0.22|^2 + |0.53 - 0.38|^2}$$

$$= \sqrt{|0.18|^2 + |0.15|^2}$$

$$= \sqrt{0.0324 + 0.0225}$$

$$= \sqrt{0.0549}$$

$$= 0.2343$$

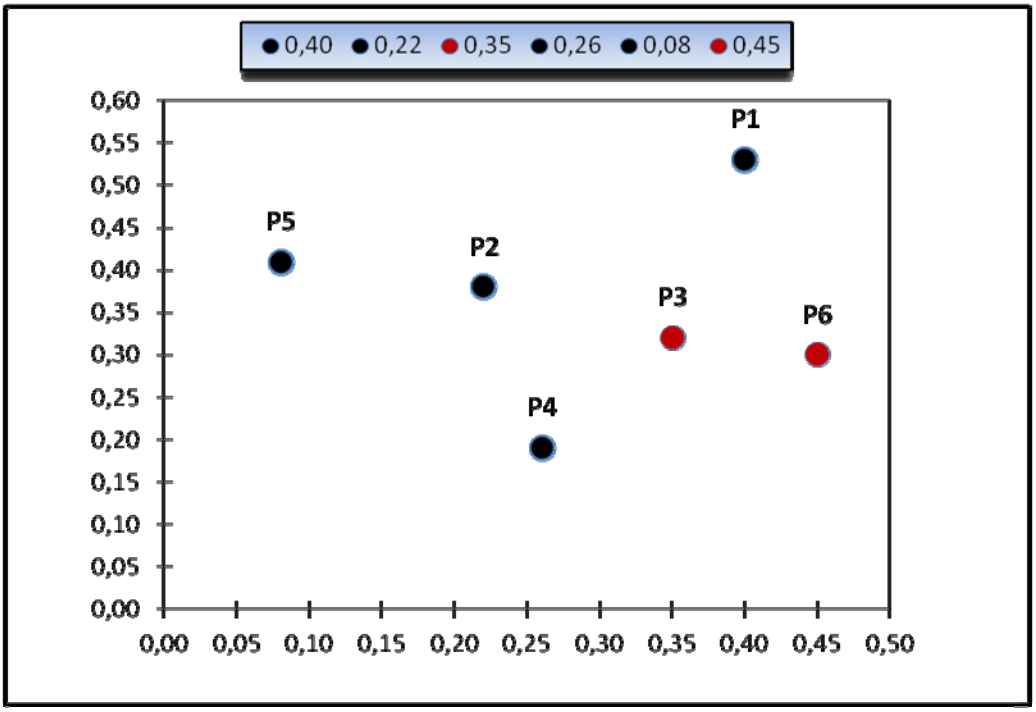
Ανάλογα, υπολογίζουμε την απόσταση των υπόλοιπων σημείων και παίρνουμε τις ακόλουθες τιμές στον πίνακα αποστάσεως:

Πίνακας Αποστάσεως.

	p1	p2	p3	p4	p5	p6
p1	0	0.2343	0.2159	0.3677	0.3418	0.2354
p2	0	0	0.1432	0.1942	0.1432	0.2435
p3	0	0	0	0.1581	0.2846	0.1020
p4	0	0	0	0	0.2843	0.2195
p5	0	0	0	0	0	0.3860
p6	0	0	0	0	0	0

Βήμα 3 Βρίσκουμε στον πίνακα τις δύο συστάδες με την μικρότερη απόσταση, και τις συγχωνεύουμε σε μία. Υπολογίζουμε ξανά τις τιμές για τον πίνακα αποστάσεως αφού αυτές οι δύο συστάδες είναι πλέον μία (δεν υφίστανται πλέον σαν μονάδες).

Κοιτώντας τον πίνακα αποστάσεως παραπάνω, βλέπουμε ότι τα σημεία p3 και p6 είναι αυτά τα οποία έχουν τη μικρότερη απόσταση μεταξύ όλων των άλλων - 0.1020 ,Επομένως συγχωνεύουμε αυτά τα δύο σε μία συστάδα και υπολογίζουμε ξανά τον πίνακα αποστάσεως.



Πίνακας Αποστάσεως

	p1	p2	{p3,p6}	p4	p5
p1	0	0.2343	0.2159	0.3677	0.3418
p2	0	0	0.1432	0.1942	0.1432
{p3,p6}	0	0	0	0.1581	0.2846
p4	0	0	0	0	0.2843
p5	0	0	0	0	0

Αφού συγχωνεύσαμε τα σημεία (p3, p6) μαζί σε μία συστάδα θα έχουμε μία εγγραφή στον πίνακα αποστάσεως για αυτά.

Επομένως δε θα έχουμε πλέον τα σημεία p3 ή p6 ξεχωριστά. Θέλουμε έτσι λοιπόν να υπολογίσουμε την απόσταση της νέας μας συστάδας - (p3, p6) από όλα τα υπόλοιπα σημεία. Ανακαλούμε το γεγονός ότι στην μέθοδο απλού δεσμού η ομοιότητα μεταξύ δύο συστάδων καθορίζεται από την κοντινότερη απόσταση μεταξύ δύο οποιοδήποτε σημείων που ανήκουν στις δύο αυτές συστάδες. Επομένως, η απόσταση για παράδειγμα της (p3, p6) από το p1 υπολογίζεται ως εξής:

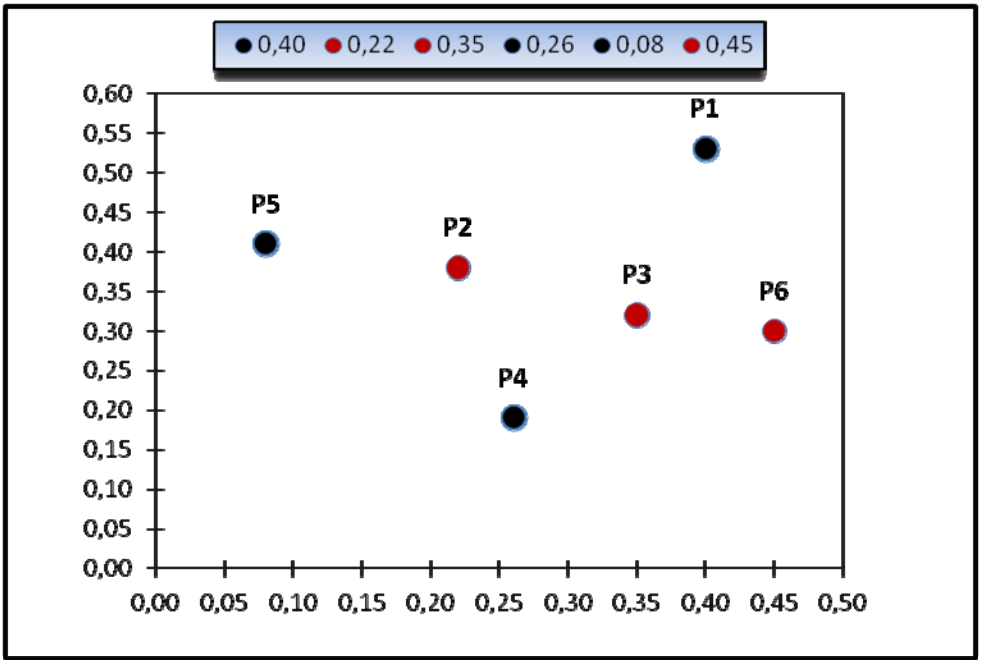
dist((p3, p6), p1) = MIN (*dist*(p3, p1) , *dist*(p6, p1))

= MIN (0.2159 , 0.2354)

= 0.2159

Βήμα 4 Επανάληψη του βήματος 3 μέχρι όλες οι συστάδες (ή γενικά όλα τα σημεία) να συγχωνευθούν σε μία συστάδα.

α. Έτσι, κοιτώντας τον πιο πρόσφατο πίνακα αποστάσεως παραπάνω, βλέπουμε πως τα σημεία p2 και p5 είναι εκείνα με την μικρότερη απόσταση - 0.1432, επίσης όμως βλέπουμε ότι και τα σημεία p2 και (p3,p6) έχουν την ίδια απόσταση - 0.1432. Στην περίπτωση αυτή, μπορούμε να επιλέξουμε οποιοδήποτε από τα δύο. Ας επιλέξουμε τα p2 και (p3,p6).Τα συγχωνεύουμε σε μία συστάδα και υπολογίζουμε ξανά τον πίνακα αποστάσεως.



Πίνακας Αποστάσεως

	p1	{p3,p6,p2}	p4	p5
p1	0	0.2159	0.3677	0.3418
{p3,p6,p2}	0	0	0.1581	0.1432

p4	0	0	0	0.2843
p5	0	0	0	0

Αφού συγχωνεύσαμε τα p2 και (p3, p6) μαζί σε μία συστάδα έχουμε πλέον μία εγγραφή για αυτά στον πίνακα αποστάσεως.

Τα σημεία p2 και (p3,p6) δεν υπάρχουν πλέον σαν μονάδες. Επομένως πρέπει να υπολογίσουμε την απόσταση της νέας συστάδας από όλα τα υπόλοιπα σημεία/συστάδες. Η απόσταση μεταξύ των (p3, p6,p2) και p5 υπολογίζεται ως εξής:

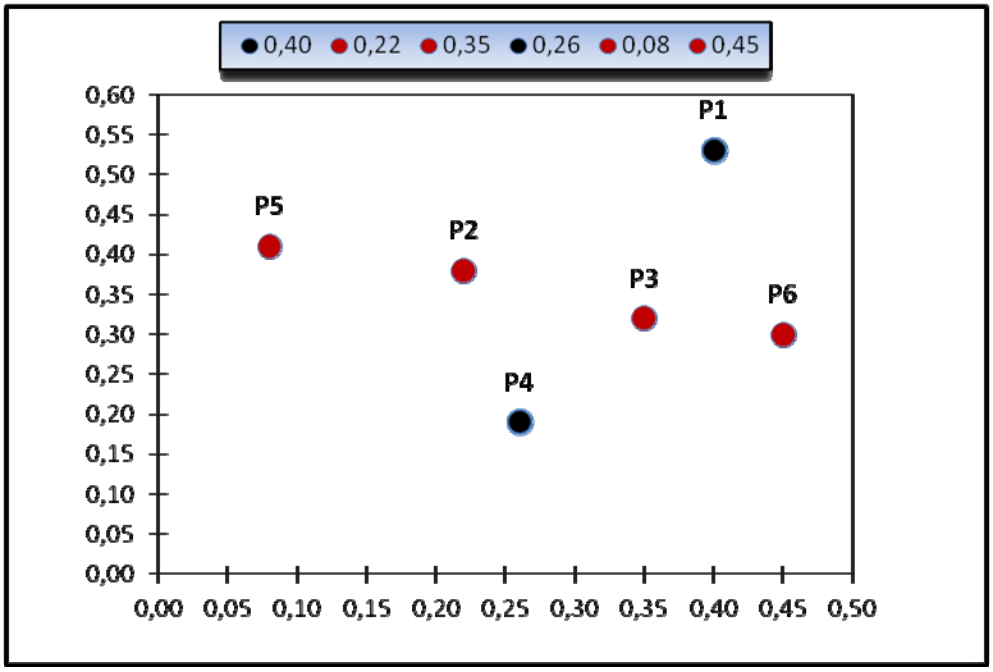
dist((p3, p6,p2), p5) = MIN (*dist*(p3, p5) , *dist*(p6, p5), *dist*(p2, p5))

= MIN (0.2846 , 0.3860, 0.1432)

= 0.1432

β. Αφού έχουμε και άλλες διαθέσιμες συστάδες για συγχώνευση, συνεχίζουμε να επαναλαμβάνουμε το βήμα 3.

Έτσι, βλέποντας τον πιο πρόσφατο πίνακα αποστάσεως παραπάνω διαπιστώνουμε ότι οι συστάδες (p3,p6,p2) και p5 έχουν τη μικρότερη απόσταση όλων - 0.1432. Επομένως συγχωνεύουμε αυτά τα δύο σε μία συστάδα και υπολογίζουμε ξανά τον πίνακα αποστάσεως.

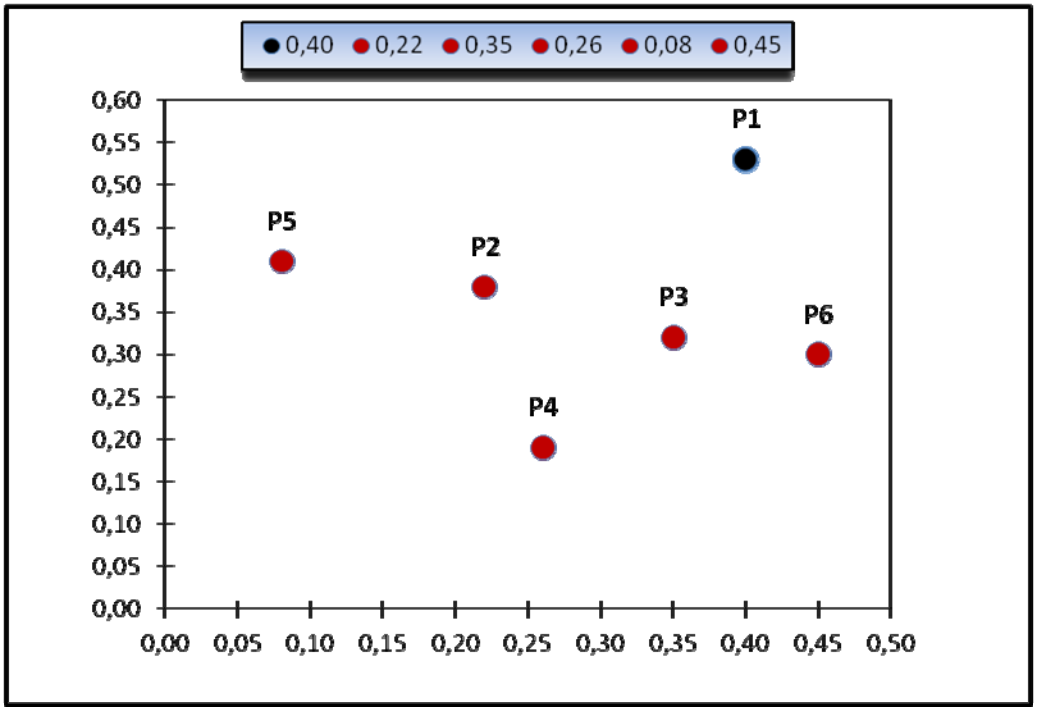


Πίνακας Αποστάσεως

	p1	{p3,p6,p2,p5}	p4
p1	0	0.2159	0.3677
{p3,p6,p2,p5}	0	0	0.1581
p4	0	0	0

γ. Αφού έχουμε και άλλες διαθέσιμες συστάδες για συγχώνευση, συνεχίζουμε να επαναλαμβάνουμε το βήμα 3.

Επομένως, βλέποντας τον πιο πρόσφατο πίνακα αποστάσεως παραπάνω, παρατηρούμε πως τα (p3, p6, p2, p5) και p4 έχουν τη μικρότερη απόσταση όλων - 0.1581 . Έτσι συγχωνεύουμε αυτά τα δύο σε μία συστάδα και υπολογίζουμε ξανά τον πίνακα αποστάσεως.

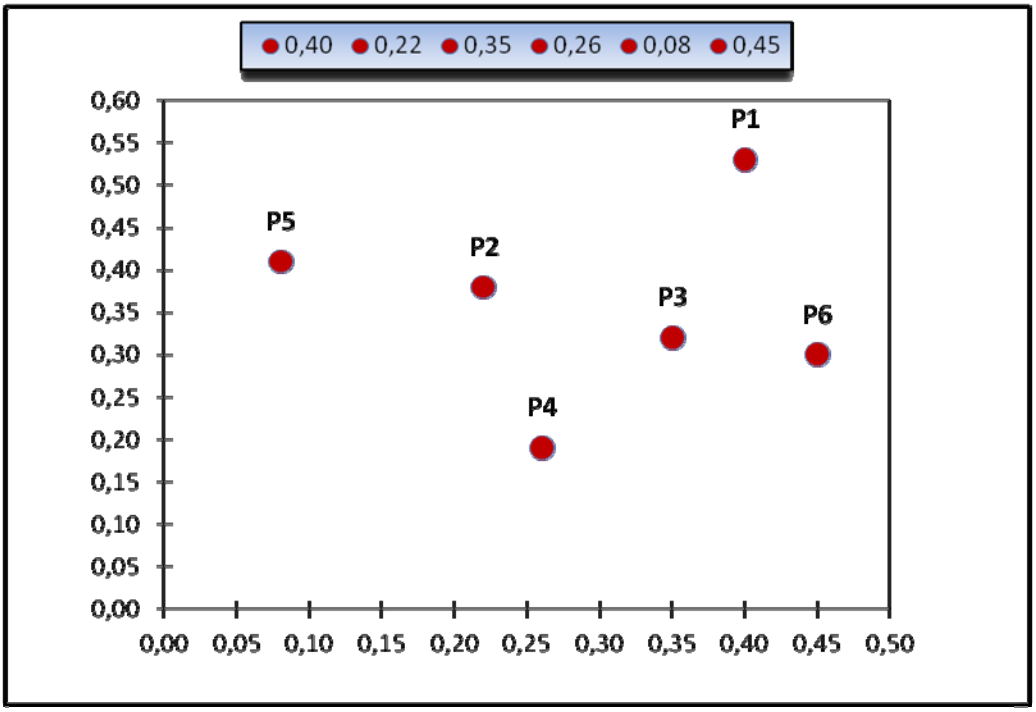


Πίνακας Αποστάσεως

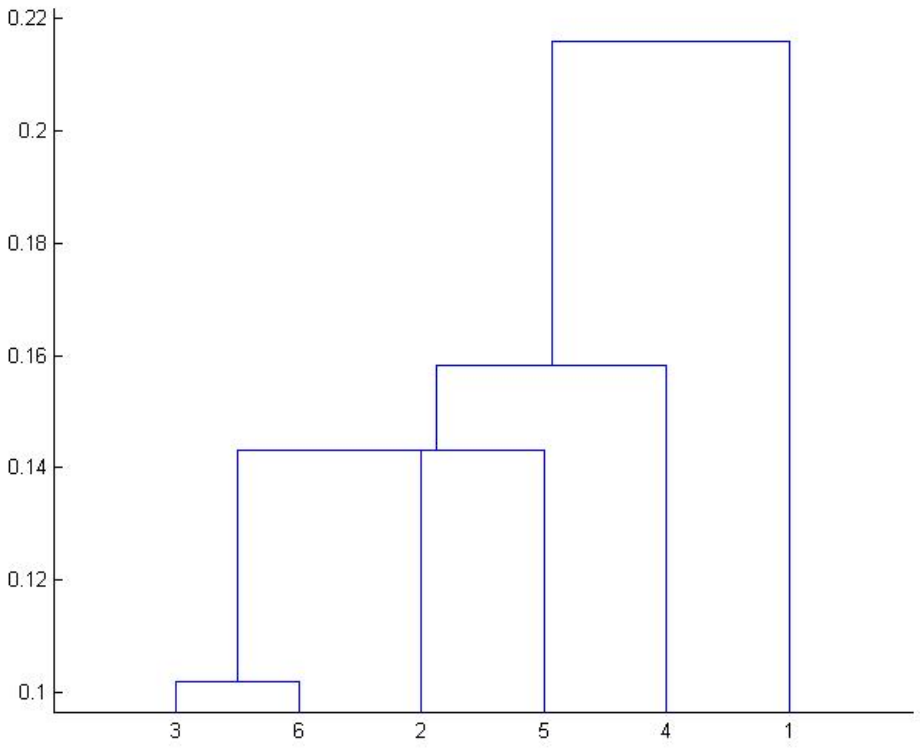
	p1	{p3,p6,p2,p5,p4}
p1	0	0.2159
{p3,p6,p2,p5,p4}	0	0

δ. Αφού έχουμε και άλλες διαθέσιμες συστάδες για συγχώνευση, συνεχίζουμε να επαναλαμβάνουμε το βήμα 3.

Επομένως, βλέποντας τον πιο πρόσφατο πίνακα αποστάσεως παραπάνω, παρατηρούμε ότι τα (p3, p6, p2, p5, p4) και p1 έχουν τη μικρότερη απόσταση όλων - 0.2159 (η τελευταία που απομένει στον πίνακα). Έτσι συγχωνεύουμε αυτά τα δύο σε μία συστάδα. Τώρα πλέον δεν υπάρχει λόγος να υπολογίσουμε τον πίνακα αποστάσεως διότι δεν υπάρχουν άλλες συστάδες για συγχώνευση.


0,40
0,22
0,35
0,26
0,08
0,45
0,60
0,55
0,50
0,45
0,40
0,35
0,30
0,25
0,20
0,15
0,10
0,05
0,00
0,00
0,05
0,10
0,15
0,20
0,25
0,30
0,35
0,40
0,45
0,50
P1
P2
P3
P4
P5
P6

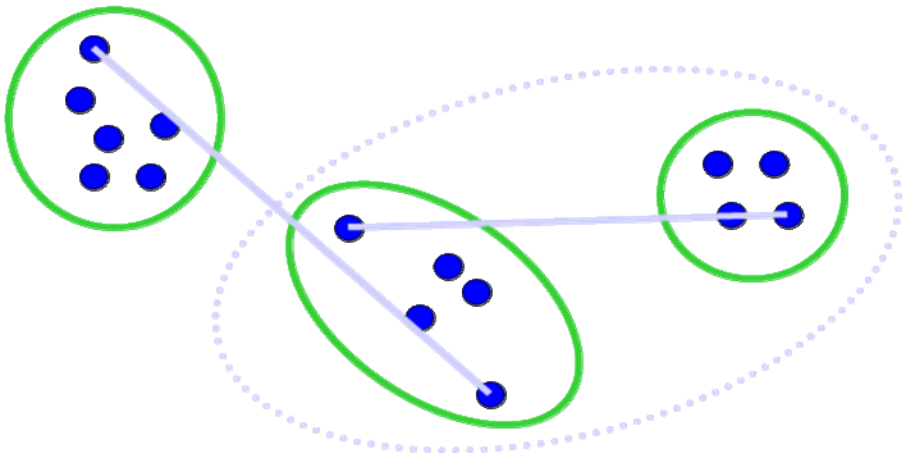
Το δενδρόγραμμα για το παράδειγμα που μελετήσαμε φαίνεται παρακάτω.



Εικόνα 5. Δενδρόγραμμα Μεθόδου Απλού Δεσμού.

4.5 Συσταδοποίηση Πλήρους Δεσμού (Complete Link Method)

Στην συσταδοποίηση πλήρους δεσμού, η ομοιότητα δύο συστάδων για συγχώνευση, είναι η ομοιότητα των δύο πιο μή όμοιων, απομακρυσμένων μελών τους.



Εικόνα 6. Παράδειγμα Πλήρους Δεσμού

Παράδειγμα (Μέθοδος Πλήρους Δεσμού)

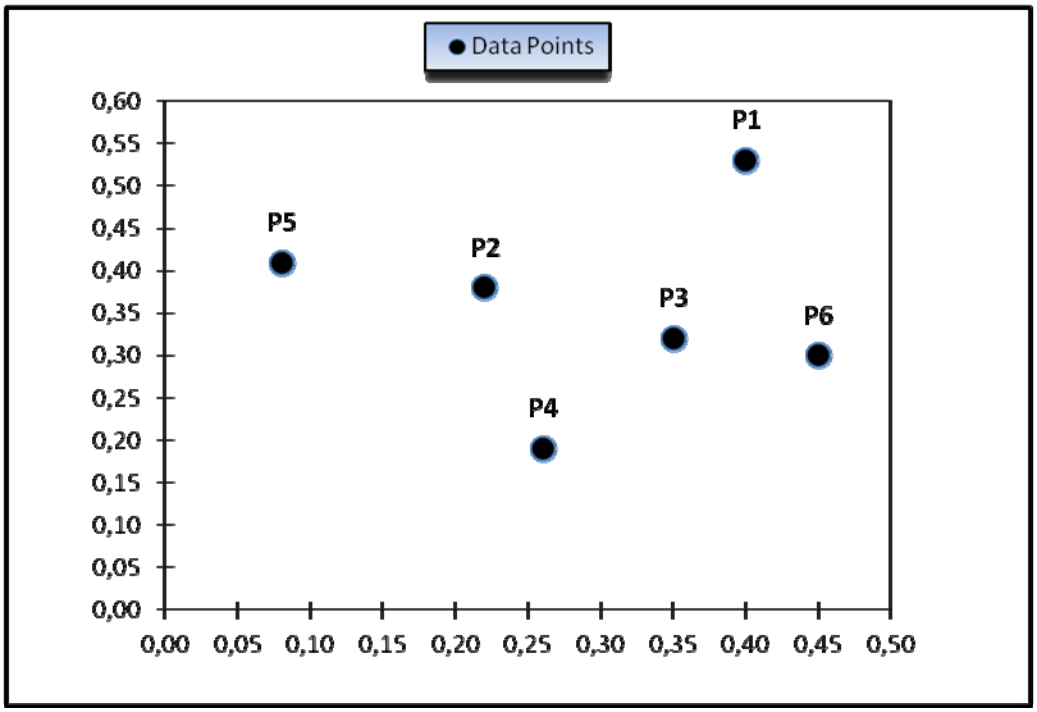
Πρόβλημα: Ας υποθέσουμε ότι η βάση δεδομένων D προς επεξεργασία δίνεται από τον πίνακα παρακάτω. Ακολουθώντας την μέθοδο πλήρους δεσμού να βρούμε τις συστάδες στην βάση δεδομένων D χρησιμοποιώντας ως μέτρο την Ευκλείδεια απόσταση.

Τα δεδομένα της βάσης D

	X	**y**
p1	0.40	0.53
p2	0.22	0.38
p3	0.35	0.32
p4	0.26	0.19
p5	0.08	0.41
p6	0.45	0.30

Λύση:

Βήμα 1. Κάνουμε την γραφική παράσταση των σημείων στον ν-διάστατο χώρο (όπου ν είναι ο αριθμός των χαρακτηριστικών), στην περίπτωσή μας είναι 2, το x και y.Επομένως κάνουμε την γραφική παράσταση των σημείων p1, p2, … p6 στον δυσδιάστατο χώρο:



Βήμα 2. Υπολογίζουμε την απόσταση κάθε σημείου από όλα τα άλλα χρησιμοποιώντας τον τύπο της ευκλείδειας απόστασης και τοποθετούμε τα νούμερα στον πίνακα αποστάσεως.

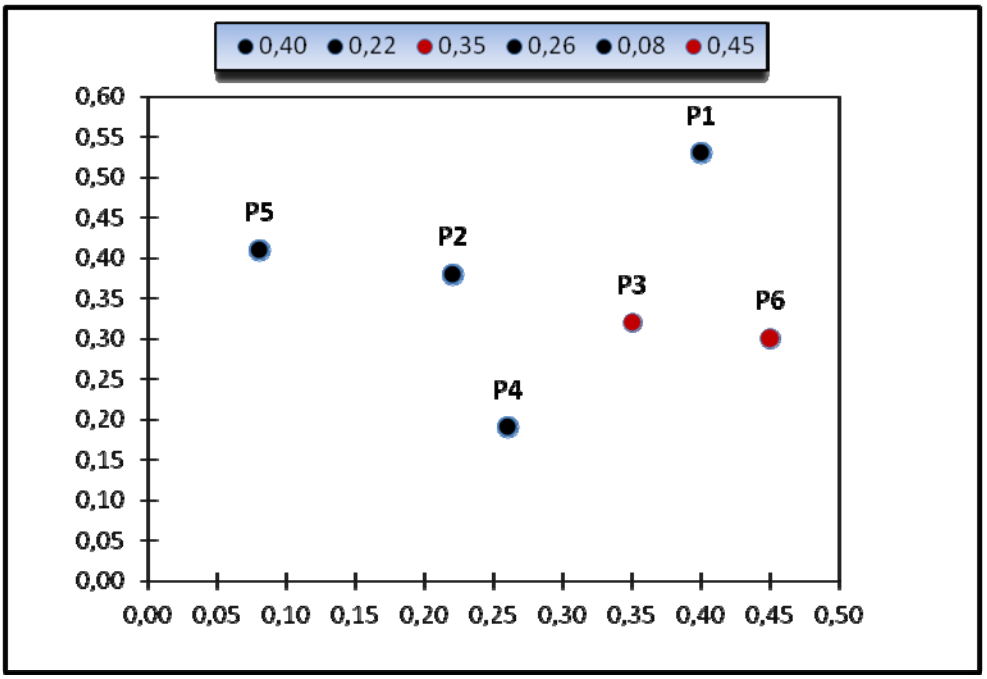
Πίνακας Αποστάσεως

	p1	p2	p3	p4	p5	p6
p1	0	0.2343	0.2159	0.3677	0.3418	0.2354
p2	0	0	0.1432	0.1942	0.1432	0.2435
p3	0	0	0	0.1581	0.2846	0.1020
p4	0	0	0	0	0.2843	0.2195

p5	0	0	0	0	0	0.3860
p6	0	0	0	0	0	0

Βήμα 3 Βρίσκουμε στον πίνακα τις δύο συστάδες με την μικρότερη απόσταση, και τις συγχωνεύουμε σε μία. Υπολογίζουμε ξανά τις τιμές για τον πίνακα αποστάσεως αφού αυτές οι δύο συστάδες είναι πλέον μία (δεν υφίστανται πλέον σαν μονάδες).

Κοιτώντας τον πίνακα αποστάσεως παραπάνω, βλέπουμε ότι τα σημεία p3 και p6 είναι αυτά τα οποία έχουν τη μικρότερη απόσταση μεταξύ όλων των άλλων - 0.1020 ,Επομένως συγχωνεύουμε αυτά τα δύο σε μία συστάδα και υπολογίζουμε ξανά των πίνακα αποστάσεως.



Πίνακας Αποστάσεως

	p1	p2	{p3,p6}	p4	p5
p1	0	0.2343	0.2354	0.3677	0.3418
p2	0	0	0.2435	0.1942	0.1432
{p3,p6}	0	0	0	0.2195	0.3860

p4	0	0	0	0	0.2843
p5	0	0	0	0	0

Θέλουμε έτσι λοιπόν να υπολογίσουμε την απόσταση της νέας μας συστάδας - (p3, p6) από όλα τα υπόλοιπα σημεία. Ανακαλούμε το γεγονός ότι στην μέθοδο πλήρους δεσμού η ομοιότητα μεταξύ δύο συστάδων καθορίζεται από την κοντινότερη απόσταση των δύο πιο απομακρυσμένων σημείων που ανήκουν στις δύο αυτές συστάδες. Έτσι, η απόσταση για παράδειγμα της (p3, p6) από το p5 υπολογίζεται ως εξής :

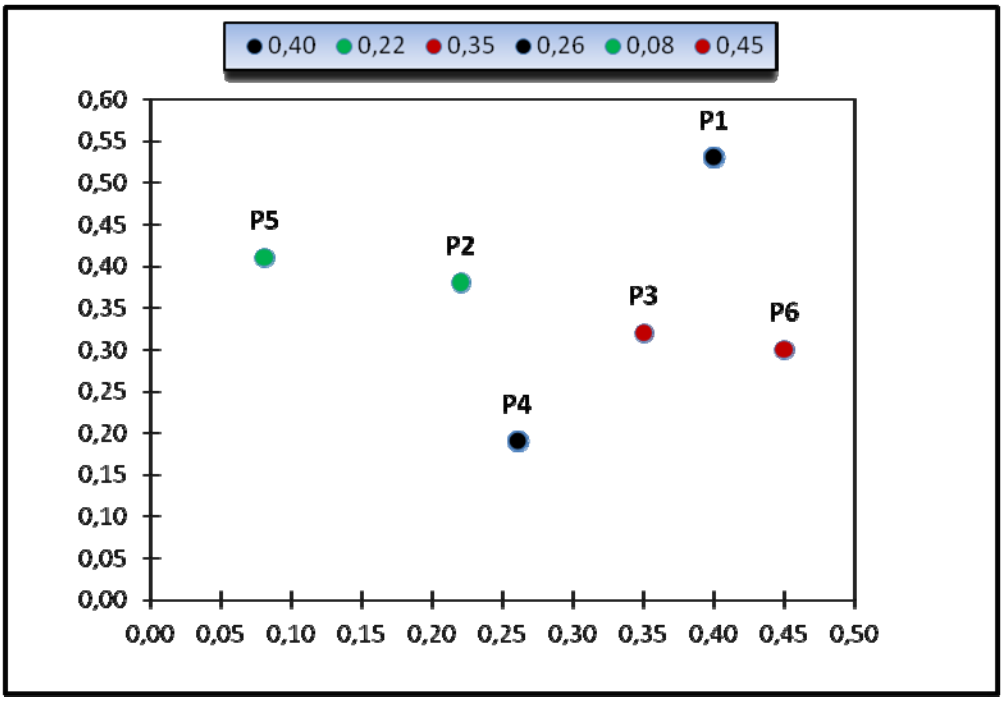
dist((p3, p6), p5) = MAX (*dist*(p3, p5) , *dist*(p6, p5))

= MAX (0.2846 , 0.3860)

= 0.3860

Βήμα 4 Επανάληψη του βήματος 3 μέχρι όλες οι συστάδες (ή γενικά όλα τα σημεία) να συγχωνευθούν σε μία συστάδα.

α. Κοιτώντας τον πιο πρόσφατο πίνακα αποστάσεως παραπάνω, βλέπουμε πως τα σημεία p2 και p5 είναι εκείνα με την μικρότερη απόσταση - 0.1432.Τα συγχωνεύουμε σε μία συστάδα και υπολογίζουμε ξανά τον πίνακα αποστάσεως.



Υπολογίζοντας την απόσταση της νέας μας συστάδας - (p2, p5) από όλα τα υπόλοιπα σημεία. Ανακαλούμε το γεγονός ότι στην μέθοδο πλήρους δεσμού η ομοιότητα μεταξύ δύο συστάδων καθορίζεται από την κοντινότερη απόσταση μεταξύ των δύο πιο απομακρυσμένων σημείων τους. Έτσι για παράδειγμα :

dist((p3, p6), {p2,p5}) = MAX (*dist*(p3, p2) , *dist*(p3, p5), *dist*(p6, p2), *dist*(p6, p5))

= MAX (0.1432 , 0.2846, 0.2435, 0.3860)

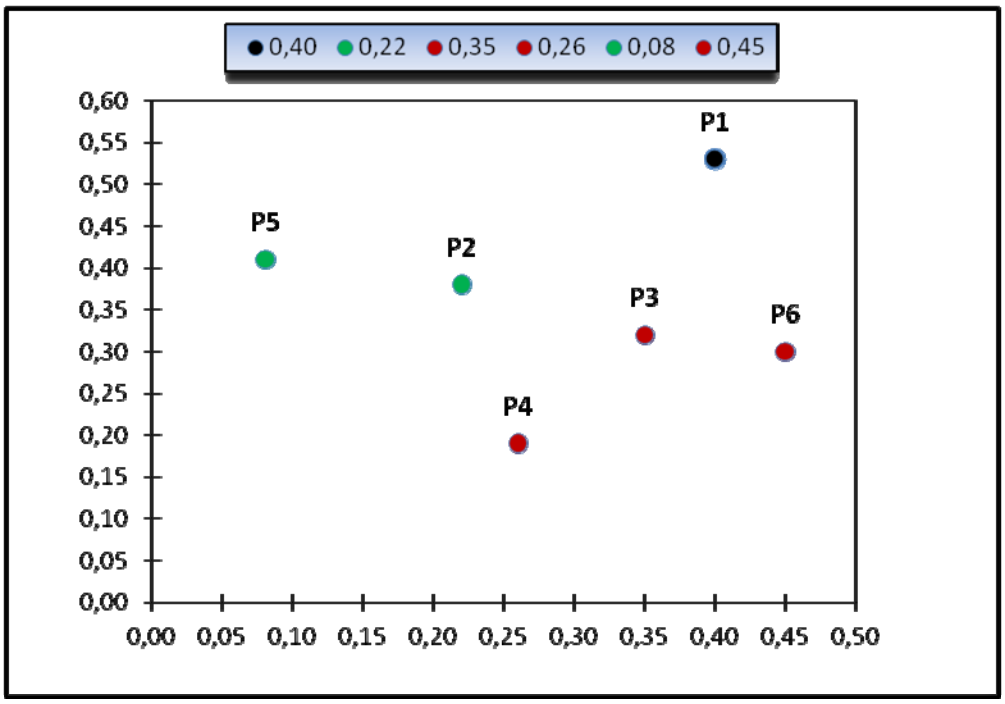
= 0.3860

Πίνακας Αποστάσεως.

	p1	{p2,p5}	{p3,p6}	p4
p1	0	0.3418	0.2354	0.3677
{p2,p5}	0	0	0.3860	0.2843
{p3,p6}	0	0	0	0.2195
p4	0	0	0	0

β. Αφού έχουμε και άλλες διαθέσιμες συστάδες για συγχώνευση, συνεχίζουμε να επαναλαμβάνουμε το βήμα 3.

Βλέποντας τον πιο πρόσφατο πίνακα αποστάσεως παραπάνω διαπιστώνουμε ότι οι συστάδες (p3,p6,) και p4 έχουν τη μικρότερη απόσταση όλων - 0.2195. Επομένως συγχωνεύουμε αυτά τα δύο σε μία συστάδα και υπολογίζουμε ξανά τον πίνακα αποστάσεως.

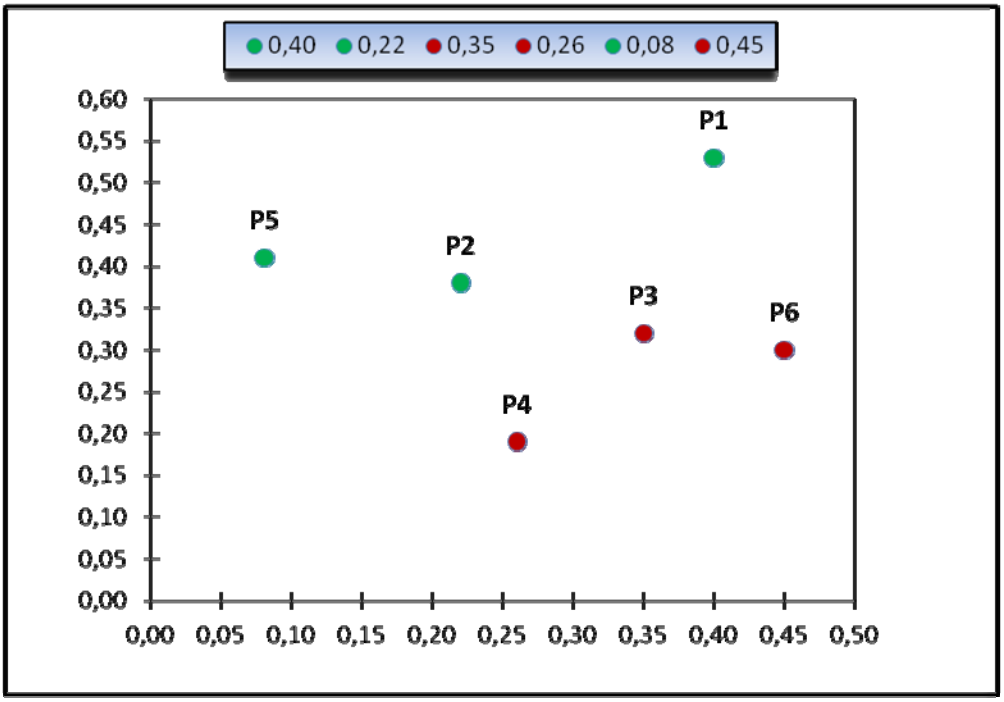


Πίνακας Αποστάσεως.

	p1	{p2,p5}	{p3,p6,p4}
p1	0	0.3418	0.3677
{p2,p5}	0	0	0.3860
{p3,p6,p4}	0	0	0

γ. Αφού έχουμε και άλλες διαθέσιμες συστάδες για συγχώνευση, συνεχίζουμε να επαναλαμβάνουμε το βήμα 3.

Τώρα, βλέποντας τον πιο πρόσφατο πίνακα αποστάσεως παραπάνω, παρατηρούμε πως τα (p2,p5) και p1 έχουν τη μικρότερη απόσταση όλων - 0.3418 . Έτσι συγχωνεύουμε αυτά τα δύο σε μία συστάδα και υπολογίζουμε ξανά τον πίνακα αποστάσεως.

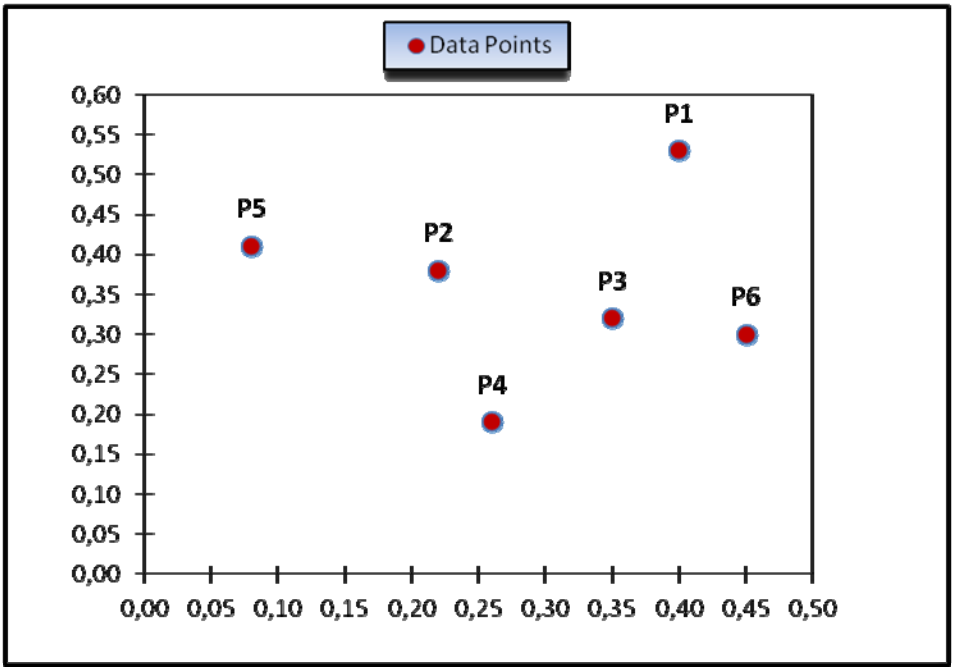


Πίνακας Αποστάσεως.

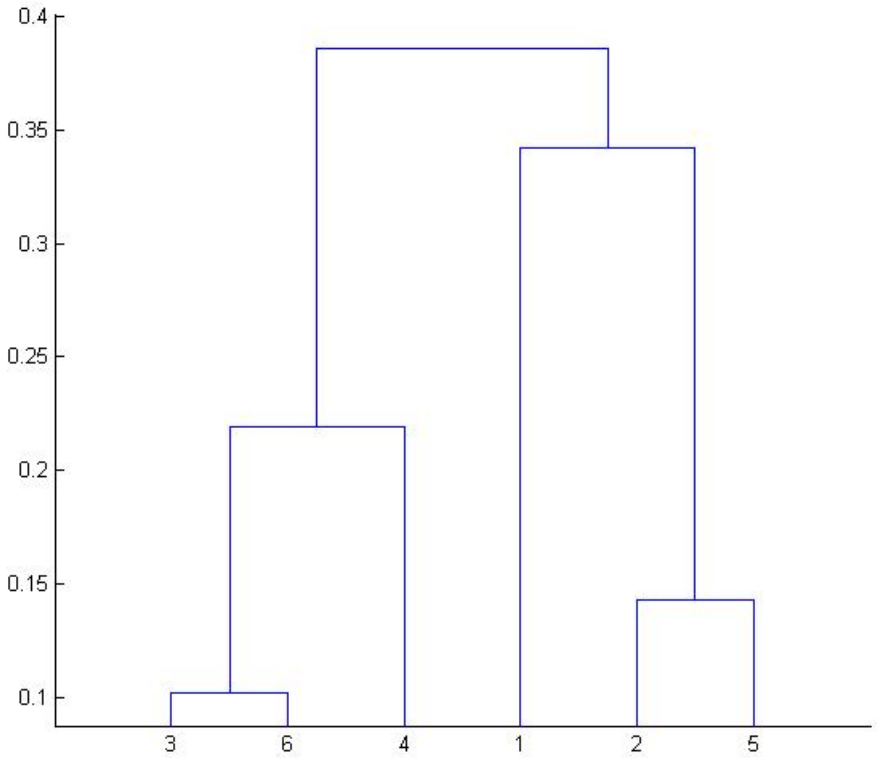
	{p2,p5,p1}	{p3,p6,p4}
{p2,p5,p1}	0	0.3860
{p3,p6,p4}	0	0

δ. Αφού έχουμε και άλλες διαθέσιμες συστάδες για συγχώνευση, συνεχίζουμε να επαναλαμβάνουμε το βήμα 3.

Επομένως, βλέποντας τον πιο πρόσφατο πίνακα αποστάσεως παραπάνω, παρατηρούμε ότι τα (p2, p5, p2, p1) και (p3, p6, p4) έχουν τη μικρότερη απόσταση όλων - 0.3860 (η τελευταία που απομένει στον πίνακα). Έτσι συγχωνεύουμε αυτά τα δύο σε μία συστάδα. Τώρα πλέον δεν υπάρχει λόγος να υπολογίσουμε τον πίνακα αποστάσεως διότι δεν υπάρχουν άλλες συστάδες για συγχώνευση.



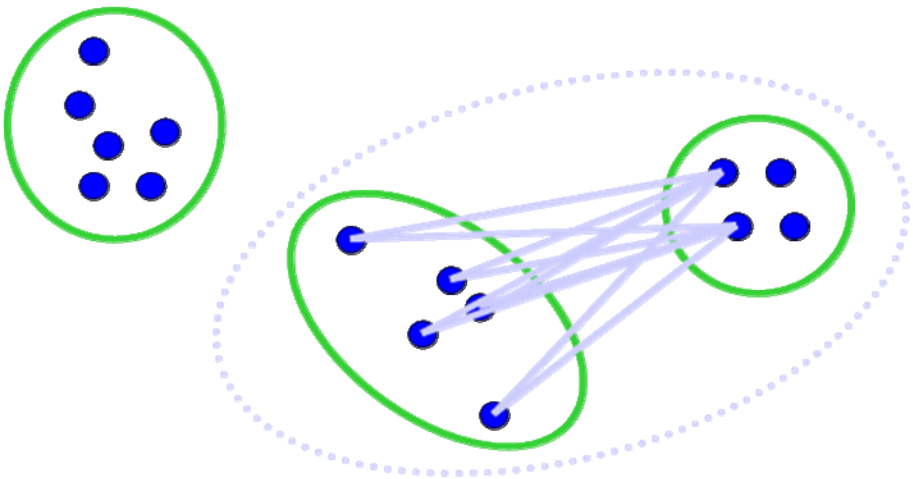
Το δενδρόγραμμα για το παράδειγμα που μελετήσαμε φαίνεται παρακάτω.



Εικόνα 7. Δενδρόγραμμα Πλήρους Δεσμού

4.6 Συσταδοποίηση Μέσου Δεσμού (Average Link Method)

Η συσταδοποίηση μέσου δεσμού είναι ένας συμβιβασμός μεταξύ των άκρων που διέπουν τις μεθόδους απλού δεσμού και πλήρους. Ως κριτήριο για την συγχώνευση χρησιμοποιούμε τη μέση ομοιότητα όλων των σημείων της μίας συστάδας με την άλλη. Κάθε στοιχείο που ανήκει σε μία συστάδα έχει μεγαλύτερη μέση ομοιότητα με τα υπόλοιπα μέλη της συστάδας που ανήκει από όλα τα υπόλοιπα στοιχεία των άλλων συστάδων.



Εικόνα 8. Παράδειγμα Μέσου Δεσμού

Παράδειγμα (Μέθοδος Μέσου Δεσμού)

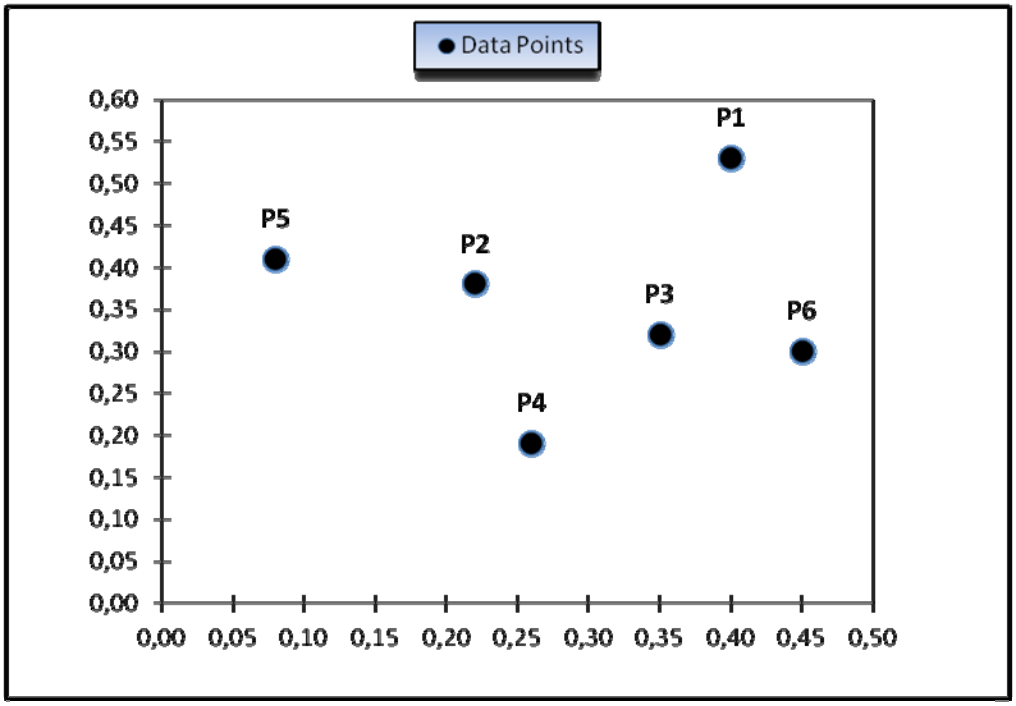
Πρόβλημα: Ας υποθέσουμε ότι η βάση δεδομένων D προς επεξεργασία δίνεται από τον πίνακα παρακάτω. Ακολουθώντας την μέθοδο μέσου δεσμού να βρούμε τις συστάδες στην βάση δεδομένων D χρησιμοποιώντας ως μέτρο την Ευκλείδεια απόσταση.

Τα δεδομένα της βάσης D

	x	**y**
p1	0.40	0.53
p2	0.22	0.38
p3	0.35	0.32
p4	0.26	0.19
p5	0.08	0.41
p6	0.45	0.30

Λύση:

Βήμα 1. Κάνουμε την γραφική παράσταση των σημείων στον ν-διάστατο χώρο (όπου ν είναι ο αριθμός των χαρακτηριστικών), στην περίπτωσή μας είναι 2, το x και y.Επομένως κάνουμε την γραφική παράσταση των σημείων p1, p2, … p6 στον δυσδιάστατο χώρο:



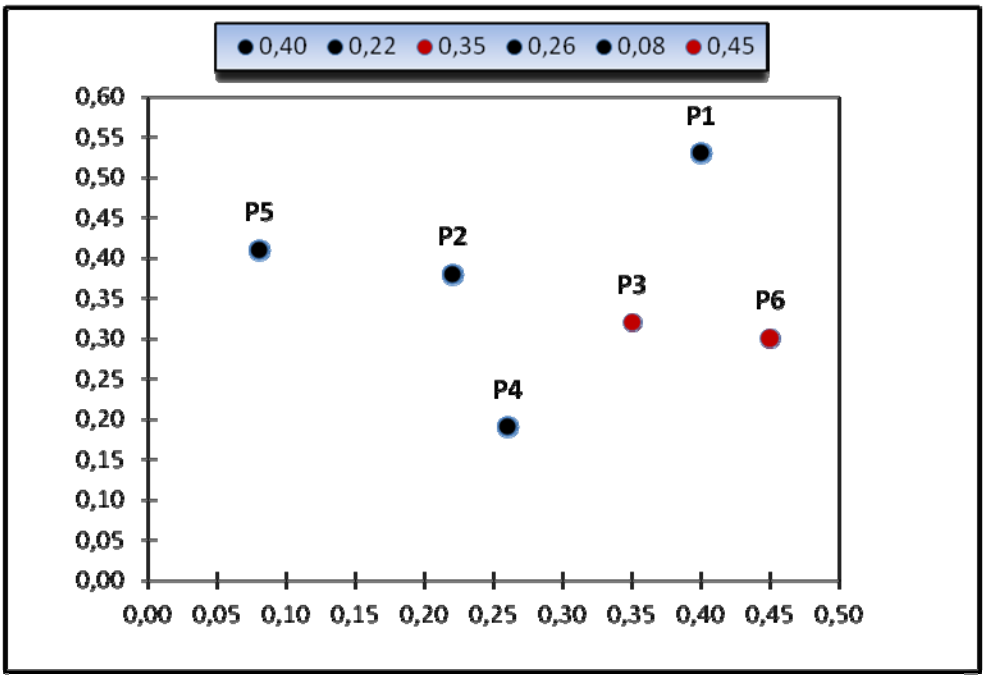
Βήμα 2. Υπολογίζουμε την απόσταση κάθε σημείου από όλα τα άλλα χρησιμοποιώντας τον τύπο της ευκλείδειας απόστασης και τοποθετούμε τα νούμερα στον πίνακα αποστάσεως.

Πίνακας Αποστάσεως

	p1	p2	p3	p4	p5	p6
p1	0	0.2343	0.2159	0.3677	0.3418	0.2354
p2	0	0	0.1432	0.1942	0.1432	0.2435
p3	0	0	0	0.1581	0.2846	0.1020
p4	0	0	0	0	0.2843	0.2195
p5	0	0	0	0	0	0.3860
p6	0	0	0	0	0	0

Βήμα 3 Βρίσκουμε στον πίνακα τις δύο συστάδες με την μικρότερη απόσταση, και τις συγχωνεύουμε σε μία. Υπολογίζουμε ξανά τις τιμές για τον πίνακα αποστάσεως αφού αυτές οι δύο συστάδες είναι πλέον μία (δεν υφίστανται πλέον σαν μονάδες).

Κοιτώντας τον πίνακα αποστάσεως παραπάνω, βλέπουμε ότι τα σημεία p3 και p6 είναι αυτά τα οποία έχουν τη μικρότερη απόσταση μεταξύ όλων των άλλων - 0.1020 ,Επομένως συγχωνεύουμε αυτά τα δύο σε μία συστάδα και υπολογίζουμε ξανά τον πίνακα αποστάσεως.



Πίνακας Αποστάσεως

	p1	p2	{p3,p6}	p4	p5
p1	0	0.2343	0.2256	0.3677	0.3418
p2	0	0	0.1933	0.1942	0.1432
{p3,p6}	0	0	0	0.1888	0.3353
p4	0	0	0	0	0.2843
p5	0	0	0	0	0

Υπολογίζοντας την απόσταση της νέας μας συστάδας - (p3, p6) από όλα τα υπόλοιπα σημεία, ανακαλούμε το γεγονός ότι στην μέθοδο μέσου δεσμού η ομοιότητα μεταξύ δύο συστάδων καθορίζεται από τη μέση ομοιότητα όλων των σημείων της μίας συστάδας με την άλλη. Έτσι για παράδειγμα :

$$D(C, C') = \frac{1}{|C||C'|} \sum_{x \in C, y \in C'} d(x, y)$$

dist((p3, p6), p4) = (*dist*(p3, p4) + *dist*(p6, p4))

= (0.1581 + 0.2195)

= 0.3776

= 0.1888

dist((p3, p6), p1) = (*dist*(p3, p1) + *dist*(p6, p1))

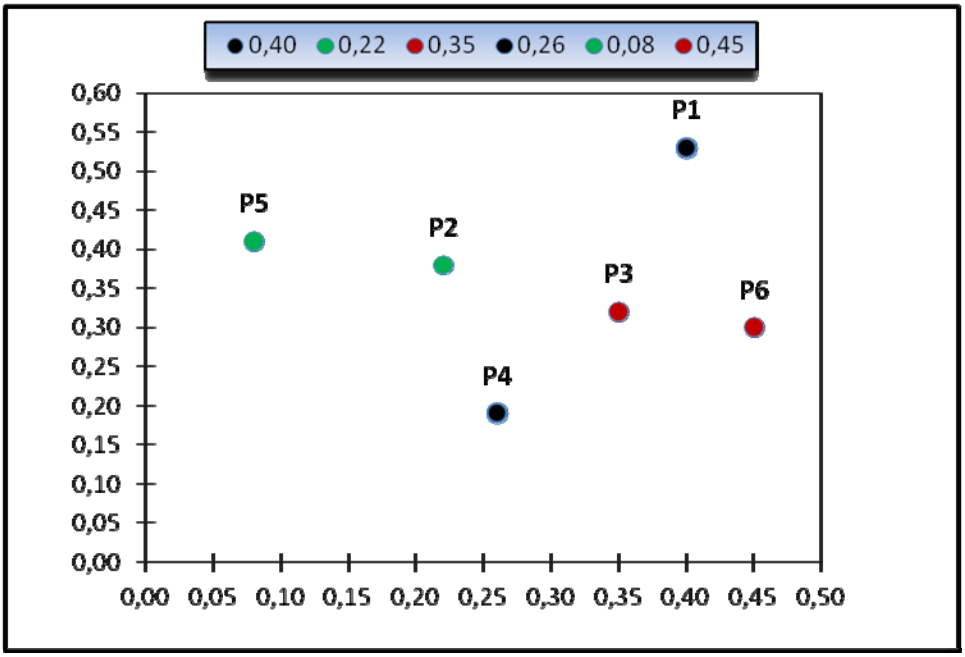
= (0.2159 + 0.2354)

= 0.4513

= 0.2257

Βήμα 4 Επανάληψη του βήματος 3 μέχρι όλες οι συστάδες (ή γενικά όλα τα σημεία) να συγχωνευθούν σε μία συστάδα.

α. Κοιτώντας τον πιο πρόσφατο πίνακα αποστάσεως παραπάνω, βλέπουμε πως τα σημεία p2 και p5 είναι εκείνα με την μικρότερη απόσταση - 0.1432.Τα συγχωνεύουμε σε μία συστάδα και υπολογίζουμε ξανά τον πίνακα αποστάσεως.



Ανακαλούμε το γεγονός ότι στην μέθοδο μέσου δεσμού η ομοιότητα μεταξύ δύο συστάδων καθορίζεται από τη μέση ομοιότητα όλων των σημείων της μίας συστάδας με την άλλη. Έτσι για παράδειγμα :

dist((p3, p6), (p2,p5)) = (*dist*(p3, p2) + *dist*(p3, p5) + *dist*(p6, p2) + *dist*(p6, p5))

= (0.1432 + 0.2846 + 0.2435 + 0.3860)

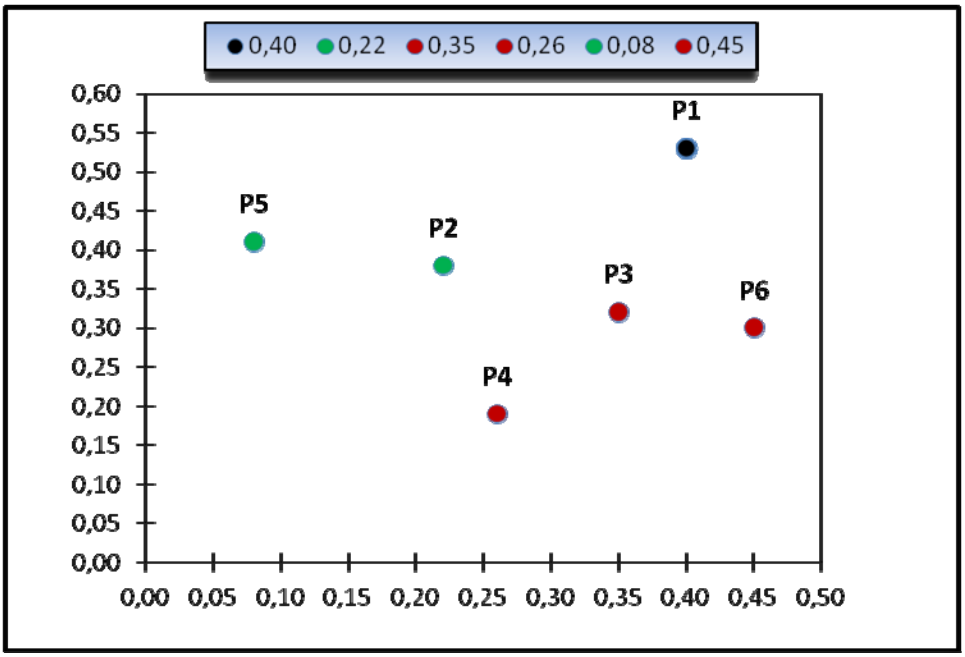
= 1.0573

= 0.2643

Πίνακας Αποστάσεως

	p1	{p2,p5}	{p3,p6}	p4
p1	0	0.2880	0.2256	0.3677
{p2,p5}	0	0	0.2643	0.2392
{p3,p6}	0	0	0	0.1888
p4	0	0	0	0

β. Αφού έχουμε και άλλες διαθέσιμες συστάδες για συγχώνευση, συνεχίζουμε να επαναλαμβάνουμε το βήμα 3.



Βλέποντας τον πιο πρόσφατο πίνακα αποστάσεως παραπάνω διαπιστώνουμε ότι οι συστάδες (p3,p6,) και p4 έχουν τη μικρότερη απόσταση όλων - 0.1888. Επομένως συγχωνεύουμε αυτά τα δύο σε μία συστάδα και υπολογίζουμε ξανά τον πίνακα αποστάσεως.

dist((p3, p6,p4),(p2,p5)) = (*dist*(p3, p2) + *dist*(p3, p5) + *dist*(p6, p2) + *dist*(p6, p5) + *dist*(p4, p2) + *dist*(p4, p5))

= (0.1432 + 0.2846 + 0.2435 + 0.3860 + 0.1942 + 0.2843)

= 1.5358

= 0.2560

dist((p3, p6,p4), p1) = (*dist*(p3, p1) + *dist*(p6, p1) + *dist*(p4, p1))

= (0.2159 + 0.2354 + 0.3677)

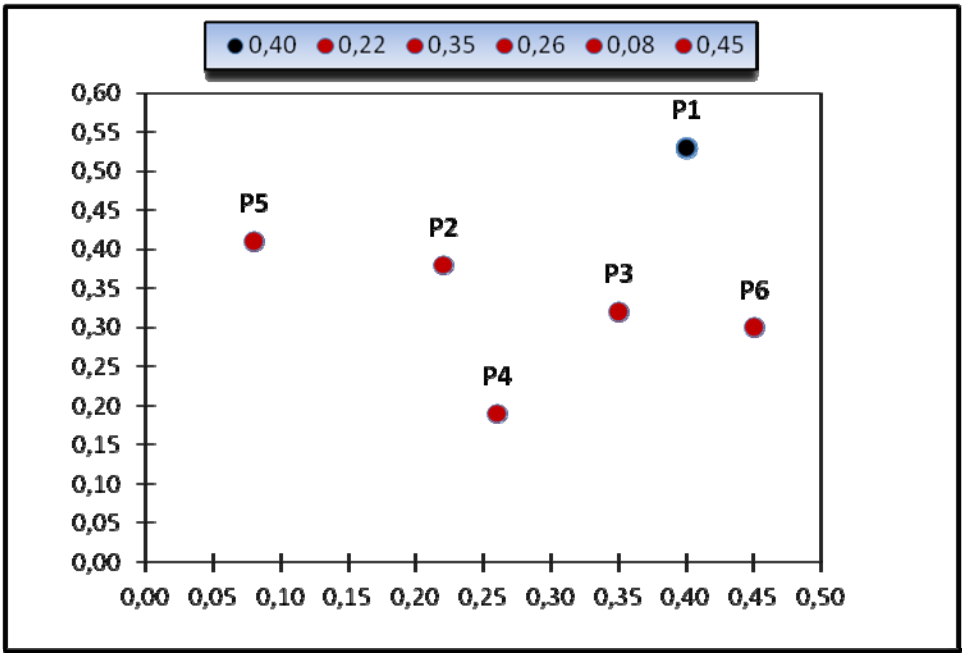
= 0.8190

= 0.2730

Πίνακας Αποστάσεως

	p1	{p2,p5}	{p3,p6,p4}
p1	0	0.2880	0.2730
{p2,p5}	0	0	0.2560
{p3,p6,p4}	0	0	0

γ. Αφού έχουμε και άλλες διαθέσιμες συστάδες για συγχώνευση, συνεχίζουμε να επαναλαμβάνουμε το βήμα 3.



Βλέποντας τον πιο πρόσφατο πίνακα αποστάσεως παραπάνω διαπιστώνουμε ότι οι συστάδες (p3,p6,p4) και (p2,p5) έχουν τη μικρότερη απόσταση όλων - 0.2560. Επομένως συγχωνεύουμε αυτά τα δύο σε μία συστάδα και υπολογίζουμε ξανά τον πίνακα αποστάσεως.

dist((p3,p6,p4,p2,p5), p1) = (*dist*(p3, p1) + *dist*(p6, p1) + *dist*(p4, p1) + *dist*(p2, p1) + *dist*(p5, p1)

= (0.2159 + 0.2354 + 0.3677 + 0.2343 + 0.3418)

= 1.3951

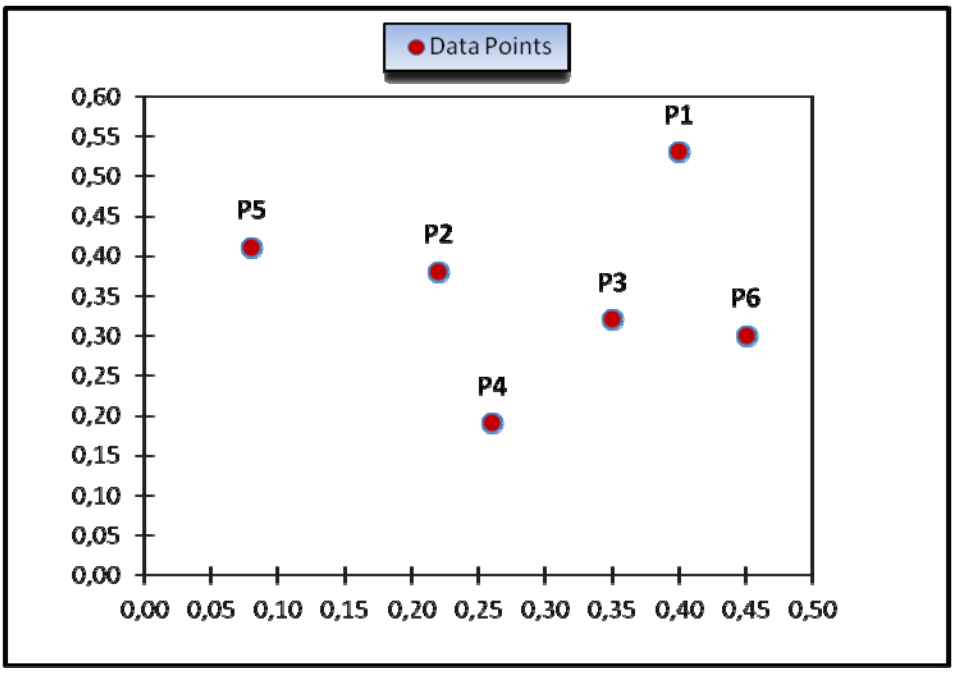
= 0.2790

Πίνακας Αποστάσεως

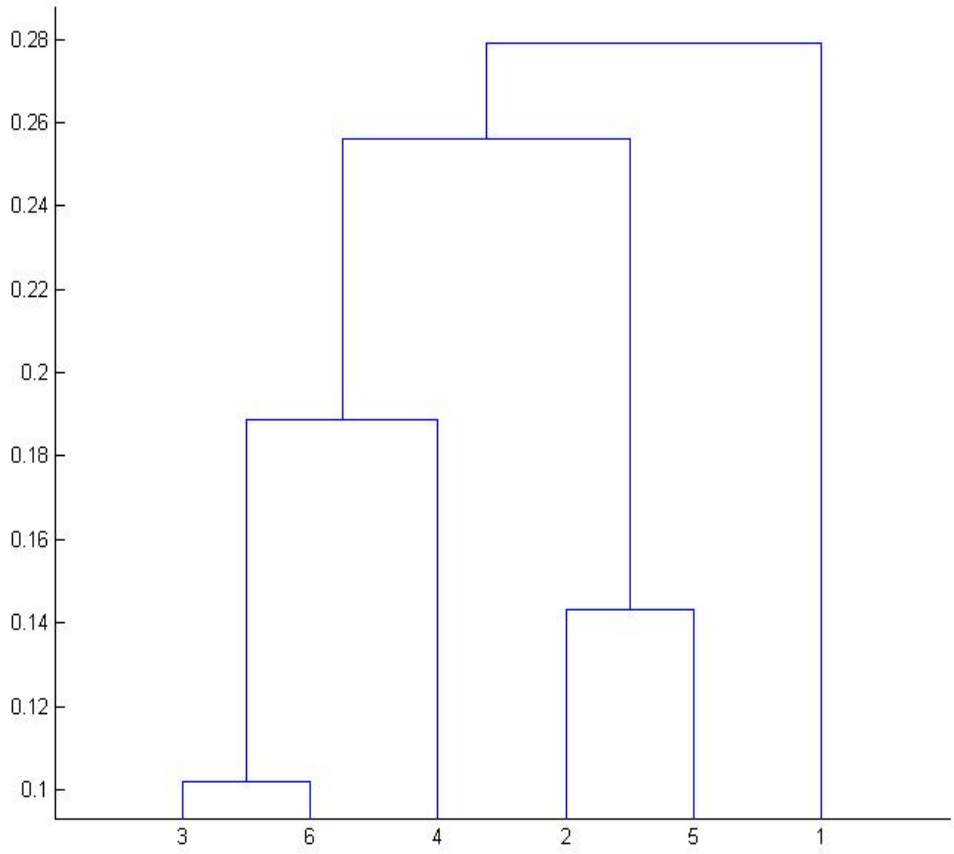
	p1	{p2,p5,p3,p6,p4}
p1	0	0.2790
{p2,p5,p3,p6,p4}	0	0

δ. Αφού έχουμε και άλλες διαθέσιμες συστάδες για συγχώνευση, συνεχίζουμε να επαναλαμβάνουμε το βήμα 3.

Βλέποντας έτσι τον πιο πρόσφατο πίνακα αποστάσεως παραπάνω, παρατηρούμε ότι τα (p2, p5, p3, p6, p4) και p1 έχουν τη μικρότερη απόσταση όλων - 0.2790 (η τελευταία που απομένει στον πίνακα). Έτσι συγχωνεύουμε αυτά τα δύο σε μία συστάδα. Τώρα πλέον δεν υπάρχει λόγος να υπολογίσουμε τον πίνακα αποστάσεως διότι δεν υπάρχουν άλλες συστάδες για συγχώνευση.



Το δενδρόγραμμα για το παράδειγμα που μελετήσαμε φαίνεται παρακάτω.



4.7 Μέθοδος μέσου δεσμού με βάρη (Weighted Average Link Clustering Method)

Αυτή η μέθοδος είναι ίδια με την μέθοδο μέσου δεσμού με την διαφορά ότι στις διάφορες πράξεις το μέγεθος των συστάδων χρησιμοποιείται ως βάρος. Αυτή η μέθοδος κρίνεται πιθανόν κατάλληλη όταν έχουμε την υποψία ότι τα μεγέθη των συστάδων είναι κατά πολύ ανόμοια.

4.8 Η Μέθοδος του Ward (Ward Clustering Method)

Σε κάθε βήμα στην μέθοδο του Ward, παίρνουμε υπόψιν την ένωση κάθε ζεύγους συστάδων και συγχωνεύουμε τις δύο συστάδες που έχουν τη μικρότερη αύξηση σε "απώλεια πληροφορίας" (Information loss). Ως απώλεια πληροφορίας στην μέθοδο του Ward ορίζεται το άθροισμα των τετράγωνων του λάθους (Error Sum of Squares – ESS)

$$\sum_{j=1}^{n} (x_{ij} - \mu_j)^2$$

$$\mu_j = \frac{1}{m}\sum_{i=1}^{m} x_{ij}$$

$$ESS = \sum_{i=1}^{m}\sum_{j=1}^{n} (x_{ij} - \mu_j)^2$$

Γενικά αυτή η μέθοδος θεωρείται ως πολύ αποτελεσματική όμως τείνει να δημιουργεί συστάδες μικρού μεγέθους. Το προτεινόμενο μέτρο απόστασης για την μέθοδο αυτή φαίνεται πως είναι η τετραγωνισμένη ευκλείδεια απόσταση.

4.9 Η Μέθοδος Centroid (Centroid Clustering Method)

Στην μέθοδο centroid η ομοιότητα μεταξύ δύο συστάδων καθορίζεται από την απόσταση μεταξύ των κέντρων τους (centroids).

Για αυτή την μέθοδο προτείνεται η χρήση της τετραγωνισμένης ευκλείδειας απόστασης.

4.10 Η Μέθοδος Median (Median Clustering Method)

Η μέθοδος median αναφέρεται και ως “weighted centroid” . Προτάθηκε πρώτα από τον Gower (1967) για να εξαλείψει κάποιες αδυναμίες της μεθόδου centroid. Στην μέθοδο centroid, εάν τα μεγέθη των συστάδων που συγχωνεύτηκαν είναι αρκετά διαφορετικά, τότε το νέο centroid (κέντρο βάρους) θα είναι κοντά σε αυτό της μεγαλύτερης συστάδας και μπορεί να παραμείνει μέσα σε αυτήν. Στην median το centroid (κέντρο βάρους) της νέας συστάδας είναι ανεξάρτητο από τα μεγέθη των δύο συστάδων.

Για αυτή την μέθοδο προτείνεται η χρήση της τετραγωνισμένης ευκλείδειας απόστασης.

4.11 Παρατηρήσεις Περί Ιεραρχικής Συσταδοποίησης

Βλέπουμε λοιπόν από τα παραδείγματα πως γενικά οι συσσωρευτικοί ιεραρχικοί αλγόριθμοι συσταδοποίησης ακολουθούν το εξής πρότυπο :

1. Ο αλγόριθμος πρέπει να παίρνει ως είσοδο ένα τετραγωνικό nxn πίνακα d αποστάσεως ο οποίος να περιέχει τις ανά ζεύγη αποστάσεις μεταξύ των σημείων.

2. Αναθέτουμε κάθε σημείο σε ξεχωριστή συστάδα.

3. Όσο υπάρχουν παραπάνω από μία συστάδες, βρες τις δύο πιο όμοιες C_1 και C_2 και συγχώνευσε αυτές σε μία νέα συστάδα C. (Εδώ το κριτήριο ομοιότητας μπορεί να είναι η μικρότερη, η μεγαλύτερη ή μέση απόσταση οπότε και έχουμε τις απλού, πλήρους και μέσου δεσμού μεθόδους συσταδοποίησης.)

4. Υπολόγισε την απόσταση της συστάδας C από όλες τις υπόλοιπες

5. Αφαίρεσε τις στήλες και γραμμές του πίνακα αποστάσεως που αντιστοιχούν στις συστάδες C_1 και C_2

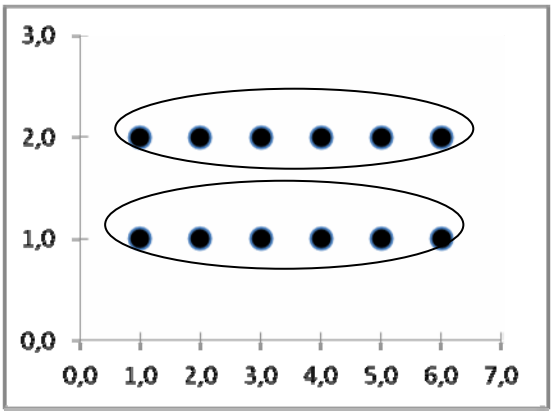
6. Προσθήκη γραμμής και στήλης στον πίνακα αποστάσεως d που να αντιστοιχεί στη νέα συστάδα C.

Συνθήκη Παύσης Αλγορίθμου: Όταν εξηγήσαμε προηγουμένως τη μέθοδο του απλού δεσμού, είπαμε πως κάθε στοιχείο τίθεται σε ξεχωριστή συστάδα και σε κάθε βήμα συγχωνεύουμε τα δύο κοντινότερα ζευγάρια. Αλλά σε περίπτωση που δεν θέσουμε κάποιο κριτήριο παύσης του αλγορίθμου, αυτός θα εκτελείται έως ότου όλα τα στοιχεία στο τέλος αποτελούν μία συστάδα. Αυτός πιθανόν όμως δεν είναι ο σκοπός για τον οποίο εκτελέσαμε τον αλγόριθμο. Επομένως θα πρέπει ο αλγόριθμος να παύει σε κάποιο σημείο. Ένα κριτήριο θα ήταν να καθορίσει ο χρήστης τον αριθμό των συστάδων που θέλει να έχει, ή να ορίσουμε εμείς κάποιο κριτήριο στον αλγόριθμο ώστε να πάρει κάποια απόφαση μόνος του, διότι όπως βλέπουμε σε κάθε βήμα η απόσταση αυξάνει και πιθανόν μετά από κάποιες επαναλήψεις να βρεθούν στη συστάδα στοιχεία πολύ απομακρυσμένα με πολύ μικρό βαθμό σχετικότητας. Έτσι, μπορεί να τεθεί ως κριτήριο μία απόσταση έτσι ώστε αν κάποια στοιχεία που βρίσκονται απομακρυσμένα και λογικά να μην πρέπει να συνχωνευθούν να παραμείνουν χωριστά και όχι μέλη της ίδιας συστάδας. Επίσης ως κριτήριο παύσης μπορεί να τεθεί και ένας προκαθορισμένος αριθμός συστάδων που θέλουμε να δημιουργήσουμε εξαρχής. Καλό όμως είναι να αφήσουμε πάρα ταύτα τον αλγόριθμο να εκτελεστεί πλήρως.

- ***Απλού και Πλήρους Δεσμού*:** Συσταδοποίηση που στηρίζεται στην ομοιότητα ένα μόνο ζεύγους δεν μπορεί να αντικατοπτρίσει πλήρως τη διανομή των στοιχείων σε μία συστάδα. Επομένως και οι δύο αυτές μέθοδοι μπορεί να παράγουν ανεπιθύμητες συστάδες.

☐ **Chaining: Ένα πρόβλημα της μεθόδου απλού δεσμού.**

Μία αλυσίδα σημείων μπορεί να επεκτείνεται για μεγάλες αποστάσεις χωρίς να λαμβάνεται υπόψιν το συνολικό σχήμα της συστάδας.



2. Βάσεις δεδομένων,ODBC και Matlab (σύνταξη βασικών συναρτήσεων)

2.1 Matlab και βασικές συναρτήσεις συσταδοποίησης

Το όνομα Matlab προέρχεται από τις λέξεις «Matrix laboratory» . Επινοήθηκε στα τέλη του 1970 από τον Cleve Moler ο οποίος ήθελε να δώσει στους μαθητές του πρόσβαση στα πακέτα LINPACK και EISPACK χωρίς να χρειάζεται η εκμάθηση της Fortran. Το Matlab κέρδισε δημοτικότητα κυρίως από στόμα σε στόμα επειδή δεν διατίθονταν επισήμως. Στην δεκαετία του 1980, το Matlab γράφτηκε σε γλώσσα C με περισσότερη λειτουργικότητα όπως ρουτίνες για παραγωγή γραφικών παραστάσεων. Η εταιρεία Mathworks, Inc. που ιδρύθηκε το 1984 είναι στο παρόν υπεύθυνη για την ανάπτυξη και υποστήριξη του Matlab.

Τα πλεονεκτήματα του Matlab :

- Το MATLAB είναι σχετικά εύκολο στην εκμάθηση

- Ο κώδικας στο MATLAB είναι βελτιστοποιημένος ώστε να είναι σχετικά γρήγορος στον υπολογισμό πινάκων.
- Το MATLAB μπορεί να συμπεριφερθεί ως γλώσσα προγραμματισμού ή ως μία αριθμομηχανή
- Το MATLAB είναι μία interpreted γλώσσα προγραμματισμού, δηλαδή κάθε εντολή μεταφράζετε την προκειμένη στιγμή σε γλώσσα μηχανής και έτσι τα λάθη είναι ευκολότερο να διορθωθούν.

Αδυναμίες του Matlab :

- Το MATLAB δεν είναι γλώσσα προγραμματισμού γενικής χρήσης.
- Το MATLAB είναι μία interpreted γλώσσα προγραμματισμού (καθιστώντας το για παράδειγμα πιο αργό από μία γλώσσα προγραμματισμού όπως η C++ που είναι compiled)
- Το MATLAB είναι σχεδιασμένο για επιστημονικούς υπολογισμούς και δεν είναι κατάλληλο για κάποια άλλα πράγματα.

Για την περίπτωσή μας, η συσταδοποίηση απαιτεί πληθώρα υπολογισμών και το Matlab διαθέτει πολλές συναρτήσεις για διάφορους υπολογισμούς. Αυτό την καθιστά ως μία καλή επιλογή για την περίπτωση που τη χρησιμοποιούμε.

Ας δούμε τις βασικές συναρτήσεις του Matlab που χρησιμοποιήθηκαν στην εφαρμογή που υλοποιήθηκε για την εκτέλεση της συσταδοποίησης.

- **y = pdist(X) :** Η συνάρτηση υπολογίζει την ανά ζεύγη ευκλείδεια απόσταση μεταξύ των σημείων και άλλες μορφές της συνάρτησης περιλαμβάνουν y = pdist(X,*metric*) και y = pdist(X,'minkowski',p). Στην πρώτη μορφή μπορούμε να δώσουμε στη θέση της μεταβλητής metric ένα από τα διαθέσιμα μέτρα απόστασης, μεταξύ αυτών οι αποστάσεις City Block ή αλλιώς και απόσταση Manhattan, η απόσταση Chebyshev αλλά και η ευκλείδεια απόσταση. Η δεύτερη

μορφή αποτελεί μία ξεχωριστή περίπτωση εάν θέλουμε να κάνουμε χρήση της απόστασης Minkowski. Στη θέση της μεταβλητής p θέτουμε την τιμή του εκθέτη για τον τύπο της απόστασης Minkowski.

Δείγμα εξόδου της συνάρτησης

Columns 1 through 9

0.2343 0.2159 0.3677 0.3418 0.2354 0.1432 0.1942 0.1432 0.2435

Columns 10 through 15

0.1581 0.2846 0.1020 0.2843 0.2195 0.3860

- **Z = linkage(y) :** Η συνάρτηση υπολογίζει τους δεσμούς μεταξύ των διαφόρων σημείων παίρνοντας ως είσοδο τον πίνακα αποστάσεων που παράγει η συνάρτηση pdist. Άλλη μορφή της συνάρτησης αποτελεί η Z = linkage(y,*method*), όπου στη θέση της μεταβλητής method δηλώνουμε τη μέθοδο που θέλουμε να κάνει χρήση η συνάρτηση για τον υπολογισμό των δεσμών και αυτές είναι οι μέθοδοι απλού δεσμού (single link), πλήρους δεσμού (complete link), μέσου δεσμού (average link), Centroid, Median μέθοδος μέσου δεσμού με βάρη (weighted average link) και η μέθοδος Ward.

  Δείγμα εξόδου της συνάρτησης

  3.0000 6.0000 0.1020

  2.0000 7.0000 0.1432

  5.0000 8.0000 0.1432

  4.0000 9.0000 0.1581

  1.0000 10.0000 0.2159

  Η συνάρτηση linkage αριθμεί τα στοιχεία αρχικά με ένα αύξοντα αριθμό ξεκινώντας από το ένα και συνεχίζει για όσα στοιχεία

υπάρχουν, αναθέτοντας έτσι κάθε στοιχείο αρχικά στη δική του συστάδα. Έτσι για παράδειγμα εάν έχουμε έξι στοιχεία η συνάρτηση θα αριθμήσει το καθένα κατάλληλα. Έπειτα, κάθε φορά που δημιουργείται ένας δεσμός, η συνάρτηση ξεκινά να δίνει ένα αύξοντα αριθμό μεγαλύτερο από τον αριθμό των στοιχείων για να αναγνωρίσει τη νέα συστάδα. Έτσι για παράδειγμα, αν δούμε τα δεδομένα του δείγματος παραπάνω, στην πρώτη επανάληψη έχουμε τα στοιχεία 3 και 6 με απόσταση 0.1020 , στη νέα συστάδα που δημιουργούν αυτά τα στοιχεία η συνάρτηση linkage θα δώσει τον αριθμό 7, επομένως στις επόμενες επαναλήψεις εάν δούμε τον αριθμό 7 ξέρουμε ότι αναφέρεται στη συστάδα που περιέχει τα στοιχεία 3 και 6. Στη δεύτερη επανάληψη έχουμε τη συστάδα 2 και τη συστάδα 7, η συστάδα 2 περιέχει μόνο το στοιχείο δύο όμως η συστάδα 7 περιέχει όπως είδαμε πριν τα στοιχεία 3 και 6. Η συνάρτηση εκτελείται μέχρι όλα τα στοιχεία να βρεθούν σε μία συστάδα, δημιουργεί δηλαδή όλους τους πιθανούς δεσμούς μεταξύ των στοιχείων.

- **T = cluster(Z,'cutoff',c) :** Η συνάρτηση κατασκευάζει συστάδες βάση του ιεραρχικού δέντρου που κατασκευάζει η συνάρτηση linkage. Άλλη μορφή της συνάρτησης αποτελεί η T = cluster(Z,'maxclust',n) . Στην περίπτωση αυτή η συνάρτηση cluster δημιουργεί συστάδες με κριτήριο το μέγιστο επιθυμητό αριθμό και το πετυχαίνει κόβοντας το δενδρόγραμμα στο ύψος που έχει δημιουργηθεί ο κατάλληλος αριθμός συστάδων. Μία άλλη μορφή της συνάρτησης αποτελεί η T = cluster(Z,'cutoff',c,'criterion',criterion), όπου στην περίπτωση αυτή δηλώνουμε ένα κριτήριο με το οποίο θέλουμε να δημιουργηθούν οι συστάδες, διαφορετικό από το μέγιστο αριθμό συστάδων και το

οποίο μπορεί να είναι το κριτήριο της απόστασης. Όσο η συνάρτηση linkage δημιουργεί δεσμούς μεταξύ των στοιχείων τόσο μεγαλώνει η απόσταση, έτσι μπορεί να βρεθούν στην ίδια συστάδα στοιχεία με πολύ μεγάλη διαφορά απόστασης, με τον τρόπο αυτό ελέγχουμε μέχρι τι μέτρο ομοιότητας θέλουμε να δημιουργηθούν συστάδες αφού ορίζοντας κάποια απόσταση η συνάρτηση cluster κόβει το δενδρόγραμμα στο προκαθορισμένο ύψος που θέσαμε.

Δείγμα εξόδου της συνάρτησης

3

2

2

1

2

2

Βλέπουμε παραπάνω από το δείγμα εξόδου πως η συνάρτηση cluster ως έξοδο δίνει μία σειρά αριθμών, ας πάρουμε για παράδειγμα το πρώτο νούμερο που είναι το 3 και βρίσκεται στην πρώτη θέση. Αυτό σημαίνει ότι το στοιχείο ένα ανήκει στη συστάδα 3, αντίστοιχα το στοιχείο δύο ανήκει στη συστάδα 2, το στοιχείο τρία στη συστάδα 2 και πάει λέγοντας μέχρι να εξαντληθεί όλος ο αριθμός των στοιχείων.

- **T = clusterdata(X,param1,val1,param2,val2,...)**

  Η συνάρτηση αυτή παρέχει την ίδια λειτουργικότητα με τις συναρτήσεις pdist, linkage και cluster. Ανάλογα με τις τιμές που θα θέσουμε στις διάφορες παραμέτρους που δέχεται η συνάρτηση μπορούμε να παραμετροποιήσουμε τον τρόπο υπολογισμού του

μέτρου απόστασης (Ευκλείδεια, Minkowski κτλ), τον τρόπο με τον όποιο θα πραγματοποιηθούν οι δεσμοί (single link, complete link, average link κτλ) και τέλος τον τρόπο δημιουργίας των συστάδων.

Δείγμα εξόδου συνάρτησης

3

2

2

1

2

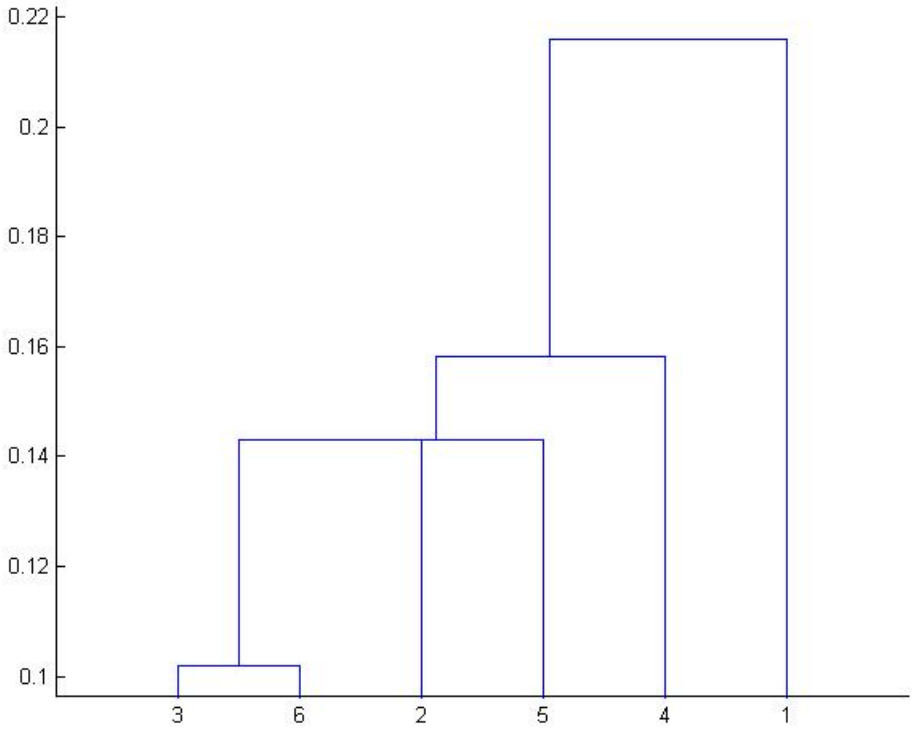
2

Όπως βλέπουμε παραπάνω στο δείγμα εξόδου της συνάρτησης, έχουμε ως έξοδο μία σειρά αριθμών που καθένας δείχνει αντίστοιχα σε ποια συστάδα ανήκει κάθε στοιχείο. Έτσι το πρώτο στοιχείο ανήκει στην συστάδα 3, το δεύτερο στοιχείο στη συστάδα 2, το τρίτο στοιχείο στη συστάδα 2 και πάει λέγοντας. Η μορφή που δίνει έξοδο η συνάρτηση είναι ακριβώς ίδια με αυτή της συνάρτησης cluster.

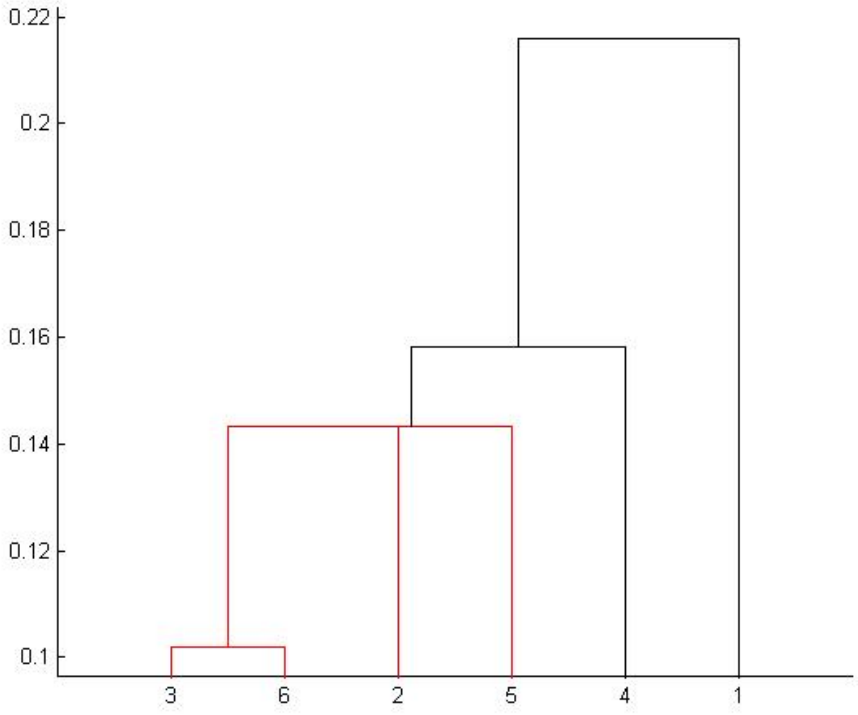
- **H = dendrogram(Z)**

Η συνάρτηση κατασκευάζει το δενδρόγραμμα από τον πίνακα που δίνει ως έξοδο η συνάρτηση linkage. Από τις παραμέτρους που μπορεί να δεχτεί η συνάρτηση, ξεχωρίζουμε την colorthreshold, δίνοντας κάποια τιμή λοιπόν η συνάρτηση dendrogram δίνει ξεχωριστό χρώμα σε κάθε ομάδα δεσμών ή αλλιώς και συστάδας που βρίσκεται κάτω από αυτή την απόσταση.

Δείγμα εξόδου συνάρτησης



Βλέποντας το δενδρόγραμμα μπορούμε να κάνουμε τις εξής παρατηρήσεις. Αρχικά έχουμε συνδέονται τα στοιχεία 3 και 6 που έχουν μεταξύ τους απόσταση περίπου λίγο παραπάνω από 0.1, κοιτώντας τον άξονα y, μετά τα στοιχεία 3 και 6 συνδέονται με το στοιχείο 2 και έπειτα όλα αυτά με το στοιχείο 5, με απόσταση και για τις δύο περιπτώσεις γύρω στο 0.14. Έπειτα συνδέεται με αυτά το στοιχείο 4 με απόσταση περίπου λίγο πιο κάτω από 0.16 και τέλος το στοιχείο 1 με απόσταση 0.22. Βλέπουμε εδώ ότι όσο προχωράμε ψηλότερα στο δενδρόγραμμα τόσο οι αποστάσεις μεταξύ των σημείων και άρα η σχετικότητα μεταξύ τους γίνεται μικρότερη. Ειδικά για το τελευταίο στοιχείο, ένα η αύξηση φαίνεται να είναι ραγδαία, κάτι που σημαίνει ότι μάλλον το σημείο ένα δεν έχει και μεγάλη σχέση με κάποια στοιχεία, ειδικά αυτά που βρίσκονται στα χαμηλότερα στρώματα του δενδρογράμματος. Για να πάρουμε βοήθεια λοιπόν για το πώς θα δημιουργήσουμε τις συστάδες, μπορούμε να πάρουμε βοήθεια βλέποντας την αύξηση της απόστασης η τον αριθμό των δεσμών που δημιουργούνται.



Στο παραπάνω σχήμα βλέπουμε τη γίνεται σε περίπτωση που κάνουμε χρήση της παραμέτρου colorthreshold. Στη συγκεκριμένη περίπτωση έχουμε θέση την τιμή 0.15 και έτσι οι δεσμοί κάτω από αυτή την απόσταση έχουν διαφορετικό χρώμα.

2.2 Η έννοια του ODBC

Ας εξηγήσουμε το πρότυπο ODBC. Τα αρχικά προέρχονται από τις λέξεις Open DataBase Connectivity. Είναι ένα πρότυπο πρόσβασης βάσης δεδομένων που αναπτύχθηκε από την ομάδα SQL Access το 1992. Η ομάδα SQL Access ήταν μία ομάδα εταρειών λογισμικού που ιδρύθηκε το 1989 για τον καθορισμό και την προώθηση προτύπων βάσης δεδομένων για τη φορητότητα και τη διαλειτουργικότητα. Ο στόχος του προτύπου ODBC είναι να καθιστά δυνατό να μπορούν να προσπελαστούν οποιαδήποτε δεδομένα από οποιαδήποτε εφαρμογή ανεξάρτητα με τη σύστημα διαχείρισης βάσης δεδομένων διαχειρίζεται τα δεδομένα (DBMS) . Το ODBC καταφέρνει κάτι τέτοιο εισάγοντας ένα ενδιάμεσο επίπεδο που καλείται database driver,μεταξύ της εφαρμογής και του DBMS. Ο σκοπός

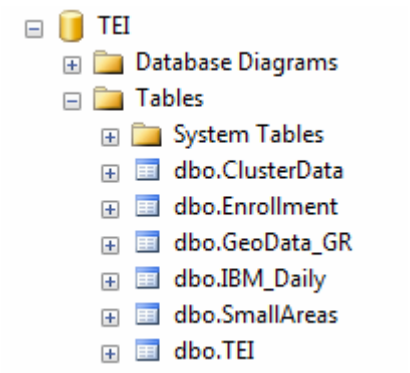
αυτό του επιπέδου είναι να μεταφράζει τα δεδομένα σε εντολές που ο DBMS να καταλαβαίνει. Για να λειτουργήσει κάτι τέτοιο πρέπει και η εφαρμογή και ο DBMS να είναι "συμβατά" με το ODBC, που σημαίνει ότι η εφαρμογή πρέπει να είναι ικανή να δίνει εντολές ODBC και ο DBMS να μπορεί να ανταποκριθεί σε αυτές.

2.3 Βάσεις Δεδομένων

Ένας από τους όρους που οι περισσότεροι άνθρωποι έχουν συνηθίσει να ακούνε είτε στο χώρο εργασίας τους ή κατά την περιήγηση στο διαδίκτυο είναι αυτός της βάσης δεδομένων. Βάση δεδομένων είναι μία δομημένη συλλογή εγγραφών που είναι αποθηκευμένη σε ένα υπολογιστή. Οι βάσεις δεδομένων όμως δεν υφίστανται απόλυτα μέσα στα όρια του υπολογιστή. Παραδείγματα τέτοιων βάσεων δεδομένων αποτελούν ένας τηλεφωνικός κατάλογος ή ένα λεξικό. Οι βάσεις δεδομένων σε υπολογιστές είναι συνήθως οργανωμένες σε έναν ή περισσότερους πίνακες. Ένας πίνακας αποτελείται από ένα αριθμό στηλών και γραμμών. Οι βάσεις που μπορούν να αποθηκευτούν σε ένα υπολογιστή, μπορούν και να διαχειριστούν με κάποιο πρόγραμμα. Αυτά τα προγράμματα καλούνται σύστημα διαχείρισης βάσης δεδομένων (DBMS - database management system). Ως το εργαλείο που χρησιμοποιείται κυρίως για τη διαχείριση των βάσεων δεδομένων, υπάρχουν πολλά συστήματα διαχείρισης βάσης δεδομένων διαθέσιμα στην αγορά. Μερικά δημοφιλή παραδείγματα αποτελούν τα Microsoft Access, FileMaker, DB2, και Oracle. Όλα αυτά τα προϊόντα παρέχουν τη δυνατότητα δημιουργίας μίας σειράς δικαιωμάτων που μπορούν να αποδοθούν σε ένα συγκεκριμένο χρήστη. Είναι δυνατό να οριστεί ένας ή περισσότεροι διαχειριστές που μπορούν να παρέχουν στους άλλους χρήστες διάφορα δικαιώματα διαχείρισης. Αυτή η ευελιξία καθιστά το έργο της επίβλεψης ενός συστήματος με χρήση ενός συστήματος διαχείρισης βάσης δεδομένων κάτι που μπορεί να ελέγχεται κεντρικά ή να είναι διαμοιρασμένο σε διάφορους χρήστες. Τα δεδομένα που είναι καταχωρημένα στη βάση μπορούμε να τα επεξεργαστούμε με μία γλώσσα που ονομάζεται SQL. Τα αρχικά SQL σημαίνουν Structured Query Language και τα οποία στην ελληνική ορολογία μεταφράζονται ως δομημένη γλώσσα αναζήτησης. Συνοπτικά με την SQL μπορούμε να έχουμε πρόσβαση σε μια βάση δεδομένων, να εκτελούμε αναζητήσεις με

κριτήρια, να εισάγουμε νέα δεδομένα, να διαγράψουμε υπάρχοντα δεδομένα και να κάνουμε αλλαγές σε υπάρχοντα δεδομένα. Η SQL άρχισε ως δημιουργία της IBM από τους Andrew Richardson, Donald C. Messerly και Raymond F. Boyce στις αρχές του 1970. Αρχικά ήταν γνωστή ως SEQUEL και αργότερα τυποποιήθηκε από το αμερικανικό εθνικό ίδρυμα προτύπων (ANSI) και το διεθνή οργανισμό για την τυποποίηση (ISO). Υπάρχει μεγάλος αριθμός διαφορετικών εκδόσεων της γλώσσας SQL αλλά κάποιες βασικές εντολές συναντώνται σε όλες τις βάσεις δεδομένων.

Στα πλαίσια της συγκεκριμένης πτυχιακής εργασίας έγινε χρήση του συστήματος διαχείρισης βάσης δεδομένων Microsoft SQL Server 2008. Τα δεδομένα που χρησιμοποιήθηκαν, τοποθετήθηκαν σε μία βάση με όνομα TEI και χωρίστηκαν σε μία σειρά πινάκων όπως φαίνεται στην παρακάτω εικόνα.

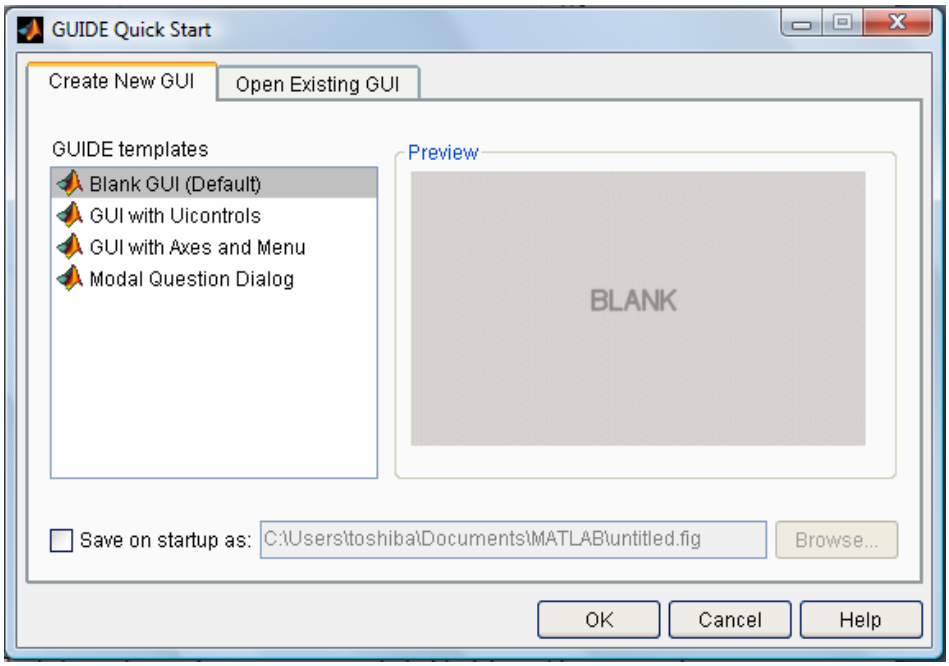


Ο πίνακας ClusterData χρησιμοποιήθηκε για την αποθήκευση δεδομένων μετά τη συσταδοποίηση. Οι πίνακες Enrollment και SmallAreas περιέχουν στατιστικά δεδομένα, ο πίνακας GeoData_GR γεωγραφικά δεδομένα και ο πίνακας TEI εκπαιδευτικά δεδομένα. Για την επικοινωνία του Microsoft SQL Server 2008 με το Matlab 7.5.0.342 (R2007b) έγινε χρήση του προτύπου ODBC.

3. Εφαρμογή Συσταδοποίησης σε Περιβάλλον Matlab

Βήμα 1

Αφού εκτελέσουμε το Matlab και ανοίξει το παράθυρο της εφαρμογής, εκτελούμε την εντολή guide. Αυτή η εντολή θα μας δώσει το ακόλουθο παράθυρο.



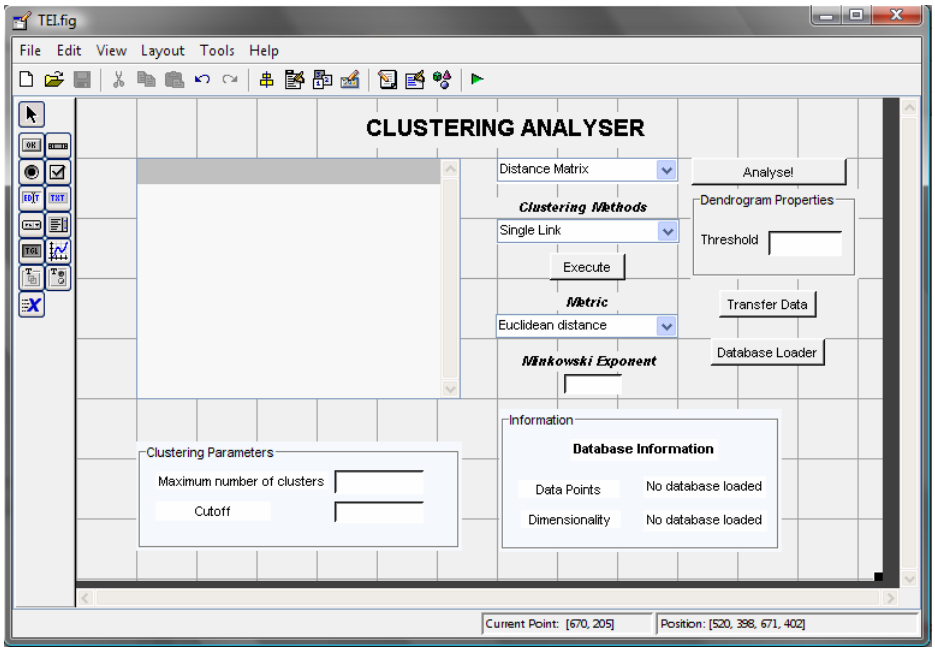
Εικόνα 9. Επιλογές με την εκτέλεση της guide

Θέλουμε να ανοίξουμε μία υπάρχουσα εφαρμογή και έτσι πάμε στην καρτέλα Open Existing GUI και εκεί επιλέγουμε από τη λίστα την εφαρμογή που θέλουμε να εκτελέσουμε.

Στην περίπτωση μας, η εφαρμογή λέγετε TEI και επομένως θα ανοίξουμε το συγκεκριμένο αρχείο.

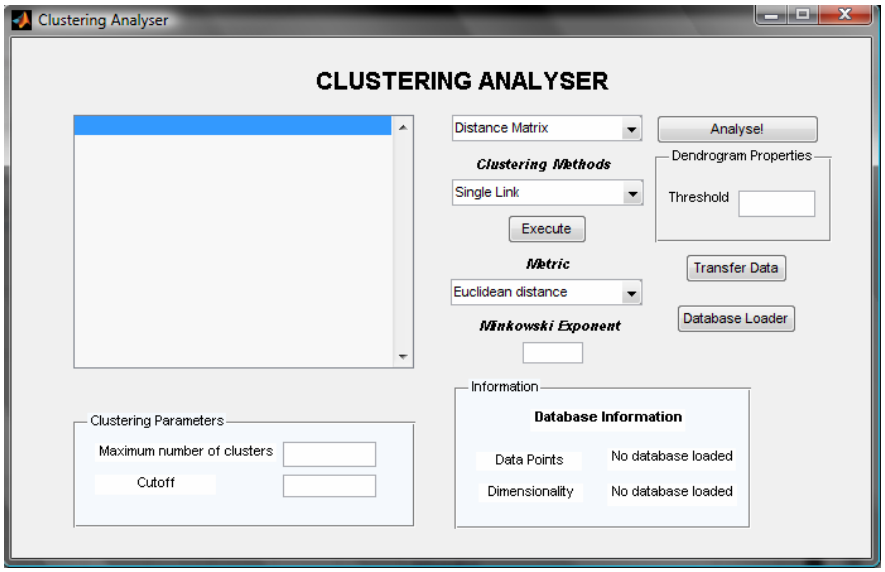
Βήμα 2

Αφού ανοίξει το αρχείο, πατάμε Run (το κουμπί με το πράσινο βελάκι) για να εκτελεστεί η εφαρμογή όπως φαίνεται στην παρακάτω εικόνα.



Εικόνα 10. Πιέζουμε το κουμπί Run για την εκτέλεση της εφαρμογής

Αφότου εκτελέσουμε την εφαρμογή θα έχουμε το ακόλουθο παράθυρο.

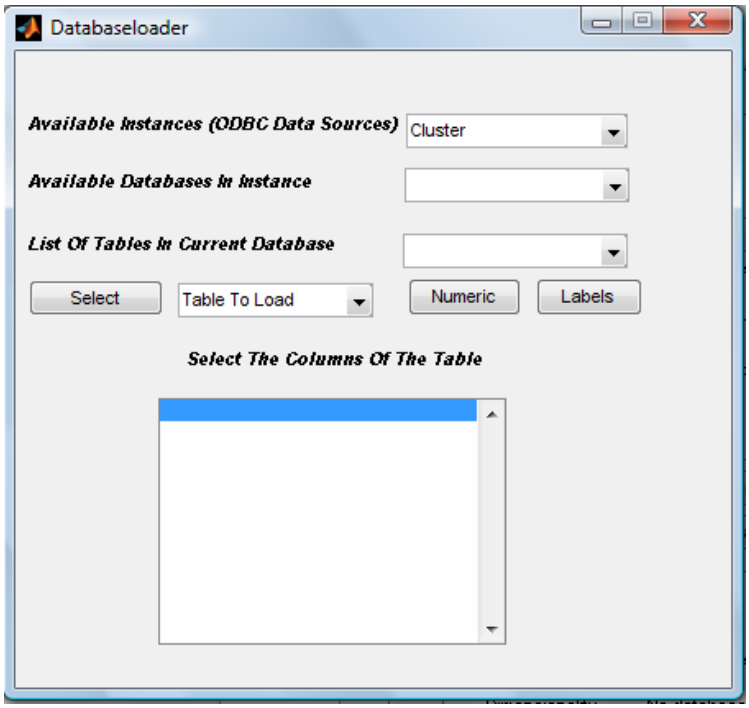


Εικόνα 11. Το παράθυρο της εφαρμογής Clustering Analyser

Βήμα 3

Στην εφαρμογή δεν έχει φορτωθεί κάποια βάση και επομένως δεν υπάρχουν δεδομένα προς επεξεργασία. Έτσι θα δούμε, το πρώτο πράγμα που πρέπει να κάνουμε είναι να πιέσουμε το κουμπί Database Loader που θα ανοίξει την αντίστοιχη εφαρμογή ώστε να φορτώσουμε δεδομένα από μία βάση της επιλογής μας.

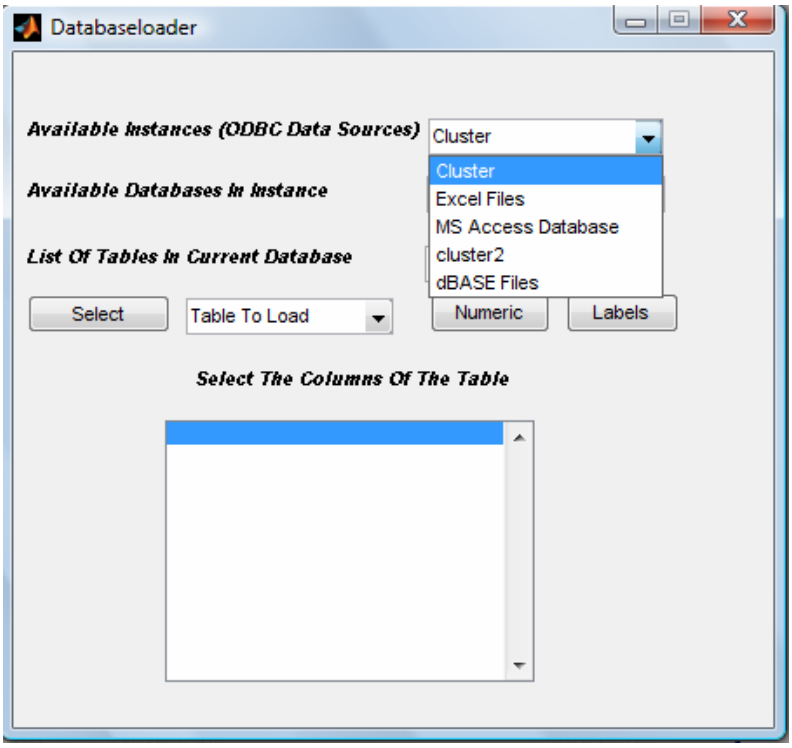
Πιέζοντας το κουμπί Database Loader θα εμφανιστεί το ακόλουθο παράθυρο.



Εικόνα 12. Το παράθυρο της εφαρμογής Database Loader

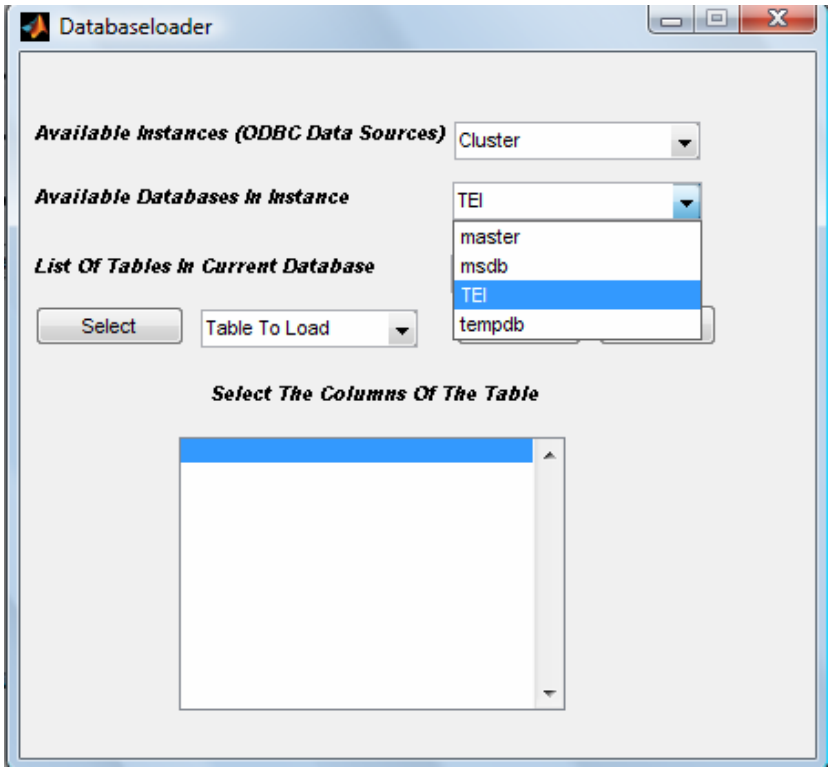
Βήμα 4

Η εφαρμογή αναγνωρίζει μόνη της τα διαθέσιμα data sources που υπάρχουν στον υπολογιστή. Στην δεδομένη περίπτωση επιλέγουμε το Cluster και άμεσα στο επόμενο μενού εμφανίζονται οι διαθέσιμες βάσεις δεδομένων για το data source που συνδεθήκαμε.



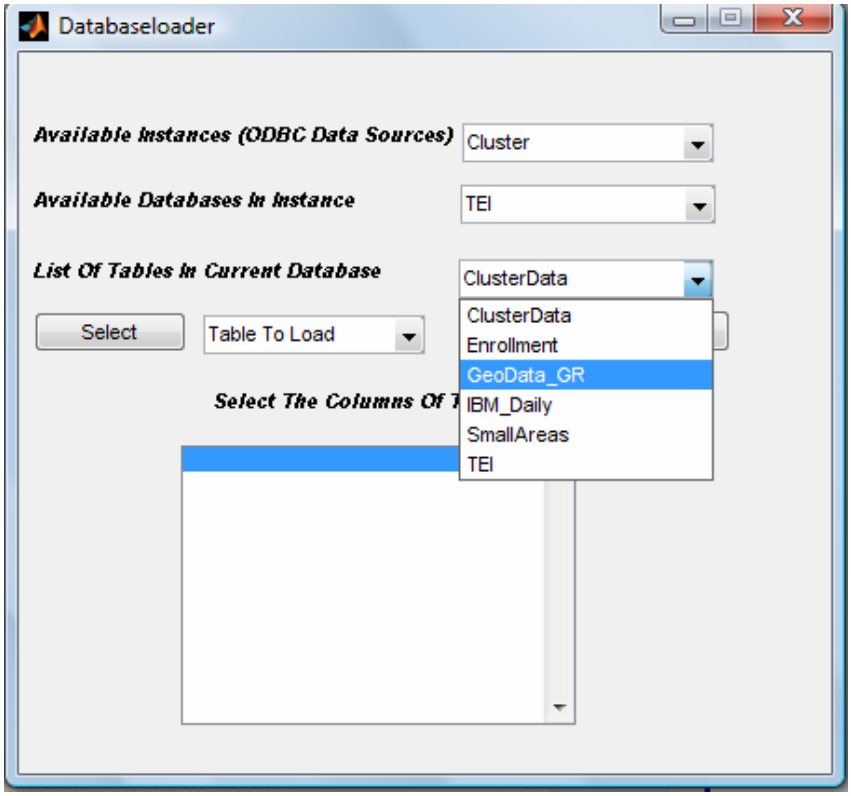
Βήμα 5

Από τις βάσεις δεδομένων που είναι διαθέσιμες μπορούμε να επιλέξουμε όποια επιθυμούμε. Ας επιλέξουμε την βάση TEI.



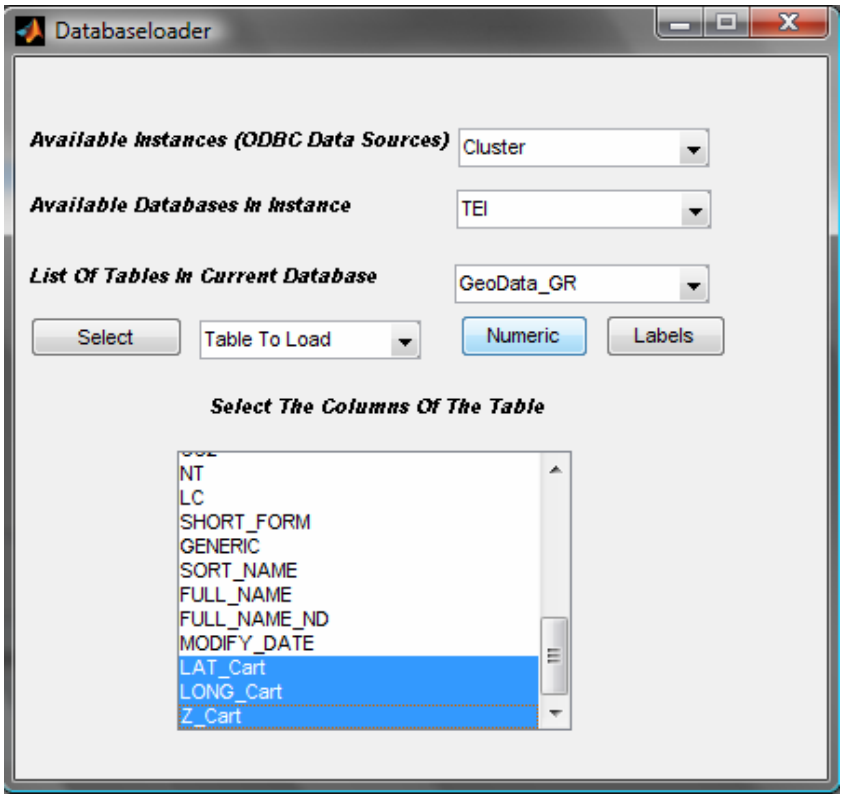
Βήμα 6

Ανάλογα με το ποιά βάση επιλέξαμε, εμφανίζονται και όλοι οι διαθέσιμοι πίνακες για την βάση δεδομένων που επιλέξαμε. Εμείς για παράδειγμα επιλέγουμε τον πίνακα GeoData_GR για την επεξεργασία των δεδομένων.

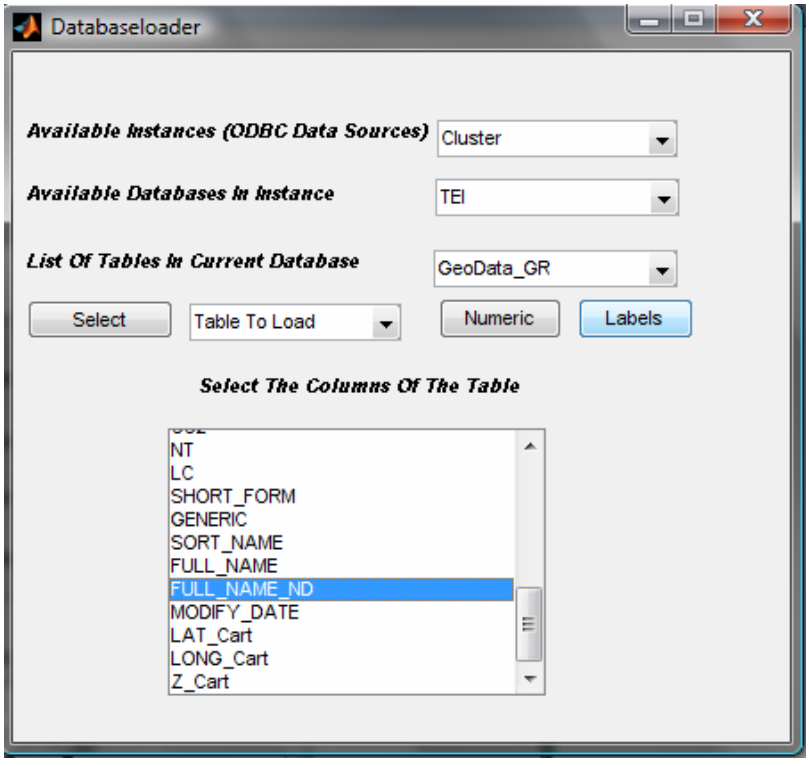


Βήμα 7

Με την επιλογή που κάναμε στον πίνακα, θα παρατεθούν όλες οι διαθέσιμες στήλες του. Στην παρακάτω εικόνα φαίνονται όλες οι στήλες του πίνακα GeoData_GR. Ας υποθέσουμε τώρα πως θέλουμε να επεξεργαστούμε πόλεις με βάση το γεωγραφικό μήκος τους και πλάτος τροποποιημένα κατάλληλα προς επεξεργασία σε καρτεσιανές συντεταγμένες. Πρέπει να επιλέξουμε τα αριθμητικά δεδομένα και να πατήσουμε το κουμπί Numeric. Έπειτα τις ετικέτες των δεδομένων και να πιέσουμε το κουμπί Labels.



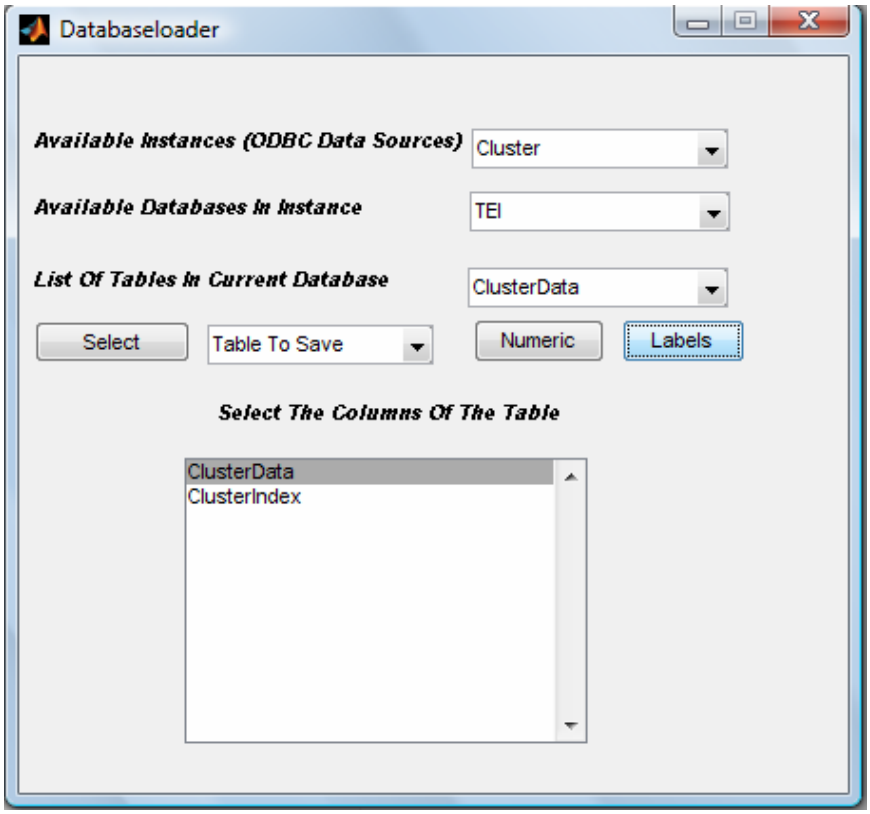
Εικόνα 13 Επιλογή Αριθμητικών Δεδομένων



Εικόνα 14 Επιλογή Ετικέτας Δεδομένων

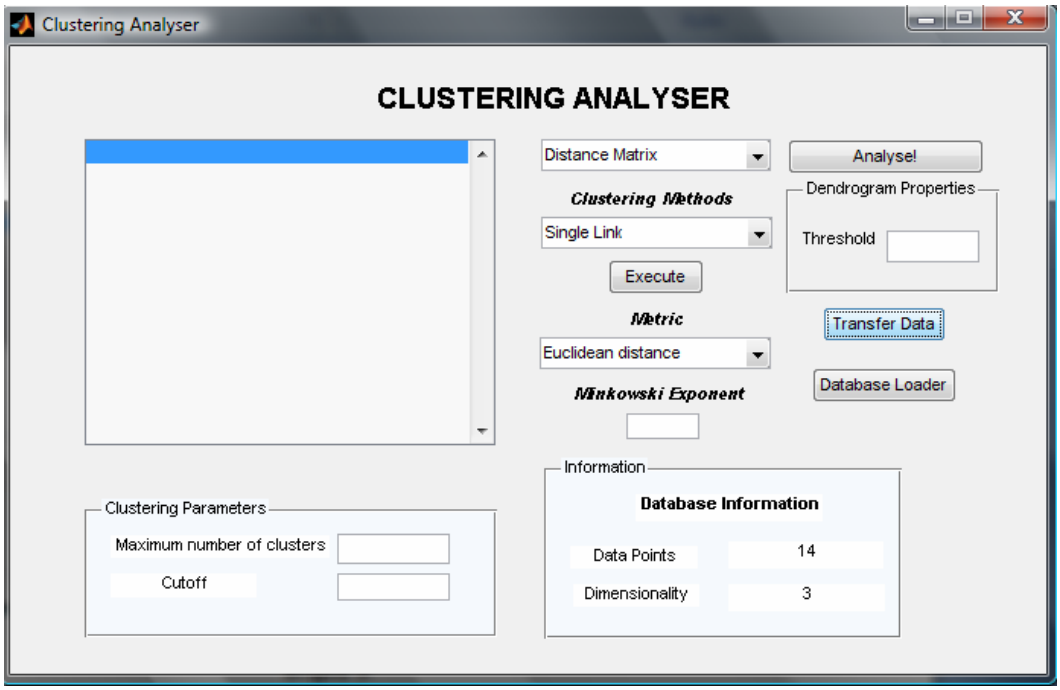
Έτσι μετά από τις επιλογές μας από το μενού, δίπλα στο κουμπί Select επιλέγουμε Table To Load ώστε να δηλώσουμε στην εφαρμογή ότι οι επιλογές μας περιέχουν τα δεδομένα προς επεξεργασία. Πρέπει τώρα να επιλέξουμε και ένα πίνακα από την βάση δεδομένων για να αποθηκεύσουμε αν χρειαστεί τα αποτελέσματα της συσταδοποίησης.

Επιλέγουμε τον πίνακα ClusterData αυτή τη φορά. Όπως φαίνεται και στην παρακάτω εικόνα, αλλάζουν και οι διαθέσιμες στήλες. Πρέπει να επιλέξουμε τα πεδία που θα αποθηκευτούν τα δεδομένα μας, δηλαδή την στήλη που θα περιέχει τα ονόματα των σημείων και την συστάδα που ανήκει το στοιχείο αντίστοιχα. Στην περίπτωσή μας θα είναι ClusterData και ClusterIndex. Πατάμε το κουμπί Labels και επιλέγουμε από το μενού Table to Save και πιέζουμε το κουμπί Select.



Βήμα 8

Είμαστε πλέον έτοιμοι να πάμε πίσω στην εφαρμογή Clustering Analyser και να μεταφέρουμε τα δεδομένα μας. Με το πάτημα του κουμπιού Transfer Data γίνεται η μεταφορά των δεδομένων και η εφαρμογή αυτόματα ενημερώνει το σύνολο των στοιχείων προς επεξεργασία και τις διαστάσεις τους, που είναι ουσιαστικά ο αριθμός των χαρακτηριστικών με τα οποία σκοπεύουμε να μελετήσουμε αυτά τα δεδομένα.



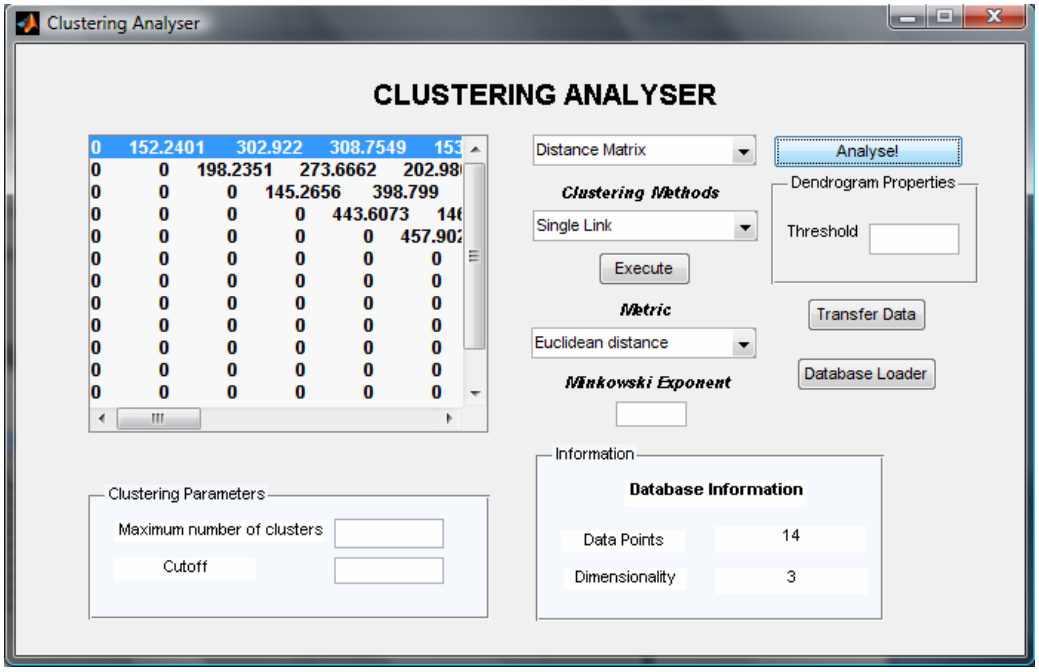
Βήμα 9

Το μενού που θα χρησιμοποιήσουμε κυρίως είναι αυτό δίπλα στο κουμπί Analyse! και περιέχει όλες τις διαθέσιμες επιλογές για την επεξεργασία των δεδομένων πλην των μεθόδων συσταδοποίησης και την επιλογή του μέτρου απόστασης.

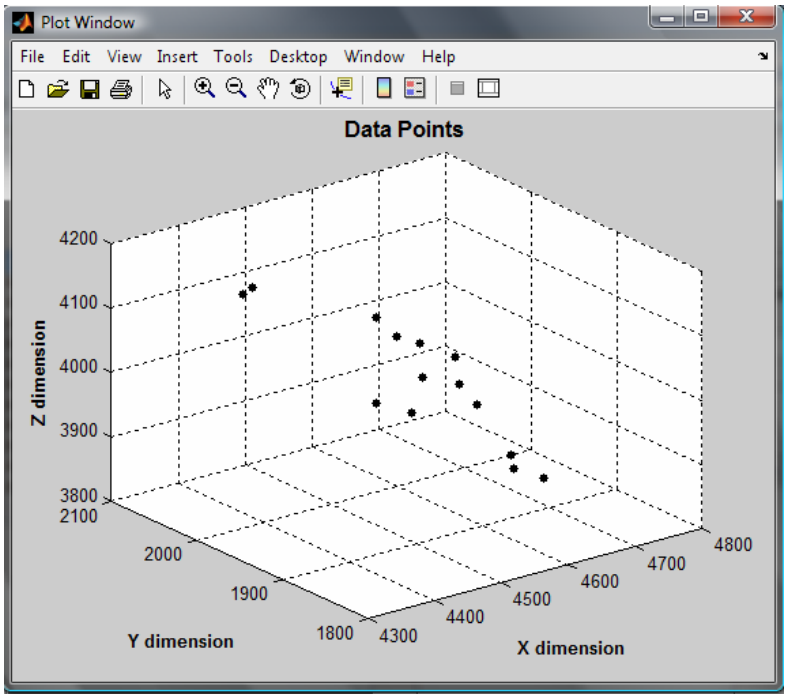
Metric : Οι διάφορες επιλογές του μενού Metric καθορίζουν με βάση τι μέτρο απόστασης θα υπολογιστεί ο πίνακας απόστασης για τις διάφορες μεθόδους συσταδοποίησης που έχουμε διαθέσιμες. Σε περίπτωση που θέλουμε να χρησιμοποιήσουμε τις μεθόδους συσταδοποίησης My Single Link ή My Complete Link πρέπει να κάνουμε χρήση του μέτρου My Euclidean distance.

Minkowski Exponent : Θέτουμε τον εκθέτη στον τύπο της μεθόδου Minkowski.

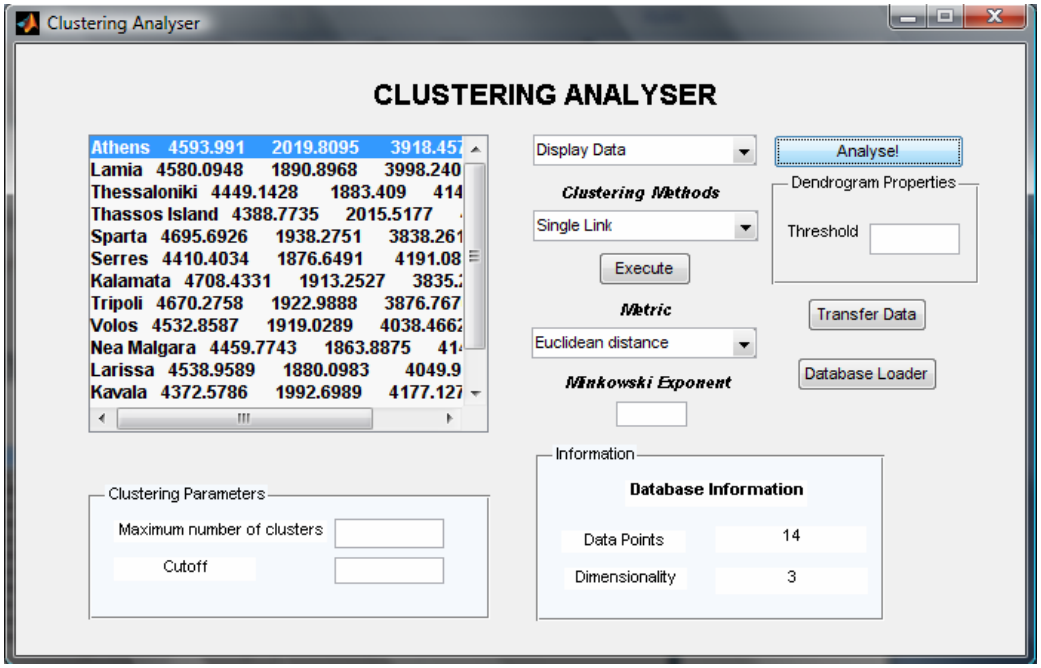
Distance Matrix : Με την επιλογή αυτή και το πάτημα του κουμπιού Analyse! Εμφανίζονται στο παράθυρο οι αποστάσεις των σημείων ανά ζεύγη σε ένα άνω τριγωνικό πίνακα αφού έχουμε όμως επιλέξει από το μενού Metric τον τρόπο με τον οποίο θέλουμε να υπολογιστεί ο πίνακας απόστασης. Στην προκειμένη περίπτωση χρησιμοποιήσαμε την ευκλείδεια απόσταση.



Plot Original Data : Με την επιλογή αυτή και το πάτημα του κουμπιού Analyse! εμφανίζεται ένα νέο παράθυρο που έχει την γραφική παράσταση των σημείων.



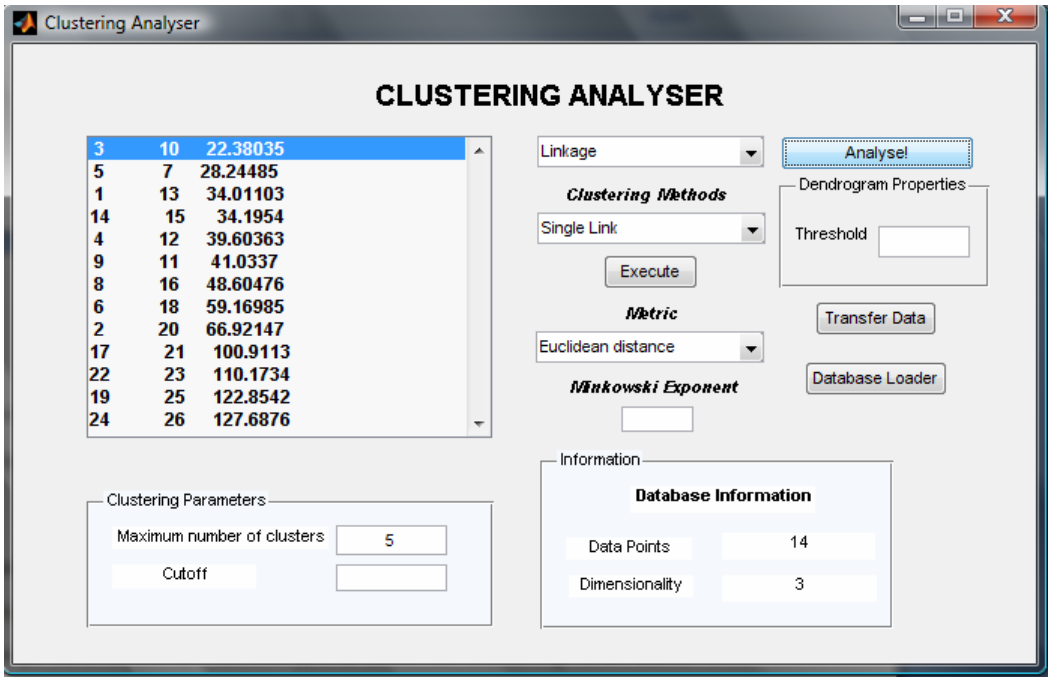
Display Data : Με την επιλογή αυτή και το πάτημα του κουμπιού Analyse! εμφανίζονται στο παράθυρο τα στοιχεία που φορτώσαμε. Στην προκειμένη περίπτωση τα ονόματα των πόλεων και οι καρτεσιανές τους συντεταγμένες.



Πριν δούμε τις υπόλοιπες επιλογές του μενού αυτού, πρέπει να δούμε πως μπορούμε να εκτελέσουμε κάποια μέθοδο συσταδοποίησης μιας και αυτές σχετίζονται με αποτελέσματα μετά από συσταδοποίηση.

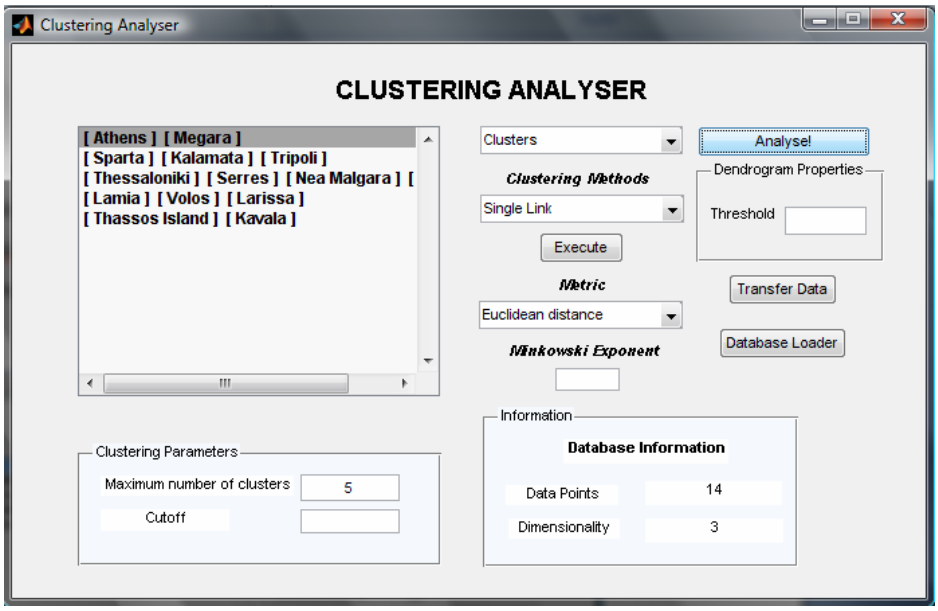
Χρησιμοποιούμε το μενού του Clustering Methods, που βρίσκεται ακριβώς αποκάτω του. Θα δούμε ότι έχουμε μια πληθώρα επιλογών για την μέθοδο συσταδοποίησης που θέλουμε να εκτελέσουμε και επίσης στα πεδία του Clustering Parameters πρέπει να ορίσουμε παραμέτρους, ανάλογα με το τί θέλουμε. Ας επιλέξουμε τη μέθοδο απλού δεσμού με το πεδίο Maximum number of clusters να είναι 5. Δηλαδή να γίνει συσταδοποίηση απλού δεσμού που να παράγει ακριβώς πέντε συστάδες και από το μενού της Metric, Euclidean Distance ώστε να χρησιμοποιήσουμε ως μέτρο τις ευκλείδειες αποστάσεις. Υπάρχουν και άλλα κριτήρια, όπως η απόσταση που ορίζεται στο πεδίο Cutoff. Η συσταδοποίηση εκτελείται πιέζοντας το κουμπί Execute.

Linkage : Με την επιλογή αυτή και το πάτημα του κουμπιού Analyse! εμφανίζονται στο παράθυρο τα βήματα της συσταδοποίησης. Το Matlab αριθμεί όλα τα στοιχεία με ένα αύξοντα αριθμό, στο παράδειγμά μας το σύνολο των στοιχείων είναι 14. Έπειτα δίνει ένα αναγνωριστικό αριθμό στις νέες συστάδες μεγαλύτερο από αυτόν, ξεκινώντας δηλαδή στην περίπτωσή μας από το 15. Ας δούμε τα παρακάτω αποτελέσματα. Στο πρώτο βήμα έχουμε τα στοιχεία 3 και 10 με απόσταση 22,38035. Στην συνέχεια τα στοιχεία 5 και 7 με απόσταση 28,24485. Έπειτα τα στοιχεία 1 και 13 με απόσταση 34,01103. Τα στοιχεία είναι σε σύνολο 14, επομένως παρακάτω που βλέπουμε το 14 με 15, ως 15 αναφέρεται πλέον η συστάδα που περιέχει τα (3,10) που είδαμε στην αρχή, απόσταση 34,1954. Η απόσταση όπως παρατηρούμε συνεχώς αυξάνεται.

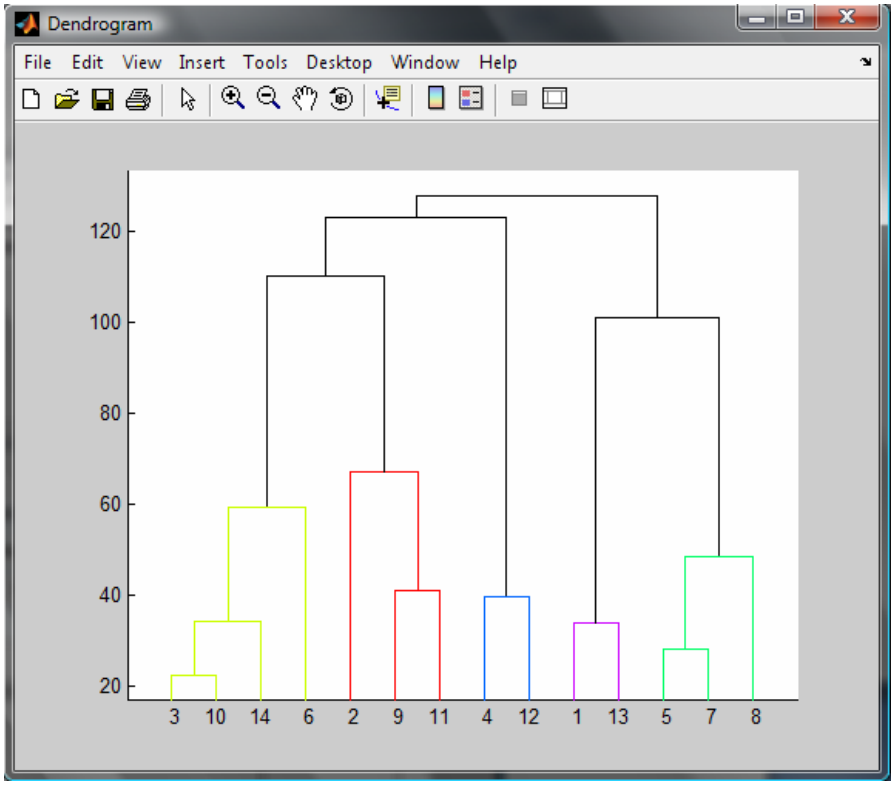


Clusters : Με την επιλογή αυτή και το πάτημα του κουμπιού Analyse! εμφανίζονται στο παράθυρο οι συστάδες που δημιουργήθηκαν με την εκτέλεση της μεθόδου συσταδοποίησης που επιλέξαμε. Βλέπουμε πως κάθε γραμμή περιέχει τα στοιχεία

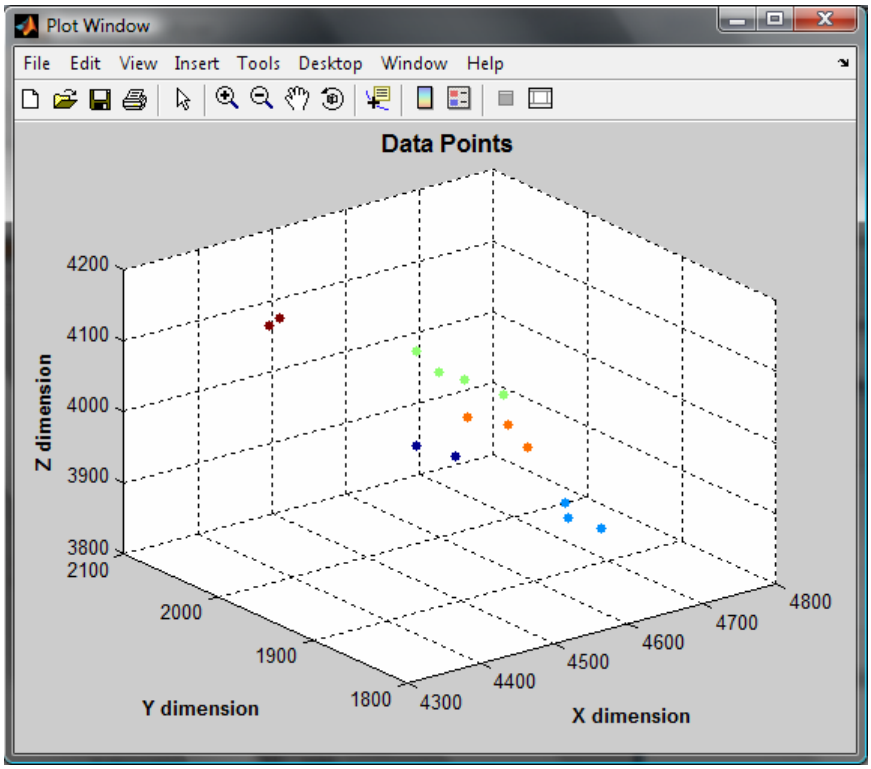
της αντίστοιχης συστάδας. Στην πρώτη συστάδα για παράδειγμα είναι η Αθήνα και τα Μέγαρα.



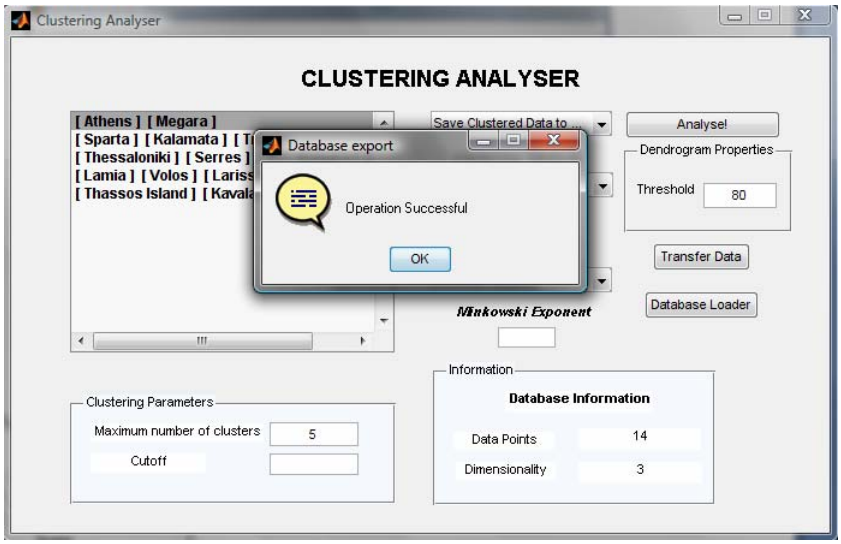
Dendrogram : Με την επιλογή αυτή και το πάτημα του κουμπιού Analyse! εμφανίζεται ένα νέο παράθυρο που περιέχει το δενδρόγραμμα. Έχουμε μία επιλογή για το δενδρόγραμμα, το Threshold όπου δίνει ξεχωριστό χρώμα σε κάθε ομάδα όπου η απόσταση είναι μικρότερη από αυτή που ορίσαμε. Για το παρακάτω δενδρόγραμμα χρησιμοποιήσαμε Threshold 80 . Για την επιλογή του Threshold μπορούμε να πάρουμε βοήθεια από τις αποστάσεις που δίνει η Linkage. Ορίζοντας Threshold 0 ουσιαστικά απενεργοποιούμε αυτή τη λειτουργία και παίρνουμε ένα βασικό δενδρόγραμμα.



Plot Clustered Data : Με την επιλογή αυτή και το πάτημα του κουμπιού Analyse! εμφανίζεται ένα νέο παράθυρο που περιέχει την γραφική παράσταση των δεδομένων σε σχέση με την συστάδα που ανήκουν. Μας δίνεται έτσι μία καλύτερη απεικόνιση στον χώρο. Δίνεται διαφορετικό χρώμα για τα στοιχεία κάθε συστάδας. Έτσι βλέπουμε και την σχετικότητα τους στον χώρο. Παρακάτω για τα στοιχεία του παραδείγματός μας, τα πράσινα αποτελούν μία συστάδα , το μπλε άλλη και το κόκκινο άλλη.



Save Clustered Date to Database: Με την επιλογή αυτή και το πάτημα του κουμπιού Analyse! , αφού πρότινος έχουμε εκτελέσει κάποια μέθοδο συσταδοποίησης, αποθηκεύονται στην βάση δεδομένων, τον πίνακα και τις στήλες που είχαμε επιλέξει τα δεδομένα μας μετά την συσταδοποίηση. Με την επιτυχής εκτέλεση εμφανίζεται και το ακόλουθο μήνυμα.



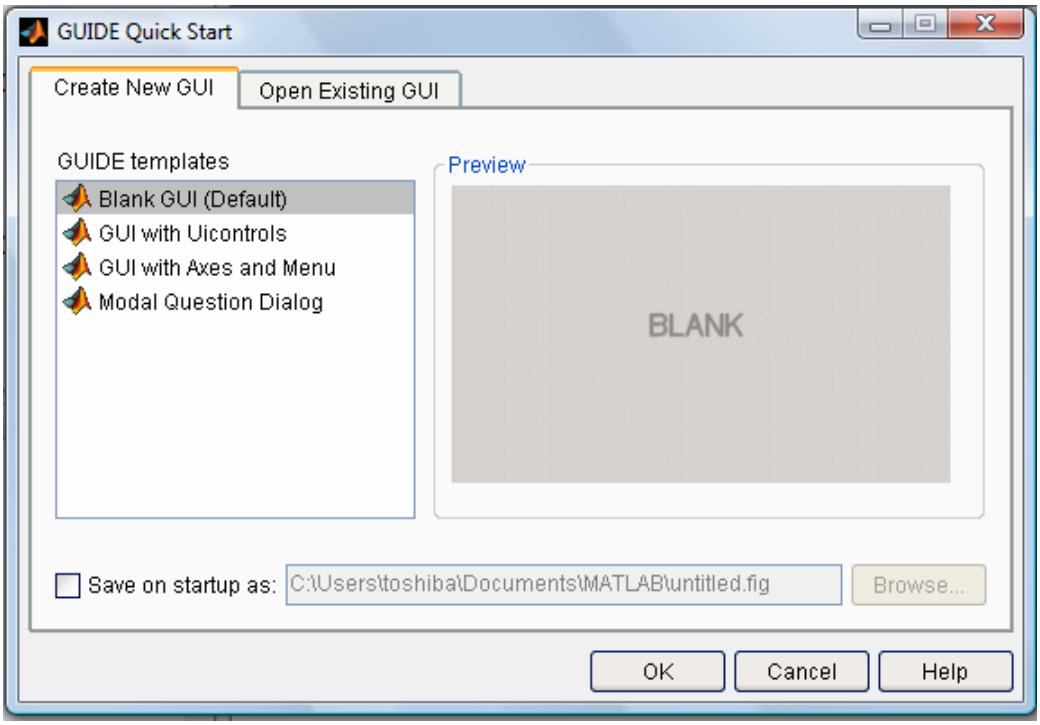
Τα δεδομένα πλέον υπάρχουν στην βάση δεδομένων υπό την παρακάτω μορφή :

	ClusterData	ClusterIndex
1	Athens	1
2	Megara	1
3	Sparta	2
4	Kalamata	2
5	Tripoli	2
6	Thessaloniki	3
7	Serres	3
8	Nea Malgara	3
9	Veria	3
10	Lamia	4
11	Volos	4
12	Larissa	4
13	Thassos Isl...	5
14	Kavala	5

4. Υλοποίηση Εφαρμογής Συσταδοποίησης σε Περιβάλλον Matlab.

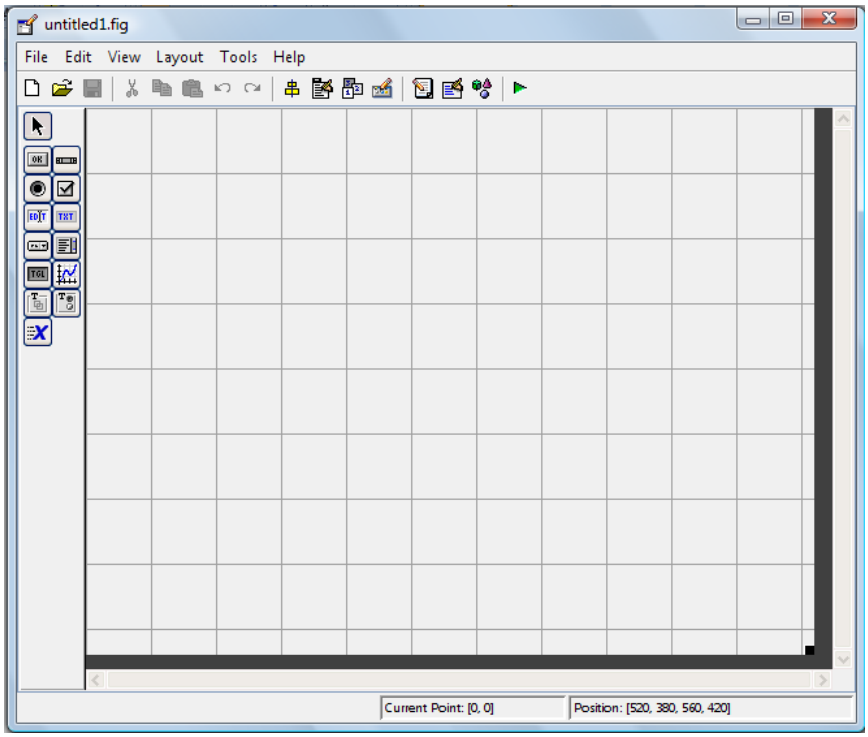
4.1 Αρχικά Βήματα Υλοποίησης

Το πρώτο που πρέπει να κάνουμε για να ξεκινήσουμε να εργαζόμαστε για να υλοποιήσουμε ένα γραφικό περιβάλλον στο Matlab, είναι να πληκτρολογήσουμε την εντολή guide. Η εντολή αυτή θα μας δώσει το ακόλουθο παράθυρο.



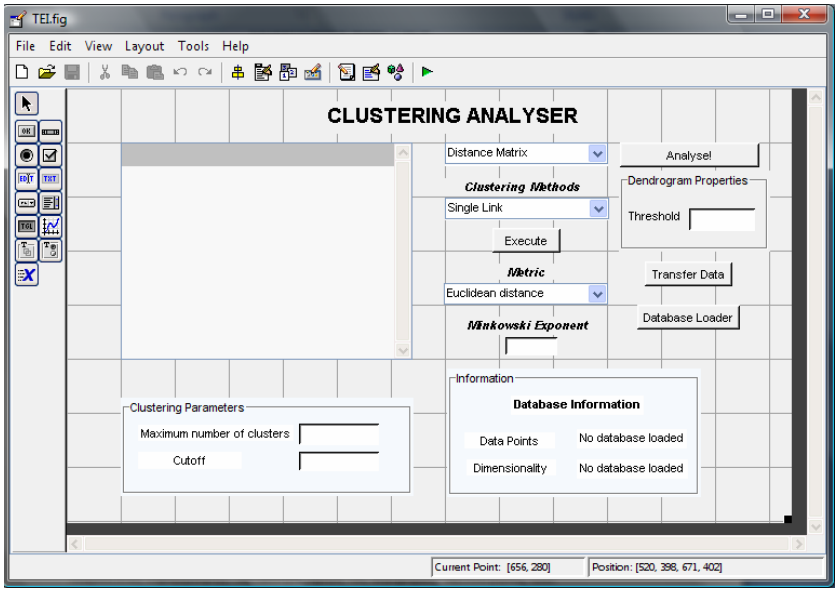
Εικόνα 15. Επιλογές με την εκτέλεση της εντολής guide

Έχουμε τις εξής επιλογές λοιπόν. Είτε να δημιουργήσουμε ένα νέο γραφικό περιβάλλον, είτε να τροποποιήσουμε ένα υπάρχον. Η τροποποίηση γίνεται αν πάμε στην καρτέλα Open Existing GUI, εκεί μπορούμε να επιλέξουμε ένα από τα διαθέσιμα γραφικά περιβάλλοντα που είτε δημιουργήσαμε, είτε προϋπάρχουν. Στην περίπτωσή μας θέλουμε να δημιουργήσουμε ένα νέο γραφικό περιβάλλον, οπότε όπως φαίνεται στην παραπάνω εικόνα είμαστε στην καρτέλα Create New GUI και για την περίπτωσή μας επιλέγουμε Blank GUI (Default), μία κενή φόρμα θα εμφανιστεί λοιπόν, παρόμοια με αυτή που φαίνεται στην παρακάτω εικόνα.



Εικόνα 16. Κενή Φόρμα

Στην παραπάνω εικόνα βλέπουμε πως στην αριστερή πλευρά υπάρχουν διαφορά components που μπορούμε να τοποθετήσουμε στην φόρμα μας ώστε να υλοποιήσουμε την εφαρμογή. Η παρούσα εφαρμογή που υλοποιήθηκε στα πλαίσια αυτής της εργασίας έχει ως εξής :



Εικόνα 17. Φόρμα Της Εφαρμογής Clustering Analyser

Εδώ βλέπουμε πως στην φόρμα έχουν τοποθετηθεί διάφορα components, όπως static text, buttons, panels, listbox,pop-up menu και edit box. Κάθε component έχει μία σειρά από συναρτήσεις (functions) τις οποίες καλεί ανάλογα με τις επιλογές του χρήστη, όπως για παράδειγμα η συνάρτηση Callback της οποίας ο κώδικας εκτελείται με το πάτημα του button ή αντίστοιχα του εκάστοτε component και αποτελεί την πιο συχνά χρησιμοποιούμενη συνάρτηση για την υλοποίηση της παρούσας εφαρμογής. Ας δούμε όμως τις διάφορες λειτουργίες του προγράμματος.

4.2 Υπολογισμός και εμφάνιση πίνακα αποστάσεως

Ο κώδικας ανήκει στο component με ετικέτα Analyse που αποτελεί ένα από τα βασικά μέρη του Clustering Analyser, συνεργάζεται κυρίως με το pop-up menu το οποίο βρίσκεται δίπλα του, το οποίο έχει ετικέτα (tag), δηλαδή ονομασία με την οποία το Matlab το αναγνωρίζει, Choices.

```
   if (get(handles.Choices,'Value')==1)
        if(strcmp(handles.CDist,'minkowski'))
            distance=
pdist(handles.DBdata,handles.CDist,str2num(get(handles.Exponent,'Stri
ng')));
            squared= squareform(distance);
        else
            if(strcmp(handles.CDist,'Myeuclidean'))

                  distance=CluD(handles.DBdata);
                  for i= 1:size(distance)
                    distance(i,i)=0;
                  end
                  squared=distance;
            else
             distance= pdist(handles.DBdata,handles.CDist);
             squared= squareform(distance);
            end
        end
        sdistance= triu(squared);
        set(handles.Temporary,'String',num2str(sdistance));
    elseif (get(handles.Choices,'Value')==3)
        Data= num2str(handles.DBdata);
        sourcesize= size(handles.DBSdata);
        for i=1:sourcesize(1,1)
            string{i}=sprintf('%s   ',num2str(handles.DBSdata{i}));
        end
        string=string' ;
        set(handles.Temporary,'String',strcat(string,Data));
```

Εδώ βλέπουμε ότι έχουμε έναν έλεγχο με την συνθήκη if στην μεταβλητή handles.Choices και συγκεκριμένα το πεδίο Value. Σε μία δομή handles το Matlab κρατάει όλες τις μεταβλητές και τα στοιχεία των component που δημιουργούμε. Επομένως η μεταβλητή handles.Choices αναφέρεται στα στοιχεία του component

με ετικέτα Choices και συγκεκριμένα στην τιμή του πεδίου Value. Το πεδίο Value σε ένα component έχει τιμή 0 εάν δεν έχει γίνει κάποια επιλογή σε αυτό, την τιμή 1 για την πρώτη επιλογή, την τιμή 2 όταν είναι επιλεγμένη η δεύτερη και πάει λέγοντας. Χρησιμοποιείται λοιπόν για να γνωρίζουμε πια επιλογή έχει κάνει ο χρήστης και είναι κυρίως χρήσιμο για τα pop-up menu και άλλα παρόμοια components όπως το listbox. Ελέγχουμε εδώ λοιπόν εάν η τιμή του πεδίου Value του component Choices έχει τιμή 1, δηλαδή ο κώδικας παρακάτω εκτελείται μόνο εάν ο χρήστης έχει επιλέξει το πρώτο αντικείμενο του pop-up menu Choices. Αν ναι, ελέγχουμε με τη συνάρτηση strcmp το είδος της απόστασης που θέλει να υπολογίσει ο χρήστης και ανάλογα γίνεται η εκτέλεση του κατάλληλου κώδικα. Ειδικές περιπτώσεις αποτελούν η Minkowski στην οποία ο χρήστης ορίζει τον εκθέτη του τύπου και η ευκλείδεια απόσταση που έχουμε υλοποίηση εμείς. Στην περίπτωση της Minkowski ή κάποιας άλλης απόστασης εκτός της υλοποίησης μας, ο κώδικας τοποθετεί σε μία μεταβλητή distance την απόσταση που υπολογίζει με την συνάρτηση pdist η οποία παίρνει ως είσοδο ένα πίνακα και την μέθοδο με την οποία θέλουμε να υπολογίσουμε τις αποστάσεις των σημείων, η μέθοδο αυτή της είναι της ευκλείδειας απόστασης (Euclidean Distance),η Minkowski, η Manhattan αλλιώς γνωστή και ως city block και η απόσταση Chebyshev ή σε περίπτωση που έχουμε την απόσταση Minkowski παίρνει ακόμα ως είσοδο και τον εκθέτη του τύπου της μεθόδου Minkowski. Παρακάτω η εντολή squareform μετατρέπει τον πίνακα που παράχθηκε από την pdist σε τετραγωνική μορφή και με τέτοιο τρόπο ώστε να φαίνονται στις θέσεις του πίνακα οι αποστάσεις των σημείων ανά ζεύγη. Το στοιχείο (1,2) δείχνει την απόσταση του σημείου 1 από το 2 για παράδειγμα. Η επόμενη εντολή triu κάνει τον πίνακά μας άνω τριγωνικό. Τέλος, με την συνάρτηση set θέτουμε στο πεδίο String του component ονόματι Temporary όπου στην προκειμένη περίπτωσή μας είναι το listbox που βρίσκεται δίπλα τα περιεχόμενα του πίνακα sdistance. Το πεδίο String αναφέρεται στο κείμενο του εκάστοτε component και συγκεκριμένα στο listbox θα εμφανιστούν τα περιεχόμενα του πίνακα. Η συνάρτηση num2str μετατρέπει τον πίνακά μας σε αλφαριθμητικό διότι τέτοια είσοδο δέχεται το πεδίο String.Όσον αφορά την υλοποίηση της ευκλείδειας απόστασης που έχουμε κάνει, καλούμε την συνάρτηση CluD η οποία είναι υπεύθυνη για τον υπολογισμό της ευκλείδειας απόστασης, με μία επαναληπτική δομή for αφαιρούμε τα στοχεία NaN από την κύρια διαγώνιο και δεν κάνουμε χρήση της εντολής squareform σε αυτή την περίπτωση διότι ο πίνακας μας έχει ήδη την κατάλληλη μορφή. Έπειτα όπως και πριν η triu κάνει τον πίνακα μας άνω τριγωνικό.

4.3 Εμφάνιση στοιχείων προς επεξεργασία από τη βάση δεδομένων

```
elseif (get(handles.Choices,'Value')==3)
    Data= num2str(handles.DBdata);
    sourcesize= size(handles.DBSdata);
    for i=1:sourcesize(1,1)
        string{i}=sprintf('%s   ',num2str(handles.DBSdata{i}));
    end
    string=string' ;
    set(handles.Temporary,'String',strcat(string,Data));
```

Εάν έχουμε κάνει την τρίτη επιλογή στο μενού με ετικέτα Choices λοιπόν η οποία εάν πάμε στο Property Inspector και στο πεδίο String βλέπουμε πως είναι η Display Data, μετατρέπουμε την μεταβλητή handles.DBdata που αποτελεί μία μεταβλητή η οποία φαίνεται από όλες τις συναρτήσεις και όχι μόνο από την παρούσα και κρατάει τα δεδομένα μας σε αριθμητική μορφή. Με την εντολή size παίρνουμε τις γραμμές και στήλες του πίνακα που περιέχεται στην μεταβλητή handles.DBSdata. Με μία επαναληπτική δομή από 1 έως τις γραμμές του πίνακα των οποίων ο αριθμός υπάρχει στη μεταβλητή sourcesize και είναι και ο συνολικός αριθμός των στοιχείων μας θέτουμε στην μεταβλητή string η οποία είναι της μορφής cellarray μία προς μία τις τιμές του handles.DBSdata και συγκεκριμένα την πρώτη στήλη που περιέχει τα ονόματα των δεδομένων μας σε αλφαριθμητική μορφή. Με την εντολή sprintf , το %s δέχεται το αλφαριθμητικό από τον πίνακα, έπειτα αφήνουμε μερικά κενά για να καλύτερη απεικόνιση μετέπειτα. Μετά αναστρέφουμε τον πίνακα string με το σύμβολο ‘ . Θέτουμε στο πεδίο String του Temporary το αποτέλεσμα της strcat(string,Data) η οποία ενώνει ένα προς ένα τα αλφαριθμητικά των πινάκων string και Data.

4.4 Εκτέλεση γραφικής παράστασης αρχικών δεδομένων

```
elseif (get(handles.Choices,'Value')==2)
%The following code creates a new figure and assings the handle to
         %Plotdata variable which handles the plot data.
         %The if condition is used to distinguish 2 dimensions and 3.
         if(str2num(get(handles.Dimensionality,'String'))==2)
            PlotData              =              figure('Name','Plot
Window','NumberTitle','off'); %Create a new figure for my plot

scatter(handles.DBdata(:,1),handles.DBdata(:,2),30,'filled','k')
            title('Data Points', 'FontWeight','Bold','FontSize',12)
            xlabel('X   dimension','FontWeight','Bold')   %label   the
xaxis
            ylabel('Y   dimension','FontWeight','Bold')   %label   the
yaxis
        %The following lines of code work for 3 dimensional arrays.
```

Για την δεύτερη επιλογή του pop-up μενού με ετικέτα Choices έχουμε την Plot Original Data, η οποία κάνει την γραφική παράσταση των αρχικών μας δεδομένων. Ο κώδικας που ακολουθεί μετά το σύμβολο % είναι σχόλιο και αγνοείται. Εδώ κύρια σημεία αποτελούν τα εξής. Έχουμε τον έλεγχο με την συνθήκη if του πεδίου String του component Dimensionality το οποίο κρατάει τις διαστάσεις

των δεδομένων μας. Με την συνάρτηση get παίρνουμε την τιμή του πεδίου String του component Dimensionality και η οποιά είναι ανάλογη της set με την διαφορά ότι επιστρέφει την τιμή του πεδίου. Με την str2num τώρα μετατρέπουμε αυτό που πήραμε από την get σε αριθμό μιας και θέλουμε να το συγκρίνουμε με τον αριθμό 2 και το πεδίο String περιέχει μεν αυτό που ζητάμε αλλά σε αλφαριθμητική μορφή. Ουσιαστικά λοιπόν εάν τα δεδομένα μας είναι δυσδιάστατα, καλούμε μία νέα φόρμα για την συνάρτηση που οι δομές τις αποθηκεύονται στην μεταβλητή PlotData. Η συνάρτηση scatter εκτελεί την γραφική παράσταση των σημείων μας με διάφορες παραμέτρους όπως το νούμερο 30 που αποτελεί το μέγεθος τον dots στην γραφική παράσταση, το filled που γεμίζει τα σημεία με χρώμα, το k που θέτει το χρώμα των σημείων σε μαύρο, στο title,xlabel και ylabel θέτουμε αντίστοιχα το όνομα των αξόνων και της γραφικής παράστασης όπως και διάφορες παραμέτρους για την γραμματοσειρά που θα χρησιμοποιηθεί.

Παρακάτω έχουμε :

```
elseif (str2num(get(handles.Dimensionality,'String'))==3)
         PlotData = figure('Name','Plot Window','NumberTitle','off');
%Create a new figure for my plot

scatter3(handles.DBdata(:,1),handles.DBdata(:,2),handles.DBdata(:,3),
30,'filled','k')
            title('Data Points', 'FontWeight','Bold','FontSize',12)
            xlabel('X   dimension','FontWeight','Bold')   %label   the
xaxis
            ylabel('Y   dimension','FontWeight','Bold')   %label   the
yaxis
            zlabel('Z   dimension','FontWeight','Bold')   %label   the
zaxis
```

Αυτό το κομμάτι του κώδικα εκτελείται για δεδομένα τριών διαστάσεων, μερικές διαφορές είναι ότι τώρα χρησιμοποιούμε την scatter3 η οποία εκτελεί την αντίστοιχη γραφική παράσταση της αρχικής scatter αλλά για τρεις διαστάσεις.

Έπειτα :

```
         elseif(str2num(get(handles.Dimensionality,'String'))==1)
      %We reversed the dimensions to display our data in Y dimension.
            PlotData              =              figure('Name','Plot
Window','NumberTitle','off'); %Create a new figure for my plot
            points= size(handles.DBdata);

scatter(1:points(1,1),handles.DBdata(:,1),30,'filled','k')
            title('Data Points', 'FontWeight','Bold','FontSize',12)
            xlabel('Segments','FontWeight','Bold') %label the xaxis
            ylabel('Data','FontWeight','Bold') %label the yaxis
```

Εδώ έχουμε μία απόπειρα για την αναπαράσταση δεδομένων μίας διάστασης και με τρόπο όσο το δυνατόν γίνεται ανεξάρτητο από τα δεδομένα. Η

μεταβλητή points κρατά αυτό που επιστρέφει η size , εμείς θέλουμε των αριθμό των γραμμών όπου είναι και ο αριθμός των στοιχείων μας. Έτσι εκτελούμε γραφική παράσταση με την συνάρτηση scatter και στον άξονα x να βρίσκονται σημεία από 1 έως το σύνολο των σημείων των δεδομένων μας. Έτσι για παράδειγμα η κάθε μονάδα μπορεί να υποδηλώνει μία μέρα, ένα χιλιόμετρο, μία ώρα και πάει λέγοντας.

4.5 Εμφάνιση συστάδων

```
elseif (get(handles.Choices,'Value')==5)
          %Tcluster=handles.Tcluster;
            Data= handles.DBSdata;
            CData= handles.Tcluster;
            Maxclusters= max(CData);
            Savesize= size(CData);
            Stringdata= [];
            counter=1;
            for i=1:Maxclusters
                Positions= find(CData==i);
                Possize= size(Positions);
                    for j=1:Possize(1,1)
                        for k=1:Savesize
                            if(k==Positions(j,1))
                                Stringdata{counter}=Data{k,1};
                                counter=counter+1;
                            end
                        end
                    end
            Dsize=size(Stringdata);

           % Stringdata= Stringdata' ;
           string{i}=sprintf(' [ %s ] ',Stringdata{:});
            %for c=1:Dsize(1,2)
            %    {c,2}=i;
            %end
            %exdata{i}= Stringdata;
            clear Stringdata;
            counter=1;
            end
            %string=string' ;
          set(handles.Temporary,'Value',1);
          set(handles.Temporary,'String',string);
```

Η πέμπτη επιλογή του μενού με ετικέτα Choices η οποία και ονομάζεται Clusters. Μετά την εκτέλεση κάποιας συσταδοποίησης, εμφανίζει τα αποτελέσματα στο listbox Temporary. Η συνάρτηση max επιστρέφει το μέγιστο στοιχείο του πίνακα handles.Tcluster ο οποίος κρατά τις συστάδες μετά την εκτέλεση της συσταδοποίησης, και επομένως παίρνοντας το μέγιστο στοιχείο από αυτό τον πίνακα, παίρνουμε τον αριθμό των συστάδων που δημιουργήθηκαν συνολικά κατά την εκτέλεση. Η μεταβλητή Savesize κρατά το μέγεθος του πίνακα CData και ουσιαστικά τον αριθμό των δεδομένων μας. Ορίζουμε μία μεταβλητή Stringdata και

τη θέτουμε κενή και μία μεταβλητή counter με τιμή 1 όπου θα δούμε τη χρησιμότητά της εντός της επαναληπτική δομής. Εκτελούμε για όσες συστάδες έχουμε τα παρακάτω. Οι συστάδες ακολουθούν μία μορφή για παράδειγμα από 1 έως και 10, αφού αν ορίσουμε πρότινος ότι θέλουμε 3 συστάδες αυτές θα είναι οι 1,2 και 3. Έτσι με την εντολή find βρίσκουμε για παράδειγμα στην πρώτη επανάληψη όλα τα δεδομένα που ανήκουν στην συστάδα 1. Η μορφή του πίνακα που κρατά τις συστάδες είναι τέτοια, ώστε περιέχει νούμερα, για παράδειγμα 1,2,2,4,4 σημαίνει πως το $1^{ο}$ στοιχείο είναι στην πρώτη συστάδα, το $2^{ο}$ στην δεύτερη, το $3^{ο}$ στην δεύτερη, το $4^{ο}$ στην τέταρτη και το $5^{ο}$ στην τέταρτη. Η find λοιπόν επιστρέφει τις θέσεις των στοιχείων του πίνακα που κρατά τις συστάδες και για την πρώτη επανάληψη θα επιστρέψει παραδείγματος χάριν τις θέσεις των στοιχείων της πρώτης συστάδας, δηλαδή που βρίσκονται όλοι οι άσσοι. Μετά παίρνουμε το μέγεθος του πίνακα Positions για να δούμε πόσα στοιχεία τελικά ανήκουν στην συστάδα και εκτελούμε τα παρακάτω. Έπειτα στις δύο παρακάτω επαναληπτικές δομές εκτελούμε για όσες συστάδες έχουμε με την μεταβλητή Savesize και βρίσκουμε τις θέσεις των δεδομένων μας η οποία είναι στην μεταβλητή k και θέτουμε την τιμή της Data η οποία περιέχει τα δεδομένα της handles.DBSdata σε μία cellarray μεταβλητή ονόματι StringData που ορίσαμε πιο πάνω. Η μεταβλητή counter χρειάζεται στο να μετακινούμαστε στην επόμενη θέση μετά από κάθε εγγραφή στην StringData εφόσον καμία από τις μεταβλητές των επαναληπτικών δομών δε φαίνεται να μας εξυπηρετεί για αυτή την δουλειά. Με την λήξη των επαναληπτικών δομών, η StringData περιέχει όλα τα ονόματα των σημείων που ανήκουν στην αντίστοιχη συστάδα με την σειρά που βρέθηκαν στον πίνακα των δεδομένων. Θέτουμε αυτά σε μία θέση της μεταβλητής string, η οποία και εξαρτάτε από την τιμή που έχει η μεταβλητή i, την προκειμένη στιγμή με την βοήθεια της sprintf. Μία διαφορά που βλέπουμε είναι το σύμβολο : εντός της Stringdata που ουσιαστικά δείχνει προς όλα τα στοιχεία της μεταβλητής Stringdata. Άξιο αναφοράς σε αυτό το σημείο είναι επίσης ότι στις μεταβλητές είδους cellarray, για να προσπελάσουμε τα περιεχόμενα του κελιού δεν χρησιμοποιούμε ( ) , αλλά { } . Έτσι παραδείγματος χάριν, StringData{1,2} αναφέρετε στο περιεχόμενο του κελιού της γραμμής 1 και στήλης 2. Με την εντολή clear σβήνουμε την μεταβλητή StringData, δηλαδή αυτήν μαζί με τα περιεχόμενα της και επαναφέρουμε τη μεταβλητή counter στην τιμή 1. Την επόμενη επανάληψη δηλαδή θα γραφούν στην StringData τα ονόματα των δεδομένων που ανήκουν στη δεύτερη συστάδα και θα περαστούν στη δεύτερη θέση της μεταβλητής string. Με το τέλος της αρχικής επαναληπτικής δομής θα έχουμε στην μεταβλητή string τα ονόματα όλων των δεδομένων σε θέσεις ανάλογες με την συστάδα που ανήκουν, και με τη βοήθεια της εντολής set τα εμφανίζουμε στο listbox Temporary. Να σημειωθεί εδώ πως θέτουμε το πεδίο Value του Temporary ίσο με ένα για να αποφύγουμε μία ιδιοτροπία του Matlab αφού αν φορτώσουμε το listbox με πολλά στοιχεία το Value μεγαλώνει αυτόματα και αν τα στοιχεία που προβάλλονται στο listbox αλλάξουν, το Value παραμένει μεγάλο και

out of bounds με αποτέλεσμα να εξαφανίζεται το component. Θέτουμε την τιμή ένα αφού το Matlab δεν το κάνει μόνο του αυτόματα κάθε φορά που αλλάζει το πεδίο Strings του listbox.

4.6 Κατασκευή δενδρογράμματος

```
    elseif (get(handles.Choices,'Value')==6)
              if(handles.methodone==0)
                msgbox('Dendrogram data not available,please execute
a clustering method','Violation','error');
           else
                PlotData                                                =
figure('Name','Dendrogram','NumberTitle','off'); %Create a new figure
for my plot
                links=handles.clusterlink;

dendrogram(links,0,'colorthreshold',str2num(get(handles.Threshold,'St
ring')))
           end
```

Εκτελείται για την επιλογή έξι του component Choices και ασχολείται με την παραγωγή δενδρογράμματος. Εδώ η συνθήκη if κάνει τον έλεγχο για το αν η μεταβλητή handles.methodone έχει τιμή 1, δηλαδή όταν έχει εκτελεστεί κάποια μέθοδο συσταδοποίησης, σε περίπτωση που είναι μηδέν εμφανίζεται αντίστοιχο μήνυμα ειδάλλως εκτελείται η εντολή dendrogram. Η εντολή παίρνει ως είσοδο τους δεσμούς που δημιουργεί η εντολή linkage και κατασκευάζει το δενδρόγραμμα, όπου το Pan είναι πόσα φύλλα του δενδρογράμματος θέλουμε να κατασκευαστούν σε περίπτωση που θέλουμε ένα συγκεκριμένο αριθμό, με τιμή μηδέν παίρνουμε το πλήρες δενδρόγραμμα. Το colorthreshold για το οποίο παίρνουμε είσοδο από το edit box Threshold χρωματίζει κάθε ομάδα του δενδρογράμματος με ένα συγκεκριμένο χρώμα όπου η απόσταση είναι μικρότερη της τιμής του colorthreshold. Η τιμή του πρέπει να είναι μέσα στα πλαίσια της τρίτης στήλης του πίνακα που παράγει η linkage και έχει τις αποστάσεις των σημείων που επιλέγονται κάθε φορά.

4.7 Εκτέλεση γραφικής παράστασης δεδομένων συσταδοποίησης

```
elseif (get(handles.Choices,'Value')==7)
         if(handles.clusterdone==1)
```

```
            if(str2num(get(handles.Dimensionality,'String'))==2)
                PlotData            =            figure('Name','Plot
Window','NumberTitle','off'); %Create a new figure for my plot

scatter(handles.DBdata(:,1),handles.DBdata(:,2),30,handles.Tcluster,'
filled')
                title('Data                                    Points',
'FontWeight','Bold','FontSize',12)
                xlabel('X dimension','FontWeight','Bold') %label the
xaxis
                ylabel('Y dimension','FontWeight','Bold') %label the
yaxis
          %The following lines of code work for a 3 dimensional array
            elseif (str2num(get(handles.Dimensionality,'String'))==3)
                PlotData            =            figure('Name','Plot
Window','NumberTitle','off'); %Create a new figure for my plot

scatter3(handles.DBdata(:,1),handles.DBdata(:,2),handles.DBdata(:,3),
30,handles.Tcluster,'filled')
                title('Data                                    Points',
'FontWeight','Bold','FontSize',12)
                xlabel('X dimension','FontWeight','Bold') %label the
xaxis
                ylabel('Y dimension','FontWeight','Bold') %label the
yaxis
                zlabel('Z dimension','FontWeight','Bold') %label the
zaxis
            elseif(str2num(get(handles.Dimensionality,'String'))==1)
                %We reversed the dimensions to display our data in Y
dimension.
                PlotData            =            figure('Name','Plot
Window','NumberTitle','off'); %Create a new figure for my plot
                points= size(handles.DBdata);

scatter(1:points(1,1),handles.DBdata(:,1),30,handles.Tcluster,'filled
')
                title('Data                                    Points',
'FontWeight','Bold','FontSize',12)
                xlabel('Segments','FontWeight','Bold')   %label   the
xaxis
                ylabel('Data','FontWeight','Bold') %label the yaxis
```

Ο κώδικας εκτελείται για την επιλογή εφτά του μενού Choices, Plot Clustered Data. Εδώ πάλι εκτελούμε γραφικές παραστάσεις με την βοήθεια της συνάρτησης scatter και scatter3 ανάλογα με τις διαστάσεις των δεδομένων μας όπως και πριν με την διαφορά ότι δεν χρησιμοποιούμε την παράμετρο ‘k’ για να ορίσουμε όλα τα σημεία με μαύρο χρώμα αλλά τον πίνακα που περιέχει τις συστάδες στην θέση της παραμέτρου ‘k’ με αποτέλεσμα να παίρνουμε ξεχωριστό χρώμα για κάθε συστάδα.

4.8 Αποθήκευση συστάδων στη βάση δεδομένων

```
elseif (get(handles.Choices,'Value')==8)
```

```
          if(handles.clusterdone==1)
  %We will use the input of the database loader to connect to DB
  %and save our Data
            Data= handles.DBSdata;

            %Data=original;
            CData= handles.Tcluster;
            Maxclusters= max(CData);
            Savesize= size(CData);
            Stringdata= [];
            counter=1;
            tbl=handles.SaveTB;
            colnames={handles.SaveCOS , handles.SaveCOSb};
            for i=1:Maxclusters
                Positions= find(CData==i);
                Possize= size(Positions);
                    for j=1:Possize(1,1)
                        for k=1:Savesize
                            if(k==Positions(j,1))
                                Stringdata{counter}=Data{k,1};
                                counter=counter+1;
                            end
                        end
                    end
            %Stringdata{counter}='End_Cluster';
            %exdata{i}= Stringdata;
            %msgbox('Operation Successful','Database export','help')
            Dsize=size(Stringdata);
            Stringdata= Stringdata' ;
            for c=1:Dsize(1,2)
                Stringdata{c,2}=i;
            end
            exdata{i}= Stringdata;
            %fastinsert(conn, tbl, colnames, Stringdata);
            clear Stringdata;
            counter=1;
            end
            rxdata= exdata' ;
            Rxside=size(rxdata);
            for i=1:Rxside(1,1)
                fastinsert(handles.conn, tbl, colnames, rxdata{i,:});
            end
            msgbox('Operation Successful','Database export','help');
          else
            msgbox('Cluster data not available,please execute a
clustering method','Violation','error');
          end
    end
end
guidata(hObject,handles);
```

Η όγδοη και τελευταία επιλογή που ονομάζεται Save Clustered Data to Database του μενού Choices, η οποία αποθηκεύει τα συσταδοποιημένα δεδομένα στη βάση. Ο κώδικας δουλεύει σχεδόν με τον ίδιο τρόπο για την πέμπτη, επιλογή η οποία και ονομάζεται Clusters με κάποιες βασικές διαφορές. Πρώτα, στην Stringdata γράφεται δίπλα σε κάθε στοιχείο και ο αριθμός της συστάδας στην οποία ανήκει με μία επαναληπτική δομή for και για το μέγεθος της Stringdata, το i έχει όπως και πριν την

τιμή της εκάστοτε συστάδας που μελετάμε. Εδώ τα ονόματα των μεταβλητών αποθηκεύονται πλέον στην exdata και της οποίας τα στοιχεία αναστρέφονται και έπειτα αποθηκεύονται στην βάση με την εντολή fastinsert. Η οποία εντολή παίρνει ως είσοδο μία μεταβλητή σύνδεσης στη βάση που στην περίπτωσή μας είναι η handles.conn , το όνομα του πίνακα της βάσης που θα περαστεί η εγγραφή, τα ονόματα των στηλών και φυσικά τα δεδομένα που θα περαστούν, στην προκειμένη περίπτωση αυτά που περιέχονται στην rxdata. Με την επαναληπτική δομή και τη βοήθεια της fastinsert περνάμε στη βάση δεδομένων όλα τα στοιχεία που περιέχονται στην rxdata, το όνομα του στοιχείου και δίπλα τον αριθμό της εκάστοτε συστάδας. Αξίζει να σημειωθεί η εντολή guidata(hObject,handles) η οποία σώζει τις μεταβολές στη δομή των handles του εκάστοτε component που επεξεργαζόμαστε.

4.9 Εκτέλεση μεθόδου συσταδοποίησης

```
%This variable will hold our data
Data= handles.DBdata;
    if(strcmp(handles.CDist,'minkowski'))
        distances=
pdist(Data,handles.CDist,str2num(get(handles.Exponent,'String')));
    elseif(strcmp(handles.CDist,'Myeuclidean'))
        distances= CluD(Data);
    else
        distances= pdist(Data,handles.CDist);
    end

if (get(handles.Clustering,'Value')==1)
    links= linkage(distances,'single');
    %This variable holds the links for display
    handles.clusterlink=links;
    handles.methodone=1;
elseif (get(handles.Clustering,'Value')==2)
    links= linkage(distances,'complete');
    handles.clusterlink=links;
    handles.methodone=1;
elseif (get(handles.Clustering,'Value')==3)
    links= linkage(distances,'average');
    handles.clusterlink=links;
    handles.methodone=1;
elseif (get(handles.Clustering,'Value')==4)
    links= linkage(distances,'centroid');
    handles.clusterlink=links;
    handles.methodone=1;
elseif (get(handles.Clustering,'Value')==5)
    links= linkage(distances,'median');
    handles.clusterlink=links;
    handles.methodone=1;
elseif (get(handles.Clustering,'Value')==6)
    links= linkage(distances,'ward');
    handles.clusterlink=links;
    handles.methodone=1;
elseif (get(handles.Clustering,'Value')==7)
    links= linkage(distances,'weighted');
    handles.clusterlink=links;
    handles.methodone=1;
```

```
end

    if          (isempty(get(handles.Parameter2,'String'))          &&
~isempty(get(handles.Parameter1,'String')))
        if (str2num(get(handles.Parameter1,'String'))<1)
            msgbox('The maximum number of clusters must be at least
1','Variable Violation','error');
        else
             if (get(handles.Clustering,'Value')==8)

handles.Tcluster=Hclu(distances,'single','maxclust',str2num(get(handl
es.Parameter1,'String')));
                handles.clusterdone=1;
             elseif (get(handles.Clustering,'Value')==9)

handles.Tcluster=Hclu(distances,'complete','maxclust',str2num(get(han
dles.Parameter1,'String')));
                handles.clusterdone=1;
             else
                Tcluster                                                =
cluster(links,'maxclust',str2num(get(handles.Parameter1,'String')));
                handles.Tcluster=Tcluster;
                handles.clusterdone=1;
             end
        end
   elseif        (isempty(get(handles.Parameter1,'String'))        &&
~isempty(get(handles.Parameter2,'String')))
              if (get(handles.Clustering,'Value')==8)

handles.Tcluster=Hclu(distances,'single','distance',str2num(get(handl
es.Parameter2,'String')));
                handles.clusterdone=1;
              elseif (get(handles.Clustering,'Value')==9)

handles.Tcluster=Hclu(distances,'complete','distance',str2num(get(han
dles.Parameter1,'String')));
                handles.clusterdone=1;
              else
                Tcluster                                                =
cluster(links,'cutoff',str2num(get(handles.Parameter2,'String')),'cri
terion','distance');
                handles.Tcluster=Tcluster;
                handles.clusterdone=1;
              end
        else
         msgbox('Clustering not performed.Check if all the fields are
empty or both maximum clusters and cutoff have values','Variable
Violation','error');
   end
```

Το button component Execute, είναι υπεύθυνο για την εκτέλεση των διαφόρων μεθόδων συσταδοποίησης. Αρχικά με την βοήθεια της συνάρτησης strcmp ελέγχουμε το μέτρο απόστασης που έχει επιλέξει ο χρήστης και ανάλογα εκτελούμε την συνάρτηση pdist με την κατάλληλη μορφή ή καλούμε την συνάρτηση CluD δίνοντας τα αποτελέσματα στην μεταβλητή distances. Με την βοήθεια μίας συνθήκης if ελέγχουμε την επιλογή του χρήστη μέσου του πεδίου Value του μενού Clustering.Ας δούμε το κομμάτι κώδικα όταν το πεδίο Value του pop-up menu

Clustering είναι 1, διότι και οι υπόλοιπες περιπτώσεις είναι σχεδόν ίδιες με τη διαφορά ότι κάθε φορά η συνάρτηση linkage παίρνει άλλη μέθοδο ως είσοδο. Μιλάμε για την πρώτη επιλογή, η οποία ονομάζεται Single Link και εκτελεί τη μέθοδο συσταδοποίησης απλού δεσμού, εκτελούνται τα παρακάτω. Με την βοήθεια μίας συνθήκης if και της εντολής strcmp ελέγχουμε την τιμή της μεταβλητής handles.CDist η οποία κρατά την επιλογή του χρήστη για το ποιο μετρό θα χρησιμοποιηθεί (Euclidean,Minkowski κτλ) Η μεταβλητή distances κρατά τις αποστάσεις που υπολογίζονται από την pdist. Έπειτα βάση των αποστάσεων η linkage εκτελεί την συσταδοποίηση απλού δεσμού και δίνει τα αποτελέσματα στην μεταβλητή links. Τα αποτελέσματα τα οποία δίνει οι linkage έχουν την ακόλουθη μορφή:

3	6	0.10198
2	7	0.143178
5	8	0.143178
4	9	0.158114
1	10	0.21587

Εικόνα 18. Παράδειγμα Αποτελεσμάτων Από Την Linkage

Η εντολή linkage αναθέτει σε κάθε στοιχείο ένα αριθμό/ταυτότητα στην αρχή και μετά αναθέτει πέρα αυτών για κάθε νέα συστάδα που δημιουργείται. Δηλαδή εάν έχουμε συνολικά 6 στοιχεία προς συσταδοποίηση η συνάρτηση αναθέτει σε κάθε ένα από αυτά και ένα αριθμό, για παράδειγμα το πρώτο στοιχείο είναι το ένα, το δεύτερο το δύο και πάει λέγοντας, επομένως όπως βλέπουμε στην παραπάνω εικόνα το 3 και 6 αναφέρεται στα στοιχεία μας 3 και 6 με απόσταση μεταξύ τους 0,10198. Η νέα αυτή συστάδα θα ονομαστεί 7 αφού τα στοιχεία μας είναι για την περίπτωση που πήραμε ως παράδειγμα 6, άρα η συστάδα 7 περιέχει το 3 και το 6, και βλέπουμε μετά ότι το 7 με το 2 έχουν απόσταση 0,143178. Δηλαδή η συστάδα που έγινε πριν και το στοιχείο 2 που τώρα η νέα αυτή συστάδα θα ονομαστεί 8. Όπως μπορούμε να ανακαλέσουμε γενικά στους αλγόριθμους συσταδοποίησης κάθε στοιχείο αρχικά αποτελεί μία συστάδα και αυτός πιθανόν είναι και ο λόγος που όλα τα στοιχεία αποτελούν ξεχωριστά σύνολα στην αρχή βασισμένα σε ένα αριθμό. Στην συνέχεια του κώδικα εκτελούνται κάποιοι έλεγχοι για την είσοδο που έδωσε ο χρήστης και ανάλογα εκτελεί την ανάλογη εντολή. Στην πρώτη συνθήκη if ελέγχουμε με την βοήθεια της isempty και των λογικών «και» εάν το edit box Parameter2 είναι άδειο ενώ το Parameter1 περιέχει κάτι. Με τη δεύτερη συνθήκη, εάν η είσοδος του χρήστη είναι μικρότερη του 1 εμφανίζεται μήνυμα λάθους και αυτό διότι το editbox με ετικέτα Parameter1 κρατά το κριτήριο maxclust, δηλαδή ο χρήστης δηλώνει πόσες συστάδες θέλει να γίνουν συνολικά και φυσικά πρέπει να γίνει τουλάχιστον μία. Σημειώνεται εδώ πως το σύμβολο ~ αντιστρέφει το αποτέλεσμα της isempty, δηλαδή ~isempty σημαίνει «δεν είναι άδειο» και το σύμβολο είναι ουσιαστικά το λογικό «όχι». Η κύρια εντολή στην περίπτωσή μας που εκτελείται με διάφορες παραμέτρους ανάλογα με την είσοδο που θα δώσει ο χρήστης στα διάφορα edit box και η οποία είσοδο του χρήστη ελέγχεται με μία

σειρά από συνθήκες if,είναι η cluster η οποία είναι υπεύθυνη για τον διαχωρισμό των συστάδων με βάση τον μέγιστο αριθμό συστάδων που θέλουμε και θα καθορίσουμε εξαρχής και το κριτήριο της απόστασης. Καλείται ανάλογα και η συνάρτηση Hclu η οποία εκτελεί τις μεθόδους συσταδοποίησης απλού και πλήρους δεσμού.

4.10 Μεταφορά δεδομένων σε άλλη φόρμα

Έχουμε το button component με ετικέτα Transfer. Η συνάρτηση callback, περιέχει τον εξής κώδικα :

```
handles.methodone=0;
handles.clusterdone=0;
handles.load=0;

DBLoaderFigureHandle  = Databaseloader; %stores the figure handle of
loader GUI here

%stores the GUI data from loader GUI here
%now we can access any of the data from the loader GUI.
LData = guidata(DBLoaderFigureHandle);
databaseload= LData.Lload;
handles.DBdata= LData.LBDataN;
handles.DBSdata= LData.LBData;
handles.SaveTB= LData.LBSave;
handles.SaveCOS= LData.LCOSave;
handles.SaveCOSb= LData.LCOSsaveb
%msgbox('Operation Successful','Database export','help');
handles.load= databaseload;
handles.conn= LData.conn;
sizeinfo= size(handles.DBdata);
set(handles.Datapoints,'String',num2str(sizeinfo(1,1)));
set(handles.Dimensionality,'String',num2str(sizeinfo(1,2)));guidata(h
Object,handles);
```

Όλες οι μεταβλητές που ορίζουμε εντός κάποιας συνάρτησης ισχύουν μόνο εντός της συνάρτησης και δεν υφίστανται σε άλλες. Αυτό πρακτικά σημαίνει επίσης ότι μπορούμε να έχουμε μία μεταβλητή με όνομα temp σε μία συνάρτηση Callback ενός button και να έχουμε επίσης μία μεταβλητή με όνομα temp σε μία συνάρτηση Callback ενός άλλου button. Οι δύο αυτές μεταβλητές θεωρούνται διαφορετικές, όταν θα αλλάξει η τιμή της μίας temp δε θα επηρεαστεί η άλλη, αλλά επίσης μετά την εκτέλεση της συνάρτησης οι τιμές που πήρανε οι μεταβλητές αυτές δεν αποθηκεύονται. Αυτή η τοπικότητα των μεταβλητών προκαλεί πρόβλημα σε περίπτωση που θέλουμε να περάσουμε κάποια αποτελέσματα από μία συνάρτηση σε μία άλλη. Μπορούμε λοιπόν να ορίσουμε μεταβλητές οι οποίες θα μπορούν να επεξεργαστούν και να αναγνωριστούν από όλες τις συναρτήσεις. Αυτό μπορεί να γίνει ως εξής. Η δομή handles κρατά τα δεδομένα των διαφόρων components. Αν δηλώσουμε μία μεταβλητή στη δομή αυτή τότε μπορεί να αναγνωριστεί από όλες

τις συναρτήσεις. Δηλαδή μία μεταβλητή handles.temp που ορίζεται εντός μίας συνάρτησης, μπορεί να χρησιμοποιηθεί και σε κάποια άλλη. Εδώ τώρα δεν μπορούμε να έχουμε πλέον δύο τέτοιες μεταβλητές με το ίδιο όνομα. Με το πέρας της εκτέλεσης της συνάρτησης επίσης οι μεταβλητές που ανήκουν στη δομή handles δεν χάνουν τις τιμές που πήραν. Ας δούμε τώρα στον παρών κώδικα τις μεταβλητές αυτές. Έχουμε ορίσει στην αρχή τρεις μεταβλητές handles.load,handles.clusterdone,handles.methodone οι οποίες μας εξυπηρετούν για διάφορους ελέγχους που κάνουμε σε άλλα σημεία του κώδικα. Συγκεκριμένα η handles.load βοηθά να καταλάβουμε εάν υπάρχει κάποια βάση φορτωμένη ή όχι, η handles.methodone εάν έχει εκτελεστή κάποια μέθοδος συσταδοποίησης και τέλος η handles.clusterdone εάν έχει υλοποιηθεί και ο διαχωρισμός των δεδομένων σε συστάδες. Παρακάτω υπάρχει μία μεταβλητή στην οποία αναθέτουμε τη νέα φόρμα που θα μελετήσουμε παρακάτω, την εφαρμογή που συνεργάζεται με τον Cluster Analyser για το φόρτωμα των βάσεων δεδομένων.
Η εντολή DBLoaderFigureHandle = Databaseloader; Αναθέτει τη δομή της νέας φόρμας Databaseloader στην μεταβλητή DBLoaderFigureHandle, έπειτα με την συνάρτηση guidata(DBLoaderFigureHandle) παίρνουμε όλη τη δομή των handles της φόρμας Databaseloader και πλέον μπορούμε να επεξεργαστούμε στοιχεία που προέρχονται από αυτήν. Παρακάτω αναθέτουμε μεταβλητές που ανήκουν στην δομή handles της άλλης φόρμας σε μεταβλητές που ανήκουν στην φόρμα του Cluster Analyser. Με βάση τα δεδομένα που πήραμε από την άλλη φόρμα του Databaseloader, η μεταβλητή handles.DBdata κρατά τα δεδομένα μας σε αριθμητική μορφή και με την βοήθεια της set παρακάτω και της size θέτουμε σε δύο components που κρατάνε τις διαστάσεις των δεδομένων μας τις στήλες της handles.DBdata που είναι ουσιαστικά και οι διαστάσεις των δεδομένων μας, δηλαδή δύο στήλες σημαίνει πως έχουμε δυσδιάστατα δεδομένα και τις γραμμές οι οποίες δείχνουν το σύνολο των δεδομένων. Αξίζει επίσης να σημειωθεί ότι η συνάρτηση guidata(hObject,handles); Στο τέλος είναι καθόλα σημαντική για την αποθήκευση των μεταβλητών στην δομή handles που ορίζουμε εμείς.

4.11 Κλήση εφαρμογής Database Loader

Έχουμε το button component με ετικέτα Database. Η συνάρτηση callback, περιέχει μόνο το εξής σημαντικό κομμάτι κώδικα :

```
figure(Databaseloader); %call my other GUI
```

Εδώ καλούμε την φόρμα του Databaseloader.

4.12 Minkowski exponent

Το editbox με ετικέτα Exponent έχει σκοπό να πάρει ως είσοδο τον εκθέτη του τύπου της μεθόδου Minkowski σε περίπτωση που την επιλέξει ο χρήστης.

4.13 Επιλογή μέτρου απόστασης

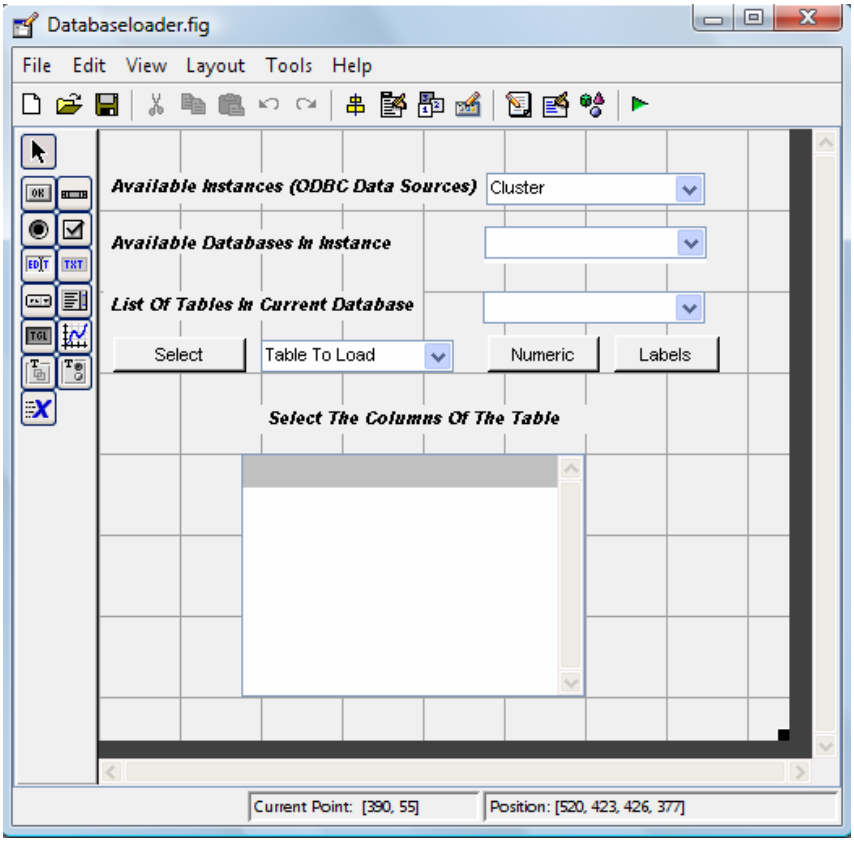
Έχουμε το button component με ετικέτα PDistance. Η συνάρτηση callback, περιέχει το εξής κομμάτι κώδικα :

```
if(get(handles.PDistance,'Value')==1)
      handles.CDist='euclidean';
elseif(get(handles.PDistance,'Value')==2)
      handles.CDist='chebychev';
elseif(get(handles.PDistance,'Value')==3)
      handles.CDist='minkowski';
elseif(get(handles.PDistance,'Value')==4)
      handles.CDist='cityblock';
elseif(get(handles.PDistance,'Value')==5)
      handles.CDist='Myeuclidean';
end
```

Ανάλογα με την επιλογή του χρήστη στο pop-up menu που βρίσκεται κάτω ακριβώς, αποθηκεύεται αυτή στην μεταβλητή handles.CDist ώστε να χρησιμοποιηθεί για την εκτέλεση των διάφορων μεθόδων συσταδοποίησης και για την παράσταση του πίνακα απόστασης. Αυτό επιτυγχάνεται με την χρήση της συνθήκης if ελέγχοντας κάθε φορά την επιλογή του χρήστη με το πεδίο Value και θέτοντας την ανάλογα στην μεταβλητή handles.CDist.

Database Loader

Η φόρμα που συνεργάζεται με τον Cluster Analyser για το φόρτωμα των διαφόρων δεδομένων προς επεξεργασία από τις βάσεις που επιθυμούμε.



Εικόνα 19. Η εμφάνιση του Database Loader

Εδώ βλέπουμε ένα αριθμό components όπως pop-up menus, buttons και static text. Ας δούμε όμως τα κύρια σημεία του κώδικα για την λειτουργία του Databaseloader.

4.14 Εμφάνιση διαθέσιμων data sources

```
setdbprefs('TempDirForRegistryOutput','c:\temp')
myODBCdir = getenv('WINDIR');
sources= getdatasources;
set(hObject,'String',sources);
guidata(hObject,handles);
```

Ο κώδικας είναι μέρος της συνάρτησης CreateFcn και εκτελείται με την δημιουργία του component με ετικέτα Showinstance.Με την εντολή setdbprefs και συγκεκριμένα το πεδίο TempDirForRegistryOutput θέτουμε τη διαδρομή στην οποία θα γραφούν προσωρινά τα δεδομένα του ODBC με την εκτέλεση της εντολής getdatasources. Η εντολή getenv ελέγχει τη λίστα του συστήματος και επιστρέφει την τιμή της μεταβλητής ανάλογα με το τί θέτουμε ως είσοδο. Εδώ η είσοδος είναι WINDIR και έτσι παίρνουμε τον κατάλογο των windows.Η getdatasources τώρα θέτει στην μεταβλητή sources τα ονόματα των ODBC και JDBC data sources που υπάρχουν διαθέσιμα και η οποία περιέχει τα ονόματα των data sources σε μορφή αλφαριθμητικών κελιών. Σε αυτό το σημείο να σημειωθεί ότι το hObject που παίρνει η συνάρτηση set ως είσοδο αναφέρεται στο ίδιο το component και με την

εντολή set θέτουμε τις τιμές της μεταβλητής sources στο πεδίο String με αποτέλεσμα να εμφανιστούν.

4.15 Αποθήκευση επιλογής χρήστη και εμφάνιση δεδομένων

Ο παρακάτω κώδικας εκτελείται με το πάτημα στο component με ετικέτα Showinstances.

Σε μία μεταβλητή list1 αποθηκεύουμε τα περιεχόμενα του πεδίου String του pop-up menu Showinstances. Έπειτα με τη βοήθεια της συνάρτησης get θέτουμε σε μία μεταβλητή την τιμή του πεδίου Value που έχει τιμή ανάλογη με την επιλογή του χρήστη. Παίρνοντας το στοιχείο τώρα με τιμή sel_inno ουσιαστικά παίρνουμε από τη μεταβλητή list1 την επιλογή του χρήστη και την αποθηκεύουμε σε μία μεταβλητή στην δομή handles ώστε να είναι προσπελάσιμη από όλες τις συναρτήσεις. Εκτελούμε τη σύνδεση στο data source της επιλογής του χρήστη με την συνάρτηση database και δίνοντας το όνομα του data source μέσω της μεταβλητής hadles.sel_in που έχει την επιλογή του χρήστη και αφήνοντας τα πεδία password και username κενά. Με την συνάρτηση dmd και με είσοδο τη μεταβλητή handles.conn που κρατάει τα περιεχόμενα της σύνδεσης με το data source τα metadata περνούν στην μεταβλητή metadata. Έπειτα, με την βοήθεια της get παίρνουμε πληροφορία για το ποιά ακριβώς στοιχεία έχουμε πληροφορία στην μεταβλητή metadata θέτοντας αυτά στη μεταβλητή sources.Μεταξύ των διάφορων πεδίων βρίσκεται και το πεδίο Catalog, το οποίο περιέχει σε μορφή κελιών τα ονόματα των βάσεων δεδομένων που είναι διαθέσιμα. Θέτουμε περιεχόμενα αυτού του πεδίου στο πεδίο String του component Showdatabases ώστε να εμφανιστούν στο μενού.

```
list1=get(handles.Showinstances,'String');
sel_inno=get(handles.Showinstances,'Value');
handles.sel_in=list1(sel_inno);
handles.conn = database(handles.sel_in,'','');
metadata= dmd(handles.conn);
sources=get(metadata);
set(handles.Showdatabases,'String',sources.Catalogs);
guidata(hObject,handles);
```

4.16 Εύρεση των πινάκων από τη βάση δεδομένων

```
sources= tables(metadata,handles.sel_db,'dbo');
sourcesize= size(sources);
for i=1:sourcesize(1,1)
   if(strcmp(sources{i,2},'TABLE'));
   string{i}=sprintf('%s',sources{i});
   end
end
set(handles.Showtables,'String',string);
guidata(hObject,handles);
```

Τώρα πλέον αναφερόμαστε στο pop-up menu που βρίσκεται δίπλα στο κείμενο με ετικέτα Showtables.

Κάνουμε χρήση της συνάρτησης tables η οποία επιστρέφει τους διάφορους πίνακες και ερωτήματα που ανήκουν στη βάση δεδομένων, παίρνοντας ως είσοδο μία μεταβλητή που κρατά τα metadata που στην περίπτωσή μας είναι η metadata, το όνομα της βάσης δεδομένων που στην περίπτωσή μας είναι το περιεχόμενο της μεταβλητής handles.sel_db και αποτελεί την επιλογή του χρήστη προηγουμένως και το σχήμα dbo. Αυτά τα δεδομένα περνούν σε μία μεταβλητή με όνομα sources της οποίας παίρνουμε τις γραμμές και στήλες με την βοήθεια της συνάρτησης size.

Η μορφή με την οποία είναι αποθηκευμένα αυτά τα στοιχεία στην μεταβλητή sources είναι για παράδειγμα:

'ClusterData' 'TABLE'
'TEI' 'TABLE'
'Basic' 'VIEW'

Έτσι, συγκρατώντας το πλήθος των γραμμών, εκτελούμε μία επαναληπτική δομή for περνώντας όλα τα στοιχεία που περιέχονται στην μεταβλητή sources και ελέγχουμε ποιά από αυτά είναι πίνακες για να τα παρουσιάσουμε στο χρήστη ώστε να μπορεί να επιλέξει. Βλέπουμε παρακάτω πως με την βοήθεια πλέον μίας if και με την strcmp ελέγχουμε σε κάθε επανάληψη εάν το πεδίο είναι πίνακας, strcmp(sources{i,2},'TABLE'**)** **,**η συνάρτηση αυτή παίρνει ως είσοδο δύο αλφαριθμητικά, που στην περίπτωσή μας είναι το όνομα TABLE και το εκάστοτε περιεχόμενο της μεταβλητής sources και αν τα δύο είναι όμοια, τότε η συνθήκη if στην περίπτωσή μας γίνεται αληθής αφού η strcmp επιστρέφει τιμή 1, εάν τα δύο δεν είναι όμοια παίρνουμε την τιμή 0 και επομένως δεν ικανοποιείται η συνθήκη if. Έτσι ξεχωρίζουμε τα ονόματα των πινάκων από όλες τις άλλες διαθέσιμες πληροφορίες που έχουμε τα οποία όπως βλέπουμε αποθηκεύουμε στην μεταβλητή string για να τα θέσουμε αργότερα στο πεδίο String του pop-up menu ώστε να μπορεί να δει ο χρήστης της διαθέσιμες επιλογές.

4.17 Εμφάνιση στηλών από τον πίνακα επιλογής

Εδώ αναφερόμαστε στο pop-up menu που βρίσκεται δίπλα στο κείμενο με ετικέτα Showcolumns.

```
metadata= dmd(handles.conn);
sel_list_table=get(handles.Showtables,'String');
sel_table_no=get(handles.Showtables,'Value');
handles.sel_table=sel_list_table(sel_table_no);
sel_cols=columns(metadata,handles.sel_db,'dbo',handles.sel_table);
```

```
set(handles.Columndisplay,'String',sel_cols);
guidata(hObject,handles);
```

Στον παραπάνω κώδικα βλέπουμε ότι για αρχή περνούμε τα metadata δεδομένα που κρατά η μεταβλητή που χειρίζεται τη σύνδεση στην μεταβλητή metadata. Έπειτα παίρνουμε την επιλογή του χρήστη όσον αφορά τον πίνακα. Κάνουμε χρήση της συνάρτησης columns η οποία επιστρέφει τα ονόματα των στηλών ενός πίνακα μίας βάσης δεδομένων. Η συνάρτηση columns μπορεί να πάρει διάφορες εισόδους από το χρήστη, στην προκειμένη περίπτωση χρησιμοποιήθηκε η πλήρης μορφή με όλες τις πιθανές εισόδους. Η γενική μορφή είναι columns(dbmeta, 'cata', 'sch','tab') , δηλαδή μία μεταβλητή που να κρατά τα metadata, το όνομα της βάσης, της σχέσης και του πίνακα που θέλουμε τις στήλες. Αφού τεθούν στην μεταβλητή sources τα ονόματα των στηλών, θέτουμε τα περιεχόμενά της στο πεδίο String του listbox Columndisplay.

4.18 Επιλογή δεδομένων από τη βάση

Ο κώδικας εκτελείται με το πάτημα του κουμπιού με ετικέτα Loader. Στις μεταβλητές DB,Table και IN θέτουμε τις επιλογές του χρήστη όσον αφορά τη βάση δεδομένων, τον πίνακα και το data source. Εκτελούμε την συνάρτηση exec**.** Με την συνάρτηση αυτή εκτελούμε ένα sql ερώτημα στο instance το οποίο είμαστε συνδεδεμένοι και η οποία σύνδεση χειρίζεται από την μεταβλητή handles.conn.Το ερώτημα που εκτελούμε έχει σκοπό να αλλάξει την προεπιλεγμένη βάση που έχει οριστεί στο instance που συνδεθήκαμε ώστε να μπορούμε να πάρουμε και να αποθηκεύσουμε δεδομένα στη βάση και τον πίνακα που έχει επιλέξει ο χρήστης και τα οποία να είναι πιθανόν διαφορετικά από την προεπιλογή. Έτσι κάνουμε χρήση του sql ερωτήματος “USE databasename” το οποίο κάνει αυτή ακριβώς τη δουλειά, αλλάζει την προεπιλεγμένη βάση σε αυτή που θέτουμε. Το όνομα της στην προκειμένη περίπτωση υπάρχει στη μεταβλητή handles.sel_db, βάζουμε [ ] αριστερά και δεξιά από την μεταβλητή handles.sel_db διότι εάν το όνομα της βάσης έχει κενά το sql ερώτημά μας δεν θα εκτελεστεί σωστά με αποτέλεσμα να μην πραγματοποιηθεί η σύνδεση στην αιτούμενη βάση. Επίσης χρησιμοποιούμε ΄ ‘ αριστερά και δεξιά της μεταβλητής handles.sel_db ώστε το Matlab να την αναγνωρίσει ως μεταβλητή και όχι ως μία λέξη που ανήκει στο sql ερώτημα. Παρακάτω βάζουμε σε μία μεταβλητή τα αριθμητικά δεδομένα και τις ετικέτες που έχει επιλέξει ο χρήστης και με τη βοήθεια μίας επαναληπτικής δομής for φτιάχνουμε κατάλληλα τα περιεχόμενα της μεταβλητής str1 εκτός του τελευταίου στοιχείου όπου και δεν θέλουμε να τοποθετήσουμε το κόμμα. Αφού θέσουμε και την τελευταία επιλογή του χρήστη στη μεταβλητή str1 χωρίς το κόμμα, τοποθετούμε τα στοιχεία από κάθετα σε οριζόντια. Ανάλογα με την επιλογή του χρήστη από το pop-up menu που βρίσκεται δίπλα από το κουμπί Select παίρνουμε

τα δεδομένα από τη βάση ή θέτουμε τα στοιχεία της αποθήκευσης. Κάνουμε χρήση της συνάρτησης setdbprefs όπου μας βοηθά να αλλάξουμε τις παραμέτρους για τον τρόπο με τον οποίο θα λάβουμε τα δεδομένα από τη βάση. Εδώ θέτουμε καταρχήν τα δεδομένα μας να επιστρέφονται ως cellarray, έπειτα με την εκτέλεση της fetch και παρέχοντας τη μεταβλητή που χειρίζεται τη σύνδεσή μας στη βάση, εκτελούμε ένα sql ερώτημα select βασισμένο στις επιλογές του χρήστη και παίρνουμε στη μεταβλητή handles.LBData τα δεδομένα που ζητήθηκαν από το χρήστη. Έπειτα με χρήστη πάλι της setdbprefs θέτουμε πως θέλουμε τα δεδομένα μας επιστρέφονται σε αριθμητική μορφή και σώζουμε τα αποτελέσματα στην μεταβλητή handles.LBDataN.

```
DB= handles.sel_db;
Table= handles.sel_table{1};
IN= handles.sel_in;
exec(handles.conn,['USE [' handles.sel_db{1} ']']);
    sel_cols=handles.sdata;
    sel_cols(1,2:(size(handles.numdata,1)+1))=handles.numdata(:,:);
    str1='';
    for i=1:size(sel_cols,2)-1
        str1{i}=str2mat(sprintf('%s,',sel_cols{1,i}));
    end

str1{size(sel_cols,2)}=str2mat(sprintf('%s',sel_cols{1,size(sel_cols,
2)}));
    str1=[str1{:}];

    setdbprefs('DataReturnFormat','cellarray');
    handles.LBData=  fetch(handles.conn,['select  '  str1  '  from  ['
Table ']']);
    setdbprefs('DataReturnFormat','numeric');
    handles.LBDataN=  fetch(handles.conn,['select  '  str1  '  from  ['
Table ']']);
    handles.LBDataN= handles.LBDataN(:,2:size(handles.LBDataN,2));
    handles.Lload=1;
```

4.19 Η συνάρτηση Hclu

```
function tpc = Hclu(Distance, Method, criterion,Cnum)

switch criterion
    case 'maxclust'
% The function gets four inputs to perform the clustering methods.

% Distance : Is the square similarity matrix containing the distances
between our
% data points.In that manner that for example element (1,2) is the
distance
% between the data points (or clusters) 1 and 2. As one can
understand element
```

```
% (1,1) value is zero because the point has zero distance from
itself!
% The main diagonal of the matrix needs to be filled with a  value so
% that the algorithm doesn't find 0 as the minimum value. We use the
value
% NaN.
%
%                   |0.0 0.1 0.3 0.9|
%                   |0.1 0.0 0.5 0.6|
%                   |0.3 0.5 0.0 0.7|
%                   |0.9 0.6 0.7 0.0|
%
% So ..the final input needs to be like this :
%
%                   |NaN 0.1 0.3 0.9|
%                   |0.1 NaN 0.5 0.6|
%                   |0.3 0.5 NaN 0.7|
%                   |0.9 0.6 0.7 NaN|
% (NaN stands for the initials of "Not a Number".)
%
%  Method  :  This  variable  is  quite  self  explenatory.  Lets  the
algorithm
% decide which clustering method to execute.
%
% criterion : This variable gets the input of the user as to how many
% clusters there is to be created. If we let the algorithm execute to
the end, all
% our  points  will  merge  into  a  single  cluster,  which  may  not  be
useful for
% our purpose.It can be either the maximum number of clusters or a
value
% for distance.
%
% Cnum: Holds the input of the user regarding the chosen criterion.
%
% tpc : This is what the algorithm returns as an output.

%First, the size of the similarity matrix is also the number of our
data points.
%Since the matrix is square, there is no worry of getting frustrated
with
%number of columns or rows. So, the variable m is the number of our
data points.
m= size(Distance);
m= m(1,1);

%Assign each data point to it's own cluster. Variable cluster changes
%according the progress of clustering.
cluster=num2cell(1:m);

%The  for  loop  will  execute  m  times,where  m  is  the  number  of  our  data
points minus the number clusters we want to
%create. If our criterion is maxclust.
for i = 1:(m-Cnum)

% min(Distance)  returns  the  minimum  value  of  each  column  to  the
variable
% MinCol and indice value of the row to variable IDrow.
   [MinCoL, IDrow] = min(Distance);
```

```
% dbp : variable holds the minimum distance that the min function
returns.
% Minimum of all the columns minimums is the distance we seek.
% MinJ : holds the column value of that distance, since the previous
command returned
% something like that :
% MinCol: 0.2159    0.1432    0.1020    0.1581    0.1432 0.1020
% IDrow :   3         3         6         3         2       3
   [dbp, MinJ] = min(MinCoL);
%Find the row of the value by giving input the column.
   MinI = IDrow(MinJ);

%Since we notice that we are at the lower diagonal, we prefer however
to
%work in the upper diagonal, so we reverse the values of MinI and
MinJ.
    if MinI > MinJ
      temp=MinI;
      MinI=MinJ;
      MinJ=temp;
    end

   %For method output delete the comment symbol of the following line
   %sprintf('The  distance  of  %d  and  %d  is    %f  ,  iteration
%d',MinI,MinJ,dbp,i)
   %Merge cluster j into cluster i, then delete j
   cluster{MinI} = [cluster{MinI} cluster{MinJ}];
   cluster(MinJ) = [];
   %For method output delete the comment symbol of the following line
   %sprintf('Merge cluster %d into %d',MinJ,MinI)

   %The new distance matrix is calculated depending on the input of
the
   %user. Min function returns the minimum value of each row from the
specified columns of the two
   %matrixes, for example.
   %
   %                    Return of minimum
   % 0.2354   0.2159        0.2159
   % 0.2435   0.1432        0.1432
   % 0.1020      NaN        0.1020   (<- notice how NaN is ignored in
the calculations)
   % 0.2195   0.1581        0.1581
   % 0.3860   0.2846        0.2846
   %    NaN   0.1020        0.1020
   % We update the columns and rows with indice of MinI because we
are
   % going to discard the MinJ ones, since we deleted the j cluster
and
   % merged it into i. We place our new distances in the place of i.
   switch Method
   case 'single'
      Distance(:, MinI) = min(Distance(:, MinI), Distance(:, MinJ));
      Distance(MinI, :) = min(Distance(MinI, :), Distance(MinJ, :));
   case 'complete'
      Distance(:, MinI) = max(Distance(:, MinI), Distance(:, MinJ));
      Distance(MinI, :) = max(Distance(MinI, :), Distance(MinJ, :));
   end

   %Deletition of the cluster j columns and place NaN.
```

```
   Distance(MinJ, :) = [];
   Distance(:, MinJ) = [];
   Distance(MinI, MinI) = NaN;

end
%Rearrange the results of clustering into a new variable tpc, which
will
%hold the results in a similar way as the output that function
cluster of
%Matlab gives.
temp=1;
for i=1:size(cluster,2)
    for j=1:size(cluster{i},2)
        Mycluster(temp)=cluster{i}(j);
        ClusterIndex(temp)=i;
        temp=temp+1;
    end
end
Cindex(:,1)=Mycluster';
Cindex(:,2)=ClusterIndex';
for i=1:size(Cindex)
    temp=find(Cindex(:,1)==i);
    tpc(i)=Cindex(temp,2);
end
tpc=tpc';
%For the distance criterion, all that changes is the condition of the
loop.
%The loop terminates anyway at m iterations.
case 'distance'
   dbp=0;
   m= size(Distance);
   m= m(1,1);
   counter=1;
   cluster=num2cell(1:m);
   while((dbp < Cnum) && counter<m)
       [MinCoL, IDrow] = min(Distance);
       [dbp, MinJ] = min(MinCoL);
       MinI = IDrow(MinJ);

       if MinI > MinJ
        temp=MinI;
        MinI=MinJ;
        MinJ=temp;
       end
       %For method output delete the comment symbol of the following
line
       %sprintf('The distance of %d and %d is   %f , iteration
%d',MinI,MinJ,dbp,counter)
       cluster{MinI} = [cluster{MinI} cluster{MinJ}];
       cluster(MinJ) = [];
        %For method output delete the comment symbol of the following
line
        %sprintf('Merge cluster %d into %d',MinJ,MinI)
    switch Method
      case 'single'
        Distance(:,  MinI)  =  min(Distance(:,  MinI),  Distance(:,
MinJ));
        Distance(MinI,  :)  =  min(Distance(MinI,  :),  Distance(MinJ,
:));
      case 'complete'
```

```
        Distance(:, MinI) = max(Distance(:, MinI), Distance(:,
MinJ));
        Distance(MinI, :) = max(Distance(MinI, :), Distance(MinJ,
:));
    end

    Distance(MinJ, :) = [];
    Distance(:, MinJ) = [];
    Distance(MinI, MinI) = NaN;
    counter=counter+1;
    %Check the next distance so the loop can terminate effectively if
it
    %exceeds the input of the user.
    [MinCoL, IDrow] = min(Distance);
    [dbp, MinJ] = min(MinCoL);
   end
temp=1;
for i=1:size(cluster,2)
    for j=1:size(cluster{i},2)
        Mycluster(temp)=cluster{i}(j);
        ClusterIndex(temp)=i;
        temp=temp+1;
    end
end
Cindex(:,1)=Mycluster';
Cindex(:,2)=ClusterIndex';
for i=1:size(Cindex)
    temp=find(Cindex(:,1)==i);
    tpc(i)=Cindex(temp,2);
end
tpc=tpc';
end
```

Αυτή η συνάρτηση εκτελεί συσταδοποίηση απλού και πλήρους δεσμού βασισμένη στον πίνακα αποστάσεως που υπολογίστηκε από την CluD.

Η συνάρτηση παίρνει ως είσοδο τις μεταβλητές Distance, Method,criterion και Cnum.

Distance : Είναι ο πίνακας που παρήγαγε η συνάρτηση CluD, και περιέχει τις ανά ζεύγη αποστάσεις των σημείων στην άνω και κάτω διαγώνιο με τα στοιχεία της κύριας διαγωνίου να είναι NaN ώστε να παραβλέπονται στους υπολογισμούς. Οι αποστάσεις υπάρχουν με τέτοιο τρόπο, ώστε για παράδειγμα το στοιχείο (1,2) του πίνακα να δείχνει ουσιαστικά την απόσταση του σημείου 1 από το 2. Το στοιχείο (1,1) δείχνει την απόσταση του σημείου 1 από το σημείο 1, δηλαδή από τον εαυτό του η οποιά είναι μηδέν.

Method : Αυτή η μεταβλητή κρατά τη μέθοδο της συσταδοποίησης που θα χρησιμοποιηθεί.

criterion : Αυτή η μεταβλητή κρατά το κριτήριο παύσης του αλγορίθμου το οποίο θα είναι είτε ο μέγιστος αριθμός των συστάδων ή μία απόσταση. Εάν αφήσουμε τον αλγόριθμο να εκτελεστεί μέχρι τέλους όλα τα σημεία θα ανήκουν σε μία συστάδα, κάτι το οποίο δεν είναι χρήσιμο για την εξαγωγή συμπερασμάτων.

Cnum : Σχετίζεται με την επιλογή του κριτηρίου, ανάλογα αν η μεταβλητή criterion έχει να κάνει με τον μέγιστο αριθμό συστάδων ή την απόσταση η μεταβλητή Cnum κρατά αυτό το κριτήριο.

Αρχικά, το μέγεθος του πίνακα αποστάσεως, θα είναι ουσιαστικά και ο συνολικός αριθμός των σημείων. Επίσης αφού είναι τετραγωνικός δεν υπάρχει λόγος ανησυχίας περί γραμμών και στηλών. Στην μεταβλητή m με την βοήθεια της συνάρτησης size δίνουμε το μέγεθος του πίνακα Distance, και κρατάμε ένα από τα δύο νούμερα, αφού όπως είπαμε λίγο πιο πριν, είτε στήλες ή γραμμές για αυτή την περίπτωση είναι ένα και το αυτό.

Τοποθετούμε κάθε στοιχείο σε ξεχωριστή συστάδα. Για την επαναληπτική δομή for, στην πρώτη περίπτωση όπου το κριτήριο είναι ο αριθμός των συστάδων, η δομή θα εκτελεστεί m φορές μείον τον αριθμό των συστάδων που δήλωσε ότι επιθυμεί ο χρήστης. Έτσι για παράδειγμα, αν έχουμε έξι στοιχεία συνολικά και δηλώσουμε ότι θέλουμε έξι συστάδες, τότε όπως καταλαβαίνουμε ουσιαστικά δεν θα πραγματοποιηθεί συσταδοποίηση γιατί όλα τα στοιχεία θα αποτελούν απλά μία ξεχωριστή συστάδα.

Στην συνέχεια, με την βοήθεια της συνάρτησης min,η οποία επιστρέφει την μικρότερη τιμή κάθε στήλης στην μεταβλητή MinCol και τον αριθμό του δείκτη της γραμμής στην οποία ανήκει στην μεταβλητή IDrow.

Παρακάτω η μεταβλητή dbp κρατά τη μικρότερη απόσταση που επιστρέφει η συνάρτηση min και είναι η μικρότερη απόσταση από τις μικρότερες αποστάσεις των στηλών, ουσιαστικά η απόσταση που ζητάμε. Η μεταβλητή MinJ θα κρατά τώρα τον αριθμό της στήλης.

Έπειτα, βρίσκουμε τον αριθμό της γραμμής δίνοντας ως είσοδο τον αριθμό της στήλης.

Παρατηρούμε τώρα, ότι βρισκόμαστε στην κάτω διαγώνιο, αλλά επειδή προτιμούμε να δουλεύουμε στην άνω, αντιστρέφουμε τις τιμές των MinI και MinJ.

Παρακάτω με την βοήθεια της συνάρτησης sprintf δίνουμε έξοδο για την συγκεκριμένη επανάληψη σε περίπτωση που το επιθυμούμε.

Συγχωνεύουμε τις συστάδες i και j σβήνοντας την j και δίνουμε έξοδο με την βοήθεια μίας sprintf ενημερώνοντας πια συστάδα συγχωνεύτηκε σε ποιά.

Ο νέος πίνακας αποστάσεως υπολογίζεται ανάλογα με την είσοδο του χρήστη. Η συνάρτηση min για παράδειγμα, επιστρέφει την μικρότερη τιμή κάθε γραμμής από τις συγκεκριμένες στήλες των δύο πινάκων. Ενημερώνουμε τις γραμμές και στήλες σε σχέση με το δείκτη της MinI διότι όπως είδαμε πριν συγχωνεύσαμε τις συστάδες i και j σε αυτήν της i, θέλουμε να σβήσουμε τα στοιχεία του j και θα ενημερώσουμε τον πίνακα με τις νέες αποστάσεις στις θέσεις του i. Έτσι έχουμε παρακάτω την

διαγραφή των στοιχείων του j και τοποθέτηση του NaN στην διαγώνιο ώστε να μην επηρεάζονται οι υπολογισμοί.

Στην περίπτωση που το κριτήριο μας είναι η απόσταση, το μόνο που αλλάζει είναι το είδος της επαναληπτικής δομής και η συνθήκη τερματισμού. Κάνουμε χρήση της while η οποία θα εκτελεστεί για όσα σημεία έχουμε αν το κριτήριο της απόστασης δεν είναι ικανό να παύση τον αλγόριθμο ενδιάμεσα, ή για την απόσταση που όρισε ο χρήστης.

Και για τις δύο περιπτώσεις διαμορφώνουμε έπειτα την μεταβλητή cluster ώστε να μπορεί να γίνει χρήση των αποτελεσμάτων. Δίνουμε σε μία μεταβλητή temp την τιμή 1. Έπειτα εκτελούμε μία επαναληπτική δομή του τύπου for για όσες συστάδες έχουμε συνολικά που είναι το πλήθος των στηλών της μεταβλητής cluster και το επιτυγχάνουμε χρησιμοποιώντας τη συνάρτηση size δίνοντας ως παράμετρο το 2 αφού επιθυμούμε να μας επιστραφούν οι στήλες. Στν συνέχεια έχουμε ακόμα μία επαναληπτική δομή for που εκτελείται για όσα στοιχεία περιέχει κάθε συστάδα και έχει σκοπό να μορφωποιήση τις μεταβλητές Mycluster και ClusterIndex. Με το πέρας των επαναληπτικών δομών η μία θα περιέχει τα σημεία με τη σειρά, δηλαδή πρώτα τα σημεία της πρώτης συστάδας,μετά της δεύτερης και πάει λέγοντας και η άλλη τον αριθμό της συστάδας που ανήκει κάθε στοιχείο. Η μεταβλητή temp χρησιμοποιείται ως δείκτης για την ορθή τοποθέτηση των στοιχείων. Έπειτα αναστρέφουμε τους δύο αυτούς πίνακες και τους συγχωνεύουμε σε ένα. Με μία επαναληπτική δομή for για το μέγεθος των γραμμών του Cindex και την βοήθεια της συνάρτησης find, για κάθε στοιχείο βρίσκουμε σε ποια συστάδα ανήκει ξεκινώντας από το πρώτο και καταλήγοντας στο τελευταίο και τοποθετούμε στον πίνακα tpc τον αριθμό της συστάδας που ανήκει.

4.20 Η συνάρτηση CluD

```
function Distance = CluD(Inputarray)

    %This function gets as input a matrix with numeric data and
calculates
    %the Euclidean distance and arranges the distances into a square
similarity matrix.
    %then places NaN in the main diagonal. r is the
    %number of our data points which is equal to the number of rows.

 [r,c]= size(Inputarray);

 for i= 1:r
    for j = i:r
      Distance(i,j)           =           sqrt(sum((Inputarray(i,:)-
Inputarray(j,:)).^2));
    end
 end
Distance=Distance+Distance';
```

```
 for i= 1:r
     Distance(i,i)=NaN;
 end

end
```

Η συνάρτηση παίρνει ως είσοδο ένα πίνακα με αριθμητικά δεδομένα και υπολογίζει την ευκλείδεια απόσταση. Θέτει τα αποτελέσματα σε ένα τετραγωνικό πίνακα αποστάσεως. Τοποθετεί το χαρακτηριστικό NaN στην κύρια διαγώνιο.

Θέτουμε στις μεταβλητές r και c τις διαστάσεις του πίνακα Inputarray που είναι ο πίνακας που η συνάρτηση έλαβε ως είσοδο. Από αυτές, το r κρατάει τη διάσταση των γραμμών, η οποία είναι και ο αριθμός των στοιχείων. Με τη βοήθεια δύο επαναληπτικών δομών for διαμορφώνουμε τον πίνακα Distance ο οποίος εν τέλει θα περιέχει τις αποστάσεις ανά ζεύγη των στοιχείων με κύρια διαγώνιο μηδέν και θα είναι άνω τριγωνικός. Έπειτα, διαμορφώνομε τον πίνακα Distance ώστε οι αποστάσεις των στοιχείων ανά ζεύγη να υπάρχουν και στην κάτω διαγώνιο.
Με μία επαναληπτική δομή for τοποθετούμε στην κύρια διαγώνιο το χαρακτηριστικό NaN.

Συμπεράσματα

Με την εφαρμογή που υλοποιήθηκε στα πλαίσια αυτής της πτυχιακής εργασίας, δίνεται η δυνατότητα στο χρήστη να κατανοήσει τις ιεραρχικές μεθόδους συσταδοποίησης καθώς και τα διάφορα στάδια για την υλοποίηση τους. Επίσης μπορεί να χρησιμοποιηθεί και ως εργαλείο επεξεργασίας δεδομένων. Η εφαρμογή αυτή μπορεί να προβάλει γραφικές παραστάσεις, να προβάλει τα δεδομένα που έχει επιλέξει ο χρήστης, να προβληθεί γραφικά η διαδικασία της συσταδοποίησης και ο τρόπος με τον οποίο έχουν χωριστεί τα δεδομένα σε συστάδες. Επιπλέον δίνεται η δυνατότητα αποθήκευσης των συσταδοποιημένων δεδομένων σε ένα πίνακα της προτίμησής του σε οποιαδήποτε βάση δεδομένων. Επίσης με την ανάπτυξη κώδικα για δύο από τους ιεραρχικούς αλγορίθμους συσταδοποίησης (πλήρους και απλού δεσμού), η εφαρμογή μπορεί να χρησιμοποιθεί για εκπαιδευτικούς σκοπούς στο ΤΕΙ ώστε να κατανοηθούν καλύτερα τεχνικές εξόρυξης γνώσης.

Η δοκιμή των αλγορίθμων συσταδοποίησης που αναπτύχθηκαν προγραμματιστικά και η σύγκριση τους με τις συναρτήσεις που παρέχει το Matlab επιβεβαίωσε και πρακτικά τη σωστή λειτουργία τους.

Για την εφαρμογή των αλγορίθμων επιλέχθηκαν δεδομένα (γεωγραφικά δεδομένα, οικονομικά, στατιστικά) από πηγές του διαδικτύου για να διαπιστωθεί η χρησιμότητα και ορθότητα των αλγορίθμων.

Ο Κώδικας Της Εφαρμογής

Αρχείο TEI.m

```
function varargout =
TEI(varargin)
% TEI M-file for TEI.fig
%      TEI, by itself, creates a
new TEI or raises the existing
%      singleton*.
%
%      H = TEI returns the handle
to a new TEI or the handle to
%      the existing singleton*.
%
%
TEI('CALLBACK',hObject,eventData,
handles,...) calls the local
%      function named CALLBACK in
TEI.M with the given input
arguments.
%
%
TEI('Property','Value',...)
creates a new TEI or raises the
%      existing singleton*.
Starting from the left, property
value pairs are
%      applied to the GUI before
TEI_OpeningFcn gets called.  An
%      unrecognized property name
or invalid value makes property
application
%      stop.  All inputs are
passed to TEI_OpeningFcn via
varargin.
%
%      *See GUI Options on
GUIDE's Tools menu.  Choose "GUI
allows only one
%      instance to run
(singleton)".
%
% See also: GUIDE, GUIDATA,
GUIHANDLES

% Edit the above text to modify
the response to help TEI

% Last Modified by GUIDE v2.5 22-
Apr-2009 20:52:13

% Begin initialization code - DO
NOT EDIT

Update handles structure
guidata(hObject, handles);

% UIWAIT makes TEI wait for user
response (see UIRESUME)
```

```
gui_Singleton = 1;
gui_State = struct('gui_Name',
mfilename, ...

'gui_Singleton',  gui_Singleton,
...

'gui_OpeningFcn',
@TEI_OpeningFcn, ...

'gui_OutputFcn',  @TEI_OutputFcn,
...

'gui_LayoutFcn',  [] , ...

'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback =
str2func(varargin{1});
end
if nargout
    [varargout{1:nargout}] =
gui_mainfcn(gui_State,
varargin{:});
else
    gui_mainfcn(gui_State,
varargin{:});
end
% End initialization code - DO
NOT EDIT
% --- Executes just before TEI is
made visible.
function TEI_OpeningFcn(hObject,
eventdata, handles, varargin)
% This function has no output
args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)
% varargin   command line
arguments to TEI (see VARARGIN)

% Choose default command line
output for TEI
handles.output = hObject;

        distances= CluD(Data);
    else
        distances=
pdist(Data,handles.CDist);
    end
```

```
% uiwait(handles.figure1);


% --- Outputs from this function
are returned to the command line.
function varargout =
TEI_OutputFcn(hObject, eventdata,
handles)
% varargout  cell array for
returning output args (see
VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)

% Get default command line output
from handles structure
varargout{1} = handles.output;
% --- Executes on button press in
Execute.
function
Execute_Callback(hObject,
eventdata, handles)
% hObject    handle to Execute
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)



%The following line of code
executes the according clustering
method
%depending on the selection of
the user.
%This variable will hold our data
Data= handles.DBdata;

if(strcmp(handles.CDist,'minkowsk
i'))
        distances=
pdist(Data,handles.CDist,str2num(
get(handles.Exponent,'String')));

elseif(strcmp(handles.CDist,'Myeu
clidean'))


    if
(isempty(get(handles.Parameter2,'
String')) &&
~isempty(get(handles.Parameter1,'
String')))
        if
(get(handles.Clustering,'Value')=
=1)
    links=
linkage(distances,'single');
    %This variable holds the
links for display
    handles.clusterlink=links;
    handles.methodone=1;
elseif
(get(handles.Clustering,'Value')=
=2)
    links=
linkage(distances,'complete');
    handles.clusterlink=links;
    handles.methodone=1;
elseif
(get(handles.Clustering,'Value')=
=3)
    links=
linkage(distances,'average');
    handles.clusterlink=links;
    handles.methodone=1;
elseif
(get(handles.Clustering,'Value')=
=4)
    links=
linkage(distances,'centroid');
    handles.clusterlink=links;
    handles.methodone=1;
elseif
(get(handles.Clustering,'Value')=
=5)
    links=
linkage(distances,'median');
    handles.clusterlink=links;
    handles.methodone=1;
elseif
(get(handles.Clustering,'Value')=
=6)
    links=
linkage(distances,'ward');
    handles.clusterlink=links;
    handles.methodone=1;
elseif
(get(handles.Clustering,'Value')=
=7)
    links=
linkage(distances,'weighted');
    handles.clusterlink=links;
    handles.methodone=1;

end

handles.Tcluster=Hclu(distances,'
complete','distance',str2num(get(
handles.Parameter2,'String')));

handles.clusterdone=1;
```

```
(str2num(get(handles.Parameter1,'String'))<1)
            msgbox('The maximum number of clusters must be at least 1','Variable Violation','error');
        else
            if (get(handles.Clustering,'Value')==8)

handles.Tcluster=Hclu(distances,'single','maxclust',str2num(get(handles.Parameter1,'String')));

handles.clusterdone=1;
            elseif (get(handles.Clustering,'Value')==9)

handles.Tcluster=Hclu(distances,'complete','maxclust',str2num(get(handles.Parameter1,'String')));

handles.clusterdone=1;
            else
                Tcluster = cluster(links,'maxclust',str2num(get(handles.Parameter1,'String')));

handles.Tcluster=Tcluster;

handles.clusterdone=1;
            end
        end
   elseif (isempty(get(handles.Parameter1,'String')) && ~isempty(get(handles.Parameter2,'String')))
            if (get(handles.Clustering,'Value')==8)

handles.Tcluster=Hclu(distances,'single','distance',str2num(get(handles.Parameter2,'String')));

handles.clusterdone=1;
            elseif (get(handles.Clustering,'Value')==9)


% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
```

```
            else
                Tcluster = cluster(links,'cutoff',str2num(get(handles.Parameter2,'String')),'criterion','distance');

handles.Tcluster=Tcluster;

handles.clusterdone=1;
            end
        else
         msgbox('Clustering not performed.Check if all the fields are empty or both maximum clusters and cutoff have values','Variable Violation','error');
   end

guidata(hObject,handles);

function Temporary_Callback(hObject, eventdata, handles)
% hObject    handle to Temporary (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of Temporary as text
%        str2double(get(hObject,'String')) returns contents of Temporary as a double
% --- Executes during object creation, after setting all properties.
function Temporary_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Temporary (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB



%databaseload= LData.Lload;
%handles.DBdata= LData.LBDataN;
%handles.DBSdata= LData.LBData;
%handles.load= databaseload;
%sizeinfo= size(handles.DBdata);
%We remove the first column so
```

```
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))

set(hObject,'BackgroundColor','wh
ite');
end
% --- Executes on button press in
Database.
function
Database_Callback(hObject,
eventdata, handles)
% hObject    handle to Database
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)
%This variable is to verify if a
database is loaded or no.Most of
my global
%variables are located here.
%databaseload= 0;
handles.methodone=0;
handles.clusterdone=0;
figure(Databaseloader); %call my
other GUI
%conn= database('Cluster','','');
%setdbprefs('DataReturnFormat','n
umeric');
%Data= fetch(conn,'select * from
TEI');
%sizeinfo= size(Data);
%This line of code loads all
lines of the data aside from the
first one.
%Useful of multidimensional data.
%handles.DBdata=Data(:,2:min(size
info));
%DBLoaderFigureHandle  =
Databaseloader; %stores the
figure handle of loader GUI here
%stores the GUI data from loader
GUI here
%now we can access any of the
data from the loader GUI.
%LData =
guidata(DBLoaderFigureHandle);


% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)
```

```
the dimensions will display
correctly.
%sizeinfo(1,2)=sizeinfo(1,2)-1;
%Data= num2str(Data);
%set(handles.Temporary,'String',D
ata);
%set(handles.Datapoints,'String',
num2str(sizeinfo(1,1)));
%set(handles.Dimensionality,'Strin
g',num2str(sizeinfo(1,2)));
guidata(hObject,handles);
% --- Executes during object
creation, after setting all
properties.
function
Database_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to Database
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called
% --- Executes on button press in
Completelink.
function
Completelink_Callback(hObject,
eventdata, handles)
% hObject    handle to
Completelink (see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)
% Hint: get(hObject,'Value')
returns toggle state of
Completelink



% --- Executes on slider
movement.
function
slider1_Callback(hObject,
eventdata, handles)
% hObject    handle to slider1
(see GCBO)




% --- Executes during object
creation, after setting all
properties.
function
Parameter1_CreateFcn(hObject,
eventdata, handles)
```

```
% Hints: get(hObject,'Value')
returns position of slider
%        get(hObject,'Min') and
get(hObject,'Max') to determine
range of slider
% --- Executes during object
creation, after setting all
properties.
function
slider1_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to slider1
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called

% Hint: slider controls usually
have a light gray background.
if
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))

set(hObject,'BackgroundColor',[.9
.9 .9]);
end
function
Parameter1_Callback(hObject,
eventdata, handles)
% hObject    handle to Parameter1
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)

% Hints: get(hObject,'String')
returns contents of Parameter1 as
text
%
str2double(get(hObject,'String'))
returns contents of Parameter1 as
a double



% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called
```

```
% hObject    handle to Parameter1
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called

% Hint: edit controls usually
have a white background on
Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))

set(hObject,'BackgroundColor','wh
ite');
end
function
Parameter2_Callback(hObject,
eventdata, handles)
% hObject    handle to Parameter2
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)

% Hints: get(hObject,'String')
returns contents of Parameter2 as
text
%
str2double(get(hObject,'String'))
returns contents of Parameter2 as
a double
% --- Executes during object
creation, after setting all
properties.
function
Parameter2_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to Parameter2
(see GCBO)




if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))
```

```
% Hint: edit controls usually
have a white background on
Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))

set(hObject,'BackgroundColor','wh
ite');
end
function
Parameter3_Callback(hObject,
eventdata, handles)
% hObject    handle to Parameter3
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)

% Hints: get(hObject,'String')
returns contents of Parameter3 as
text
%
str2double(get(hObject,'String'))
returns contents of Parameter3 as
a double

% --- Executes during object
creation, after setting all
properties.
function
Parameter3_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to Parameter3
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called
% Hint: edit controls usually
have a white background on
Windows.
%       See ISPC and COMPUTER.


% --- Executes on button press in
Analyse.
function
Analyse_Callback(hObject,
eventdata, handles)
% hObject    handle to Analyse
```

```
end
% --- Executes on selection
change in listbox2.
function
listbox2_Callback(hObject,
eventdata, handles)
% hObject    handle to listbox2
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)

% Hints: contents =
get(hObject,'String') returns
listbox2 contents as cell array
%
contents{get(hObject,'Value')}
returns selected item from
listbox2
% --- Executes during object
creation, after setting all
properties.
function
listbox2_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to listbox2
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called
% Hint: listbox controls usually
have a white background on
Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))

set(hObject,'BackgroundColor','wh
ite');
end



        end
        string=string' ;

set(handles.Temporary,'String',st
rcat(string,Data));
    elseif
```

```
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)
if(handles.load==1)
   if
(get(handles.Choices,'Value')==1)

if(strcmp(handles.CDist,'minkowsk
i'))
            distance=
pdist(handles.DBdata,handles.CDis
t,str2num(get(handles.Exponent,'S
tring')));
            squared=
squareform(distance);
        else

if(strcmp(handles.CDist,'Myeuclid
ean'))


distance=CluD(handles.DBdata);
                  for i=
1:size(distance)

distance(i,i)=0;
                  end

squared=distance;
            else
             distance=
pdist(handles.DBdata,handles.CDis
t);
             squared=
squareform(distance);
            end
        end
        sdistance= triu(squared);

set(handles.Temporary,'String',nu
m2str(sdistance));
    elseif
(get(handles.Choices,'Value')==3)
        Data=
num2str(handles.DBdata);
        sourcesize=
size(handles.DBSdata);
        for i=1:sourcesize(1,1)
            string{i}=sprintf('%s
',num2str(handles.DBSdata{i}));


elseif(str2num(get(handles.Dimens
ionality,'String'))==1)
            %We reversed the
dimensions to display our data in
Y dimension.
(get(handles.Choices,'Value')==2)
         %The following code
creates a new figure and assings
the handle to
         %Plotdata variable which
handles the plot data.
         %The if condition is
used to distinguish 2 dimensions
and 3.

if(str2num(get(handles.Dimensiona
lity,'String'))==2)
            PlotData =
figure('Name','Plot
Window','NumberTitle','off');
%Create a new figure for my plot

scatter(handles.DBdata(:,1),handl
es.DBdata(:,2),30,'filled','k')
            title('Data Points',
'FontWeight','Bold','FontSize',12
)
            xlabel('X
dimension','FontWeight','Bold')
%label the xaxis
            ylabel('Y
dimension','FontWeight','Bold')
%label the yaxis
            %The following lines
of code work for 3 dimensional
arrays.
         elseif
(str2num(get(handles.Dimensionali
ty,'String'))==3)
            PlotData =
figure('Name','Plot
Window','NumberTitle','off');
%Create a new figure for my plot

scatter3(handles.DBdata(:,1),hand
les.DBdata(:,2),handles.DBdata(:,
3),30,'filled','k')
            title('Data Points',
'FontWeight','Bold','FontSize',12
)
            xlabel('X
dimension','FontWeight','Bold')
%label the xaxis
            ylabel('Y
dimension','FontWeight','Bold')
%label the yaxis
            zlabel('Z
dimension','FontWeight','Bold')
%label the zaxis



if(k==Positions(j,1))

Stringdata{counter}=Data{k,1};

counter=counter+1;
```

```
            PlotData =
figure('Name','Plot
Window','NumberTitle','off');
%Create a new figure for my plot
            points=
size(handles.DBdata);

scatter(1:points(1,1),handles.DBd
ata(:,1),30,'filled','k')
            title('Data Points',
'FontWeight','Bold','FontSize',12
)

xlabel('Segments','FontWeight','B
old') %label the xaxis

ylabel('Data','FontWeight','Bold'
) %label the yaxis
         end
    elseif
(get(handles.Choices,'Value')==4)

links=handles.clusterlink;

set(handles.Temporary,'String',nu
m2str(links));
    elseif
(get(handles.Choices,'Value')==5)

%Tcluster=handles.Tcluster;
            Data=
handles.DBSdata;
            CData=
handles.Tcluster;
            Maxclusters=
max(CData);
            Savesize=
size(CData);
            Stringdata= [];
            counter=1;
            for i=1:Maxclusters
                Positions=
find(CData==i);
                Possize=
size(Positions);
                    for
j=1:Possize(1,1)
                        for
k=1:Savesize




                PlotData =
figure('Name','Plot
Window','NumberTitle','off');
%Create a new figure for my plot

scatter(handles.DBdata(:,1),handl
                                end
                            end
                        end

Dsize=size(Stringdata);

           % Stringdata=
Stringdata' ;
           string{i}=sprintf(' [
%s ] ',Stringdata{:});
            %for c=1:Dsize(1,2)
            %    {c,2}=i;
            %end
            %exdata{i}=
Stringdata;
            clear Stringdata;
            counter=1;
            end
            %string=string' ;

set(handles.Temporary,'Value',1);

set(handles.Temporary,'String',st
ring);
    elseif
(get(handles.Choices,'Value')==6)

if(handles.methodone==0)

msgbox('Dendrogram data not
available,please execute a
clustering
method','Violation','error');
            else
                PlotData =
figure('Name','Dendrogram','Numbe
rTitle','off'); %Create a new
figure for my plot

links=handles.clusterlink;

dendrogram(links,0,'colorthreshol
d',str2num(get(handles.Threshold,
'String')))
            end
     elseif
(get(handles.Choices,'Value')==7)

if(handles.clusterdone==1)

if(str2num(get(handles.Dimensiona
lity,'String'))==2)



scatter(1:points(1,1),handles.DBd
ata(:,1),30,handles.Tcluster,'fil
led')
                title('Data
Points',
```

```
es.DBdata(:,2),30,handles.Tcluste
r,'filled')
                title('Data
Points',
'FontWeight','Bold','FontSize',12
)
                xlabel('X
dimension','FontWeight','Bold')
%label the xaxis
                ylabel('Y
dimension','FontWeight','Bold')
%label the yaxis
                %The following
lines of code work for a 3
dimensional array
            elseif
(str2num(get(handles.Dimensionali
ty,'String'))==3)
                PlotData =
figure('Name','Plot
Window','NumberTitle','off');
%Create a new figure for my plot

scatter3(handles.DBdata(:,1),hand
les.DBdata(:,2),handles.DBdata(:,
3),30,handles.Tcluster,'filled')
                title('Data
Points',
'FontWeight','Bold','FontSize',12
)
                xlabel('X
dimension','FontWeight','Bold')
%label the xaxis
                ylabel('Y
dimension','FontWeight','Bold')
%label the yaxis
                zlabel('Z
dimension','FontWeight','Bold')
%label the zaxis

elseif(str2num(get(handles.Dimens
ionality,'String'))==1)
                %We reversed the
dimensions to display our data in
Y dimension.
                PlotData =
figure('Name','Plot
Window','NumberTitle','off');
%Create a new figure for my plot
                points=
size(handles.DBdata);




Stringdata{counter}=Data{k,1};

counter=counter+1;
                            end
                        end
'FontWeight','Bold','FontSize',12
)

xlabel('Segments','FontWeight','B
old') %label the xaxis

ylabel('Data','FontWeight','Bold'
) %label the yaxis


            end
        else
            msgbox('Cluster data
not available,please execute a
clustering
method','Violation','error');
        end
    elseif
(get(handles.Choices,'Value')==8)

if(handles.clusterdone==1)
            %We will use the
input of the database loader to
connect to DB
            %and save our Data
            Data=
handles.DBSdata;

            CData=
handles.Tcluster;
            Maxclusters=
max(CData);
            Savesize=
size(CData);
            Stringdata= [];
            counter=1;
            tbl=handles.SaveTB;

colnames={handles.SaveCOS ,
handles.SaveCOSb};
            for i=1:Maxclusters
                Positions=
find(CData==i);
                Possize=
size(Positions);
                    for
j=1:Possize(1,1)
                        for
k=1:Savesize

if(k==Positions(j,1))




%
contents{get(hObject,'Value')}
returns selected item from
Choices
% --- Executes during object
creation, after setting all
```

```
                    end

Dsize=size(Stringdata);
            Stringdata=
Stringdata' ;
            for c=1:Dsize(1,2)

Stringdata{c,2}=i;

            end
            exdata{i}=
Stringdata;
            clear Stringdata;
            counter=1;
            end
            rxdata= exdata' ;
            Rxside=size(rxdata);
            for i=1:Rxside(1,1)

fastinsert(handles.conn, tbl,
colnames, rxdata{i,:});
            end
            msgbox('Operation
Successful','Database
export','help');
          else
            msgbox('Cluster data
not available,please execute a
clustering
method','Violation','error');
          end
    end
end
guidata(hObject,handles);
% --- Executes on selection
change in Choices.
function
Choices_Callback(hObject,
eventdata, handles)
% hObject    handle to Choices
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)

% Hints: contents =
get(hObject,'String') returns
Choices contents as cell array




% hObject    handle to Threshold
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
properties.
function
Choices_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to Choices
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called

% Hint: popupmenu controls
usually have a white background
on Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))

set(hObject,'BackgroundColor','wh
ite');
end
function
Threshold_Callback(hObject,
eventdata, handles)
% hObject    handle to Threshold
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)

% Hints: get(hObject,'String')
returns contents of Threshold as
text
%
str2double(get(hObject,'String'))
returns contents of Threshold as
a double
% --- Executes during object
creation, after setting all
properties.
function
Threshold_CreateFcn(hObject,
eventdata, handles)




if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))
```

```
created until after all
CreateFcns called

% Hint: edit controls usually
have a white background on
Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))

set(hObject,'BackgroundColor','wh
ite');
end
function Pan_Callback(hObject,
eventdata, handles)
% hObject    handle to Pan (see
GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)

% Hints: get(hObject,'String')
returns contents of Pan as text
%
str2double(get(hObject,'String'))
returns contents of Pan as a
double
% --- Executes during object
creation, after setting all
properties.
function Pan_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to Pan (see
GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called

% Hint: edit controls usually
have a white background on
Windows.
%       See ISPC and COMPUTER.



guidata(hObject,handles);


% --- Executes during object
creation, after setting all
```

```
set(hObject,'BackgroundColor','wh
ite');
end
% --- Executes on button press in
Transfer.
function
Transfer_Callback(hObject,
eventdata, handles)
% hObject    handle to Transfer
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)
%This variable is to verify if a
database is loaded or no.Most of
my global
%variables are located here.
handles.methodone=0;
handles.clusterdone=0;
handles.load=0;
DBLoaderFigureHandle  =
Databaseloader; %stores the
figure handle of loader GUI here

%stores the GUI data from loader
GUI here
%now we can access any of the
data from the loader GUI.
LData =
guidata(DBLoaderFigureHandle);

databaseload= LData.Lload;
handles.DBdata= LData.LBDataN;
handles.DBSdata= LData.LBData;
handles.SaveTB= LData.LBSave;
handles.SaveCOS= LData.LCOSave;
handles.SaveCOSb= LData.LCOSsaveb
%msgbox('Operation
Successful','Database
export','help');
handles.load= databaseload;
handles.conn= LData.conn;
sizeinfo= size(handles.DBdata);
set(handles.Datapoints,'String',n
um2str(sizeinfo(1,1)));
set(handles.Dimensionality,'Strin
g',num2str(sizeinfo(1,2)));




% handles    structure with
handles and user data (see
GUIDATA)

% Hints: contents =
get(hObject,'String') returns
```

```
properties.
function
Transfer_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to Transfer
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called



% --- Executes during object
creation, after setting all
properties.
function
Analyse_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to Analyse
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called
% --- Executes during object
creation, after setting all
properties.
function
Execute_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to Execute
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called
% --- Executes on selection
change in Clustering.
function
Clustering_Callback(hObject,
eventdata, handles)
% hObject    handle to Clustering
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB



% --- Executes during object
creation, after setting all
properties.
function Dsave_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to Dsave (see
```

```
Clustering contents as cell array
%
contents{get(hObject,'Value')}
returns selected item from
Clustering
% --- Executes during object
creation, after setting all
properties.
function
Clustering_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to Clustering
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called

% Hint: popupmenu controls
usually have a white background
on Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))

set(hObject,'BackgroundColor','wh
ite');
end
% --- Executes on selection
change in Dsave.
function Dsave_Callback(hObject,
eventdata, handles)
% hObject    handle to Dsave (see
GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)

% Hints: contents =
get(hObject,'String') returns
Dsave contents as cell array
%
contents{get(hObject,'Value')}
returns selected item from Dsave



handles.CDist='cityblock';

elseif(get(handles.PDistance,'Val
ue')==5)
```

```
GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called

% Hint: popupmenu controls
usually have a white background
on Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))

set(hObject,'BackgroundColor','wh
ite');
end
% --- Executes on selection
change in PDistance.
function
PDistance_Callback(hObject,
eventdata, handles)
% hObject    handle to PDistance
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)

if(get(handles.PDistance,'Value')
==1)

handles.CDist='euclidean';

elseif(get(handles.PDistance,'Val
ue')==2)

handles.CDist='chebychev';

elseif(get(handles.PDistance,'Val
ue')==3)

handles.CDist='minkowski';

elseif(get(handles.PDistance,'Val
ue')==4)



% Hints: get(hObject,'String')
returns contents of Exponent as
text
%
str2double(get(hObject,'String'))
returns contents of Exponent as a
```

```
handles.CDist='Myeuclidean';
          end


guidata(hObject,handles);

% Hints: contents =
get(hObject,'String') returns
PDistance contents as cell array
%
contents{get(hObject,'Value')}
returns selected item from
PDistance




% --- Executes during object
creation, after setting all
properties.
function
PDistance_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to PDistance
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called
% Hint: popupmenu controls
usually have a white background
on Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))

set(hObject,'BackgroundColor','wh
ite');
end
function
Exponent_Callback(hObject,
eventdata, handles)
% hObject    handle to Exponent
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)

% The main diagonal of the matrix
needs to be filled with a  value
so
% that the algorithm doesn't find
0 as the minimum value. We use
the value
```

```
double


% --- Executes during object
creation, after setting all
properties.
function
Exponent_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to Exponent
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called
% Hint: edit controls usually
have a white background on
Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))

set(hObject,'BackgroundColor','wh
ite');
end



function tpc = Hclu(Distance,
Method, criterion,Cnum)

switch criterion
    case 'maxclust'
% The function gets four inputs
to perform the clustering
methods.
% Distance : Is the square
similarity matrix containing the
distances between our
% data points.In that manner that
for example element (1,2) is the
distance
% between the data points (or
clusters) 1 and 2. As one can
understand element
% (1,1) value is zero because the
point has zero distance from
itself!



%First, the size of the
similarity matrix is also the
number of our data points.
%Since the matrix is square,
there is no worry of getting
frustrated with
% NaN.
%
%                    |0.0 0.1 0.3
0.9|
%                    |0.1 0.0 0.5
0.6|
%                    |0.3 0.5 0.0
0.7|
%                    |0.9 0.6 0.7
0.0|
%
% So ..the final input needs to
be like this :
%
%                    |NaN 0.1 0.3
0.9|
%                    |0.1 NaN 0.5
0.6|
%                    |0.3 0.5 NaN
0.7|
%                    |0.9 0.6 0.7
NaN|
% (NaN stands for the initials of
"Not a Number".)
%
% Method : This variable is quite
self explenatory. Lets the
algorithm
% decide which clustering method
to execute.
%
% criterion : This variable gets
the input of the user as to how
many
% clusters there is to be
created. If we let the algorithm
execute to the end, all
% our points will merge into a
single cluster, which may not be
useful for
% our purpose.It can be either
the maximum number of clusters or
a value
% for distance.
%
% Cnum: Holds the input of the
user regarding the chosen
criterion.
%
% tpc : This is what the
algorithm returns as an output.




    if MinI > MinJ
      temp=MinI;
      MinI=MinJ;
      MinJ=temp;
    end
```

```
%number of columns or rows. So, the variable m is the number of our data points.
m= size(Distance);
m= m(1,1);

%Assign each data point to it's own cluster. Variable cluster changes
%according the progress of clustering.
cluster=num2cell(1:m);
%The for loop will execute m times,where m is the number of our data points minus the number clusters we want to
%create. If our criterion is maxclust.
for i = 1:(m-Cnum)

% min(Distance) returns the minimum value of each column to the variable
% MinCol and indice value of the row to variable IDrow.
   [MinCoL, IDrow] = min(Distance);
% dbp : variable holds the minimum distance that the min function returns.
% Minimum of all the columns minimums is the distance we seek.
% MinJ : holds the column value of that distance, since the previous command returned
% something like that :
% MinCol: 0.2159    0.1432    0.1020    0.1581    0.1432 0.1020
% IDrow :   3         3         6         2       3
   [dbp, MinJ] = min(MinCoL);
%Find the row of the value by giving input the column.
   MinI = IDrow(MinJ);
%Since we notice that we are at the lower diagonal, we prefer however to
%work in the upper diagonal, so we reverse the values of MinI and MinJ.




   switch Method
   case 'single'
      Distance(:, MinI) = min(Distance(:, MinI), Distance(:, MinJ));
      Distance(MinI, :) = min(Distance(:, MinI), Distance(:, MinJ));
   case 'complete'
      Distance(:, MinI) = max(Distance(:, MinI), Distance(:, MinJ));
      Distance(MinI, :) = max(Distance(:, MinI), Distance(:, MinJ));
   case 'average'
      Distance(:, MinI) = (Distance(:, MinI) + Distance(:, MinJ))/2;
      Distance(MinI, :) = (Distance(:, MinI) + Distance(:, MinJ))/2;
   case 'weighted'
      Distance(:, MinI) = (Distance(:, MinI) + Distance(:, MinJ))/2;
      Distance(MinI, :) = (Distance(:, MinI) + Distance(:, MinJ))/2;
   otherwise
      error('Unknown method.');
   end
   Distance(MinI, MinI) = inf;
   Distance(MinJ, :) = inf;
   Distance(:, MinJ) = inf;
   %For method output delete the comment symbol of the following line
   %sprintf('The distance of %d and %d is  %f , iteration %d',MinI,MinJ,dbp,i)
   %Merge cluster j into cluster i, then delete j
   cluster{MinI} = [cluster{MinI} cluster{MinJ}];
   cluster(MinJ) = [];
   %For method output delete the comment symbol of the following line
   %sprintf('Merge cluster %d into %d',MinJ,MinI)
   %The new distance matrix is calculated depending on the input of the
   %user. Min function returns the minimum value of each row from the specified columns of the two
   %matrixes, for example.
   %
   %                    Return of minimum
   % 0.2354   0.2159                    0.2159
   % 0.2435   0.1432                    0.1432
   % 0.1020      NaN                     0.1020   (<- notice how NaN is ignored in the calculations)
   % 0.2195   0.1581                    0.1581
   % 0.3860   0.2846                    0.2846
   %    NaN   0.1020                    0.1020
   % We update the columns and rows with indice of MinI because we are
   % going to discard the MinJ ones, since we deleted the j cluster and
   % merged it into i. We place our new distances in the place of i.




   cluster=num2cell(1:m);
   while((dbp < Cnum) && counter<m)
      [MinCoL, IDrow] = min(Distance);
      [dbp, MinJ] = min(MinCoL);
```

```
min(Distance(MinI, :),
Distance(MinJ, :));
   case 'complete'
      Distance(:, MinI) =
max(Distance(:, MinI),
Distance(:, MinJ));
      Distance(MinI, :) =
max(Distance(MinI, :),
Distance(MinJ, :));
   end

   %Deletition of the cluster j
columns and place NaN.
   Distance(MinJ, :) = [];
   Distance(:, MinJ) = [];
   Distance(MinI, MinI) = NaN;
end
%Rearrange the results of
clustering into a new variable
tpc, which will
%hold the results in a similar
way as the output that function
cluster of
%Matlab gives.
temp=1;
for i=1:size(cluster,2)
    for j=1:size(cluster{i},2)

Mycluster(temp)=cluster{i}(j);
        ClusterIndex(temp)=i;
        temp=temp+1;
    end
end
Cindex(:,1)=Mycluster';
Cindex(:,2)=ClusterIndex';
for i=1:size(Cindex)
    temp=find(Cindex(:,1)==i);
    tpc(i)=Cindex(temp,2);
end
tpc=tpc';
%For the distance criterion, all
that changes is the condition of
the loop.
%The loop terminates anyway at m
iterations.
case 'distance'
   dbp=0;
   m= size(Distance);
   m= m(1,1);
   counter=1;




temp=1;
for i=1:size(cluster,2)
    for j=1:size(cluster{i},2)

Mycluster(temp)=cluster{i}(j);
```

```
       MinI = IDrow(MinJ);

       if MinI > MinJ
        temp=MinI;
        MinI=MinJ;
        MinJ=temp;
       end
       %For method output delete
the comment symbol of the
following line
       %sprintf('The distance of
%d and %d is  %f , iteration
%d',MinI,MinJ,dbp,counter)
       cluster{MinI} =
[cluster{MinI} cluster{MinJ}];
       cluster(MinJ) = [];
        %For method output delete
the comment symbol of the
following line
        %sprintf('Merge cluster
%d into %d',MinJ,MinI)
    switch Method
      case 'single'
        Distance(:, MinI) =
min(Distance(:, MinI),
Distance(:, MinJ));
        Distance(MinI, :) =
min(Distance(MinI, :),
Distance(MinJ, :));
      case 'complete'
        Distance(:, MinI) =
max(Distance(:, MinI),
Distance(:, MinJ));
        Distance(MinI, :) =
max(Distance(MinI, :),
Distance(MinJ, :));
    end
    Distance(MinJ, :) = [];
    Distance(:, MinJ) = [];
    Distance(MinI, MinI) = NaN;
    counter=counter+1;
    %Check the next distance so
the loop can terminate
effectively if it
    %exceeds the input of the
user.
    [MinCoL, IDrow] =
min(Distance);
    [dbp, MinJ] = min(MinCoL);
   end
```

```
        ClusterIndex(temp)=i;
        temp=temp+1;
    end
end
Cindex(:,1)=Mycluster';
Cindex(:,2)=ClusterIndex';
for i=1:size(Cindex)
    temp=find(Cindex(:,1)==i);
    tpc(i)=Cindex(temp,2);
end
tpc=tpc';
end
function Distance =
CluD(Inputarray)

    %Gets as input a matrix with
numeric data and calculates
    %the eucledian distance and
arranges the distances into a
square similarity matrix.
    %then places NaN in the main
diagonal. r is the
    %number of our data points
which is equal to the number of
rows.

 [r,c]= size(Inputarray);
for i= 1:r
    for j = i:r
      Distance(i,j) =
sqrt(sum((Inputarray(i,:)-
Inputarray(j,:)).^2));
    end
 end
Distance=Distance+Distance';

 for i= 1:r
     Distance(i,i)=NaN;
 end
```

Αρχείο Databaseloader.m

```
function varargout =
Databaseloader(varargin)
% DATABASELOADER M-file for
Databaseloader.fig
%      DATABASELOADER, by itself,


gui_Singleton = 1;
gui_State = struct('gui_Name',
mfilename, ...
```

```
creates a new DATABASELOADER or
raises the existing
%      singleton*.
%
%      H = DATABASELOADER returns
the handle to a new
DATABASELOADER or the handle to
%      the existing singleton*.
%
%
DATABASELOADER('CALLBACK',hObject
,eventData,handles,...) calls the
local
%      function named CALLBACK in
DATABASELOADER.M with the given
input arguments.
%
%
DATABASELOADER('Property','Value'
,...) creates a new
DATABASELOADER or raises the
%      existing singleton*.
Starting from the left, property
value pairs are
%      applied to the GUI before
Databaseloader_OpeningFcn gets
called.  An
%      unrecognized property name
or invalid value makes property
application
%      stop.  All inputs are
passed to
Databaseloader_OpeningFcn via
varargin.
%
%      *See GUI Options on
GUIDE's Tools menu.  Choose "GUI
allows only one
%      instance to run
(singleton)".
%
% See also: GUIDE, GUIDATA,
GUIHANDLES

% Edit the above text to modify
the response to help
Databaseloader

% Last Modified by GUIDE v2.5 02-
May-2009 17:00:18

% Begin initialization code - DO
NOT EDIT

% UIWAIT makes Databaseloader
wait for user response (see
UIRESUME)
% uiwait(handles.Databaseloader);
```

```
'gui_Singleton',  gui_Singleton,
...

'gui_OpeningFcn',
@Databaseloader_OpeningFcn, ...

'gui_OutputFcn',
@Databaseloader_OutputFcn, ...

'gui_LayoutFcn',  [] , ...

'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback =
str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] =
gui_mainfcn(gui_State,
varargin{:});
else
    gui_mainfcn(gui_State,
varargin{:});
end
% End initialization code - DO
NOT EDIT
% --- Executes just before
Databaseloader is made visible.
function
Databaseloader_OpeningFcn(hObject
, eventdata, handles, varargin)
% This function has no output
args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)
% varargin   command line
arguments to Databaseloader (see
VARARGIN)

% Choose default command line
output for Databaseloader
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);




%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))
```

```
% --- Outputs from this function
are returned to the command line.
function varargout =
Databaseloader_OutputFcn(hObject,
eventdata, handles)
% varargout  cell array for
returning output args (see
VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)

% Get default command line output
from handles structure
varargout{1} = handles.output;
function edit1_Callback(hObject,
eventdata, handles)
% hObject    handle to edit1 (see
GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)

% Hints: get(hObject,'String')
returns contents of edit1 as text
%
str2double(get(hObject,'String'))
returns contents of edit1 as a
double
% --- Executes during object
creation, after setting all
properties.
function edit1_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to edit1 (see
GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called
% Hint: edit controls usually
have a white background on
Windows.


% --- Executes during object
creation, after setting all
properties.
function
Showdatabases_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to
```

```
set(hObject,'BackgroundColor','wh
ite');
end
% --- Executes on selection
change in Showdatabases.
function
Showdatabases_Callback(hObject,
eventdata, handles)
% hObject    handle to
Showdatabases (see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)

% Hints: contents =
get(hObject,'String') returns
Showdatabases contents as cell
array
%
contents{get(hObject,'Value')}
returns selected item from
%        Showdatabases
list1=get(handles.Showdatabases,'
String');
sel_dbno=get(handles.Showdatabase
s,'Value');
handles.sel_db=list1(sel_dbno);
%exec(handles.conn,['USE ['
handles.sel_db{1} ']']);
metadata=dmd(handles.conn);
sources=
tables(metadata,handles.sel_db,'d
bo');
sourcesize= size(sources);
for i=1:sourcesize(1,1)

if(strcmp(sources{i,2},'TABLE'));

string{i}=sprintf('%s',sources{i}
);
   end
end
set(handles.Showtables,'String',s
tring);
guidata(hObject,handles);




sel_cols=columns(metadata,handles
.sel_db,'dbo',handles.sel_table);
set(handles.Columndisplay,'String
',sel_cols);
guidata(hObject,handles);
% --- Executes during object
```

```
Showdatabases (see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called

% Hint: popupmenu controls
usually have a white background
on Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))

set(hObject,'BackgroundColor','wh
ite');
end
% --- Executes on selection
change in Showtables.
function
Showtables_Callback(hObject,
eventdata, handles)
% hObject    handle to Showtables
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)

% Hints: contents =
get(hObject,'String') returns
Showtables contents as cell array
%
contents{get(hObject,'Value')}
returns selected item from
Showtables
metadata= dmd(handles.conn);
sel_list_table=get(handles.Showta
bles,'String');
sel_table_no=get(handles.Showtabl
es,'Value');
handles.sel_table=sel_list_table(
sel_table_no);




% --- Executes on button press in
Selecttable.
function
Selecttable_Callback(hObject,
eventdata, handles)
% hObject    handle to
Selecttable (see GCBO)
```

```
creation, after setting all
properties.
function
Showtables_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to Showtables
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called
% Hint: popupmenu controls
usually have a white background
on Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))

set(hObject,'BackgroundColor','wh
ite');
end
% --- Executes on button press in
Connectistance.
function
Connectistance_Callback(hObject,
eventdata, handles)
% hObject    handle to
Connectistance (see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)
% --- Executes on button press in
SelectDB.
function
SelectDB_Callback(hObject,
eventdata, handles)
% hObject    handle to SelectDB
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)



% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called
```

```
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)
% --- Executes on selection
change in Showinstances.
function
Showinstances_Callback(hObject,
eventdata, handles)
% hObject    handle to
Showinstances (see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)
% Hints: contents =
get(hObject,'String') returns
Showinstances contents as cell
array
%
contents{get(hObject,'Value')}
returns selected item from
Showinstances

%The following lines of code get
the available ODBC instances
list1=get(handles.Showinstances,'
String');
sel_inno=get(handles.Showinstance
s,'Value');
handles.sel_in=list1(sel_inno);
handles.conn =
database(handles.sel_in,'','');
metadata= dmd(handles.conn);
sources=get(metadata);
set(handles.Showdatabases,'String
',sources.Catalogs);
guidata(hObject,handles);
% --- Executes during object
creation, after setting all
properties.
function
Showinstances_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to
Showinstances (see GCBO)



    for i=1:size(sel_cols,2)-1

str1{i}=str2mat(sprintf('%s,',sel
_cols{1,i}));
    end

str1{size(sel_cols,2)}=str2mat(sp


% Hint: popupmenu controls
usually have a white background
on Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))

set(hObject,'BackgroundColor','wh
ite');

end
setdbprefs('TempDirForRegistryOut
put','c:\temp')
myODBCdir = getenv('WINDIR');
sources= getdatasources;
set(hObject,'String',sources);
guidata(hObject,handles);
% --- Executes on button press in
Load.
function Loader_Callback(hObject,
eventdata, handles)
% hObject    handle to Loader
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)
DB= handles.sel_db;
Table= handles.sel_table{1};
IN= handles.sel_in;
exec(handles.conn,['USE ['
handles.sel_db{1} ']']);

%sel_list_cols=get(handles.Column
display,'String');

%sel_cols_no=get(handles.Columndi
splay,'Value');

%sel_cols=sel_list_cols(sel_cols_
no);
    sel_cols=handles.sdata;

sel_cols(1,2:(size(handles.numdat
a,1)+1))=handles.numdata(:,:);
    str1='';



% --- Executes on selection
change in Options.
function
Options_Callback(hObject,
eventdata, handles)
% hObject    handle to Options
```

```
rintf('%s',sel_cols{1,size(sel_co
ls,2)}));
    str1=[str1{:}];

    %str2='';
    %for
i=1:size(handles.numdata,1)-1
    %
str2{i}=str2mat(sprintf('%s,',han
dles.numdata{i,1}));
    %end

%str2{size(handles.numdata,1)}=st
r2mat(sprintf('%s',handles.numdat
a{size(sel_cols,1),1}));
    %str2=[str2{:}];
%handles.LBData=
fetch(handles.conn,['select ['
str1 '] from [' Table ']']);
if
(get(handles.LoadSave,'Value')==1
)

setdbprefs('DataReturnFormat','ce
llarray');
    handles.LBData=
fetch(handles.conn,['select '
str1 ' from [' Table ']']);

setdbprefs('DataReturnFormat','nu
meric');
    handles.LBDataN=
fetch(handles.conn,['select '
str1 ' from [' Table ']']);
    handles.LBDataN=
handles.LBDataN(:,2:size(handles.
LBDataN,2));
    handles.Lload=1;
elseif
(get(handles.LoadSave,'Value')==2
)
    handles.LBSave= Table;
    handles.LCOSave=
handles.sdata{1,1};
    handles.LCOSsaveb=
handles.sdata{2,1};
end

guidata(hObject,handles);




% hObject    handle to test (see
GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
```

```
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)

% Hints: contents =
get(hObject,'String') returns
Options contents as cell array
%
contents{get(hObject,'Value')}
returns selected item from
Options



% --- Executes during object
creation, after setting all
properties.
function
Options_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to Options
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called
% Hint: popupmenu controls
usually have a white background
on Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))

set(hObject,'BackgroundColor','wh
ite');
end
% --- Executes on selection
change in test.
function test_Callback(hObject,
eventdata, handles)




% --- Executes on selection
change in LoadSave.
function
LoadSave_Callback(hObject,
eventdata, handles)
% hObject    handle to LoadSave
```

```
GUIDATA)

% Hints: contents =
get(hObject,'String') returns
test contents as cell array
%
contents{get(hObject,'Value')}
returns selected item from test
% --- Executes during object
creation, after setting all
properties.
function test_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to test (see
GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called

% Hint: listbox controls usually
have a white background on
Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))

set(hObject,'BackgroundColor','wh
ite');
end
% --- Executes during object
creation, after setting all
properties.
function
Connectistance_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to
Connectistance (see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called




% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)
```

```
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)

% Hints: contents =
get(hObject,'String') returns
LoadSave contents as cell array
%
contents{get(hObject,'Value')}
returns selected item from
LoadSave
% --- Executes during object
creation, after setting all
properties.
function
LoadSave_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to LoadSave
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called

% Hint: popupmenu controls
usually have a white background
on Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))

set(hObject,'BackgroundColor','wh
ite');
end
% --- Executes on selection
change in Showschema.
function
Showschema_Callback(hObject,
eventdata, handles)
% hObject    handle to Showschema
(see GCBO)




% Hints: contents =
get(hObject,'String') returns
Showcolumns contents as cell
array
%
contents{get(hObject,'Value')}
```

```
% Hints: contents =
get(hObject,'String') returns
Showschema contents as cell array
%
contents{get(hObject,'Value')}
returns selected item from
Showschema
% --- Executes during object
creation, after setting all
properties.
function
Showschema_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to Showschema
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called

% Hint: popupmenu controls
usually have a white background
on Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))

set(hObject,'BackgroundColor','wh
ite');
end
% --- Executes on selection
change in Showcolumns.
function
Showcolumns_Callback(hObject,
eventdata, handles)
% hObject    handle to
Showcolumns (see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)
```

```
returns selected item from
Showcolumns
%metadata= dmd(handles.conn);
%cata=
handles.Databases(handles.DBselec
t);
%sch=handles.SHinfo(handles.SHsel
ect);
%tab=handles.TBinfo(handles.TBsel
ect);
% --- Executes during object
creation, after setting all
properties.
function
Showcolumns_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to
Showcolumns (see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called

% Hint: popupmenu controls
usually have a white background
on Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))

set(hObject,'BackgroundColor','wh
ite');
end
% --- Executes on selection
change in Selector.
function
Selector_Callback(hObject,
eventdata, handles)
% hObject    handle to Selector
(see GCBO)
```

```
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)
```

```
% --- Executes on button press in
ClearColumn.
function
ClearColumn_Callback(hObject,
eventdata, handles)
% hObject    handle to
```

```
% Hints: contents =
get(hObject,'String') returns
Selector contents as cell array
%
contents{get(hObject,'Value')}
returns selected item from
Selector
% --- Executes during object
creation, after setting all
properties.
function
Selector_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to Selector
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called

% Hint: popupmenu controls
usually have a white background
on Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))

set(hObject,'BackgroundColor','wh
ite');
end
% --- Executes on button press in
LoadColumn.
function
LoadColumn_Callback(hObject,
eventdata, handles)
% hObject    handle to LoadColumn
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)
```

```
ClearColumn (see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)
% --- Executes during object
creation, after setting all
properties.
function
LoadColumn_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to LoadColumn
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called
% --- Executes during object
creation, after setting all
properties.
function
Loader_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to Loader
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called
% --- Executes on selection
change in Temp.
function Temp_Callback(hObject,
eventdata, handles)
% hObject    handle to Temp (see
GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)
```

```
% Hints: contents =
get(hObject,'String') returns
Temp contents as cell array
%
contents{get(hObject,'Value')}
returns selected item from Temp
% --- Executes during object
```

```
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called
```

```
creation, after setting all
properties.
function Temp_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to Temp (see
GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called
% Hint: listbox controls usually
have a white background on
Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))

set(hObject,'BackgroundColor','wh
ite');
end
% --- Executes on selection
change in Temp.
function
listbox4_Callback(hObject,
eventdata, handles)
% hObject    handle to Temp (see
GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)

% Hints: contents =
get(hObject,'String') returns
Temp contents as cell array
%
contents{get(hObject,'Value')}
returns selected item from Temp
% --- Executes during object
creation, after setting all
properties.
function
listbox4_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to Temp (see
GCBO)

% Hint: listbox controls usually
have a white background on
Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
```

```
% Hint: listbox controls usually
have a white background on
Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundCo
lor'),
get(0,'defaultUicontrolBackground
Color'))

set(hObject,'BackgroundColor','wh
ite');
end
% --- Executes on selection
change in Columndisplay.
function
Columndisplay_Callback(hObject,
eventdata, handles)
% hObject    handle to
Columndisplay (see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)

% Hints: contents =
get(hObject,'String') returns
Columndisplay contents as cell
array
%
contents{get(hObject,'Value')}
returns selected item from
Columndisplay
% --- Executes during object
creation, after setting all
properties.
function
Columndisplay_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to
Columndisplay (see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    empty - handles not
created until after all
CreateFcns called




sel_list_cols=get(handles.Columnd
isplay,'String');

sel_cols_no=get(handles.Columndis
play,'Value');
```

```
get(0,'defaultUicontrolBackground
Color'))

set(hObject,'BackgroundColor','wh
ite');
end
% --- Executes on button press in
Blabels.
function
Blabels_Callback(hObject,
eventdata, handles)
% hObject    handle to Blabels
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)

sel_list_cols=get(handles.Columnd
isplay,'String');

sel_cols_no=get(handles.Columndis
play,'Value');

sel_cols=sel_list_cols(sel_cols_n
o);
    handles.sdata=sel_cols;

set(handles.Columndisplay,'Value'
,1);
    guidata(hObject,handles);
% --- Executes on button press in
Bnumeric.
function
Bnumeric_Callback(hObject,
eventdata, handles)
% hObject    handle to Bnumeric
(see GCBO)
% eventdata  reserved - to be
defined in a future version of
MATLAB
% handles    structure with
handles and user data (see
GUIDATA)
```

```
sel_cols=sel_list_cols(sel_cols_n
o);
    handles.numdata=sel_cols;

set(handles.Columndisplay,'Value'
,1);
    guidata(hObject,handles);
```

Ευρετήριο Και Συνοπτικός Οδηγός Χρήσης της Εφαρμογης:

Analyse! (σελ 110, Analyse_Callback(hObject, eventdata, handles)
Distance Matrix : Εμφάνιση πίνακα αποστάσεως.
Plot Original Data : Γραφική παράσταση αρχικών δεδομένων.

Display Data : Εμφάνιση αρχικών δεδομένων.
Linkage : Εμφάνιση δεσμών μεθόδου συσταδοποίησης.
Clusters : Εμφάνιση συστάδων.
Dendrogram : Κατασκευή δενδρογράμματος.
Plot Clustered Data : Γραφική παράσταση συσταδοποιημένων δεδομένων.
Save Clustered Data to Database : Αποθύκευση δεδομένων συσταδοποίησης στη βάση δεδομένων.

Execute (σελ 105, Execute_Callback(hObject, eventdata, handles)
Single Link : Εκτελεί μέθοδο συσταδοποίησης απλού δεσμού.
Complete Link : Εκτελεί μέθοδο συσταδοποίησης πλήρους δεσμού.
Average Link : Εκτελεί μέθοδο συσταδοποίησης μέσου δεσμού.
Centroid : Εκτελεί μέθοδο συσταδοποίησης Centroid.
Median : Εκτελεί μέθοδο συσταδοποίησης Median.
Ward : Εκτελεί μέθοδο συσταδοποίησης Ward.
Average Link (Weighted) : Εκτελεί μέθοδο συσταδοποίησης μέσου δεσμού με βάρη.
My Single Link : Εκτελεί μέθοδο συσταδοποίησης απλού δεσμού.
My Complete Link : Εκτελεί μέθοδο συσταδοποίησης πλήρους δεσμού.

Clustering Parameters
Maximum number of clusters : Καθορισμός μέγιστου αριθμού συστάδων κατά τη συσταδοποίηση.
Cutoff : Καθορισμός αριθμού συστάδων βάση του κριτηρίου της απόστασης.

Metric (σελ 116, PDistance_Callback(hObject, eventdata, handles)
Euclidean distance : Χρήση ευκλείδειας απόστασης για την συσταδοποίηση και τον πίνακα απόστασης.
Chebyshev Distance : Χρήση απόστασης Chebyshev για την συσταδοποίηση και τον πίνακα απόστασης.
Minkowski Distance : Χρήση απόστασης Minkowski για την συσταδοποίηση και τον πίνακα απόστασης.
Manhattan Distance (City Block) : Χρήση απόστασης Manhattan για την συσταδοποίηση και τον πίνακα απόστασης.
My Euclidean distance : Χρήση ευκλείδειας απόστασης για την συσταδοποίηση και τον πίνακα απόστασης.

Minkowski Exponent
Καθορισμός του εκθέτη στον τύπο της απόστασης Minkowski.

Transfer Data *(σελ 114,* Transfer_Callback(hObject, eventdata, handles)
Μεταφορά δεδομένων από την εφαρμογή Database Loader στον Clustering Analyser.

Database Loader *(σελ 107,* Database_Callback(hObject, eventdata, handles)
Κλήση της εφαρμογής Database Loader.

Dendrogram Properties
Threshold : Δίνει ξεχωριστό χρώμα για κάθε ομάδα δεσμών κάτω από την απόσταση που δόθηκε ως είσοδο.

Database Information
Data Points : Ο αριθμός των σημείων που είναι διαθέσιμα προς επεξεργασία.
Dimensionality : Η διάσταση των σημείων προς επεξεργασία.

Available Instances (ODBC Data Sources) (σελ 124 Showinstances_Callback(hObject, eventdata, handles) , Showinstances_CreateFcn(hObject, eventdata, handles)
Επιλογή ενός από τα διαθέσιμα data sources .

Available Databases in Instance (σελ 122, Showdatabases_Callback(hObject, eventdata, handles)
Επιλογή μίας από τις διαθέσιμες βάσεις δεδομένων.

List of Tables in Current Database (σελ 123, Showtables_Callback(hObject, eventdata, handles)

Επιλογή πίνακα από τους διαθέσιμους.

Select The Columns Of The Table

Επιλογή στηλών από τον πίνακα για τον καθορισμό των στοιχείων προς επεξεργασία.

Select (σελ 124, Loader_Callback(hObject, eventdata, handles)
Table to Load : Φορτώνει τις επιλογές του χρήστη σε αριθμητικά δεδομένα και ετικέτες και τις θέτει ως δεδομένα προς επεξεργασία.
Table to Save : Κρατά τις επιλογές του χρήστη για την αποθήκευση των στοιχείων σε βάση δεδομένων.

Numeric (σελ 130, Bnumeric_Callback(hObject, eventdata, handles)
Οι επιλογές του χρήστη από τη λίστα των διαθέσιμων στηλών δηλώνονται ως αριθμητικά δεδομένα.

Labels (σελ 130 Blabels_Callback(hObject, eventdata, handles)
Οι επιλογές του χρήστη από τη λίστα των διαθέσιμων στηλών δηλώνονται ως ετικέτες.

Βιβλιογραφία :

Data Clustering: Theory, Algorithms, and Applications (ASA-SIAM Series on Statistics and Applied Probability)

Clustering, Riu Xu and Donald C. Wunsch II

Constrained Clustering, Advances in Algorithms, Theory and applications Edited by Sugato Basu,Ian Davidson,Kiri L. Wagstaff

http://en.wikipedia.org/wiki/Data_clustering

http://www.albionresearch.com/data_mining/market_basket.php (market-basket data)

Ka Yee Yeung , Clustering 101 University of Washington

Cluster Analysis : Document prepared by Robert L. Andrews, April 2005

Kejun (Kevin) Mei : Question about the Wards clustering method

http://statsoft.eu/uk/textbook/stcluan.html

Jia Huang CS 157B , SJSU Department of Computer Science

Chien Chin Chen, Hierarchical Clustering - Department of Information Management , National Taiwan University

SingleLinkExample - Angelina A Tzacheva, Ph.D. - Department of Informatics. University of South Carolina Upstate.

Northeastern University: College of Computer and Information Science

http://en.wikipedia.org/wiki/MATLAB

http://en.wikipedia.org/wiki/Centroid

http://www.stat.psu.edu/online/development/stat505/18_cluster/09_cluster_wards.html

Γεωγραφικά Δεδομένα για πόλεις της Ελλάδας (γεωγραφικό μήκος, πλάτος κτλ..) http://earth-info.nima.mil/gns/html/namefiles.htm

Τιμές των μετοχών της IBM. (Open for Day,High for Day..κτλ) http://www.econstats.com/eqty/eq_d_in_19.htm

Στατιστικά Δεδομένα. http://new.wales.gov.uk/topics/statistics/headlines/agric-2007/agric-2006/hdw20060801/?lang=en (Statistics on agriculture at a local level produced by the Welsh Assembly Government)

Εκπαιδευτικά δεδομένα. http://faculty.uscupstate.edu/atzacheva/

Εγγραφή σε δημόσια γυμνάσια και δημοτικά σχολεία ανά πολιτεία : 1980 έως 2004 (Public Elementary and Secondary School Enrollment by State: 1980 to 2004, Source: U.S. National Center for Education Statistics, Digest of Education Statistics, annual. , In thousands (27,647 represents 27,647,000), except rate. http://infochimps.org/home, http://www.nces.ed.gov/ .)

http://en.wikipedia.org/wiki/Chebyshev_distance