ΑΡΙΣΤΟΤΕΛΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΟΝΙΚΗΣ
ΤΜΗΜΑ ΜΑΘΗΜΑΤΙΚΩΝ
ΜΕΤΑΠΤΥΧΙΑΚΟ ΠΡΟΓΡΑΜΜΑ ΣΠΟΥΔΩΝ
"ΘΕΩΡΗΤΙΚΗ ΠΛΗΡΟΦΟΡΙΚΗ & ΘΕΩΡΙΑ ΣΥΣΤΗΜΑΤΩΝ ΚΑΙ ΕΛΕΓΧΟΥ"

Ανάπτυξη διαδραστικού περιβάλλοντος (GUI) μέσω του Matlab για τον υπολογισμό της παραμετρικής οικογένειας των αντισταθμιστών οι οποίοι εξασφαλίζουν τον προσδιορισμό επιθυμητού «πίνακα παρανομαστή» του προκύπτοντος μέσω ανάδρασης κλειστού συστήματος με σκοπό την σταθεροποίηση ασταθών γραμμικών πολυμεταβλητών συστημάτων.

ΜΕΤΑΠΤΥΧΙΑΚΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Γεώργιος Χρ. Μακρής

Επιβλέπων: Α. Βαρδουλάκης
Καθηγητής Α.Π.Θ.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή στις **15 / 12 / 2008**

…………………………	…………………………	…………………………
. Α. Βαρδουλάκης	Μ. Γουσίδου	Ν. Καραμπετάκης
Καθηγητής Α.Π.Θ.	Αν. Καθηγήτρια Α.Π.Θ	Επικ. Καθηγητής Α.Π.Θ

Θεσσαλονίκη, Δεκέμβριος 2008

……………………………………………..

Γεώργιος Χρ. Μακρής
Πτυχιούχος Μαθηματικός - ΑΠΘ

Περιεχόμενα

Περίληψη

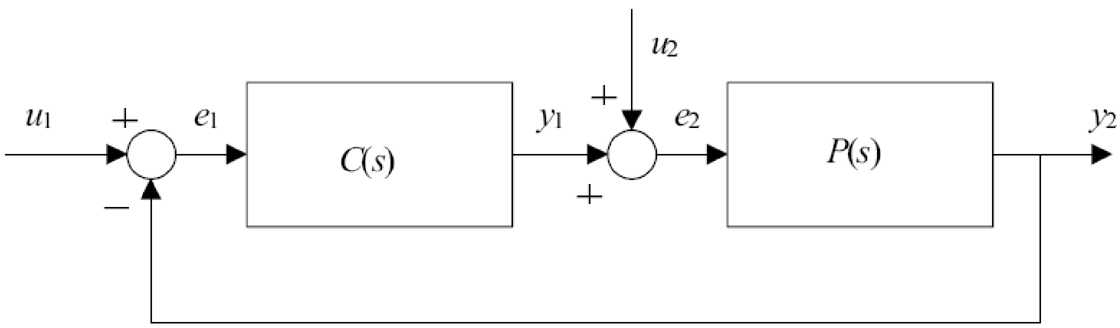
Η παρούσα διπλωματική εργασία έχει ως στόχο την ανάπτυξη ενός διαδραστικού περιβάλλοντος για την εύρεση ενός κατάλληλου αντισταθμιστή – ελεγκτή με τον οποίο θα μπορέσουμε να έχουμε την επιθυμητή έξοδο του συστήματος . Η ανάπτυξη του διαδραστικού περιβάλλοντος υλοποιήθηκε με την χρήση του graphical user interface του λογισμικού πακέτου MATLAB. Το πλεονέκτημα χρήσης ενός παραθυρικού περιβάλλοντος σε μια εφαρμογή είναι πολύ μεγάλο, επειδή η εφαρμογή γίνεται πιο φιλική στον τελικό χρηστή και δεν είναι απαραίτητη προϋπόθεση ο χρήστης να γνωρίζει αρκετά καλά το λογισμικό πακέτο MATLAB. Παράλληλα για την ανάπτυξη της εφαρμογής χρησιμοποιήθηκε το πακέτο POLYX.

Η εργασία αποτελείται από τρία κεφαλαία εκ των οποίων στο πρώτο γίνεται μια αναφορά στο θεωρητικό υπόβαθρο στο οποίο στηρίζεται η εύρεση του αντισταθμιστή , στο δεύτερο γίνεται μια εισαγωγή στο graphical user interfaces του MATLAB καθώς και στο πακέτο polyx και τέλος στο τρίτο γίνεται μια λεπτομερή περιγραφή της εφαρμογής.

1. Θεωρητικό Υπόβαθρο

1.1 Καθορισμός του προβλήματος

Θεωρούμε ένα γραμμικό και χρονικά αμετάβλητο πολυμεταβλητό σύστημα p x m , όπου p: ο αριθμός των εξόδων και m: ο αριθμός των εισόδων. Το σύστημα αυτό έχει ένα strictly proper plant P(s) . Θεωρούμε την δεξιά πρώτη πολυωνυμική περιγραφή $P(s) = N_R(s) \cdot D_R(s)^{-1}$. Ορίζουμε την κλάση $\Phi(P(s), D_C(s))$ όλων των κανονικών (proper) αντισταθμιστών C(s) για τους οποίους έχει επιλεγεί ένας επιθυμητός παρανομαστής του κλειστού συστήματος του παρακάτω σχήματος :



Η κλάση $\Phi(P(s), D_C(s))$ αποτελείται από ζεύγη πολυωνυμικών πινάκων :

$$X_L(s) \in \mathbb{R}[s]^{mxm} \quad , \quad Y_L(s) \in \mathbb{R}[s]^{mxp}$$

οι οποίοι ικανοποιούν την διοφαντική εξίσωση :

$$X_L(s) \cdot D_R(s) + Y_L(s) \cdot N_R(s) = D_C(s) \qquad (1.1.1)$$

Τότε ο ζητούμενος αντισταθμιστής του συστήματος θα είναι :

$$C(s) = X_L(s)^{-1} \cdot Y_L(s) \in \mathbb{R}_{pr}[s]^{mxp} \qquad (1.1.2)$$

1.2 Ανάλυση του προβλήματος

Αν θεωρήσουμε τους πίνακες :

$$\Omega(s) := \begin{bmatrix} X_L(s) & Y_L(s) \end{bmatrix} \in \mathbb{R}[s]^{mx(pxm)} \quad \text{και} \quad F(s) := \begin{bmatrix} D_R(s) \\ N_R(s) \end{bmatrix} \in \mathbb{R}[s]^{(pxm)xm}$$

τότε η εξίσωση : $X_L(s) \cdot D_R(s) + Y_L(s) \cdot N_R(s) = D_C(s)$ γίνεται :

$$\begin{bmatrix} X_L(s) & Y_L(s) \end{bmatrix} \cdot \begin{bmatrix} D_R(s) \\ N_R(s) \end{bmatrix} = D_C(s) \quad \Leftrightarrow \quad \Omega(s) \cdot F(s) = D_C(s) \qquad (1.2.1)$$

Αν θεωρήσουμε ότι k-1 είναι ο μέγιστος βαθμός πολυωνύμων που υπάρχει στον πίνακα Ω(s) , τότε μπορούμε να γράψουμε :

$$\Omega(s) = \overline{\Omega}_k \cdot S_{m+p,k}(s) = \begin{bmatrix} \Omega_0 & \Omega_1 & \dots & \Omega_{k-1} \end{bmatrix} \cdot \begin{bmatrix} I_{m+p} \\ s \cdot I_{m+p} \\ \dots \\ s^{k-1} \cdot I_{m+p} \end{bmatrix} = \Omega_0 + \Omega_1 \cdot s + \Omega_2 \cdot s^2 + \dots + \Omega_{k-1} \cdot s^{k-1}$$

με : $\Omega_i \in \mathbb{R}^{mx\,(m+p)} \quad ,i = 0,1,\dots,k-1 \quad \kappa\alpha\iota \;\; \overline{\Omega}_k = \begin{bmatrix} \Omega_0 & \Omega_1 & \dots & \Omega_{k-1} \end{bmatrix} \mathbb{R}^{mxk(m+p)}$

Από τον ορισμό της γενικευμένης Resultant του Wolovich $M_k^{F(s)}$ του πίνακα F(s) τάξης k ισχύει :

$$S_{m+p,k}(s) \cdot F(s) = M_k^{F(s)} \cdot b\lo ckdiag\left\{ S_{1,k_j+k}(s) \right\} =$$

$$= M_k^{F(s)} \begin{bmatrix} 1 & & & & \\ s & & & & \\ \dots & & & & \\ s^{k_1+k-1} & & & & \\ & 1 & & & \\ & s & & & \\ & \dots & & & \\ & s^{k_2+k-1} & & & \\ & & \dots & & \\ & & \dots & & \\ & & \dots & & \\ & & & 1 \\ & & & s \\ & & & \dots. \\ & & & s^{k_m+k-1} \end{bmatrix}$$

Οπότε η διοφαντική εξίσωση (1.2.1) :

$\left[X_L(s)\ \ Y_L(s) \right] \cdot \begin{bmatrix} D_R(s) \\ N_R(s) \end{bmatrix} = D_C(s) \quad \Leftrightarrow \quad \Omega(s) \cdot F(s) = D_C(s)$ γίνεται :

$$\left[X_L(s)\ \ Y_L(s) \right] \cdot \begin{bmatrix} D_R(s) \\ N_R(s) \end{bmatrix} = D_C(s) \quad \Leftrightarrow \quad \Omega(s) \cdot F(s) = D_C(s) \quad \Leftrightarrow$$

$$\overline{\Omega}_k \cdot S_{m+p,k}(s) \cdot F(s) = D_C(s) \quad \Leftrightarrow$$

$$\overline{\Omega}_k \cdot M_k^{F(s)} \begin{bmatrix} 1 & & & & & \\ s & & & & & \\ \ldots & & & & & \\ s^{k_1+k-1} & & & & & \\ & 1 & & & & \\ & s & & & & \\ & \ldots & & & & \\ & s^{k_2+k-1} & & & & \\ & & \ldots & & & \\ & & \ldots & & & \\ & & \ldots & & & \\ & & & 1 & & \\ & & & s & & \\ & & & \ldots. & & \\ & & & s^{k_m+k-1} & & \end{bmatrix} = D_C(s) \quad \Leftrightarrow$$

$$\overline{D}_k \cdot \begin{bmatrix} 1 & & & \\ s & & & \\ \ldots & & & \\ s^{k_1+k-1} & & & \\ & 1 & & \\ & s & & \\ & \ldots & & \\ & s^{k_2+k-1} & & \\ & & \ldots & \\ & & \ldots & \\ & & \ldots & \\ & & & 1 \\ & & & s \\ & & & \ldots. \\ & & & s^{k_m+k-1} \end{bmatrix} = D_C(s) \qquad (1.2.2)$$

Όπου :

$$\overline{D}_k = \overline{\Omega}_k \cdot M_k^{F(s)} \quad (1.2.3)$$

$$\acute{o}\pi o \upsilon \ \overline{D}_k \in \mathbb{R}^{m x \left(mk + \sum_{j=1}^{m} k_j \right)}$$

Για να φτάσουμε στη λύση $\Omega(s) := \left[X_L(s) \ \ Y_L(s) \right] \in \mathbb{R}[s]^{mx(pxm)}$ της διοφαντικής

εξίσωσης $\Omega(s) \cdot F(s) = D_C(s)$ πρέπει να λύσουμε την αριθμητική εξίσωση :

$$\overline{D}_k = \overline{\Omega}_k \cdot M_k^{F(s)} \quad \mu\varepsilon \ \overline{D}_k \in \mathbb{R}^{m x \left(mk + \sum_{j=1}^{m} k_j \right)}$$

Εφόσον πρώτα :

1. μας δοθεί ένας κατάλληλος παρονομαστής $D_C(s)$ του κλειστού συστήματος και
2. μας δοθεί ο μέγιστος βαθμός k-1 του Ω(s) , δηλαδή το k στην Resultant του Wolovich $M_k^{F(s)}$
3. υπολογίσουμε τον πίνακα $\overline{D}_k$ από την σχέση :

$$\overline{D}_k \cdot \underset{j=1,2,\ldots,m}{b\,\text{lo}\,ckdiag} \left\{ S_{1,k_j+k}(s) \right\} = D_C(s) \quad (1.2.2)$$

Κάτω από αυτές τις προϋποθέσεις ο πίνακας XL(s) (ο πίνακας παρανομαστής του αντισταθμιστή C(s)) είναι κανονικός ως προς τις γραμμές του (row reduced) με βαθμούς γραμμών $\deg_{ri} X_L(s) = \xi_i \ , \ i = 1, 2, \ldots, m.$

Συμβολίζοντας με :

$\omega_\iota^\tau(s) \ \in \ \mathbb{R}[s]^{1x(m+p)}$ τις γραμμές του πίνακα Ω(s),

$\overline{\omega}_\iota^\tau \ \in \ \mathbb{R}^{1x(\xi_\iota+1)(m+p)}$ τις γραμμές του πίνακα $\overline{\Omega}_k$,

$d_\iota^\tau(s) \ \in \ \mathbb{R}^{1xm}$ τις γραμμές του πίνακα $D_C(s)$

και $\overline{d}_\iota^\tau(s) \ \in \ \mathbb{R}^{1x\left(m(\xi_\iota+1) + \sum_{j=1}^{m} k_j \right)}$ τις γραμμές του πίνακα $\overline{D}_k$, προκύπτουν οι εξισώσεις :

$$d_\iota^\tau(s) = \overline{d}_\iota^\tau(s) \cdot \underset{j=1,2,\ldots,m}{blockdiag}\left\{S_{1,k_j+\xi_i+1}(s)\right\}, \quad i = 1,2,\ldots,m. \quad (1.2.4)$$

$$\omega_\iota^\tau(s)F(s) = d_\iota^\tau(s), \quad i = 1,2,\ldots,m. \quad (1.2.5)$$

$$\omega_\iota^\tau(s) = \overline{\omega}_\iota^\tau S_{m+p,\xi_i+1}(s), \quad i = 1,2,\ldots,m. \quad (1.2.6)$$

και

$$\boxed{\overline{\omega}_\iota^\tau \mathrm{M}_{\xi_{\iota+1}}^{F(S)} = \overline{d}_\iota^\tau}, \quad i = 1,2,\ldots,m. \quad (1.2.7)$$

1.3 Αλγόριθμος υπολογισμού Αντισταθμιστή

Βήμα 1^{o}

Βρίσκουμε διαδοχικά τις γενικευμένες Resultant του Wolovich $M_k^{F(s)}$ του πίνακα F(s) τάξης k, μέχρις ότους η $M_k^{F(s)}$ να αποκτήσει πλήρη τάξη γραμμών (full row rank) , δηλαδή μέχρι να βρεθεί το k στο οποίο : $rank_R M_k^{F(s)} = mk + \sum_{i=1}^{m} k_i$. Το ελάχιστο αυτό k για το οποίο ισχύει η παραπάνω σχέση είναι ο δείκτης παρατηρησιμότητας του συστήματος (observability index) και το συμβολίζουμε με μ.

Βήμα 2^{o}

Επιλέγουμε ξ_i τα οποία : $\xi_i \geq \mu - 1,, \quad i = 1, 2, ..., m.$

Βήμα 3^{o}

Επιλέγουμε τον επιθυμητό παρονομαστή $D_C(s)$ του κλειστού συστήματος (mxm διάστασης) ώστε να πληρούνται οι παρακάτω προϋποθέσεις :

1. Να είναι row and column reduced (κανονικός ως προς τις γραμμές και τις στήλες του) με ίσους βαθμούς των γραμμών και των στηλών δηλαδή

$$\deg_{ci} D_C(s) = \deg_{ri} D_C(s) = \xi_i + k_i \quad , \quad i = 1, 2, ..., m$$

2. Να έχει τους επιθυμητούς πόλους εντός του ανοιχτού αριστερού μιγαδικού ημιεπιπέδου .

Για την απλότητα των υπολογισμών μπορεί να επιλεχτεί ο πίνακας $D_C(s)$ να είναι διαγώνιος.

Βήμα 4^o

Λύνουμε τις εξισώσεις : $\boxed{\overline{\omega}_i^{\tau} \mathrm{M}_{\xi_{i+1}}^{F(S)} = \overline{d}_i^{\tau}}$, $i = 1, 2, ..., m.$ (1.2.7) ως προς $\overline{\omega}_i^{\tau}$

Βήμα 5^o

Λύνουμε τις εξισώσεις : $\omega_i^{\tau}(s) = \overline{\omega}_i^{\tau} S_{m+p,\xi_i+1}(s)$, $i = 1, 2, ..., m.$ (1.2.6) ως ω_i^{τ} ,

για να βρούμε τις γραμμές του πίνακα $\Omega(s) := \left[X_L(s) \;\; Y_L(s) \right] \in \mathbb{R}[s]^{mx(pxm)}$

Βήμα 6^o

Ο ζητούμενος αντισταθμιστής του συστήματος θα είναι :

$$C(s) = X_L(s)^{-1} \cdot Y_L(s) \in \mathbb{R}_{pr}[s]^{mxp} \qquad (1.1.2)$$

2. Το Περιβάλλον Ανάπτυξης MatLab

2.1 Matlab - GRAPHICAL USER INTERFACE

Το MATLAB προσφέρει στο χρήστη τη δυνατότητα να κατασκευάσει δικές του γραφικές διεπιφάνειες. Η χρησιμότητα της λειτουργίας αυτής είναι μεγάλη, επειδή τα προγράμματα - εφαρμογές που περιέχουν γραφική διεπιφάνεια γίνονται πιο φιλικά στον τελικό χρήστη.
Η ανάπτυξη μιας εφαρμογής σε GUI χωρίζεται σε τέσσερα στάδια:

1. Θεωρητική σχεδίαση της εφαρμογής πριν την υλοποίηση της σε MATLAB.
2. Σχεδιασμός της εφαρμογής στο κατάλληλο περιβάλλον (GUIDE Layout Editor) και δημιουργία των απαιτούμενων αντικειμένων.
3. Καθορισμός ιδιοτήτων του GUI και του κάθε αντικειμένου.
4. Προγραμματισμός του κάθε αντικειμένου , εάν χρειάζεται.

Το MATLAB προσφέρει μια ικανοποιητική εργαλειοθήκη, η οποία διευκολύνει πολύ τη δημιουργία ενός GUI (ΓΔΧ). Αυτή η εργαλειοθήκη ή αλλιώς GUIDE (Graphical User Interface Design Envrionment), περιέχει μια πληθώρα χρήσιμων εργαλείων ελέγχου όπως κουμπιά, πλαίσια κ.α. Για να αξιοποιήσει πλήρως ο χρήστης τις δυνατότητες του GUIDE, είναι απαραίτητο να γνωρίζει πρώτα κάποιες βασικές εντολές της γλώσσας MATLAB.
Ο χρήστης έχει τη δυνατότητα να μη χρησιμοποιήσει τον GUIDE και αντί αυτού να δημιουργήσει μόνος του κάποιο figure προγραμματίζοντας κατάλληλα σε ένα m–file. Επίσης μπορεί να εισάγει και να προγραμματίσει τα διάφορα στοιχεία ελέγχου εκ του μηδενός.

Ένα GUI αποτελείται συνήθως από διάφορα παράθυρα, τα οποία περιέχουν ποικίλα στοιχεία ελέγχου όπως πεδία κειμένου, γραμμές κύλισης, λίστες, πεδία εισαγωγής κειμένου κ. α. Τα παράθυρα αυτά είναι δυνατόν να καλούν το ένα το άλλο, να δέχονται δεδομένα από τον χρήστη, να μεταβιβάζουν πιθανώς τα δεδομένα από το ένα παράθυρο στο άλλο και γενικά να επιτελούν διάφορες λειτουργίες. Το GUIDE για να το πετύχει αυτό δημιουργεί για κάθε νέο παράθυρο δύο αρχεία. Τα αρχεία αυτά είναι το fig - file και το m-file.

Το fig-file, ουσιαστικά είναι το παράθυρο–figure, όπου το MATLAB αποθηκεύει τα στοιχεία ελέγχου και την ακριβή θέση τους. Εδώ ο προγραμματιστής σχεδιάζει την εμφάνιση του παραθύρου.

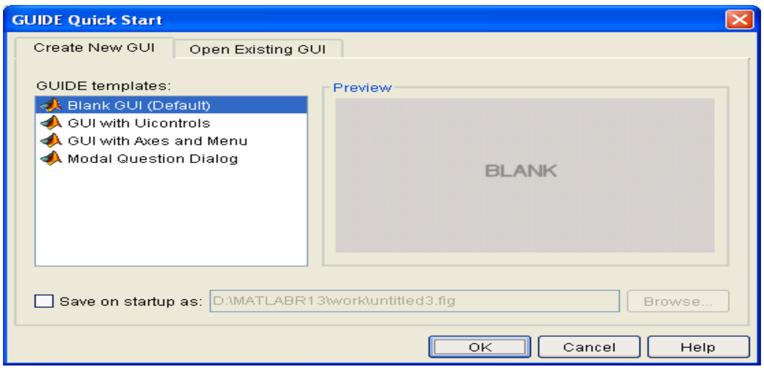
Το m-file, όπου ο προγραμματιστής πρέπει να γράψει τον κώδικα που θα ενσωματωθεί στα στοιχεία ελέγχου (π.χ. κουμπιά ή φόρμες εισαγωγής δεδομένων), ώστε αυτά να επιτελέσουν τις επιθυμητές λειτουργίες.

Πρέπει να σημειωθεί ότι κάθε αρχείο *.fig πρέπει να συνοδεύεται από το αντίστοιχο αρχείο *.m (με το ίδιο όνομα). Αν για κάποιο λόγο, χαθεί ή καταστραφεί το *.m αρχείο, τότε το παράθυρο *.fig δεν θα είναι λειτουργικό. Κάθε φορά που κάποιος χρήστης δημιουργεί ένα νέο παράθυρο (figure), το GUIDE δημιουργεί αυτομάτως και τους δυο προαναφερθέντες τύπους αρχείων. Στη συνέχεια θα περάσουμε στην πράξη, δείχνοντας πως γίνεται η εκκίνηση του GUIDE. Επίσης θα περιγράψουμε τις ιδιότητες μερικών βασικών στοιχείων ελέγχου.

Η εκκίνηση του GUIDE γίνεται εύκολα, με δυο τρόπους. Πρώτον, με την κλήση της ομώνυμης συνάρτησης από τη γραμμή εντολών του MATLAB.

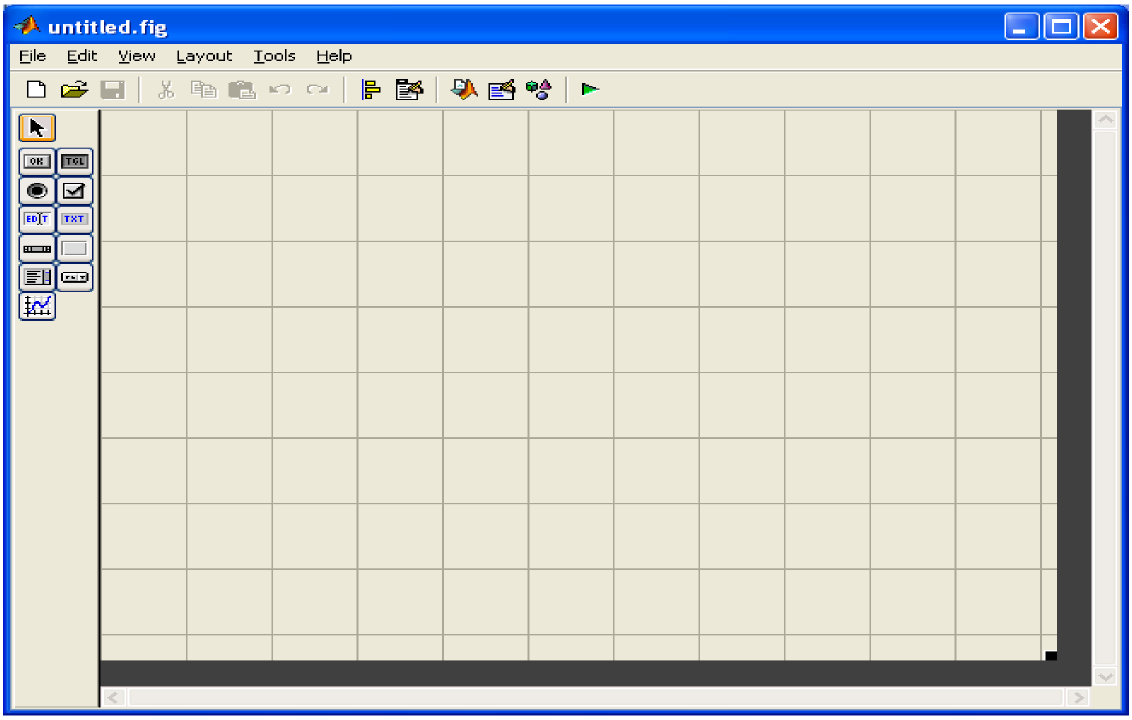
» guide

Δεύτερον, επιλέγοντας από τη γραμμή μενού File → New → GUI . Όποιο τρόπο και να επιλέξει ο χρήστης, θα ξεκινήσει ένας οδηγός που θα τον καθοδηγεί στη δημιουργία ενός παραθύρου, όπως στην εικόνα 2.1.1. Ο οδηγός ρωτάει τον χρήστη αν θέλει να δημιουργήσει ένα κενό παράθυρο (Blank GUI), αν θέλει να δημιουργήσει ένα παράθυρο βασισμένο σε κάποια πρότυπα, (π.χ. με άξονες και μενού) ή αν θέλει να ανοίξει ένα έτοιμο παράθυρο.



Εικόνα 2.1.1 Ο αρχικός οδηγός του GUIDE.

Αν ο χρήστης αφήσει την προεπιλεγμένη επιλογή και πατήσει το κουμπί OK, τότε θα δημιουργηθεί ένα νέο άδειο παράθυρο, όπως στην εικόνα 2.1.2



Εικόνα 2.1.2. Ένα νέο άδειο παράθυρο.

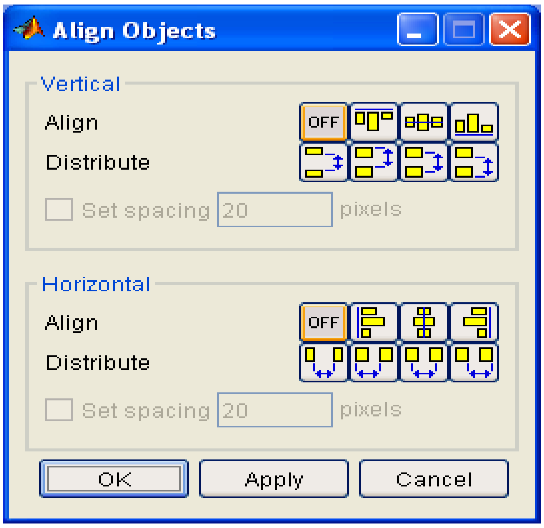
Ο χρήστης μπορεί να δει ότι το περιβάλλον δημιουργίας του παραθύρου αποτελείται από μια κεντρική γραμμή επιλογών, μια γραμμή εργαλείων, καθώς και μια κάθετη εργαλειοθήκη στα αριστερά. Η γκρίζα περιοχή με το πλέγμα το οποίο καλύπτει το παράθυρο, είναι το φόντο, η ταπετσαρία του παραθύρου.

2.1.1 Γραμμή εργαλείων

Ιδιαίτερης σημασίας είναι τα τελευταία έξι κουμπιά της γραμμής εργαλείων, που φαίνονται σε μεγέθυνση στην εικόνα 2.1.3.

Εικόνα 2.1.3. Κουμπιά από τη γραμμή εργαλείων.

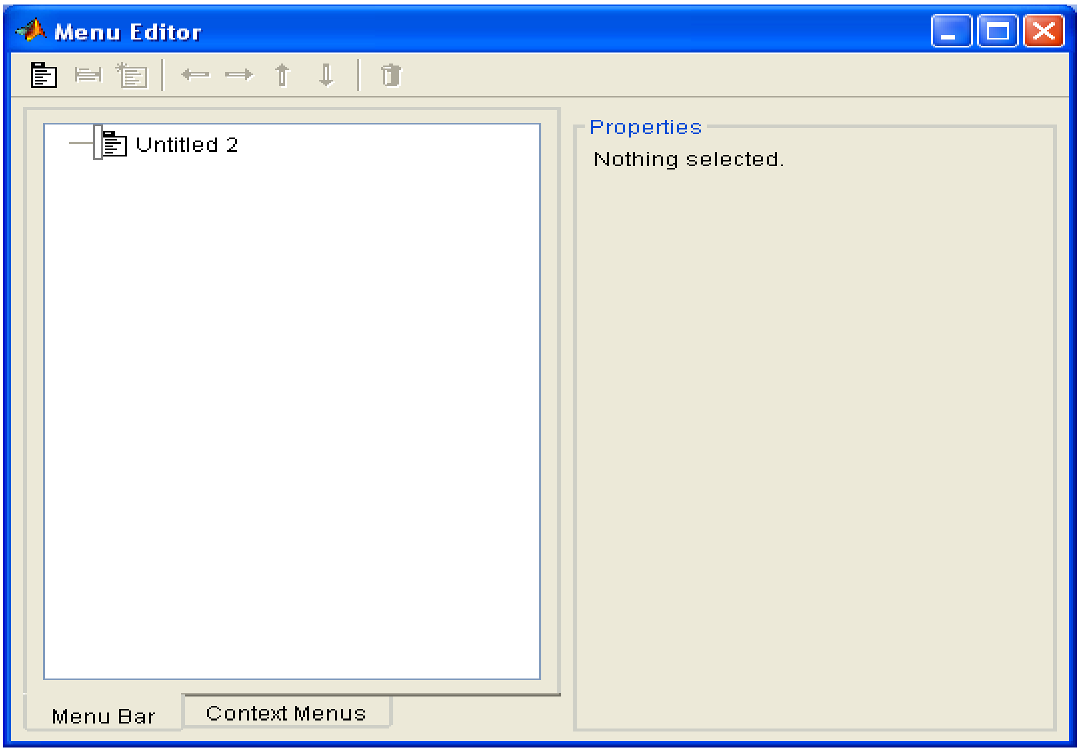
Στη συνέχεια θα δοθεί μια περιγραφή των λειτουργιών που προσφέρονται από αυτήν τη γραμμή εργαλείων. Το **πρώτο** εικονίδιο, από τα αριστερά, ενεργοποιεί το παράθυρο της εικόνας 2.1.4



Εικόνα 2.1.4. Στοίχιση αντικειμένων.

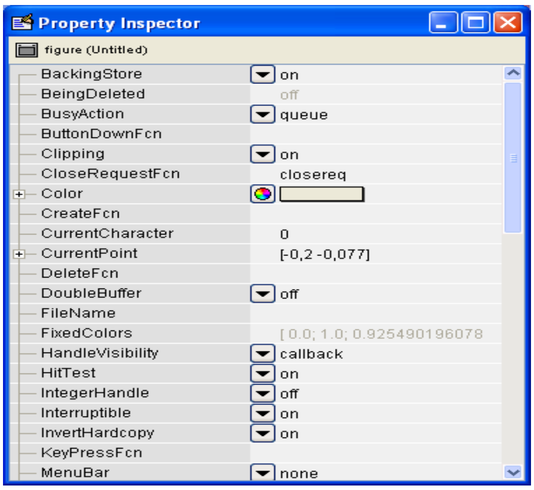
Ο χρήστης, μέσω αυτού του παραθύρου, μπορεί να στοιχίσει είτε κατακόρυφα είτε οριζόντια, όποια στοιχεία ελέγχου επιθυμεί. Κλασικό παράδειγμα είναι η εισαγωγή στοιχείων σε φόρμες, όπου συνηθίζεται τα πεδία εισαγωγής κειμένου να είναι ομοιόμορφα τοποθετημένα και όχι σε τυχαία θέση.

Το **δεύτερο** από τα κουμπιά της εικόνας 2.1.3 μας επιτρέπει να δημιουργήσουμε γραμμή μενού στην παραθυρική μας εφαρμογή. Η γραμμή μενού είναι δυνατό να περιέχει και πτυσσόμενα υπό-μενού. Στην επόμενη εικόνα 2.1.5 φαίνεται το παράθυρο που ενεργοποιείται με το πάτημα του σχετικού κουμπιού.



Εικόνα 2.1.5. Δημιουργία μενού και υπό - μενού.

Στο παράδειγμα που ακολουθεί θα εξετάσουμε αναλυτικότερα την εισαγωγή μενού ή και υπό-μενού. Το πάτημα του **τρίτου** κουμπιού της εικόνας 2.1.3 εκκινεί τον γνωστό m–file Editor, ανοίγοντας ταυτόχρονα το αντίστοιχο m-file του παραθύρου μας. Το **τέταρτο** κουμπί εκκινεί τον Property Inspector όπως φαίνεται στην εικόνα 2.1.6 .



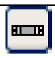
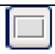
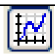
Εικόνα 2.1.6. Ρύθμιση ιδιοτήτων των στοιχείων ελέγχου.

Από το προηγούμενο παράθυρο, και έχοντας επιλέξει πρώτα κάποιο στοιχείο ελέγχου, ή και το ίδιο το παράθυρο της εφαρμογής μας, αλλάζουμε όποιες ιδιότητες επιθυμούμε. Το **πέμπτο** κουμπί της εικόνας 2.1.3 ανοίγει ένα παράθυρο, που μας δείχνει πόσα και ποια αντικείμενα υπάρχουν στο συγκεκριμένο παράθυρο της εφαρμογής μας. Τέλος, το **έκτο** κουμπί της εικόνας 2.1.3 εκτελεί την εφαρμογή μας αλλά ταυτόχρονα μας ενημερώνει ότι πρέπει να αποθηκευτεί προηγουμένως.

2.1.2 Εργαλειοθήκη

Η εργαλειοθήκη που βρίσκεται κάθετα αριστερά στην εικόνα 2.1.2, περιέχει όλα τα στοιχεία ελέγχου, τα οποία είναι διαθέσιμα στον χρήστη.

Η εισαγωγή τους στο παράθυρο είναι απλή και γίνεται με απλό σύρσιμο (drag and drop). Στην συνεχεία θα αναφερθούμε εκτενέστερα για κάθε στοιχείο ελέγχου.

	Select
	Push Button
	Toggle Button
	Radio Button
	Checkbox
	Edit Text
	Static Text
	Slider
	Frame
	List box
	Popup Menu
	Axes

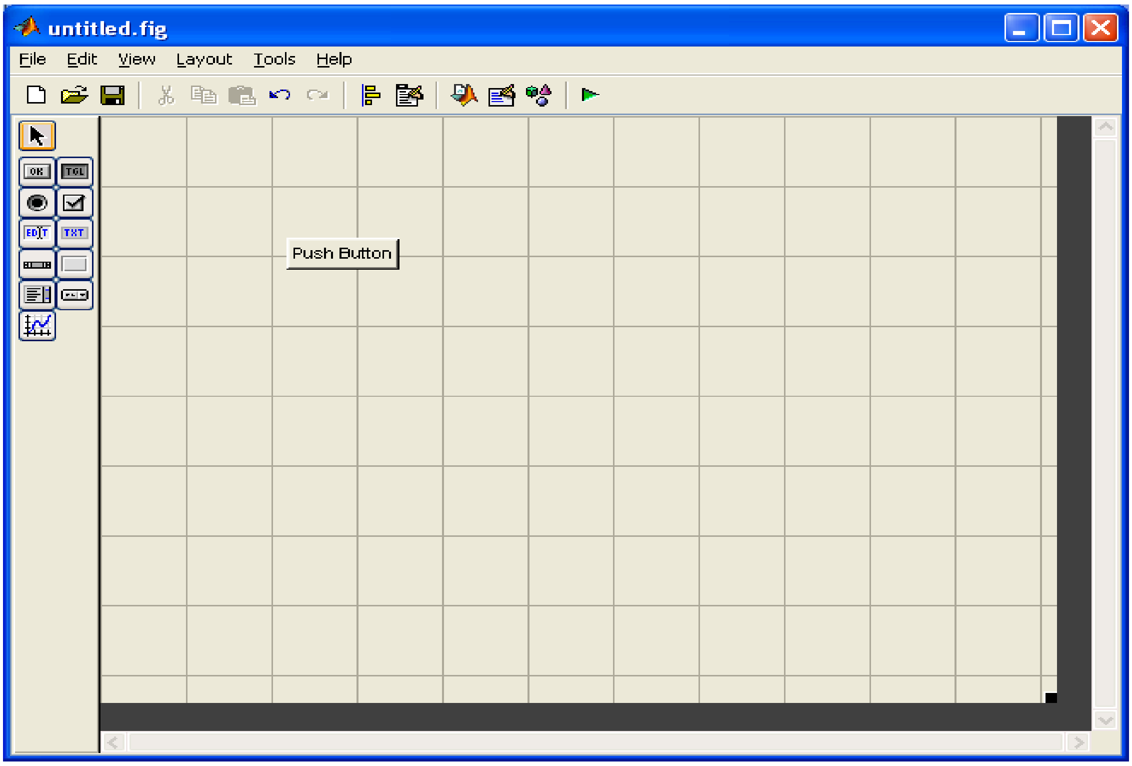
Πίνακας στοιχείων ελέγχου

2.1.2.1 Select 

Όταν είναι ενεργοποιημένο το κουμπί επιλογής έχουμε τη δυνατότητα να επιλέξουμε οποιοδήποτε στοιχείο ελέγχου, αλλά και να κάνουμε πολλαπλή επιλογή έτσι ώστε να δημιουργήσουμε μια ομάδα στοιχείων ελέγχου.

2.1.2.2 Push Button 

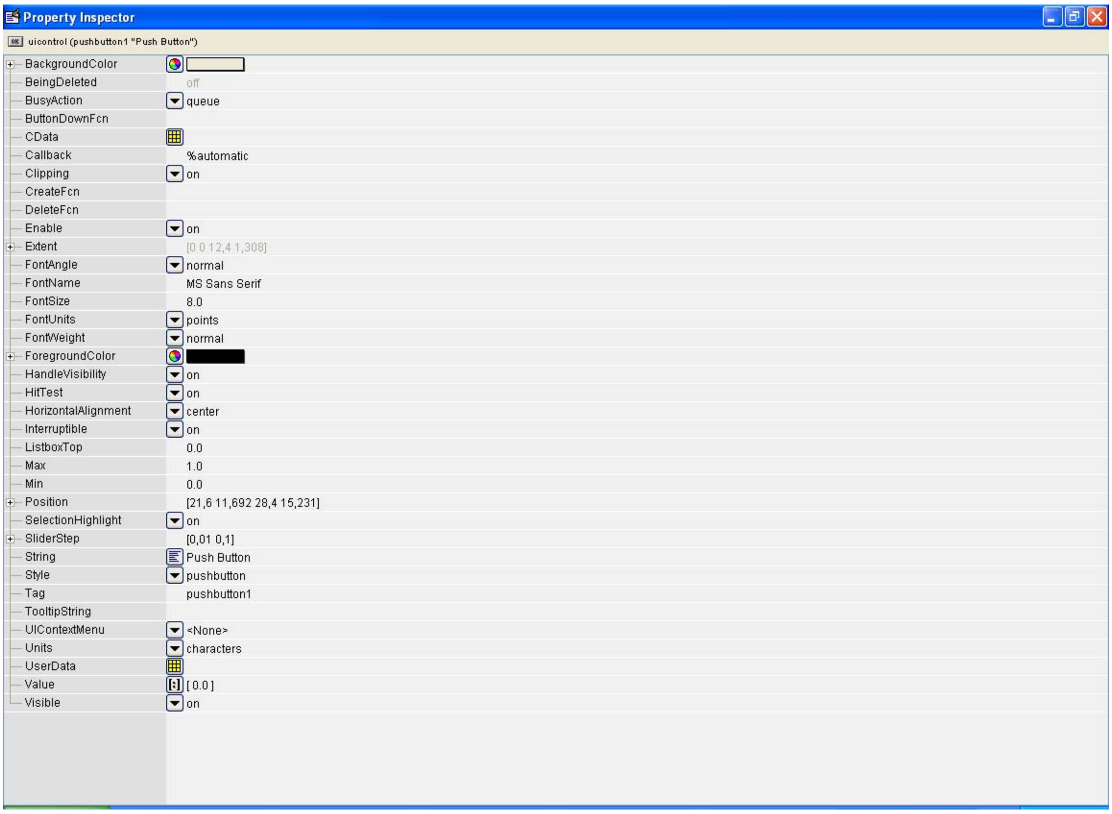
Αφού εισάγουμε στο παράθυρο ένα push button τότε αυτόματα δημιουργείται ένα, σε συγκεκριμένες διαστάσεις, το οποίο έχει κάποιες ιδιότητες.



Εικόνα 2.1.7. Παράθυρο με το στοιχείο ελέγχου push button.

Χρησιμοποιώντας τον Property Inspector μπορούμε να αλλάξουμε κάποιες ιδιότητες. Στον παρακάτω πίνακα παρουσιάζονται οι ιδιότητες που είναι κοινές σε όλα τα στοιχεία ελέγχου και χρησιμοποιούνται περισσότερο. Στην πρώτη στήλη έχουμε το όνομα της ιδιότητας, στην δεύτερη στήλη έχουμε τις επιλογές για κάθε ιδιότητα και στην τρίτη στήλη έχουμε την περιγραφή.

Στις επιλογές έχουμε τις προκαθορισμένες τιμές (default) για κάθε ιδιότητα ή, όταν μπορούμε να διαλέξουμε μεταξύ κάποιων επιλογών, σε άγκιστρα { } είναι η προκαθορισμένη επιλογή (default) ενώ σε παρενθέσεις ( ) είναι οι άλλες επιλογές .



ΙΔΙΟΤΗΤΑ		ΕΠΙΛΟΓΕΣ	ΠΕΡΙΓΡΑΦΗ
Background Color		Default	Ορίζουμε το χρώμα του φόντου
Enable		{on} (off)	Ορίζουμε αν είναι ενεργό ή όχι το στοιχείο ελέγχου
FontName		MS Sans Serif	Ορίζουμε την γραμματοσειρά
FontSize		8.0	Ορίζουμε το μέγεθος των γραμμάτων
FontWeight		{normal} (light) (demi) (bold)	Ορίζουμε το στυλ των γραμμάτων
HorizontalAlignment		{center} (right) (left)	Οριζόντια στοίχιση
Position			Ορίζουμε τις διαστάσεις και την θέση του στοιχείου ελέγχου
	X		Οριζόντια απόσταση από την αριστερή κάτω γωνία
	Y		Κάθετη απόσταση από την

			αριστερή κάτω γωνία
	Width		Πλάτος στοιχείου ελέγχου
	Height		Ύψος στοιχείου ελέγχου
String		pushbutton	Ορίζουμε το κείμενο που θα είναι πάνω στο στοιχείο ελέγχου
Tag		pushbutton1	Ορίζουμε την ετικέτα του στοιχείου ελέγχου
Units		{characters} (points) (normalized) (pixels) (centimeters) (inches)	Ορίζουμε την μονάδα μέτρησης του στοιχείου ελέγχου
Visible		{on} (off)	Ορίζουμε αν είναι ορατό ή όχι το στοιχείο ελέγχου

Πίνακας ιδιοτήτων στοιχείων ελέγχου

Το κουμπί pushbutton1 συνοδεύεται από μια συνάρτηση στο αντίστοιχο m-file του παραθύρου την

function pushbutton1_Callback(hObject, eventdata, handles)

στην οποία ενσωματώνουμε τις εντολές που θέλουμε να εκτελεί το κουμπί αφού το πατήσουμε .

2.1.2.3 Toggle Button 

Το στοιχείο ελέγχου togglebutton μας δίνει τη δυνατότητα επιλογής ή όχι μιας λειτουργίας. Εάν το έχουμε πατημένο θα εκτελεστεί η λειτουργία που θα περιγράφεται στο αντίστοιχο m-file, αλλιώς, αν δεν είναι πατημένο, δεν επηρεάζει το πρόγραμμα.

Το στοιχείο ελέγχου togglebutton συνοδεύεται από μια συνάρτηση στο αντίστοιχο m-file του παραθύρου:

function togglebutton_Callback(hObject, eventdata, handles)

Μπορούμε βέβαια, να αλλάξουμε τις ιδιότητες του στοιχείου ελέγχου με τη βοήθεια του Property Inspector, όπως περιγράφονται στον πίνακα ιδιοτήτων .

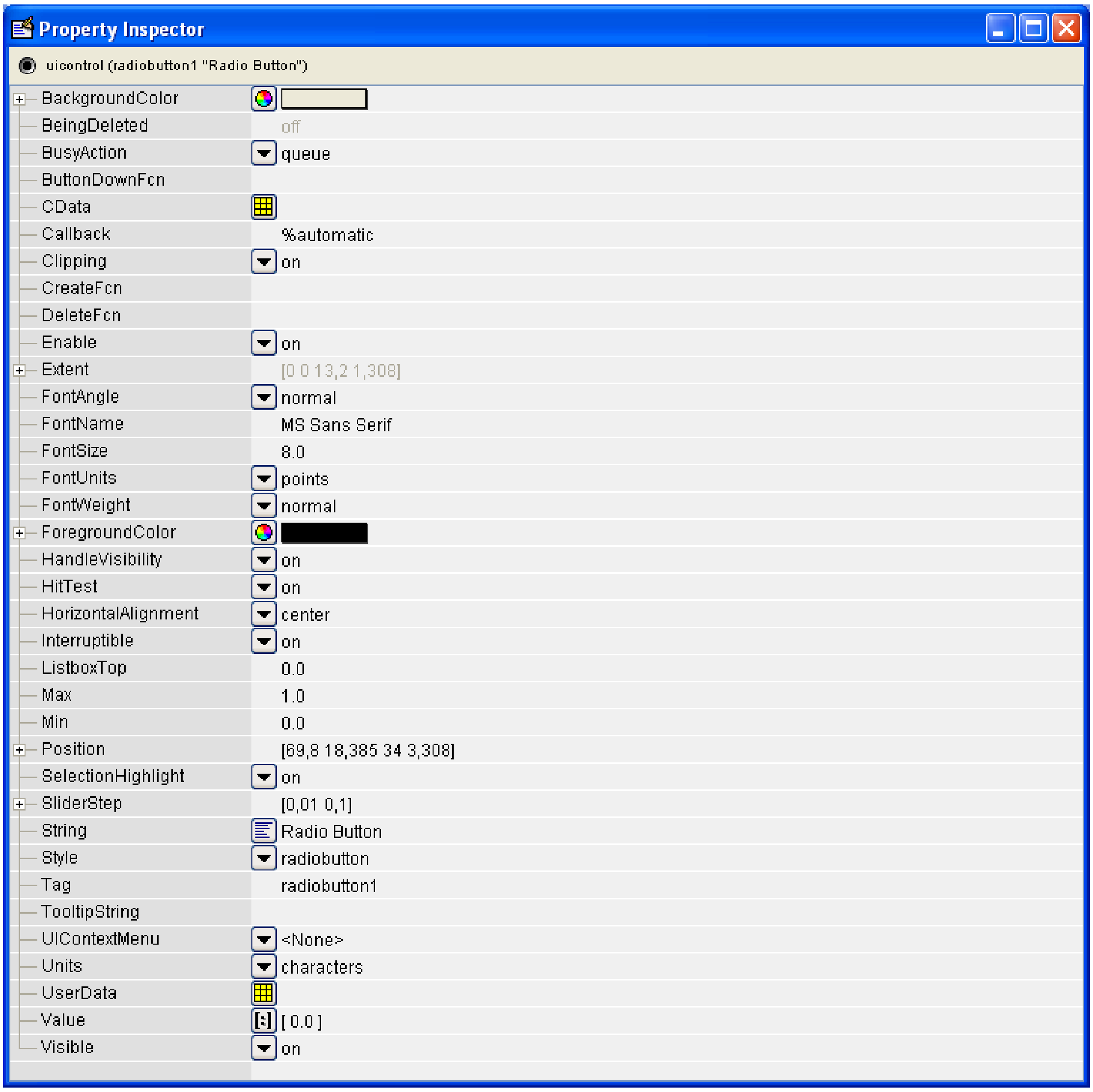
2.1.2.4 Radio Button

Το στοιχείο ελέγχου radiobutton μας δίνει τη δυνατότητα να επιλέξουμε μια μόνο επιλογή από ένα πλήθος επιλογών. Διαλέγοντας μια αναιρείται η προηγούμενη.

Το στοιχείο ελέγχου radiobutton συνοδεύεται από μια συνάρτηση στο αντίστοιχο m-file του παραθύρου:

function radiobutton_Callback(hObject, eventdata, handles)

Μπορούμε βέβαια, να αλλάξουμε τις ιδιότητες του στοιχείου ελέγχου με τη βοήθεια του Property Inspector, όπως περιγράφονται στον πίνακα ιδιοτήτων.



2.1.2.5 Checkbox

Το στοιχείο ελέγχου checkbox μας δίνει τη δυνατότητα να επιλέξουμε μια ή περισσότερες επιλογές, δηλαδή να τσεκάρουμε τις επιλογές που θέλουμε.

Το στοιχείο ελέγχου checkbox συνοδεύεται από μια συνάρτηση στο αντίστοιχο m-file του παραθύρου:

function checkbox_Callback(hObject, eventdata, handles)

Μπορούμε βέβαια, να αλλάξουμε τις ιδιότητες του στοιχείου ελέγχου με τη βοήθεια του Property Inspector, όπως περιγράφονται στον πίνακα ιδιοτήτων.

2.1.2.6 Edit Text 

Το στοιχείο ελέγχου edittext μας δίνει τη δυνατότητα να πάρουμε την τιμή που έχει εισάγει μέσα σ'αυτό ο χρήστης .

Το στοιχείο ελέγχου edittext συνοδεύεται από δυο συναρτήσεις στο αντίστοιχο m-file του παραθύρου:

function edit_Callback(hObject, eventdata, handles) και

function edit_CreateFcn(hObject, eventdata, handles) .

Η δεύτερη είναι η συνάρτηση που ορίζει παραμέτρους κατά την δημιουργία του edittext .

Μπορούμε βέβαια, να αλλάξουμε τις ιδιότητες του στοιχείου ελέγχου με τη βοήθεια του Property Inspector, όπως περιγράφονται στον πίνακα ιδιοτήτων.

2.1.2.7 Static Text 

Το στοιχείο ελέγχου statictext μας δίνει τη δυνατότητα να εισάγουμε στατικά κείμενα σε οποιοδήποτε σημείο του παραθύρου .

Το στοιχείο ελέγχου statictext δε συνοδεύεται από καμιά συνάρτηση στο αντίστοιχο m-file του παραθύρου.

Μπορούμε βέβαια, να αλλάξουμε τις ιδιότητες του στοιχείου ελέγχου με τη βοήθεια του Property Inspector, όπως περιγράφονται στον πίνακα ιδιοτήτων .

2.1.2.8 Slider 

Το στοιχείο ελέγχου slider μας δίνει τη δυνατότητα να μεταβάλουμε κάποια τιμή μιας μεταβλητής με τη βοήθεια της μπάρας (slider).

Το στοιχείο ελέγχου slider συνοδεύεται από δυο συναρτήσεις στο αντίστοιχο m-file του παραθύρου:

function slider_Callback(hObject, eventdata, handles) και *function slider_CreateFcn(hObject, eventdata, handles) .*

Η δεύτερη είναι η συνάρτηση που ορίζει παραμέτρους κατά την δημιουργία του slider .

Μπορούμε βέβαια, να αλλάξουμε τις ιδιότητες του στοιχείου ελέγχου με τη βοήθεια του Property Inspector, όπως περιγράφονται στον πίνακα ιδιοτήτων .

2.1.2.9 Frame 

Το στοιχείο ελέγχου frame χρησιμοποιείται για την ομαδοποίηση πολλών στοιχείων ελέγχου.

Το στοιχείο ελέγχου frame δε συνοδεύεται από καμία συνάρτηση στο αντίστοιχο m-file του παραθύρου.

Μπορούμε βέβαια, να αλλάξουμε τις ιδιότητες του στοιχείου ελέγχου με τη βοήθεια του Property Inspector, όπως περιγράφονται στον πίνακα ιδιοτήτων .

Μας δίνει την δυνατότητα επεξεργαστούμε πολλά στοιχεία που βρίσκονται πάνω σε αυτό.

Εάν για παράδειγμα έχω τοποθετήσει διάφορα αντικείμενα πάνω σε αυτό και ζητήσω να κρύψω το Frame τότε αυτόματα θα κρυφτούν όλα τα αντικείμενα που βρίσκονται πάνω σε αυτό.

2.1.2.10 List box 

Το στοιχείο ελέγχου list box μας δίνει τη δυνατότητα να διαλέξουμε μέσα από μια λίστα μια λειτουργία ή να παρουσιάσουμε ένα αποτέλεσμα μιας λειτουργίας.

Το στοιχείο ελέγχου list box συνοδεύεται από δυο συναρτήσεις στο αντίστοιχο m-file του παραθύρου:

function listbox_Callback(hObject, eventdata, handles) και

function listbox_CreateFcn(hObject, eventdata, handles) .

Η δεύτερη είναι η συνάρτηση που ορίζει παραμέτρους κατά τη δημιουργία του list box.

Μπορούμε βέβαια, να αλλάξουμε τις ιδιότητες του στοιχείου ελέγχου με τη βοήθεια του Property Inspector, όπως περιγράφονται στον πίνακα ιδιοτήτων .

2.1.2.11 Popup Menu 

Το στοιχείο ελέγχου popup menu μας δίνει τη δυνατότητα να διαλέξουμε μέσα από ένα μενού πολλαπλών επιλογών την λειτουργία που επιθυμούμε .

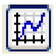
Το στοιχείο ελέγχου popup menu συνοδεύεται από δυο συναρτήσεις στο αντίστοιχο m-file του παραθύρου:

function popupmenu_Callback(hObject, eventdata, handles) και

function popupmenu_CreateFcn(hObject, eventdata, handles) .

Η δεύτερη είναι η συνάρτηση που ορίζει παραμέτρους κατά τη δημιουργία του popup menu .

Μπορούμε βέβαια, να αλλάξουμε τις ιδιότητες του στοιχείου ελέγχου με τη βοήθεια του Property Inspector, όπως περιγράφονται στον πίνακα ιδιοτήτων.

2.1.2.12 Axes 

Το στοιχείο ελέγχου axes μας δίνει τη δυνατότητα να εισάγουμε μια γραφική παράσταση στο παράθυρο .

Το στοιχείο ελέγχου axes δε συνοδεύεται από καμία συνάρτηση στο αντίστοιχο m-file του παραθύρου .

Μπορούμε βέβαια, να αλλάξουμε τις ιδιότητες του στοιχείου ελέγχου με τη βοήθεια του Property Inspector, όπως περιγράφονται στον πίνακα ιδιοτήτων, όμως το στοιχειό ελέγχου axes έχει περισσότερες ιδιότητες από τα άλλα, οι οποίες είναι για την γραφική παράσταση.

2.1.3 Προγραμματισμός στο m-file

Όπως είδαμε τα περισσότερα στοιχεία ελέγχου συνοδεύονται από μια συνάρτηση callback, στην οποία προσθέτουμε τις εντολές που θέλουμε να εκτελεί η ενεργοποίηση του κάθε στοιχείου ελέγχου.

Κάθε στοιχείο ελέγχου μπορεί να δημιουργηθεί και να οριστούν οι ιδιότητες του από το m-file του παραθύρου. Και σε ένα έτοιμο στοιχείο ελέγχου, που δημιουργήθηκε με την πιο πάνω διαδικασία, μπορούμε με προγραμματισμό στο m-file να αλλάξουμε τις ιδιότητες του.

Κάθε στοιχείο ελέγχου είναι ένα αντικείμενο (handles) στο m-file και καλείται ως handles.tag_of_button. Οι εντολές set και get δίνουν τη δυνατότητα να αλλάξουμε κάποια ιδιότητα σε ένα στοιχείο ελέγχου ή να πάρουμε κάποια τιμή από στοιχεία ελέγχου που επιστρέφουν τιμές (Toggle Button, Radio Button, Checkbox, Edit Text, Slider, List box ,Popup Menu). Για παράδειγμα η εντολή

set(handles.text1,’string’,’hello’);

αλλάζει το κείμενο που υπήρχε στο text1 και τοποθετεί το κείμενο hello, ενώ η εντολή

temp=get(handles.edit1,’string’);

παίρνει ότι υπάρχει μέσα στο αντικείμενο edit1 και το καταχωρεί στην μεταβλητή temp.

Οι ιδιότητες που χρησιμοποιούνται περισσότερο στον προγραμματισμό του m-file είναι οι εξής : string,value .

Η ιδιότητα **string** δίνει τη δυνατότητα να εισάγουμε σε όλα τα αντικείμενα εκτός του axes, με τη βοήθεια της set, κάποιο κείμενο πάνω στο αντικείμενο, καθώς και να πάρουμε κάποιο κείμενο από ένα αντικείμενο με τη βοήθεια της get.

Η ιδιότητα **value** δίνει τη δυνατότητα να ορίσουμε μια τιμή ή να πάρουμε μια τιμή από ένα αντικείμενο. Στον παρακάτω πίνακα έχουμε τα αντικείμενα, τα οποία έχουν την ιδιότητα value και τις τιμές που παίρνει η ιδιότητα, αλλά και μια περιγραφή.

ΑΝΤΙΚΕΙΜΕΝΟ	ΤΙΜΕΣ	ΠΕΡΙΓΡΑΦΗ
Toggle Button	1 0	Όταν έχει πατηθεί Όταν δεν έχει πατηθεί
Radio Button	1 0	Όταν έχει επιλεχθεί Όταν δεν έχει επιλεχθεί
Checkbox	1 0	Όταν έχει τσεκαριστεί Όταν δεν έχει τσεκαριστεί
Slider	1 0	Μας επιστρέφει την θέση που βρίσκεται το slider (αριθμός μεταξύ 0 και 1)
List box	1 2 3	Όταν έχει επιλεχθεί η πρώτη γραμμή Όταν έχει επιλεχθεί η δεύτερη γραμμή Όταν έχει επιλεχθεί η τρίτη γραμμή

Popup Menu	1 2 3	Όταν έχει επιλεχθεί η πρώτη επιλογή Όταν έχει επιλεχθεί η δεύτερη επιλογή Όταν έχει επιλεχθεί η τρίτη επιλογή

Πίνακας τιμών ιδιότητας value

Ο τρόπος με τον οποίο μπορούμε να ορίσουμε ή να πάρουμε κάποια τιμή από την ιδιότητα value είναι ο ίδιος με εκείνο της ιδιότητας string.

2.2 Πακέτο Polyx

Το πακέτο POLYX είναι ένα νέο toolbox του MATLAB, το Polynomial Toolbox, το οποίο έχει εφαρμογή στα συστήματα , στα σήματα , στον έλεγχο στην ανάλυση και στον σχεδιασμό συστημάτων με βάση προχωρημένες πολυωνυμικές μεθόδους. Το POLYX περιέχει πάνω από 200 m – files σε κώδικα MATLAB. Πιο συγκεκριμένα το POLYX έχει τα παρακάτω χαρακτηριστικά και δυνατότητες:

- Απλές πράξεις μεταξύ πολυωνυμικών πινάκων και ευανάγνωστη παρουσίαση αυτών με την βοήθεια ενός νέου αντικειμένου (pol-object).
- Σύνθετες πράξεις και συναρτήσεις για πολυωνυμικούς πίνακες καθώς και αρκετές ρουτίνες για την επίλυση γραμμικών και τετραγωνικών εξισώσεων με πολυωνυμικούς πίνακες.
- Πολυωνυμικούς πίνακες με μιγαδικούς συντελεστές για εφαρμογή στην επεξεργασία σήματος.
- Σήματα και συστήματα σε συνεχή ή διακριτό χρόνο βασισμένα σε κλασματικούς (ρητούς) πολυωνυμικούς πίνακες
- Κλασσική και εύρωστη ανάλυση των γραμμικά χρονικά ανεξάρτητων LTI συστημάτων και φίλτρων.
- Κλασσικός και βέλτιστος σχεδιασμός στην επανατοποθέτηση πόλων και στους σταθεροποιητικούς ελεγκτές.
- Εύκολη μετατροπή ενός αντικειμένου σε μορφή LTI object του MATLAB και από μορφή pol object σε sym object του Symbolic Math Toolbox του MATLAB.

2.2.1 Συναρτήσεις

Στα πλαίσια της εφαρμογής μας, επικεντρωθήκαμε σε συναρτήσεις οι οποίες έχουν εφαρμογή στις πράξεις πολυωνύμων ή πολυωνυμικών πινάκων, για παράδειγμα η πρόσθεση μεταξύ δυο πινάκων.

Επίσης, επικεντρωθήκαμε σε συναρτήσεις οι οποίες έχουν εφαρμογή στις ιδιότητες των πολυωνύμων ή πολυωνυμικών πινάκων, για παράδειγμα ο υπολογισμός της ορίζουσας καθώς και σε συναρτήσεις που έχουν εφαρμογή στις ειδικές μορφές πινάκων, για παράδειγμα ο υπολογισμός της Companion μορφής ενός πολυωνυμικού πίνακα.

Τέλος, επικεντρωθήκαμε σε συναρτήσεις οι οποίες έχουν εφαρμογή στην επίλυση εξισώσεων με πολυώνυμα ή πολυωνυμικούς πίνακες, για παράδειγμα η επίλυση διοφαντικών εξισώσεων αλλά και η επίλυση εξισώσεων Lyapunov.

Στο παράρτημα δίνεται αναλυτικότερα ο τρόπος σύνταξης καθώς και η περιγραφή όλων των συναρτήσεων του POLYX που χρησιμοποιούνται από την εφαρμογή.

2.2.2 Κλάσεις

Οι κλάσεις των αντικειμένων (objects) του POLYX που χρησιμοποιούνται στην εφαρμογή είναι οι εξής:

Η πολυωνυμική μορφή **pol-object** που έχει ως στοιχεία του πίνακα απλά πολυώνυμα, όπως φαίνεται και παρακάτω.

```
   s^2          -1
  -1 + s     1 + s^2
```

pol-object

Η ρητή πολυωνυμική μορφή **sdf-object** που έχει ως στοιχεία του πίνακα πολυώνυμα και ένα πολυώνυμο ως παρονομαστή, όπως φαίνεται και παρακάτω.

```
1.5 + 0.5s    -1.5s
1.5 - 1.5s     1.5s
-------------------
        s-1
```

sdf-object

Η ρητή πολυωνυμική μορφή **rdf-object** με αριθμητή πολυωνυμικό πίνακα και δεξιό παρονομαστή πολυωνυμικό πίνακα, όπως φαίνεται και παρακάτω.

```
s      s^2          /      s         s
5      3 + s^3     /      -2 + s     2 + s
```

rdf-object

Η ρητή πολυωνυμική μορφή **ldf-object** με αριθμητή πολυωνυμικό πίνακα και αριστερό παρονομαστή πολυωνυμικό πίνακα, όπως φαίνεται και παρακάτω.

```
s      s^2         \       s         s
1      1 + s^3      \     -1 + s     1 + s
```

ldf-object

2.2.3 Μεταβλητές

Οι μεταβλητές του POLYX που χρησιμοποιούνται στην εφαρμογή είναι οι εξής:

- Η μεταβλητή **s** που χρησιμοποιείται από το POLYX για σήματα και συστήματα συνεχή χρόνου.
- Η μεταβλητή **z** που χρησιμοποιείται από το POLYX για σήματα και συστήματα διακριτού χρόνου.

2.2.4 Βασικές Συναρτήσεις του Πακέτου POLYX

Οι Βασικές συναρτήσεις του πακέτου POLYX είναι :

Συναρτήση - Σύνταξη	Λειτουργία
adj $[adjA, detA] = adj(A)$	Η εντολή adj μας τον προσαρτημένο πολυωνυμικό πίνακα και την ορίζουσα ενός τετραγωνικού πολυωνυμικού πίνακα A . Ο προσαρτημένος πίνακας ορίζεται ως $adj(A(s)) = (-1)^{i+j} det(A^{ij}(s))$ όπου $A^{ij}(s)$ είναι ένας πίνακας από τον οποίο λείπει η i στήλη και η j γραμμή και χρησιμοποιείται στην εύρεση αντίστροφου πίνακα.
axb $X = xab(A,B)$	Η εντολή xab επιλύει το γραμμικό πολυωνυμικό σύστημα εξισώσεων $X \cdot A = B$, όπου A και B πολυωνυμικοί πίνακες .
axbc $X = axbc(A,B,C)$	Η εντολή axbc επιλύει το γραμμικό πολυωνυμικό σύστημα εξισώσεων $A \cdot X \cdot B = C$, όπου A , B και C πολυωνυμικοί πίνακες .
axby0 $[X,Y] = axby0(A,B,C)$ $[X,Y] = axby0(A,B,C,'minx')$ $[X,Y] = axby0(A,B,C,'miny')$	Η εντολή axby0 επιλύει την διοφαντική πολυωνυμική εξίσωση $A \cdot X + B \cdot Y = 0$, όπου A , B και C πολυωνυμικοί πίνακες . Οι παράμετροι minx , miny καθορίζουν ποιος άγνωστος θα έχει τον μικρότερο βαθμό .
axbyc $[X,Y] = axbyc(A,B,C)$ $[X,Y] = axbyc(A,B,C,'minx')$ $[X,Y] = axbyc(A,B,C,'miny')$	Η εντολή axbyc επιλύει την διοφαντική πολυωνυμική εξίσωση $A \cdot X + B \cdot Y = C$, όπου A , B και C πολυωνυμικοί πίνακες . Οι παράμετροι minx , miny καθορίζουν ποιος άγνωστος θα έχει τον μικρότερο βαθμό .
axxab $X = axxab(A,B)$	Η εντολή axxab επιλύει την αμφίπλευρη συμμετρική πολυωνυμική εξίσωση $A' \cdot X + X' \cdot A = B$, όπου A

	πολυωνυμικός πίνακας και B ένας para-Hermitian (εάν ο Β είναι σε συνεχές χρόνο τότε πρέπει $B(s)=B^T(-s)$ ενώ σε διακριτό χρόνο πρέπει $B(z)=B^T\left(1/z\right)$) πολυωνυμικός πίνακας.
axyab $[X,Y]=axyab(A,BL,BR)$	Η εντολή axyab επιλύει την μη συμμετρική πολυωνυμική εξίσωση $A'\cdot X+Y'\cdot A=BL'+BR$, όπου A ευσταθής (stable) πολυωνυμικός πίνακας και *BL,BR* δυο πολυωνυμικοί πίνακες.
axybc $[X,Y]=axybc(A,B,C)$	Η εντολή axybc επιλύει την διοφαντική πολυωνυμική εξίσωση $A\cdot X+Y\cdot B=C$, όπου A , B και *C* πολυωνυμικοί πίνακες.
clements $[C,u,p]=clements(P)$	Η εντολή clements μετασχηματίζει έναν para-Hermitian (εάν ο P είναι σε συνεχές χρόνο τότε πρέπει $P(s)=P^T(-s)$ ενώ σε διακριτό χρόνο πρέπει $P(z)=P^T\left(1/z\right)$) πραγματικό pencil πίνακα $P(s)=sE+A$, όπου Ε ένας μη συμμετρικός και Α ένας συμμετρικός πίνακας , σε Clements μορφή. $C(s)=uP(s)u^T=\begin{bmatrix} 0 & 0 & sE_1+A_1 \\ 0 & A_2 & sE_3+A_3 \\ -sE_1^T+A_1^T & -sE_3^T+A_3^T & sE_4+A_4 \end{bmatrix}$ όπου u ένας σταθερός ορθογώνιος πίνακας.
colred $[D,rk,U,Ui]=colred(A)$	Η εντολή colred επιστρέφει την column reduced form ενός πολυωνυμικού πίνακα *A*. Θα ισχύει $D=A\cdot U$, όπου *U* ένας unimodular πίνακας και *D* πολυωνυμικός πίνακας με στήλες τοποθετημένες έτσι ώστε να οι βαθμοί (degree) των στηλών να έχουν φθίνουσα σειρά. Το *rk* μας δείχνει τις μη μηδενικές στήλες του *D*. Το *Ui* είναι ο αντίστροφος του *U*.
company	Η εντολή compan επιστρέφει τον Companion

$C = compan(P)$	matrix του τετραγωνικού πολυωνυμικού πίνακα P. Ο πολυωνυμικός πίνακας P γράφεται με την βοήθεια των συντελεστών και ο Companion matrix είναι ένας σταθερός πίνακας με blocks τους πίνακες συντελεστές του P, δηλαδή, εάν $$P(s) = P_0 + P_1 \cdot s + P_2 \cdot s^2 + ... + P_d \cdot s^d$$ τότε $$C = \begin{bmatrix} 0 & I & 0 & \dots & \dots \\ 0 & 0 & I & 0 & \dots \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & 0 & I \\ \frac{P_0}{-P_d} & \frac{P_1}{-P_d} & \dots & \dots & \frac{P_{d-1}}{-P_d} \end{bmatrix}$$
<u>conj</u> $conj(A)$	Η εντολή conj επιστρέφει τον συζυγή ενός πολυωνυμικού πίνακα A , δηλαδή , τους συζυγείς συντελεστές των πολυωνύμων (στοιχείων) του πίνακα A.
<u>deg</u> $deg(A)$ $deg(A,'ent')$ $deg(A,'col')$ $deg(A,'row')$	Η εντολή deg, στην πρώτη περίπτωση, επιστρέφει τον βαθμό ενός πολυωνυμικού πίνακα A. Στην δεύτερη περίπτωση επιστρέφει τον βαθμό του κάθε στοιχείου του πολυωνυμικού πίνακα A. Στην τρίτη και στην τέταρτη περίπτωση επιστρέφει, σε διάνυσμα, τον βαθμό της κάθε γραμμής ή στήλης αντίστοιχα, του πολυωνυμικού πίνακα A.
<u>den</u> $den(A)$	Η εντολή den επιστρέφει τον παρονομαστή ενός πολυωνυμικού πίνακα A.
<u>det</u> $det(A)$	Η εντολή det μας την ορίζουσα ενός τετραγωνικού πολυωνυμικού πίνακα A .
<u>echelon</u> $[E,U,ind] = echelon(A)$	Η εντολή echelon επιστρέφει την echelon column μορφή ενός τετραγωνικού και column-reduced πολυωνυμικού πίνακα A. Ο U είναι ένας

	unimodular πίνακας που ικανοποιεί την σχέση $A \cdot U = E$.
gld $G = gld(N_1, N_2, ..., N_k)$	Η εντολή gld επιστρέφει τον μέγιστο αριστερό κοινό διαιρετή (greatest left common divisor) των πολυωνυμικών πινάκων $N_1, N_2, ..., N_k$ οι οποίοι πρέπει να έχουν ίδιο πλήθος γραμμών.
grd $G = grd(N_1, N_2, ..., N_k)$	Η εντολή grd επιστρέφει τον μέγιστο δεξιό κοινό διαιρετή (greatest right common divisor) των πολυωνυμικών πινάκων $N_1, N_2, ..., N_k$ οι οποίοι πρέπει να έχουν ίδιο πλήθος στηλών.
hermite $[H, U, ind] = hermite(A)$	Η εντολή hermite επιστρέφει την hermite column μορφή ενός πολυωνυμικού πίνακα A. Ο U είναι ένας unimodular πίνακας που ικανοποιεί την σχέση $A \cdot U = H$.
hurwitz $H = hurwitz(P)$	Η εντολή hurwitz επιστρέφει τον Hurwitz matrix ενός πολυωνυμικού πίνακα P. Ο πολυωνυμικός πίνακας P γράφεται με την βοήθεια των συντελεστών και ο Hurwitz matrix είναι ένας σταθερός πίνακας με blocks τους πίνακες συντελεστές του P, δηλαδή, εάν $$P(s) = P_0 + P_1 \cdot s + P_2 \cdot s^2 + ... + P_d \cdot s^d$$ τότε $$H = \begin{array}{c} \left[\begin{array}{ccccccc} P_{d-1} & P_{d-3} & P_{d-5} & .... & .... & .... & .... \\ P_d & P_{d-2} & P_{d-4} & .... & .... & .... & .... \\ 0 & P_{d-1} & P_{d-3} & P_{d-5} & .... & .... & .... \\ 0 & P_d & P_{d-2} & P_{d-4} & .... & .... & .... \\ 0 & 0 & P_{d-1} & P_{d-3} & P_{d-5} & .... & .... \\ .... & .... & .... & .... & .... & .... & .... \\ .... & .... & .... & .... & .... & .... & .... \end{array}\right] \end{array} \begin{array}{l} block\ row 1 \\ block\ row 2 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ block\ row d \end{array}$$
inv $inv(A)$	Η εντολή inv επιστρέφει τον αντίστροφο ρητό πολυωνυμικό πίνακα ενός πολυωνυμικού

	τετραγωνικού και ομαλού (non-singular) πίνακα A. Ο αντίστροφος ρητός πολυωνυμικός πίνακας δίνεται από την σχέση $A^{-1}(s) = \frac{1}{det(A(s))} adj(A(s))$.
<u>isfullrank</u> $isfullrank(A)$	Η εντολή isfullrank ελέγχει αν ένας πολυωνυμικός πίνακας A έχει πλήρη τάξη (full rank) , δηλαδή εάν η τάξη (rank) του πολυωνυμικού πίνακα A είναι ίση με την μικρότερη διάσταση του πίνακα .
<u>isprime</u> $isprime(A)$ $isprime([N,D])$	Η εντολή isprime ελέγχει αν ένας πολυωνυμικός πίνακας είναι αριστερά πρώτος (left prime) ή δεξιά πρώτος (right prime) . Επίσης, στην δεύτερη περίπτωση, ελέγχει αν δυο πολυωνυμικοί πίνακες είναι πρώτοι μεταξύ τους .
<u>isproper</u> $isproper(A)$	Η εντολή isproper ελέγχει αν ένας ρητός πολυωνυμικός πίνακας είναι (proper) .
<u>issingular</u> $issingular(A)$	Η εντολή issingular ελέγχει αν ένας τετραγωνικός πολυωνυμικός πίνακας A είναι μη-ομαλός (singular) .
<u>isstable</u> $isstable(A)$	Η εντολή isstable ελέγχει αν ένας πολυωνυμικός πίνακας είναι ευσταθής (stable) ανάλογα με την μεταβλητή που έχει οριστεί (συνεχή ή διακριτή) .
<u>isunimod</u> $isunimod(A)$	Η εντολή isunimod ελέγχει αν ένας πολυωνυμικός πίνακας A είναι (unimodular) .
<u>ldiv</u> $[Q,R] = ldiv(N,D)$	Η εντολή ldiv επιστρέφει το αριστερό πηλίκο (quotient) και το υπόλοιπο (remainder) της ευκλείδειας διαίρεσης δυο πολυωνυμικών πινάκων N και D για τους οποίους ισχύει $N = D \cdot Q + R$. Ο D πρέπει να είναι τετραγωνικός και ο N πρέπει να έχει ίδιο πλήθος γραμμών με τον D .
<u>llm</u>	Η εντολή llm επιστρέφει το ελάχιστο αριστερό κοινό

$L = llm(N_1, N_2, ..., N_k)$	πολλαπλάσιο (least left common multiple) των πολυωνυμικών πινάκων $N_1, N_2, ..., N_k$ οι οποίοι πρέπει να έχουν ίδιο πλήθος στηλών.
lrm $R = lrm(N_1, N_2, ..., N_k)$	Η εντολή lrm επιστρέφει το ελάχιστο δεξιό κοινό πολλαπλάσιο (least right common multiple) των πολυωνυμικών πινάκων $N_1, N_2, ..., N_k$ οι οποίοι πρέπει να έχουν ίδιο πλήθος γραμμών.
lu $[L, U] = lu(A)$	Η εντολή lu υλοποιεί την παραγοντοποίηση ενός τετραγωνικού ομαλού (non-singular) πολυωνυμικού πίνακα A έτσι ώστε $A = L \cdot U$, όπου U ένας άνω τριγωνικός πολυωνυμικός πίνακας που τα στοιχεία της διαγώνιου έχουν μη μηδενικό πραγματικό μέρος και L ένας unimodular πίνακας .
minbasis $minbasis(A)$	Η εντολή minbasis επιστρέφει μια ελάχιστη πολυωνυμική βάση (minimal basis) ενός πολυωνυμικού πίνακα A .
minus(-) $A - B$	Η εντολή minus(-) επιστρέφει την διαφορά των πολυωνυμικών πινάκων A και B.
mtimes(*) $A * B$	Η εντολή mtimes(*) επιστρέφει το γινόμενο των πολυωνυμικών πινάκων A και B.
null $N = null(A)$	Η εντολή null επιστρέφει μια πολυωνυμική βάση (null space) για τον δεξιό μηδενικό χώρο ενός πολυωνυμικού πίνακα A έτσι ώστε να ισχύει $A(s) \cdot N(s) = 0$, όπου $N(s)$ η πολυωνυμική βάση .
num $num(A)$	Η εντολή num επιστρέφει τον αριθμητή ενός πολυωνυμικού πίνακα A.
pencan $[C, Q, Z, dims] = pencan(P)$	Η εντολή pencan μετασχηματίζει έναν ομαλό τετραγωνικό πραγματικό pencil πίνακα $P(s)$ σε Kronecker κανονική μορφή .

	$$C(s) = Q \cdot P(s) \cdot Z = \begin{bmatrix} a + sI & 0 \\ 0 & I + se \end{bmatrix}$$ όπου Q και Z σταθεροί ορθογώνιοι πίνακες , a σταθερός πίνακας με ιδιοτημές τις αρνητικές ρίζες του $P(s)$ και e ένας σταθερός πίνακας
pinv $[Q,d] = pinv(A)$	Η εντολή pinv επιστρέφει έναν πολυωνυμικό πίνακα Q και ένα πολυώνυμο d έτσι ώστε ο ρητός πολυωνυμικός πίνακας Q/d να είναι ο ψευδοαντίστροφος ενός μη τετραγωνικού πολυωνυμικού πίνακα Α , ο οποίος πρέπει να έχει πλήρη τάξη (full rank) . Οι πίνακες επαληθεύουν τις εξής σχέσεις : $A \cdot Q \cdot A = A \cdot d$ $Q \cdot A \cdot Q = Q \cdot d$
plus (+) $A + B$	Η εντολή plus(+) επιστρέφει το άθροισμα των πολυωνυμικών πινάκων A και B.
plyap $[X,Y] = plyap(A,B,C)$	Η εντολή plyap επιλύει την εξίσωση $A(s) \cdot X + Y \cdot B(s) = C(s)$, όπου Α και Β δυο τετραγωνικοί pencil πίνακες και C pencil πίνακας με πλήθος γραμμών ίδιο με το πλήθος γραμμών του Α και πλήθος στηλών ίδιο με το πλήθος στηλών του Β. Χ και Υ είναι σταθεροί πίνακες
rank $rank(A)$	Η εντολή rank επιστρέφει τον αριθμό τον γραμμικά ανεξάρτητων γραμμών ή στηλών ενός πολυωνυμικού πίνακα A .
rdiv $[Q,R] = rdiv(N,D)$	Η εντολή rdiv επιστρέφει το δεξιό πηλίκο (quotient) και το υπόλοιπο (remainder) της ευκλείδειας διαίρεσης δυο πολυωνυμικών πινάκων N και D για τους οποίους ισχύει $N = Q \cdot D + R$. Ο D πρέπει να είναι τετραγωνικός και ο N πρέπει

	να έχει ίδιο αριθμό στηλών με τον D .
roots $roots(A)$ $roots(A,'all')$ $roots(A,'det')$ $roots(A,'eig')$	Η εντολή roots, στην πρώτη περίπτωση, επιστρέφει τις πεπερασμένες ρίζες ενός πολυωνυμικού πίνακα *A*. Στην δεύτερη περίπτωση επιστρέφει όλες τις ρίζες, πεπερασμένες ή μη, ενός πολυωνυμικού πίνακα *A*. Στην τρίτη περίπτωση επιστρέφει τις ρίζες της ορίζουσας ενός πολυωνυμικού πίνακα *A*. Τέλος, στην τέταρτη περίπτωση επιστρέφει ρίζες οι οποίες είναι οι γενικευμένες ιδιοτημές του Companion πίνακα ενός πολυωνυμικού πίνακα *A*.
rowred $[D,rk,U,Ui]=rowred(A)$	Η εντολή rowred επιστρέφει την row reduced form ενός πολυωνυμικού πίνακα *A*. Θα ισχύει $D=U\cdot A$, όπου U ένας unimodular πίνακας και D πολυωνυμικός πίνακας με γραμμές τοποθετημένες έτσι ώστε να οι βαθμοί (degree) των γραμμών να έχουν φθίνουσα σειρά. Το *rk* μας δείχνει τις μη μηδενικές γραμμές του *D*. Το *Ui* είναι ο αντίστροφος του *U*.
smith $[S,U,V]=smith(A)$	Η εντολή smith επιστρέφει την smith μορφή ενός πολυωνυμικού πίνακα *A*. Οι *U* και *V* είναι unimodular πίνακες που ικανοποιούν την σχέση $U\cdot A\cdot V=S$. Η smith μορφή του πίνακα *A* είναι: $$S=\begin{bmatrix} s_1 & 0 & \dots & \dots & 0 \\ 0 & s_2 & 0 & \dots & \dots \\ \dots & \dots & \ddots & \dots & \dots \\ 0 & \dots & 0 & s_k & 0 \\ 0 & \dots & \dots & \dots & 0 \end{bmatrix}$$ k είναι η τάξη (rank) του S , και τα πολυώνυμα $s_1,s_2,....,s_k$ που βρίσκονται στην διαγώνιο έχουν την ιδιότητα το s_j να είναι διαιρέτης του s_{j+1} για

	$j = 1,2,....,k-1$
spcof $[A,J] = spcof(B)$	Η εντολή spcof υλοποιεί την φασματική παραγοντοποίηση (spectral co-factorization) ενός τετραγωνικού para-Hermitian (εάν ο B είναι σε συνεχές χρόνο τότε πρέπει $B(s) = B^T(-s)$ ενώ σε διακριτό χρόνο πρέπει $B(z) = B^T\left(1/z\right)$) πολυωνυμικού πίνακα B. $B(s) = A(s) \cdot J \cdot A^T(-s)$ $B(z) = A(z) \cdot J \cdot A^T\left(1/z\right)$ όπου A ένας ευσταθής (stable) τετραγωνικός πολυωνυμικός πίνακας και J ένας διαγώνιος σταθερός πίνακας .
spf $[A,J] = spf(B)$	Η εντολή spf υλοποιεί την φασματική παραγοντοποίηση (spectral factorization) ενός τετραγωνικού para-Hermitian (εάν ο B είναι σε συνεχές χρόνο τότε πρέπει $B(s) = B^T(-s)$ ενώ σε διακριτό χρόνο πρέπει $B(z) = B^T\left(1/z\right)$) πολυωνυμικού πίνακα B. $B(s) = A^T(-s) \cdot J \cdot A(s)$ $B(z) = A^T\left(1/z\right) \cdot J \cdot A(z)$ όπου A ένας ευσταθής (stable) τετραγωνικός πολυωνυμικός πίνακας και J ένας διαγώνιος σταθερός πίνακας .
sylv $S = sylv(P)$	Η εντολή sylv επιστρέφει τον Sylvester matrix του πολυωνυμικού πίνακα P. Ο πολυωνυμικός πίνακας P γράφεται με την βοήθεια των συντελεστών και ο Sylvester matrix είναι ένας σταθερός πίνακας με blocks τους πίνακες συντελεστές του P , δηλαδή, εάν $P(s) = P_0 + P_1 \cdot s + P_2 \cdot s^2 + ... + P_d \cdot s^d$

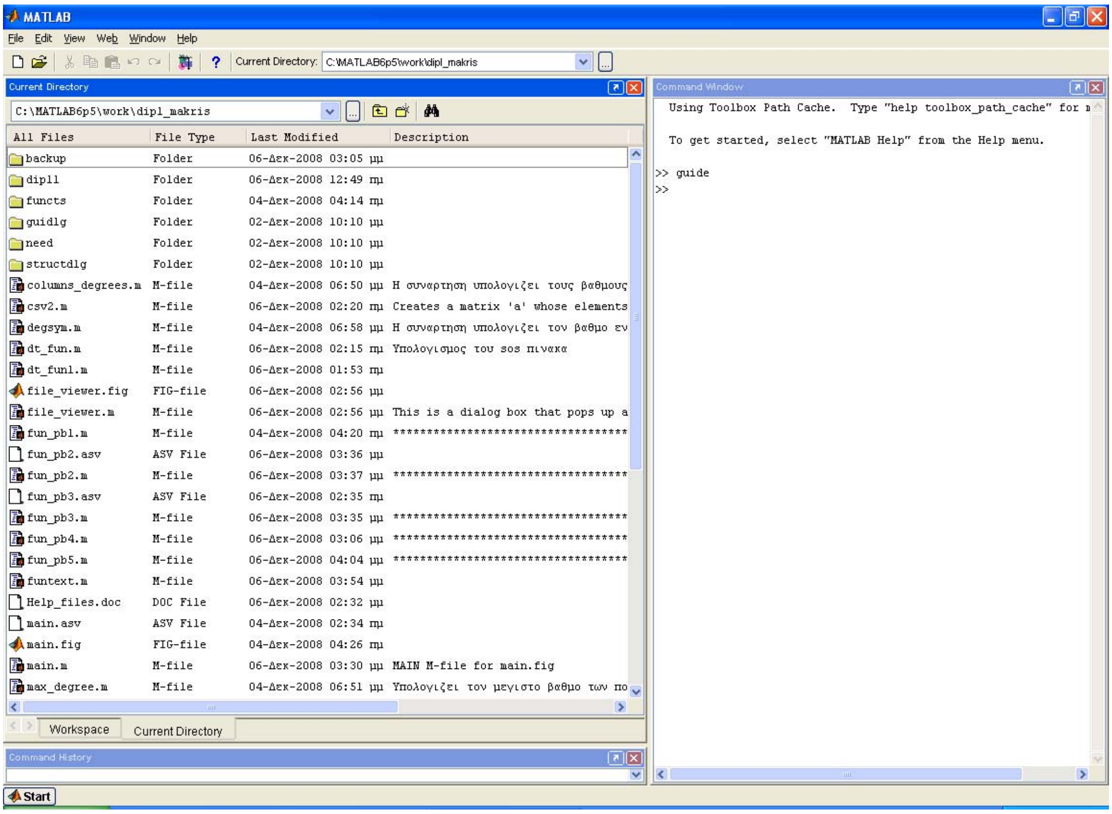
	τότε $S=\begin{bmatrix} P_0 & P_1 & \dots & P_d & 0 & \dots & \dots & 0 \\ 0 & P_0 & P_1 & \dots & P_d & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & 0 & P_0 & P_1 & \dots & P_d \end{bmatrix}\begin{matrix} block\ row1 \\ block\ row2 \\ \vdots \\ block\ row d+1 \end{matrix}$
transpose $transpose(A)$	Η εντολή transpose επιστρέφει τον ανάστροφο ενός πολυωνυμικού πίνακα A.
tri $[T,U,rowind]=tri(A,'col')$ $[T,U,colind]=tri(A,'row')$	Η εντολή tri , στην πρώτη περίπτωση, επιστρέφει την column staircase form ενός πολυωνυμικού πίνακα Α. Θα ισχύει $T=A\cdot U$, όπου U ένας unimodular πίνακας. Στην δεύτερη περίπτωση, η εντολή tri, επιστρέφει την row staircase form ενός πολυωνυμικού πίνακα Α. Θα ισχύει $T=U\cdot A$, όπου U ένας unimodular πίνακας.
xaaxb $X=xaaxb(A,B)$	Η εντολή xaaxb επιλύει την αμφίπλευρη συμμετρική πολυωνυμική εξίσωση $X\cdot A'+A\cdot X'=B$, όπου A πολυωνυμικός πίνακας και B ένας para-Hermitian (εάν ο Β είναι σε συνεχές χρόνο τότε πρέπει $B(s)=B^T(-s)$ ενώ σε διακριτό χρόνο πρέπει $B(z)=B^T\left(1/z\right)$) πολυωνυμικός πίνακας.
xaybc $[X,Y]=xaybc(A,B,C)$ $[X,Y]=xaybc(A,B,C,'minx')$ $[X,Y]=xaybc(A,B,C,'miny')$	Η εντολή xaybc επιλύει την διοφαντική πολυωνυμική εξίσωση $X\cdot A+Y\cdot B=C$, όπου A , B και C πολυωνυμικοί πίνακες. Οι παράμετροι minx , miny καθορίζουν ποιος άγνωστος θα έχει τον μικρότερο βαθμό.
xaybc0 $[X,Y]=xayb0(A,B,C)$ $[X,Y]=xayb0(A,B,C,'minx')$ $[X,Y]=xayb0(A,B,C,'miny')$	Η εντολή xayb0 επιλύει την διοφαντική πολυωνυμική εξίσωση $X\cdot A+Y\cdot B=0$, όπου A , B και C πολυωνυμικοί πίνακες. Οι παράμετροι minx, miny καθορίζουν ποιος άγνωστος θα έχει τον

	μικρότερο βαθμό.

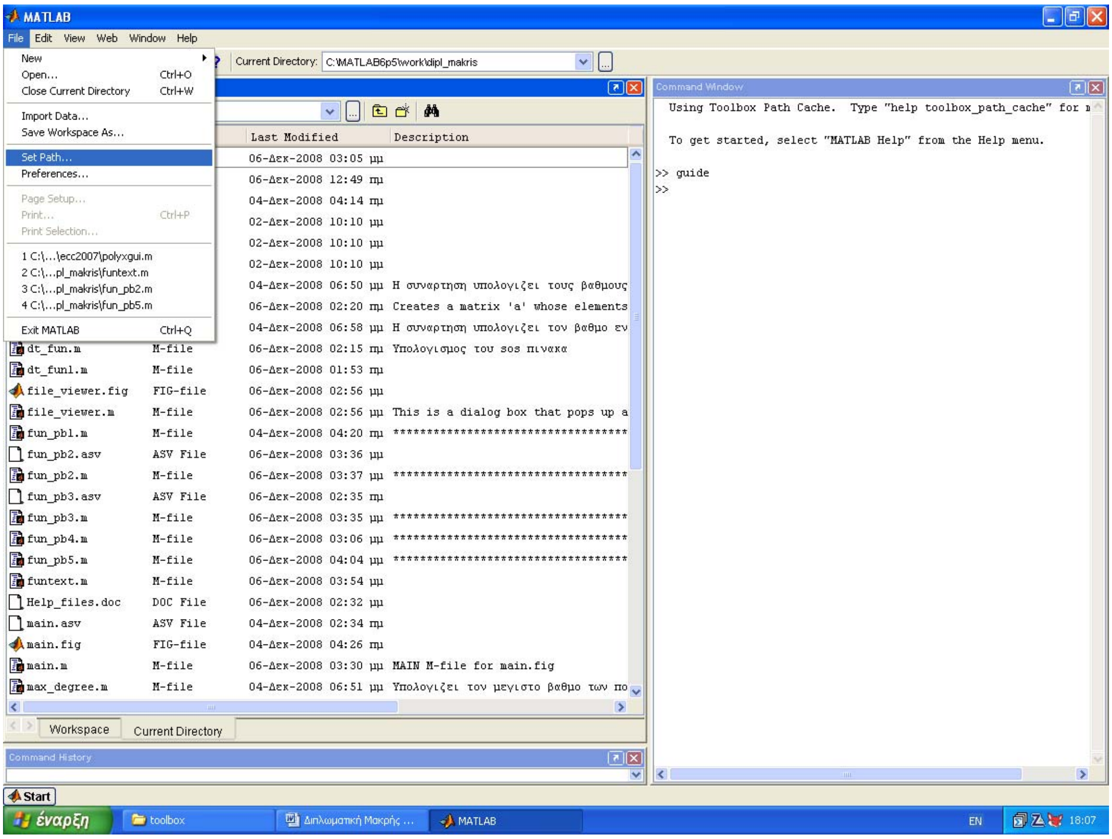
2.2.4 Εγκατάσταση του POLYX

Για να μπορέσουμε να δουλέψουμε – χρησιμοποιήσουμε το πακέτο POLYX πρέπει:

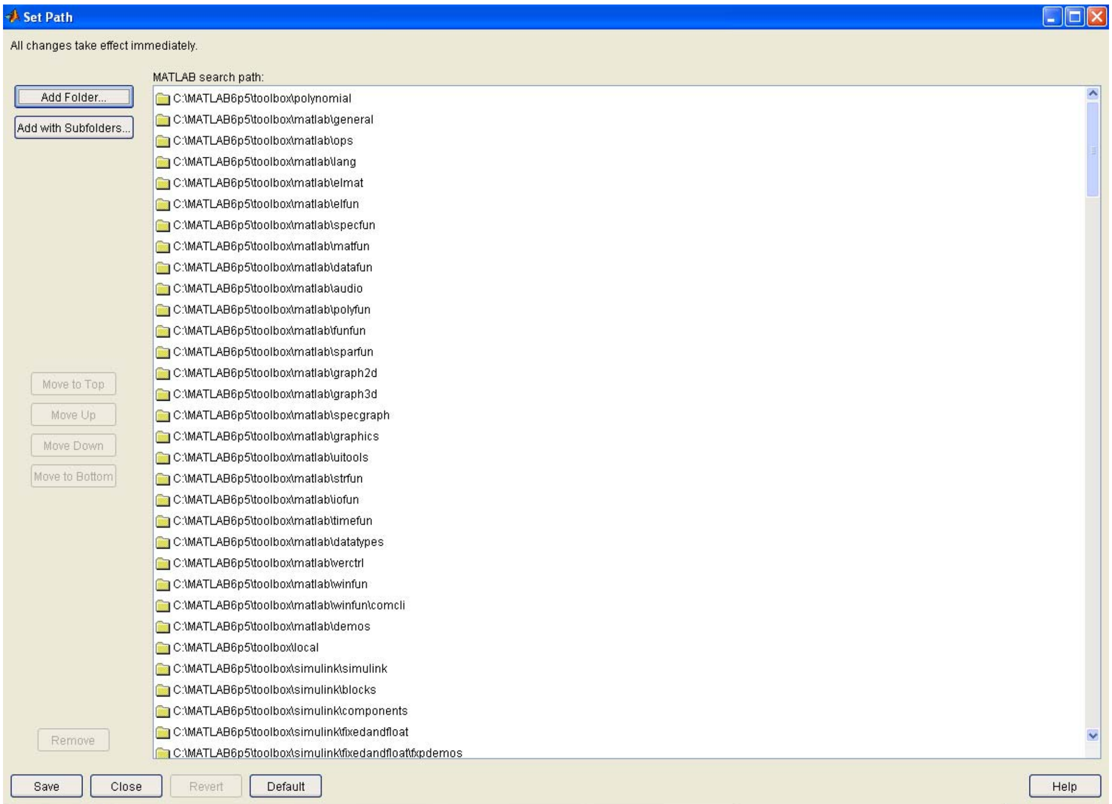
1. Να αντιγράψουμε τον φάκελο **polynomial** στον υποφάκελο **\MATLAB6p5\toolbox** (εάν χρησιμοποιείτε το MATLAB6.5) ή στον φάκελο **C:\Program Files\MATLAB\R2008a\toolbox** (εάν έχετε την τελευταία έκδοση του Matlab R2008a) ή γενικά στον υποφάκελο **\toolbox** του εγκατεστημένου Matlab στον υπολογιστή σας.
2. Ανοίγουμε το Matlab :



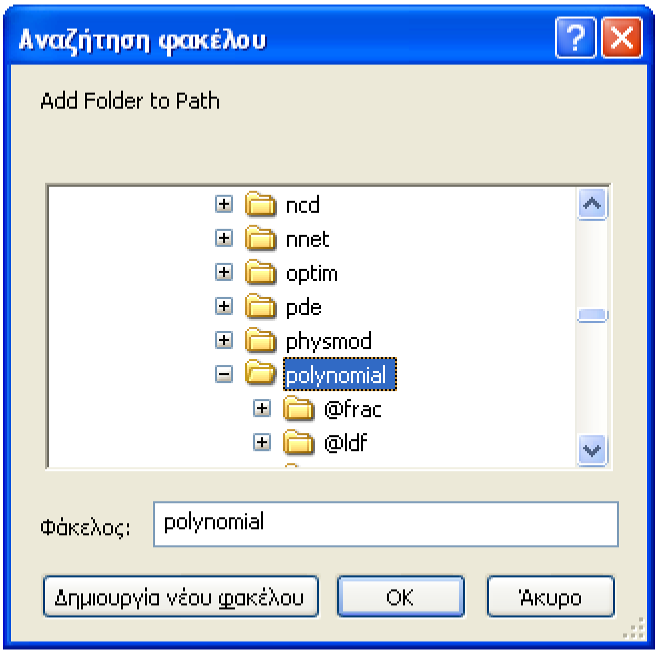
3. Να πάμε στην επιλογή **File→ Set Path**



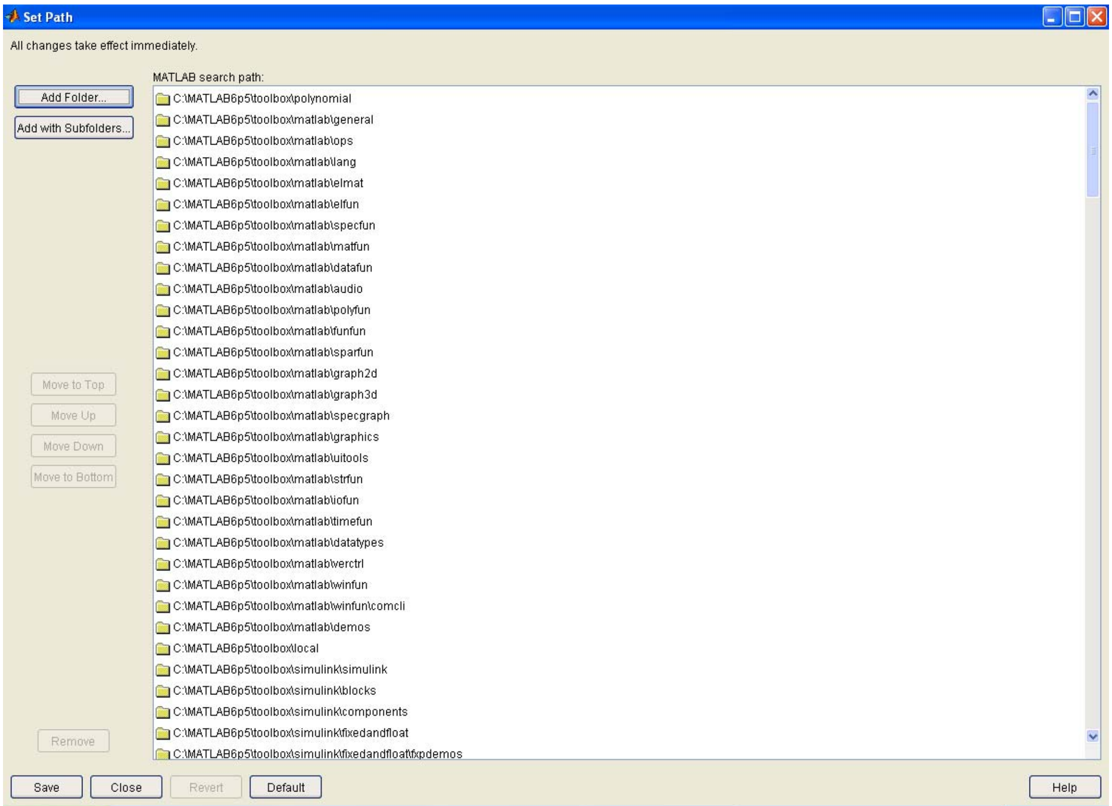
4. Στο παράθυρο που ανοίγει να πατήσουμε το κουμπί Add Folder



5. Στο επόμενο παράθυρο να επιλέξουμε τον κατάλογο **polynomial** στον υποφάκελο **\toolbox** του Matlab



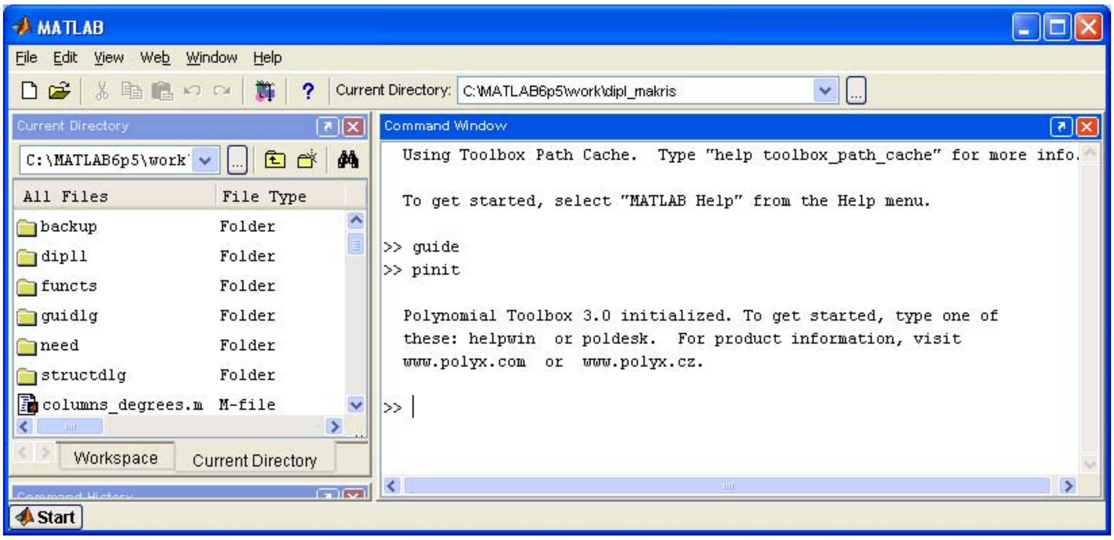
6. Να πατήσουμε **OK** και κατόπιν στο παράθυρο :



Το κουμπί **Save**

Αυτό που κάναμε είναι να δημιουργήσουμε στο MatLab μία διαδρομή αναζήτησης συναρτήσεων έτσι ώστε να είναι δυνατή η εκτέλεση των συναρτήσεων του POLYX.

Για να ενεργοποιήσετε το POLYX απλά γράψτε στην γραμμή εντολών : **“pinit”**



Κάθε φορά που θέλετε να χρησιμοποιήσετε της συναρτήσεις του POLYX αρκεί να πληκτρολογείτε στην γραμμή εντολών **“pinit”** και όλα είναι έτοιμα ….

3. Περιγραφή της εφαρμογής

3.1 Η προγραμματιστική λύση του αλγορίθμου

Για να υλοποιήσουμε τα βήματα του αλγορίθμου που περιγράφηκαν στον πρώτο κεφάλαιο υλοποιήθηκαν αρκετά m-files τα οποία σταδιακά υλοποιούν τον αλγόριθμο.

Αξίζει να σημειωθεί όταν η διαχείριση ρητών πολυωνυμικών πινάκων μέσω του MatLab αλλά και μέσω του πακέτου POLYX (παρόλο που είναι κατασκευασμένο για την διαχείριση πολυωνυμικών πινάκων , όχι όμως για την διαχείριση ρητών πινάκων) είναι αρκετή επίπονη διαδικασία.

Παρακάτω παραθέτονται όλα τα m-files που αναπτύχθηκαν για την εφαρμογή

columns_degrees.m

Η συνάρτηση δέχεται ως όρισμα έναν πίνακα F και υπολογίζει τους βαθμούς των στηλών (της κάθε στήλης ξεχωριστά) του πίνακα και τους επιστρέφει σε πίνακα k

csv2.m

Η συνάρτηση δέχεται ως όρισμα έναν αριθμό k και έναν αριθμό start1 και δημιουργεί έναν πίνακα z ο οποίος περιέχει k Symbolic μεταβλητές από start1 δηλαδή Zs, Zs+1, ..., Zs+k

degsym.m

Η συνάρτηση υπολογίζει τον βαθμό ενός πολυωνύμου σε συμβολική μορφή

Δέχεται ένα πολυώνυμο p και επιστρέφει στο d τον βαθμό του πολυωνύμου

display_array.m

Η συνάρτηση δέχεται έναν οποιοδήποτε πίνακα και έναν αριθμό και τον τοποθετεί μέσα στο ενεργό gui. Ο αριθμός δηλώνει την θέση που θα έχει ο πίνακας μέσα στο gui , μπορεί να πάρει τιμές 1,2,3. Η τιμές καθορίζουν από αριστερά προς τα δεξιά την θέση του πίνακα (το gui χωρίζεται σε 3 περιοχές όπου ο κάθε αριθμός αντιστοιχεί σε μία από αυτές).

dt_fun.m , dt_fun1.m

Βοηθητικές συναρτήσεις για τον υπολογισμό του 4^{ου}, 5^{ου} και 6^{ου} βήματος του αλγορίθμου

file_viewer.fig (file_viewer.m)

Η συνάρτηση δέχεται ως όρισμα ένα όνομα αρχείου και το προβάλλει στην οθόνη . Ανοίγει στην ουσία to file_viewer.fig το οποίο οποίο περιέχει ένα list_box και τοποθετείτε το αρχείο μέσα σε αυτό. Έτσι ο χρήστης μπορεί να δει οποιοδήποτε αρχείο επιθυμεί. Η συνάρτηση χρησιμοποιείται για προεπισκόπιση όλων των αποτελεσμάτων της εφαρμογής.

find_n.m

Υπολογίζεται ο πίνακας n από τα ni του συστήματος

find_r.m

Υπολογίζεται ο πίνακας r του συστήματος

find_smp.m

Υπολογίζεται ο πίνακας smp του συστήματος

find_wis.m

Υπολογίζεται ο πίνακας wi του συστήματος

fun_pb1.m

Η συνάρτηση εκτελείται με το πάτημα του κουμπιού **«Create Matrix P(s)»**. Μας ζητά τις γραμμές και τις στήλες του πίνακα που θέλουμε να φτιάξουμε και δημιουργεί τόσα στοιχεία ελέγχου (Edit Texts) όσα και οι διαστάσεις του πίνακα που έχουμε καθορίσει x 2 γιατί ο πίνακας είναι ρητός πολυωνυμικός οπότε τα στοιχεία αφορούν και τον αριθμητή αλλά και τον παρονομαστή του πίνακα. Στην συνάρτηση γίνεται έλεγχος για ορθή καταχώρηση των στοιχείων για τις διαστάσεις του πίνακα και εμφανίζονται ανάλογα πλαίσια κειμένου που προειδοποιούν ανάλογα. Εμφανίζεται το πλήκτρο **«Save Matrix P(s)»**

fun_pb2.m

Η συνάρτηση εκτελείται με το πάτημα του κουμπιού **«Save Matrix P(s)»**. Η συνάρτηση ελέγχει την ορθότητα των στοιχείων καταχώρησης (Εμφανίζει κατάλληλα μηνύματα για την μη ορθή καταχώρηση δεδομένων). Μετατρέπει τα στοιχεία σε αντίστοιχα πολυώνυμα τα οποία αποθηκεύονται στους αντίστοιχους πίνακες NrArx , DrArx. Κατόπιν βρίσκεται μία δεξιά κανονική μορφή του πίνακα P(s) Nr, Dr. Τα πολυώνυμα παραγοντοποιούνται. Υπολογίζονται όλοι οι ενδιάμεσοι πίνακες που απαιτούν τα βήματα του αλγορίθμου , καθώς και ο βαθμός παρατηρησιμότητας ή ελεξιμότητας του συστήματος. Υπολογίζεται ο πίνακας k σύμφωνα με τον αλγόριθμο.

Εφόσον όλα τα στοιχεία είναι εντάξει και τα δεδομένα είναι σωστά και μπορούν να γίνουν όλοι οι υπολογισμοί :

Ενεργοποιούνται τα στοιχεία ελέγχου για την εισαγωγή των ξi καθώς και για την εισαγωγή του πίνακα Dc(s).

Ενεργοποιείται το κουμπί **«Next step»** καθώς , το κουμπί **«Previous Window»** , το κουμπί **«Previous Results»**

Όλα τα αποτελέσματα αποθηκεύονται στο αρχείο step1.txt

Στην ουσία εδώ υλοποιείται το πρώτο βήμα του αλγορίθμου :

Βήμα 1^{ο}

Βρίσκουμε διαδοχικά τις γενικευμένες Resultant του Wolovich $M_k^{F(s)}$ του πίνακα F(s) τάξης k, μέχρις ότους η $M_k^{F(s)}$ να αποκτήσει πλήρη τάξη γραμμών (full row rank) , δηλαδή μέχρι να βρεθεί το k στο οποίο : $rank_R M_k^{F(s)} = mk + \sum_{i=1}^{m} k_i$. Το ελάχιστο αυτό k για το οποίο ισχύει η παραπάνω σχέση είναι ο δείκτης παρατηρησιμότητας του συστήματος (observability index) και το συμβολίζουμε με μ.

fun_pb3.m

Η συνάρτηση εκτελείται με το πάτημα του κουμπιού **«Next step»** . Η συνάρτηση ελέγχει την ορθότητα των στοιχείων καταχώρησης (Εμφανίζει κατάλληλα μηνύματα για την μη ορθή καταχώρηση δεδομένων). Μετατρέπει τα στοιχεία σε αντίστοιχα πολυώνυμα τα οποία αποθηκεύονται στους αντίστοιχους πίνακες Dc(s) , και ksi(i). Κάνει όλους τους υπολογισμούς και αποθηκεύει όλα τα δεδομένα στο αρχείο step2.txt καθώς και τα τελικά αποτελέσματα στο αρχείο report.txt

Στην ουσία εδώ υλοποιούνται τα βήματα 4,5,6 αλγορίθμου :

Βήμα 4^o

Λύνουμε τις εξισώσεις : $\boxed{\overline{\omega}_\iota^\tau \mathrm{M}_{\xi_{\iota+1}}^{F(S)} = \overline{d}_\iota^\tau}$, $i = 1,2,...,m.$ (1.2.7) ως προς $\overline{\omega}_\iota^\tau$

Βήμα 5^o

Λύνουμε τις εξισώσεις : $\omega_\iota^\tau(s) = \overline{\omega}_\iota^\tau S_{m+p,\xi_i+1}(s)$, $i = 1,2,...,m.$ (1.2.6) ως ω_ι^τ ,

για να βρούμε τις γραμμές του πίνακα $\Omega(s) := \left[X_L(s) \;\; Y_L(s) \right] \in \mathbb{R}[s]^{mx(pxm)}$

Βήμα 6^o

Ο ζητούμενος αντισταθμιστής του συστήματος θα είναι :

$$C(s) = X_L(s)^{-1} \cdot Y_L(s) \in \mathbb{R}_{pr}[s]^{mxp}$$

fun_pb4.m

Η συνάρτηση εκτελείται με το πάτημα του κουμπιού **«Previous Results»**. Το κουμπί ενεργοποιείται μόνο όταν είμαστε στο παράθυρο καταχώρησης των ksi(i) και του Dc(s) καθώς και στο τελικό παράθυρο των αποτελεσμάτων . Εμφανίζει τα αποτελέσματα του βήματος 1 του αλγορίθμου .

fun_pb5.m

Η συνάρτηση εκτελείται με το πάτημα του κουμπιού **«Show Results…»**. Το κουμπί ενεργοποιείται μόνο όταν είμαστε στο τελικό παράθυρο των αποτελεσμάτων . Εμφανίζει τα αποτελέσματα των βημάτων 4,5,6 του αλγορίθμου .

fun_pb6.m

Η συνάρτηση εκτελείται για να ελέγξει εάν ο χρήστης θέλει να τερματίσει την εφαρμογή Βγάζει κατάλληλο μήνυμα εξόδου του προγράμματος και εάν ο χρήστης απαντήσει καταφατικό (Yes) τότε τερματίζεται η εφαρμογή

fun_pb7.m

Η συνάρτηση ενεργοποιείται με το πάτημα του κουμπιού **«Previous Window»** στο παράθυρο με την εισαγωγή των ksi(i) και Dc(s) Ενεργοποιεί τα στοιχεία του προηγούμενου παραθύρου με τα δεδομένα που περιέχουν μέσα και δίνει την δυνατότητα στον χρήστη να κάνει τροποποιήσεις στα στοιχεία.

fun_pb8.m

Η συνάρτηση ενεργοποιείται με το πάτημα του κουμπιού **«Previous Window»** στο παράθυρο με τα τελικά αποτελέσματα Ενεργοποιεί τα στοιχεία του προηγούμενου παραθύρου με τα δεδομένα που περιέχουν μέσα και δίνει την δυνατότητα στον χρήστη να κάνει τροποποιήσεις στα στοιχεία.

fun_popm1.m

Η συνάρτηση εκτελείται για να δείξει επιλεκτικά τα αποτελέσματα κάποιων υπολογισμών των πινάκων.

main.fig , main.m

Η αρχική φόρμα του προγράμματος , εδώ αρχικοποιούνται όλα τα στοιχεία του προγράμματος καθώς και παραγοντοποιούνται όλες οι μεταβλητές που χρησιμοποιούνται στο πρόγραμμα

Είναι η κεντρική φόρμα του προγράμματος, το βασικό Gui πάνω στο οποίο δουλεύουν όλες οι συναρτήσεις που περιγράφονται σε αυτήν την παράγραφο.

max_degree.m

Η συνάρτηση δέχεται σαν όρισμα έναν πολυωνυμικό πίνακα και επιστρέφει τον μέγιστο βαθμό του πολυωνύμου που δέχεται ο πίνακας.

poly_to_symbolic.m

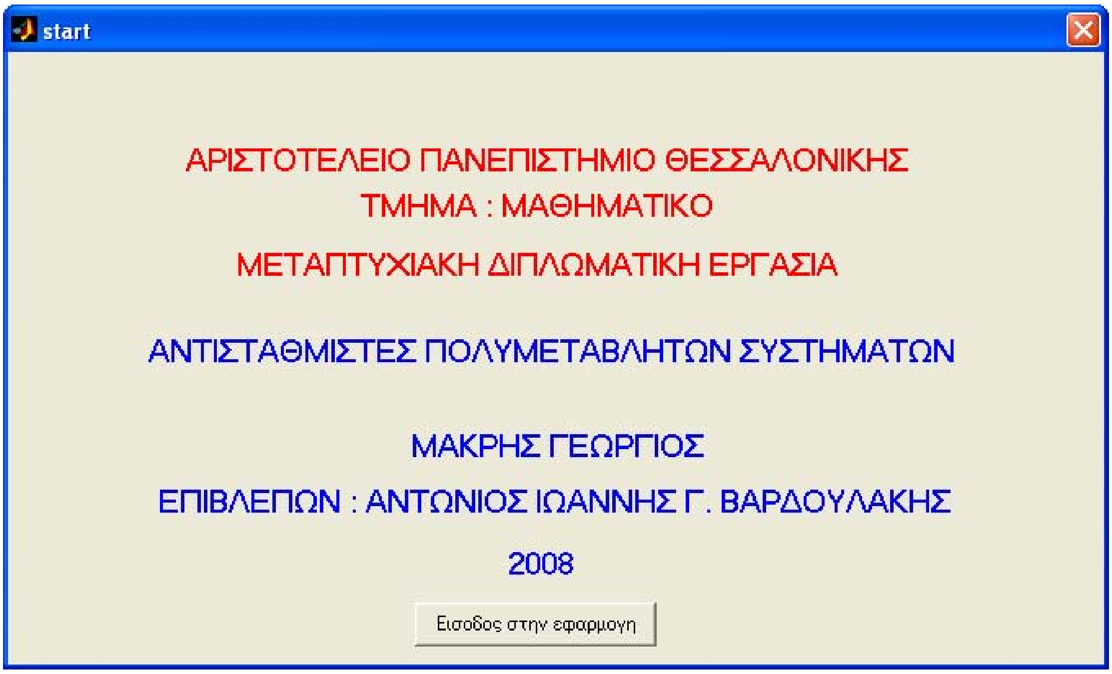
Η συνάρτηση δέχεται έναν πολυωνυμικό πίνακα και τον επιστρέφει σε συμβολική μορφή.

report.txt

Αρχείο κειμένου της εφαρμογής που περιέχει τα αποτελέσματα των υπολογισμών που έγιναν για την επίλυση ενός συστήματος

start.fig, start.m

Εισαγωγική φόρμα που εμφανίζει το τίτλο της εφαρμογής



step1.txt

Αρχείο αποτελεσμάτων που περιέχει τα πρώτα στοιχεία που υπολογίζονται από την εφαρμογή

step2.txt

Αρχείο αποτελεσμάτων που περιέχει την τελική επίλυση του αλγορίθμου με όλα τα ενδιάμεσα αποτελέσματα που υπολογίζονται από την εφαρμογή

sylforwol.m

Επιστρεφει την WOLOVICH Resultant (R) του πινακα F με K order της στήλης cj

symbolic_to_poly.m

Η συνάρτηση δέχεται έναν πίνακα σε συμβολική μορφή και τον επιστρέφει πολυωνιμικό

wolovich.m

Επιστρεφει την WOLOVICH Resultant (M) του πινακα F με K order.

3.1.1 Ο κώδικας της εφαρμογής

Start.m

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

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

% Last Modified by GUIDE v2.5 26-Jul-2007 00:57:46

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @start_OpeningFcn, ...
                   'gui_OutputFcn',  @start_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin & isstr(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before start is made visible.
function start_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
```

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

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

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes start wait for user response (see UIRESUME)
% uiwait(handles.figure1);


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

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
closereq;
main;
```

Main.m

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

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

% Last Modified by GUIDE v2.5 04-Dec-2008 04:25:57

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @main_OpeningFcn, ...
                   'gui_OutputFcn',  @main_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin & isstr(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before main is made visible.
function main_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to main (see VARARGIN)

% Choose default command line output for main
```

```
handles.output = hObject;

% Update handles structure
% Αρχικοποιηση μεταβλητων και ρυθμισεων προγραμματος
pinit;
format short g;
pformat symbs;
guidata(hObject, handles);



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

% Get default command line output from handles structure
varargout{1} = handles.output;




% Η συναρτηση εκτελειται οταν παμε να κλεισουμε το παραθυρο
function makris_CloseRequestFcn(hObject, eventdata, handles)
% hObject    handle to figure1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: delete(hObject) closes the figure

h=questdlg('exit from program?','Exit Main Form','Yes','No','No');
switch h
    case 'Yes'
        closereq;
end


% --------------------------------------------------------------------
-
function Sart_Callback(hObject, eventdata, handles)
% hObject    handle to Sart (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Ορισμος του πρωτου κουμπιου για εισαγωγη στοιχειων στον πινακα
% Εδω θα ζητηθει η διασταση του P(s) και θα εμφανιστουν τα στοιχεια
% ελεγχου για την εισαγωγη του πινακα
pb1=uicontrol('style','pushbutton',...
    'units','normalized','visible','on','string','Create Matrix
P(S)',...
    'position',[0.40 0.04 0.10 0.04]);
handles.pb1=pb1;
set(handles.pb1,'Callback','fun_pb1(gcbo,[],guidata(gcbo))');

% Ορισμος κουμπιου για την εισαγωγη και την πρωτη επεξεργασια των
στοιχειων
% του πινακα
pb2=uicontrol('style','pushbutton','backgroundcolor',get(0,'defaultUi
controlBackgroundColor'),...
```

```
    'units','normalized','visible','off','string','Save Matirix
P(s)',...
    'position',[0.40 0.04 0.10 0.04]);
handles.pb2=pb2;
set(handles.pb2,'Callback','fun_pb2(gcbo,[],guidata(gcbo))');

% Ορισμος κουμπιου για την εισαγωγη και την πρωτη επεξεργασια των
στοιχειων
% του πινακα
pb3=uicontrol('style','pushbutton','backgroundcolor',get(0,'defaultUi
controlBackgroundColor'),...
    'units','normalized','visible','off','string','Next Step -->
',...
    'position',[0.40 0.04 0.10 0.04]);
handles.pb3=pb3;
set(handles.pb3,'Callback','fun_pb3(gcbo,[],guidata(gcbo))');

pb4=uicontrol('style','pushbutton','backgroundcolor',get(0,'defaultUi
controlBackgroundColor'),...
    'units','normalized','visible','off','string','Show previus
results ...',...
    'position',[0.05 0.04 0.15 0.04]);
handles.pb4=pb4;
set(handles.pb4,'Callback','fun_pb4(gcbo,[],guidata(gcbo))');

pb5=uicontrol('style','pushbutton','backgroundcolor',get(0,'defaultUi
controlBackgroundColor'),...
    'units','normalized','visible','off','string','Show results
...',...
    'position',[0.25 0.04 0.15 0.04]);
handles.pb5=pb5;
set(handles.pb5,'Callback','fun_pb5(gcbo,[],guidata(gcbo))');

pb6=uicontrol('style','pushbutton','backgroundcolor',get(0,'defaultUi
controlBackgroundColor'),...
    'units','normalized','visible','off','string','Exit',...
    'position',[0.45 0.04 0.15 0.04]);
handles.pb6=pb6;
set(handles.pb6,'Callback','fun_pb6(gcbo,[],guidata(gcbo))');

pb7=uicontrol('style','pushbutton','backgroundcolor',get(0,'defaultUi
controlBackgroundColor'),...
    'units','normalized','visible','off','string','Previous
Window',...
    'position',[0.68 0.04 0.15 0.04]);
handles.pb7=pb7;
set(handles.pb7,'Callback','fun_pb7(gcbo,[],guidata(gcbo))');

pb8=uicontrol('style','pushbutton','backgroundcolor',get(0,'defaultUi
controlBackgroundColor'),...
    'units','normalized','visible','off','string','Previous
Window',...
    'position',[0.68 0.04 0.15 0.04]);
handles.pb8=pb8;
set(handles.pb8,'Callback','fun_pb8(gcbo,[],guidata(gcbo))');


popm1=uicontrol('style','popupmenu',...
             'backgroundcolor','w','units','normalized',...
             'string',{'Display XL(s) Array','Display YL(s)
Array',...
```

```
             'Display Ksi Array','Display Dc(s) Array'},....
             'visible','off',....
             'position',[.70 .04 .15 .031]);
handles.popm1=popm1;
set(handles.popm1,'Callback','fun_popm1(gcbo,[],guidata(gcbo))');

resxl=uicontrol('style','listbox','backgroundcolor',get(0,'defaultUic
ontrolBackgroundColor'),...
    'units','normalized','visible','off','string','save',...
    'position',[0.01 0.20 0.90 0.80]);
handles.resxl=resxl;

% UIWAIT makes main wait for user response (see UIRESUME)
% uiwait(handles.figure1);
guidata(hObject, handles);
```

fun_pb1.m

```
%*****************************************************************
******
%                 Προγραμματισμος Πρωτου κουμπιου
%*****************************************************************
******
function fun_pb1(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

answer=inputdlg({['give rows of matrix P(s) '],['give columns of
matrix P(s) ']},...
    'define of dimension',[1,30]);

if isempty(answer)
    return;
end

rows=str2num(answer{1});
columns=str2num(answer{2});
if isempty(rows)
    warndlg('must define rows of matrix P(s)');;
    return;
end
if isempty(columns)
    warndlg('must define columns of matrix P(s)');;
    return;
end
if rows<1 | rows>6
    warndlg('rows of P(s) must be 1-6');;
    return;
end

if columns<1 | columns>6
    warndlg('columns of P(s) must be 1-6');;
    return;
end

for i=1:rows
    for j=1:columns
        u_Nr(i,j)=uicontrol('style','edit','backgroundcolor','w',...
```

```
            'units','normalized','position',[0.13*j-0.05 1.05-0.13*i
0.11 0.03]);
    end
end
handles.u_Nr=u_Nr;

for i=1:rows
    for j=1:columns
        u_Dr(i,j)=uicontrol('style','edit','backgroundcolor','w',...
            'units','normalized','position',[0.13*j-0.05 1.02-0.13*i
0.11 0.03]);
    end
end
handles.u_Dr=u_Dr;
set(handles.pb1,'visible','off');
set(handles.pb2,'visible','on');
% Αποθηκευση οτι υπαρχει στο handles στο hObject για να μπορει να
% υπαρχει προσβαση απο οποιαδηποτε αλλη συναρτηση
guidata(hObject,handles)

% warndlg('Krybontai .....');
% set(handles.u_Nr,'visible','off');
% set(handles.u_Dr,'visible','off');


% set(handles.text11,'visible','on','HorizontalAlignment','left',...
%    'string',text);
```

Fun_pb2.m

```
%*****************************************************************
******
%                 Προγραμματισμος Δευτερου κουμπιου
%*****************************************************************
******
function fun_pb2(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Μετατροπη του αριθμητη Nr απο τα εικονιδια σε πολυωνυμικο πινακα
% και αпποθηκευση του ως Nr στο GUI
syms s;
pinit;
tem=get(handles.u_Nr,'string');
tem=cellstr(tem);
[r,c]=size(tem);
for i=1:r
    for j=1:c
        tem1(i,j)=strcmp(tem{i,j},'');
    end
end

if any(tem1==1)
    h=warndlg('must define values for all boxes');
    return;
end

[rows,columns]=size(handles.u_Nr);
for i=1:rows
```

```
    for j=1:columns
        Nr(i,j)=sym(get(handles.u_Nr(i,j),'string'));
    end
end

var = findsym(Nr);
switch var
    case {'s','p','z','q','d','zi','z^-1'};
        Nr=pol(Nr);
    case ''
        Nr=numeric(Nr);
    otherwise
        errordlg('Invalid variable symbol in symbolic expression.');
        return;
end;
handles.NrArx=Nr;

% Μετατροπη του παρονομαστη  Dr απο τα εικονιδια σε πολυωνυμικο
πινακα
% και αποθηκευση του ως Dr στο GUI
tem=get(handles.u_Dr,'string');
tem=cellstr(tem);
[r,c]=size(tem);
for i=1:r
    for j=1:c
        tem1(i,j)=strcmp(tem{i,j},'');
    end
end

if any(tem1==1)
    h=warndlg('must define values for all boxes');
    return;
end

[rows,columns]=size(handles.u_Dr);
for i=1:rows
    for j=1:columns
        Dr(i,j)=sym(get(handles.u_Dr(i,j),'string'));
    end
end

var = findsym(Dr);
switch var
    case {'s','p','z','q','d','zi','z^-1'};
        Dr=pol(Dr);
    case ''
        Dr=numeric(Dr);
    otherwise
        errordlg('Invalid variable symbol in symbolic expression.');
        return;
end;
handles.DrArx=Dr;

% Αρχη αλγοριθμου υπολογισμου
if exist('step1.txt')
    delete('step1.txt');
clc
diary('step1.txt');
disp('**************************************************************
***********')
```

```
disp('*************************************************************
***********')
disp('                        Programmer : George C Makris ')
disp('*************************************************************
***********')
disp('*************************************************************
***********')
disp('                          Outputs 1st step results ')
disp('*************************************************************
***********')
disp('Data Input :')
disp('----------------------------')
Nr
Dr
disp('Data computing .....')
disp('----------------------------')
disp('Size of array Nr.....')
[p,m]=size(Nr)
disp('Size of array Dr.....')
[pDr,mDr]=size(Dr)
if p~=1 | m~=1                  % MIMO system
    disp(' convert to a  right coprime MFD .....')
    disp('----------------------------')
    [n,d] = rat2rmf(Nr,Dr)    % convert to a  right coprime MFD
    disp('Data computing .....')
    disp('----------------------------')
    dmat = pol2mat(d)
    [pd,md] = size(d)
    [pdmat,mdmat] = size(dmat)
    if pdmat~=pd | mdmat~=md
        x = dmat;
        dsym = poly2sym(dmat,'s')
    else
    dsym = poly_to_symbolic(dmat)
    end


    %*********
    T = dsym;
    disp('New arrays after conversions ')
    disp('----------------------------')

    Dr = dsym
    Nr = pol2mat(n)
    Nr = poly_to_symbolic(Nr)

    if rank(T)~=min(p,m)

        fs = sprintf('\n');
        disp(fs)
        disp('The Denominator Dr(s) is not column reduced.')
        fs = sprintf('\n');
        disp(fs)
        dcr = colred(d)
        Tr = dcr*inv(d)
        ncr = n*Tr
        Dr = pol2mat(dcr);
        Dr = poly_to_symbolic(Dr);
        Nr = pol2mat(ncr);
        Nr = poly_to_symbolic(Nr);
```

```
        fs = sprintf('\n');
        disp(fs)
        disp('A column reduced form is :')
        fs = sprintf('\n');
        disp(fs)
        disp('Nr(s) = ')
        disp(Nr)
        disp('Dr(s) = ')
        disp(Dr)
    else
        disp('The Denominator Dr(s) is column reduced.')
    end
else
    Nr=poly2sym(Nr,'s');
    Dr=poly2sym(Dr,'s');
end

    disp('New arrays after factoring ')
    disp('----------------------------')

    Nr = factor(Nr)
    Dr = factor(Dr)

    P = Nr*Dr^-1;

 if p==1 & m==1
    P = factor(P);
    [Nr,Dr]=numden(P);
 end

    disp('array P(s) :  ')
    disp('----------------------------')

    P

    disp('New arrays after factoring P(s)')
    disp('----------------------------')
    disp('Nr(s) = ')
    disp(Nr)
    disp('Dr(s) = ')
    disp(Dr)

    clear n d


    disp('System (inputs - outputs) ')
    disp('----------------------------')

    fs = sprintf('\n');
    disp(fs)
    sf = sprintf('The inputs of the system are : m = %d ',m);
    disp(sf)

    fs = sprintf('\n');
    disp(fs)
    sf = sprintf('The outputs of the system are : p = %d ',p);
    disp(sf)

    disp('Calculating Observability index')
    disp('---------------------------------')
```

```
    disp('F(s)=')
    F = [Dr ; Nr]

    k = columns_degrees(F);
    for ik = 1:length(k)
        sf = sprintf('k(%d) = %d',ik,k(ik));
        disp(sf)
    end

    kk = 1;
    M = wolovich(F,kk)
    mmm = m*kk+sum(k);
    ra = rank(M);

    while ra < mmm
        kk = kk+1;

        M = wolovich(F,kk)
        mmm = m*kk+sum(k);
        ra = rank(M);
    end

    mi = kk;
    xxxx = sprintf('\n Observability index Oi = %d \n \n \n',mi);
    disp(xxxx)

disp('**********************************************************
***********')
    disp('                         End - 1st step results ')

disp('**********************************************************
***********')


    % **********************
    % Αποθηκευση ολων των παραμετρων για να μπορουν να προσπελαστουν
απο
    % αλλη συναρτηση .
    handles.Nr=Nr;
    handles.Dr=Dr;
    handles.P=P;
    handles.m=m;
    handles.p=p;
    handles.F=F;
    handles.k=k;
    handles.M=M;
    handles.mi=mi;
    handles.kk=k;
    diary('off');


    % Κλεισιμο των στοιχειων του πινακα
    set(handles.u_Dr,'visible','off');
    set(handles.u_Nr,'visible','off');


    % Δημιουργια νεων στοιχειων για εισαγωγη
    u_tksi=uicontrol('style','text','backgroundcolor',get(0,...
    'defaultUicontrolBackgroundColor'),...
    'units','normalized',...
    'string','Δωστε τα ξi ','position',[0.13 0.72 0.11 0.03]);
```

```
    for ksii=1:length(k)

u_ksi(ksii)=uicontrol('style','edit','backgroundcolor','w',...
        'units','normalized','position',[0.13 0.72-0.05*ksii 0.11
0.03]);
        end
    end
    handles.u_tksi=u_tksi;
    handles.u_ksi=u_ksi;


     u_tdc=uicontrol('style','text','backgroundcolor',get(0,...
    'defaultUicontrolBackgroundColor'),...
    'units','normalized',...
    'string','ΔΩΣΤΕ ΤΟΝ ΠΙΝΑΚΑ Dc(s) : ','position',[0.30 0.72 0.30
0.03]);

    for dci=1:m
       for j=1:m
           if dci==j
                u_dc(dci,j)=uicontrol('style','edit',...
                    'backgroundcolor','w',...
                    'units','normalized',...
                    'position',[0.30+(j-1)*0.12 0.72-0.05*dci 0.11
0.03]);
            else
                u_dc(dci,j)=uicontrol('style','edit',...
                    'backgroundcolor','w',...
                    'units','normalized',...
                    'string','0','enable','off',...
                    'position',[0.30+(j-1)*0.12 0.72-0.05*dci 0.11
0.03]);
            end
      end
    end
    handles.u_tdc=u_tdc;
    handles.u_dc=u_dc;


    % Display results in next window
    if exist('temp1.txt')
        delete('temp1.txt');
    end
    diary('temp1.txt');
    clc
    xxxx = sprintf('Observability index Oi = %d\n',mi);
    disp(xxxx)
    disp('k array :');
    for ik = 1:length(k)
        sf = sprintf('k(%d) = %d',ik,k(ik));
        disp(sf)
    end
    diary('off')


resul=uicontrol('style','text','backgroundcolor',get(0,'defaultUicont
rolBackgroundColor'),...
    'units','normalized','visible','off','string','save',...
    'position',[0.01 0.76 0.60 0.24]);
    handles.resul=resul;
```

```
    text=textread('temp1.txt','%s','delimiter','\n','whitespace','');
    set(handles.resul,'visible','on','HorizontalAlignment','left',...
    'string',text);



guidata(hObject,handles);

set(handles.pb2,'visible','off');
set(handles.pb3,'visible','on');
set(handles.pb4,'visible','on');
set(handles.pb7,'visible','on');
guidata(hObject,handles);
uiresume;
```

fun_pb3.m

```
%**********************************************************************
******
%                  Προγραμματισμος τριτου κουμπιου
%**********************************************************************
******
function fun_pb3(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
    syms s;
    Nr=handles.Nr;
    Dr=handles.Dr;
    P=handles.P;
    m=handles.m;
    p=handles.p;
    F=handles.F;
    k=handles.k;
    M=handles.M;
    mi=handles.mi;
    k=handles.kk;

tem=get(handles.u_ksi,'string');
tem=cellstr(tem);
[r,c]=size(tem);
for i=1:r
    for j=1:c
        tem1(i,j)=strcmp(tem{i,j},'');
    end
end

if any(tem1==1)
    h=warndlg('must define values for all boxes off Ksi');
    return;
end

[rows,columns]=size(handles.u_ksi);
for i=1:rows
    for j=1:columns
        ksi(i,j)=sym(get(handles.u_ksi(i,j),'string'));
    end
end
```

```
var = findsym(ksi);
switch var
    case {'s','p','z','q','d','zi','z^-1'};
        ksi=pol(ksi);
    case ''
        ksi=numeric(ksi);
    otherwise
        errordlg('Invalid variable symbol in symbolic expression. of
Ksi');
    return;
end;
for i=1:rows
    for j=1:columns
        if ksi(i,j)<mi-1
            errordlg('Give the ξi. Remember that there must be  ξi >=
μ-1   !');
            return;
        end
    end
end

handles.ksi=ksi;

% Αρχη αλγοριθμου υπολογισμου

tem=get(handles.u_dc,'string');
tem=cellstr(tem);
[r,c]=size(tem);
for i=1:r
    for j=1:c
        tem1(i,j)=strcmp(tem{i,j},'');
    end
end

if any(tem1==1)
    h=warndlg('must define values for all boxes Dc');
    return;
end

[rows,columns]=size(handles.u_dc);
for i=1:rows
    for j=1:columns
        dc(i,j)=sym(get(handles.u_dc(i,j),'string'));
    end
end

var = findsym(dc);
switch var
    case {'s','p','z','q','d','zi','z^-1'};
        dc=pol(dc);
    case ''
        dc=numeric(dc);
    otherwise
        errordlg('Invalid variable symbol in symbolic expression. of
dc');
    return;
end;

 for dci=1:rows
        ddc=dc(dci,dci);
            if isreal(poly_to_symbolic(ddc))
```

```
                errordlg('Dc is not a polynomial matrix ');
                return;
            else
                if degsym(poly_to_symbolic(ddc)) ~= k(dci)+ksi(dci)
                    % mess0=char(dci);
                    % mess1=char(degsym(poly_to_symbolic(ddc)));
                    % mess2=k(dci)+ksi(dci);
                    % mess2=char(mess2);
                    % mess1=strcat('The deggree
Dc(',mess0,')=',mess1,'  must be ',mess2);
                    errordlg('The degree is not proper (matrix Dc)');
                    % errordlg(mess1);
                return;
                end
            end
   end

if exist('step2.txt')
    delete('step2.txt');
end
diary('step2.txt');
clc
disp('*************************************************************
***********')
disp('*************************************************************
***********')
disp('                       Programmer : George C Makris ')
disp('*************************************************************
***********')
disp('*************************************************************
***********')
disp('                         Outputs 2st step results ')
disp('*************************************************************
***********')
disp('Data Input :')
disp('----------------------------')
disp('Array of ksi :')
ksi
disp(' ')
disp(' ')
disp('Array of dc(s) :')
dc
disp(' ')
disp(' ')

Dc=dc;
handles.Dc=Dc;


%**********************************
W = [];
arxi=1;

disp('Start Solving ..... :')
disp('----------------------------')

for i=1:m

    ro(i) = m*(ksi(i)+1)+sum(k);
    sf = sprintf('ro(%d) = %d',i,ro(i));
    disp(sf)
```

```
    ss(i) = (ksi(i)+1)*(m+p);
    sf = sprintf('si(%d) = %d',i,ss(i));
    disp(sf)

    ni(i) = ss(i)-ro(i);
    sf = sprintf('v(%d) = %d',i,ni(i));
    disp(sf)


%*********************************
  dt_fun;

    %---------Wolovich Resultnt M_ξi+1 of F(s)  of order ξi+1------
    kk = ksi(i)+1;

    Mkk = wolovich(F,kk);
    sf = sprintf('M_ksi+1_%d = ',i);
    disp(sf)
    disp(Mkk)


    Md = [Mkk ; dt] ;


    tMd = transpose(Md);
    rr = rref(tMd);
    EE = transpose(rr) % Column Echelon Form of Md


    %-----------------------------------
    find_r  %(call rrr)    -> computtes the matrix of  ρi

    find_n  %(call nnn)    -> computtes the matrix of  νi


    %-----------------------------------------------------------------
------
    %--------------------------- computation of Gi -----------------
------
    if i == 1

        x = EE(r(1)+1:r(1)+n(1),:);
        Gi = [x -eye(n(1))]

    else



% -------------- apo ton EE ston E2

        gr = r(1)+1;

        E2 = EE(gr:gr-1+n(1),:);

        for kk = 2:ik
            gr = r(kk-1)+n(kk-1)+r(kk)+1;
            E2 = [E2 ; EE(gr:gr-1+n(kk),:)];
        end
        E2;
```

```
% --------------- apo ton E2 ston Gi
        [pE2,mE2] = size(E2);


        Gi(:,1:r(1)) = E2(:,1:r(1));
        Gi = [Gi zeros(pE2,n(1))];


        Gi(1:n(1),r(1)+1:r(1)+n(1)) = -eye(n(1));
        gr = n(1);
        st = r(1)+n(1);

        for jj = 2:ik

            Gi = [Gi E2(:,r(jj-1)+1:r(jj-1)+r(jj))];

            Gi = [Gi zeros(pE2,n(jj))];

            Gi(gr+1:gr+n(jj),st+r(jj)+1:st+r(jj)+n(jj)) = -
eye(n(jj));

            gr = gr+n(jj);
            st = st+r(jj)+n(jj);

        end
        G{i} = Gi;
        sf = sprintf('G%d = %d',i);
        disp(sf)
        disp(G{i})
    end % end_if

    %-------------------------------------------------------------
------
    %-------------------------------------------------------------
------


    find_wis


    W = [W ; wis]      % -> computes ωi(s)
    disp('--------------------------------------------')

    if lez ~= 0
        arxi = arxi+length(z);
    end
    clear sos sost y w xx Gi
end

    disp('  ')
    disp('  ')

W;
    disp('Omega(s) = ');
    disp(W)
    disp('  ')
    disp('  ')
    disp('  ')
    disp('============================================')
    disp('             S O L U T I O N ')
```

```
    disp('         C(S) = XL(S)^(-1) * YL(S)')
    disp('============================================')
    disp('  ')

XL = W(1:m,1:m);   % mxm
XL = factor(XL);
disp('The controller''s Denominator is')
disp('XL(s) = ')
disp(XL)
    disp('  ')
    disp('============================================')
    disp('  ')

YL = W(1:m,m+1:m+p);  % mxp
YL = factor(YL);
disp('The controller''s Numerator is')
disp('YL(s) = ')
disp(YL)
    disp('  ')
    disp('============================================')
    disp('  ')
C = XL^(-1)*YL;
C=factor(C);
    disp('  ')
disp('The controller is ');
disp('C(s) = ');
disp(C);
handles.XL=XL;
handles.YL=YL;
handles.C=C;
    disp(' ')
    disp(' ')

disp('*************************************************************
***********')
    disp('                       End - 2st step results ')

disp('*************************************************************
***********')

diary('off');

set(handles.u_tksi,'visible','off');
set(handles.u_ksi,'visible','off');
set(handles.u_tdc,'visible','off');
set(handles.u_dc,'visible','off');
set(handles.pb3,'visible','off');
set(handles.pb5,'visible','on');
set(handles.pb6,'visible','on');
set(handles.pb7,'visible','off');
set(handles.pb8,'visible','on');
set(handles.resul,'visible','off');
% set(handles.popm1,'visible','on');



% To report για το τελικο παραθυρο
if exist('report.txt')
    delete('report.txt');
end
diary('report.txt');
```

```
clc
disp('  ');
disp('Solving Results ')
disp('*******************************************')
disp('  ');
disp('User Inputs')
disp('================================')
disp('Nr(s) = ')
disp(handles.Nr)
disp('  ');
disp('Dr(s) = ')
disp(handles.Dr)
disp('  ');
disp('Dc(s) = ')
disp(poly2sym(handles.Dc))
disp('  ');
disp('Ksi = ')
disp(handles.ksi)
disp('  ');
disp('Results')
disp('================================')
xxxx = sprintf('Observability index Oi = %d\n',handles.mi);
disp(xxxx)
disp('k array :');
for ik = 1:length(handles.k)
   sf = sprintf('k(%d) = %d',ik,handles.k(ik));
   disp(sf)
end
disp('  ');
disp('  ');
disp('The controller''s Denominator is')
disp('================================')
disp('XL(s) = ')
disp(handles.XL)
disp('  ');
disp('  ');
disp('The controller''s Numerator is')
disp('==============================')
disp('YL(s) = ')
disp(handles.YL)
diary('off')
text=textread('report.txt','%s','delimiter','\n','whitespace','');
set(handles.resxl,'visible','on','HorizontalAlignment','left',...
    'string',text);


guidata(hObject,handles);
% if handles.p==1
%   handles.a=display_array(handles.XL,1);
% else
%    handles.a=display_array(sym2poly(handles.XL),1);
% end
% guidata(hObject,handles);

% set(handles.pb2,'visible','off');
% set(handles.pb3,'visible','on');
uiresume;
```

file_viewer.m

```
function varargout=file_viewer(varargin)

varargout{1}=NaN;
fpathfilename=varargin{1};
[pathstr,name,ext,versn]=fileparts(fpathfilename);
filename=[name ext];
usingscratch=0;
if strcmp(ext,'.mat')
    usingscratch=1;
    scratchfilename='foo.bar';
    s=load(fpathfilename);
    f=fieldnames(s);
    delete(scratchfilename);
    clc;
    format;
    diary(scratchfilename);
    for i=1:length(f)
        eval([char(f(i)) '=s.' char(f(i))])
    end
    diary off;
    clc;
    viewfilename=scratchfilename;
else
    viewfilename=fpathfilename;
end
if 2~=exist(viewfilename)
    return;
end

fig=openfig(mfilename,'reuse');
set(fig,'Color',get(0,'defaultUicontrolBackgroundColor'));
handles=guihandles(fig);
guidata(fig,handles);

FigPos=get(0,'DefaultFigurePosition'); % where Matlab likes it to sit
OldUnits=get(fig, 'Units');
set(fig, 'Units', 'pixels');
OldPos=get(fig,'Position');
FigWidth=OldPos(3);
FigHeight=OldPos(4);
if isempty(gcbf)
    ScreenUnits=get(0,'Units');
    set(0,'Units','pixels');
    ScreenSize=get(0,'ScreenSize');
    set(0,'Units',ScreenUnits);
    FigPos(1)=1/2*(ScreenSize(3)-FigWidth);
    FigPos(2)=2/3*(ScreenSize(4)-FigHeight);
else
    GCBFOldUnits=get(gcbf,'Units');
    set(gcbf,'Units','pixels');
    GCBFPos=get(gcbf,'Position');
    set(gcbf,'Units',GCBFOldUnits);
    FigPos(1:2)=[(GCBFPos(1) + GCBFPos(3) / 2) - FigWidth / 2, ...
            (GCBFPos(2) + GCBFPos(4) / 2) - FigHeight / 2];
end
FigPos(3:4)=[FigWidth FigHeight];
set(fig, 'Position', FigPos);
set(fig, 'Units', OldUnits);

maxlines=500; % max lines in file
```

```
text_lines=cell(1,maxlines);
fid=fopen(viewfilename,'r');
i=1;
filetoolarge=0;
while 1
    s=fgets(fid);
    if ~ischar(s), break, end
    s(s<32|s>127)=32; % ignore these characters
    text_lines(i)=cellstr(s);
    i=i+1;
    if i>maxlines
        filetoolarge=1;
        break;
    end
end
fclose(fid);
text_lines=text_lines(1:i-1);
handles.text_lines=text_lines;
guidata(handles.fooey,handles);
set(handles.file_viewer,'String',handles.text_lines,'Value',1);
if filetoolarge
    [s,err]=sprintf('File was truncated to %d
lines',maxlines);warndlg(s,'file_viewer');
end
set(handles.fooey,'Name',filename);
varargout{1}=fig;
if usingscratch
    delete(scratchfilename);
    clc;
end
```

sylforwol.m

```
function R = sylforwol(F,cj,K)
% SYLvester resultant of order 'K' For the column 'cj' of 'F' column
in WOLovich form
%    where F = [Dr ; Nr]

[n,m] = size(F);
p = n-m   ;          % p : number of rows of Dr
                    % m : number of columns of Dr


%------------------------------
for j=1:m
    f(:,1,j) = F(:,j);               % creation of the columns  fj(s)
    q(j) = max_degree(f(:,:,j));         % q(j): the maximum degree
of column j
end                                 % f  : symbolic matrix-column
                                    % fff: polynomial expression of
matrix f

fff(:,1,:) = symbolic_to_poly(f(:,1,:));   % 1st index : index of the
row
                                     % 2nd index : 1 (because they
are
                                     %                matrix of 1
column)
                                     % 3rd index : index of the
column
```

```
%------------------------------
for j=1:m
    for i=1:(p+m)
        le = length(fff{i,1,j});

        for k=1:le
            ffff(i,1,j,k) = fff{i,1,j}(le+1-k);    % 4th index :
number of the coefficient vector
        end

        if k<q(j)+1
            for mam=(k+1):q(j)+1
                ffff(i,1,j,mam) = 0;
            end
        end
    end
end

%--------------------------------------------
x = ffff(:,:,cj,1);
R1 = x;

for i=2:q(cj)+1
    x = ffff(:,:,cj,i);
    R1 = [R1 x];
end

R=R1;
%--------------------------------------------
%--------------------------------------------

Midenikos = zeros(m+p,1);  % Matrix of zeros . Dimensions (m+p)x1

Rrow1 = R1;

for i=1:K-1
    Rrow1 = [Rrow1  Midenikos];
end

R = Rrow1;

for i=2:K
    Rrow = R1;

    for j=1:i-1
        Rrow = [Midenikos Rrow];
    end
    for j=1:K-i
        Rrow = [Rrow Midenikos];
    end

R = [R ; Rrow]; % R : The Sylvester Resultant of order K !!!

end
```

wolovich.m
```
% Επιστρεφει την  WOLOVICH Resultant
% του πινακα F με K order
function M = wolovich(F,K)
[n,m] = size(F);
p = n-m;
M = sylforwol(F,1,K);
if K~=1
    for j=2:m
        M = [M  sylforwol(F,j,K)];
    end
end
```

columns_degrees.m
```
% Η συναρτηση υπολογιζει τους βαθμους των στηλων
% του πινακα F και τους επιστρεφει στον πινακα k
function k = columns_degrees(F)
[p1,m] = size(F);
p = p1-m;
for j=1:m
    fj = F(:,j);
    k(j)=max_degree(fj);
end
```

cvs2.m
```
function  z = csv2(k,arxi)
% Creates a matrix 'a' whose elements are  k  Symbolic Variables from
s='arxi'
% Zs, Zs+1, ...., Zs+k
zed = 'z';

for i=arxi:k+arxi-1
    ii = num2str(i);
    var = [zed ii];
    d = sym(var);
    z(i-arxi+1) = d;
end
```

degsym.m
```
% Η συναρτηση υπολογιζει τον βαθμο ενος πολυωνυμου σε συμβολικη μορφη
% (poly)
function d = degsym(p)
x = sym2poly(p);
y = poly2sym(x);
x = sym2poly(y);
d = length(x)-1;
```

display_array.m
```
% Η συναρτηση εμφανιζει οποιοδποτε πινακα μεσα στο GUI
function a=display_array(mat,str)
pinit;
if str==1
    n1=0.02;
```

```
    n2=0.90;
elseif str==2
    n1=0.20;
    n2=0.60;
else
    n1=0.50;
    n2=0.60;
end
[rows,columns]=size(mat);
if rows==1 & columns==1

a=uicontrol('style','text','backgroundcolor','w','units','normalized'
,...
        'FontSize',10,'position',[n1 0.80 0.30 0.03]);
    handles.a=a;
    set(handles.a,'visible','on');
    if isnumeric(mat)
        set(handles.a,'string',num2str(mat));
    else
        set(handles.a,'string',char(mat));
    end
else
    if isnumeric(mat)
        m=char(pol(mat));
    else
        m=char(mat);
    end
    len=zeros(rows,columns);
    for i=1:rows
        for j=1:columns
            len(i,j)=length(m{i,j});
        end
    end
    lenmax=max(max(len))+1;
    n3=lenmax*0.00555555555555;
    if n3 <= 0.12
        n4=0.03;
    elseif n3 <= 0.24
        n3=0.12;
        n4=0.05;
    else
        n3=0.12;
        n4=0.07
    end
    for i=1:rows
        for j=1:columns
            a(i,j)=uicontrol('style','text','backgroundcolor','w',...
                           'units','normalized','FontSize',10,...
                           'position',[n1+(n3+0.01)*(j-1) n2-
(n4+0.02)*i n3 n4]);
        end
    end
    handles.a=a;
    set(handles.a,'visible','on');
    if isnumeric(mat)
        for i=1:rows
            for j=1:columns
                set(handles.a(i,j),'string',num2str(mat(i,j)));
            end
        end
    else
```

```
        mat=char(mat);
        for i=1:rows
            for j=1:columns
                set(handles.a(i,j),'string',mat(i,j))
            end
        end
    end
end
```

dt_fun.m

```
% Υπολογισμος του sos πινακα
DDc=poly_to_symbolic(Dc);
for q = 1:(k(1)+ksi(i)+1)
    sos(q) = s^(q-1);
end

for j= 2:length(k)
    for q = 1:(k(j)+ksi(i)+1)
        xx(q) = s^(q-1);
    end
    sos = blkdiag(sos,xx);
    clear xx
end

%------------------ gia to d1

sost = transpose(sos);


y = sym2poly(DDc(i,i));

    for w=1:(length(y)/2)
       t = y(w);
       y(w) = y(length(y)+1-w);
       y(length(y)+1-w) = t;
    end

w=1;

[ps,ms] = size(sost);
for q=1:ps
    if sost(q,i)==0
        dt(q) = 0;
    else
        dt(q) = y(w);
        w = w+1;
    end
end
dt
```

dt_fun1.m

```
for q = 1:(k(1)+ksi(i)+1)
        sos(q) = s^(q-1);
    end
```

```
    for j= 2:length(k)
        for q = 1:(k(j)+ksi(i)+1)
            xx(q) = s^(q-1);
        end
        sos = blkdiag(sos,xx);
        clear xx
    end
```

find_n.m

```
% Υπολογίζεται ο πίνακας n του συστήματος
n(1) = ni(1);
gr = r(1)+1;
st = r(1)+1;
[pEE,mEE]=size(EE);
sumn = n(1);
if ik~=1
for ikk=2:ik-1
    n(ikk)=0;
    while gr<=pEE & st<=mEE & EE(gr,st)==0
        n(ikk)=n(ikk)+1;

        gr=gr+1;
    end
    sumn = sumn+n(ikk);
    if ikk+1<=length(r)
        st = st+r(ikk+1);
    end
end
n(ik)=ni(ik)-sumn;
end % end_if
```

find_r.m

```
% Υπολογίζω τον πίνακα r του συστήματος

r(1) = ro(1);


gr = r(1)+ni(1)+1;
st = r(1)+1;

[pEE,mEE]=size(EE);

sumr=r(1);
ik=2;

while sumr<ro(i)
    r(ik)=0;
    while gr<=pEE & st<=mEE & EE(gr,st)==1
        r(ik)=r(ik)+1;

        gr=gr+1;
        st=st+1;
    end

    sumr = sumr+r(ik);
    ik = ik+1;
end
ik=ik-1;
```

find_smp.m

```
% smp

smp = eye(m+p);

for kk = 1:ksi(i)
    sb = s^kk*eye(m+p);

    smp = [smp ; sb];

end
smp
```

find_wis.m

```
%gia to w_i

d = EE(ss(i)+1,1:r(1));
d = [d zeros(1,ni(1))];
st = r(1)+1;

for kkk=2:ik
    d = [d EE(ss(i)+1,st:st+r(kkk)-1)];
    d = [d zeros(1,n(kkk))];
end
d;
sf = sprintf('d%d = %d',i);
disp(sf)
disp(d)

[pGi,mGi]=size(Gi);

lez = 0;
if pGi ~= 0
    z = csv2(pGi,arxi)
    lez = length(z);

    wi = d+z*Gi;

    sf = sprintf('w%d = %d',i);
    disp(sf)
    disp(wi)

else
    wi = d;
    sf = sprintf('w%d = %d',i);
    disp(sf)
    disp(wi)
end


find_smp

wis = wi*smp;

sf = sprintf('w%d(s) = %d',i);
```

```
disp(sf)
disp(wis)
```

max_degree.m

```
% Υπολογιζει τον μεγιστο βαθμο των πολυωνυμων που περιεχει ενας
πινακας
function maxd = max_degree(pin)
[p,m] = size(pin);
maxd = 0;

for i=1:p
    for j=1:m
        if degsym(pin(i,j)) > maxd
            maxd = degsym(pin(i,j));
        end
    end
end
```

poly_to_symbolic.m

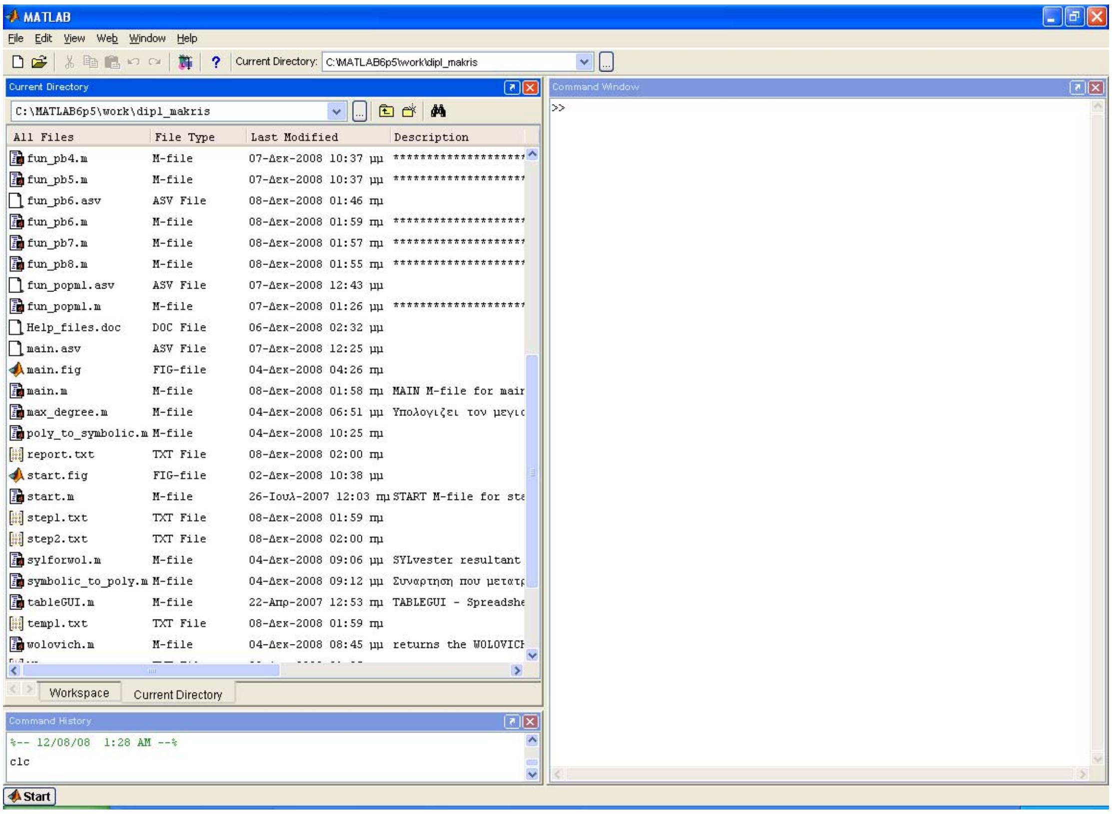
```
function newpin = poly_to_symbolic(pin)

[p,m]=size(pin);

for i=1:p
    for j=1:m
        if iscell(pin)==1
            y = pin{i,j};
            x=poly2sym(y,'s');
        else
            y = pin(i,j);
            x=poly2sym(y,'s');
        end
        newpin(i,j)=x;
    end
end
```

symbolic_to_poly.m

```
% Συναρτηση που μετατρεπει εναν πινακα απο συμβολικη
% σε πουωνυμικη μορφη
function SP = symbolic_to_poly(pin)
[p,m] = size(pin);

for i=1:p
    for j=1:m
        SP{i,j} = sym2poly(pin(i,j));
    end
end
```

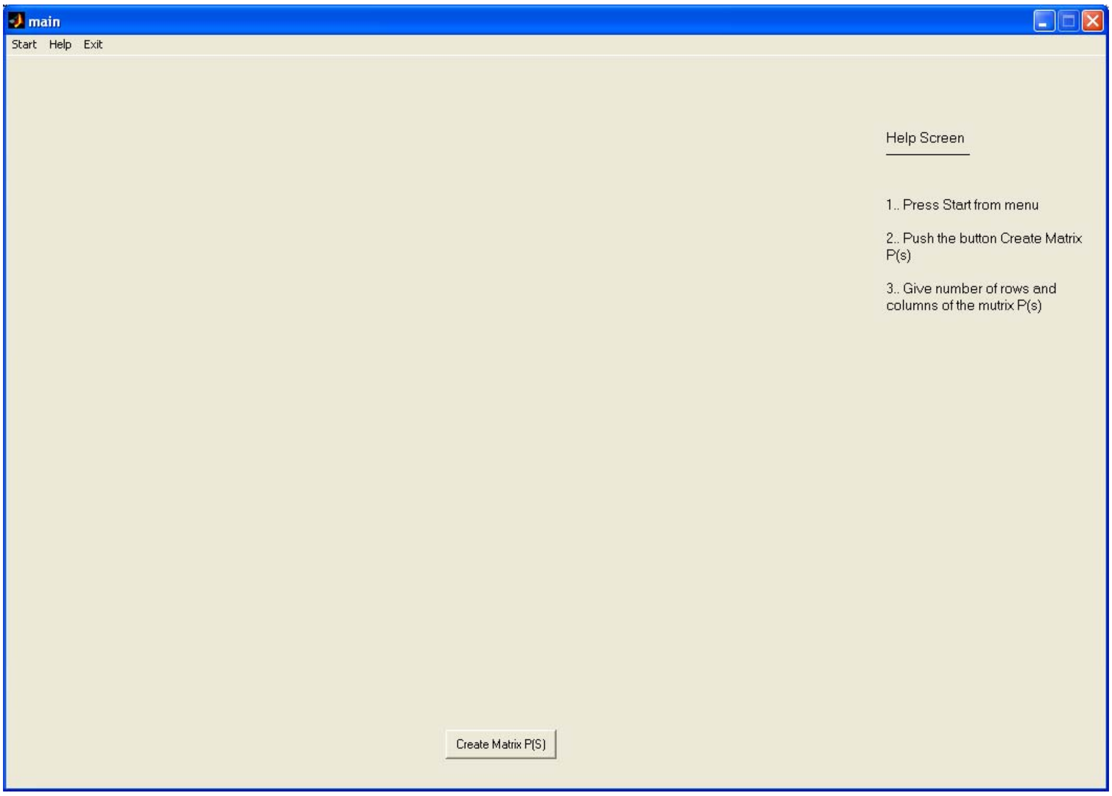
3.2 Εγχειρίδιο χρήσης της εφαρμογής

Για να ξεκινήσει η εφαρμογή χρειάζεται να είναι εγκατεστημένο το πακέτο POLYX στον MATLAB . (Κοιτάξτε την παράγραφο 2.2.4 για τον τρόπο εγκατάστασης του POLYX) .

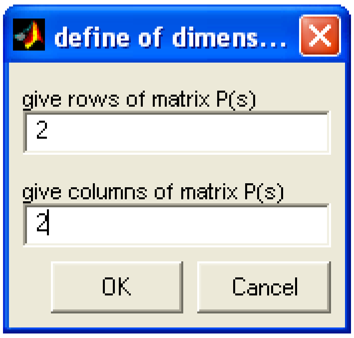


Από το βασικό παράθυρο του MatLab κάντε διπλό κλικ στο main.fig . Το πρόγραμμα θα εκτελεστεί αυτόματα

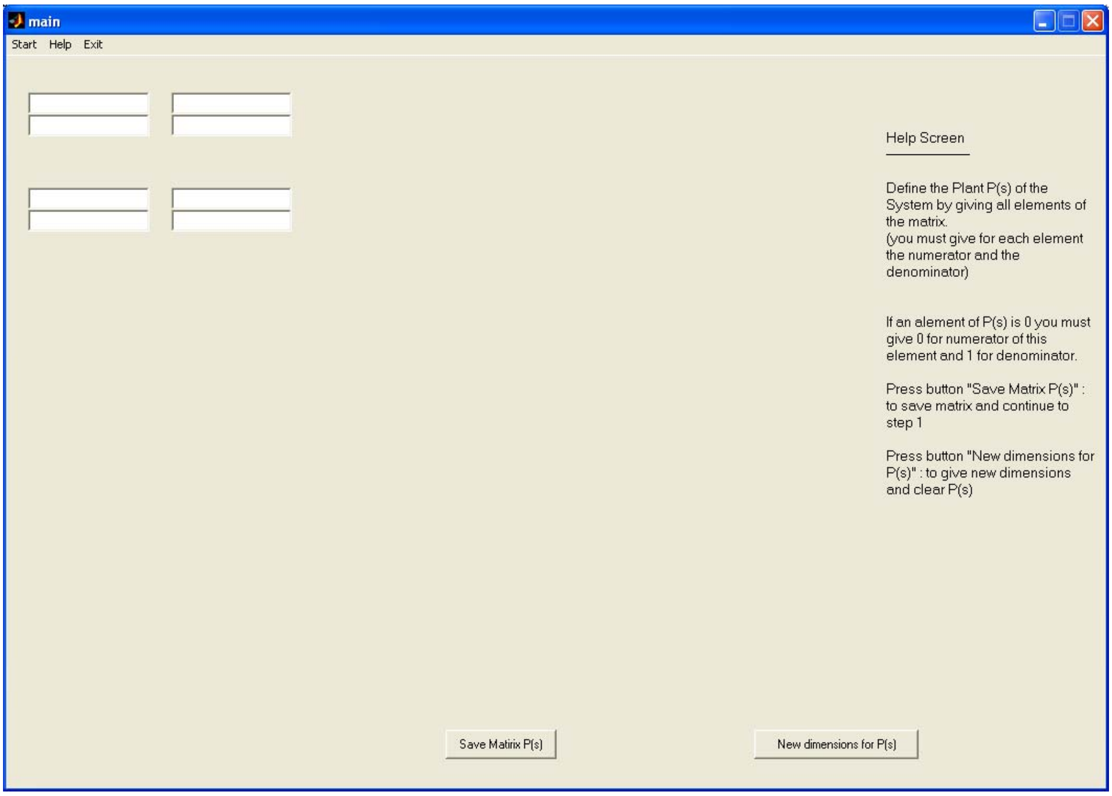
Από το μενού επιλογής κάντε κλικ στην επιλογή Start οπότε έχουμε το παρακάτω παράθυρο :



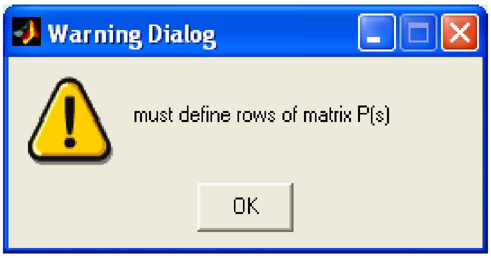
Στο οποίο εμφανίζεται το κουμπί (Create Matrix P(s)) με το οποίο μπορούμε να δημιουργήσουμε τον πίνακα P(s). Κάνουμε κλικ στο κουμπί



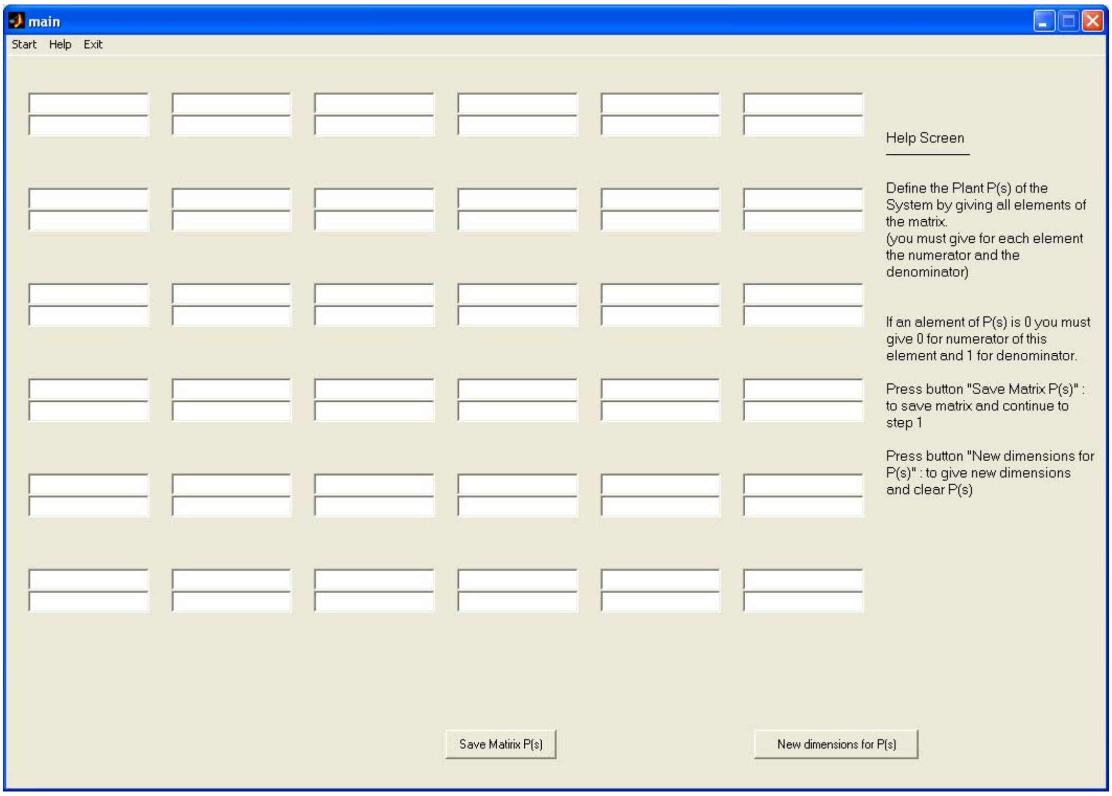
Στο παραπάνω παράθυρο δίνουμε τις διαστάσεις του πίνακα P(s) που θέλουμε να εισάγουμε . Έστω 2 x 2 οπότε με το που θα κάνουμε OK θα εμφανιστεί στην οθόνη μας :



Οι διαστάσεις του πίνακα μπορούν να πάρουν τιμές από 1 μέχρι 6 (εάν εισάγουμε λάθος στοιχεία ή δεν εισάγουμε στοιχεία βγαίνει ανάλογο μήνυμα) Εάν για παράδειγμα δεν βάλουμε μία διάσταση για τον πίνακα τότε θα βγει το μήνυμα :



Εάν βάλουμε για διαστάσεις του πίνακα 6 x 6 τότε θα έχουμε την εξής έξοδο

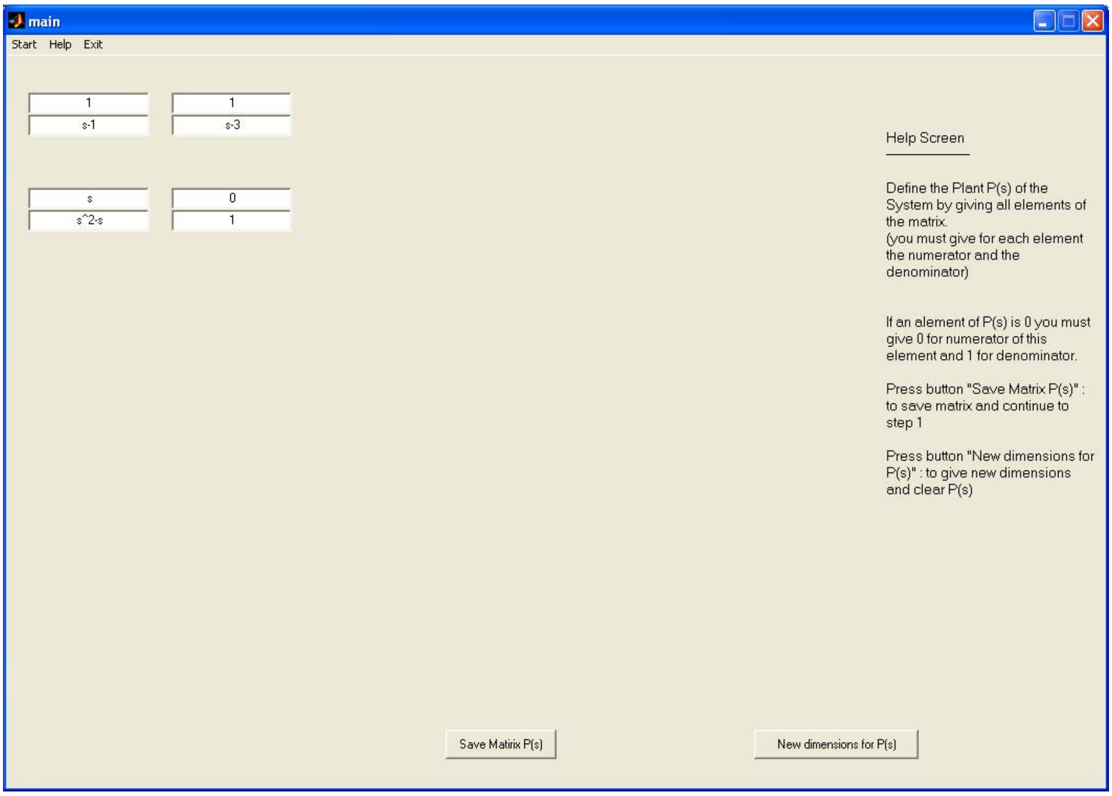


Στο παράθυρο με τον πίνακα P(s) πρέπει να συμπληρώσουμε όλα τα στοιχεία έτσι ώστε να είναι δυνατή η εισαγωγή των στοιχείων στο πρόγραμμα

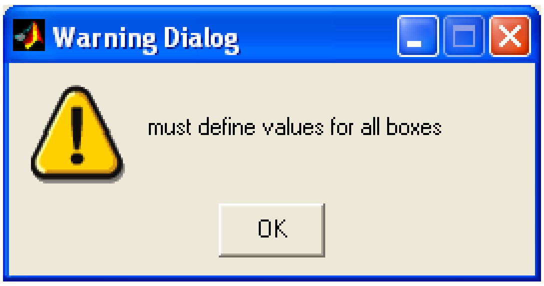
Εάν κάποιο στοιχείο του πίνακα είναι 0 , τότε πρέπει να βάλουμε 0 στον αριθμητή της αντίστοιχης θέσης και 1 στον παρονομαστή .

Εάν θέλουμε να αλλάξουμε οποιαδήποτε στιγμή τις διαστάσεις του πίνακα αρκεί να πατήσουμε το κουμπί «New dimensions for P(s)»

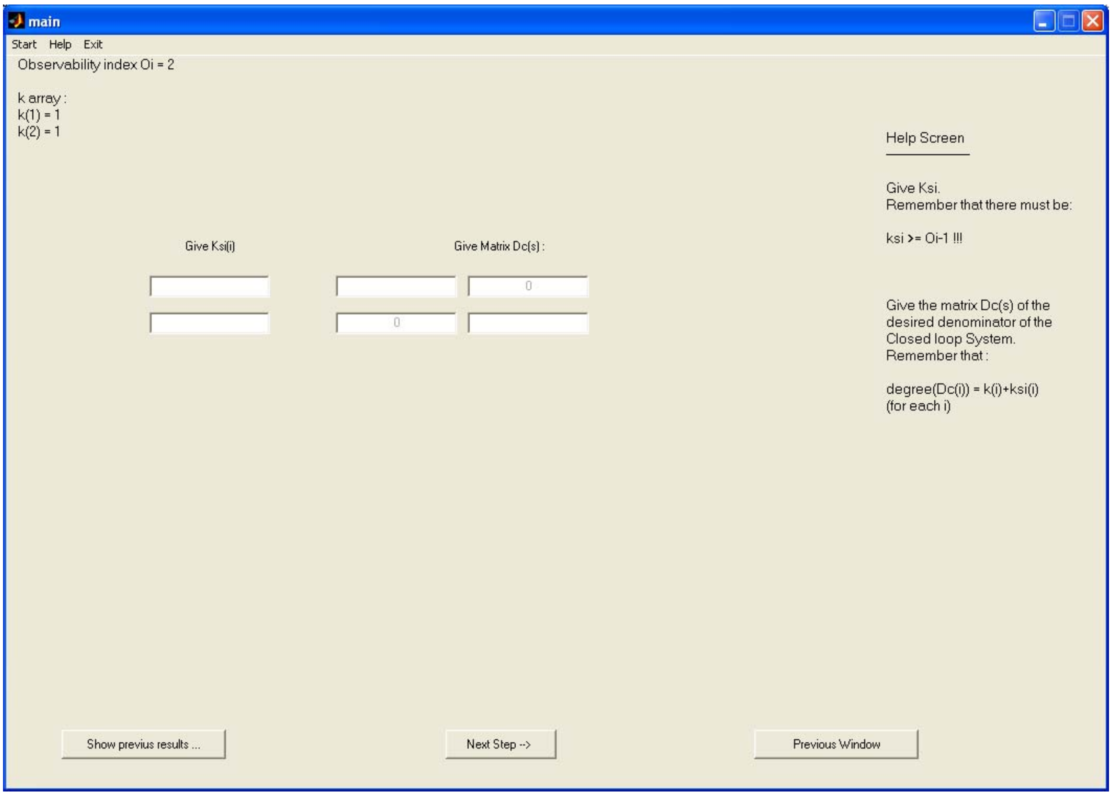
Για παράδειγμα σε έναν πίνακα 2x2 μπορούμε να βάλουμε



Εάν δεν συμπληρώσουμε κάποιο στοιχείο του πίνακα τότε βγαίνει προειδοποιητικό μήνυμα το οποίο ζητά να συμπληρωθούν όλα τα στοιχεία και το πρόγραμμα δεν συνεχίζει.



Εάν συμπληρώσουμε όλα τα στοιχεία σωστά και πατήσουμε Save Matrix P(s), τότε θα εμφανιστεί το επόμενο παράθυρο :

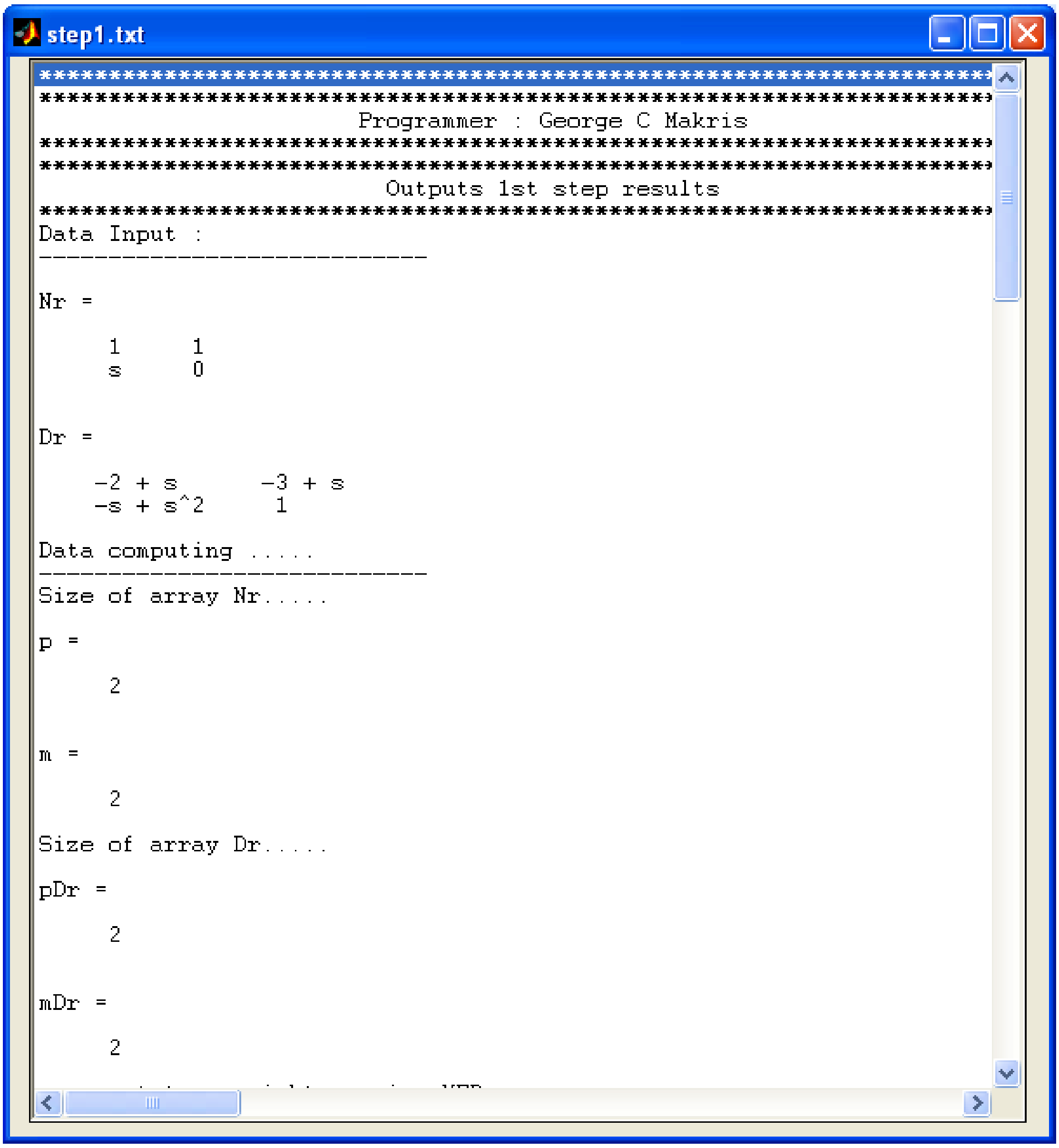


Όπου στο επάνω μέρος του παραθύρου φαίνονται Ο δείκτης ελεξιμότητας του συστήματος καθώς και οι τιμές των K_i που υπολογίστηκαν .

Στο νέο παράθυρο μας ζητείται να δώσουμε τις τιμές των ξi (όσες απαιτούνται από το σύστημα) καθώς και τον πίνακα Dc.

Εάν θέλουμε να δούμε όλους τους προηγούμενους υπολογισμούς που έκανε η εφαρμογή τότε αρκεί να κάνουμε κλικ στο **Show previous results** .

Τότε εμφανίζεται ένα νέο παράθυρο με όλα τα αποτελέσματα που έκανε η εφαρμογή μέχρι τον υπολογισμό των k και του δείκτη της ελεξιμότητας συστήματος.

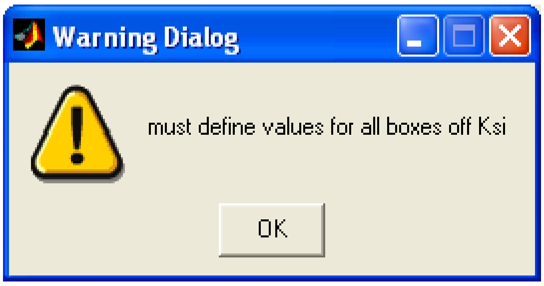

step1.txt

```
*********************************************************************
*********************************************************************
                       Programmer : George C Makris
*********************************************************************
*********************************************************************
                         Outputs 1st step results
*********************************************************************
Data Input :
----------------------------

Nr =

     1      1
     s      0


Dr =

    -2 + s       -3 + s
    -s + s^2      1

Data computing .....
----------------------------
Size of array Nr.....

p =

     2


m =

     2

Size of array Dr.....

pDr =

     2


mDr =

     2
```

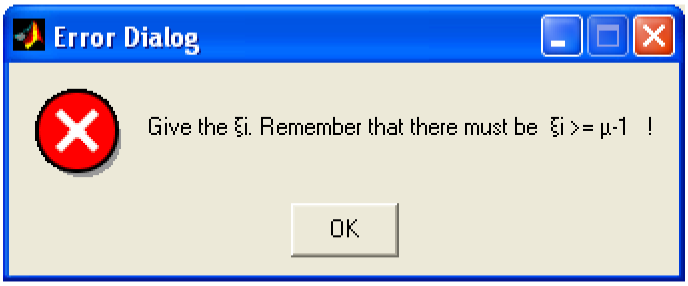
Το νέο παράθυρο περιέχει όλα τα ενδιάμεσα βήματα και αποτελέσματα των υπολογισμών που έκανε η εφαρμογή . Τα αποτελέσματα αποθηκεύονται μέσα στο αρχείο **step1.txt** στο φάκελο της εφαρμογής . Το αρχείο είναι διαθέσιμο στον χρήση μέχρι να εισαχθεί νέος πίνακας και υπολογιστούν στοιχεία από την αρχή.

Εάν κλείσει το παράθυρο ο χρήστης τότε εάν έκανε λάθος εισαγωγή στοιχείων στον πίνακα P(s) μπορεί να επιστρέψει με τον πλήκτρο **Previous Window** έτσι ώστε να διορθώσει όποιο στοιχείο θέλει.

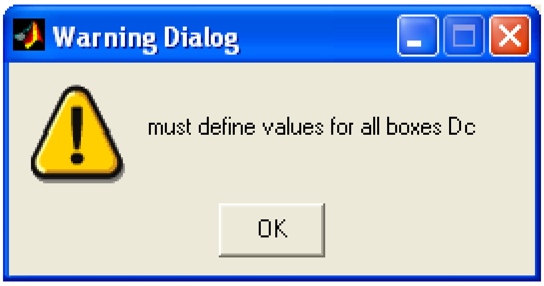
Βάζοντας τα δεδομένα στο νέο παράθυρο τότε πατώντας το πλήκτρο **Next Step -->** γίνεται έλεγχος των στοιχείων εισαγωγής και εφόσον πληρούν όλα τα κριτήρια εισόδου τότε προχωρά στο επόμενο παράθυρο , όμως εάν δεν πληρούνται τα κριτήρια που απαιτεί ο αλγόριθμος τότε βγαίνουν αντίστοιχα προειδοποιητικά μηνύματα



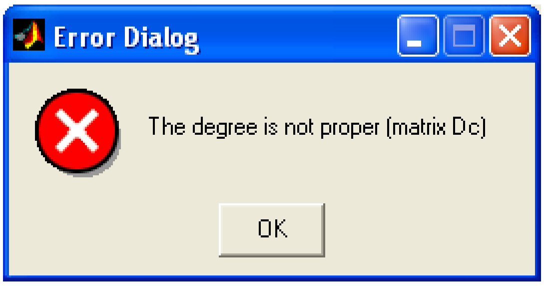
Εάν δεν έχουμε δώσει τιμές για τον πίνακα ξi



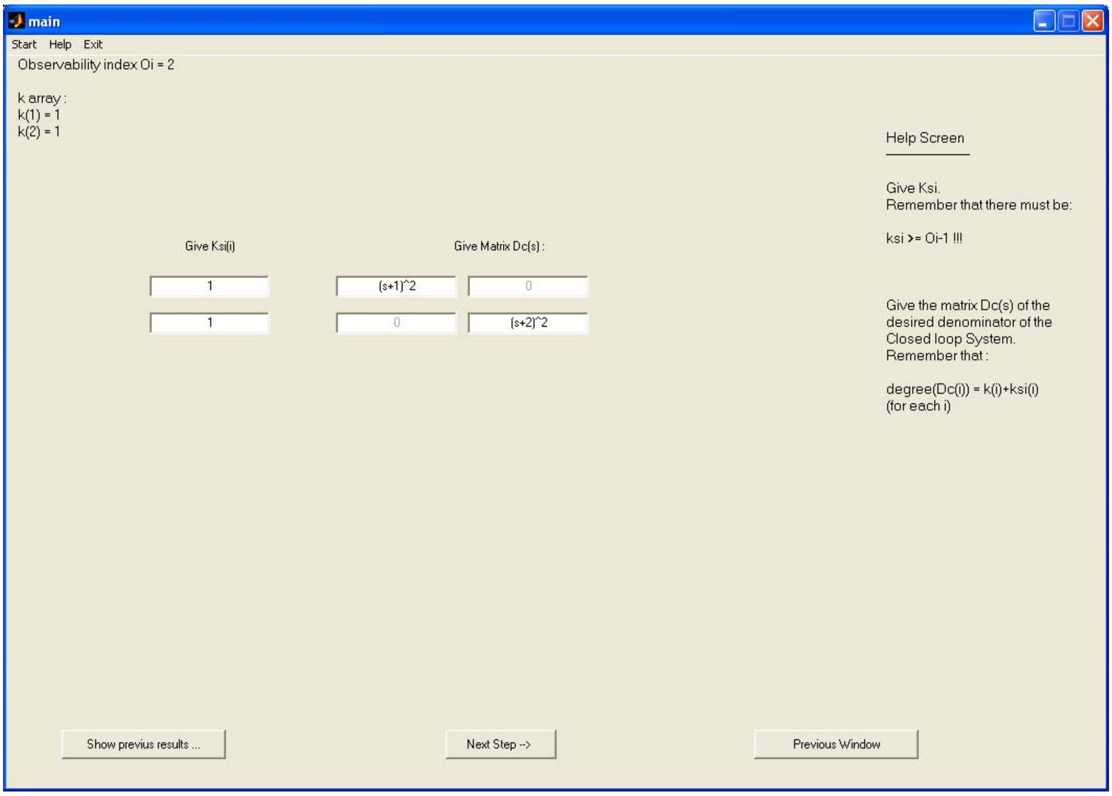
Εάν οι τιμές των ξi <μ-1 (μ = δείκτης ελεξιμότητας του συστήματος)



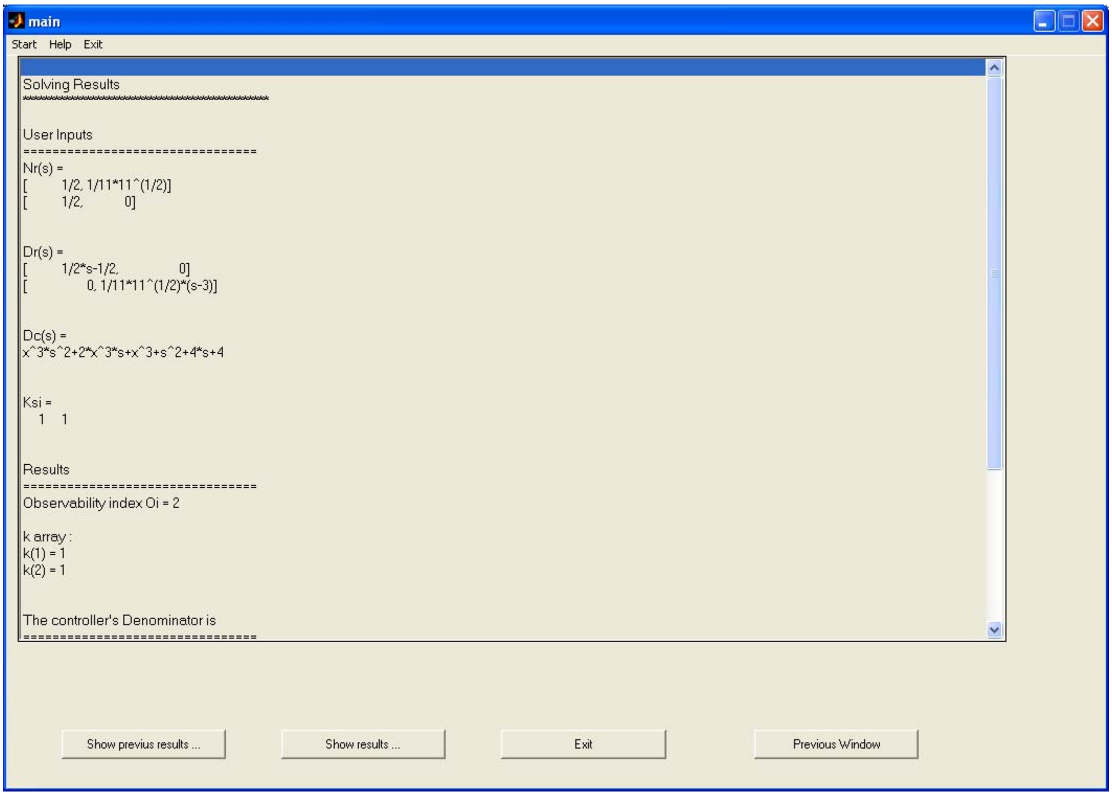
Εάν δεν έχουμε δώσει τιμές στον πίνακα Dc(s)



Εάν ο βαθμός Dc(x,x) <> K(x) + ξ(x)

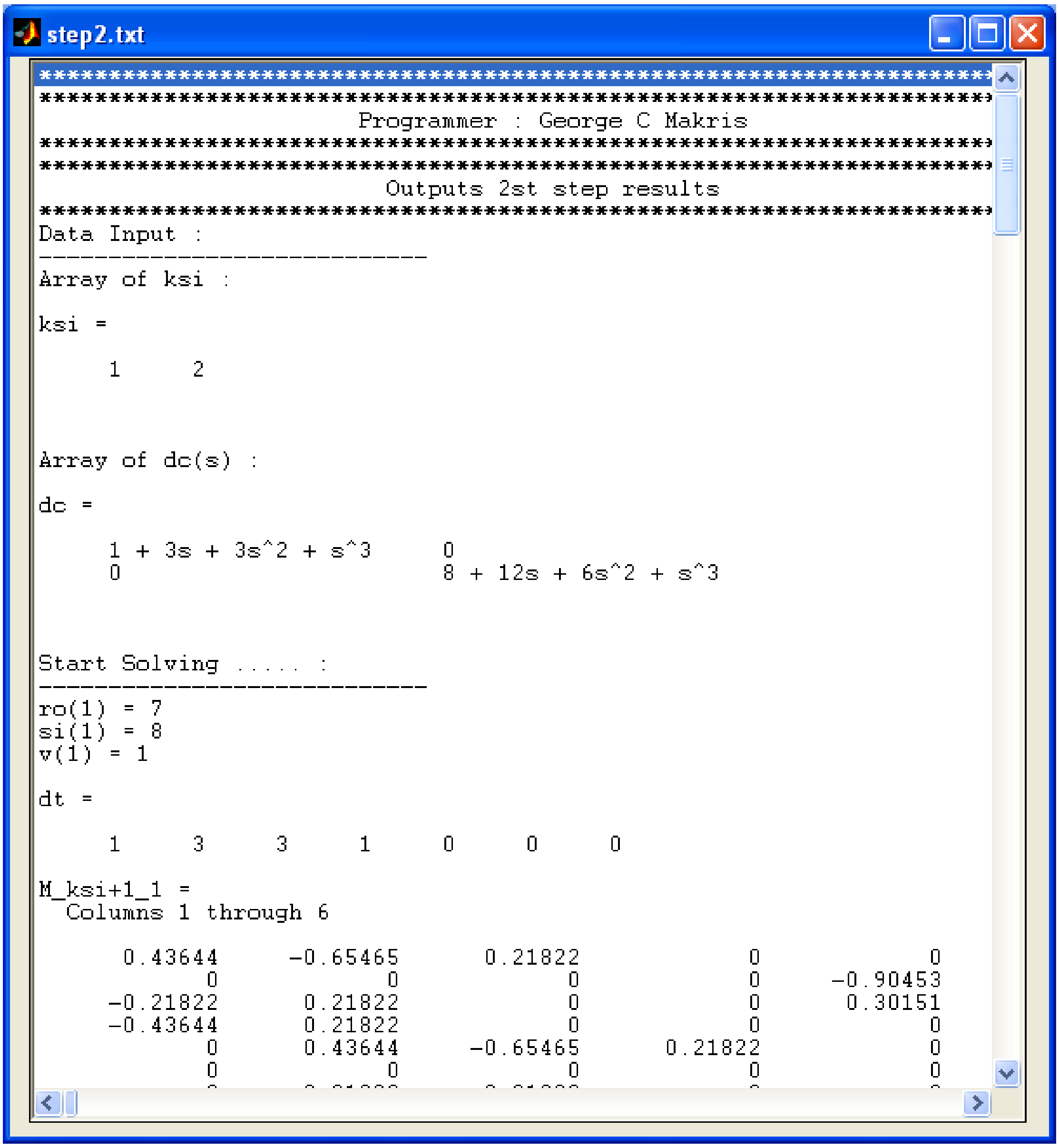


Εάν δώσουμε τα σωστά στοιχεία και πατήσουμε **Next Step -->** τότε θα εμφανιστεί το :



Το οποίο περιέχει τα αποτελέσματα των υπολογισμών καθώς και κουμπιά για να δούμε τους ενδιάμεσους υπολογισμούς όλων των πινάκων

Για παράδειγμα εάν πατήσουμε το πλήκτρο **Show results** τότε θα πάρουμε :



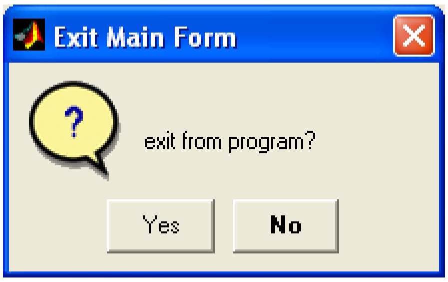
step2.txt

```
**********************************************************************
**********************************************************************
                       Programmer : George C Makris
**********************************************************************
**********************************************************************
                         Outputs 2st step results
**********************************************************************
Data Input  :
----------------------------
Array of ksi  :

ksi =

     1      2


Array of dc(s)  :

dc =

     1 + 3s + 3s^2 + s^3      0
     0                        8 + 12s + 6s^2 + s^3


Start Solving ..... :
----------------------------
ro(1) = 7
si(1) = 8
v(1) = 1

dt =

     1      3      3      1      0      0      0

M_ksi+1_1 =
  Columns 1 through 6

      0.43644     -0.65465      0.21822            0            0
            0            0            0            0     -0.90453
     -0.21822      0.21822            0            0      0.30151
     -0.43644      0.21822            0            0            0
            0      0.43644     -0.65465      0.21822            0
            0            0            0            0            0
```

Στο παράθυρο εμφανίζονται όλα τα αποτελέσματα των ενδιάμεσων πινάκων και των αναλυτικών στοιχείων του αλγορίθμου

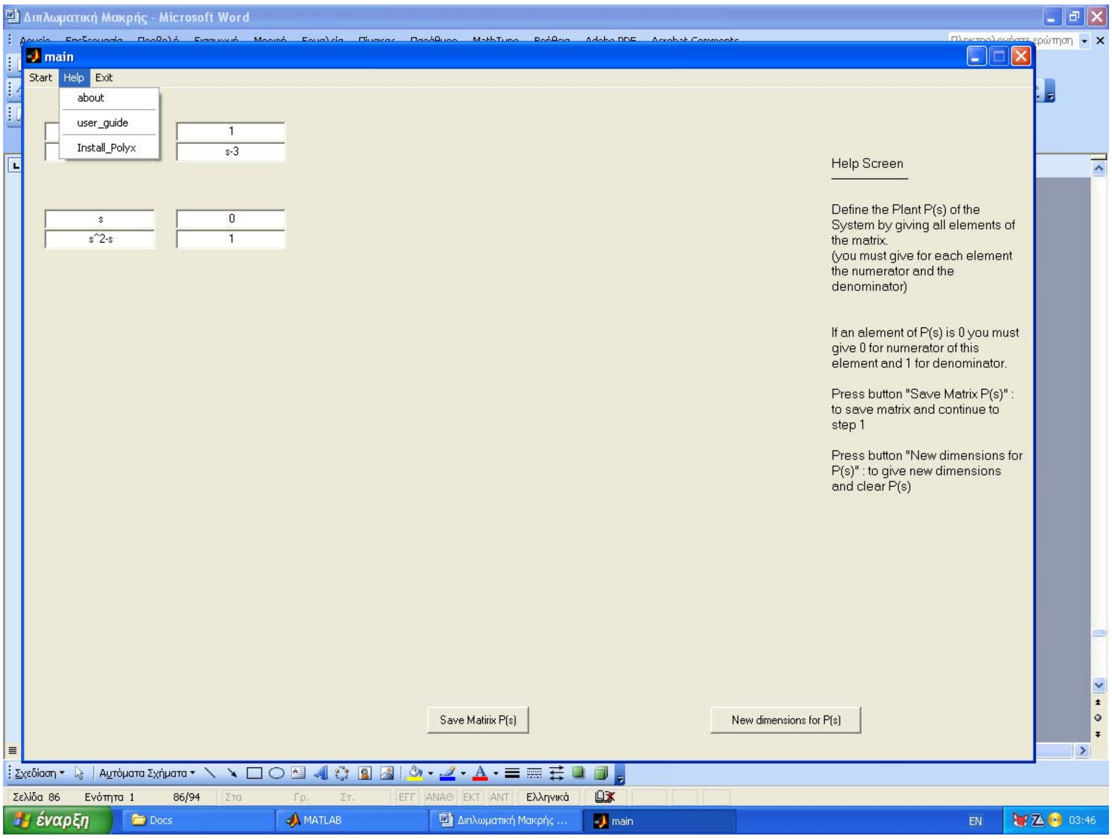
Με το κουμπί **Previous Window** μας δίνεται η δυνατότητα να διορθώσουμε τα στοιχεία ξi καθώς και τα στοιχεία του πίνακα Dc .

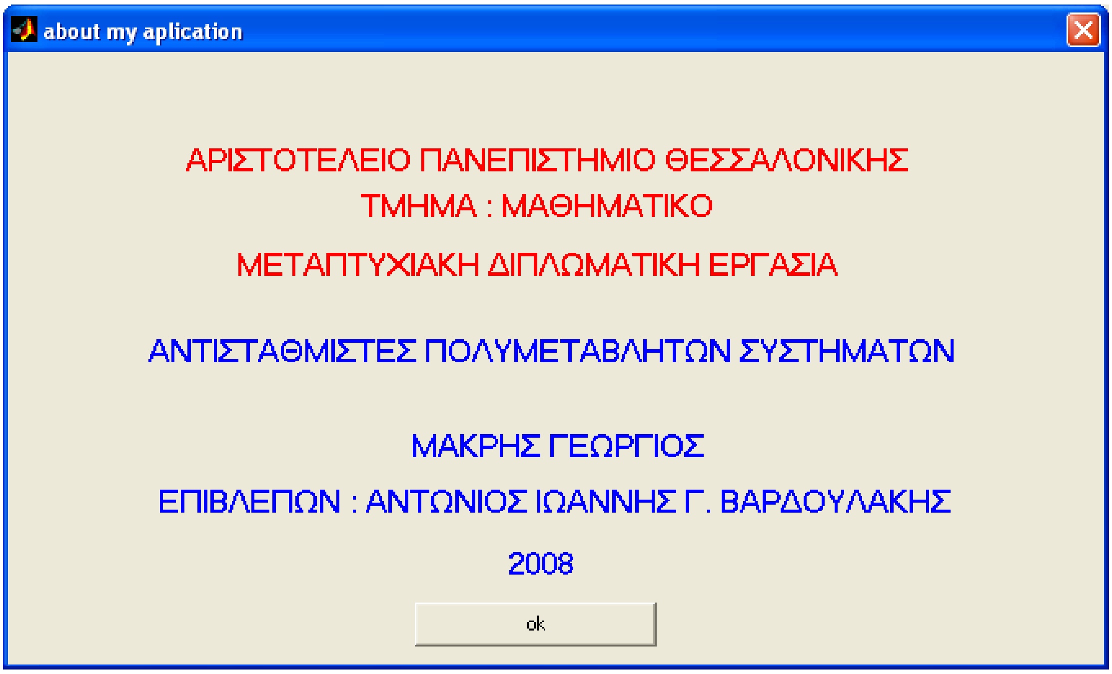
Με το κουμπί έξοδος

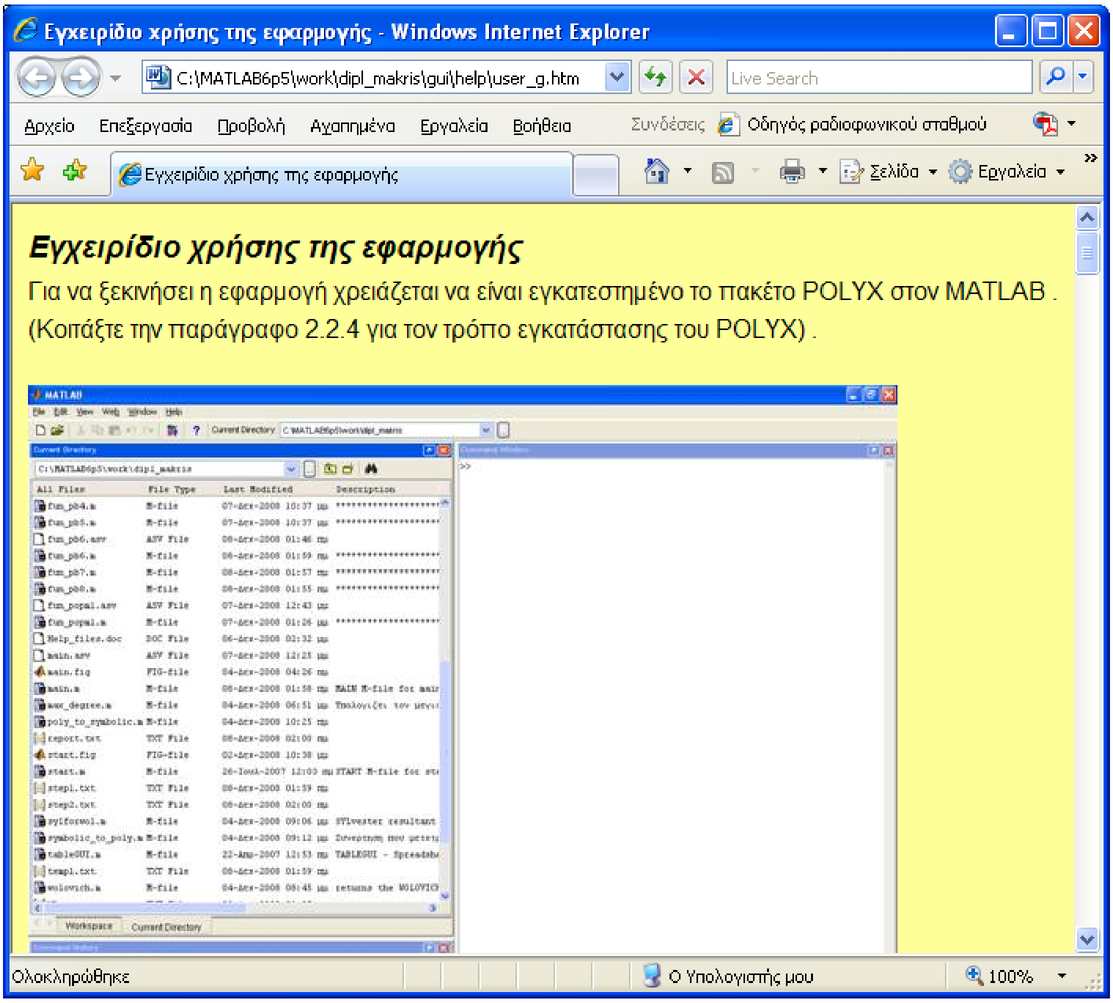


Μας ζητείται να βγούμε από την εφαρμογή.

Πατώντας οποιαδήποτε στιγμή στο μενού Help μας δίνεται η δυνατότητα να δούμε τόσο πληροφοριές για την εφαρμογή , το παρόν εγχειρίδιο της εφαρμογής σε HTML καθώς και τον τρόπο που κάνουμε εκγατάσταση το POLYX στο MatLab σε HTML




about my aplication
ΑΡΙΣΤΟΤΕΛΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΟΝΙΚΗΣ
ΤΜΗΜΑ : ΜΑΘΗΜΑΤΙΚΟ
ΜΕΤΑΠΤΥΧΙΑΚΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ
ΑΝΤΙΣΤΑΘΜΙΣΤΕΣ ΠΟΛΥΜΕΤΑΒΛΗΤΩΝ ΣΥΣΤΗΜΑΤΩΝ
ΜΑΚΡΗΣ ΓΕΩΡΓΙΟΣ
ΕΠΙΒΛΕΠΩΝ : ΑΝΤΩΝΙΟΣ ΙΩΑΝΝΗΣ Γ. ΒΑΡΔΟΥΛΑΚΗΣ
2008
ok


Εγχειρίδιο χρήσης της εφαρμογής - Windows Internet Explorer
C:\MATLAB6p5\work\dipl_makris\gui\help\user_g.htm
Live Search
Αρχείο
Επεξεργασία
Προβολή
Αγαπημένα
Εργαλεία
Βοήθεια
Συνδέσεις
Οδηγός ραδιοφωνικού σταθμού
Εγχειρίδιο χρήσης της εφαρμογής
Σελίδα
Εργαλεία
Εγχειρίδιο χρήσης της εφαρμογής
Για να ξεκινήσει η εφαρμογή χρειάζεται να είναι εγκατεστημένο το πακέτο POLYX στον MATLAB .
(Κοιτάξτε την παράγραφο 2.2.4 για τον τρόπο εγκατάστασης του POLYX) .
Ολοκληρώθηκε
Ο Υπολογιστής μου
100%

3.3 Αρχεία Εξόδου από την εφαρμογή

Step1.txt

```
*********************************************************************
*********************************************************************
           Programmer : George C Makris
*********************************************************************
*********************************************************************
            Outputs 1st step results
*********************************************************************
Data Input :
----------------------------

Nr =

   1


Dr =

   s^3

Data computing .....
----------------------------
Size of array Nr.....

p =

   1


m =

   1

Size of array Dr.....

pDr =

   1


mDr =

   1

New arrays after factoring
----------------------------

Nr =

1
```

```
Dr =

s^3

array P(s) :
----------------------------

P =

1/s^3

New arrays after factoring P(s)
----------------------------
Nr(s) =
1

Dr(s) =
s^3

System (inputs - outputs)
----------------------------


The inputs of the system are : m = 1


The outputs of the system are : p = 1
Calculating Observability index
---------------------------------
F(s)=

F =

[ s^3]
[   1]

k(1) = 3

M =

     0     0     0     1
     1     0     0     0


M =

     0     0     0     1     0
     1     0     0     0     0
     0     0     0     0     1
     0     1     0     0     0


M =

     0     0     0     1     0     0
```

1	0	0	0	0	0
0	0	0	0	1	0
0	1	0	0	0	0
0	0	0	0	0	1
0	0	1	0	0	0

Observability index Oi = 3

****
End - 1st step results
****

step2.txt

```
**************************************************************************
**************************************************************************
            Programmer : George C Makris
**************************************************************************
**************************************************************************
             Outputs 2st step results
**************************************************************************
Data Input :
----------------------------
Array of ksi :

ksi =

     2



Array of dc(s) :

dc =

   32 + 80s + 80s^2 + 40s^3 + 10s^4 + s^5



Start Solving ..... :
----------------------------
ro(1) = 6
si(1) = 6
v(1) = 0

dt =

    32    80    80    40    10     1

M_ksi+1_1 =
     0     0     0     1     0     0
     1     0     0     0     0     0
     0     0     0     0     1     0
     0     1     0     0     0     0
     0     0     0     0     0     1
     0     0     1     0     0     0


EE =

     1     0     0     0     0     0
     0     1     0     0     0     0
     0     0     1     0     0     0
     0     0     0     1     0     0
     0     0     0     0     1     0
     0     0     0     0     0     1
    40    32    10    80     1    80
```

```
Gi =

   Empty matrix: 0-by-6

d1 =
    40    32    10    80     1    80

w1 =
    40    32    10    80     1    80


smp =

[   1,   0]
[   0,   1]
[   s,   0]
[   0,   s]
[ s^2,   0]
[   0, s^2]

w1(s) =
[    s^2+10*s+40, 32+80*s+80*s^2]


W =

[    s^2+10*s+40, 32+80*s+80*s^2]

--------------------------------------------


Omega(s) =
[    s^2+10*s+40, 32+80*s+80*s^2]



==============================================
          S O L U T I O N
       C(S) = XL(S)^(-1) * YL(S)
==============================================

The controller's Denominator is
XL(s) =
s^2+10*s+40


==============================================

The controller's Numerator is
YL(s) =
32+80*s+80*s^2
```

==

The controller is
C(s) =
16*(2+5*s+5*s^2)/(s^2+10*s+40)

****
End - 2st step results
****

4. Βιβλιογραφία – Πηγές

[1] Vardoulakis A.I.G., (1991) Linear Multivariable Control, Algebraic analysis and synthesis Methods , WIley , Chichester , UK

[2] Antoniou , E.N. , Vardulakis A.I.G , (2003) , A numerical method for the computation of proper denominator assigning compenstors for strictly proper plants , IMA J. Math. Control Inform 22

[3] Antoniou , E.N. , Vardulakis A.I.G , (2005) Generalises Wolvich Resultant

[4] Κωνσταντίνος Παπαρρίζος , 2004, MatLab 6.5 , Εκδόσεις Ζυγός , Θεσσαλονίκη

[5] Marchhand, P, Holland ,O.T. , 2003, Graphics and GUIS with MATLAB third edition, Chapman & Hall/CRC , New York

[6] Κεσογλίδης , Α, 2006, Μεταπτυχιακή Εργασία «Υλοποίηση Αντισταθμιστών Πολυμεταβλητών Συστημάτων με την βοήθεια Ηλεκτρονικού Υπολογιστή , Τμήμα Μαθηματικό , ΑΠΘ, Θεσσαλονίκη

[7] Βαρσάμης , Δ, 2006, Ανάπτυξη διαδραστικού περιβάλλοντος (GUI) για την Ανάλυση και Σύνθεση Πολυμεταβλητών Συστημάτων Αυτομάτου Ελέγχου , Τμήμα Μαθηματικό , ΑΠΘ, Θεσσαλονίκη.

[8] Όλα τα αρχεία Βοήθειας και τα εγχειρίδια αναφοράς του MATLAB 6.5

[9] Όλα τα αρχεία Βοήθειας και τα εγχειρίδια αναφοράς του πακέτου POLYX