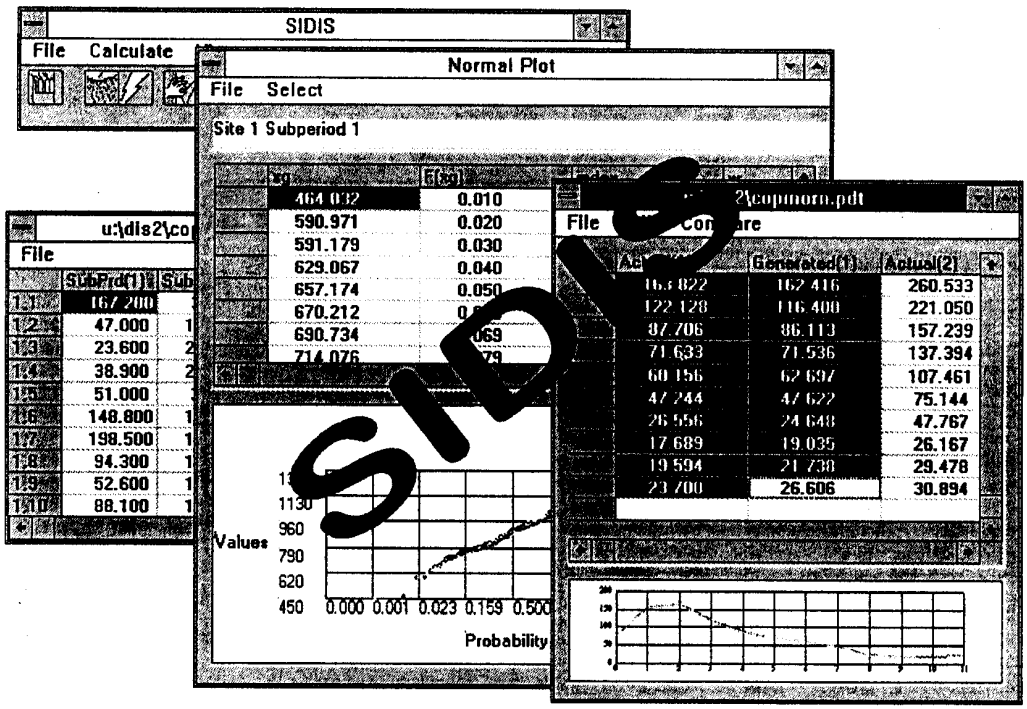
Εθνικό Μετσόβιο Πολυτεχνείο
Τμήμα Πολιτικών Μηχανικών
Τομέας Υδατικών πόρων, Υδραυλικών και Θαλάσσιων έργων
Επιβλέπων : Δημήτρης Κουτσογιάννης

ΑΝΑΠΤΥΞΗ ΛΟΓΙΣΜΙΚΟΥ ΓΙΑ ΤΗΝ ΠΟΛΥΜΕΤΑΒΛΗΤΗ ΣΤΟΧΑΣΤΙΚΗ ΠΡΟΣΟΜΟΙΩΣΗ ΥΔΡΟΛΟΓΙΚΩΝ ΧΡΟΝΟΣΕΙΡΩΝ ΜΕ ΧΡΗΣΗ ΕΠΙΜΕΡΙΣΤΙΚΩΝ ΜΕΘΟΔΩΝ

Αλέξανδρος Μανέτας



Αθήνα Φεβρουάριος 1994

Περίληψη εργασίας

Το αντικείμενο αυτής της εργασίας είναι ο επιμερισμός υδρολογικών χρονοσειρών με τη χρήση στοχαστικών διαδικασιών. Η μεθοδολογία που παρουσιάζουμε είναι εντελώς πρωτότυπη και το βασικό πλεονέκτημά της σε σχέση με τις ήδη υπάρχουσες είναι η απλότητά της. Η απλότητα αυτή δεν είναι βέβαια εις βάρος της ποιότητας των αποτελεσμάτων, όπως θα δούμε παρακάτω.

Στο πρώτο κεφάλαιο εξηγούμε, σε γενικές γραμμές, τι είναι ο επιμερισμός, ποιά είναι η χρησιμότητά του και ποιές είναι οι διαφορές του μοντέλου που αναπτύξαμε, με τα αντίστοιχα μοντέλα της διεθνούς βιβλιογραφίας. Στο δεύτερο κεφάλαιο παρουσιάζουμε την πλήρη μαθηματική διατύπωση του μοντέλου και ένα απλό παράδειγμα εφαρμογής. Στο τρίτο αναφερόμαστε σε θέματα σχετικά με τον υπολογιστή όπως τον τρόπο με τον οποίο παράγει τυχαίους αριθμούς και το χρόνο που απαιτεί για να κάνει επιμερισμό, ανάλογα με την ποσότητα των δεδομένων. Στο τέταρτο κεφάλαιο εξηγούμε συνοπτικά τον τρόπο λειτουργίας του προγράμματος και στο πέμπτο και τελευταίο παρουσιάζουμε τα αποτελέσματα που προέκυψαν από μια εφαρμογή του προγράμματος με δεδομένα από τη λεκάνη του Μόρνου. Ακολουθεί παράρτημα με τον πηγαίο κώδικα για την περίπτωση που κάποιος θέλει να χρησιμοποιήσει έναν απο τους αλγόριθμους που γράψαμε.

Η εργασία αυτή έγινε υπο την ακούραστη επίβλεψη του Δημήτρη Κουτσογιάννη, χωρίς την βοήθεια του οποίου τα πράγματα θα ήταν πολύ πιο δύσκολα. Τέλος θέλω να ευχαριστήσω τον Γιώργο Τσακαλία για τα ουσιαστικά του σχόλια στην συγγραφή του κειμένου και το Νίκο Μαμάση για τις εύστοχες παρατηρήσεις του σε θέματα στατιστικής.

Αλέξανδρος Μανέτας
Φεβρουάριος 1994

Abstract

The purpose of this report is the disaggregation of hydrologic timeseries using stochastic procedures. The method presented is completely new. It's main advantage with respect to the already existing is its simplicity. The later does not affect the precession of the results, as we can see later on.

In the first chapter we explain in general what is disaggregation, what is its use and which are the differences between our model and the others already known. In the second chapter we present a full mathematical description of the model and a simple example. In the third one we refer to matters relative to the computer i.e. random numbers generation and the time needed to disaggregate for various data series. In the forth chapter we explain in brief the way the program runs and in the fifth one we present the results of an application regarding Morno's basin. Following is an appendix with the source code in case someone wants to use any of the subroutines written.

This report was supervised tirelessly by Demetri Koutsoyiannis, whose help proved indispensable. I also want to thank George Tsakalias for his useful comments on writing this report and Nikos Mamasis for his observations in statistic matters.

Alex Manetas February 1994

ΚΕΦΑΛΑΙΟ 1

ΕΙΣΑΓΩΓΗ

1.1 Σκοπός στοχαστικής υδρολογίας

Το βασικό αντικείμενο της στοχαστικής υδρολογίας είναι η συνθετική προσομοίωση. Σκοπός της συνθετικής προσομοίωσης είναι η παραγωγή συνθετικών χρονοσειρών υδρολογικών μεταβλητών, οποιουδήποτε μήκους, οι οποίες είναι το ίδιο πιθανές να πραγματοποιηθούν όσο και οι πραγματικές. Οι συνθετικές χρονοσειρές μπορούν να χρησιμοποιηθούν για την ορθολογική διαχείριση συστημάτων υδατικών πόρων γιατί μας βοηθούν να μελετήσουμε τη συμπεριφορά ενός συστήματος σε ακραίες πιθανοτικές καταστάσεις., πράγμα που είναι αδύνατον να γίνει από τα ιστορικά δεδομένα επειδή αυτά, κατά κανόνα, είναι διαθέσιμα για μικρό χρονικό διάστημα. Είναι σημαντικό να τονίσουμε ότι οι στατιστικές πληροφορίες, με τις οποίες τροφοδοτούμε τα στοχαστικά μοντέλα, προέρχονται από τα ιστορικά δείγματα. Έτσι η αξιοπιστία των αποτελεσμάτων είναι ανάλογη αυτής των ιστορικών δεδομένων.

1.2 Γενική περιγραφή του μοντέλου επιμερισμού

Στα περισσότερα προβλήματα που αντιμετωπίζουμε δεν αρκεί η προσομοίωση μίας μόνο μεταβλητής αλλά πολλών. Πολυμεταβλητότητα στην προσομοίωση μπορούμε να έχουμε χρησιμοποιώντας μεταβλητές ίδιου μεγέθους σε διαφορετικές θέσεις (π.χ. βροχή σε πολλούς σταθμούς) ή διαφορετικά μεγέθη στοχαστικά εξαρτημένα (π.χ. βροχή - απορροή) ή και τα δυο. Εργαλεία που μας βοηθούν σημαντικά σ' αυτή την προσομοίωση είναι τα πολυμεταβλητά μοντέλα επιμερισμού τα οποία είναι μια κατηγορία μοντέλων προσομοίωσης που χρησιμοποιούνται για την πύκνωση μιας χρονοσειράς, δηλαδή τον μετασχηματισμό της διακριτότητας, από μιά αραιή χρονική κλίμακα σε μια πυκνότερη (π.χ. επιμερισμός ετήσιων απορροών σε μηνιαίες).

Για να κατανοήσουμε τη λειτουργία και τις θεμελιώδεις αρχές των μοντέλων επιμερισμού, πρέπει πρώτα να ορίσουμε κάποιες βασικές έννοιες στις οποίες θα αναφερόμαστε στα επόμενα κεφάλαια. Με τον όρο *μεταβλητές υψηλότερου επιπέδου* εννοούμε τις μεταβλητές ετήσιου χρονικού βήματος. Επιλέχθηκε το ετήσιο βήμα γιατί σ' αυτό εξαφανίζονται οι ενδοετήσιες περιοδικότητες των χρονοσειρών και έτσι αυτές

παρουσιάζουν στάσιμο χαρακτήρα (βλέπε και παράγραφο 2.1). Με τον όρο *μεταβλητές χαμηλότερου επιπέδου* εννοούμε τις μηνιαίες, εβδομαδιαίες ή εποχιακές μεταβλητές. Επίσης με τον όρο περίοδο και υποπερίοδο εννοούμε μια χρονική στιγμή μέτρησης των μεταβλητών υψηλότερου και χαμηλότερου επιπέδου αντίστοιχα. Το μοντέλο επιμερισμού που χρησιμοποιούμε στο πρόγραμμα SIDIS διαφέρει αρκετά από το πλέον διαδεδομένο μοντέλο που εισηγήθηκαν οι Valencia-Shaake (1972, 1973) καθώς και τα μεταγενέστερα μοντέλα του ίδιου τύπου. Στο SIDIS ο επιμερισμός αποτελεί μια διορθωτική διαδικασία που εκτελείται μετά από την παραγωγή των μεταβλητών υψηλότερου και χαμηλότερου επιπέδου. Με τον τρόπο αυτό ο επιμερισμός είναι πολύ απλούστερος και οι απαιτήσεις του προγράμματος σε αριθμό παραμέτρων ελαχιστοποιούνται. Συγκριτικά με τη μέθοδο των Valencia-Shaake για ένα πρόβλημα που περιέχει 6 σταθμούς και 12 υποπεριόδους (μήνες) ο μεγαλύτερος πίνακας που χρησιμοποιείται έχει μέγεθος 72X72 δηλαδή αποτελείται από 5184 στοιχεία. Το ίδιο πρόβλημα στο SIDIS απαιτεί 12 πίνακες μεγέθους 6X6 δηλαδή μόνο 432 στοιχεία. Εκτός από το γεγονός ότι η μέθοδος είναι φειδωλή σε παραμέτρους (parsimony of parameters) παρουσιάζει και αριθμητικά πλεονεκτήματα. Κατά τη διάρκεια του καθορισμού των παραμέτρων του μοντέλου, όπως θα δούμε παρακάτω, πρέπει να αντιστραφούν και να διασπαστούν κάποια μητρώα. Οι διαδικασίες αυτές οδηγούν σε αναπόφευκτα αριθμητικά σφάλματα τα οποία ελλατώνονται με τη μείωση του μεγέθους των πινάκων. Για παράδειγμα τα αναμενόμενα σφάλματα στην αντιστροφή ενός πίνακα 72X72 είναι πολύ μεγαλύτερα απ' αυτά ενός 6X6.

1.3 Το μοντέλο SIDIS

Στη εργασία αυτή αναπτύχθηκε λογισμικό και έγινε η αντίστοιχη θεωρητική έρευνα σχετικά με τα πολυμεταβλητά στοχαστικά μοντέλα. Το λογισμικό είναι διαθέσιμο σε εκτελέσιμη μορφή (standalone program) και λειτουργεί εντελώς αυτόνομα, χωρίς να απαιτεί υποστήριξη από άλλα εμπορικά πακέτα. Το πρόγραμμα έχει ονομαστεί SIDIS (SImple DISaggregation) υπονοώντας τον απλό αλλά πολύ αποτελεσματικό τρόπο λειτουργίας του επιμερισμού. Η μαθηματική μεθοδολογία ανάπτυξης των προγραμμάτων δεν ακολουθεί κανένα από τα μοντέλα της βιβλιογραφίας, αλλά ένα πολύ απλούστερο τρόπο προσέγγισης του ίδιου προβλήματος. Στο SIDIS η παραγωγή μεταβλητών χαμηλού επιπέδου λειτουργεί ανεξάρτητα από την παραγωγή των αντίστοιχων υψηλού. Οι παραπάνω συνθετικές παραγωγές γίνονται με τη χρήση ενός σειριακού πολυμεταβλητού μοντέλου, όπως το πολυμεταβλητό AR(1). Στη συνέχεια επειδή τα ετήσια αθροίσματα δεν κλείνουν, (π.χ. το άθροισμα των μηνιαίων τιμών δεν είναι ίσο με την ετήσια τιμή), εφαρμόζεται μια τεχνική διόρθωσης που αποσκοπεί στην εξίσωσή τους. Η τεχνική αυτή είναι θεωρητικά τεκμηριωμένη (Koutsoyiannis, 1994).

1.4 Στατιστικές Παράμετροι

Για κάθε μεταβλητή του μοντέλου χρησιμοποιούνται και συνεπώς διατηρούνται οι παρακάτω στατιστικές παράμετροι :

- μέση τιμή
- διασπορά
- συντελεστής ασυμμετρίας

Στη συγκεκριμένη έκδοση του προγράμματος έχουμε ασχοληθεί μόνο με κανονικές κατανομές έτσι οι συντελεστές ασυμμετρίας του συνθετικού δείγματος είναι μηδέν. Μελλοντικά πρόκειται να γίνει επέκταση του μοντέλου και για ασύμμετρες κατανομές

Οι παράμετροι των από κοινού συναρτήσεων κατανομής που διατηρούνται είναι:

- συντελεστές συσχέτισης βήματος μηδέν μεταξύ διαφορετικών θέσεων
- συντελεστές συσχέτισης βήματος ένα μεταξύ μεταβλητών της ίδιας θέσης

Οι συντελεστές συσχέτισης βήματος ένα μεταξύ των διαφορετικών θέσεων δεν κρίθηκε σκόπιμο να διατηρηθούν γιατί κατά κανόνα είναι ασήμαντοι και προσθέτουν επιπλέον μεταβλητές χωρίς ουσιαστικό νόημα.

ΚΕΦΑΛΑΙΟ 2

ΜΑΘΗΜΑΤΙΚΟ ΥΠΟΒΑΘΡΟ

2.1 Γενικές έννοιες

Θεωρούμε σκόπιμο ν' αναφέρουμε μερικές βασικές έννοιες από τη θεωρία των τυχαίων χρονοσειρών για να εξοικειωθεί ο αναγνώστης με την ορολογία που θα χρησιμοποιήσουμε στη συνέχεια της εργασίας αυτής. Με τον όρο *στοχαστική ανέλιξη* ονομάζουμε μια οικογένεια τυχαίων μεταβλητών X_t όπου t είναι παράμετρος που παίρνει τιμές από ένα κατάλληλο σύνολο T (Papoulis, 1965, Taylor and Karlin, 1984). Το σύνολο αυτό στην περίπτωση της υδρολογίας περιέχει το χρόνο και μπορεί να είναι διακριτό ή συνεχές. *Χρονοσειρά* θα ονομάσουμε μια υλοποίηση της στοχαστικής ανέλιξης, δηλαδή ένα σύνολο παρατηρήσεων x_t της τυχαίας μεταβλητής X_t.

Στην υδρολογία είναι πολλές φορές αδύνατον να καταπιαστούμε με γενικούς ορισμούς των τυχαίων διαδικασιών γι' αυτό χρειάζεται να γίνουν αρχικά ορισμένες παραδοχές. Η πιο σημαντική απ' αυτές είναι η παραδοχή της *στασιμότητας*. Η στασιμότητα μας οδηγεί στο συμπέρασμα ότι οι κατανομές πρώτης τάξης είναι ανεξάρτητες του χρόνου. Δηλαδή :

$$f(x(t_1)) = f(x(t_2)) = f(x(t_3)) = \ldots = f(x(t_n)) = f(x)$$

Όπου με $f(x)$ συμβολίζουμε τη συνάρτηση πυκνότητας μιας κατανομής (Bras, Rodriguez, 1985)

Άμεση συνέπεια για τη μέση τιμή αλλά και για τη διασπορά είναι :

$$m = m(t_1) = m(t_2) = \ldots = m(t_n) = \int_{-\infty}^{\infty} x\, f(x)\, dx$$

$$\sigma = \sigma(t_1) = \ldots = \sigma(t_n) = \int_{-\infty}^{\infty} (x-m)^2 f(x)\, dx$$

Για τις συναρτήσεις δεύτερης τάξης ο όρος στασιμότητα οδηγεί στο συμπέρασμα ότι η από κοινού κατανομή δυο τυχαίων μεταβλητών $x(t_1)$, $x(t_2)$ δεν εξαρτάται από τις απόλυτες τιμές των t_1 και t_2 αλλά από τη διαφορά $\tau = t_1 - t_2$

Το επόμενο θέμα που θα μας απασχολήσει είναι κατά πόσον οι κατανομές και οι ροπές τους μπορούν να εκτιμηθούν από μια και μοναδική πραγματοποίηση μιας χρονικής ακολουθίας γεγονότων, δηλαδή μιας μοναδικής χρονοσειράς. Αυτό κατ' αρχήν δεν είναι εφικτό. Εξαίρεση αποτελεί η περίπτωση κατά την οποία η διαδικασία είναι *εργοδική* (ή τη θεωρούμε εμείς εργοδική). Επειδή ένας ευρύτερος ορισμός είναι πολύπλοκος θα δώσουμε ένα παράδειγμα για την περίπτωση της μέσης τιμής :

$$m = \int_{-\infty}^{\infty} x\,\mathrm{f}(x)dx = \lim_{T\to\infty} \frac{1}{T}\int_{0}^{T} x(t)dt$$

Από τα παραπάνω καταλαβαίνουμε ότι η εργοδικότητα είναι μια θεωρητική έννοια που δεν είναι δυνατόν να αποδειχθεί για καμμιά υδρολογική χρονοσειρά.

2.2 Πολυμεταβλητό περιοδικό μοντέλο αυτοπαλινδρόμησης ταξης 1 (Multivariate PAR(1))

Θεωρούμε την τυχαία μεταβλητή $X_{l,p}^{s}$ όπου οι δείκτες συμβολίζουν :

$s = 1,2,...,n$: υποπερίοδο (subperiod)
$l = 1,2,...,k$: θέση (location)
$p = 1,2,...,\mathrm{m}$: περίοδο (period)

Η τυχαία αυτή μεταβλητή είναι μέλος μιας *ανέλιξης χαμηλότερου επιπέδου* :

$$X_{l,1}^{1}, X_{l,1}^{2}, \ldots, X_{l,1}^{n}, X_{l,2}^{1}, X_{l,2}^{2}, \ldots X_{l,2}^{n}, \ldots, X_{l,m}^{1}, X_{l,m}^{2}, \ldots, X_{l,m}^{n}$$

Θα ορίσουμε και μια *ανέλιξη υψηλότερου επιπέδου* (αθροιστική) ως εξής :

$$S_{l,1}, S_{l,2}, \ldots, S_{l,m}$$

όπου για μια θέση l και μια περίοδο i ισχύει :

$$S_{l,i} = \sum_{t=1}^{n} X_{l,i}^{t}$$

Για να έχουμε εννιαίο συμβολισμό, ανεξάρτητο από το επίπεδο της ανέλιξης θα ορίσουμε έναν δείκτη t τέτοιο ώστε :

$$t = (p-1)n + s$$

Με βάση τα παραπάνω μπορούμε να ορίσουμε μια καινούρια μεταβλητή Z για χαμηλότερο επίπεδο :

$$Z_l^{t^t} = X_{lp}^s$$
$$Z_l^{t^{t-1}} = X_{lp}^{s-1}$$
$$Z_l^t = X_{lp}^s - \mathrm{E}\left(X_{lp}^s\right)$$

και για υψηλότερο επίπεδο :

$$Z_l^{t^t} = S_{lp}$$
$$Z_l^{t^{t-1}} = S_{lp-1}$$
$$Z_l^t = S_{lp} - \mathrm{E}\left(S_{lp}\right)$$

Παρατηρούμε ότι ο δείκτης t για την πρώτη περίπτωση εκφράζεται σε μονάδες υποπεριόδων ενώ για τη δεύτερη σε μονάδες περιόδων. Ο συμβολισμός E() παριστάνει την μέση τιμή.

Θεωρώντας το διάνυσμα που περιέχει τις μεταβλητές υψηλότερου ή χαμηλότερου επιπέδου για τις διάφορες θέσεις θα έχουμε συμβολικά :

$$\mathbf{Z}^t = \left[Z_1^t, Z_2^t, \ldots, Z_k^t\right]^{\mathrm{T}}$$

όπου ο συμβολισμός T σημαίνει ανάστροφο πίνακα.

Η βασική σχέση του μοντέλου αυτοπαλινδρόμησης AR(1) (Matalas 1967) έχει ως εξής :

$$\mathbf{Z}^{t^t} = \mathbf{A}^t \mathbf{Z}^{t^{t-1}} + \mathbf{B}^t \mathbf{V}^{t^t} \qquad (1\alpha)$$

$$\mathbf{Z}^t = \mathbf{A}^t \mathbf{Z}^{t-1} + \mathbf{B}^t \mathbf{V}^t \qquad (1\beta)$$

όπου :

$\mathbf{A}^t$, $\mathbf{B}^t$: πίνακες παραμέτρων (τόσοι όσες και οι υποπερίοδοι)

$\mathbf{V}^{t^t}$: διάνυσμα τυχαίων όρων μοναδιαίας τυπικής απόκλισης.

και : $\mathbf{V}^t = \mathbf{V}^{t^t} - \mathrm{E}\left(\mathbf{V}^{t^t}\right)$

2.1.2 Προσδιορισμός μητρώου A

Πολλαπλασιάζοντας τη σχέση (1β) από δεξιά με $\left(\mathbf{Z}^{t-1}\right)^{\mathrm{T}}$ και παίρνοντας μέσες τιμές στα δυο μέλη έχουμε :

$$\mathrm{E}\left(\mathbf{Z}^t\left(\mathbf{Z}^{t-1}\right)^{\mathrm{T}}\right) = \mathrm{E}\left(\mathbf{A}^t \mathbf{Z}^{t-1}\left(\mathbf{Z}^{t-1}\right)^{\mathrm{T}} + \mathbf{B}^t \mathbf{V}^t\left(\mathbf{Z}^{t-1}\right)^{\mathrm{T}}\right)$$
$$\mathrm{E}\left(\mathbf{Z}^t\left(\mathbf{Z}^{t-1}\right)^{\mathrm{T}}\right) = \mathbf{A}^t \mathrm{E}\left(\mathbf{Z}^{t-1}\left(\mathbf{Z}^{t-1}\right)^{\mathrm{T}}\right) + \mathbf{B}^t \mathrm{E}\left(\mathbf{V}^t\left(\mathbf{Z}^{t-1}\right)^{\mathrm{T}}\right)$$

Αν λάβουμε υπ' όψιν μας την ανεξάρτητη γέννηση του διανύσματος των τυχαίων αριθμών τότε :

$$E\left(\mathbf{V}^t\left(\mathbf{Z}^{t-1}\right)^T\right)=0$$

άρα :

$$E\left(\mathbf{Z}^t\left(\mathbf{Z}^{t-1}\right)^T\right)=\mathbf{A}^t E\left(\mathbf{Z}^{t-1}\left(\mathbf{Z}^{t-1}\right)^T\right)$$

δηλαδή :

$$\mathbf{A}^t=E\left(\mathbf{Z}^t\left(\mathbf{Z}^{t-1}\right)^T\right)E\left(\mathbf{Z}^{t-1}\left(\mathbf{Z}^{t-1}\right)^T\right)^{-1}$$

Υπενθυμίζεται ότι :

$$E\left(\mathbf{Z}^t\left(\mathbf{Z}^{t-1}\right)^T\right)=\mathrm{Cov}\left(\mathbf{Z}^t,\mathbf{Z}^{t-1}\right)$$

Ο πίνακας $\mathbf{A}^t$ που προκύπτει είναι πλήρης. Δηλαδή μας δίνει πληροφορία και για τις συσχετίσεις βήματος ένα μεταξύ των διαφορετικών θέσεων. Στο SIDIS, όπως αναφέρθηκε παραπάνω, αυτές οι συσχετίσεις δεν θα μας απασχολήσουν. Για αυτό θα θεωρήσουμε τον πίνακα $\mathbf{A}^t$ διαγώνιο με στοιχεία που προκύπτουν από τις χρονικές συσχετίσεις μόνο της ίδιας θέσης. Γράφωντας την παραπάνω σχέση πινάκων για τα διαγώνια στοιχεία έχουμε :

$$a_{ii}=\frac{\mathrm{Cov}\left(Z_i^t,Z_i^{t-1}\right)}{\mathrm{Var}\left(Z_i^{t-1}\right)}$$

Στο σημείο αυτό θα πρέπει να εξηγήσουμε την ουσιαστική διαφορά μεταξύ του ετήσιου μοντέλου αυτοπαλινδρόμησης και του εποχιακού. Έχοντας υπ' όψιν τη στασιμότητα που παρουσιάζουν οι ετήσιες χρονοσειρές αρκεί να προσδιοριστεί ένας πίνακας $\mathbf{A}^t$ για να περιγραφεί η συσχέτιση των ιστορικών δεδομένων. Στην περιοδική όμως περίπτωση όπου τα στατιστικά χαρακτηριστικά των εποχιακών χρονοσειρών δεν είναι όμοια, απαιτείται ο προσδιορισμός τόσων πινάκων όσες είναι και οι υποπερίοδοι. Για παράδειγμα αν επιλέξουμε ως χαμηλότερο βήμα το μηνιαίο πρέπει να υπολογίσουμε 12 πινακες $\mathbf{A}^t$.

2.1.1 Προσδιορισμός μητρώου B

Αν στη σχέση (1β) πολλαπλασιάσουμε από τα δεξια με $\left(\mathbf{Z}^t\right)^T$ και πάρουμε τους μέσους όρους στα δυο μέλη της εξίσωσης έχουμε :

$$E\left(\mathbf{Z}^t\left(\mathbf{Z}^t\right)^T\right)=E\left(\mathbf{A}^t\mathbf{Z}^{t-1}\left(\mathbf{Z}^t\right)^T+\mathbf{B}^t\mathbf{V}^t\left(\mathbf{Z}^t\right)^T\right)$$

$$E\left(\mathbf{Z}^t\left(\mathbf{Z}^t\right)^T\right)=\mathbf{A}^tE\left(\mathbf{Z}^{t-1}\left(\mathbf{A}^t\mathbf{Z}^{t-1}+\mathbf{B}^t\mathbf{V}^t\right)^T\right)+\mathbf{B}^tE\left(\mathbf{V}^t\left(\mathbf{A}^t\mathbf{Z}^{t-1}+\mathbf{B}^t\mathbf{V}^t\right)^T\right)$$

$$E\left(\mathbf{Z}^t\left(\mathbf{Z}^t\right)^T\right)=\mathbf{A}^tE\left(\mathbf{Z}^{t-1}\left(\mathbf{Z}^{t-1}\right)^T\right)\mathbf{A}^{tT}+\mathbf{B}^tE\left(\mathbf{V}^t\left(\mathbf{V}^t\right)^T\left(\mathbf{B}^t\right)^T+\mathbf{V}^t\left(\mathbf{Z}^{t-1}\right)^T\mathbf{A}^t\right)$$

Επειδή το διάνυσμα των τυχαίων όρων παράγεται με διασπορά ίση με μονάδα, χωρίς έτσι να βλάψουμε τη γενικότητα λόγω ύπαρξης του διανύσματος διασπορών $\mathbf{B}^t$, έχουμε :

$$E\left(\mathbf{Z}^t\left(\mathbf{Z}^t\right)^T\right)=\mathbf{A}^tE\left(\mathbf{Z}^{t-1}\left(\mathbf{Z}^{t-1}\right)^T\right)\mathbf{A}^{tT}+\mathbf{B}^t\mathbf{B}^{tT}$$

δηλαδή :

$$\mathbf{B}^t\mathbf{B}^{tT}=E\left(\mathbf{Z}^t\left(\mathbf{Z}^t\right)^T\right)-\mathbf{A}^tE\left(\mathbf{Z}^{t-1}\left(\mathbf{Z}^{t-1}\right)^T\right)\mathbf{A}^{tT}$$

Άρα έχουμε προσδιορίσει μια δευτεροβάθμια (γκραμμιανή) μορφή του μητρώου $\mathbf{B}^t$. Σε επόμενο κεφάλαιο θ' αναπτύξουμε την αντίστοιχη θεωρία βάση της οποίας από τη δευτεροβάθμια μορφή μπορεί να προκύψει ο πραγματικός πίνακας. Και σ' αυτή την περίπτωση για το ετήσιο μοντέλο απαιτείται ένας πίνακας, ενώ για το περιοδικό τόσοι όσες οι περίοδοι.

2.1.3 Προσδιορισμός στατιστικών χαρακτηριστικών τυχαίου διανύσματος.

Τα στατιστικά χαρακτηριστικά του διανύσματος $\mathbf{V}^t$ προκύπτουν ως εξής :

$$E\left(\mathbf{Z}'^t\right)=\mathbf{A}^tE\left(\mathbf{Z}'^{t-1}\right)+\mathbf{B}^tE\left(\mathbf{V}'^t\right)$$

$$\mathbf{B}^tE\left(\mathbf{V}'^t\right)=E\left(\mathbf{Z}'^t\right)-\mathbf{A}^tE\left(\mathbf{Z}'^{t-1}\right)$$

άρα :

$$E\left(\mathbf{V}'^t\right)=\mathbf{B}^{t-1}\left(E\left(\mathbf{Z}'^t\right)-\mathbf{A}^tE\left(\mathbf{Z}'^{t-1}\right)\right)$$

καί εξ ορισμού :

$$\mathrm{Var}\left(\mathbf{V}'^t\right)=\mathbf{I}$$

2.4 ΔΙΑΣΠΑΣΗ ΠΙΝΑΚΑ $\mathbf{BB}^T$

Αν συμβολίσουμε με **C** το εξαγόμενο της εξίσωσης (7) (παραλείπωντας τον άνω δείκτη s) τότε έχουμε:

$$\mathbf{C}=\mathbf{BB}^T$$

Το πρόβλημα είναι να βρούμε έναν πίνακα **B** που να ικανοποιεί την παραπάνω εξίσωση. Στο πρόβλημα αυτό υπάρχουν άπειρες λύσεις. Θα αναπτύξουμε τις δυο επικρατέστερες.

2.4.1 Διάσπαση σε κάτω τριγωνικό πίνακα

Χρησιμοποιώντας ως παράδειγμα έναν πίνακα 3X3 έχουμε :

$$\mathbf{B} = \begin{bmatrix} b_{11} & 0 & 0 \\ b_{21} & b_{22} & 0 \\ b_{31} & b_{32} & b_{33} \end{bmatrix}$$

οπότε :

$$\mathbf{B}^{\mathrm{T}} = \begin{bmatrix} b_{11} & b_{21} & b_{31} \\ 0 & b_{22} & b_{32} \\ 0 & 0 & b_{33} \end{bmatrix}$$

κάνοντας τις πράξεις :

$$\mathbf{BB}^{\mathrm{T}} = \begin{bmatrix} b^2{}_{11} & b_{11}b_{21} & b_{11}b_{31} \\ b_{21}b_{11} & b_{21}^2 + b^2{}_{22} & b_{21}b_{31} + b_{22}b_{32} \\ b_{11}b_{31} & b_{21}b_{31} + b_{22}b_{32} & b_{31}^2 + b_{32}^2 + b_{33}^2 \end{bmatrix}$$

Η λύση του παραπάνω συστήματος είναι :

$$b_{11} = \sqrt{d_{11}} \qquad b_{21} = \frac{d_{21}}{b_{11}} \qquad b_{31} = \frac{d_{31}}{b_{11}}$$

$$b_{22} = \sqrt{d_{22} - b_{21}^2} \quad b_{33} = \sqrt{d_{33} - b_{31}^2 - b_{32}^2}$$

$$b_{32} = \frac{d_{32} - b_{21}b_{31}}{b_{22}}$$

Η παραπάνω λύση γενικεύεται και για μητρώα διαστάσεως NXN. Έτσι :

$$b_{ij} = 0 \qquad (i\langle j)$$

$$b^t{}_{ij} = \sqrt{c^t{}_{ij} - \sum_{k=1}^{j-1}\left(b^t{}_{jk}\right)^2} \qquad (i = j)$$

$$b^t{}_{ij} = \frac{c^t{}_{ij} - \sum_{k=1}^{j-1} b^t{}_{jk} b^t{}_{ik}}{b^t{}_{ij}} \qquad (i\rangle j)$$

Παρατηρούμε ότι για να καθοριστούν τα στοιχεία b_{ij} χρειάζεται το υπόριζο της παραπάνω ποσότητας να είναι θετικό ή μηδέν. Αυτό προυποθέτει ένα θετικά ορισμένο πίνακα $\mathbf{BB}^{\mathrm{T}}$ πράγμα που δεν συμβαίνει πάντα είτε λόγω αριθμητικών σφαλμάτων είτε εξ αιτίας των ιστορικών δεδομένων. Σε αυτές τις περιπτώσεις ακολουθείται μια διαδικασία διόρθωσης του μητρώου η οποία αποβλέπει στην ενίσχυση των στοιχείων της κύριας

διαγωνίου, πράγμα που επηρεάζει τους συντελεστές συσχέτισης των χρονοσειρών στο χώρο (Koutsoyiannis, 1994).

2.4.2 Μέθοδος Ιδιοτιμών

Ορίζουμε έναν πίνακα **P** ο οποίος μορφώνεται από τα ιδιοδιανύσματα του πίνακα **C**. Ορίζουμε επίσης ένα διαγώνιο πίνακα **E** του οποίου τα στοιχεία είναι οι ιδιοτιμές του **C**. Επειδή ο πίνακας **P** είναι ορθογώνιος, γιατί είναι πίνακας ιδιοδιανυσμάτων, ισχύει :

$$\mathbf{P}\mathbf{P}^T = \mathbf{P}^T\mathbf{P} = \mathbf{I}$$

Αν χρησιμοποιήσουμε τη θεμελιώδη σχέση ιδιοτιμών και ιδιοδιανυσμάτων όπου :

$$\mathbf{C}\mathbf{P} = \mathbf{P}\mathbf{E}$$

τότε :

$$\mathbf{C} = \mathbf{P}\mathbf{E}\mathbf{P}^{-1}$$

$$\mathbf{C} = \mathbf{P}\mathbf{E}\mathbf{P}^T = \mathbf{P}\sqrt{\mathbf{E}}\sqrt{\mathbf{E}}\mathbf{P}^T = \mathbf{P}\sqrt{\mathbf{E}}\left(\mathbf{P}\sqrt{\mathbf{E}}\right)^T$$

$$\mathbf{B} = \mathbf{P}\sqrt{\mathbf{E}}$$

Παρατηρούμε ότι και αυτή η μέθοδος έχει κάποιο περιορισμό. Δεν πρέπει να υπάρχει αρνητική ιδιοτιμή γιατί αλλιώς ο πίνακας **B** περιέχει και μιγαδικά στοιχεία. Μια πρακτική αντιμετώπιση του παραπάνω προβλήματος είναι να μηδενίσουμε την αρνητική ιδιοτιμή, χωρίς να αλλάξουμε το αντίστοιχο ιδιοδιάνυσμα.

Οι ιδιοτιμές και τα ιδιοδιανύσματα του πίνακα βρίσκονται με τη μέθοδο Jacobi (Press, Flannery, Teukolsky, Vetterling, 1988)

2.5 Διαδικασία γέννησης δεδομένων

Για τη γέννηση συνθετικών δεδομένων χρησιμοποιούμε τη σχέση (1α) η οποία είναι αναδρομική. Έτσι για να παραχθούν δεδομένα ενός έτους χρειάζονται ως προηγούμενη πληροφορία οι τιμές του προηγούμενου έτους. Στο πρώτο βήμα αυτή η πληροφορία δεν υφίσταται και στο ετήσιο μοντέλο θεωρούμε τις προηγούμενες τιμές ίσες με τις μέσες τιμές των χρονοσειρών. Αντίστοιχα στο περιοδικό λαμβάνουμε το μέσο όρο της προηγούμενης περιόδου, από αυτή που έχουμε ορίσει σαν αρχική, από τα ιστορικά δεδομένα.

2.6 Παράδειγμα εφαρμογής (παραγωγή ετήσιων μεταβλητών)

Τα ιστορικά δεδομένα που περιέχονται στον πίνακα 1 είναι βροχές στο Λιδωρίκι, επιφανειακές βροχές στο Μόρνο και απορροή στο Μόρνο. Στον πίνακα 2 παρουσιάζονται τα στατιστικά χαρακτηριστικά των δειγμάτων.

Λιδωρίκι βροχή (mm)	**Μόρνος βροχή (mm)**	**Μόρνος Απορροή (m^3/sec)**
872	1360	450
765	1451	577
997	1772	785
806	1458	689
837	1439	651
1009	1691	627
1209	1982	783
1029	1721	604
749	1187	390
950	1546	590
681	1234	498
980	1861	576
884	1587	459
821	1290	380
830	1332	390
636	1043	151
1113	1469	542
555	799	168

Πίνακας 1

	Μέση τιμή	Διασπορά	Ασυμμετρία
Λιδωρίκι	873	28102	0.065
βροχή Μόρνου	1456	86338	-0.3
Απορροή Μόρνου	517	31636	-0.59

Πίνακας 2

Ο διαγώνιος πίνακας A ως γνωστόν προκύπτει από το πηλίκο της συνδιασποράς της χρονοσειράς κάθε θέσης με τον εαυτό της μετατοπισμένο κατά μια θέση, με τη διασπορά και είναι ίσο με :

$$\mathbf{A} = \begin{bmatrix} -0.233 & 0 & 0 \\ 0 & 0.097 & 0 \\ 0 & 0 & 0.325 \end{bmatrix}$$

Ο πίνακας S των συνδιασπορών για τον προσδιορισμό του B είναι ίσος με :

$$\mathbf{S} = \begin{bmatrix} 26515 & 40877 & 20838 \\ 40877 & 80059 & 41429 \\ 20838 & 41429 & 29884 \end{bmatrix}$$

Ο πίνακας $BB^T = S\text{-}ASA^T$ είναι ίσος με :

$$\mathbf{BB}^T = \begin{bmatrix} 25678 & 41802 & 22414 \\ 41802 & 79303 & 40121 \\ 22414 & 40121 & 26728 \end{bmatrix}$$

Μετά την αποσύνθεσή του (με τη μέθοδο των ιδιοτιμών) προκύπτει ο πίνακας B :

$$\mathbf{B} = \begin{bmatrix} 42.95 & 152.39 & -3.139 \\ -19.278 & 279.073 & -32.39 \\ -7.745 & 150.57 & 63.22 \end{bmatrix}$$

Στη συνέχεια προσδιορίζουμε τους μέσους όρους με τους οποίους θα παραχθεί το τυχαίο διάνυσμα V με τη βοήθεια του τύπου $\mathrm{E}(\mathbf{V}^t) = \mathbf{B}^{-1}\left(\mathrm{E}(\mathbf{X}^t) - \mathbf{A}\mathrm{E}(\mathbf{X}^{t-1})\right)$. Το διάνυσμα έχει ως εξής :

$$\mathrm{E}(\mathbf{V}) = \begin{bmatrix} 8.0188 \\ 4.712 \\ -4.713 \end{bmatrix}$$

Τέλος αφού εκτιμήθηκαν όλες οι παράμετροι του μοντέλου αρχίζει η παραγωγή συνθετικών δεδομένων. Η αρχική τιμή των μεταβλητών για κάθε θέση είναι ίση με τη μέση τιμή των χρονοσειρών. Έτσι για το τυχαίο διάνυσμα V :

$$\mathbf{V} = \begin{bmatrix} 6.45 \\ 5.88 \\ -2.57 \end{bmatrix}$$

Προκύπτει η πρώτη σειρά συνθετικών δεδομένων :

$$\mathbf{X}^t = \begin{bmatrix} 1181.77 \\ 1600.02 \\ 673.22 \end{bmatrix}$$

Στη συνέχεια τα παραπάνω δεδομένα χρησιμοποιούνται ως προηγούμενη πληροφορία για τη γέννηση μίας ακόμη σειράς και η διαδικασία αυτή επαναλαμβάνεται μέχρις ότου να συμπληρωθεί ο αριθμός των δεδομένων που θέλουμε.

2.7 Διαδικασία επιμερισμού

2.7.1 Μαθηματικό μοντέλο

Θεωρούμε ότι με κάποιο τρόπο είναι γνωστές οι μεταβλητές υψηλότερου επιπέδου (ετήσιες). Αυτές μπορεί να είναι ή ιστορικά δεδομένα ή συνθετικά τα οποία παρήχθησαν με ένα ετήσιο μοντέλο προσομοίωσης (στην περίπτωσή μας AR(1)). Επίσης, θεωρούμε ότι έχουν παραχθεί, τελείως ανεξάρτητα, συνθετικά δεδομένα χαμηλότερου επιπέδου (π.χ. μηνιαία) με τη χρήση AR(1). Είναι προφανές ότι αν συγκρίνουμε τις ετήσιες μεταβλητές με τα αθροίσματα των μεταβλητών χαμηλότερου επιπέδου για κάθε έτος θα υπάρχει κάποια διαφορά. Το πρόβλημα είναι να βρεθεί ένας τρόπος εξίσωσης των δυο αυτών ποσοτήτων με τους παρακάτω περιοριστικούς όρους :

1) Να μην αλλάξουν οι μέσες τιμές των περιόδων
2) Να μήν αλλάξουν οι διασπορές
3) Να μην αλλάξουν οι αυτοσυσχετίσεις (συσχετίσεις μεταξύ των περιόδων κάθε θέσης)
4) Να μεταβληθούν όσο το δυνατόν λιγότερο οι ετεροσυσχετίσεις (συσχετίσεις μεταξύ διαφορετικών θέσεων)

Έχει δειχθεί (Koutsoyiannis, 1994) ότι αυτό μπορεί να γίνει χρησιμοποιώντας την εξής διορθωτική σχέση :

$$X_{lp}^{s^*} = X_{lp}^{s} + \delta_l^s \mathrm{D}S_{lp}$$

όπου $\mathrm{D}S_{lp}$ είναι η διαφορά μεταξύ του αθροίσματος των τιμών των περιόδων ενός έτους p στη θέση l με την ετήσια τιμή που προέκυψε από το ετήσιο μοντέλο γέννησης μεταβλητών. Δηλαδή :

$$\mathrm{D}S_{lp} = \sum_{j=1}^{n} X_{lp}^{j} - S_{lp}$$

και :

$$\delta_l^s = \frac{\mathrm{Cov}\left(X_{lp}^s, S_{lp}\right)}{\mathrm{Var}\left(S_{lp}\right)}$$

Όπως παρατηρούμε από τα παραπάνω η σχέση επιδιώκει να μοιράσει την ετήσια διαφορά στις αντίστοιχες περιόδους ανάλογα με το βαθμό που αυτές συσχετίζονται με

τις ετήσιες χρονοσειρές. Η επιρροή του μετασχηματισμού αυτού στο μέσο όρο των περιοδικών χρονοσειρών έχει ως εξής :

$$\mathrm{E}\left(X_{lp}^{s^*}\right) = \mathrm{E}\left(X_{lp}^{s}\right) + \delta_l^s \mathrm{E}\left(\mathrm{D}S_{lp}\right)$$
$$\mathrm{E}\left(\mathrm{D}S_{lp}\right) = 0$$

Παρατηρούμε δηλαδή ότι δεν επηρεάζει τους μέσους όρους, κάτι που και αρχικά είχαμε θέσει ως όρο. Αντίστοιχα έχει αποδειχθεί (Koutsoyiannis, 1994) ότι με το μετασχηματισμό αυτό δεν μεταβάλλονται ούτε οι αυτοσυσχετίσεις μεταξύ των περιόδων μιας θέσης.

2.7.2 Επαναληπτική διαδικασία διόρθωσης

Στους περιοριστικούς όρους που αναφέραμε στην αρχή του κεφαλαίου ζητούσαμε η επιρροή της διορθωτικής διαδικασίας στους συντελεστές ετεροσυσχέτισης να είναι ελάχιστη. Προκειμένου να γίνει αυτό ακολουθούμε την παρακάτω επαναληπτική διαδικασία.

Παράγουμε πρώτα λ εναλακτικές σειρές που αντιστοιχούν σε μια περίοδο. Απ' αυτές επιλέγουμε την πιο κατάλληλη με το εξής κριτήριο :

$$\sum_{j=1}^{k}\left(S_{pj} - \sum_{t=1}^{n} X_{pj}^{t}\right)^2 = \min$$

Αφού γίνει η επιλογή προχωράμε στην επόμενη περίοδο. Με τον τρόπο αυτό, όπως θα δούμε, πετυχαίνουμε πολύ καλή προσέγγιση μεταξύ ιστορικών και συνθετικών ετεροσυσχετίσεων. Σημειώνεται πάντως ότι ακόμα και χωρίς την επαναληπτική διαδικασία τα αποτελέσματα είναι ικανοποιητικά.

ΚΕΦΑΛΑΙΟ 3
ΥΠΟΛΟΓΙΣΤΙΚΟ ΠΕΡΙΒΑΛΛΟΝ

3.1 Περιβάλλον ανάπτυξης εφαρμογής

Για να λειτουργήσει το SIDIS απαιτείται προσωπικός υπολογιστής IBM ή συμβατός και περιβάλλον MSWindows (έκδοση 3.0 ή μεταγενέστερη). Ο πηγαίος κώδικας είναι γραμμένος σε δυο γλώσσες προγραμματισμού. Τη Visual Basic (VB) και τη C++. Η VB είναι ένα πολύ καλό εργαλείο για την ανάπτυξη λογισμικού επικοινωνίας με το χρήστη (User interface). Περιέχει μια εργαλειοθήκη (toolbox) η οποία περιλαμβάνει όλα τα αντικείμενα που συνθέτουν ένα πρόγραμμα που τρέχει κάτω απο MSWindows, όπως κουμπιά (button), πλέγμα (grid), μπάρες κύλισης (scroll bar), πεδία επιλογών (list box) και πολλά άλλα. Ο προγραμματιστής συνδυάζει όλα αυτά τα αντικείμενα (object) μέσα σε ένα παράθυρο και η VB αυτομάτως αντιστοιχεί σε καθένα από αυτά μια διαδικασία. Στο επόμενο βήμα, ο προγραμματιστής, γράφει μέσα σ' αυτές τις διαδικασίες τις κατάλληλες εντολές που θα εκτελούνται κάθε φορά που ο χρήστης επιδρά σε κάποιο από τα αντικείμενα. Η VB φροντίζει από μόνη της να μετατρέψει τις ενέργειες του χρήστη σε μηνύματα που πυροδοτούν τις διαδικασίες. Το μόνο μειονέκτημα της VB είναι ότι εκτελεί αργά τις εντολές που περιλαμβάνουν μαθηματικούς υπολογισμούς. Το πρόβλημα αυτό λύθηκε με τη βοήθεια της C. Οι ρουτίνες των μαθηματικών γράφτηκαν σε C και μεταγλωτίστηκαν σε μορφή δυναμικής βιβλιοθήκης (DLL). Οι δυναμικές βιβλιοθήκες είναι μια καινοτομία των MSWindows. Είναι σύνολα υπορουτίνων που συνδέονται με το πρόγραμμα τη στιγμή που αυτό φορτώνεται στη μνήμη (δυναμική σύνδεση), αντί να ενσωματώνονται στον κώδικά του (στατική σύνδεση). Το βασικό πλεονέκτημα μιας τέτοιας βιβλιοθήκης είναι ότι μπορεί να εξυπηρετεί ταυτόχρονα περισσότερα του ενός προγράμματα, εξοικονομώντας μ' αυτό τον τρόπο μνήμη.

3.2 Απαιτήσεις προγράμματος

Όλες οι μεταβλητές του προγράμματος έχουν διπλή ακρίβεια (double precision) συνεπώς για κάθε μια απ' αυτές απαιτούνται 8 bytes για να αποθηκευτεί. Σε μια τυπική εφαρμογή με 3 διαφορετικούς σταθμούς, 12 υποπεριόδους και 5000 περιόδους συνθετικών δεδομένων θα χρειαστεί να δεσμευτούν 3*12*5000*8 = 1.5 MB μνήμης RAM. Στην

περίπτωση που η μνήμη αυτή δεν είναι διαθέσιμη το πρόγραμμα καταφεύγει στο σκληρό δίσκο δημιουργώντας ένα αρχείο αντιμετάθεσης (swap file). Η δημιουργία του αρχείου αυτού μειώνει αισθητά την ταχύτητα λειτουργίας του προγράμματος. Αυτό είναι αναπόφευκτο όταν θέλουμε να παράξουμε μεγάλο αριθμό συνθετικών δεδομένων. Επειδή το πρόγραμμα σε κάθε τρέξιμο κάνει πολλές μαθηματικές πράξεις, καλό είναι ο υπολογιστής να είναι εξοπλισμένος με μαθηματικό επεξεργαστή, χωρίς πάντως να απαιτείται η ύπαρξή του για να λειτουργήσει. Πρέπει επίσης να υπάρχει αρκετός ελεύθερος χώρος στο σκληρό δίσκο επειδή τα αρχεία αποτελεσμάτων μπορούν να ξεπεράσουν και τα 5MB ανάλογα με τον αριθμό των περιόδων των συνθετικών δεδομένων.

3.3 Χρόνοι εκτέλεσης

Η δοκιμή που ακολουθεί έγινε σε έναν 486 με 16 MB μνήμης RAM και ταχύτητα 33Mhz. Στον πίνακα 3 φαίνονται τα αποτελέσματα.

Μοντέλο	**Μεταβλητές**	**Χρόνος σε sec**
Ετήσιο AR(1)	3*1000	3.45
	3*5000	14.28
Περιοδικό	3*12*500	15.03
AR(1)	3*12*1000	29.21
	3*12*2000	56.92
Περιοδικό με 10	3*12*500	123
επαναλήψεις	3*12*1000	245

Πίνακας 3

3.4 Τυχαίοι αριθμοί και υπολογιστής

Τα προγράμματα προσομοίωσης φυσικών φαινομένων βασίζονται σε πολύ μεγάλο ποσοστό στη επιτυχημένη γέννηση τυχαίων αριθμών από τον υπολογιστή. Το θέμα είναι αν μπορούμε να φτιάξουμε έναν αλγόριθμο που να παράγει πραγματικά τυχαίους αριθμούς.

Η ιδανική περίπτωση θα ήταν να μπορούσαμε να έχουμε ως πήγη ένα κατάλληλα επιλεγμένο φυσικό πείραμα, πράγμα που τις περισσότερες φορές είναι αδύνατον. Πίνακες τυχαίων αριθμών που προέκυψαν από τυχαία πειράματα υπάρχουν αλλά η χρήση τους σε

ένα πρόγραμμα θα απαιτούσε πολλή μνήμη, θα καθυστερούσε σημαντικά την εκτέλεσή του και θα ήταν δύσκολο οι αριθμοί των πινάκων να προσαρμοστούν στις απαιτούμενες συνθήκες.
Αλγόριθμοι παραγωγής τυχαίων αριθμών υπάρχουν πολλοί με δημοφιλέστερο αυτόν που προτάθηκε από τον Lehmer το 1948.

3.4.1 Αλγόριθμος Lehmer

Αν επιλέξουμε έναν μεγάλο περιττό αριθμό m και έναν ακέραιο *a* μεταξύ 2 και *m*-1 τότε από την ακολουθία:

$$z_n = az_{n-1} \bmod m \quad n \geq 1$$

διαλέγοντας ένα αρχικό αριθμό z_0 διάφορο από το μηδέν έχουμε μια σειρά τυχαίων αριθμών με όρια 1 και *m*-1 και περίοδο *m*-1

Η επιλογή των παραμέτρων της παραπάνω σχέσης είναι ουσιαστικής σημασίας για την ποιότητα των παραγόμενων αριθμών. Για την παράμετρο *m* προτάθηκε από τον Lehmer η ποσότητα:

$$m = 2^{31} - 1 = 2147483647$$

Το παραπάνω νούμερο είναι αρκετά μεγάλο για τις περισσότερες εφαρμογές. Η επιλογή της παραμέτρου *a* έχει ως στόχο την όσο το δυνατόν μεγαλύτερη περίοδο της ακολουθίας των τυχαίων αριθμών. Οι S.K. Park και K.W. Miller πρότειναν τον αριθμό 16807.
Βάση των παραπάνω η αναδρομική σχέση παραγωγής τυχαίων αριθμών μπορεί να πάρει τη μορφή:

$$z_n = 16807\, z_{n-1} \bmod 2147483647$$

Αυτή τη μορφή χρησιμοποιούμε στο πρόγραμμα SIDIS.

Ο παραπάνω αλγόριθμος παράγει τυχαίους αριθμούς, ομοιόμορφης κατανομής, στο διάστημα [1 m-1]. Αν θέλουμε να παράγουμε αριθμούς στο [0 1) πρέπει να κάνουμε τον παρακάτω μετασχηματισμο :

$$u_i = \frac{z_i}{m}$$

3.4.2 Δοκιμή τυχαίου δείγματος

Μια εμπειρική δοκιμή μπορεί να γίνει αν παράξουμε ένα αρκετά μεγάλο αριθμό μεταβλητών, έστω N το πλήθος, και δούμε πως αυτές κατανέμονται σε διάφορα κατάλληλα επιλεγμένα διαστήματα. Για παράδειγμα η ακολουθία u_i παίρνει τιμές από το διάστημα [0 1). Σε ένα υποδιάστημα αυτού, το [0.1 0.2), περιμένουμε να ανήκουν N/10 νούμερα.

Η παραπάνω δοκιμή είναι ίσως η απλούστερη που μπορεί να γίνει. Υπάρχουν βεβαίως και άλλοι τρόποι για να ελεγχθεί μια γεννήτρια οι οποίοι όμως, είναι αρκετά πιο περίπλοκοι. (Papoulis, 1990).

3.4.3 Παραγωγή τυχαίων αριθμών κανονικής κατανομής

Οι τυχαίοι αριθμοί ομοιόμορφης κατανομής που παράγονται με τον παραπάνω τρόπο πρέπει να μετασχηματιστούν με κάποιο κατάλληλο αλγόριθμο σε αριθμούς κανονικής κατανομής. Οι Box και Muller (1958) πρότειναν την παρακάτω μέθοδο.

Αν u_i και u_{i+1} είναι δυο αριθμοί ομοιόμορφης κατανομής τέτοιοι ώστε $0 \le u_i < 1.0$ τότε με το μετασχηματισμό :

$$n_i = \sqrt{-2\ln(u_i)}\cos(2\pi u_{i+1})$$

$$n_{i+1} = \sqrt{-2\ln(u_i)}\sin(2\pi u_{i+1})$$

Παίρνουμε δυο αριθμούς τυποποιημένης κανονικής κατανομής (N.T. Kottegoda, 1980)

ΚΕΦΑΛΑΙΟ 4

ΑΝΑΠΤΥΞΗ ΠΡΟΓΡΑΜΜΑΤΩΝ ΚΑΙ ΧΡΗΣΗ ΤΟΥΣ

4.1 Ανάπτυξη προγραμμάτων

Το SIDIS γράφτηκε, κατά κύριο λόγο, για να κάνει επιμερισμό υδρολογικών χρονοσειρών. Για τη διαδικασία αυτή, όπως ήδη έχει αναφερθεί, απαιτείται η παραγωγή ετήσιων και περιοδικών συνθετικών χρονοσειρών. Οι σχετικές ρουτίνες που κάνουν αυτή την παραγωγή γράφτηκαν με τέτοιο τρόπο ώστε να εκτελούνται εντελώς αυτόνομα. Έτσι το SIDIS μπορεί να χρησιμοποιηθεί και για την παραγωγή ετήσιων ή περιοδικών χρονοσειρών, ανεξάρτητα του επιμερισμού.

Το τμήμα παραγωγής *ετήσιων χρονοσειρών* απαιτεί ένα αρχείο δεδομένων (κατασκευασμένο με συγκεκριμένο τρόπο) και βάση αυτού παράγει ένα άλλο αρχείο με τα συνθετικά δεδομένα. Δεν υπάρχει κάποιο όριο στον μέγιστο αριθμό δεδομένων που μπορούν να παραχθούν, ο μόνος ίσως περιορισμός είναι ο χρόνος εκτέλεσης και η περιοδικότητα της γεννήτριας τυχαίων αριθμών (βλ. κεφάλαιο 3).

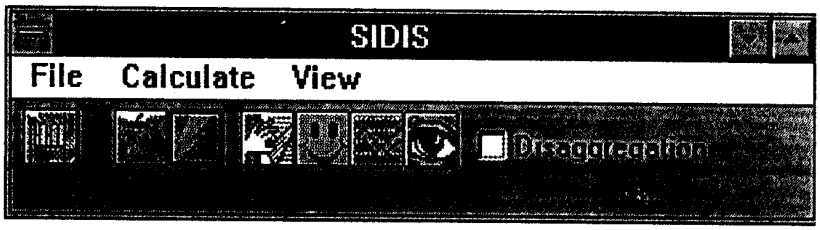
Το τμήμα παραγωγής *περιοδικών χρονοσειρών* δουλεύει με όμοιο τρόπο με μόνο περιορισμό τον αριθμό των υποπεριόδων. Αυτές δεν μπορεί να ξεπερνούν τις 12 (μηνιαίο βήμα) αλλά ούτε να είναι λιγότερες από δυο.

Ο *επιμερισμός* απαιτεί δυο αρχεία. Ένα ετήσιων δεδομένων (συνθετικό ή όχι) και ένα περιοδικών. Ο αριθμός των δεδομένων που θα παραχθούν δεν καθορίζεται από το χρήστη αλλά από το πλήθος των περιόδων του ετήσιου αρχείου.

4.2 Χρήση προγραμμάτων

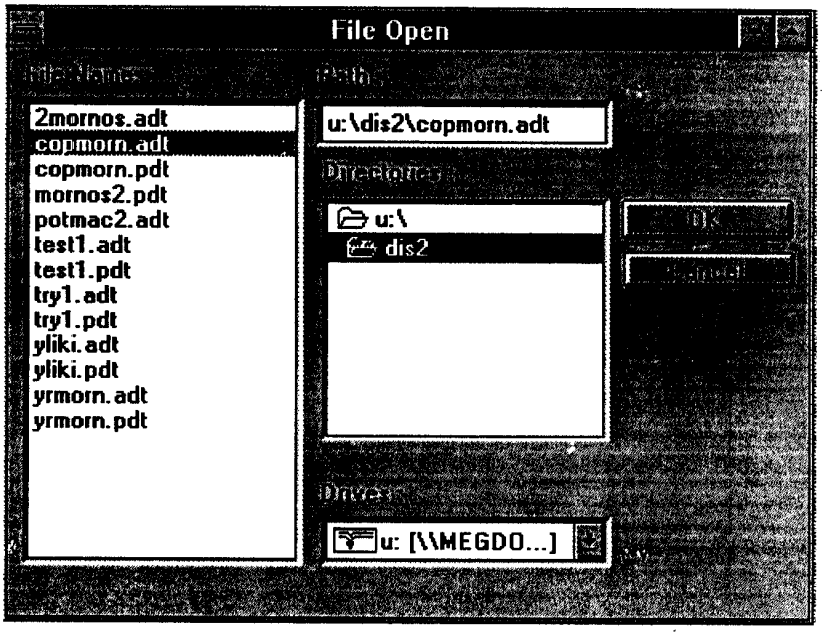
Ο καλύτερος ίσως τρόπος για να δείξουμε πως λειτουργεί κάποιο πρόγραμμα είναι με ένα παράδειγμα.

Στο παράδειγμα αυτό θα χρησιμοποιήσουμε δεδομένα από βροχές στο Λιδωρίκι, επιφανειακές βροχές στο Μόρνο και απορροή στο Μόρνο (βλ παράγραφο 2.6). Οι περίοδοι των ιστορικών δεδομένων είναι 18 (ετήσιο βήμα). Τα ετήσια δεδομένα βρίσκονται στο αρχείο copmorn.adt ενώ τα περιοδικά στο αρχείο copmorn.pdt. Θα γίνει παραγωγή 100 περιόδων ετήσιων δεδομένων.



Σχήμα 1

Το αρχικό παράθυρο της εφαρμογής απεικονίζεται στο σχήμα 1. Η πρώτη ενέργεια του χρήστη είναι να φέρει στη μνήμη τα δεδομένα του ετήσιου αρχείου. Από το μενού File με την εντολή Open, ή πατώντας το πρώτο κατά σειρά κουμπί, εμφανίζεται ένα πλαίσιο διαλόγου (βλ. σχήμα 2)

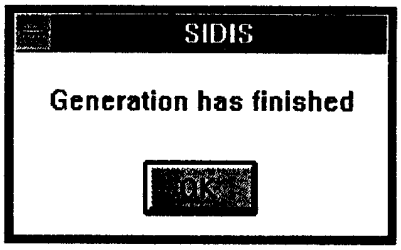


Σχήμα 2

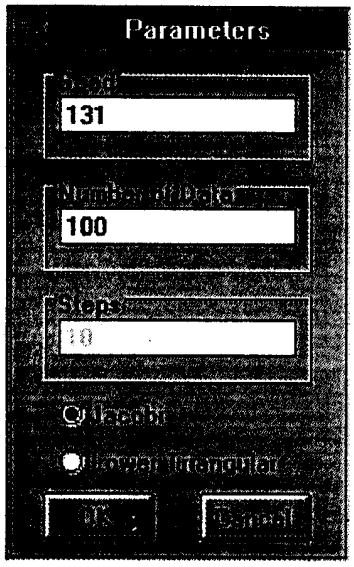
Στο παράθυρο αυτό ο χρήστης μπορεί να επιλέξει με το ποντίκι το αρχείο που τον ενδιαφέρει, από το πεδίο με τίτλο File Name, ή να πληκτρολογήσει το πλήρες όνομα του αρχείου στο πεδίο με τίτλο Path :. Από το πεδίο Directories μπορεί να πάει στον κατάλογο που επιθυμεί ενώ από το πεδίο Drives μπορεί να αλλάξει δίσκο. Με το κουμπί OK επιβεβαιώνει τις επιλογές του ενώ με το Cancel τις ακυρώνει. Στο παράδειγμα επιλέξαμε το αρχείο ετήσιων δεδομένων copmorn.adt.

Η επόμενη διαδικασία του χρήστη είναι να καθορίσει τις παραμέτρους για την παραγωγή των συνθετικών δεδομένων. Για να γίνει αυτό πρέπει να επιλέξει απο το μενού Calculate την εντολή Annual AR(1), ή να πατήσει το δεύτερο κουμπί (βλ. σχήμα 1). Μόλις πατήσει το σχετικό κουμπί ενφανίζεται το πλαίσιο του σχήματος 3. Στο πλαίσιο αυτό απο το πεδίο Seed καθορίζεται ο αριθμός που εγκαινιάζει τη γεννήτρια τυχαίων

αριθμών, από το πεδίο Number of Data καθορίζεται ο αριθμός των παραγόμενων περιόδων και από το πεδίο Steps καθορίζεται ο αριθμός των βημάτων επιμερισμού. Οι διακόπτες Jacobi και Lower Triangular καθορίζουν τη μέθοδο διάσπασης των πινάκων B. Στο παράδειγμα επιλέξαμε τον αριθμό 131 για να ξεκινήσει η γεννήτρια, επιλέξαμε 100 περιόδους συνθετικών δεδομένων και μέθοδο διάσπασης Jacobi. Μόλις πατήσουμε το κουμπί OK το πρόγραμμα αρχίζει την παραγωγή δεδομένων. Μόλις τελειώσει μας βγάζει το μήνυμα του σχήματος 4. Στο σημείο αυτό τελείωσε η παραγωγή. Τα συνθετικά δεδομένα βρίσκονται στο αρχείο copmorn.arg.

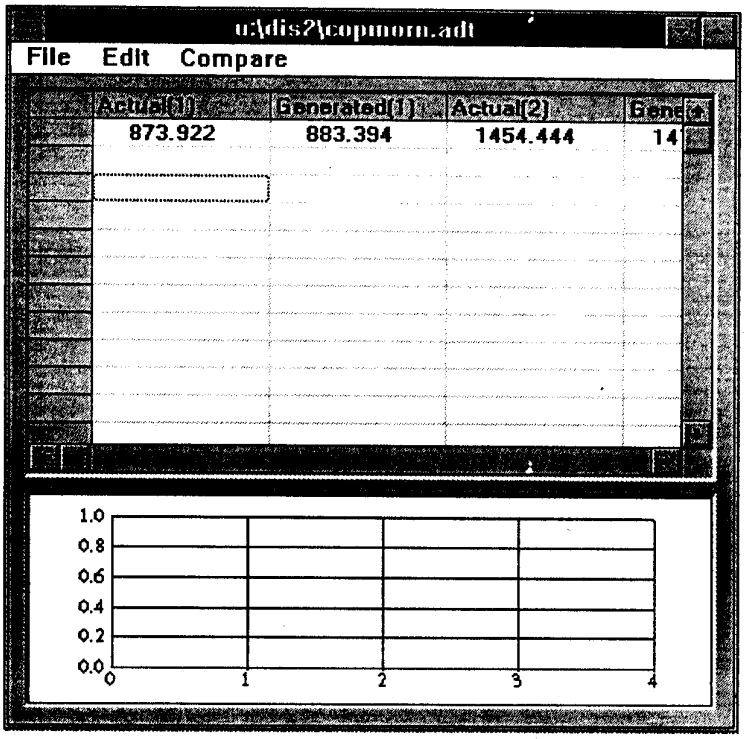


Σχήμα 4



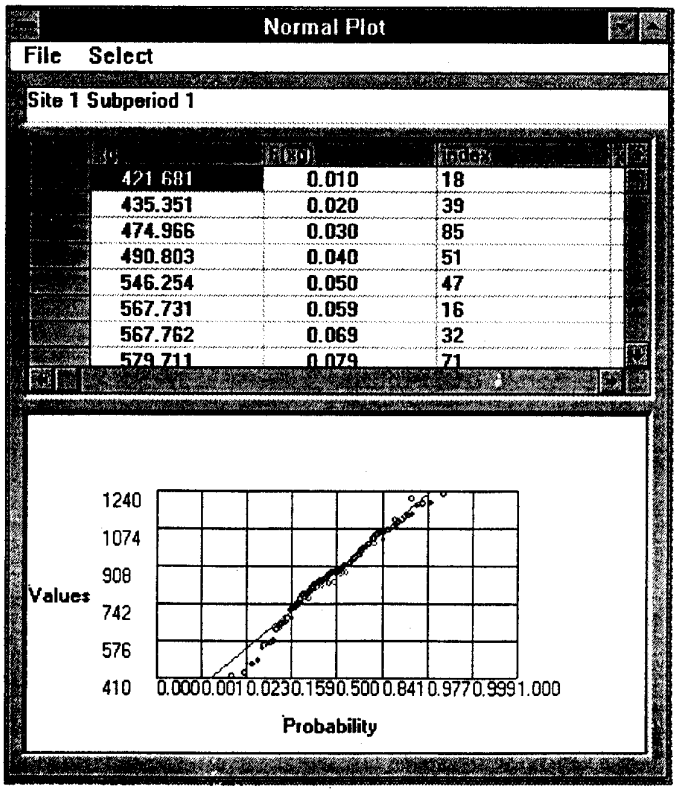
Σχήμα 3

Αφού έγινε η παραγωγή ο χρήστης μπορεί να συγκρίνει τα αποτελέσματα του με τα ιστορικά δεδομένα. Οι δυνατότητες του προγράμματος είναι δυο. Η πρώτη παρουσιάζει τα στατιστικά στοιχεία του συνθετικού και ιστορικού δείγματος για σύγκριση ενώ η δεύτερη σχεδιάζει τα συνθετικά και ιστορικά δεδομένα σε χαρτί κανονικής κατανομής. Για να γίνει η στατιστική σύγκριση πρέπει να πατηθεί το πέμπτο κατά σειρά κουμπί (βλ. σχήμα 1). Με το πάτημα του κουμπιού εμφανίζεται το παράθυρο του σχήματος 5. Από το μενού File με την εντολή Open εμφανίζεται το παράθυρο του σχήματος 2 και ο χρήστης επιλέγει το αρχείο που τον ενδιαφέρει. Στην περίπτωσή μας το copmorn.adt γιατί γι' αυτό παράξαμε δεδομένα. Τα στατιστικά δεδομένα για τα συνθετικά και ιστορικά δεδομένα εμφανίζονται με τις αντίστοιχες εντολές του μενού Compare. Στη στήλη Actual είναι τα ιστορικά δεδομένα ένω στη στήλη Generated τα συνθετικά. Στο παράδειγμα εμφανίζονται οι μέσοι όροι.



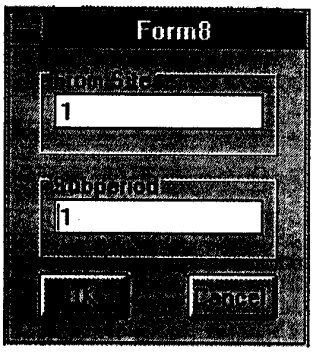
Σχήμα 5

Η επόμενη μέθοδος σύγκρισης είναι η σχεδίαση σε κανονικό χαρτί των δεδομένων. Για να γίνει αυτό πρέπει ο χρήστης να επιλέξει απο το μενού View την εντολή Normal Plot ή να πατήσει το έκτο κουμπί. Μετά απ' αυτή του τη ενέργεια εμφανίζεται το παράθυρο του σχήματος 6. Επιλέγοντας το αρχείο με τη μέθοδο που αναφέραμε πιό πάνω το



Σχήμα 6

πρόγραμμα φέρνει στη μνήμη τα αντίστοιχα στοιχεία. Απο το μενού Select με την εντολή Site-Period εμφανίζεται το παράθυρο του σχήματος 7 απ' όπου ο χρήστης επιλέγει τη θέση και την υποπερίοδο που θέλει. Στο παράδειγμα επιλέξαμε το Λιδωρίκι και την υποπερίοδο 1 (λόγω ετήσιων δεδομένων δεν υπάρχουν άλλες υποπερίοδοι).



Σχήμα 7

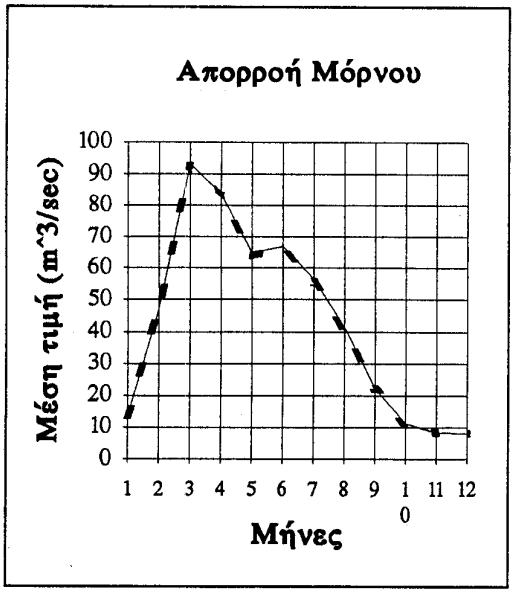
ΚΕΦΑΛΑΙΟ 5

ΧΡΗΣΗ ΠΡΟΓΡΑΜΜΑΤΟΣ-ΑΠΟΤΕΛΕΣΜΑΤΑ

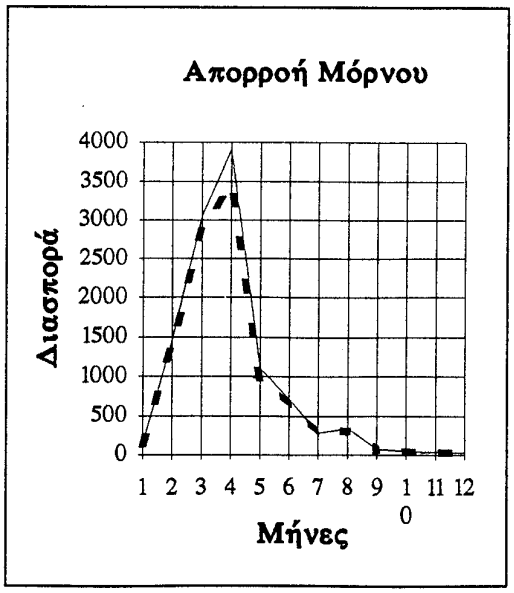
5.1 Εφαρμογή

Η εφαρμογή που ακολουθεί έγινε με δεδομένα από την περιοχή του Μόρνου. Οι θέσεις-σταθμοί είναι τρείς. Η πρώτη είναι ο βροχομετρικός σταθμός στο Λιδωρίκι, η δεύτερη είναι οι επιφανειακές βροχές Μόρνου και η τρίτη οι απορροές Μόρνου. Η επιλογή αυτή έγινε με το εξής σκεπτικό. Να έχουμε ποικιλλία μεταβλητών (βροχή, απορροή), να έχουμε συσχέτιση μεταξύ των βροχών (Λιδωρίκι, επιφανειακές Μόρνου), να έχουμε τέλος δείγμα με σημαντικές μηνιαίες αυτοσυσχετίσεις (απορροές). Το δείγμα έχει μέγεθος μόνο 19 ετών και τα συνθετικά δεδομένα που παράχθηκαν είναι χιλίων ετών (σχετικά μικρό μήκος). Παρακάτω ακολουθούν διαγράμματα στα οποία γίνεται σύγκριση των πραγματικών στατιστικών δεδομένων με τα συνθετικά.

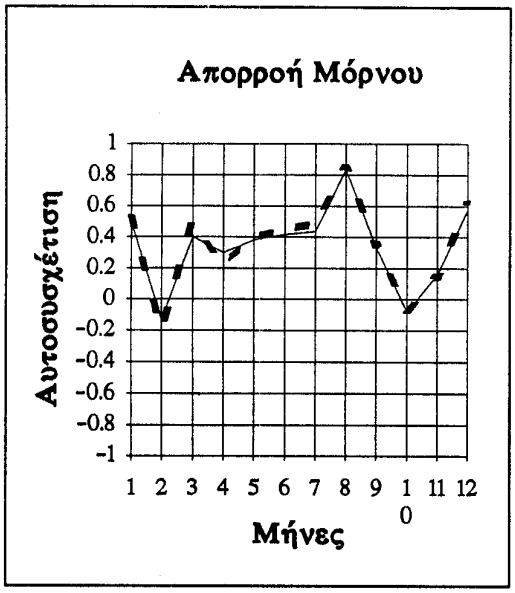
Στα διαγράμματα που ακολουθούν η συνεχής γραμμή αντιστοιχεί στα στατιστικά μεγέθη του πραγματικού δείγματος, ενώ η διακεκκομένη στα αντίστοιχα του συνθετικού.



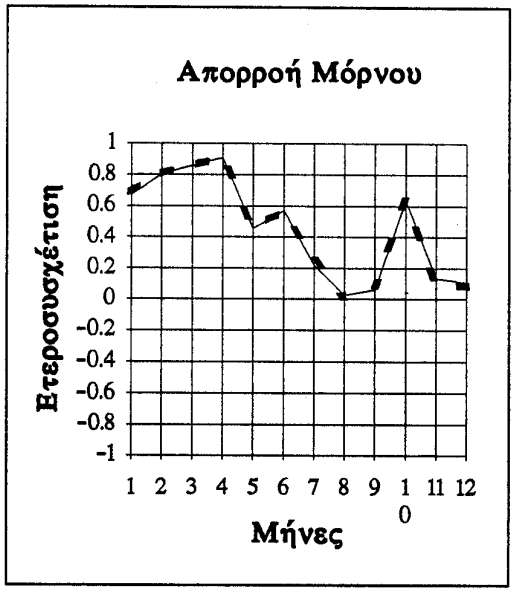
Διάγραμμα 1



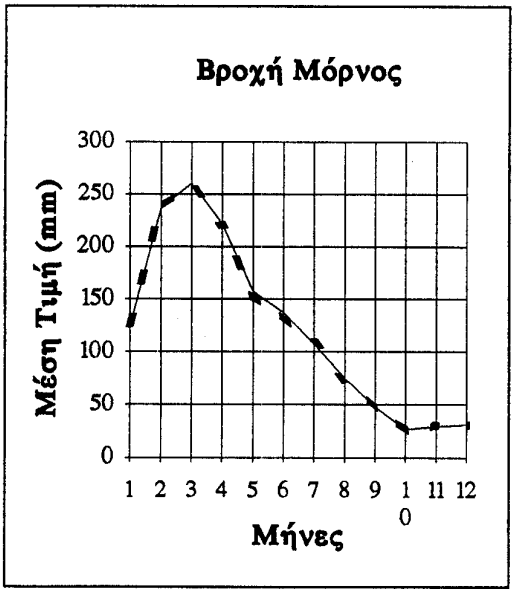
Διάγραμμα 2



Διάγραμμα 3

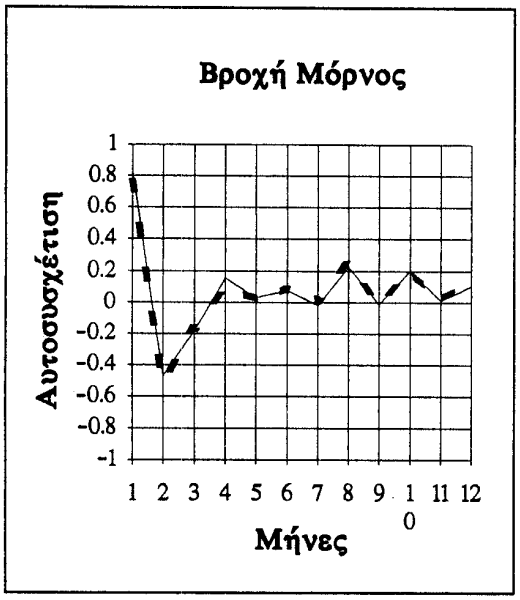


Διάγραμμα 4

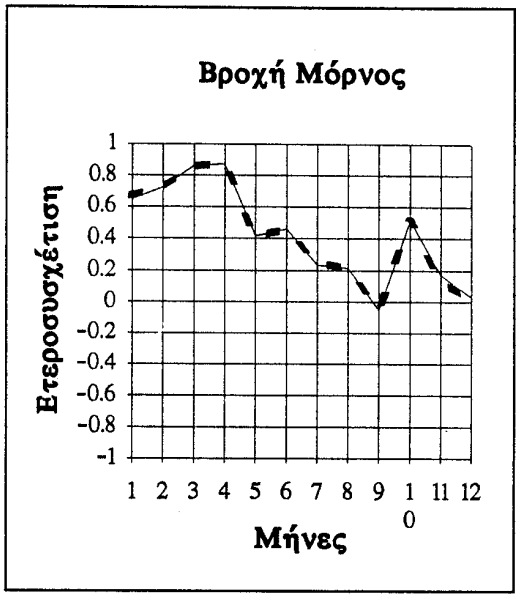


Διάγραμμα 5

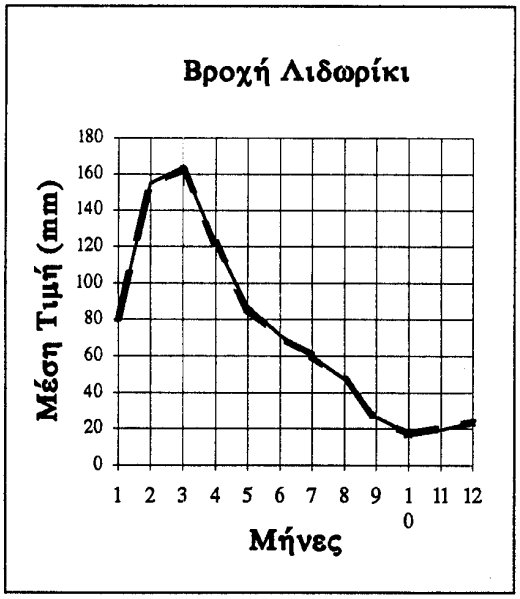
Διάγραμμα 6



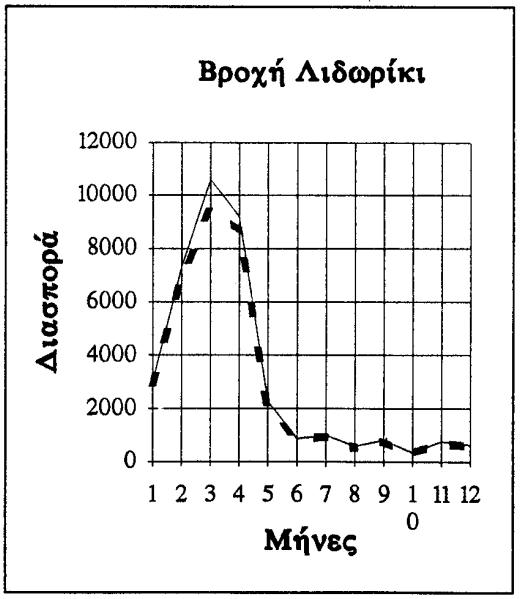
Διάγραμμα 7



Διάγραμμα 8

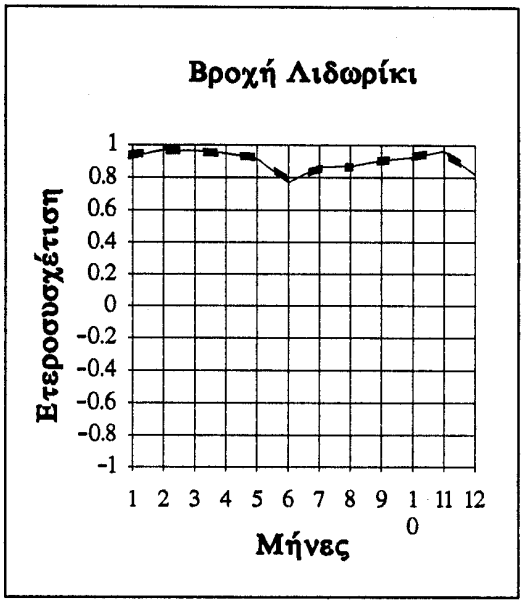


Διάγραμμα 9



Διάγραμμα 10

Διάγραμμα 11



Διάγραμμα 12

5.2 Σχόλια

Παρατηρούμε ότι ακόμη και με την παραγωγή μόνο χιλίων περιόδων συνθετικών δεδομένων το μοντέλο επιμερισμού ακολουθεί πιστά την πραγματικότητα. Οι μέσες τιμές, οι διασπορές, οι αυτοσυσχετίσεις και οι ετεροσυσχετίσεις των συνθετικών δεδομένων σχεδόν ταυτίζονται με τα αντίστοιχα στατιστικά μεγέθη των πραγματικών. Το παραπάνω παράδειγμα έχει γίνει με δέκα βήματα επιμερισμού αλλά ακόμη και με δυο μόνο βήματα τα αποτελέσματα είναι εξίσου ικανοποιητικά. Η σύγκλιση του μοντέλου εξαρτάται τελικά σε μεγάλο βαθμό μόνο από το πλήθος των παραγόμενων περιόδων. Έχει παρατηρηθεί ότι ο ελάχιστος αριθμός σύγκλισης είναι χίλιες περίοδοι. Κατά τη διάρκεια της διάσπασης των μητρώων **B** δεν χρειάστηκε καμμία διόρθωση γιατί οι διασπορές των σειρών ήταν αρκετά μεγαλύτερες από τις συνδιασπορές.

ΚΩΔΙΚΑΣ ΠΡΟΓΡΑΜΜΑΤΟΣ

Ο κώδικας του προγράμματος που ακολουθεί παρουσιάζει μόνο τις μαθηματικές διαδικασίες που εκτελούνται στο SIDIS. Τα τμήματα που σχετίζονται με την επικοινωνία του χρήστη με το πρόγραμα δεν περιλαμβάνονται γιατί ένα μεγάλο μέρος τους δημιουργείται αυτόματα από την Visual Basic χωρίς να παράγεται πηγαίος κώδι.κας.

Αλγόριθμος Ετήσιου μοντέλου (κώδικας Basic)

```
Sub arl (inpt() As Double, ByVal stations As Integer, ByVal years As Integer)
 Static i, j, k, t As Integer
 Static temp(2) As Double
 Static myerr As Integer
 Dim a() As Double
 Dim s() As Double
 Dim bbt() As Double
 Dim workl() As Double
 Dim work2() As Double
 Dim b() As Double
 Dim v() As Double
 Dim x() As Double
 Dim ax() As Double
 Dim bv() As Double
 Dim a__e__x() As Double
 Dim b__1() As Double
 Dim stats() As Double
 Dim ev() As Double
  ReDim a(stations)
 ReDim v(stations)
 ReDim ev(stations)
 ReDim a__e__x(stations)
 ReDim b__1(stations, stations)
 ReDim stats(stations, 1)
 ReDim x(stations)
 ReDim ax(stations)
 ReDim bv(stations)
 ReDim s(stations, stations)
 ReDim bbt(stations, stations)
 ReDim workl(years)
 ReDim work2(years)
```

```
ReDim b(stations, stations)

'open result file
titlename = Mid$(inputfile, 1, Len(inputfile) - 4)
Open titlename + ".arr" For Output As #1

'calc matrix a
Print #1, "**Matrix a**"
For i = 1 To stations
  For j = 1 To years
    work1(j) = inpt(i, j)
  Next j
  Call auto__covariance(work1(0), years, 1, temp(0))
  a(i) = temp(1)
  Call var(work1(0), years, temp(0))
  a(i) = a(i) / temp(1)
  Print #1, fixnum(a(i), 3) + " ";
  Call mean(work1(0), years, temp(0))
  stats(i, 1) = temp(1)
Next i
Print #1,

'calc matrix s
Print #1, "**Matrix s**"
For i = 1 To stations
  For j = 1 To i
    For k = 1 To years
      work1(k) = inpt(i, k)
      work2(k) = inpt(j, k)
    Next k
    Call covariance(work1(0), work2(0), years, years, temp(0))
    s(i, j) = temp(1)
    s(j, i) = temp(1)
    Print #1, fixnum(s(i, j), 0) + ",";
  Next j
  Print #1,
Next i

'calc bbt matrix
Print #1, "**Matrix BBt**"
For i = 1 To stations
  For j = 1 To stations
    bbt(i, j) = s(i, j) * a(i) * a(j)
    bbt(i, j) = s(i, j) - bbt(i, j)
```

```
      Print #1, fixnum(bbt(i, j), 0) + ",";
    Next j
    Print #1,
  Next i

  'decomposition of BBt
  Print #1, "**Matrix B**"
  ReDim work1(stations * stations)
  ReDim work2(stations * stations)
  k = -1
  For i = 1 To stations
    For j = 1 To stations
      k = k + 1
      work1(k) = bbt(i, j)
    Next j
  Next i
  If decomp = 1 Then
    myerr = jacobi(work1(0), work2(0), stations)
  Else
    myerr = decomposition(work1(0), work2(0), stations)
  End If
  If myerr = -1 Then
    MsgBox "Not positive defined matrix sorry..."
    Exit Sub
  End If
  k = -1
  For i = 1 To stations
    For j = 1 To stations
      k = k + 1
      b(i, j) = work2(k)
      Print #1, fixnum(b(i, j), 5) + ",";
    Next j
    Print #1,
  Next i
  Erase work1, work2, bbt, s

  ' invert bbt matrix
  k = -1
  ReDim work1(stations * stations)
  ReDim work2(stations * stations * 2)
  For i = 1 To stations
   For j = 1 To stations
    k = k + 1
    work1(k) = b(i, j)
```

```
 Next j
Next i
myerr = matrix__inverse(work1(0), work2(0), stations)
If myerr = -1 Then MsgBox "Zero Diagonal sorry ...": Exit Sub
t = -1
  For k = 1 To stations
    t = t + stations
    For j = 1 To stations
      t = t + 1
      b__1(k, j) = work2(t)
      Print #1, fixnum(b__1(k, j), 5) + ",";
    Next j
    Print #1,
  Next k
Close #1

For i = 1 To stations
  x(i) = stats(i, 1)
Next i

Open titlename + ".arg" For Output As #1
Print #1, stations, 1, num__of__data
For i = 1 To num__of__data
  For j = 1 To stations
    ax(j) = a(j) * x(j)
  Next j

  For j = 1 To stations
    a__e__x(j) = stats(j, 1) * a(j)
    a__e__x(j) = stats(j, 1) - a__e__x(j)
  Next j

  For j = 1 To stations
   ev(j) = 0#
   For k = 1 To stations
     ev(j) = ev(j) + b__1(j, k) * a__e__x(k)
   Next k
   Call gen__normal(ev(j), 1, temp(0))
   v(j) = temp(1)
  Next j

  For j = 1 To stations
    bv(j) = 0#
    For k = 1 To stations
```

```
            bv(j) = bv(j) + b(j, k) * v(k)
        Next k
    Next j

    For j = 1 To stations
        x(j) = ax(j) + bv(j)
        Print #1, fixnum(x(j), 5)
    Next j
Next i
Close #1
MsgBox "Generation has finished"
End Sub
```

Αλγόριθμος Περιοδικού μοντέλου (κώδικας Basic)

```
Sub arlpoly (inpt() As Double, ByVal stations As Integer, ByVal periods As Integer, ByVal years As Integer)
 Static i, j, k, t, l As Integer
 Static temp(2) As Double
 Static myerr As Integer
 Static mn As Double
 Dim a__e__x() As Double
 Dim e__ae() As Double
 Dim b() As Double
 Dim b__1() As Double
 Dim a() As Double
 Dim work1() As Double
 Dim work2() As Double
 Dim s() As Double
 Dim bbt() As Double
 Dim x() As Double
 Dim ax() As Double
 Dim v() As Double
 Dim bv() As Double
 Dim stats() As Double

 ReDim ax(stations)
 ReDim bv(stations)
 ReDim x(stations)
 ReDim a__e__x(stations)
 ReDim e__ae(stations)
 ReDim v(stations)
 ReDim stats(stations, periods, 2)
 ReDim b(periods, stations, stations)
 ReDim bbt(periods, stations, stations)
```

```
ReDim a(stations, periods)
ReDim s(periods, stations, stations)
ReDim work1(years + 1)
ReDim work2(years + 1)
ReDim b__1(periods, stations, stations)

'Output file
inputfile = Mid$(inputfile, 1, Len(inputfile) - 4)
Open inputfile + ".prr" For Output As #1

'calculation of matrix a
Print #1, "Matrix A"
For i = 1 To periods
  For j = 1 To stations
    For k = 1 To years
      If i = 1 Then
        work1(k) = inpt(j, i, k)
        work2(k) = inpt(j, periods, k)
      Else
        work1(k) = inpt(j, i, k)
        work2(k) = inpt(j, i - 1, k)
      End If
    Next k
    Call mean(work1(0), years, temp(0))
    stats(j, i, 1) = temp(1)
    Call var(work1(0), years, temp(0))
    stats(j, i, 2) = Sqr(temp(1))
    If i = 1 Then
      Call covariance(work1(1), work2(0), years - 1, years - 1, temp(0))
    Else
      Call covariance(work1(0), work2(0), years, years, temp(0))
    End If
    a(j, i) = temp(1)
    If i = 1 Then
      Call var(work2(0), years - 1, temp(0))
    Else
      Call var(work2(0), years, temp(0))
    End If
    a(j, i) = a(j, i) / temp(1) * (years + 1) / years
    Print #1, fixnum(a(j, i), 2) + "  ";
  Next j
  Print #1,
Next i
```

```
'calculation of matrix s
Print #1,
Print #1, "Matrix S"
For i = 1 To periods
  For j = 1 To stations
    For k = 1 To j
      For l = 1 To years
        work1(l) = inpt(j, i, l)
        work2(l) = inpt(k, i, l)
      Next l
      Call covariance(work1(0), work2(0), years, years, temp(0))
      s(i, j, k) = temp(1)
      s(i, k, j) = temp(1)
      Print #1, fixnum(s(i, j, k), 2) + "  ";
    Next k
    Print #1,
  Next j
  Print #1,
Next i

'calculation of bbt matrix
Print #1,
Print #1, "Matrix BBt"
For i = 1 To periods
  For j = 1 To stations
    For k = 1 To j
      If i = 1 Then
        bbt(i, j, k) = a(j, i) * a(k, i) * s(periods, j, k)
        bbt(i, j, k) = s(i, j, k) - bbt(i, j, k)
      Else
        bbt(i, j, k) = a(j, i) * a(k, i) * s(i - 1, j, k)
        bbt(i, j, k) = s(i, j, k) - bbt(i, j, k)
      End If
      Print #1, fixnum(bbt(i, j, k), 2) + "  ";
      bbt(i, k, j) = bbt(i, j, k)
    Next k
    Print #1,
  Next j
  Print #1,
Next i

'Calculate b from bbt
Print #1,
ReDim work1(stations * stations)
```

```
ReDim work2(stations * stations)
Print #1, "Matrix B"
For i = 1 To periods
  t = -1
  For j = 1 To stations
    For k = 1 To stations
      t = t + 1
      work1(t) = bbt(i, j, k)
    Next k
  Next j
  If decomp = 2 Then
    myerr = decomposition(work1(0), work2(0), stations)
  Else
    myerr = jacobi(work1(0), work2(0), stations)
  End If
  If myerr = -1 Then
    MsgBox "Negative eigenvalue sorry..."
    Close #1
    Exit Sub
  End If
  t = -1
  For k = 1 To stations
    For j = 1 To stations
      t = t + 1
      b(i, k, j) = work2(t)
      Print #1, fixnum(b(i, k, j), 3) + "  ";
    Next j
    Print #1,
  Next k
  Print #1,
Next i

'Calculate b__1 from b
ReDim work1(stations * stations)
ReDim work2(stations * stations * 2)
Print #1,
Print #1, "Matrix B__1"
For i = 1 To periods
  t = -1
  For j = 1 To stations
    For k = 1 To stations
      t = t + 1
      work1(t) = b(i, j, k)
    Next k
```

```
    Next j
    myerr = matrix__inverse(work1(0), work2(0), stations)
    If myerr = -1 Then
       MsgBox "Zero diagonal sorry ... "
       Close #1
       Exit Sub
    End If

    t = -1
    For k = 1 To stations
       t = t + stations
       For j = 1 To stations
          t = t + 1
          b__1(i, k, j) = work2(t)
          Print #1, fixnum(b__1(i, k, j), 4) + "   ";
       Next j
       Print #1,
    Next k
    Print #1,
 Next i
 Print #1,
 Print #1, "Statistics Of Historic Data Mean, Variance"
 For i = 1 To stations
    For j = 1 To periods
          Print #1, "site " + CStr(i) + " subperiod " + CStr(j) + " " + CStr(fixnum(stats(i, j, 1), 3)) + " " +
CStr(fixnum(stats(i, j, 2), 2))
    Next j
 Next i
 Close #1
 Erase bbt, work1, work2, s

 'Data generation
 Open inputfile + ".prg" For Output As #1

 'first x's for the generation
 For k = 1 To stations
    x(k) = stats(k, periods, 1)
 Next k

 'start generation
 Print #1, stations, periods, num__of__data
 For i = 1 To num__of__data
    For j = 1 To periods
       For k = 1 To stations
```

```
            ax(k) = a(k, j) * x(k)
            If j = 1 Then
               a__e__x(k) = a(k, j) * stats(k, periods, 1)
            Else
               a__e__x(k) = a(k, j) * stats(k, j - 1, 1)
            End If
            e__ae(k) = stats(k, j, 1) - a__e__x(k)
         Next k
         For k = 1 To stations
            a__e__x(k) = 0#
            For l = 1 To stations
               a__e__x(k) = a__e__x(k) + b__l(j, k, l) * e__ae(l)
            Next l
            Call gen__normal(a__e__x(k), 1, temp(0))
            'Call gen__gamma(1.35, 3.45, temp(0))
            v(k) = temp(1)
         Next k
         For k = 1 To stations
            bv(k) = 0#
            For l = 1 To stations
               bv(k) = bv(k) + b(j, k, l) * v(l)
            Next l
            x(k) = ax(k) + bv(k)
            Print #1, x(k)
         Next k
      Next j
   Next i
   Close #1
   MsgBox "Generation has finished"
End Sub
```

Αλγόριθμος επιμερισμού (κώδικας Basic)

```
Sub disaggr (inpt() As Double, sum() As Double, aar() As Double, ByVal stations As Integer, ByVal periods
As Integer, ByVal years As Integer)
 Static i, j, k, t, l, index As Integer
 Static temp(2) As Double
 Static myerr, select__val As Integer
 Static mn As Double
 Static crit As Double
 Static mincrit As Double
 Dim a__e__x() As Double
 Dim e__ae() As Double
 Dim b() As Double
```

```
Dim b__1() As Double
Dim a() As Double
Dim work1() As Double
Dim work2() As Double
Dim s() As Double
Dim bbt() As Double
Dim tempx() As Double
Dim x() As Double
Dim ax() As Double
Dim v() As Double
Dim lamda() As Double
Dim bv() As Double
Dim stats() As Double
Dim xk() As Double

ReDim ax(stations)
ReDim xk(stations)
ReDim bv(stations)
ReDim x(stations)
ReDim a__e__x(stations)
ReDim e__ae(stations)
ReDim v(stations)
ReDim stats(stations, periods, 2)
ReDim b(periods, stations, stations)
ReDim bbt(periods, stations, stations)
ReDim a(stations, periods)
ReDim s(periods, stations, stations)
ReDim work1(years + 1)
ReDim work2(years + 1)
ReDim b__1(periods, stations, stations)

'calculation of matrix a
For i = 1 To periods
  For j = 1 To stations
    For k = 1 To years
      If i = 1 Then
        work1(k) = inpt(j, i, k)
        work2(k) = inpt(j, periods, k)
      Else
        work1(k) = inpt(j, i, k)
        work2(k) = inpt(j, i - 1, k)
      End If
    Next k
    Call mean(work1(0), years, temp(0))
```

```
      stats(j, i, 1) = temp(1)
      Call var(work1(0), years, temp(0))
      stats(j, i, 2) = Sqr(temp(1))
      If i = 1 Then
         Call covariance(work1(1), work2(0), years - 1, years - 1, temp(0))
      Else
         Call covariance(work1(0), work2(0), years, years, temp(0))
      End If
      a(j, i) = temp(1)
      If i = 1 Then
         Call var(work2(0), years - 1, temp(0))
      Else
         Call var(work2(0), years, temp(0))
      End If
      a(j, i) = a(j, i) / temp(1)
   Next j
Next i

'calculation of matrix s
For i = 1 To periods
   For j = 1 To stations
      For k = 1 To j
         For l = 1 To years
            work1(l) = inpt(j, i, l)
            work2(l) = inpt(k, i, l)
         Next l
         Call covariance(work1(0), work2(0), years, years, temp(0))
         s(i, j, k) = temp(1)
         s(i, k, j) = temp(1)
      Next k
   Next j
Next i

'calculation of lamda coefficients
ReDim lamda(stations, periods)
For i = 1 To stations
   mn = 0#
   For j = 1 To periods
      For k = 1 To years
         work1(k) = inpt(i, j, k)
         work2(k) = sum(i, k)
      Next k
      Call covariance(work1(0), work2(0), years, years, temp(0))
      lamda(i, j) = temp(1)
```

```
      Call covariance(work2(0), work2(0), years, years, temp(0))
      lamda(i, j) = lamda(i, j) / temp(1)
   Next j
Next i
Erase sum

'calculation of bbt matrix
For i = 1 To periods
   For j = 1 To stations
      For k = 1 To j
         If i = 1 Then
            bbt(i, j, k) = a(j, i) * a(k, i) * s(periods, j, k)
            bbt(i, j, k) = s(i, j, k) - bbt(i, j, k)
         Else
            bbt(i, j, k) = a(j, i) * a(k, i) * s(i - 1, j, k)
            bbt(i, j, k) = s(i, j, k) - bbt(i, j, k)
         End If
         bbt(i, k, j) = bbt(i, j, k)
      Next k
   Next j
Next i

'Calculate b from bbt
ReDim work1(stations * stations)
ReDim work2(stations * stations)
For i = 1 To periods
   t = -1
   For j = 1 To stations
      For k = 1 To stations
         t = t + 1
         work1(t) = bbt(i, j, k)
      Next k
   Next j
   myerr = jacobi(work1(0), work2(0), stations)
   If myerr = -1 Then
      MsgBox "Negative eigenvalue sorry..."
      Exit Sub
   End If
   t = -1
   For k = 1 To stations
      For j = 1 To stations
         t = t + 1
         b(i, k, j) = work2(t)
      Next j
```

```
  Next k
Next i

'Calculate b__1 from b
ReDim work1(stations * stations)
ReDim work2(stations * stations * 2)
For i = 1 To periods
  t = -1
  For j = 1 To stations
    For k = 1 To stations
      t = t + 1
      work1(t) = b(i, j, k)
    Next k
  Next j
  myerr = matrix__inverse(work1(0), work2(0), stations)
  If myerr = -1 Then
    MsgBox "Zero diagonal sorry ... "
    Exit Sub
  End If

  t = -1
  For k = 1 To stations
    t = t + stations
    For j = 1 To stations
      t = t + 1
      b__1(i, k, j) = work2(t)
    Next j
  Next k
Next i
Erase bbt, work1, work2, s

'first x's for the generation
For k = 1 To stations
  x(k) = stats(k, periods, 1)
Next k

'start generation
Open Left$(inputfile, Len(inputfile) - 4) + ".dsg" For Output As #1
Print #1, stations, periods, num__of__data
ReDim tempx(stations, periods, num__of__steps)
For index = 1 To num__of__data
For i = 1 To num__of__steps
  For k = 1 To stations
    xk(k) = x(k)
```

```
    Next k
    For j = 1 To periods
      For k = 1 To stations
        ax(k) = a(k, j) * xk(k)
        If j = 1 Then
          a__e__x(k) = a(k, j) * stats(k, periods, 1)
        Else
          a__e__x(k) = a(k, j) * stats(k, j - 1, 1)
        End If
        e__ae(k) = stats(k, j, 1) - a__e__x(k)
      Next k
      For k = 1 To stations
        a__e__x(k) = 0#
        For l = 1 To stations
          a__e__x(k) = a__e__x(k) + b__1(j, k, l) * e__ae(l)
        Next l
        Call gen__normal(a__e__x(k), 1, temp(0))
        'Call gen__gamma(1.35, 3.45, temp(0))
        v(k) = temp(1)
      Next k
      For k = 1 To stations
        bv(k) = 0#
        For l = 1 To stations
          bv(k) = bv(k) + b(j, k, l) * v(l)
        Next l
        xk(k) = ax(k) + bv(k)
        tempx(k, j, i) = xk(k)
      Next k
    Next j
  Next i

  'find best fit, correct and write to file
  ReDim sum(stations, num__of__steps)
  mincrit = 1E+51
  For i = 1 To num__of__steps
    crit = 0#
    For j = 1 To stations
      sum(j, i) = 0#
      For k = 1 To periods
        sum(j, i) = sum(j, i) + tempx(j, k, i)
      Next k
      sum(j, i) = aar(j, index) - sum(j, i)
      crit = crit + sum(j, i)
    Next j
```

```
    If Abs(crit) < Abs(mincrit) Then mincrit = crit: select__val = i
  Next i
  For i = 1 To periods
    For j = 1 To stations
      tempx(j, i, select__val) = tempx(j, i, select__val) + sum(j, select__val) * lamda(j, i)
      Print #1, tempx(j, i, select__val)
    Next j
  Next i
  For i = 1 To stations
    x(i) = tempx(i, periods, select__val)
  Next i
  Next index
  Close #1
  MsgBox "Generation and disaggregation has finished"
End Sub
```

Αλγόριθμος σχεδιασμού δεδομένων σε χαρτί κανονικής κατανομής (κώδικας Basic)

```
Sub form__activate ()
 Static i, j, k, t As Integer
 Static sp1, sp2, sp3, sp4 As Double
 Dim timp() As array
 Dim timp2() As array
 Dim ltemp() As Double
 Static mn(2) As Double
 Static sigma(2) As Double
 Static mymax As Double

If message = "redraw" Then GoTo 12
If message <> "to__normal__plot" Then Exit Sub
message = ""

'settle the graph
picture1.Cls

gwidth = picture1.Width
gheight = picture1.Height
picture1.ScaleWidth = picture1.Width
picture1.ScaleHeight = picture1.Height

ofsetx = 1200
ofsety = 700
```

```
rwidth = gwidth - 2 * ofsetx
rheight = gheight - 2 * ofsety

'input data file
If Right$(inputfile, 3) = "pdt" Or Right$(inputfile, 3) = "prg" Then
  inputfile = Left$(inputfile, Len(inputfile) - 4) + ".prg"
ElseIf Right$(inputfile, 3) = "dsg" Then
  inputfile = Left$(inputfile, Len(inputfile) - 4) + ".dsg"
Else
  inputfile = Left$(inputfile, Len(inputfile) - 4) + ".arg"
End If

Open inputfile For Input As #1
Input #1, stations, periods, years
num_of_dat = stations * periods * years
ReDim dat(stations, periods, years)
For i = 0 To years - 1
  For j = 0 To periods - 1
    For k = 0 To stations - 1
      Input #1, dat(k, j, i)
    Next k
  Next j
Next i
Close #1

If Right$(inputfile, 3) = "pdt" Or Right$(inputfile, 3) = "prg" Or Right$(inputfile, 3) = "dsg" Then
  inputfile = Left$(inputfile, Len(inputfile) - 4) + ".pdt"
Else
  inputfile = Left$(inputfile, Len(inputfile) - 4) + ".adt"
End If

Open inputfile For Input As #1
Input #1, stations2, periods2, years2
num_of_dat2 = stations2 * periods2 * years2
ReDim dat2(stations2, periods2, years2)
For i = 0 To years2 - 1
  For j = 0 To periods2 - 1
    For k = 0 To stations2 - 1
      Input #1, dat2(k, j, i)
    Next k
  Next j
Next i
Close #1
```

```
 MsgBox "Loading data finished Subperiods " + CStr(periods) + " Sites " + CStr(stations) + " Periods " +
CStr(years)
Exit Sub

12 :
 max = -1E+40
 min = 1E+31
 picture1.Cls
 picture1.FontBold = True
 picture1.PSet (picture1.Width / 2 - 500, picture1.Height - ofsety / 2)
 picture1.Print "Probability"
 picture1.PSet (10, picture1.Height / 2)
 picture1.Print "Values"
 picture1.FontBold = False

 picture1.FontSize = 8
 ReDim timp(years)
 ReDim ltemp(years)
 For k = 0 To years - 1
   timp(k).value = dat(from__site, period__, k)
   timp(k).index = k
   ltemp(k) = timp(k).value
   If dat(from__site, period__, k) › max Then max = dat(from__site, period__, k)
   If dat(from__site, period__, k) ‹ min Then min = dat(from__site, period__, k)
 Next k
 min = 0
 mymax = Int(max / 5 + 1) * 5

 Call mean(ltemp(0), years - 1, mn(0))
 Call var(ltemp(0), years - 1, sigma(0))
 sigma(1) = Sqr(sigma(1))

 ReDim timp2(years2)
 For k = 0 To years2 - 1
   timp2(k).value = dat2(from__site, period__, k)
   timp2(k).index = k
   If dat2(from__site, period__, k) › max Then max = dat2(from__site, period__, k)
   If dat2(from__site, period__, k) ‹ min Then min = dat2(from__site, period__, k)
 Next k

 min = 0
 max = Int(max / 5 + 1) * 5

 'fix frame
```

```
picture1.Line (ofsetx, ofsety)-(rwidth + ofsetx, ofsety)
picture1.Line (rwidth + ofsetx, ofsety)-(rwidth + ofsetx, rheight + ofsety)

'fix axes and ticks
picture1.Line (ofsetx, rheight + ofsety)-(rwidth + ofsetx, rheight + ofsety), QBColor(4)
picture1.Line (ofsetx, ofsety)-(ofsetx, rheight + ofsety)

'horizondial axe
For i = -3 To 3
   picture1.Line (ofsetx + rwidth / 6 * (i + 3), rheight + ofsety - 50)-(ofsetx + rwidth / 6 * (i + 3), rheight +
ofsety + 50)
   Call normal_area(0#, 1#, CDbl(i), temp(0))
   picture1.Print fixnum(temp(1), 3)
Next i

'vertical axe
For i = 1 To 6
   picture1.Line (ofsetx + 50, ofsety + rheight - (i - 1) * rheight / 5)-(ofsetx - 100, ofsety + rheight - (i - 1) *
rheight / 5)
   picture1.PSet (ofsetx - 500, ofsety + rheight - (i - 1) * rheight / 5), QBColor(7)
   picture1.Print fixnum(min + (i - 1) * (max - min) / 5, 0)
Next i

'sort the data
Call qsort(timp(), 0, years - 1)
Call qsort(timp2(), 0, years2 - 1)
grid1.Rows = years + 1
grid1.Cols = 7
For i = 1 To 6
   grid1.ColWidth(i) = 1600
Next i
grid1.Row = 0
grid1.Col = 1
grid1.Text = "xg"
grid1.Col = 2
grid1.Text = "F(xg)"
grid1.Col = 3
grid1.Text = "index"
grid1.Col = 4
grid1.Text = "xr"
grid1.Col = 5
grid1.Text = "F(xr)"
grid1.Col = 6
grid1.Text = "index"
```

```
Text1.Text = "Site " + CStr(from_site + 1) + " Subperiod " + CStr(period_ + 1)
For i = 0 To years - 1
  grid1.Col = 1
  grid1.Row = i + 1
  grid1.Text = CStr(fixnum(timp(i).value, 3))
  grid1.Col = 2
  grid1.Text = CStr(fixnum((i + 1) / (years + 1), 3))
  grid1.Col = 3
  grid1.Text = CStr(timp(i).index)
Next i
For i = 0 To years2 - 1
  grid1.Col = 4
  grid1.Row = i + 1
  grid1.Text = CStr(fixnum(timp2(i).value, 3))
  grid1.Col = 5
  grid1.Text = CStr(fixnum((i + 1) / (years2 + 1), 3))
  grid1.Col = 6
  grid1.Text = CStr(timp2(i).index)
Next i

'plot the points
t = 0
num_of_dat = years
sp1 = ofsetx + rwidth / 2
sp2 = rwidth / 6
sp3 = ofsety - rheight / (min - max) * max
sp4 = rheight / (min - max)
For k = 0 To years - 1
  t = t + 1
  tmp = t / (years + 1)
  If timp(k).value < 0 Then GoTo 13
  Call inv_normal_area(0#, 1#, tmp, temp(0))
  'Call normal_area(sum, sum2, dat(i, j, k), temp(0))
  picture1.Circle (sp1 + temp(1) * sp2, (sp3 + sp4 * timp(k).value)), 20

13 'skip negative values
Next k
t = 0
num_of_dat = years2
For k = 0 To years2 - 1
  t = t + 1
  tmp = t / (years2 + 1)
  Call inv_normal_area(0#, 1#, tmp, temp(0))
```

```
    'Call normal__area(sum, sum2, dat(i, j, k), temp(0))
    picture1.Circle (sp1 + temp(1) * sp2, (sp3 + sp4 * timp2(k).value)), 20, &HFF
  Next k

'normal line

  picture1.Line (sp1 + sp2 * (-mn(1) / sigma(1)), (sp3 + sp4 * 0#))-(sp1 + sp2 * (mymax - mn(1)) / sigma(1), sp3 +
'sp4 * mymax), &HFF
  message = ""
  Erase timp, timp2
End Sub
```

Στατιστική βιβλιοθήκη (κώδικας C++)

```
#include ‹math.h›
#include ‹stdlib.h›
#include ‹windows.h›
#define  MISSINGVAL -999

long seed;
double myerfc(double);

int FAR PASCAL LibMain(HANDLE hInstance, WORD wDataSeg,
                       WORD wHeapSize, LPSTR lpszCmdLine)
{
        if(wHeapSize!=0)
                UnlockData(0);
        return 1;
}

void FAR PASCAL normal__density(double mean, double sigma, double x, double *nd)
{
        x = (x - mean) / sigma;
        nd[1] = exp(-(x * x) / 2.0) / sqrt(2.0 * M__PI);
}

double myerf(double x)
{
        double accuracy;
        double x2, product, sum;
        long n;
        accuracy = 1e-10;

        if (x ‹ accuracy)
```

```
        {
                return (0);
                }

        if (x > 1.5)
          {
                return 1.0 - myerfc(x);
                }
        else
                x2 = x * x;
                n = 0;
                product = x;
                sum = x;
                do
                {
                n = n + 1;
                        product = product * 2.0 * x2 / (1.0 + 2.0 * (double)n);
                        sum = sum + product;
                }
                while(product > sum * accuracy);

                return 2.0 * sum * exp(-x2) / sqrt(M__PI);
}

double myerfc(double x)
{
        int limit, i;
        double term, compound, v;
        limit = 12;
        if (x <= 1.5)
                {
              return 1.0 - myerf(x);
          }
        else
                v = 0.5 / x / x;
                term = 1.0 + v * (double)(limit + 1);
                for (i = limit; i > 0; i--)
                        {
                                compound = 1.0 + i * v / term;
                                term = compound;
                        }
                return exp(- x * x) / (x * compound * sqrt(M__PI));
}
```

```
double c__normal__area(double mean, double sigma, double x)
{

        x = (x - mean) / sigma;
        if (x == 0.0)
                return 0.5;
    if (x > 0.0)
        {
                return 0.5 * (1.0 + myerf(x/sqrt(2.0)));
        }

        if (x < 0.0)
                {
                return 1.0 - 0.5 * (1.0 + myerf(-x/sqrt(2.0)));
                }
}

void FAR PASCAL normal__area(double mean, double sigma, double x, double *na)
{

        x = (x - mean) / sigma;
        if (x == 0.0)
    {
                na[1] =  0.5;
                return;
    }
    if (x > 0.0)
        {
                na[1] = 0.5 * (1.0 + myerf(x/sqrt(2.0)));
        }

        if (x < 0.0)
                {
                na[1] = 1.0 - 0.5 * (1.0 + myerf(-x/sqrt(2.0)));
                }
}

void FAR PASCAL inv__normal__area(double mean, double sigma, double Fx, double *ina)
{
        double sv, oldsv, sol, parag, parag1, parag2;
        int i;

        oldsv = mean;
```

```
        for (i = 1; i < 150; i++)
        {
                sol = c__normal__area(mean, sigma, oldsv) - Fx;
                parag1 = c__normal__area(mean, sigma, oldsv + 0.1);
                parag2 = c__normal__area(mean, sigma, oldsv - 0.1);
           parag = (parag1 - parag2) / 0.2;
                sv = oldsv - sol / parag;
                if (fabs(sv - oldsv) < .00000001)
                        {
                                ina[1] = sv;
                        }
                oldsv = sv;
        }
}

void FAR PASCAL set__seed(long set)
{
        seed = set;
}

double myrand(void)
{
        long  a, b, c, x;
      double result;
        a = 313;
        b = 251;
        c = 6860969;
        x = (a * seed + b) % (c);
        seed = x;
      result = (double)x/(double)c;
        if (result == 0)
                result = 10e-6;
        return result;
}

void FAR PASCAL gen__normal(double mean, double std__dev, double *result)
{
        double f, nrnd1, z;
        z = sqrt(-2.0 * log(myrand()));
        f=2.0 * M__PI * myrand();
        nrnd1=z * cos(f);
        result[1]=std__dev * nrnd1 + mean;
}
```

```
void FAR PASCAL gen__gamma(double kapa, double lamda, double *result)
{
        double acc=1e-5;
        double GaussLimit=30;
        int k, i;
        double n, rnd1, rnd2, s1, s2, s, product;
        if(kapa <= 0.0)
                exit(1);
        if(kapa > GaussLimit)
                gen__normal(kapa / lamda, sqrt(kapa) / lamda, result);
        else
                {
                        k = (int)floor(kapa);
                        n = kapa-k;
                        if (n >= 1-acc)
                                {
                                k = k+1;
                                rnd1 = 0.0;
                                }
                        else
                                {
                                if(n <= acc)
                                        rnd1 = 0.0;
                                else
                                        {
                                                while((s>=1.0)&&(s<0.0))
                                                        {
                                                                s1 = pow(myrand(), 1.0/n);
                                                                s2 = pow(myrand(), 1.0/(1.0-n));
                                                                s = s1 + s2;
                                                        }
                                                rnd1=-(s1/s)*log(myrand());
                                        }
                                }
                        if(k == 0)
                        rnd2=0.0;
                        else
                                {
                                product = 1;
                                for(i = 1;i <= k;i++)
                                product = product * myrand();
                                rnd2 = -log(product);
                                }
                        result[1] = (rnd1+rnd2)/lamda;
```

```
                }
}

void FAR PASCAL mean(double *array, int size, double *result)
{
        int i;
        double sum;
    sum=0.0;
        for(i=1;i<=size;i++)
                sum=sum+array[i];
        result[1] = sum/(double)size;
}

void FAR PASCAL dev(double *array, int size, double *result)
{
        int i;
        double mn[2], sum, dif;
        sum=0.0;
        mean(array, size, mn);
        for(i=1;i<=size;i++)
        {
                dif=array[i]
                   -mn[1];
                sum=sum
                   +dif*dif;
    }
        result[1] = sum/(double)(size-1);
}

void FAR PASCAL central_moment(double *array, int size, int order, double *result)
{
        int i;
        double sum, mn[2];
        sum = 0.0;
        mean(array, size, mn);
        for(i = 1; i <= size; i++)
                sum = sum
                    + pow(array[i] - mn[1], order);
        result[1] = sum / (double)size;
}

void FAR PASCAL central_cross_moment(double *array1, double *array2, int size1,
                                int size2, int order1, int order2, double *result)
{
```

```
        int i;
        double sum, mean1[2], mean2[2];
        sum = 0.0;
        mean(array1, size1, mean1);
        mean(array2, size2, mean2);
        for(i = 1; i <= size1; i++)
                sum = sum
                    + pow(array1[i] - mean1[1], order1)
                    * pow(array2[i] - mean2[1], order2);
        result[1] = sum
                / (double)size1;
}

void FAR PASCAL skewness(double *array, int size, double *result)
{
        double m3[2], m2[2];
        dev(array, size, m2);
        central_moment(array, size, 3, m3);
        if (m2[1] != 0.0)
    {
                result[1] = (m3[1] / pow(m2[1], 1.5)) * size / (size-1) * size / (size-2);
                return;
    }
        else if (m3[1] == 0.0)
    {
                result[1] = 0.0;
          return;
    }
        else
                result[1] = 1e100;
}

void FAR PASCAL kurtosis(double *array, int size, double *result)
{
        double m4[2], m2[2];
        dev(array, size, m2);
        central_moment(array, size, 4, m4);
        if(m2[1] != 0.0)
    {
          result[1] = m4[1] / m2[1] / m2[1]-3.0;
          return;
    }
        else if(m4[1] == 0.0)
    {
```

```
                result[1] = 0.0;
                return;
        }
        else
                result[1] = 1e100;

}

void FAR PASCAL covariance(double *array1, double *array2, int size1, int size2,
                                double *result)
{
        int i;
        double m1[2], m2[2], sum;
        sum = 0.0;
        mean(array1, size1, m1);
        mean(array2, size2, m2);
        for(i = 1; i <= size1; i++)
                sum = sum + (array1[i] - m1[1]) * (array2[i] - m2[1]);
        result[1] = sum / (double)size1;
}

void FAR PASCAL correlation(double *array1, double *array2, int size1, int size2,
                                double *result)
{
        double cv[2], cm[2], cm2[2];
        covariance(array1, array2, size1, size2, cv);
        central_moment(array1, size1, 2, cm);
        central_moment(array2, size2, 2, cm2);
        result[1] = cv[1] / sqrt(cm[1]) / sqrt(cm2[1]);
}

void FAR PASCAL mean1( double *array, int size, int from, int tto, double *result)
{
        int i;
        double sum;
        if (tto > size)
                exit(1);
        if (from < 1)
                exit(1);
        sum = 0.0;
        for(i = from; i <= tto; i++)
                sum = sum + array[i];
        result[1] = sum / (double)(tto - from + 1);
}
```

```
void FAR PASCAL central__moment1(double *array, int size, int from, int tto, int order,
                                 double *result)
{
        int i;
        double sum, mn[2];
        mean(array, size, mn);
        sum = 0.0;
        for(i = from; i <= tto; i++)
                sum = sum + pow(array[i] - mn[1], order);
        result[1] = sum / (double)(tto - from + 1);
}

void FAR PASCAL auto__covariance(double *array, int size, int order, double *result)
{
        int i;
        double sum, m1[2], m2[2];
        int terma;
        sum = 0.0;
        mean1(array, size, 1, size - order, m1);
        mean1(array, size, order + 1, size, m2);
        terma = size - order;
        for(i = 1; i <= terma; i++)
                sum = sum + (array[i] - m1[1]) * (array[i + order] - m2[1]);
        result[1] = sum / (double)(size - order);
}

void FAR PASCAL regression(double *array1, int size1, double *array2,
                           int size2, double *a, double *b, double *r)
{
        int i, j;
        double sumy, sumxx, sumx, sumxy, sumyy;
    sumy = 0.0; sumxx = 0.0; sumx = 0.0; sumxy = 0.0; sumyy = 0.0;
    j = 0;
        if(size1 != size2)
                exit(1);
    for (i=1; i<=size1; i++)
        {
          j = j + 1;
                if((array1[i] != MISSINGVAL)&&(array2[i] != MISSINGVAL))
                {
                        array1[j] = array1[i];
                        array2[j] = array2[i];
          }
```

```
                else
                        j = j-1;
        }
        size1 = (double)j;
        for(i=1; i<=size1; i++)
        {
                sumx = sumx + array1[i];
                sumy = sumy + array2[i];
                sumxx = sumxx + array1[i]*array1[i];
          sumyy = sumyy + array2[i]*array2[i];
                sumxy = sumxy + array1[i]*array2[i];
        }
        a[1] = (sumy*sumxx - sumx*sumxy) / (size1*sumxx - sumx*sumx);
        b[1] = (size1*sumxy - sumx*sumy) / (size1*sumxx - sumx*sumx);
        r[1] = (size1*sumxy - sumx*sumy) / (sqrt((size1*sumxx - sumx*sumx)*
          (size1*sumyy - sumy*sumy)));
}


void FAR PASCAL auto__correlation(double *array, int size, int order, double *result)
{
        double s1, s2, *temp1, *temp2, a[2], b[2], r[2];
        int i, i1, i2;
        temp1 = (double *)malloc(sizeof(double)*(size + 2));
        temp2 = (double *)malloc(sizeof(double)*(size + 2));
      i1 = 0; i2 = 0;
        for(i=1; i<=size-order; i++)
        {
                if (array[i]!=MISSINGVAL && array[i+order]!=MISSINGVAL)
                {
               i2 = i - i1;
                        temp1[i2] = array[i];
                temp2[i2] = array[i+order];
                }
                else
                        i1 = i1 + 1;
        }
        regression(temp1, i2, temp2, i2, a, b, r);
        free(temp1);
      free(temp2);
        result[1] = r[1];
}

void FAR PASCAL cross__correlation(double *array1, double *array2, int size1, int size2,
```

```
                                        int lag, double *result)
{
        double a[2], b[2], r[2];
            if (size1 != size2)
            exit(1);
            regression((array1+lag), size1-lag, array2, size1-lag, a, b, r);
            result[1]=r[1];
}

void FAR PASCAL max__in__list(double *array, int size, double *max)
{
            int i;
        max[1] = array[1];
            for(i=2; i<=size; i++)
                        if(array[i] > max[1])
                                    max[1] = array[i];
}

void FAR PASCAL min__in__list(double *array, int size, double *min)
{
            int i;
        min[1] = array[1];
            for(i=2; i<=size; i++)
                        if(array[i] < min[1])
                                    min[1] = array[i];
}
```

Μαθηματική βιβλιοθήκη (κώδικας C++)

```
#include <windows.h>
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#define b(i,j) *(b+(i-1)*stations+j-1)
#define d(i,j) *(d+(i-1)*stations+j-1)
#define mb(i,j) *(mb+(i-1)*2*stations+j-1)
#define ROTATE(d, i, j, k, l) g=d(i,j);h=d(k,l);d(i,j)=g-s*(h+g*tau);\
                        d(k,l)=h+s*(g-h*tau);

int FAR PASCAL LibMain(HANDLE hInstance, WORD wDataSeg,
                                    WORD wHeapSize, LPSTR lpszCmdLine)
{
            return 1;
}
```

```
void swap(v, i, j)
double *v;
long i, j;
{
        double temp;
        temp = v[i];
        v[i] = v[j];
        v[j] = temp;
}

void FAR PASCAL csort(v, left, right)
double *v;
long left, right;
{
        long i, last;
        if(left >= right)
                return;
        swap(v, left, (left + right) / 2);
        last = left;
        for(i = left + 1; i <= right; i++)
                if(v[i] < v[left])
                        swap(v, ++last, i);
                swap(v, left, last);
                csort(v, left, last - 1);
                csort(v, last + 1, right);
}

int FAR PASCAL decomposition(double *d, double *b, int stations)
{
        double sum, dif;
        int i, j, t, k, l;

      /* error -1 diagonal element less or equal to zero */
        if (d(1,1)<=0.0)
                return(-1);

        b(1,1) = sqrt(d(1,1));

        for(i=2; i<=stations; i++)
                b(i,1) = d(1,i) / b(1,1);

        for(k=2; k<=stations; k++)
        {
```

```
            sum = 0.0;
                for(l=1; l<=k-1; l++)
                        sum = sum + b(k,l) * b(k,l);
                dif = d(k,k)-sum;

            /* not positive defined matrix */
                if (dif<0.0)
                        return(-2);

                b(k,k) = sqrt(dif);
                for(i=k+1; i<=stations; i++)
                {
                sum = 0.0;
                        for(t=1; t<=i-1; t++)
                                sum = sum + b(k,t) * b(i,t);
                        b(i,k) = (d(k,i)-sum) / b(k,k);
                }
        }
}

int FAR PASCAL matrix__inverse(double *d, double *mb, int stations)
{
        int i,j,k,n;
      double temp;

        for(i=1; i<=stations; i++)
        {
                for(j=1; j<=stations; j++)
                        mb(i,j) = d(i,j);
        }

        for(i=1; i<=stations; i++)
        {
                for(j=stations+1; j<=2*stations; j++)
                {
               if((i+stations)==j)
                                mb(i,j) = 1.0;
                        else
                                mb(i,j) = 0.0;
          }
        }

        for(j=1; j<=stations-1; j++)
        {
```

```
                for(i=j+1; i<=stations; i++)
                {
                /*diagonal zero*/
                        if(mb(j,j)==0.0)
                return(-1);
                        temp = mb(i,j) / mb(j,j);
                        for(k=1; k<=2*stations; k++)
                                mb(i,k) = mb(i,k)-temp * mb(j,k);
          }
        }

        for(j=stations; j>=2; j--)
        {
                for(i=j-1; i>=1; i--)
                {
                        temp = mb(i,j) / mb(j,j);
                        for(k=1; k<=2*stations; k++)
                                mb(i,k) = mb(i,k)-temp * mb(j,k);
          }
        }

        for(i=1; i<=stations; i++)
        {
                temp = mb(i,i);
                for(j=1; j<=2*stations; j++)
                        mb(i,j) = mb(i,j) / temp;
        }
    return(0);
}

int FAR PASCAL jacobi(double *d, double *b, int stations)
{
        int j, ip, iq, i, n, bug;
        double tresh, theta, tau, t, sm, s, h, g, c, bi[20], z[20], di[20];

        n = stations;

        for(ip=1;ip<=n;ip++)
        {
                for(iq=1;iq<=n;iq++)
                        b(ip,iq) = 0.0;
                b(ip,ip) = 1.0;
        }
```

```
for(ip=1;ip<=n;ip++)
{
        bi[ip]=di[ip]=d(ip,ip);
        z[ip]=0.0;
}

for(i=1;i<=120;i++)
{
        sm=0.0;
        for(ip=1;ip<=n-1;ip++)
        {
                for(iq=ip+1;iq<=n;iq++)
                        sm+=fabs(d(ip,iq));
        }
        if(fabs(sm) < 10e-200)
        {
                for(bug=1;bug<=n;bug++)
                {
                        for(j=1;j<=n;j++)
            {
                /* negative eigenvalue balamuti */
                        if (di[j]<0.0)
                di[j] = 0.01;

                        b(bug,j) = b(bug,j) * sqrt(di[j]);
            }
                }
        return(0);
        }
        if(i<4)
                tresh=0.2*sm/(n*n);
        else
                tresh=0.0;
        for(ip=1;ip<=n-1;ip++)
        {
                for(iq=ip+1;iq<=n;iq++)
                {
                        g=100.0*fabs(d(ip,iq));
                        if(i>4 && (double)(fabs(di[ip])+g)==(double)fabs(di[ip])
                            && (double)(fabs(di[iq])+g)==(double)fabs(di[iq]))
                                d(ip,iq)=0.0;
                        else if(fabs(d(ip,iq))>tresh)
                        {
                                h=di[iq]-di[ip];
```

```
                                if((double)(fabs(h)+g)==(double)fabs(h))
                                        t=(d(ip,iq))/h;
                                else
                                {
                                        theta=0.5*h/(d(ip,iq));
                                        t=1.0/(fabs(theta)+sqrt(1.0+theta*theta));
                                        if(theta<0.0) t= -t;
                                }
                                c=1.0/sqrt(1+t*t);
                                s=t*c;
                                tau=s/(1.0+c);
                                h=t*d(ip,iq);
                                z[ip]-=h;
                                z[iq]+=h;
                                di[ip]-=h;
                                di[iq]+=h;
                                d(ip,iq)=0.0;
                                for(j=1;j<=ip-1;j++)
                                {
                                        ROTATE(d,j,ip,j,iq)
                                }
                                for(j=ip+1;j<=iq-1;j++)
                                {
                                        ROTATE(d,ip,j,j,iq)
                                }
                                for(j=iq+1;j<=n;j++)
                                {
                                        ROTATE(d,ip,j,iq,j)
                                }
                                for(j=1;j<=n;j++)
                                {
                                        ROTATE(b,j,ip,j,iq)
                                }
                        }
                }
        }
        for(ip=1;ip<=n;ip++)
        {
                bi[ip]+=z[ip];
                di[ip]=bi[ip];
                z[ip]=0.0;
        }
}
```

```
    /* 120 interations and didn't convert */
    return(-2);
}
```

Βιβλιογραφία

- Koutsoyiannis D.
  Simple Disaggregation by accurate adjusting procedures
  unpublished report, N.T.U.A. 1994
- Papoulis A.
  Probability, Random Variables and Stochastic Processes
  Mc Graw-Hill 1965
- Papoulis A.
  Probability & Statistics
  Prentice-Hall International Editions 1990
- Rafael L. Bras, Ignacio Rodriguez-Iturbe
  Random Functions and Hydrology
  Addison-Wesley Publishing Company 1985
- J. D. Salas, J. W. Delleur, V. Yevjevich and W. L. Lane
  Applied Modeling of Hydrologic Time Series
  Water Resources Publication 1980
- W. H. Press, B. P. Flannery, S. A. Teukolsky, W. T. Vetterling
  Numerical Recipes in C
  Cambridge University Press 1988
- Kottegoda N. T.
  Stochastic Water Resources Technology
  Mac Millan Press, London, U.K. 1980