ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΜΗΧΑΝΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ

Διπλωματική Εργασία

# **Αλγόριθμοι Δρομολόγησης Οχήματος με Πολλαπλά Διαμερίσματα**

υπό

**ΦΩΤΙΑΔΗ ΓΕΡΑΣΙΜΟ**

Υπεβλήθη για την εκπλήρωση μέρους των

απαιτήσεων για την απόκτηση του

Διπλώματος Μηχανολόγου Μηχανικού

2012

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΜΗΧΑΝΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ

Διπλωματική Εργασία

# Αλγόριθμοι Δρομολόγησης Οχήματος με Πολλαπλά Διαμερίσματα

υπό

**ΦΩΤΙΑΔΗ ΓΕΡΑΣΙΜΟ**

Υπεβλήθη για την εκπλήρωση μέρους των

απαιτήσεων για την απόκτηση του

Διπλώματος Μηχανολόγου Μηχανικού

2012

**Εγκρίθηκε από τα Μέλη της Τριμελούς Εξεταστικής Επιτροπής:**

| | |
|---|---|
| Πρώτος Εξεταστής (Επιβλέπων) | Δρ. Δημήτριος Παντελής<br>Καθηγητής, Τμήμα Μηχανολόγων Μηχανικών<br>Πανεπιστήμιο Θεσσαλίας |
| Δεύτερος Εξεταστής | Δρ. Γεώργιος Κοζανίδης<br>Καθηγητής, Τμήμα Μηχανολόγων Μηχανικών<br>Πανεπιστήμιο Θεσσαλίας |
| Τρίτος Εξεταστής | Δρ. Τάσος Σταματέλλος<br>Καθηγητής, Τμήμα Μηχανολόγων Μηχανικών<br>Πανεπιστήμιο Θεσσαλίας |

## Ευχαριστίες

Πρώτα απ' όλα, θέλω να ευχαριστήσω τον επιβλέποντα της διπλωματικής εργασίας μου, κ. Δημήτρη Παντελή, για την πολύτιμη βοήθεια και καθοδήγησή του κατά τη διάρκεια της δουλειάς μου.

Επίσης, είμαι ευγνώμων στα υπόλοιπα μέλη της εξεταστικής επιτροπής , Καθηγητές κ. Γεώργιο Κοζανίδη και κ. Τάσο Σταματέλλο για την ανάγνωση της εργασίας μου και για τις πολύτιμες υποδείξεις τους..

Πάνω απ' όλα, είμαι ευγνώμων στους γονείς μου και στην αδερφή μου, στους οποίους και αφιερώνω αυτή την εργασία, για την αγάπη και την υποστήριξή τους όλα αυτά τα χρόνια.

*Γεράσιμος Φωτιάδης*

## Αλγόριθμοι Δρομολόγησης Οχήματος με Πολλαπλά Διαμερίσματα

Γεράσιμος Φωτιάδης

Πανεπιστήμιο Θεσσαλίας, Τμήμα Μηχανολόγων Μηχανικών, 2012

Επιβλέπων Καθηγητής: Δρ. Δημήτριος Παντελής

**Περίληψη**

Η επιχειρησιακή έρευνα είναι μια νέα σχετικά επιστήμη η οποία άρχισε να παίρνει μορφή κατά το δεύτερο παγκόσμιο πόλεμο. Έχει ως αντικείμενο την εύρεση βέλτιστης ή σχεδόν βέλτιστης λύσης, σε περίπλοκα προβλήματα αποφάσεων χρησιμοποιώντας ένα σύνολο από μαθηματικές τεχνικές. Το πρόβλημα δρομολόγησης οχημάτων (ΠΔΟ) διατυπώθηκε για πρώτη φορά από τους Dantzig και Ramser και είναι ένα από τα πιο γνωστά και μελετημένα προβλήματα αυτής της επιστήμης. Περιγράφει την παράδοση αγαθών που βρίσκονται σε μια κεντρική αποθήκη προς τους πελάτες. Συνήθης στόχος είναι η ελαχιστοποίηση του κόστους διανομής των εμπορευμάτων. Σκοπός της παρούσας διπλωματικής εργασίας είναι η ανάπτυξη κώδικα που θα βρίσκει την βέλτιστη στρατηγική διανομής. Αρχικά το πρόβλημα λύνεται με αναλυτική μέθοδο. Στην συνέχεια, με ευρετική που απαιτεί λιγότερο υπολογιστικό φόρτο. Τέλος, γίνετε σύγκριση των δύο μεθόδων και παρουσιάζονται τα αντίστοιχα διαγράμματα. Στο παράρτημα υπάρχει ο κώδικας που χρησιμοποιήθηκε για την δημιουργία της εργασίας.

# Πίνακας Περιεχομένων

## Κατάλογος Σχημάτων

# Κεφάλαιο 1 Εισαγωγή

Σε αυτό το κεφάλαιο, παρουσιάζουμε πληροφορίες εισαγωγικού χαρακτήρα που δίνουν το κίνητρο και το υπόβαθρο αυτής της διπλωματικής εργασίας, παραθέτουμε μια ανασκόπηση της σχετικής με την εργασία βιβλιογραφίας και περιγράφουμε συνοπτικά τις βασικές ενότητές της.

## 1.1 Κίνητρο και Υπόβαθρο

Κίνητρο αυτής της διπλωματικής εργασίας ήταν η έλλειψη εργαστηριακών ασκήσεων στα μαθήματα του Μαθηματικού Προγραμματισμού και των Στοχαστικών Προτύπων στην Επιχειρησιακή Έρευνα καθώς και το ιδιαίτερο ενδιαφέρον του φοιτητή για το αντικείμενο. Η επιχειρησιακή έρευνα είναι ένα αντικείμενο που η θεωρία του είναι αρκετά μεγάλη και μαθηματικοποιημένη με αποτέλεσμα να φαίνεται σαν ένα μάθημα καθαρά θεωρητικό. Η αλήθεια όμως είναι ότι είναι ένα μάθημα καθαρά πρακτικό, αφού η ίδια η επιστήμη γεννήθηκε από την ανάγκη να λυθούν προβλήματα και να παρθούν αποφάσεις που εμφανίζονται καθημερινά στην κοινωνία. Η συνεισφορά αυτής της διπλωματικής εργασίας είναι ότι, αναπτύσσοντας τους κώδικες για να λυθεί μια παραλλαγή ενός από τα πιο γνωστά προβλήματα της επιχειρησιακής έρευνας, γεφυρώνει το κενό ανάμεσα στη θεωρία και στην πράξη.

## 1.2 Βιβλιογραφική Ανασκόπηση

Η βιβλιογραφία που σχετίζεται με την επιστήμη της Επιχειρησιακής Έρευνας είναι εκτενής. Η συγκεκριμένη εργασία βασίστηκε στην εργασία [1] η οποία περιέχει την περιγραφή, την ανάλυση καθώς και τις μαθηματικές σχέσεις για το πρόβλημα που θα ασχοληθούμε.

Ακόμα χρησιμοποιήθηκαν τα βιβλία [2], [3] και [4] κυρίως για την θεωρητική κατάρτιση του φοιτητή. Εκεί μπορεί να ανατρέξει ο αναγνώστης για περισσότερες πληροφορίες για θέματα όπως δυναμικός προγραμματισμός, μοντέλα επιχειρησιακής έρευνας, στοχαστικές ζητήσεις και ότι άλλο σχετίζεται με την Επιχειρησιακή Έρευνα.

Όσο αναφορά την αλληλεπίδραση με το MATLAB, εξαιρετικά χρήσιμα αποδείχτηκαν τα βιβλία [5] και [6]. Το πρώτο είναι αφιερωμένο μόνο σε ότι έχει να κάνει με το προγραμματισμό σε περιβάλλον MATLAB, ενώ το δεύτερο καλύπτει θέματα πιο γενικά, χρήσιμα για όλους τους μηχανικούς.

Τέλος, πολύ σημαντικές είναι οι πληροφορίες που αντλήθηκαν από τον επίσημο ισότοπο της εταιρίας που αναπτύσσει το MATLAB "The MathWorks" [7] καθώς και από τον ισότοπο της εταιρίας "gliffy"[8], ο οποίος χρησιμοποιήθηκε για τον σχεδιασμό του διαγράμματος ροής.

## 1.3 Οργάνωση Διπλωματικής Εργασίας

Στο κεφάλαιο 2 γίνεται αναλυτική περιγραφή του προβλήματος, των μαθηματικών σχέσεων καθώς και των μεταβλητών που το διέπουν.

Στο κεφάλαιο 3 περιγράφετε η αναλυτική μέθοδος που χρησιμοποιήθηκε για την επίλυση την προβλήματος για 1, 2 και 3 είδη προϊόντων.

Στο κεφάλαιο 4 περιγράφετε η ευρετική μέθοδος που χρησιμοποιήθηκε για την επίλυση του προβλήματος για 2 και 3 είδη προϊόντων.

Στο κεφάλαιο 5 πραγματοποιείτε η σύγκριση των 2 μεθόδων όσο αναφορά την ταχύτητα και την απόκλιση των αποτελεσμάτων και παρουσιάζονται τα σχετικά διαγράμματα.

Τα τελικά συμπεράσματα της διπλωματικής εργασίας καθώς και κατευθύνσεις για περαιτέρω έρευνα παρουσιάζονται στο κεφάλαιο 6.

# Κεφάλαιο 2 Περιγραφή του Προβλήματος

## 2.1 Πρόβλημα Δρομολόγησης Οχημάτων

Ένα από τα πιο ευρέως μελετημένα προβλήματα στην επιχειρησιακή έρευνα είναι το πρόβλημα δρομολόγησης οχημάτων. Διατυπώθηκε από τους Dantzig και Ramser το 1959 και θεωρείται ως μια γενίκευση του προβλήματος του πλανόδιου πωλητή. Το πρόβλημα περιγράφεται ως εξής: ας υποθέσουμε ότι έχουμε να διαχειριστούμε μια ομάδα οχημάτων που ανεφοδιάζεται από μια ή περισσότερες αποθήκες και πρέπει να παραδώσει τα εμπορεύματα σε Ν γεωγραφικά διάσπαρτους πελάτες. Κάθε όχημα ξεκινά την διαδρομή του από μια αποθήκη, επισκέπτεται ένα υποσύνολο πελατών, ικανοποιεί τις ανάγκες τους και τελικά επιστρέφει στην αποθήκη.

Στόχος είναι ο προσδιορισμός της βέλτιστης διαδρομής για κάθε όχημα. Το κόστος περιλαμβάνει τα έξοδα της διαδρομής από το ένα πελάτη στον άλλο καθώς και από τον πελάτη πίσω στην αποθήκη για ανεφοδιασμό. Το σύνηθες κριτήριο βελτιστοποίησης είναι η ελαχιστοποίηση του συνολικού κόστους της διαδρομής. Μια διαφορετική ερμηνεία του ΠΔΟ είναι ο προσδιορισμός της βέλτιστης διαδρομής συλλογής για κάθε όχημα, εφόσον τα οχήματα συλλέγουν ληγμένα προϊόντα αντί να διανέμουν νέα. Η συλλογή των οικιακών αποβλήτων, η παράδοση των εμπορευμάτων σούπερ μάρκετ, η παράδοση βενζίνης σε πρατήρια, η προμήθεια προμαγειρεμένων τροφίμων σε καταστήματα μπορούν να θεωρηθούν σαν εφαρμογές του ΠΔΟ.

Οι τρεις σημαντικές παραλλαγές του ΠΔΟ που έχουν μελετηθεί εκτενώς είναι (i) the capacitated VRP, (ii) the VRP with time-window, (iii) the VPR with pick-up delivery. Στο πρώτο πρόβλημα όλα τα οχήματα είναι ίδια με πεπερασμένη χωρητικότητα, στο δεύτερο τα οχήματα διανέμουν τα αγαθά κατά την διάρκεια συγκεκριμένου "χρονικού παραθύρου" και στο τρίτο πρόβλημα μια συλλογή των αλλοιωμένων προϊόντων πραγματοποιείται πριν την διανομή των νέων. Ακόμα υπάρχουν και στοχαστικές παραλλαγές του προβλήματος, αν θεωρήσουμε ότι ένας οι περισσότεροι παράμετροι του προβλήματος είναι τυχαίες μεταβλητές. Για παράδειγμα, τυχαίες μεταβλητές μπορεί να είναι ο αριθμός των πελατών που εξυπηρετούνται, οι απαιτήσεις των πελατών και οι χρόνοι ταξιδιού.

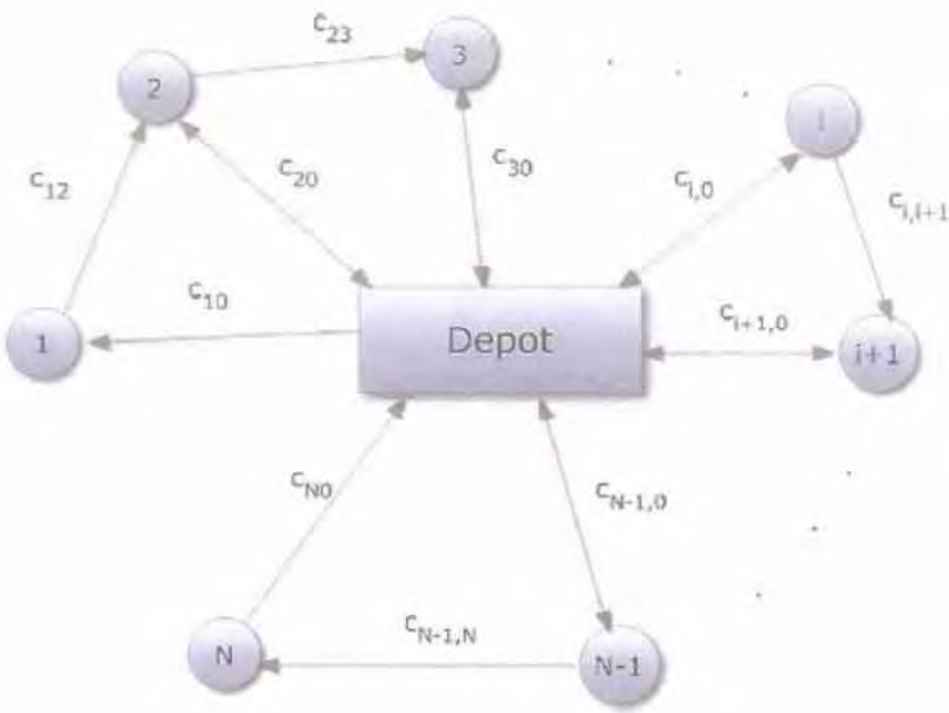
## 2.2 Ορισμός του Προβλήματος

Στην παρούσα εργασία θα ασχοληθούμε με μία απλή αλλά ενδιαφέρουσα παραλλαγή του ΠΔΟ.

Θεωρούμε ένα σύνολο κόμβων V={0,1,...,N} με τον κόμβο 0 να υποδηλώνει την αποθήκη και τους κόμβους 1,...,N τους πελάτες. Οι πελάτες εξυπηρετούνται με την σειρά 1,2,...,N από το όχημα που αποτελείται από K τμήματα με χωρητικότητα $Q_1, Q_2, \ldots, Q_k$. Κάθε τμήμα είναι κατάλληλο για ένα τύπο προϊόντος και μόνο. Υπάρχουν k διαφορετικά προϊόντα που πρέπει να παραδοθούν στους πελάτες. Το όχημα ξεκινά την διαδρομή του από την αποθήκη φορτωμένο με Qi στοιχεία του προϊόντος $i \in \{1,\ldots,k)$ και μετά την εξυπηρέτηση όλων των πελατών, επιστρέφει στην αποθήκη. Η ζήτηση του πελάτη j,j=1,2...,N για το προϊόν i,i=1,2,......, είναι διακριτή τυχαία μεταβλητή $\xi_i^j$, υποθέτουμε ότι η από κοινού κατανομή πιθανότητας των ζητήσεων του κάθε πελάτη είναι γνωστή. Η ζήτηση του κάθε πελάτη γίνεται γνωστή στην άφιξη του οχήματος στην τοποθεσία του πελάτη. Υποθέτουμε ότι η ζήτηση των πελατών για ένα συγκεκριμένο προϊόν δεν μπορεί να υπερβαίνει την χωρητικότητα που αντίστοιχου τμήματος στο όχημα $\max_{j=1,2,\ldots,N} \xi_i^j < O_i$ με $i \in \{1,\ldots,k\}$. Όταν το όχημα φτάσει στο πελάτη j για πρώτη φορά ικανοποιεί όση περισσότερη ζήτηση είναι δυνατόν. Εάν μέρος της ζήτησης για το προϊόν $i \in \{1,\ldots,k\}$ δεν είναι ικανοποιήσιμο, το όχημα γυρνάει στην αποθήκη, γεμίζει τα τμήματα με φορτία $Q_1,\ldots,Q_k$ και επιστρέφει στον πελάτη για να τον ικανοποιήσει. Μετά την εξυπηρέτηση και του τελευταίου πελάτη το όχημα επανέρχεται στην αποθήκη. Ορίζουμε $C_{j,j+1}$ J=1,2,.....,N το κόστος μετακίνησης μεταξύ των πελατών j και j+1 και $C_{j0}$ j=1,2,....,N το κόστος μετακίνησης μεταξύ πελάτη και αποθήκης. Άλλη μια υπόθεση που κάνουμε είναι ότι αυτά τα κόστη ικανοποιούν την τριγωνική ανισότητα.

$C_{i,i+1} \leq C_{i,0} + C_{0,i+1}$, i=1,......,N-1

Το οδικό δίκτυο απεικονίζεται στο παρακάτω σχήμα.




Σχήμα 2-1: Το οδικό δίκτυο

Ορίζουμε $z_i \in \{0,\ldots,Qj\}$,i=1,2,...,k το φορτίο του προϊόντος i στο όχημα μετά την ικανοποίηση της ζήτησης κάποιου πελάτη. Το όχημα είτε προχωράει άμεσα στον επόμενο πελάτη είτε πηγαίνει στην αποθήκη ανεφοδιάζεται με προϊόντα $Q_i$ i=1,2...,k και στην συνέχεια προχωράει προς τον επόμενο πελάτη. Στόχος είναι να καθοριστεί μια πολιτική δρομολόγησης που θα ελαχιστοποιεί το αναμενόμενο κόστος κατά την διάρκεια ενός κύκλου. Μια πρακτική εφαρμογή του προβλήματος είναι το λεγόμενο ex-van sales όπου ο οδηγός του οχήματος δρα ως πωλητής. Επισκέπτεται τους πελάτες (σουπερ μάρκετ, περίπτερα, κ.λ.π) συνήθως με μια προκαθορισμένη σειρά. Ορίζουμε τα διανύσματα $\bar{Z}=(z_1,z_2,...,z_k)$, $\bar{Q}=(Q_1,Q_2,...,Q_k)$, $\bar{\xi}_i^j=(\bar{\xi}_1^j,\bar{\xi}_2^j,...,\bar{\xi}_k^j)$ με $P^j(\bar{\xi})=\Pr(\bar{\xi}^j=\bar{\xi})$, $\bar{\xi}=(\xi_1,\xi_2,...,\xi_k)\in S=\{0,...,Q_1\}\times...\times\{0,...,Q_k\}$ την από κοινού συνάρτηση πυκνότητας πιθανότητας για την ζήτηση του πελάτη $j\in\{1,...,N\}$.

## 2.3 Διατύπωση των εξισώσεων

Έστω ότι $f_j(\bar{Z})$ είναι το ελάχιστο αναμενόμενο κόστος όταν το φορτίο του προϊόντος $i$ μεταφέρεται από το όχημα αφού έχει ικανοποιήσει την ζήτηση του πελάτη $j$ είναι $z_i$. Ακόμα ορίζουμε $\Xi_{\bar{Z}}=\{\xi:\xi_i<z_i,1<i<K\}$. Μια βέλτιστη στρατηγική δρομολόγησης μπορεί να προσδιοριστεί από τις παρακάτω εξισώσεις δυναμικού προγραμματισμού. Για $j$=1, 2, ..., N-1 έχουμε:

$$f_j(\bar{Z})=\min\{H_j(\bar{Z}),A_j\},\quad \bar{Z}\in\{0,...,Q_1\}\times...\times\{0,...,Q_k\} \tag{1}$$

Όπου

$$H_j(\bar{Z})=C_{j,j+1}+\sum_{\xi\in\Xi_{\bar{Z}}} f_{j+1}(\bar{Z}-\bar{\xi})P^{j+1}(\bar{\xi})+\sum_{\xi\notin\Xi_{\bar{Z}}}\left[2C_{j+1,0}+f_{j+1}\left(\bar{Q}+(\bar{Z}-\bar{\xi})^-\right)\right]P^{j+1}(\bar{\xi}) \tag{2}$$

$$A_j=C_{j,0}+C_{j+1,0}+\sum_{\xi\in S} f_{j+1}(\bar{Q}-\bar{\xi})P^{j+1}(\bar{\xi}) \tag{3}$$

Στο σύνορο έχουμε

$$f_N(\bar{Z})=C_{NO},\quad \bar{Z}\in S \tag{4}$$

Το ελάχιστο αναμενόμενο κόστος για ένα κύκλο είναι

$$f_0 = c_{10} + \sum_{\xi \in S} f_1\left(\bar{Q} - \bar{\xi}\right) P^1\left(\bar{\xi}\right) \tag{5}$$

Ο αριστερός όρος μέσα στις αγκύλες στο (1) αντιστοιχεί στην επιλογή του να προχωρήσουμε στο επόμενο πελάτη και ο δεξιός στην επιλογή του να επιστρέψουμε στην αποθήκη για ανεφοδιασμό. Στην εξίσωση (2) η ποσότητα $\left(\bar{Z} - \bar{\xi}\right)^{-}$ ορίζεται ως $\left(\min\left(z_1 - \xi_1, 0\right), \ldots, \min\left(z_k - \xi_k, 0\right)\right)$

# Κεφάλαιο 3 Αναλυτική Μέθοδος Εκτίμησης

Ο αλγόριθμος που χρησιμοποιούμε για τον προσδιορισμό της βέλτιστης διαδρομής βασίζεται στις εξισώσεις (1), (2), (3), (4).

## 3.1 Επεξήγηση του αλγορίθμου

Αρχίζοντας από τον N-1 πελάτη και φτάνοντας μέχρι τον πρώτο υπολογίζει το $A_j$ και το $H_j(\bar{z})$ σύμφωνα με τις εξισώσεις (3) και (2). Στην συνέχεια τα συγκρίνει για να δημιουργήσει τον πίνακα $f_j(\bar{z})$ σύμφωνα με την εξίσωση (1) και τον πίνακα $R_j(\bar{z})$ Τέλος, αφού έχει υπολογιστεί ο πίνακας $f_j$ για κάθε δυνατό απόθεμα $\bar{z}$ υπολογίζουμε το αναμενόμενο κόστος ενός κύκλου σύμφωνα με την εξίσωση (5).

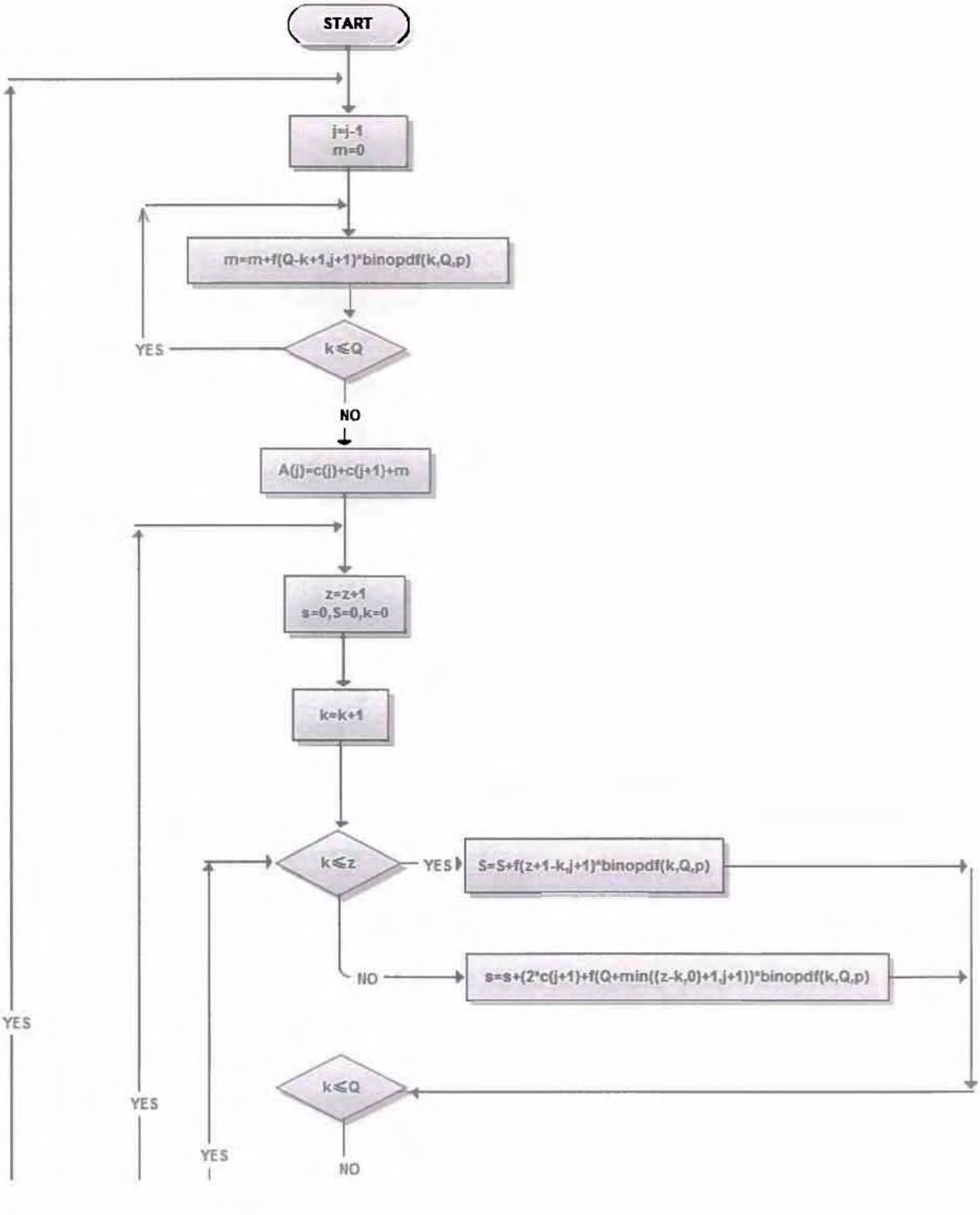
Ο πίνακας $R_j(\bar{z})$ περιέχει την πληροφορία για το τι πρέπει να κάνει ο οδηγός του οχήματος με απόθεμα $\bar{z}$ προτού επισκεφτεί τον επόμενο πελάτη. Για παράδειγμα, έστω ένα όχημα που έχει δύο τμήματα με χωρητικότητα 10 και 15 προϊόντων. Το όχημα πρέπει να ικανοποιήσει 15 πελάτες των οποίων οι ζητήσεις ακολουθούν διωνυμική κατανομή. Σε αυτήν την περίπτωση, ο πίνακας $R_j(\bar{z})$ είναι τρισδιάστατος με διαστάσεις 11×21×14. Υποθέτουμε ότι έχει ικανοποιηθεί η ζήτηση του 7$^{ου}$ πελάτη και ο οδηγός έχει απόθεμα 3 μονάδες από το προϊόν 1 και 12 μονάδες από το 2.

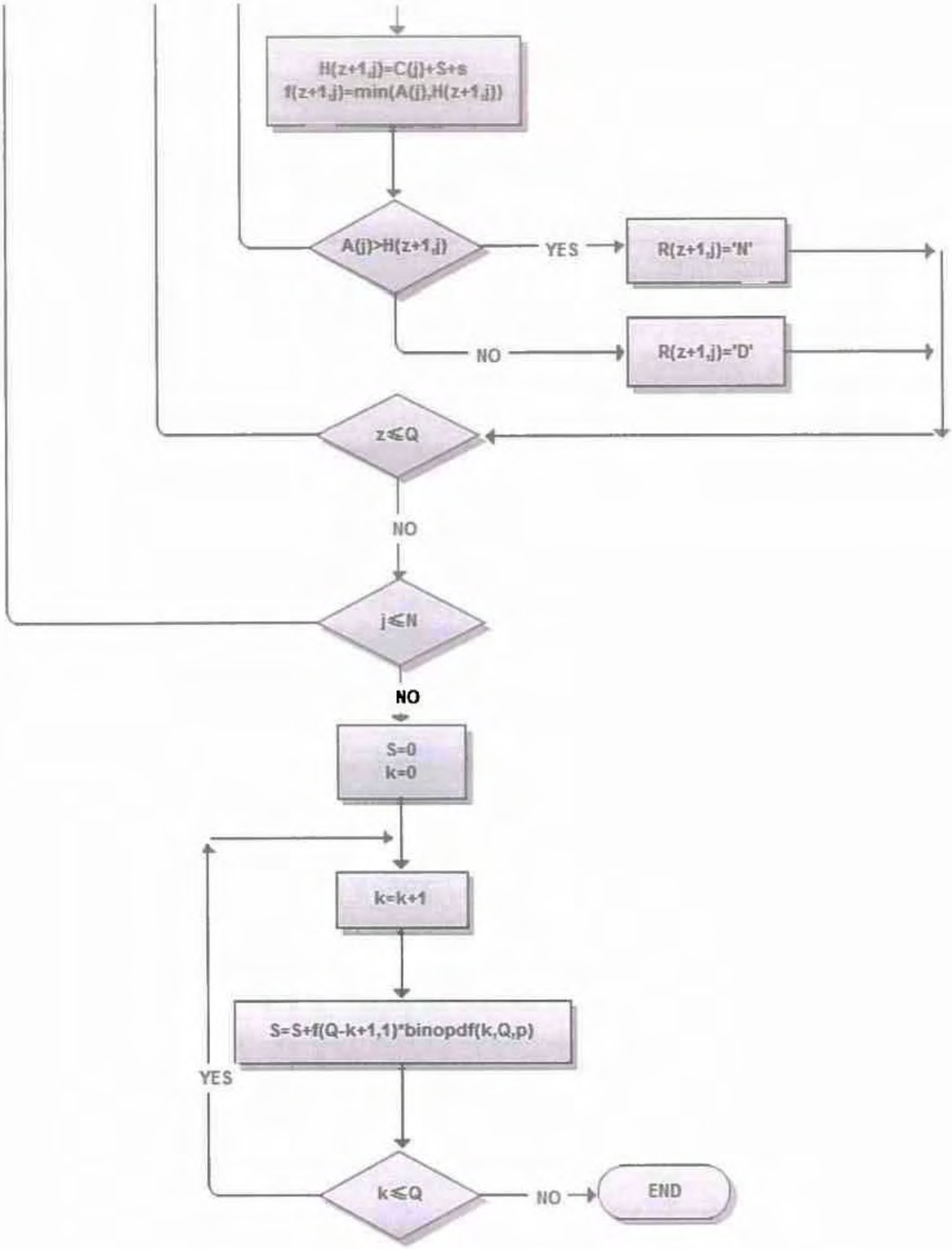
```
val(:,:,7) =

DDDDDDDDDDDDDDDDDDDDD
DDDDDDDDDDDDDDDDDDDDD
DDDDDDDDDDDNNNNNNNNNN
DDDDDDDDDDNNNNNNNNNNN
DDDDDDDDDNNNNNNNNNNNN
DDDDDDDDDNNNNNNNNNNNN
DDDDDDDDDNNNNNNNNNNNN
DDDDDDDDDNNNNNNNNNNNN
DDDDDDDDDNNNNNNNNNNNN
DDDDDDDDDNNNNNNNNNNNN
DDDDDDDDDNNNNNNNNNNNN
```

Σχήμα 3-:1 Ο Πίνακας R

Όπως φαίνεται από το παραπάνω σχήμα στην 7η φέτα του πίνακα $R$, ( $\bar{z}$ ),στην 4η γραμμή και στην 13η στήλη αντιστοιχεί το γράμμα N, γεγονός που σημαίνει ότι η βέλτιστη απόφαση για τον οδηγό είναι να προχωρήσει στον επόμενο πελάτη. Μια ακόμα χρήσιμη παρατήρηση είναι ότι η γραμμή και η στήλη 1 αντιστοιχούν σε μηδενικό απόθεμα.

## 3.2 Διάγραμμα ροής για k=1






Σχήμα 3-2: Διάγραμμα ροής για k=1

# Κεφάλαιο 4 Ευρετική Μέθοδος Εκτίμησης

Με τον όρο ευρετική εννοούμε εκείνη την μέθοδο επίλυσης, στην οποία η πορεία προς το τελικό αποτέλεσμα στηρίζεται σε μια σειρά προσεγγιστικών αποτελεσμάτων. Συνήθως, χρησιμοποιούμαι τέτοιες μεθόδους διότι απαιτούν λιγότερο υπολογιστικό φόρτο και κατά συνέπεια δίνουν λύσεις γρηγορότερα από τις αναλυτικές.

## 4.1 Η μέθοδος

Η ευρετική μέθοδος που χρησιμοποιούμε για την επίλυση του προβλήματος που μας απασχολεί αντιμετωπίζει τα δυο ή τρία είδη προϊόντων σαν ένα. Αυτό γίνεται προστίθοντας τις κατανομές των ζητήσεων των δύο ή τριών προϊόντων με την εξής τρόπο. Έστω ότι έχουμε 2 είδη προϊόντων με ζητήσεις $X_1$ και $X_2$ και με γνωστές κατανομές ζητήσεων. Θέλουμε να υπολογίσουμε ποια είναι η πιθανότητα να έχουμε συνολική ζήτηση X=3. Αυτό συμβαίνει όταν $P(X_1=0)$ και $P(X_2=3)$ ή $P(X_1=1)$ και $P(X_2=2)$ ή $P(X_1=2)$ και $P(X_2=1)$ ή $P(X_1=0)$ και $P(X_2=3)$. Όπως γνωρίζουμε από την στατιστική

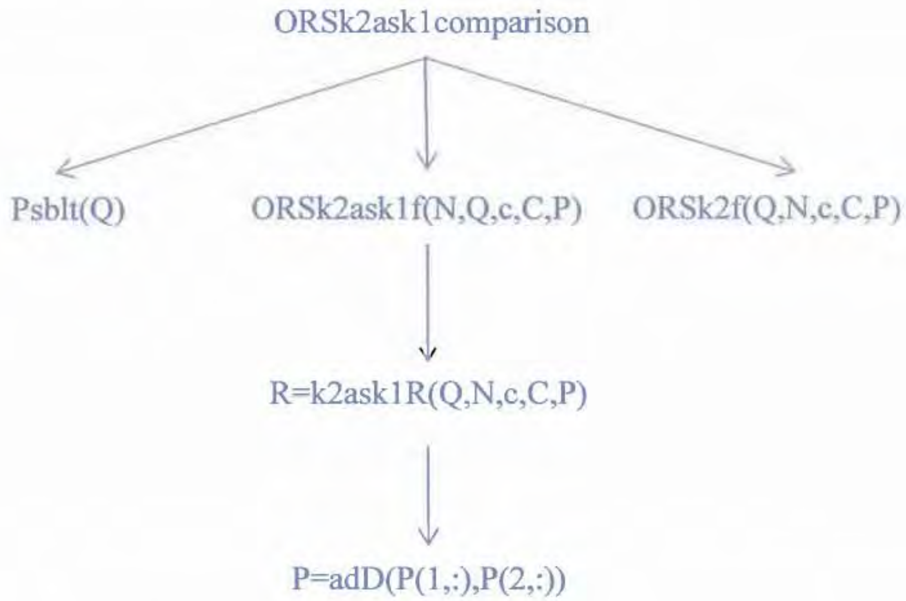
$$P(X=3) = P(X_1=0)*P(X_2=3) + P(X_1=1)*P(X_2=2) + P(X_1=2)*P(X_2=1) + P(X_1=\theta)*P(X_2=3).$$

Ανάλογα υπολογίζουμε και τις πιθανότητες για τις υπόλοιπες συνολικές ζητήσεις για k=2 και k=3.

Στην συνέχεια, σύμφωνα με το πίνακα των ολικών ζητήσεων υπολογίζεται ο πίνακας R και το αναμενόμενο κόστος, αν ακολουθούσαμε την στρατηγική που είναι αποτυπωμένη σε αυτόν. Όπως θα περιμέναμε, η μέθοδος αυτή είναι πιο γρήγορη αλλά υστερεί σε ακρίβεια.

## 4.2 Ανάλυση του αλγόριθμου

Παρακάτω παραθέτουμε ένα σχήμα το οποίο αναπαριστά την δομή του αλγόριθμου ORSk2ask1comparison, ο οποίος συγκρίνει την ευρετική και την αναλυτική λύση και εξηγούμε ποιος είναι ο ρόλος των συναρτήσεων.




Σχήμα 4-1: Αναπαράσταση του ORSk2ask1 comparison

- Psblt: προτρέπει το χρήστη να γεμίσει τον πίνακα P με τις πιθανότητες των ζητήσεων για τα 2 προϊόντα
- adD : προσθέτει τις κατανομές των ζητήσεων των 2 προϊόντων με τον τρόπο που περιγράψαμε παραπάνω
- k2ask1R: υπολογίζει τον πίνακα R σύμφωνα με τον πίνακα των πιθανοτήτων των συνολικών ζητήσεων
- ORSk2ask1f: υπολογίζει ποιο είναι το μέσο αναμενόμενο κόστος αν το όχημα κινείται σύμφωνα με τον πίνακα R

- ORSk2f: υπολογίζει ποιο είναι το αναμενόμενο κόστος σύμφωνα με την αναλυτική μέθοδο.

Ίδια ακριβώς είναι και η δομή του ORSk3ask1 comparison που ακολουθεί παρόμοια διαδικασία για 3 είδη προϊόντων.

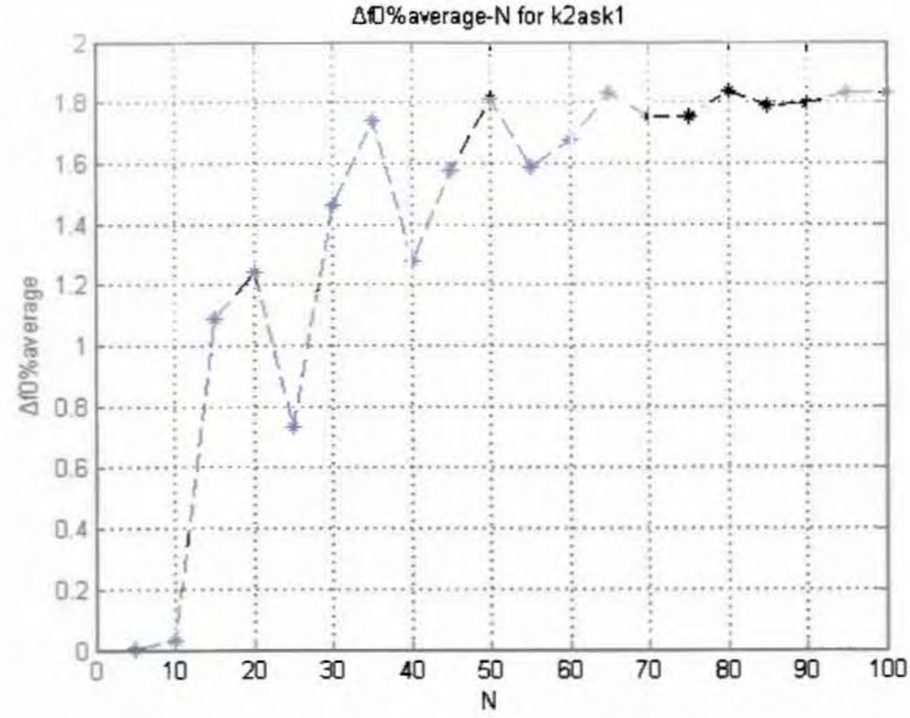
# Κεφάλαιο 5 Σύγκριση ευρετικής και αναλυτικής μεθόδου

Σε αυτό το κεφάλαιο θα αναφερθούμε στα αριθμητικά αποτελέσματα από την σύγκριση των δύο μεθόδων.

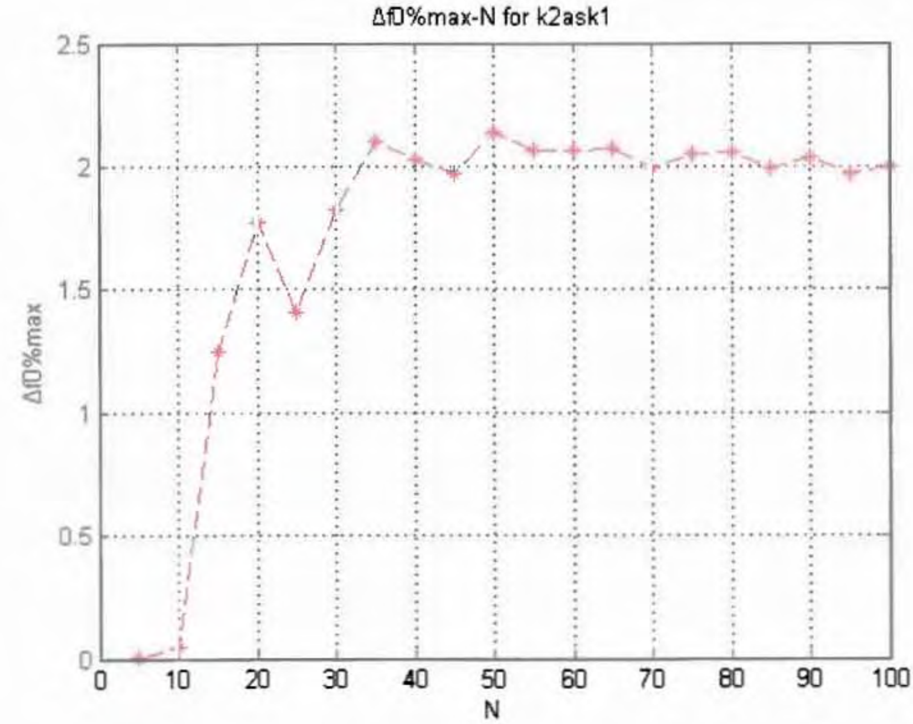
## 5.1 Σύγκριση ως προς την ακρίβεια

Παρακάτω θα παρουσιάσουμε γραφικές παραστάσεις που παρουσιάζουν την ποσοστιαία απόκλιση της λύσης της ευρετικής μεθόδου από την αναλυτική συναρτήσει των μεταβλητών του προβλήματος. Οι παρακάτω γραφικές παραστάσεις σχεδιάστηκαν με τον εξής τρόπο. Δημιουργούμε με την γεννήτρια τυχαίων ακεραίων αριθμών 30 διαφορετικά σύνολα αποστάσεων και στην συνέχεια υπολογίζουμε την ποσοστιαία διαφορά των δύο μεθόδων για αυτά συναρτήσει του φορτίου Q, των πελατών N, καθώς και του παράγοντα p της δυωνιμικής κατανομής. Τέλος, βρίσκουμε ποια είναι η μέγιστη διαφορά καθώς και ο μέσος όρος τον διαφορών και τα παρουσιάζουμε στα διαγράμματα. Έχουμε υποθέσει ότι οι πιθανότητες των ζητήσεων για κάθε προϊόν δίνονται από την διωνυμική κατανομή. Παρακάτω παραθέτουμε τα σχετικά διαγράμματα.

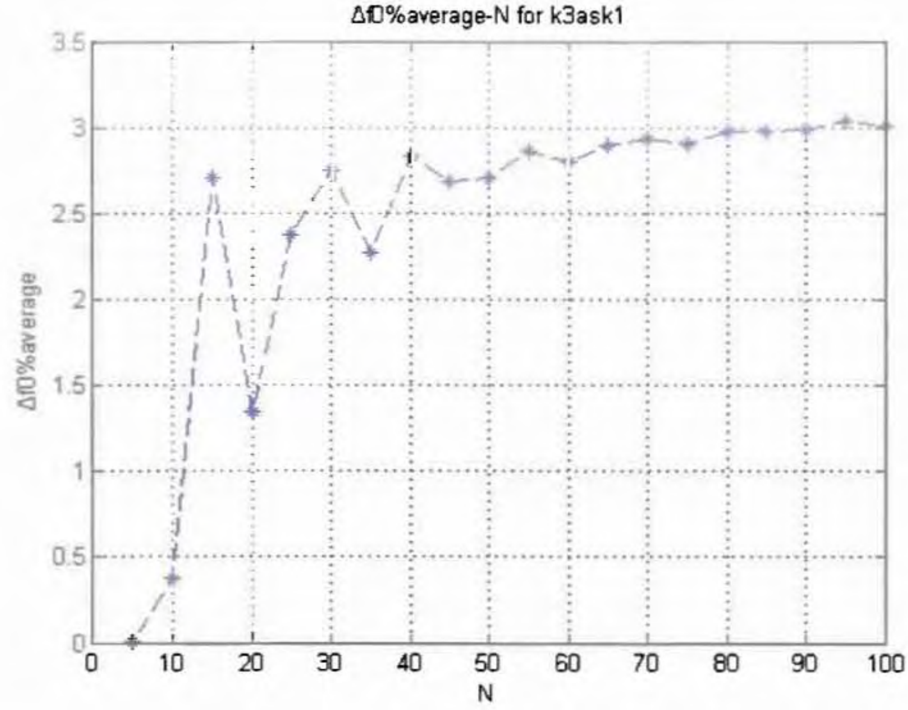
Όπως βλέπουμε από τα διαγράμματα η συμπεριφορά που ακολουθεί το Δf0% ως προς N είναι η ίδια για k=2 και k=3. Μέχρι να φτάσουμε στον αριθμό των 60-70 πελατών υπάρχει μια ανοδική συμπεριφορά αλλά με αυξομειώσεις. Έπειτα η απόκλιση σταθεροποιείται περίπου στο 1.8% για k=2 και στο 3% για k=3. Όσο αναφορά τα μέγιστα φτάνουν στο 2% και το 3% για k=2 και k=3 αντίστοιχα.



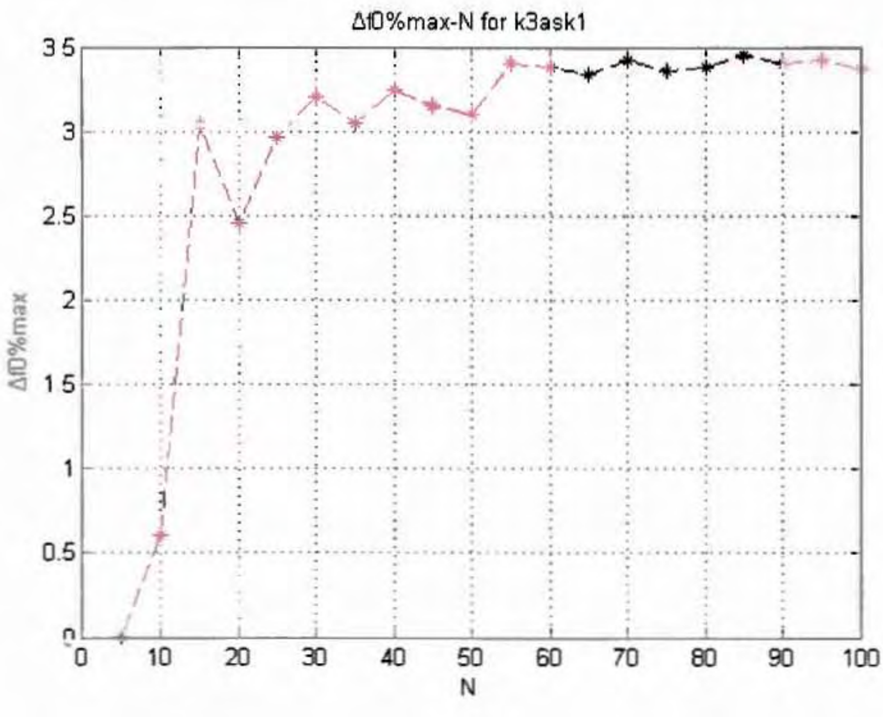

Σχήμα 5-1: Μέση τιμή του Δf0%-N για k=2




Σχήμα 5-2: Μέγιστο του Δf0%-N για k=2

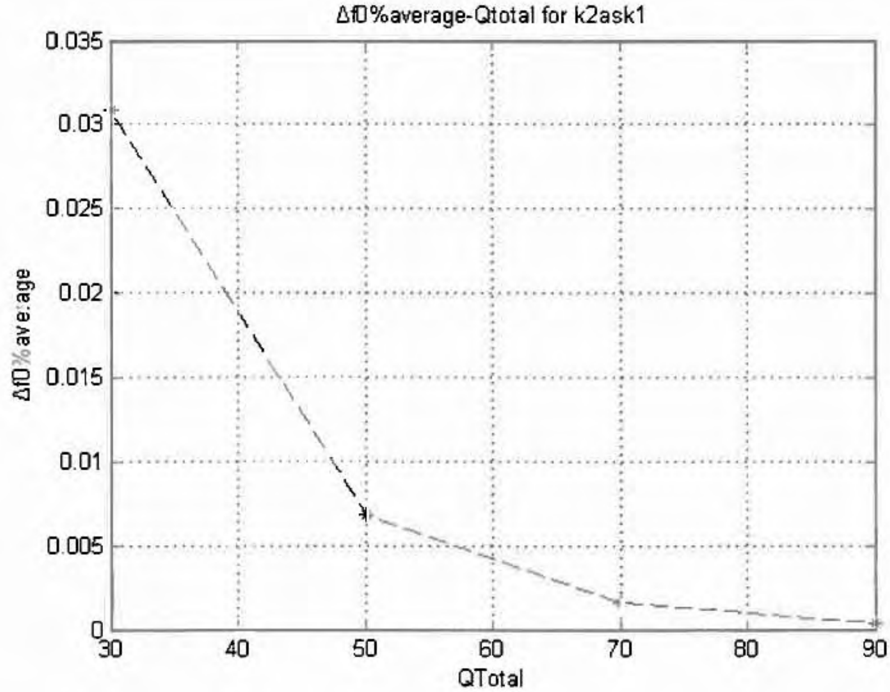



Σχήμα 5-3: Μέση τιμή του Δf0%-N για k=3

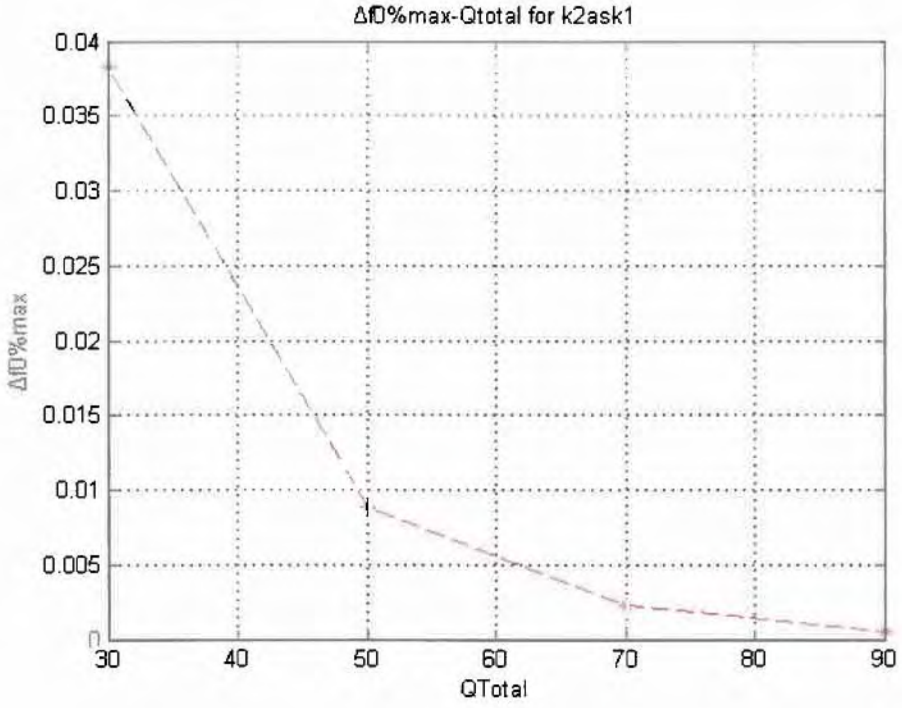



Σχήμα 5-4: Μέγιστο του Δf0%-N για k=3

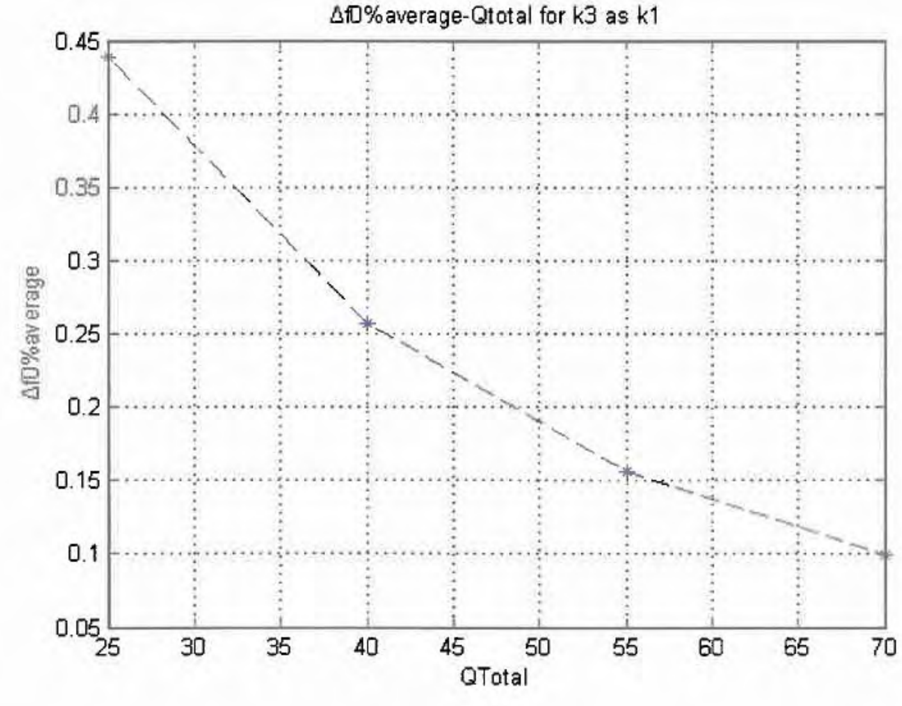
Στα επόμενα 4 διαγράμματα βλέπουμε την συμπεριφορά που παρουσιάζει το Δf0% συναρτήσει του ολικού φορτίου Q. Οι γραφικές παραστάσεις θυμίζουν υπερβολή. Για το ελάχιστο αριθμό πελατών παίρνουν την μέγιστη τιμή τους και σε μεγάλους αριθμούς πελατών το σφάλμα σχεδόν μηδενίζεται. Μεγαλύτερες φυσικά είναι οι τιμές για k=3.



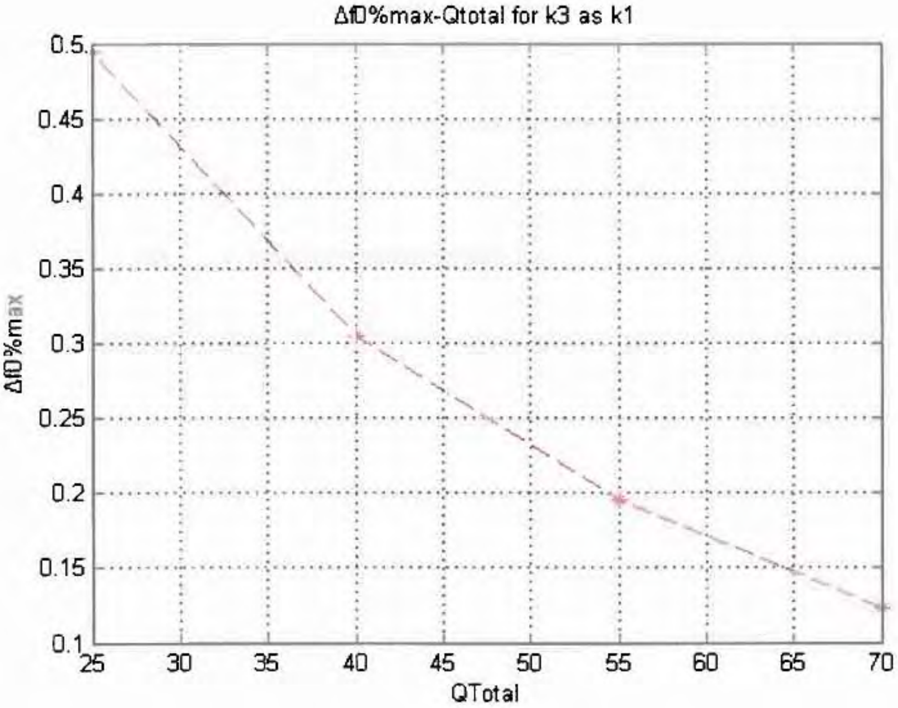

Σχήμα 5-5: Μέση τιμή του Δf0%-Q για k=2




Σχήμα 5-6: Μέγιστο του Δf0%-Q για k=2

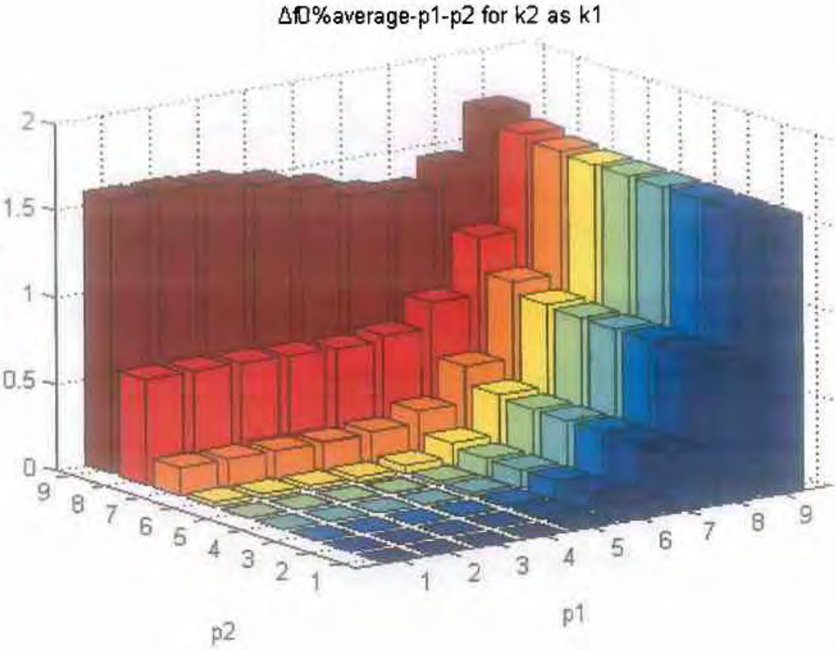



Σχήμα 5-7: Μέση τιμή του Δf0%-Q για k=3

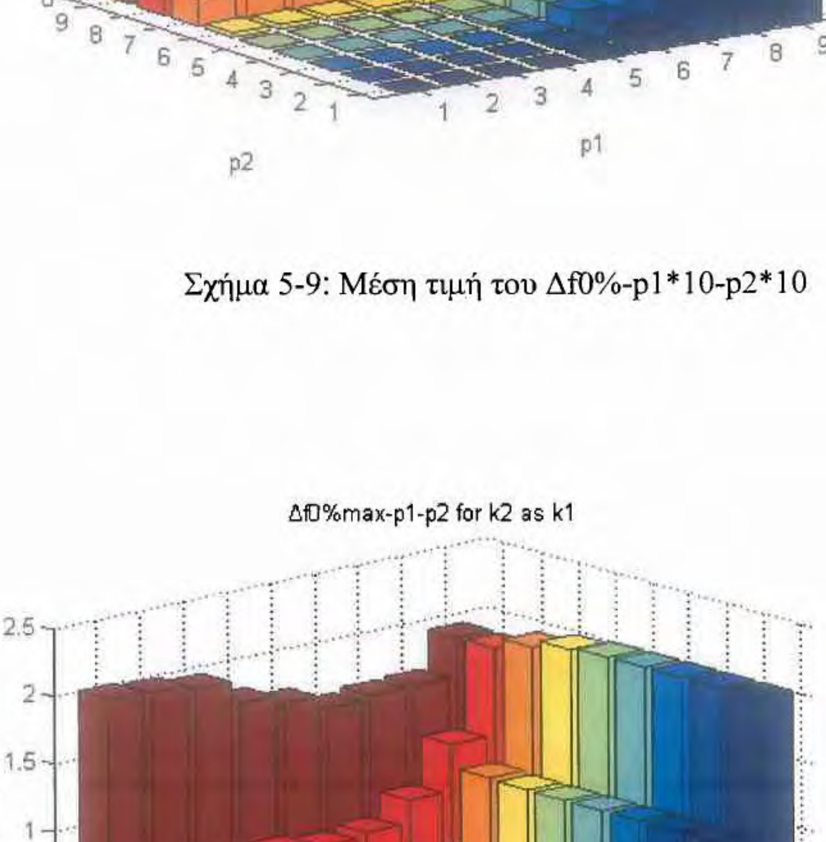



Σχήμα 5-8: Μέγιστο του Δf0%-Q για k=3

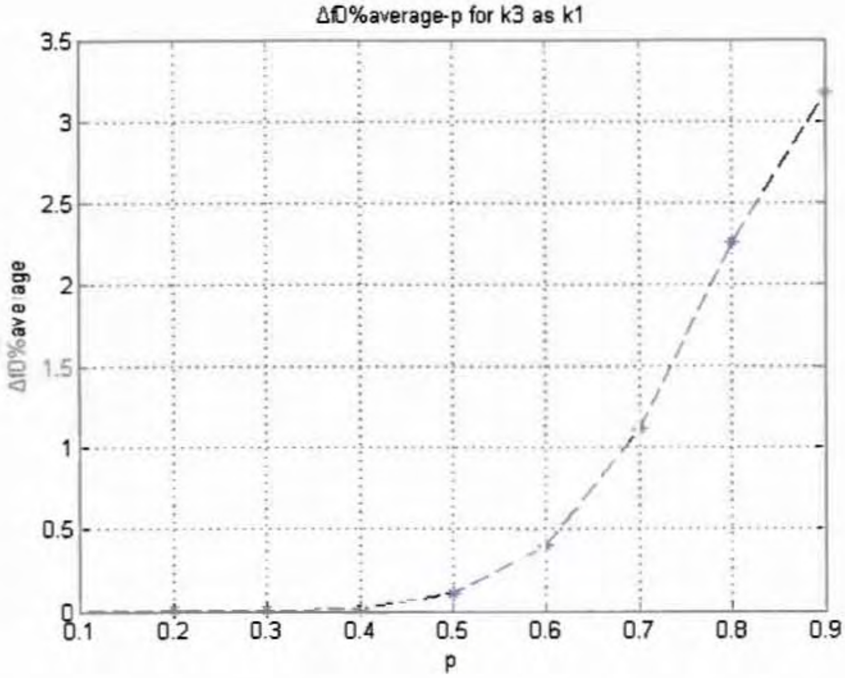
Παρακάτο στα δύο πρώτα διαγράμματα βλέπουμε ποιο είναι το σφάλμα σε συνάρτηση με το p1 και p2. Παρατηρούμε ότι όσο αυξάνονται τα p το ποσοστιαίο σφάλμα αυξάνεται. Ενώ στα επόμενα δύο διαγράμματα βλέπουμε ποιο είναι το σφάλμα για k=3 και ταυτόχρονη αύξηση του p1,p2 και p3. Τέλος στο παράρτημα παρατίθονται δύο πίνακες, ο πίνακας που μας δίνει τον μέσο όρο της απόκλισης είναι πρώτος ενώ ακολουθεί ο πίνακας με τα μέγιστα. Όπως περιμέναμε, για k=2 και k=3 έχουμε την ίδια συμπεριφορά με υψηλότερες τιμές λαθών για k=3.



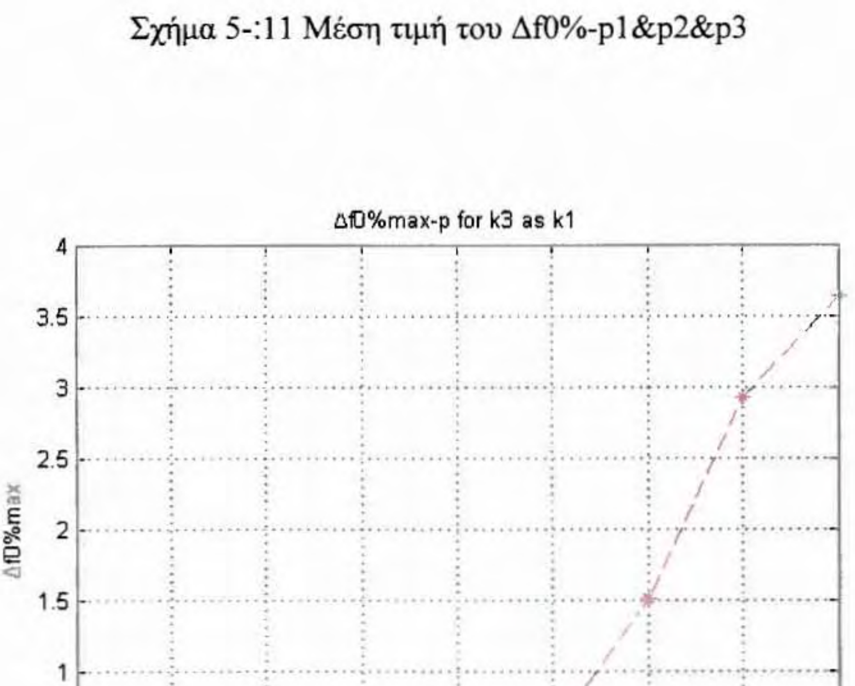

Σχήμα 5-9: Μέση τιμή του Δf0%-p1*10-p2*10




Σχήμα 5-:10 Μέγιστο του Δf0%-p1*10-p2*10




Σχήμα 5-:11 Μέση τιμή του Δf0%-p1&p2&p3




Σχήμα 5-12: Μέγιστο του Δf0%-p1&p2&p3

## 5.2 Σύγκριση ως προς την ταχύτητα

Για να συγκρίνουμε τις ταχύτητες εκτέλεσης των δύο μεθόδων χρησιμοποιήσαμε ένα εργαλείο του matlab το οποίο ονομάζεται profiler. Το συγκεκριμένο εργαλείο δίνει στο χρήστη πληροφορίες όπως ποιός είναι ο χρόνος που καταναλώνεται από κάθε μια από τις εντολές ενός κώδικα ή πόσες φορές καλέστηκε μια συνάρτηση κ.ά. Σκοπός του profiler είναι να βοηθήσει το χρήστη να αποφύγει δαπανηρές λειτουργίες και περιττούς υπολογισμούς και σαν αποτέλεσμα να κάνει τον κώδικα του πιο αποτελεσματικό.

Εκτελέσαμε τους κώδικες k2ask1comparison και k3ask1comparison μέσο του profiler για να προσδιορίσουμε πόσο χρόνο καταναλώνουν οι συναρτήσεις που καλούνται και μας δίνουν την αναλυτική και την ευρετική λύση. Έτσι μπορέσαμε να κάνουμε μια εκτίμηση για την ταχύτητα εκτέλεσης της μιας μεθόδου συγκριτικά με την άλλη. Παρακάτω βλέπουμε τα αποτελέσματα.

| Function Name | Calls | Total Time | Self Time* | Total Time Plot (dark band = self time) |
|---|---|---|---|---|
| ORSk2ask1comparison | 1 | 1.397 s | 0.008 s | |
| ORSk2ask1f | 1 | 0.645 s | 0.636 s | |
| ORSk2f | 1 | 0.745 s | 0.745 s | |
| adD | 1 | 0.001 s | 0.001 s | |
| k2ask1R | 1 | 0.009 s | 0.008 s | |

Σχήμα 5-13: Αποτελέσματα profiler για k2ask1comparison

| Function Name | Calls | Total Time | Self Time* | Total Time Plot (dark band = self time) |
|---|---|---|---|---|
| ORSk3ask1comparison | 1 | 974.962 s | 0.008 s | |
| ORSk3ask1f | 1 | 463.646 s | 463.618 s | |
| ORSk3f | 1 | 511.308 s | 511.308 s | |
| adD3 | 1 | 0.008 s | 0.008 s | |
| k3ask1R | 1 | 0.028 s | 0.020 s | |

Σχήμα 5-14: Αποτελέσματα profiler για k3ask1comparison

Οι συναρτήσεις ORSk2ask1f και ORSk3ask1f υπολογίζουν την ευρετική λύση ενώ οι ORSk3f και ORSk2f την αναλυτική. Όπως περιμέναμε η αναλυτική μέθοδος κάνει περισσότερο χρόνο για να δώσει λύση απ'ότι η ευρετική. Για k=2 η ευρετική είναι κατά 15,5% γρηγορότερη από την αναλυτική ενώ για k=3 κατά 10,3%.

# Κεφάλαιο 6 Σύνοψη και Μελλοντικές Εργασίες

Σε αυτήν την εργασία μελετήσαμε εκτενώς μια παραλλαγή του προβλήματος δρομολόγησης οχημάτων. Είδαμε ποιές είναι οι εξισώσεις που το διέπουν και αναπτύξαμε πάνω σε αυτές τον κώδικα που δίνει την αναλυτική λύση. Έπειτα, αναπτύξαμε μια ευρετική μέθοδο επίλυσης η οποία παρουσιάζει αποκλίσεις σε σχέση με την αναλυτική όμως λύνει το πρόβλημά μας γρηγορότερα. Τέλος, κάναμε συγκρίσεις των δύο μεθόδων ως προς την ταχύτητα και την ακρίβεια και αποτυπώσαμε τα αποτελέσματα σε διαγράμματα.

Μερικά σχετικά θέματα τα οποία θα μπορούσαν να αποτελέσουν αντικείμενο περαιτέρω εργασίας είναι:

- Γενίκευση του κώδικα εύρεσης της αναλυτικής λύσης για k είδη προϊόντων
- Γενίκευση του κώδικα εύρεσης της ευρετικής λύσης για k είδη προϊόντων
- Σύγκριση των 2 μεθόδων για περισσότερα ήδη προϊόντων

# Βιβλιογραφία


[1] D.G.Pandelis, E.G Kyriakidis, T.D. Dimitrakos "Single vehicle routing problems with a predefined customer sequence, compartmentalizes load and stochastic demands", Elsevier 2011

[2] Hamdy A. Taha "Εισαγωγή στην Επιχειρησιακή Έρευνα", 9η έκδοση, Εκδόσεις Τζιόλα, Αθήνα 2012

[3] Ν. Δ. Τσαντάς, Π. Γ. Βασιλείου, "Εισαγωγή στην Επιχειρησιακή Έρευνα", 1η Έκδοση, Εκδόσεις Ζήτη Θεσσαλονίκη 2001

[4] F. S. Hillier, Gerald J. Lieberman, "Introduction to Operations Research", 7th edition, McGraw Hill, New York 2001

[5] Stormy Attaway, "MATLAB: A practical introduction to programming and problem solving", Second Edition , Elsevier Science and Technology, USA 2011

[6] A.Biran, M.Breiner, "MATLAB 6: Για μηχανικούς", 3η Έκδοση, Εκδόσεις Τζιόλα Θεσσαλονίκη 2003

[7] www.mathworks.com

[8] www.gliffy.com

# Παράρτημα

Παρακάτω υπάρχουν οι δυο πίνακες που αναφέραμε στο κεφάλαιο 5.1 και ο κώδικας MATLAB όλων τον εφαρμογών που χρησιμοποιήθηκαν για την ολοκλήρωση της διπλωματικής εργασίας.

```
G=sum(DF,4)/5

G(:,:,1) =

  Column 1

     8.66967089313903e-013
     6.63965241346711e-006
      0.000848838343689293
        0.0129783671123102
        0.0626501438748371
         0.204751428859847
         0.453129195334219
         0.808973864720106
          1.26542124827406

  Column 2

     7.67391271412567e-007
     7.41467640679065e-006
      0.000849722125380969
        0.0129796834331766
        0.0626525497748384
         0.204755096747503
         0.453134050583087
         0.808980031920564
          1.26542918825692

  Column 3

      0.000190240682293875
      0.000197223106531576
       0.00104369053183478
        0.0131897041737157
        0.0629029375438744
         0.205048182006443
```

```
      0.453464185762579
      0.809343861921195
       1.26584667183368

Column 4

     0.0035602141328008
    0.00356942161739597
    0.00445306640050146
     0.0167485010507226
     0.0668011163407024
       0.20925858586588
      0.457897392259359
       0.81402228662907
       1.27101015231029

Column 5

     0.0252131420831772
     0.0252427194947908
     0.0263379798850439
     0.0392905372296161
     0.0907775945257011
      0.234509408236977
      0.484271698881021
      0.841383037769487
        1.3001247523388

Column 6

      0.115818091276002
      0.115903686322838
      0.117489533461926
      0.131948089101253
      0.186590008799572
       0.33254820714403
      0.583853818139615
      0.941941803391434
        1.4022871497926

Column 7

      0.356710264541332
      0.356915931690014
      0.359475857071452
      0.376881739484101
      0.436757122896721
      0.585359883162562
      0.835687539849624
       1.19174968014839
       1.64779103991418

Column 8

      0.855829908675294
      0.856150544884543
      0.859603812536529
      0.879299153124425
      0.942064923460029
```

```
          1.08696211058595
          1.32818966609218
            1.668070865546
          2.10370706287797

  Column 9

           1.6256647101132
           1.6260659077185
          1.63014733576197
          1.65036092756285
          1.71121841594786
          1.84555367957782
          2.06609758453123
          2.37216842303167
          2.76022262277317


G(:,:,2) =

  Column 1

      2.36976592847261e-008
      6.66340419711148e-006
       0.000848863214397773
         0.0129783971598504
         0.0626501876935892
           0.20475148962873
          0.453129268961353
          0.808973949999729
           1.26542135202929

  Column 2

      7.91089765851562e-007
      7.43842975448875e-006
       0.000849746997798534
         0.0129797134824143
         0.0626525935955387
          0.204755157517263
          0.453134124210886
          0.808980117200446
            1.2654292920126

  Column 3

       0.000190264469447845
       0.000197246948570758
        0.00104371549201762
          0.013189734308057
         0.0629029814593283
          0.205048242835681
           0.45346425943439
          0.809343947230407
           1.26584677562806

  Column 4

         0.0035602385803499
```

```
    0.00356944611834493
    0.00445309201777875
     0.0167485318165463
      0.066801160893909
      0.209258647108924
      0.457897466237989
      0.814022372016419
       1.27101025612961

Column 5

     0.0252131706692779
     0.0252427481280874
     0.0263380096017377
     0.0392905719579831
     0.0907776430520279
      0.234509471882168
      0.484271774101086
      0.841383123580278
       1.30012485240961

Column 6

      0.115818129973187
      0.115903725052734
      0.117489573248996
       0.13194813312135
      0.186590064879249
      0.332548276294299
      0.583853889205516
      0.941941881679514
       1.40228724306545

Column 7

      0.356710329942337
      0.356915996908444
       0.35947592234498
      0.376881807092457
      0.436757192050251
      0.585359954449387
       0.83568761153652
       1.19174973681041
        1.6477910980386

Column 8

        0.8558300007506
      0.856150636450775
      0.859603900350266
      0.879299236365863
      0.942065003543971
       1.08696216925291
       1.32818970546077
       1.66807082499431
       2.10370698845197

Column 9
```

```
          1.62566481902496
          1.62606601478267
          1.63014743743326
          1.65036101313473
          1.71121848075841
           1.8455536655344
           2.0660975020371
           2.3721681378482
          2.76022222014046


G(:,:,3) =

  Column 1

     1.01020904804783e-005
     1.67542753321299e-005
      0.000859198739187679
        0.0129897413437682
         0.062663972674127
         0.204767951659063
          0.45314778056363
         0.808994651788279
          1.26544582038733

  Column 2

     1.08698334172207e-005
     1.75296510662574e-005
      0.000860082869082813
        0.0129910579948706
         0.062666378962966
         0.204771619777585
         0.453152635997852
         0.809000819126316
          1.26545376035057

  Column 3

      0.000200358785079648
      0.000207353713329617
        0.0010540668044888
        0.0132010937789638
        0.0629167827513497
          0.20506471596105
         0.453482777333871
         0.809364650769551
          1.26587124674506

  Column 4

       0.00357045371365463
       0.00357967360051479
       0.00446356406758603
        0.0167600075279871
        0.0668150700509516
         0.209275183640532
         0.457916031875637
         0.814043055736692
```

```
         1.27103470732607

Column 5

       0.0252241686925217
       0.0252537578625292
       0.0263492508653062
        0.039302772817423
       0.0907922497534155
        0.234526387601163
        0.484290565877011
           0.841403910942
         1.30014907830526

Column 6

        0.115831096850783
        0.115916691585369
        0.117502741999693
         0.13196202690924
        0.186606076353861
        0.332566265021387
        0.583872853927125
         0.94196242662542
         1.40231026026585

Column 7

        0.356727844450014
        0.356933448088232
         0.35949339438224
        0.376899710671431
        0.436776040016608
        0.585379484046808
         0.83570652660934
         1.19176848509991
         1.64780819594355

Column 8

        0.855851905034154
        0.856172419614243
        0.859625299266751
        0.879320249666862
        0.942085158318283
         1.08698084196113
         1.32820289653251
         1.66808133894621
         2.10371621901364

Column 9

         1.62568922417467
         1.62609015536896
         1.63017051727789
         1.65038240597876
         1.71123512805923
         1.84556586894377
         2.06610398107003
```

```
         2.37215483949426
         2.76019971579981


G(:,:,4) =

  Column 1

     0.000547104775804712
     0.000554364091362807
      0.00140404293749538
       0.0135609837252916
       0.0632987517983889
        0.205465724011985
        0.453899353142597
        0.809798956865398
         1.2663420871133

  Column 2

     0.000547884389806176
     0.000555151307710811
      0.00140493880942679
       0.0135623116527406
       0.0633011693802321
        0.205469398123521
        0.453904209715368
        0.809805122606924
         1.26635000900855

  Column 3

     0.000737854684341659
     0.000745455836888818
      0.00159940091271362
       0.0137727979322687
        0.063551986308721
        0.205762792604309
        0.454234384813494
        0.810168872683293
         1.26676736569446

  Column 4

       0.0041119404222983
      0.00412174850779452
      0.00501281022587814
       0.0173354274922133
       0.0674531978099938
        0.209975038316661
        0.458668710313328
        0.814846889191301
         1.27192986814401

  Column 5

       0.0257875346416151
        0.025817627552931
       0.0269194940276006
```

```
        0.0398975816526605
        0.0914479286362563
         0.235236774643875
         0.485049226761677
         0.842206754952675
          1.30103905892621

  Column 6

         0.116443019875838
         0.116528538033015
         0.118119747915793
         0.132598197810103
         0.187294864985976
         0.333296249248488
          0.58463736934953
         0.942746340393876
          1.40316645294479

  Column 7

         0.357451225859967
         0.357655259146504
         0.360215613286651
         0.377628169749337
          0.43753514896257
         0.586133140277055
         0.836465955555377
          1.19252968653803
          1.64851890436321

  Column 8

         0.856671786116594
         0.856987201909736
         0.860431459022865
         0.880102294304919
         0.942860844963158
          1.08772846925278
           1.3288145405165
           1.6686206340726
          2.10403185077155

  Column 9

          1.62655328297609
          1.62694148147238
          1.63099739647112
          1.65117184787571
          1.71190466341913
          1.84612182615406
          2.06633556167981
          2.37221871710346
          2.76013059089938


G(:,:,5) =

  Column 1
```

```
        0.00904035792410375
        0.00905379425768191
        0.00996958222615741
         0.0223571875882166
         0.0726029478992794
          0.215201315311354
          0.463944496461531
          0.820132183780547
           1.27715151060078

Column 2

         0.0090412641114824
        0.00905470764689853
        0.00997059945513278
         0.0223586221631594
         0.0726054563315957
          0.215204974480533
          0.463949277147627
          0.820138208238517
           1.27715919614088

Column 3

        0.00923581705734503
        0.00924956082017703
          0.010169500840794
          0.022572908225649
         0.0728593329023329
          0.215499995212967
          0.464278885564936
          0.820499748382909
             1.277572206928

Column 4

         0.0126471147136529
         0.0126625591673585
         0.0136182408658919
         0.0261676368397069
         0.0767841799901573
          0.219723817776997
          0.468711475116553
          0.825167536610931
            1.2827012745169

Column 5

         0.0345075202644686
         0.0345415591671525
         0.0356994517629946
         0.0488860009387286
          0.100901507168008
          0.245061426220713
          0.495088450860225
          0.852473173764977
           1.31173427615635
```

```
  Column 6

        0.125576752786501
         0.12565990611317
        0.127282766062346
        0.141924076190445
        0.196961730288156
        0.343216573836832
        0.594691428471053
        0.952685307968889
          1.4133501776146

  Column 7

        0.367363789438395
        0.367548535814531
        0.370065407819389
        0.387521927947136
         0.44761933168328
        0.595944065607843
        0.846080554534843
         1.20141129850743
         1.65739695180069

  Column 8

        0.867059539397693
        0.867315778847527
        0.870658978227863
        0.890009035457665
        0.952544053775236
         1.09644788772009
         1.33697852705244
         1.67617860092629
         2.11108732750284

  Column 9

         1.63691088766332
         1.63713233340468
          1.6408823407118
         1.66022591397407
         1.72030988241749
         1.85363653969078
         2.07275436764018
         2.37753630261493
         2.76393770651923


G(:,:,6) =

  Column 1

       0.0676491592756061
       0.0676799942282329
       0.0688138754048269
       0.0818894062891192
         0.13362747549964
        0.277110194554249
```

0.526054515873356
0.88197992728434
1.33905984070656

Column 2

0.0676504568193432
0.0676812910642842
0.0688152110138462
0.0818910375818449
0.133629991970419
0.27711344184686
0.52605848786968
0.881983955594474
1.33906500276263

Column 3

0.0678591278449503
0.0678899239100489
0.0690269065501692
0.0821148869633831
0.133888762697843
0.277402634065431
0.526374005502201
0.882309874711529
1.33943562004021

Column 4

0.0713839201682434
0.0714135609499047
0.0725775598927057
0.0857851865740982
0.137855394282745
0.281579322506426
0.530704895606634
0.886843639128333
1.3442346054679

Column 5

0.0937932469477991
0.0938245202010405
0.0951511849054195
0.108860875760865
0.162177535902474
0.306956505117952
0.556594800742553
0.913355404963639
1.37185534850634

Column 6

0.185979602167998
0.185992242857589
0.187651157144301
0.202635764334188
0.25818419744711

```
        0.404369625045151
        0.654280472203157
         1.01130534045926
          1.4711385987892

  Column 7

        0.429541521304635
        0.429574059155342
        0.431613011889404
         0.44868287634078
         0.50772792658373
        0.655058254130346
        0.903424774241033
          1.2566595873439
          1.7107325792035

  Column 8

        0.929078906165433
        0.928861004236012
        0.930720225182582
        0.948805053446698
         1.00984821874317
         1.15143788029381
         1.38896133405444
         1.72382905757836
          2.1509526752398

  Column 9

         1.69580304689224
         1.69507394707494
         1.69720731485301
         1.71434980990205
         1.77171895505597
         1.90045491076268
         2.11017180519234
         2.40349311495545
         2.78431336443519


G(:,:,7) =

  Column 1

        0.272918425041783
         0.27298941718712
        0.274608886316961
         0.28931416881973
        0.344363200053727
        0.488459596755311
        0.735631575547023
         1.08852231475349
         1.54343900868837

  Column 2

        0.272920554498547
```

```
   0.27299149353198
  0.274610795119623
   0.28931592039982
   0.34436414532668
  0.488459642634076
  0.735627306674124
    1.0885158392768
   1.54343115528849

Column 3

  0.273159352893856
  0.273229001997647
  0.274845114754112
  0.289552296179852
  0.344601279184885
  0.488698812025743
  0.735866445776324
   1.08865939666206
   1.54356477044866

Column 4

  0.276922250977952
  0.276978618843617
  0.278566177638559
  0.293313998755683
   0.34854475322907
  0.492471476681367
    0.7395475765614
   1.09188959441502
   1.54692654712164

Column 5

  0.300542446622066
  0.300532177813568
  0.302148039734806
  0.316810363353878
  0.372588562534876
  0.516167598200135
  0.763245224994793
   1.11599036818205
   1.57143762157672

Column 6

  0.394595011526477
  0.394237278986704
  0.395563710908426
  0.409838378438942
  0.466658756796999
   0.61097771995975
  0.857011303701334
   1.20864244720501
   1.66247010093768

Column 7
```

```
          0.640352970962753
           0.63892634248632
          0.639559288488501
          0.655075980087592
          0.713078194232813
          0.856074051377804
            1.0968358842548
           1.43553981788414
           1.87385923124589

  Column 8

           1.13548073322488
           1.13346947859678
           1.13270908190886
           1.14723887436631
           1.20280649718837
           1.32989250707216
           1.54815469706215
             1.870432025217
            2.2767169112209

  Column 9

           1.89022986905174
           1.88686452386367
           1.88371831489782
           1.88894734286857
           1.92868924136847
           2.04393277062197
           2.22954619512101
           2.49233304716539
           2.83699832244434


G(:,:,8) =

  Column 1

          0.765140732500557
           0.76532153940049
          0.767906701899278
          0.785402376786616
          0.844740565123769
          0.987747589383772
           1.22759046142668
           1.56765282708353
           2.00635644183974

  Column 2

          0.765145068697623
          0.765325500111154
          0.767909613836521
          0.785401445973545
          0.844734977800391
          0.987735553101007
           1.22755601942978
           1.56760283191191
```

2.0062339277506

Column 3

0.765452273116846
0.765623041093071
0.768186567072412
0.785647813139797
0.844866294699833
0.987754923244713
1.22720834837246
1.56706551525322
2.00561397206632

Column 4

0.769677535452866
0.769795409818711
0.772134146613027
0.78933237592517
0.847846867942886
0.990129876215058
1.22917006626237
1.56829083715861
2.00644881328765

Column 5

0.795057097107989
0.794831811330473
0.79614404633497
0.812401980887257
0.870590173813465
1.01133179511072
1.24892095260445
1.58589344994397
2.0184335865212

Column 6

0.891294027750468
0.889940239218525
0.890175510230083
0.904458989129365
0.961094417635899
1.09811796014624
1.32482495141409
1.64945601831767
2.08059064903417

Column 7

1.13633541926908
1.13308021631668
1.13076916510609
1.14159321874177
1.18725600424638
1.30765014895287
1.52854504304729

```
          1.83457359597481
          2.23017264723076

  Column 8

          1.61522447621563
          1.60861248038785
          1.59598193223475
           1.5904560864873
          1.63167664365235
          1.72944843971472
          1.90374069413816
          2.16899390316922
          2.54742449597756

  Column 9

          2.33079729951374
          2.31087677525223
          2.29368999849939
          2.27438351236782
          2.27724234806646
          2.33508078057224
          2.49089481184024
          2.71332921224235
          2.99288353358798


G(:,:,9) =

  Column 1

          1.65682172709113
          1.65714554007458
           1.6607624538984
          1.67983745579696
          1.73938423903641
           1.8719908867913
          2.09101868006675
           2.3990671546106
          2.79167867861136

  Column 2

           1.6568282993978
          1.65715088577486
          1.66076428802212
          1.67982575635992
           1.7393536483455
          1.87188277685955
          2.09084848400992
          2.39884491804917
          2.79139108557617

  Column 3

          1.65719545597639
          1.65748625795122
          1.66102347180804
```

1.67981352542967
1.73905217583845
1.87128342492494
2.08980114029973
2.39738967985152
2.78910331628689

Column 4

1.66182673952524
1.66177212623163
1.66474132242004
1.68283631886539
1.74091114030232
1.87178796980885
2.08815972970359
2.39127453272481
2.77678218434194

Column 5

1.68787119008715
1.68705620174045
1.68837360234295
1.70427584024953
1.75894493669924
1.88143623962152
2.08629890394076
2.386273826667
2.76371093829752

Column 6

1.78142264562797
1.77856542256752
1.77582376265668
1.78157244947891
1.82105022389657
1.93733245826311
2.12650699191026
2.39778138201489
2.75236398602105

Column 7

2.01238789174708
2.00170846984099
1.98269322964637
1.98105728453003
2.00276429224949
2.07986438840092
2.23343449160782
2.49663883440815
2.82970472732124

Column 8

2.43851620815985
2.42012043535811

```
          2.38677672699221
          2.34920711463743
          2.33148721690015
          2.39531273096933
           2.5194655813745
          2.70022087495371
          2.99793925126277

  Column 9

          3.06986663940711
           3.0294674918122
          2.96145028061831
          2.91105596310499
          2.86873856987132
          2.84930231046628
            2.934942249808
          3.05409479080833
          3.26702271386941


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

H=max(DF,[],4)

H(:,:,1) =

  Column 1

     4.14835196829855e-012
     2.58059602015949e-005
       0.00100233091081671
        0.0208470259991422
        0.0856417837154069
         0.252088154228514
         0.533865873370361
         0.927000281155815
          1.40898533002836

  Column 2

     2.14165659902038e-006
     2.58639163941664e-005
       0.00100317147967103
        0.0208498357176391
        0.0856455401685027
         0.252093393571758
         0.533872173953911
         0.927007810868575
          1.40899434934968

  Column 3

      0.000289288297592806
      0.000294983590006527
       0.00129618282949716
          0.02112845546735
```

0.0859627563517384
0.252457949919731
0.534277593498057
0.927445810821213
1.40950291972479

Column 4

0.00463626627039766
0.00464112144421543
0.00556783930045279
0.0257241654159399
0.0908860921347209
0.257736266012445
0.539753219483512
0.933201299946002
1.41596314575128

Column 5

0.0332416803209371
0.0332719685219976
0.0345359630467072
0.0553516256724191
0.121886800041618
0.289978911360953
0.573050367815913
0.967579606206702
1.45255324811143

Column 6

0.141258500498537
0.141347879077356
0.143276942565469
0.165277472676256
0.2343875450574
0.405150566692935
0.689923096038174
1.08411674798498
1.57237261175944

Column 7

0.414826969974047
0.41502814352072
0.417729445217177
0.43573437171369
0.51092911677288
0.678831841352847
0.962978685816206
1.35974611764403
1.84782853940111

Column 8

0.953838572015885
0.954387972655065
0.959776429246508

```
         0.988958114695123
          1.06904611160074
          1.24117890600596
           1.5181072185803
          1.89494144351001
          2.35958207393771

  Column 9

          1.83199038382942
           1.8325732596625
          1.83837641958467
          1.86606237105085
          1.94059760172045
          2.09787586347392
          2.34513384215683
          2.67818015366122
          3.08758552773333


H(:,:,2) =

  Column 1

      4.26155801440063e-008
      2.58237344594712e-005
        0.00100237457752301
         0.0208470620587161
         0.0856418317214669
          0.25208822299904
         0.533865955150096
          0.92700037268838
          1.40898543702049

  Column 2

      2.16971002920482e-006
       2.5881691661625e-005
        0.00100321515029793
         0.0208498717790578
          0.085645588176702
          0.252093462342821
          0.533872255734402
          0.927007902401695
           1.40899445634548

  Column 3

       0.000289331015760181
       0.000295026363588643
         0.0012962265991743
         0.0211284916248485
         0.0859628044631387
          0.252458018763642
          0.534277675332899
          0.927445902391642
           1.40950302684367

  Column 4
```

```
     0.00463629510135567
     0.00464115034648784
     0.00556787106299569
      0.0257242022226146
      0.0908861409138459
        0.25773633527528
       0.539753301630655
       0.933201392225624
        1.41596325372087

Column 5

      0.0332417142968694
      0.0332720025599904
      0.0345359998345395
      0.0553516672685884
       0.121886853599974
       0.289978984197011
       0.573050451287047
       0.967579697806234
        1.45255335069661

Column 6

       0.141258543336351
       0.141347921953827
       0.143276987953404
       0.165277522717291
       0.234387604434753
       0.405150642256345
       0.689923167540295
        1.08411682533653
        1.57237270478379

Column 7

        0.41482705934237
       0.415028232127228
       0.417729532328374
       0.435734462752666
       0.510929208133733
       0.678831916297381
       0.962978757446316
        1.35974618331174
         1.8478286104103

Column 8

       0.953838707959519
       0.954388107713294
       0.959776558649921
       0.988958233791261
         1.0690462175739
        1.24117899275502
        1.51810728308325
        1.89494116583623
        2.35958171548675
```

```
  Column 9

           1.8319905294316
          1.83257340221079
           1.8383765535144
          1.86606248639472
          1.94059769073395
          2.09787566396253
          2.34513347432994
          2.67817961620623
          3.08758483060938


H(:,:,3) =

  Column 1

      1.35597424422367e-005
      3.58963022700342e-005
        0.00101615835256028
          0.020861360105573
         0.0856583347056838
          0.252107923076814
          0.533887638085424
          0.927024444295808
           1.40901361966613

  Column 2

      1.50002727659544e-005
      3.59544174215979e-005
        0.00101699962401244
         0.0208641701189611
         0.0856620915337164
          0.252113162612315
          0.533893938820922
          0.927031974209392
           1.40902263985704

  Column 3

       0.000302866054072254
       0.000308580560099884
        0.00131002821577727
         0.0211428060595779
         0.0859793253492012
          0.252477731612322
          0.534299369463456
          0.927469975931915
           1.40953122362133

  Column 4

        0.00464927264621289
        0.00465413961336591
        0.00558135987005546
         0.0257386438714023
         0.0909027958089924
          0.257756115388756
```

```
         0.539775045934044
           0.9332254502723
          1.41599153881446

Column 5

        0.0332556256933675
        0.0332859249812707
        0.0345504061628329
        0.0553669549082912
         0.121904312988209
         0.289999110454585
         0.573072430450596
         0.967603784578374
          1.45258184027358

Column 6

         0.141274167556258
         0.141363537750419
         0.143293014858842
         0.165294118833328
          0.23440614067965
         0.405171556963713
         0.689945339609211
          1.08414130207574
          1.57239746765603

Column 7

         0.414848511741715
         0.415049618842867
          0.41775089476843
         0.435756893207768
         0.510951544244749
         0.678856097359882
         0.962998290528859
          1.35976482139452
           1.8478487458433

Column 8

         0.953869495480807
         0.954418696923475
         0.959806855548138
         0.988987895514961
          1.06907089261907
          1.24120005422017
          1.51812421076259
          1.89495326724034
          2.35959293951509

Column 9

          1.83202216931385
          1.83260472851931
          1.83840521839376
          1.86608758833161
          1.94061798364833
```

```
           2.09789015092297
            2.3451396928786
           2.67814541321662
            3.0875350670944


H(:,:,4) =

  Column 1

       0.000671764289081818
       0.000678194409106794
        0.00168438016048615
         0.0214836706626907
         0.0863398549612933
          0.252866126924824
          0.534708320649208
          0.927896482240481
           1.41001576791037

  Column 2

       0.000672480146103262
       0.000678926849701771
        0.00168524635517389
         0.0214864889528666
         0.0863436210723576
          0.252871372793983
          0.534714616923321
          0.927904007469248
           1.41002477010426

  Column 3

       0.000961787570561285
       0.000968546989146638
        0.00197897016413172
         0.0217655826736701
         0.0866612656470272
          0.253236243615836
          0.535119967549327
          0.928341841855003
            1.4105333922414

  Column 4

        0.00523916417670127
        0.00524456269367732
        0.00620681206938705
         0.0263650674129435
         0.0915875189525752
          0.258515972069036
          0.540596312568773
          0.934098962391078
           1.41699711815208

  Column 5

         0.0338693165764847
```

```
        0.0339000175491905
        0.0351756986462058
        0.0560136009311236
         0.122607300131158
         0.290772314082325
         0.573904526834954
         0.968467780678113
          1.45357593003985

  Column 6

         0.141929968544406
         0.142018891805633
         0.143957498837697
         0.165977510486947
         0.235143091078383
         0.405946891038859
         0.690757679229072
            1.084999701868
          1.57337679622121

  Column 7

         0.415723982866841
         0.415923187571894
         0.418624174073003
         0.436654764358418
         0.511868298853532
           0.6797000291522
          0.96386517186708
          1.36063932191713
          1.84858571558589

  Column 8

         0.954887732903321
         0.955424242055341
         0.960796947458284
         0.989951760900222
          1.07001846794817
          1.24211494780758
          1.51875242778049
          1.89542537069968
          2.35941536666356

  Column 9

          1.83305580367877
          1.83362210124554
          1.83939416463716
          1.86702866247475
          1.94131932827363
          2.09840148110049
          2.34481071398072
           2.6775052460867
          3.08660397051486


H(:,:,5) =
```

Column 1

0.0121453759115216
0.0121566933442403
0.0132232841569587
0.0300900397305407
0.0957887258662577
0.262764617902919
0.545117005549982
0.938882817481368
1.42198398507237

Column 2

0.0121462591973745
0.0121575925901386
0.0132243001356321
0.0300929871041798
0.0957917419076419
0.262769813199165
0.545123175401702
0.938890122892387
1.42199287068995

Column 3

0.0124396508752356
0.012451214256432
0.013521784994483
0.030377271553983
0.0961453766145529
0.263137168405666
0.545529938431583
0.939327814775968
1.42249432526437

Column 4

0.0166549173456271
0.0166684293668405
0.017779899400778
0.0350214996993742
0.10086030849787
0.268444672930764
0.551001817444985
0.945068646898821
1.42895455616318

Column 5

0.0423625006816055
0.0423997913164944
0.0437940830214097
0.0648967495290889
0.132036512528275
0.300789744548052
0.584378760893811
0.97945473689583

```
         1.46553742974776

  Column 6

        0.150950795402507
        0.151040795722245
        0.153038954309275
        0.175260570032721
        0.244926528525996
        0.416226816208072
        0.701364767328794
         1.09544834476242
         1.58447805065791

  Column 7

        0.428241949757499
        0.428423182028754
        0.430915954238667
        0.448899985061318
        0.523896186274579
        0.689919981333605
        0.973894966650094
         1.36904704318452
          1.8571997256536

  Column 8

        0.966812092699176
        0.967276957997504
        0.972561667808495
         1.00105665274968
         1.08064234490048
          1.2505588287849
         1.52625977505183
         1.90178943775364
         2.36570919058689

  Column 9

         1.84486200982924
         1.84512582915239
         1.85041178143609
         1.87613826718031
         1.94923023800445
         2.10473443891884
         2.34988703933047
           2.681000035135
         3.08872036667639


H(:,:,6) =

  Column 1

        0.080491212225729
       0.0805214719735678
       0.0817742157629242
       0.0974799885733853
```

0.165954826446329
0.333217837697921
0.616035532498503
1.00974905880067
1.49453807478533

Column 2

0.0804926109104114
0.0805228663862566
0.0817756766023601
0.0974831857990829
0.165957918604791
0.333222659474768
0.616040890840598
1.00975104853094
1.49454042234463

Column 3

0.0807976394529919
0.0808278012477344
0.0820835948974845
0.0977831330298091
0.166311061649432
0.333574453641596
0.616415856654372
1.01014838221443
1.49500099092171

Column 4

0.0851276650477422
0.0851560698530898
0.0864416350326377
0.102516502549975
0.171061085956663
0.338863538479322
0.621830983518361
1.01581114868694
1.50093507774546

Column 5

0.109678120149223
0.109717914537396
0.111161155737377
0.132909241406347
0.201530775148703
0.371417985354708
0.654354724375637
1.04880313895549
1.53459373329672

Column 6

0.219436950134029
0.219453649936555
0.221682990807975

```
         0.244361185629514
         0.314074353069577
         0.485313840235952
         0.767302053979589
          1.16052292456623
          1.64958217365835

  Column 7

         0.498599851383201
         0.498413635809866
         0.500481809843141
         0.518188728254882
         0.592066026488873
         0.755758480618869
            1.037184862163
          1.43054612617594
          1.91692844216118

  Column 8

          1.04124428530954
          1.04089539910649
          1.04279886915018
          1.06888065957725
          1.14548212608234
          1.31287920238109
          1.58398882119943
          1.95451173668724
          2.41439521017668

  Column 9

          1.91448356087452
          1.91279793022228
          1.91507517697689
           1.9378718899111
          2.00660629121478
          2.15656060049617
          2.39497465810935
           2.6906293363286
          3.08819086753077


H(:,:,7) =

  Column 1

         0.314800203706883
         0.314913511295829
         0.317176926635895
         0.339971537004124
         0.410284728751049
         0.579089870414851
          0.86011876589172
          1.25168265047007
          1.73657592672059

  Column 2
```

```
        0.314804098299796
        0.314917311034739
        0.317180616497519
        0.339974948364905
        0.410283207604047
         0.57908514821001
        0.860110457440159
          1.2516701420453
         1.73656059516724

Column 3

        0.315119990685056
        0.315231748249045
        0.317484964118383
        0.340270500113715
        0.410584142352405
        0.579385099417657
        0.860408892593309
         1.25173548741196
         1.73660069343696

Column 4

        0.320025803029133
         0.32010748329995
        0.322362983441767
        0.345180351001957
        0.415642874888729
        0.583942602167206
        0.864704447433153
         1.25451325677918
         1.73941810309391

Column 5

        0.351092386231943
        0.351094997433228
        0.353439954055839
        0.375637353174719
        0.446143567313117
        0.612405513518366
        0.892502598996007
         1.28291525530633
         1.76841096615334

Column 6

        0.463300950474612
        0.462611648609785
        0.464308788289485
        0.483859685888629
        0.553938055591916
        0.721635838204093
         1.00033761842331
          1.3878218361629
          1.8724675927666
```

```
  Column 7

          0.731490612445559
          0.727806859414093
          0.727942966590414
          0.749638358312004
          0.821590663487518
          0.987903895711913
           1.26151621183172
           1.64468493462194
           2.08642065132595

  Column 8

           1.28887634473136
            1.2847326246386
           1.28329799035198
           1.30321236639753
           1.37225043921604
           1.52881144815883
           1.74559193117881
           2.09537146156088
           2.50225376830652

  Column 9

           2.14411766214078
           2.13768267376139
           2.13300443709304
           2.14656785884176
           2.16420759648088
           2.29062770775168
           2.46842959045246
           2.76246269948469
           3.13497061175223


H(:,:,8) =

  Column 1

          0.890285328034157
          0.890603265801853
          0.894532139849899
          0.920257236412068
            0.9946244357101
           1.16360470332781
           1.43633468116224
           1.81262137064957
           2.27694396873581

  Column 2

          0.890292419137157
          0.890609212361313
            0.8945354266707
          0.920255344735939
          0.994614670475318
           1.16358455766853
```

1.43626607057617
1.8125279690527
2.27657922166478

Column 3

0.890713432909689
0.891017182396545
0.894917197378489
0.920590847756151
0.99470434261248
1.16347760369638
1.43504466722277
1.81085366133045
2.27478324925458

Column 4

0.8963991198242
0.896628265013442
0.900113557994327
0.925277391061535
0.997342886524745
1.16477089494231
1.43569894816931
1.81037643730813
2.27358568693985

Column 5

0.929868018851857
0.92949039005226
0.930378143436255
0.95335241824818
1.02464760018472
1.1895905183452
1.45781786059593
1.82977115396798
2.29129732122196

Column 6

1.04468935512929
1.04181776084142
1.04111894567223
1.0606360943656
1.12848608250483
1.28870175169197
1.5509890966406
1.87569641028867
2.32689265412934

Column 7

1.31589000185468
1.30982429002226
1.3051492377124
1.32039337182458
1.38399344349786

```
          1.48862888491312
          1.73084728015721
          2.03655658332935
           2.4675224542562

  Column 8

          1.85735213880306
          1.84688459200359
          1.83652488081418
          1.79641953770901
          1.83854773192737
          1.91692408070156
          2.12557337621596
          2.42881481021067
          2.81869369788604

  Column 9

          2.66476214049693
          2.60912066023137
          2.57775954345779
          2.52223403966209
          2.53215257832097
          2.61300550826359
          2.77481888997185
          3.02184653907521
          3.15756411038158


H(:,:,9) =

  Column 1

          1.92994228394063
          1.93038013479595
          1.93521782820656
          1.96101293498522
          2.03375430555725
          2.18633801328843
          2.42920222533327
          2.76039673431675
          3.17320280170182

  Column 2

          1.92995110220074
          1.93038691134389
          1.93521894000508
          1.96098690923459
          2.03369607598033
           2.1860139590576
          2.42873540282515
          2.75980067953865
          3.17247309338249

  Column 3

          1.93042278487154
```

```
          1.93079841746177
          1.93550181333358
          1.96051246361171
          2.03254990723782
            2.184260014231
          2.42616379892204
          2.75643028386337
          3.16837282790545

Column 4

          1.93629379189994
          1.93572471225737
          1.93921954533344
          1.96296981609888
          2.03300994804939
          2.18231009414829
          2.42169072778863
          2.74965082367982
          3.13464453004679

Column 5

          1.96893906081139
          1.96689498770495
          1.96764045023484
           1.9876162350596
          2.05302635403302
          2.19691953941364
          2.39417527005155
          2.71018154218293
          3.08574853081654

Column 6

          2.07728544182086
          2.07153974442113
          2.06689004966027
          2.07989648807346
          2.09438932394108
          2.21796083850502
          2.39532622954412
          2.69270225710987
           3.0740919302703

Column 7

          2.33439310945228
          2.32252269402572
          2.26609388778223
          2.25523421330414
          2.24604668035009
           2.3402406846655
          2.52206602717494
          2.79489251189644
          3.15001608914681

Column 8
```

```
           2.78635002394732
           2.75059884394163
             2.676500788645
           2.63239354289991
           2.62585346510264
            2.6874831417864
           2.83204498386971
           2.88040609361967
           3.17363355640216

  Column 9

           3.45950491385946
           3.39370521981756
           3.31681629232049
           3.24598384847078
           3.20759354476773
            3.0422557244097
           3.13131702222487
            3.2327156108979
           3.46042318147226

>> G(:,:,1)

ans =

  Column 1

       8.66967089313903e-013
       6.63965241346711e-006
        0.000848838343689293
          0.0129783671123102
          0.0626501438748371
           0.204751428859847
           0.453129195334219
           0.808973864720106
            1.26542124827406

  Column 2

       7.67391271412567e-007
       7.41467640679065e-006
        0.000849722125380969
          0.0129796834331766
          0.0626525497748384
           0.204755096747503
           0.453134050583087
           0.808980031920564
            1.26542918825692

  Column 3

        0.000190240682293875
        0.000197223106531576
         0.00104369053183478
          0.0131897041737157
          0.0629029375438744
           0.205048182006443
           0.453464185762579
```

0.809343861921195
1.26584667183368

Column 4

0.0035602141328008
0.00356942161739597
0.00445306640050146
0.0167485010507226
0.0668011163407024
0.20925858586588
0.457897392259359
0.81402228662907
1.27101015231029

Column 5

0.0252131420831772
0.0252427194947908
0.0263379798850439
0.0392905372296161
0.0907775945257011
0.234509408236977
0.484271698881021
0.841383037769487
1.3001247523388

Column 6

0.115818091276002
0.115903686322838
0.117489533461926
0.131948089101253
0.186590008799572
0.33254820714403
0.583853818139615
0.941941803391434
1.4022871497926

Column 7

0.356710264541332
0.356915931690014
0.359475857071452
0.376881739484101
0.436757122896721
0.585359883162562
0.835687539849624
1.19174968014839
1.64779103991418

Column 8

0.855829908675294
0.856150544884543
0.859603812536529
0.879299153124425
0.942064923460029
1.08696211058595

```
          1.32818966609218
            1.668070865546
          2.10370706287797

  Column 9

           1.6256647101132
           1.6260659077185
          1.63014733576197
          1.65036092756285
          1.71121841594786
          1.84555367957782
          2.06609758453123
          2.37216842303167
          2.76022262277317


val(:,:,1) =

  Column 1

     4.14835196829855e-012
     2.58059602015949e-005
       0.00100233091081671
        0.0208470259991422
        0.0856417837154069
         0.252088154228514
         0.533865873370361
         0.927000281155815
          1.40898533002836

  Column 2

     2.14165659902038e-006
     2.58639163941664e-005
       0.00100317147967103
        0.0208498357176391
        0.0856455401685027
         0.252093393571758
         0.533872173953911
         0.927007810868575
          1.40899434934968

  Column 3

      0.000289288297592806
      0.000294983590006527
       0.00129618282949716
          0.02112845546735
        0.0859627563517384
         0.252457949919731
         0.534277593498057
         0.927445810821213
          1.40950291972479

  Column 4

       0.00463626627039766
```

```
 0.00464112144421543
 0.00556783930045279
  0.0257241654159399
  0.0908860921347209
   0.257736266012445
   0.539753219483512
   0.933201299946002
    1.41596314575128

Column 5

  0.0332416803209371
  0.0332719685219976
  0.0345359630467072
  0.0553516256724191
   0.121886800041618
   0.289978911360953
   0.573050367815913
   0.967579606206702
    1.45255324811143

Column 6

   0.141258500498537
   0.141347879077356
   0.143276942565469
   0.165277472676256
     0.2343875450574
   0.405150566692935
   0.689923096038174
    1.08411674798498
    1.57237261175944

Column 7

   0.414826969974047
    0.41502814352072
   0.417729445217177
    0.43573437171369
    0.51092911677288
   0.678831841352847
   0.962978685816206
    1.35974611764403
    1.84782853940111

Column 8

   0.953838572015885
   0.954387972655065
   0.959776429246508
   0.988958114695123
    1.06904611160074
    1.24117890600596
     1.5181072185803
    1.89494144351001
    2.35958207393771

Column
```

```
          1.83199038382942
           1.8325732596625
          1.83837641958467
          1.86606237105085
          1.94059760172045
          2.09787586347392
          2.34513384215683
          2.67818015366122
          3.08758552773333


val(:,:,2) =

  Column 1

     4.26155801440063e-008
     2.58237344594712e-005
       0.00100237457752301
        0.0208470620587161
        0.0856418317214669
          0.25208822299904
         0.533865955150096
          0.92700037268838
          1.40898543702049

  Column 2

     2.16971002920482e-006
      2.5881691661625e-005
       0.00100321515029793
        0.0208498717790578
         0.085645588176702
         0.252093462342821
         0.533872255734402
         0.927007902401695
          1.40899445634548

  Column 3

      0.000289331015760181
      0.000295026363588643
        0.0012962265991743
        0.0211284916248485
        0.0859628044631387
         0.252458018763642
         0.534277675332899
         0.927445902391642
          1.40950302684367

  Column 4

       0.00463629510135567
       0.00464115034648784
       0.00556787106299569
        0.0257242022226146
        0.0908861409138459
          0.25773633527528
         0.539753301630655
         0.933201392225624
```

1.41596325372087

Column 5

0.0332417142968694
0.0332720025599904
0.0345359998345395
0.0553516672685884
0.121886853599974
0.289978984197011
0.573050451287047
0.967579697806234
1.45255335069661

Column 6

0.141258543336351
0.141347921953827
0.143276987953404
0.165277522717291
0.234387604434753
0.405150642256345
0.689923167540295
1.08411682533653
1.57237270478379

Column 7

0.41482705934237
0.415028232127228
0.417729532328374
0.435734462752666
0.510929208133733
0.678831916297381
0.962978757446316
1.35974618331174
1.8478286104103

Column 8

0.953838707959519
0.954388107713294
0.959776558649921
0.988958233791261
1.0690462175739
1.24117899275502
1.51810728308325
1.89494116583623
2.35958171548675

Column 9

1.8319905294316
1.83257340221079
1.8383765535144
1.86606248639472
1.94059769073395
2.09787566396253
2.34513347432994

```
          2.67817961620623
          3.08758483060938


val(:,:,3) =

  Column 1

      1.35597424422367e-005
      3.58963022700342e-005
        0.00101615835256028
         0.020861360105573
        0.0856583347056838
         0.252107923076814
         0.533887638085424
         0.927024444295808
          1.40901361966613

  Column 2

      1.50002727659544e-005
      3.59544174215979e-005
        0.00101699962401244
        0.0208641701189611
        0.0856620915337164
         0.252113162612315
         0.533893938820922
         0.927031974209392
          1.40902263985704

  Column 3

       0.000302866054072254
       0.000308580560099884
        0.00131002821577727
        0.0211428060595779
        0.0859793253492012
         0.252477731612322
         0.534299369463456
         0.927469975931915
          1.40953122362133

  Column 4

        0.00464927264621289
        0.00465413961336591
        0.00558135987005546
        0.0257386438714023
        0.0909027958089924
         0.257756115388756
         0.539775045934044
           0.9332254502723
          1.41599153881446

  Column 5

        0.0332556256933675
        0.0332859249812707
        0.0345504061628329
```

```
        0.0553669549082912
         0.121904312988209
         0.289999110454585
         0.573072430450596
         0.967603784578374
          1.45258184027358

  Column 6

         0.141274167556258
         0.141363537750419
         0.143293014858842
         0.165294118833328
          0.23440614067965
         0.405171556963713
         0.689945339609211
          1.08414130207574
          1.57239746765603

  Column 7

         0.414848511741715
         0.415049618842867
          0.41775089476843
         0.435756893207768
         0.510951544244749
         0.678856097359882
         0.962998290528859
          1.35976482139452
           1.8478487458433

  Column 8

         0.953869495480807
         0.954418696923475
         0.959806855548138
         0.988987895514961
          1.06907089261907
          1.24120005422017
          1.51812421076259
          1.89495326724034
          2.35959293951509

  Column 9

          1.83202216931385
          1.83260472851931
          1.83840521839376
          1.86608758833161
          1.94061798364833
          2.09789015092297
           2.3451396928786
          2.67814541321662
           3.0875350670944


val(:,:,4) =

  Column 1
```

```
      0.000671764289081818
      0.000678194409106794
       0.00168438016048615
        0.0214836706626907
        0.0863398549612933
         0.252866126924824
         0.534708320649208
         0.927896482240481
          1.41001576791037

Column 2

      0.000672480146103262
      0.000678926849701771
       0.00168524635517389
        0.0214864889528666
        0.0863436210723576
         0.252871372793983
         0.534714616923321
         0.927904007469248
          1.41002477010426

Column 3

      0.000961787570561285
      0.000968546989146638
       0.00197897016413172
        0.0217655826736701
        0.0866612656470272
         0.253236243615836
         0.535119967549327
         0.928341841855003
           1.4105333922414

Column 4

       0.00523916417670127
       0.00524456269367732
       0.00620681206938705
        0.0263650674129435
        0.0915875189525752
         0.258515972069036
         0.540596312568773
         0.934098962391078
          1.41699711815208

Column 5

        0.0338693165764847
        0.0339000175491905
        0.0351756986462058
        0.0560136009311236
         0.122607300131158
         0.290772314082325
         0.573904526834954
         0.968467780678113
          1.45357593003985
```

```
  Column 6

        0.141929968544406
        0.142018891805633
        0.143957498837697
        0.165977510486947
        0.235143091078383
        0.405946891038859
        0.690757679229072
           1.084999701868
         1.57337679622121

  Column 7

        0.415723982866841
        0.415923187571894
        0.418624174073003
        0.436654764358418
        0.511868298853532
          0.6797000291522
         0.96386517186708
         1.36063932191713
         1.84858571558589

  Column 8

        0.954887732903321
        0.955424242055341
        0.960796947458284
        0.989951760900222
         1.07001846794817
         1.24211494780758
         1.51875242778049
         1.89542537069968
         2.35941536666356

  Column 9

         1.83305580367877
         1.83362210124554
         1.83939416463716
         1.86702866247475
         1.94131932827363
         2.09840148110049
         2.34481071398072
          2.6775052460867
         3.08660397051486


val(:,:,5) =

  Column 1

       0.0121453759115216
       0.0121566933442403
       0.0132232841569587
       0.0300900397305407
       0.0957887258662577
        0.262764617902919
```

0.545117005549982
0.938882817481368
1.42198398507237

Column 2

0.0121462591973745
0.0121575925901386
0.0132243001356321
0.0300929871041798
0.0957917419076419
0.262769813199165
0.545123175401702
0.938890122892387
1.42199287068995

Column 3

0.0124396508752356
0.012451214256432
0.013521784994483
0.030377271553983
0.0961453766145529
0.263137168405666
0.545529938431583
0.939327814775968
1.42249432526437

Column 4

0.0166549173456271
0.0166684293668405
0.017779899400778
0.0350214996993742
0.10086030849787
0.268444672930764
0.551001817444985
0.945068646898821
1.42895455616318

Column 5

0.0423625006816055
0.0423997913164944
0.0437940830214097
0.0648967495290889
0.132036512528275
0.300789744548052
0.584378760893811
0.97945473689583
1.46553742974776

Column 6

0.150950795402507
0.151040795722245
0.153038954309275
0.175260570032721
0.244926528525996

```
        0.416226816208072
        0.701364767328794
         1.09544834476242
         1.58447805065791

  Column 7

        0.428241949757499
        0.428423182028754
        0.430915954238667
        0.448899985061318
        0.523896186274579
        0.689919981333605
        0.973894966650094
         1.36904704318452
          1.8571997256536

  Column 8

        0.966812092699176
        0.967276957997504
        0.972561667808495
         1.00105665274968
         1.08064234490048
          1.2505588287849
         1.52625977505183
         1.90178943775364
         2.36570919058689

  Column 9

         1.84486200982924
         1.84512582915239
         1.85041178143609
         1.87613826718031
         1.94923023800445
         2.10473443891884
         2.34988703933047
           2.681000035135
         3.08872036667639


val(:,:,6) =

  Column 1

        0.080491212225729
       0.0805214719735678
       0.0817742157629242
       0.0974799885733853
        0.165954826446329
        0.333217837697921
        0.616035532498503
         1.00974905880067
         1.49453807478533

  Column 2

       0.0804926109104114
```

0.0805228663862566
0.0817756766023601
0.0974831857990829
0.165957918604791
0.333222659474768
0.616040890840598
1.00975104853094
1.49454042234463

Column 3

0.0807976394529919
0.0808278012477344
0.0820835948974845
0.0977831330298091
0.166311061649432
0.333574453641596
0.616415856654372
1.01014838221443
1.49500099092171

Column 4

0.0851276650477422
0.0851560698530898
0.0864416350326377
0.102516502549975
0.171061085956663
0.338863538479322
0.621830983518361
1.01581114868694
1.50093507774546

Column 5

0.109678120149223
0.109717914537396
0.111161155737377
0.132909241406347
0.201530775148703
0.371417985354708
0.654354724375637
1.04880313895549
1.53459373329672

Column 6

0.219436950134029
0.219453649936555
0.221682990807975
0.244361185629514
0.314074353069577
0.485313840235952
0.767302053979589
1.16052292456623
1.64958217365835

Column 7

```
         0.498599851383201
         0.498413635809866
         0.500481809843141
         0.518188728254882
         0.592066026488873
         0.755758480618869
            1.037184862163
          1.43054612617594
          1.91692844216118

  Column 8

          1.04124428530954
          1.04089539910649
          1.04279886915018
          1.06888065957725
          1.14548212608234
          1.31287920238109
          1.58398882119943
          1.95451173668724
          2.41439521017668

  Column 9

          1.91448356087452
          1.91279793022228
          1.91507517697689
           1.9378718899111
          2.00660629121478
          2.15656060049617
          2.39497465810935
           2.6906293363286
          3.08819086753077


val(:,:,7) =

  Column 1

         0.314800203706883
         0.314913511295829
         0.317176926635895
         0.339971537004124
         0.410284728751049
         0.579089870414851
          0.86011876589172
          1.25168265047007
          1.73657592672059

  Column 2

         0.314804098299796
         0.314917311034739
         0.317180616497519
         0.339974948364905
         0.410283207604047
          0.57908514821001
         0.860110457440159
           1.2516701420453
```

```
   1.73656059516724

Column 3

  0.315119990685056
  0.315231748249045
  0.317484964118383
  0.340270500113715
  0.410584142352405
  0.579385099417657
  0.860408892593309
   1.25173548741196
   1.73660069343696

Column 4

  0.320025803029133
   0.32010748329995
  0.322362983441767
  0.345180351001957
  0.415642874888729
  0.583942602167206
  0.864704447433153
   1.25451325677918
   1.73941810309391

Column 5

  0.351092386231943
  0.351094997433228
  0.353439954055839
  0.375637353174719
  0.446143567313117
  0.612405513518366
  0.892502598996007
   1.28291525530633
   1.76841096615334

Column 6

  0.463300950474612
  0.462611648609785
  0.464308788289485
  0.483859685888629
  0.553938055591916
  0.721635838204093
   1.00033761842331
    1.3878218361629
    1.8724675927666

Column 7

  0.731490612445559
  0.727806859414093
  0.727942966590414
  0.749638358312004
  0.821590663487518
  0.987903895711913
   1.26151621183172
```

```
          1.64468493462194
          2.08642065132595

  Column 8

          1.28887634473136
           1.2847326246386
          1.28329799035198
          1.30321236639753
          1.37225043921604
          1.52881144815883
          1.74559193117881
          2.09537146156088
          2.50225376830652

  Column 9

          2.14411766214078
          2.13768267376139
          2.13300443709304
          2.14656785884176
          2.16420759648088
          2.29062770775168
          2.46842959045246
          2.76246269948469
          3.13497061175223


val(:,:,8) =

  Column 1

         0.890285328034157
         0.890603265801853
         0.894532139849899
         0.920257236412068
           0.9946244357101
          1.16360470332781
          1.43633468116224
          1.81262137064957
          2.27694396873581

  Column 2

         0.890292419137157
         0.890609212361313
           0.8945354266707
         0.920255344735939
         0.994614670475318
          1.16358455766853
          1.43626607057617
           1.8125279690527
          2.27657922166478

  Column 3

         0.890713432909689
         0.891017182396545
         0.894917197378489
```

```
        0.920590847756151
         0.99470434261248
         1.16347760369638
         1.43504466722277
         1.81085366133045
         2.27478324925458

Column 4

          0.8963991198242
        0.896628265013442
        0.900113557994327
        0.925277391061535
        0.997342886524745
         1.16477089494231
         1.43569894816931
         1.81037643730813
         2.27358568693985

Column 5

        0.929868018851857
         0.92949039005226
        0.930378143436255
         0.95335241824818
         1.02464760018472
          1.1895905183452
         1.45781786059593
         1.82977115396798
         2.29129732122196

Column 6

         1.04468935512929
         1.04181776084142
         1.04111894567223
          1.0606360943656
         1.12848608250483
         1.28870175169197
          1.5509890966406
         1.87569641028867
         2.32689265412934

Column 7

         1.31589000185468
         1.30982429002226
          1.3051492377124
         1.32039337182458
         1.38399344349786
         1.48862888491312
         1.73084728015721
         2.03655658332935
          2.4675224542562

Column 8

         1.85735213880306
         1.84688459200359
```

```
          1.83652488081418
          1.79641953770901
          1.83854773192737
          1.91692408070156
          2.12557337621596
          2.42881481021067
          2.81869369788604

  Column 9

          2.66476214049693
          2.60912066023137
          2.57775954345779
          2.52223403966209
          2.53215257832097
          2.61300550826359
          2.77481888997185
          3.02184653907521
          3.15756411038158


val(:,:,9) =

  Column 1

          1.92994228394063
          1.93038013479595
          1.93521782820656
          1.96101293498522
          2.03375430555725
          2.18633801328843
          2.42920222533327
          2.76039673431675
          3.17320280170182

  Column 2

          1.92995110220074
          1.93038691134389
          1.93521894000508
          1.96098690923459
          2.03369607598033
           2.1860139590576
          2.42873540282515
          2.75980067953865
          3.17247309338249

  Column 3

          1.93042278487154
          1.93079841746177
          1.93550181333358
          1.96051246361171
          2.03254990723782
            2.184260014231
          2.42616379892204
          2.75643028386337
          3.16837282790545
```

Column 4

1.93629379189994
1.93572471225737
1.93921954533344
1.96296981609888
2.03300994804939
2.18231009414829
2.42169072778863
2.74965082367982
3.13464453004679

Column 5

1.96893906081139
1.96689498770495
1.96764045023484
1.9876162350596
2.05302635403302
2.19691953941364
2.39417527005155
2.71018154218293
3.08574853081654

Column 6

2.07728544182086
2.07153974442113
2.06689004966027
2.07989648807346
2.09438932394108
2.21796083850502
2.39532622954412
2.69270225710987
3.0740919302703

Column 7

2.33439310945228
2.32252269402572
2.26609388778223
2.25523421330414
2.24604668035009
2.3402406846655
2.52206602717494
2.79489251189644
3.15001608914681

Column 8

2.78635002394732
2.75059884394163
2.676500788645
2.63239354289991
2.62585346510264
2.6874831417864
2.83204498386971
2.88040609361967
3.17363355640216

```
  Column 9

          3.45950491385946
          3.39370521981756
          3.31681629232049
          3.24598384847078
          3.20759354476773
           3.0422557244097
          3.13131702222487
           3.2327156108979
          3.46042318147226

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


% Copyright (C) Gerasimos Fotiadis 2012



%optimal routing strategy for k=1

%f(z,j):minimum expected cost
%H(z,j):next costumer cost
%R(z,j):vihicle route after costumer j has been satisfied
       %D stands for depot and N for next costumer
%A(1,j):depot cost
%c(j):travel cost between j->depot
%C(j):travel cost between j->j+1
%z   :the load of product after the demand of j has been satisfied
%k   :product demand(ξ)
%N   :costumers
%Q   :vihicle capacity

%for load z=0 the index for h(z,j)&f(z,j) is z=1
%for load z=1 the index for h(z,j)&f(z,j) is z=2
%for load z=Q the index for h(z,j)&f(z,j) is z=Q+1


format longG

N=input('Please enter the number of costumers:');
Q=input('Please enter the capacity of the vihicle:');

tic

%preallocate vector&matrices
f=zeros(Q+1,N);
H=zeros(Q+1,N-1);
A=zeros(1,N-1);
R=char(zeros(Q+1,N-1));


%fill cost vectors in a way that they satisfy triangle inequality
```

```
%c stands for costumer->depot cost
%C stands for costumer j->j+1 cost
c=randi([19,25],1,N);
C=randi([18,27],1,N-1);

%in the boundary
f(:,N)=c(N);


p=0.3;


for j=N-1:-1:1;

     m=0;


     %calculate A(j) sum
    for k=0:Q;
        m=m+f(Q-k+1,j+1)*binopdf(k,Q,p);
    end


     %fill A(j)
    A(j)=c(j)+c(j+1)+m;



    for z=0:Q;

        s=0;
        S=0;


        for k=0:Q;


            %calculate 1st and 2d sum of H(z,j)
            if k <= z;
                S=S+f(z+1-k,j+1)*binopdf(k,Q,p);
            else
                s=s+(2*c(j+1)+f(Q+min((z-k),0)+1,j+1))*binopdf(k,Q,p);
            end


        end


        %fill H(z,j) and f(z,j)
        H(z+1,j)=C(j)+S+s;

        f(z+1,j)=min(A(j),H(z+1,j));
```

```
        %fill R(z,j)
        if A(j)>H(z+1,j);
        R(z+1,j)='N';
        else
        R(z+1,j)='D';
        end


    end


end

S=0;

%calculate minimum expected cost during a cycle
for k=0:Q;

S=S+f(Q-k+1,1)*binopdf(k,Q,p);

end

f0=c(1)+S;

toc




%optimal routing strategy for k=2

%f(z1,z2,j):minimum expected cost
%H(z1,z2,j):next costumer cost
%R(z1,z2,j):vihicle route after costumer j has been satisfied
            %D stands for depot and N for next costumer
%A(1,1,j)   :depot cost
%c(j)       :travel cost between j->depot
%C(j)       :travel cost between j->j+1
%z1         :the load of product 1 after the demand of j has been satisfied
%z2         :the load of product 1 after the demand of j has been satisfied
%k1         :product demand(ξ1)
%k2         :product demand(ξ2)
%N          :costumers
%Q1         :vihicle capacity for product 1
%Q2         :vihicle capacity for product 2

%for load z1=0 & z2=0    the index for h(z1,z2,j)&f(z1,z2,j) is z1=1 & z2=1
%for load z1=1 & z2=1    the index for h(z1,z2,j)&f(z1,z2,j) is z1=2 & z2=2
%for load z1=Q1& z2=Q2+1 the index for h(z1,z2,j)&f(z1,z2,j) is
z1=Q1+1&z2=Q2+1


format longG

Q=zeros(1,2);
```

```
N=input('Please enter the number of costumers:');
Q(1,1)=input('Please enter the capacity of the vihicle for product 1:');
Q(1,2)=input('Please enter the capacity of the vihicle for product 2:');

tic

%preallocate vector&matrices
f=zeros(Q(1)+1,Q(2)+1,N);
H=zeros(Q(1)+1,Q(2)+1,N-1);
A=zeros(1,1,N-1);
R=char(zeros(Q(1)+1,Q(2)+1,N-1));


%fill cost vectors in a way that they satisfy triangle inequality
%c stands for costumer->depot cost
%C stands for costumer j->j+1 cost
c=randi([19,25],1,N);
C=randi([18,27],1,N-1);

%in the boundary
f(:,:,N)=c(N);


p=[0.3,0.5];


for j=N-1:-1:1;

    m=0;

    %calculate A(j) sum
    for k1=0:Q(1);
       for k2=0:Q(2);

           m=m+f(Q(1)-k1+1,Q(2)-k2+1,j+1) ...
               *binopdf(k1,Q(1),p(1))*binopdf(k2,Q(2),p(2));

       end
    end


    %fill A(j)
    A(j)=c(j)+c(j+1)+m;


    for z1=0:Q(1);
        for z2=0:Q(2);

            s=0;
            S=0;


            for k1=0:Q(1);
                for k2=0:Q(2);
```

```
                  %calculate 1st and 2d sum of H(z1,z2,j)
                  if k1 <= z1&&k2 <= z2;
                    S=S+f(z1-k1+1,z2-k2+1,j+1) ...
                        *binopdf(k1,Q(1),p(1))*binopdf(k2,Q(2),p(2));
                  else
                    s=s+(2*c(j+1)+f(Q(1)+min((z1-k1),0)+1,Q(2)+min((z2-
k2),0)+1,j+1)) ...
                        *binopdf(k1,Q(1),p(1))*binopdf(k2,Q(2),p(2));
                  end


              end
          end


        %fill H(z1,z2,j) and f(z1,z2,j)
        H(z1+1,z2+1,j)=C(j)+S+s;

        f(z1+1,z2+1,j)=min(A(j),H(z1+1,z2+1,j));


        %fill R(z1,z2,j)
        if A(j)>H(z1+1,z2+1,j);
        R(z1+1,z2+1,j)='N';
        else
        R(z1+1,z2+1,j)='D';
        end


        end
    end
end

S=0;

%calculate minimum expected cost during a cycle
for k1=0:Q(1);
     for k2=0:Q(2);

            S=S+f(Q(1)-k1+1,Q(2)-k2+1,1) ...
                *binopdf(k1,Q(1),p(1))*binopdf(k2,Q(2),p(2));

     end
end

f0=c(1)+S;

toc
```

```
%optimal routing strategy for k=3

%f(z1,z2,z3,j):minimum expected cost
%H(z1,z2,z3,j):next costumer cost
%R(z1,z2,z3,j):vihicle route after costumer j has been satisfied
               %D stands for depot and N for next costumer
%A(          j):depot cost
%c(j)          :travel cost between j->depot
%C(j)          :travel cost between j->j+1
%z1            :the load of product 1 after the demand of j has been
satisfied
%z2            :the load of product 1 after the demand of j has been
satisfied
%k1            :product demand(ξ1)
%k2            :product demand(ξ2)
%k3            :product demand(ξ3)
%N             :costumers
%Q1            :vihicle capacity for product 1
%Q2            :vihicle capacity for product 2
%Q3            :vihicle capacity for product 3


format longG

Q=zeros(1,3);

N=input('Please enter the number of costumers:');
Q(1,1)=input('Please enter the capacity of the vihicle for product 1:');
Q(1,2)=input('Please enter the capacity of the vihicle for product 2:');
Q(1,3)=input('Please enter the capacity of the vihicle for product 3:');


tic

%preallocate vector&matrices
f=zeros(Q(1)+1,Q(2)+1,Q(3)+1,N);
H=zeros(Q(1)+1,Q(2)+1,Q(3)+1,N-1);
A=zeros(N-1);
R=char(zeros(Q(1)+1,Q(2)+1,Q(3)+1,N-1));


%fill cost vectors in a way that they satisfy triangle inequality
c=randi([19,25],1,N);
C=randi([18,27],1,N-1);

%in the boundary
f(:,:,:,N)=c(N);


p=[0.3,0.5,0.7];


for j=N-1:-1:1;
```

```
m=0;

%calculate A(j) sum
for k1=0:Q(1);
    for k2=0:Q(2);
        for k3=0:Q(3)


            m=m+f(Q(1)-k1+1,Q(2)-k2+1,Q(3)-k3+1,j+1) ...
*binopdf(k1,Q(1),p(1))*binopdf(k2,Q(2),p(2))*binopdf(k3,Q(3),p(3));


        end
    end
end


%fill A(j)
A(j)=c(j)+c(j+1)+m;


for z1=0:Q(1);
     for z2=0:Q(2);
         for z3=0:Q(3);

         s=0;
         S=0;


         for k1=0:Q(1);
              for k2=0:Q(2);
                  for k3=0:Q(3)


                  %calculate 1st and 2d sum of H(z1,z2,z3,j)
                 if k1 <= z1&&k2 <= z2&&k3 <= z3;
                   S=S+f(z1-k1+1,z2-k2+1,z3-k3+1,j+1) ...
*binopdf(k1,Q(1),p(1))*binopdf(k2,Q(2),p(2))*binopdf(k3,Q(3),p(3));
                 else
                   s=s+(2*c(j+1)+f(Q(1)+min((z1-k1),0)+1,Q(2)+min((z2-
k2),0)+1,Q(3)+min((z3-k3),0)+1,j+1)) ...
*binopdf(k1,Q(1),p(1))*binopdf(k2,Q(2),p(2))*binopdf(k3,Q(3),p(3));
                 end


                  end
              end
         end
```

```
        %fill H(z1,z2,z3,j) and f(z1,z2,z3,j)
        H(z1+1,z2+1,z3+1,j)=C(j)+S+s;

        f(z1+1,z2+1,z3+1,j)=min(A(j),H(z1+1,z2+1,z3+1,j));


        %fill R(z1,z2,j)
        if A(j)>H(z1+1,z2+1,z3+1,j);
        R(z1+1,z2+1,z3+1,j)='N';
        else
        R(z1+1,z2+1,z3+1,j)='D';
        end

            end
        end
    end
end


%calculate minimum expected cost during a cycle
for k1=0:Q(1);
     for k2=0:Q(2);
         for k3=0:Q(3);


            S=S+f(Q(1)-k1+1,Q(2)-k2+1,Q(3)-k3+1,1) ...

*binopdf(k1,Q(2),p(1))*binopdf(k2,Q(2),p(2))*binopdf(k3,Q(3),p(3));


         end
     end
end

f0=c(1)+S;


toc



%optimal routing strategy for k=2 and manual P



format longG

Q=zeros(1,2);

N=input('Please enter the number of costumers:');
Q(1,1)=input('Please enter the capacity of the vihicle for product 1:');
Q(1,2)=input('Please enter the capacity of the vihicle for product 2:');

%call function in order the user to fill the possibility matrix
P=psblt(Q);
```

```
tic

%preallocate vector&matrices
f=zeros(Q(1)+1,Q(2)+1,N);
H=zeros(Q(1)+1,Q(2)+1,N-1);
A=zeros(1,1,N-1);
R=char(zeros(Q(1)+1,Q(2)+1,N-1));


%fill cost vectors in a way that they satisfy triangle inequality
%c stands for costumer->depot cost
%C stands for costumer j->j+1 cost
c=randi([19,25],1,N);
C=randi([18,27],1,N-1);


%in the boundary
f(:,:,N)=c(N);



for j=N-1:-1:1;

    m=0;

    %calculate A(j) sum
    for k1=0:Q(1);
       for k2=0:Q(2);

            m=m+f(Q(1)-k1+1,Q(2)-k2+1,j+1) ...
                 *P(1,k1+1)*P(2,k2+1);

       end
    end


    %fill A(j)
    A(j)=c(j)+c(j+1)+m;


    for z1=0:Q(1);
        for z2=0:Q(2);

            s=0;
            S=0;


            for k1=0:Q(1);
                for k2=0:Q(2);


                    %calculate 1st and 2d sum of H(z1,z2,j)
                   if k1 <= z1&&k2 <= z2;
```

```
                        S=S+f(z1-k1+1,z2-k2+1,j+1) ...
                              *P(1,k1+1)*P(1,k2+1);
                    else
                        s=s+(2*c(j+1)+f(Q(1)+min((z1-k1),0)+1,Q(2)+min((z2-
k2),0)+1,j+1)) ...
                              *P(1,k1+1)*P(2,k2+1);
                    end



                end
            end



        %fill H(z1,z2,j) and f(z1,z2,j)
        H(z1+1,z2+1,j)=C(j)+S+s;

        f(z1+1,z2+1,j)=min(A(j),H(z1+1,z2+1,j));


        %fill R(z1,z2,j)
        if A(j)>H(z1+1,z2+1,j);
        R(z1+1,z2+1,j)='N';
        else
        R(z1+1,z2+1,j)='D';
        end


        end
    end
end

S=0;

%calculate minimum expected cost during a cycle
for k1=0:Q(1);
      for k2=0:Q(2);

            S=S+f(Q(1)-k1+1,Q(2)-k2+1,1) ...
                  *P(1,k1+1)*P(2,k2+1);


      end
end

f0=c(1)+S;

toc



function P=psblt(Q)

%it prompts the user to fill the possibility maxtrix P
```

```
%Q:capacity vecor
%Q(1):capacity of the first product
%Q(2):capacity of the second product

%P(1,3)=possibility the first type has demand 2
%P(2,1)=possibility the second type has demand 0
%P(k,Q+1)=possibility the k type has demand Q


P=zeros(length(Q),max(Q)+1);


for i=1:length(Q)
    for j=1:Q(i);
        fprintf('Give possibility for product %d for demand %d',i,j-1);
        P(i,j)=input(':');
    end
    P(i,Q(i)+1)=1-sum(P(i,:));
end




%compare exact and heuristic solution for k=2


format longG

Q=zeros(1,2);

%N=input('Please enter the number of costumers:');
%Q(1,1)=input('Please enter the capacity of the vihicle for product 1:');
%Q(1,2)=input('Please enter the capacity of the vihicle for product 2:');


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

P=[0.03,0.07,0.05,0.06,0.1,0.15,0.17,0.2,0.12,0.025,0.025,0,0,0,0,0;...

0.02,0.03,0.06,0.08,0.12,0.14,0.17,0.18,0.02,0.03,0.04,0.05,0.02,0.02,0.01,
0.01];

N=15;

Q(1,1)=10;
Q(1,2)=15;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%fill cost vectors in a way that they satisfy triangle inequality
%c stands for costumer->depot cost
%C stands for costumer j->j+1 cost
c=randi([19,25],1,N);
C=randi([18,27],1,N-1);
```

```
%call function in order the user to fill the possibility matrix
%P=psblt(Q);

%call function so as to find TEC k2ask1
f0k2ask1=ORSk2ask1f(Q,N,c,C,P);

%call function so as to find TEC k2
f0k2=ORSk2f(Q,N,c,C,P);


function R=k2ask1R(Q,N,c,C,B)
%it finds routing maxtrix R for k2ask1

format longG

Q=Q(1,1)+Q(1,2);

%call function in order to add the demands
P=adD(B(1,:),B(2,:));

%preallocate vector&matrices
f=zeros(Q+1,N);
H=zeros(Q+1,N-1);
A=zeros(1,N-1);
R=char(zeros(Q+1,N-1));


%in the boundary
f(:,N)=c(N);



for j=N-1:-1:1;

     m=0;


     %calculate A(j) sum
    for k=0:Q;
        m=m+f(Q-k+1,j+1)*P(1,k+1);
    end


     %fill A(j)
    A(j)=c(j)+c(j+1)+m;



    for z=0:Q;

        s=0;
```

```
        S=0;


        for k=0:Q;


            %calculate 1st and 2d sum of H(z,j)
            if k <= z;
                S=S+f(z+1-k,j+1)*P(1,k+1);
            else
                s=s+(2*c(j+1)+f(Q+min((z-k),0)+1,j+1))*P(1,k+1);
            end


        end


        %fill H(z,j) and f(z,j)
        H(z+1,j)=C(j)+S+s;

        f(z+1,j)=min(A(j),H(z+1,j));


        %fill R(z,j)
        if A(j)>H(z+1,j);
        R(z+1,j)='N';
        else
        R(z+1,j)='D';
        end


    end


end


function f0=ORSk2ask1f(Q,N,c,C,P)
%it finds TECk2ask1

format longG


%preallocate vector&matrices
f=zeros(Q(1,1)+1,Q(1,2)+1,N);

%call function so as to find R
R=k2ask1R(Q,N,c,C,P);


%in the boundary
f(:,:,N)=c(N);
```

```
for j=N-1:-1:1;


    for z1=0:Q(1);
        for z2=0:Q(2);


              z=z1+z2;


              if R(z+1,j)=='D'
              %f(z1,z2,j) should be A(j)


               m=0;


                     %calculate A(j) sum
                      for k1=0:Q(1);
                          for k2=0:Q(2);

                              m=m+f(Q(1)-k1+1,Q(2)-k2+1,j+1) ...
                              *P(1,k1+1)*P(2,k2+1);

                          end
                      end
                     %fill f(z1,z2,j)
                     f(z1+1,z2+1,j)=c(j)+c(j+1)+m;




              else
              %f(z1,z2,j) should be H(z1,z2)


                          s=0;
                          S=0;


                     %calculate H(z1,z2) sum
                      for k1=0:Q(1);
                           for k2=0:Q(2);


                              if k1 <= z1&&k2 <= z2;
                                  S=S+f(z1-k1+1,z2-k2+1,j+1) ...
                                  *P(1,k1+1)*P(2,k2+1);
                              else
                                  s=s+(2*c(j+1)+f(Q(1)+min((z1-
k1),0)+1,Q(2)+min((z2-k2),0)+1,j+1)) ...
                                  *P(1,k1+1)*P(2,k2+1);
                              end
```

```
                    end
                end
                %fill f(z1,z2,j)
                f(z1+1,z2+1,j)=C(j)+S+s;

            end


        end
    end
end


S=0;

%calculate minimum expected cost during a cycle
for k1=0:Q(1);
    for k2=0:Q(2);

            S=S+f(Q(1)-k1+1,Q(2)-k2+1,1) ...
                *P(1,k1+1)*P(2,k2+1);


    end
end

f0=c(1)+S;


function P=adD(D1,D2)
%it adds D1+D2 and returns combined vector P

%D1:possibility-demand vector 1st product
%D2:possibility-demand vector 2d product


L1=length(D1);
L2=length(D2);


P=zeros(1,L1+L2-1);
P1=zeros(1,L1+L2-1);
P2=zeros(1,L1+L2-1);


P1(1:L1)=D1;
P2(1:L2)=D2;



for j=1:(L1+L2-1);
    S=0;
    for i=1:j;
        S=S+P1(i)*P2(j-i+1);
```

```
    end
    P(j)=S;
end


function f0=ORSk2f(Q,N,c,C,P)
%it finds the ORS for k=2

format longG


%preallocate vector&matrices
f=zeros(Q(1)+1,Q(2)+1,N);
H=zeros(Q(1)+1,Q(2)+1,N-1);
A=zeros(1,1,N-1);
R=char(zeros(Q(1)+1,Q(2)+1,N-1));


%in the boundary
f(:,:,N)=c(N);


for j=N-1:-1:1;

    m=0;

    %calculate A(j) sum
    for k1=0:Q(1);
       for k2=0:Q(2);

            m=m+f(Q(1)-k1+1,Q(2)-k2+1,j+1) ...
                *P(1,k1+1)*P(2,k2+1);

       end
    end


    %fill A(j)
    A(j)=c(j)+c(j+1)+m;


    for z1=0:Q(1);
        for z2=0:Q(2);

            s=0;
            S=0;


            for k1=0:Q(1);
                for k2=0:Q(2);

                    %calculate 1st and 2d sum of H(z1,z2,j)
                   if k1 <= z1&&k2 <= z2;
```

```
                S=S+f(z1-k1+1,z2-k2+1,j+1) ...
                     *P(1,k1+1)*P(2,k2+1);
              else
                s=s+(2*c(j+1)+f(Q(1)+min((z1-k1),0)+1,Q(2)+min((z2-
k2),0)+1,j+1)) ...
                     *P(1,k1+1)*P(2,k2+1);
              end



           end
         end



      %fill H(z1,z2,j) and f(z1,z2,j)
      H(z1+1,z2+1,j)=C(j)+S+s;

      f(z1+1,z2+1,j)=min(A(j),H(z1+1,z2+1,j));


      %fill R(z1,z2,j)
      if A(j)>H(z1+1,z2+1,j);
      R(z1+1,z2+1,j)='N';
      else
      R(z1+1,z2+1,j)='D';
      end


     end
   end
end

S=0;

%calculate minimum expected cost during a cycle
for k1=0:Q(1);
     for k2=0:Q(2);

         S=S+f(Q(1)-k1+1,Q(2)-k2+1,1) ...
              *P(1,k1+1)*P(2,k2+1);


     end
end

f0=c(1)+S;


format longG

tic

Q=zeros(1,3);

%N=input('Please enter the number of costumers:');
```

```
%Q(1,1)=input('Please enter the capacity of the vihicle for product 1:');
%Q(1,2)=input('Please enter the capacity of the vihicle for product 2:');
%Q(1,3)=input('Please enter the capacity of the vihicle for product 3:');


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

P=[0.03,0.07,0.05,0.06,0.1,0.15,0.17,0.2,0.12,0.025,0.025,0,0,0,0,0,0,0,0,0
,0;...

0.02,0.03,0.06,0.08,0.12,0.14,0.17,0.18,0.02,0.03,0.04,0.05,0.02,0.02,0.01,
0.01,0,0,0,0,0;...

0.01,0.02,0.04,0.06,0.09,0.08,0.03,0.11,0.01,0.05,0.02,0.01,0.03,0.02,0.03,
0.06,0.08,0.05...
    ,0.05,0.125,0.025];

N=20;

Q(1,1)=10;
Q(1,2)=15;
Q(1,3)=20;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%fill cost vectors in a way that they satisfy triangle inequality
%c stands for costumer j->depot cost
%C stands for costumer j->j+1 cost
c=randi([19,25],1,N);
C=randi([18,27],1,N-1);


%p=[0.9,0.9,0.9];


%call function in order the user to fill the possibility matrix
%P=psbltB2(Q,p);

%call function so as to find TEC k2ask1
f0k3ask1=ORSk3ask1f(Q,N,c,C,P);

%call function so as to find TEC k2
f0k3=ORSk3f(Q,N,c,C,P);

%DF=((f0k3ask1-f0k3)./f0k3)*100;

toc



function R=k3ask1R(Q,N,c,C,B)
%it finds routing maxtrix R for k3ask1

format longG
```

```
Q=Q(1,1)+Q(1,2)+Q(1,3);

%call function in order to add the demands
P=adD3(B(1,:),B(2,:),B(3,:));

%preallocate vector&matrices
f=zeros(Q+1,N);
H=zeros(Q+1,N-1);
A=zeros(1,N-1);
R=char(zeros(Q+1,N-1));


%in the boundary
f(:,N)=c(N);



for j=N-1:-1:1;

     m=0;


     %calculate A(j) sum
    for k=0:Q;
        m=m+f(Q-k+1,j+1)*P(1,k+1);
    end


     %fill A(j)
    A(j)=c(j)+c(j+1)+m;



    for z=0:Q;

        s=0;
        S=0;


        for k=0:Q;


            %calculate 1st and 2d sum of H(z,j)
            if k <= z;
                S=S+f(z+1-k,j+1)*P(1,k+1);
            else
                s=s+(2*c(j+1)+f(Q+min((z-k),0)+1,j+1))*P(1,k+1);
            end


        end
```

```
        %fill H(z,j) and f(z,j)
        H(z+1,j)=C(j)+S+s;

        f(z+1,j)=min(A(j),H(z+1,j));


        %fill R(z,j)
        if A(j)>H(z+1,j);
        R(z+1,j)='N';
        else
        R(z+1,j)='D';
        end


    end


end


function f0=ORSk3ask1f(Q,N,c,C,P)
%it finds TECk2ask1

format longG


%preallocate vector&matrices
f=zeros(Q(1,1)+1,Q(1,2)+1,Q(1,3)+1,N);

%call function so as to find R
R=k3ask1R(Q,N,c,C,P);


%in the boundary
f(:,:,:,N)=c(N);



for j=N-1:-1:1;


    for z1=0:Q(1);
        for z2=0:Q(2);
          for z3=0:Q(3);

           z=z1+z2+z3;


           if R(z+1,j)=='D'
           %f(z1,z2,j) should be A(j)


            m=0;
```

```
            %calculate A(j) sum
             for k1=0:Q(1);
                for k2=0:Q(2);
                   for k3=0:Q(3)

                       m=m+f(Q(1)-k1+1,Q(2)-k2+1,Q(3)-k3+1,j+1) ...
                       *P(1,k1+1)*P(2,k2+1)*P(3,k3+1);

                   end
                end
             end
            %fill f(z1,z2,j)
            f(z1+1,z2+1,z3+1,j)=c(j)+c(j+1)+m;



      else
      %f(z1,z2,j) should be H(z1,z2)


                s=0;
                S=0;


            %calculate H(z1,z2) sum
             for k1=0:Q(1);
                for k2=0:Q(2);
                   for k3=0:Q(3)


                    %calculate 1st and 2d sum of H(z1,z2,z3,j)
                     if k1 <= z1&&k2 <= z2&&k3 <= z3;
                       S=S+f(z1-k1+1,z2-k2+1,z3-k3+1,j+1) ...
                       *P(1,k1+1)*P(2,k2+1)*P(3,k3+1);
                     else
                       s=s+(2*c(j+1)+f(Q(1)+min((z1-
k1),0)+1,Q(2)+min((z2-k2),0)+1,Q(3)+min((z3-k3),0)+1,j+1)) ...
                       *P(1,k1+1)*P(2,k2+1)*P(3,k3+1);
                     end


                   end
                end
             end
             %fill f(z1,z2,,z3,j)
             f(z1+1,z2+1,z3+1,j)=C(j)+S+s;

      end

     end
    end
  end
end
```

```
S=0;

%calculate minimum expected cost during a cycle
for k1=0:Q(1);
     for k2=0:Q(2);
          for k3=0:Q(3);


               S=S+f(Q(1)-k1+1,Q(2)-k2+1,Q(3)-k3+1,1) ...
               *P(1,k1+1)*P(2,k2+1)*P(3,k3+1);


          end
     end
end

f0=c(1)+S;


function P=adD3(D1,D2,D3)
%it adds D1+D2+D3 and returns combined vector P

%D1:possibility-demand vector 1st product
%D2:possibility-demand vector 2d product
%D3:possibility-demand vector 3d product


L1=length(D1);
L2=length(D2);
L3=length(D3);


P=zeros(1,L1+L2+L3-2);
P1=zeros(1,L1+L2+L3-2);
P2=zeros(1,L1+L2+L3-2);
P3=zeros(1,L1+L2+L3-2);


P1(1:L1)=D1;
P2(1:L2)=D2;
P3(1:L3)=D3;



for n=1:(L1+L2+L3-2);
     S=0;
     for j=1:n
          for i=1:n-j+1;
               S=S+P1(i)*P2((n-j+1)-i+1)*P3(j);
          end
     end
     P(n)=S;
end
```

```
function f0=ORSk3f(Q,N,c,C,P)
%it finds the ORS for k=3

format longG


%preallocate vector&matrices
f=zeros(Q(1)+1,Q(2)+1,Q(3)+1,N);
H=zeros(Q(1)+1,Q(2)+1,Q(3)+1,N-1);
A=zeros(N-1);
R=char(zeros(Q(1)+1,Q(2)+1,Q(3)+1,N-1));


%in the boundary
f(:,:,:,N)=c(N);


for j=N-1:-1:1;

    m=0;

    %calculate A(j) sum
    for k1=0:Q(1);
       for k2=0:Q(2);
           for k3=0:Q(3)


               m=m+f(Q(1)-k1+1,Q(2)-k2+1,Q(3)-k3+1,j+1) ...
                    *P(1,k1+1)*P(2,k2+1)*P(3,k3+1);


           end
       end
    end


    %fill A(j)
    A(j)=c(j)+c(j+1)+m;


    for z1=0:Q(1);
        for z2=0:Q(2);
            for z3=0:Q(3);

            s=0;
            S=0;


            for k1=0:Q(1);
                for k2=0:Q(2);
                    for k3=0:Q(3)


                    %calculate 1st and 2d sum of H(z1,z2,z3,j)
```

```
                    if k1 <= z1&&k2 <= z2&&k3 <= z3;
                      S=S+f(z1-k1+1,z2-k2+1,z3-k3+1,j+1) ...
                          *P(1,k1+1)*P(2,k2+1)*P(3,k3+1);
                    else
                      s=s+(2*c(j+1)+f(Q(1)+min((z1-k1),0)+1,Q(2)+min((z2-
k2),0)+1,Q(3)+min((z3-k3),0)+1,j+1)) ...
                          *P(1,k1+1)*P(2,k2+1)*P(3,k3+1);
                    end


                     end
                 end
             end



         %fill H(z1,z2,j) and f(z1,z2,j)
         H(z1+1,z2+1,z3+1,j)=C(j)+S+s;

         f(z1+1,z2+1,z3+1,j)=min(A(j),H(z1+1,z2+1,z3+1,j));


         %fill R(z1,z2,j)
         if A(j)>H(z1+1,z2+1,z3+1,j);
         R(z1+1,z2+1,z3+1,j)='N';
         else
         R(z1+1,z2+1,z3+1,j)='D';
         end

             end
         end
     end
end

S=0;

%calculate minimum expected cost during a cycle
for k1=0:Q(1);
     for k2=0:Q(2);
         for k3=0:Q(3);


            S=S+f(Q(1)-k1+1,Q(2)-k2+1,Q(3)-k3+1,1) ...
            *P(1,k1+1)*P(2,k2+1)*P(3,k3+1);


         end
     end
end

f0=c(1)+S;
```

```
function P=psbltB2(Q,p)

% %it creates possibility maxtrix for each p1 and p2

%p:vector contains p1 p2 p3 etc
%k:number of types
%Q:capacity vector
%Q(1):capacity of the first product
%Q(3):capacity of the third product

%P(1,3)=possibility the first type has demand 2
%P(2,1)=possibility the second type has demand 0
%P(t,Q+1)=possibility the t type has demand Q


P=zeros(length(Q),max(Q)+1);


for i=1:length(Q)
    for k=0:Q(i)

        P(i,k+1)=binopdf(k,Q(i),p(i)*0.1);

    end
end



end


tic


%costumers
N=10;

%p vector for B distribution
p=[0.4,0.6];


%capacity vector
Q=[10,20];

%fill distance matrix
c=randi([19,25],30,N);
C=randi([18,27],30,N-1);


%plots
Qdependance30(N,p,c,C);
Ndependance30(Q,p);
pdependence30(N,Q)
[A,B,DF]=p1p2dependance30(N,Q,c,C);
```

```
toc


function Ndependance30(Q,p)
%%plot graphs AvΔf0%-N and maxΔf0%-N

%call function in order to fill the possibility matrix
P=psbltB2(Q,p);


%preallocate vector&matrices
f0k2ask1=zeros(30,20);
f0k2=zeros(30,20);


%fill distance matrix
c=randi([19,25],30,100);
C=randi([18,27],30,99);



for i=1:30

   for n=1:20;


   %call function so as to find TEC k2ask1
   f0k2ask1(i,n)=ORSk2ask1f(Q,(n*5),c(i,:),C(i,:),P);

   %call function so as to find TEC k2
   f0k2(i,n)=ORSk2f(Q,(n*5),c(i,:),C(i,:),P);


   end

end



DF=((f0k2ask1-f0k2)./f0k2)*100;

A=sum(DF,1)/30;

B=max(DF,[],1);


%create plots
figure(1)
C=5:5:100;
plot(C,A,'b*--')
grid on
xlabel('N')
ylabel('Δf0%average')
```

```
title('Δf0%average-N for k2ask1')

figure(2)
D=5:5:100;
plot(D,B,'r*--')
grid on
xlabel('N')
ylabel('Δf0%max')
title('Δf0%max-N for k2ask1')




end


function [A,B,DF]=p1p2dependance30(N,Q,c,C)
%%plot graphs AvΔf0%-p1p2 and maxΔf0%-p1p2


%preallocate vector&matrices
f0k2ask1=zeros(9,9,30);
f0k2=zeros(9,9,30);


for i=1:30


    for p1=1:9;
          for p2=1:9;


          p=[p1*0.1,p2*0.1];

          %call function in order to fill the possibility matrix
          P=psbltB2(Q,p);

          %call function so as to find TEC k2ask1
          f0k2ask1(p1,p2,i)=ORSk2ask1f(Q,N,c(i,:),C(i,:),P);

          %call function so as to find TEC k2
          f0k2(p1,p2,i)=ORSk2f(Q,N,c(i,:),C(i,:),P);


          end
    end


end


DF=((f0k2ask1-f0k2)./f0k2)*100;

A=sum(DF,3)/30;
```

```
B=max(DF,[],3);


%create plots
figure(1)
bar3(A)
grid on
xlabel('p2')
ylabel('p1')
zlabel('Δf%average')
title('Δf0%average-p1-p2 for k2 as k1')


figure(2)
bar3(B)
grid on
xlabel('p2')
ylabel('p1')
zlabel('Δfmax%')
title('Δf0%max-p1-p2 for k2 as k1')


end



function pdependence30(N,Q)
%%plot graph Δf0%-p


%fill cost vectors in a way that they satisfy triangle inequality
c=randi([19,25],30,N);
C=randi([18,27],30,N-1);


%preallocate vector&matrices
f0k2ask1=zeros(30,9);
f0k2=zeros(30,9);

for i=1:30

    for p=1:9;

    %call function in order to fill the possibility matrix
    P=psbltB2(Q,[p*0.1,p*0.1]);

    %call function so as to find TEC k2ask1
    f0k2ask1(i,p)=ORSk2ask1f(Q,N,c(i,:),C(i,:),P);

    %call function so as to find TEC k2
    f0k2(i,p)=ORSk2f(Q,N,c(i,:),C(i,:),P);

    end
```

```
end


DF=((f0k2ask1-f0k2)./f0k2)*100;

A=sum(DF,1)/30;

B=max(DF,[],1);


%create plots
figure(1)
C=0.1:0.1:0.9;
plot(C,A,'b*--')
grid on
xlabel('p')
ylabel('Δf0%average')
title('Δf0%average-p for k2 as k1')

figure(2)
D=0.1:0.1:0.9;
plot(D,B,'r*--')
grid on
xlabel('p')
ylabel('Δf0%max')
title('Δf0%max-p for k2 as k1')


end



function Qdependance30(N,p,c,C)
%%plot graphs AvΔf0%-Qtotal and maxΔf0%-Qtotal


%preallocate vector&matrices
f0k2ask1=zeros(30,4);
f0k2=zeros(30,4);



for i=1:30;

    Q=[0,10];

    for q=1:4;

    Q=Q+10;

    %call function in order to fill the possibility matrix
    P=psbltB2(Q,p);

    %call function so as to find TEC k2ask1
```

```
    f0k2ask1(i,q)=ORSk2ask1f(Q,N,c(i,:),C(i,:),P);

    %call function so as to find TEC k2
    f0k2(i,q)=ORSk2f(Q,N,c(i,:),C(i,:),P);

    end

    fprintf('%d\n',i)


end



DF=((f0k2ask1-f0k2)./f0k2)*100;

A=sum(DF,1)/30;

B=max(DF,[],1);



%create plots
figure(1)
C=30:20:90;
plot(C,A,'b*--')
grid on
xlabel('QTotal')
ylabel('Δf0%average')
title('Δf0%average-Qtotal for k2ask1')

figure(2)
D=30:20:90;
plot(D,B,'r*--')
grid on
xlabel('QTotal')
ylabel('Δf0%max')
title('Δf0%max-Qtotal for k2ask1')



end



tic


%costumers
N=10;

%p vector for B distribution
p=[0.4,0.6,0.7];
```

```
%capacity vector
Q=[5,10,15];

%fill distance matrix
c=randi([19,25],30,N);
C=randi([18,27],30,N-1);


%plots
%Qdependance30(N,p,c,C);
%Ndependance30(Q,p);
pdependence30(N,Q)
%[A,B,DF]=p1p2p3dependance30(N,Q,c,C);


toc



function P=adD3(D1,D2,D3)
%it adds D1+D2+D3 and returns combined vector P

%D1:possibility-demand vector 1st product
%D2:possibility-demand vector 2d product
%D3:possibility-demand vector 3d product


L1=length(D1);
L2=length(D2);
L3=length(D3);


P=zeros(1,L1+L2+L3-2);
P1=zeros(1,L1+L2+L3-2);
P2=zeros(1,L1+L2+L3-2);
P3=zeros(1,L1+L2+L3-2);


P1(1:L1)=D1;
P2(1:L2)=D2;
P3(1:L3)=D3;



for n=1:(L1+L2+L3-2);
    S=0;
    for j=1:n
        for i=1:n-j+1;
            S=S+P1(i)*P2((n-j+1)-i+1)*P3(j);
        end
    end
    P(n)=S;
end
```

```
function Ndependance30(Q,p)
%%plot graphs AvΔf0%-N and maxΔf0%-N

%call function in order to fill the possibility matrix
P=psbltB2(Q,p);


%preallocate vector&matrices
f0k3ask1=zeros(30,20);
f0k3=zeros(30,20);


%fill distance matrix
c=randi([19,25],30,100);
C=randi([18,27],30,99);



for i=1:30

    for n=1:20;


    %call function so as to find TEC k2ask1
    f0k3ask1(i,n)=ORSk3ask1f(Q,(n*5),c(i,:),C(i,:),P);

    %call function so as to find TEC k2
    f0k3(i,n)=ORSk3f(Q,(n*5),c(i,:),C(i,:),P);


    end

    fprintf('%d\n',i)

end



DF=((f0k3ask1-f0k3)./f0k3)*100;

A=sum(DF,1)/30;

B=max(DF,[],1);


%create plots
figure(1)
C=5:5:100;
plot(C,A,'b*--')
grid on
xlabel('N')
ylabel('Δf0%average')
title('Δf0%average-N for k3ask1')
```

```
figure(2)
D=5:5:100;
plot(D,B,'r*--')
grid on
xlabel('N')
ylabel('Δf0%max')
title('Δf0%max-N for k3ask1')




end


function [A,B,DF]=p1p2p3dependance30(N,Q,c,C)
%%plot graphs AvΔf0%-p1p2 and maxΔf0%-p1p2


%preallocate vector&matrices
f0k3ask1=zeros(9,9,9,30);
f0k3=zeros(9,9,9,30);


for i=1:30


    for p1=1:9;
        for p2=1:9;
            for p3=1:9;


        p=[p1*0.1,p2*0.1,p3*0.1];

        %call function in order to fill the possibility matrix
        P=psbltB2(Q,p);

        %call function so as to find TEC k2ask1
        f0k3ask1(p1,p2,p3,i)=ORSk3ask1f(Q,N,c(i,:),C(i,:),P);

        %call function so as to find TEC k2
        f0k3(p1,p2,p3,i)=ORSk3f(Q,N,c(i,:),C(i,:),P);


            end

        end
    end

   fprintf('%d\n',i)

end


DF=((f0k3ask1-f0k3)./f0k3)*100;
```

```
A=sum(DF,4)/30;

B=max(DF,[],4);


end


function pdependence30(N,Q)
%%plot graph Δf0%-p


%fill cost vectors in a way that they satisfy triangle inequality
c=randi([19,25],30,N);
C=randi([18,27],30,N-1);


%preallocate vector&matrices
f0k3askl=zeros(30,9);
f0k3=zeros(30,9);



for i=1:30

    for p=1:9;

    %call function in order to fill the possibility matrix
    P=psbltB2(Q,[p*0.1,p*0.1,p*0.1]);

    %call function so as to find TEC k2askl
    f0k3askl(i,p)=ORSk3asklf(Q,N,c(i,:),C(i,:),P);

    %call function so as to find TEC k2
    f0k3(i,p)=ORSk3f(Q,N,c(i,:),C(i,:),P);

    end

    %fprintf('%d\n',i)

end



DF=((f0k3askl-f0k3)./f0k3)*100;

A=sum(DF,1)/30;

B=max(DF,[],1);
```

```
%create plots
figure(1)
C=0.1:0.1:0.9;
plot(C,A,'b*--')
grid on
xlabel('p')
ylabel('Δf0%average')
title('Δf0%average-p for k3 as k1')

figure(2)
D=0.1:0.1:0.9;
plot(D,B,'r*--')
grid on
xlabel('p')
ylabel('Δf0%max')
title('Δf0%max-p for k3 as k1')


end



function Qdependance30(N,p,c,C)
%%plot graphs AvΔf0%-Qtotal and maxΔf0%-Qtotal


%preallocate vector&matrices
f0k3ask1=zeros(10,4);
f0k3=zeros(10,4);



for i=1:30;

    Q=[0,5,10];

    for q=1:4;

    Q=Q+5;

    %call function in order to fill the possibility matrix
    P=psbltB2(Q,p);

    %call function so as to find TEC k2ask1
    f0k3ask1(i,q)=ORSk3ask1f(Q,N,c(i,:),C(i,:),P);

    %call function so as to find TEC k2
    f0k3(i,q)=ORSk3f(Q,N,c(i,:),C(i,:),P);

    end

    fprintf('%d\n',i)
```

```
end

DF=((f0k3ask1-f0k3)./f0k3)*100;

A=sum(DF,1)/30;

B=max(DF,[],1);

%create plots
figure(1)
C=25:15:75;
plot(C,A,'b*--')
grid on
xlabel('QTotal')
ylabel('Δf0%average')
title('Δf0%average-Qtotal for k3 as k1')

figure(2)
D=25:15:75;
plot(D,B,'r*--')
grid on
xlabel('QTotal')
ylabel('Δf0%max')
title('Δf0%max-Qtotal for k3 as k1')

end
```