

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ Η/Υ, ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ ΚΑΙ ΔΙΚΤΥΩΝ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
HY200: ΕΠΙΣΤΗΜΟΝΙΚΟΣ ΥΠΟΛΟΓΙΣΜΟΣ

ΕΡΓΑΣΙΑ 1: Αριθμητική Γραμμική Άλγεβρα.

(Ημερομηνία Παράδοσης: Κυριακή 10 Απριλίου 2005, (Ώρα: 23:55))

(Η παράδοση της εργασίας θα γίνει μέσα από την σελίδα του μαθήματος στο eClass.)

Η εργασία αυτή έχει σκοπό την εξοικίωση με τις άμεσες και επαναληπτικές μεθόδους για την επίλυση γραμμικών συστημάτων.

Η πλειοψηφία των πραγματικών προβλημάτων μοντελοποιούνται με διαφορικές εξισώσεις. Η σχεδίαση ημιαγωγών (semiconductor devices), η μοντελοποίηση της θερμοκρασίας και της κατανάλωσης ενέργειας των ημιαγωγών μοντελοποιείται με ελλειπτικές διαφορικές εξισώσεις, π.χ.:

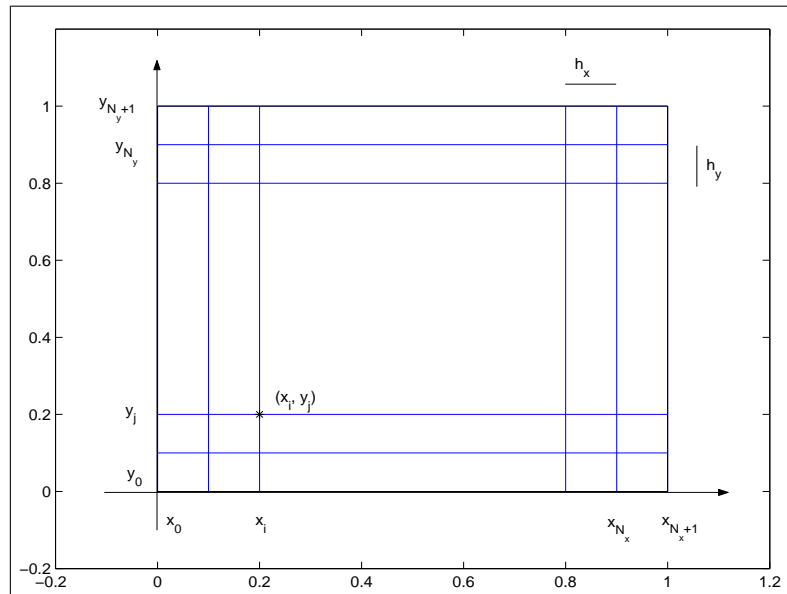
$$-\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) + c_1 \frac{\partial u}{\partial x} + c_2 \frac{\partial u}{\partial y} + cu = f, \text{ όπου } x, y \in (0, 1) \times (0, 1),$$
$$u(x, y) = 0, \text{ για } (x, y) \text{ στο σύνορο.}$$

Παρά το ότι θα τα δούμε αναλυτικά στο κεφάλαιο που αφορά την αριθμητική λύση συνήθων/μερικών διαφορικών εξισώσεων αναφέρουμε εδώ τα βασικά βήματα της διαδικασίας.

1. Διακριτοποιούμε το χώρο για να υπολογίσουμε τη λύση σε συγκεκριμένα (πεπερασμένα το πλήθος) σημεία. Στη συγκεκριμένη περίπτωση ορίζουμε τα σημεία

$$(x_i, y_j), \text{ όπου } x_i = i * h_x, h_x = 1/(N_x + 1), i = 0, 1, \dots, N_x + 1,$$
$$\text{και } y_j = j * h_y, h_y = 1/(N_y + 1), j = 0, 1, \dots, N_y + 1,$$

τα οποία αποτελούν ομοιόμορφη διαμέριση του $[0, 1] \times [0, 1]$, όπως φαίνεται στο Σχήμα 1. Τις τιμές της συνάρτησης u πάνω στα σημεία αυτά τις συμβολίζουμε με $u_{ij} = u(x_i, y_j)$.



Σχήμα 1. Ομοιόμορφο πλέγμα του $[0, 1] \times [0, 1]$.

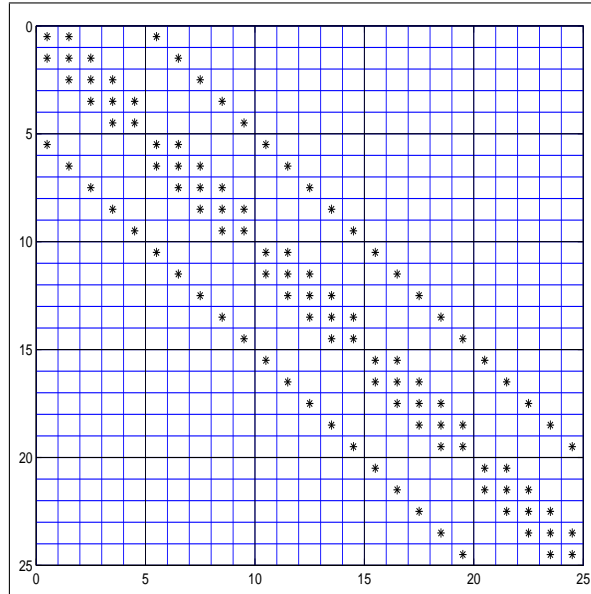
2. Διακριτοποιούμε τις παραγώγους χρησιμοποιώντας πεπερασμένες διαφορές, δηλαδή

$$\frac{\partial^2 u}{\partial x^2}(x_i, y_j) = \frac{-u_{i-1,j} + 2u_{i,j} - u_{i+1,j}}{h_x^2}, \quad \frac{\partial^2 u}{\partial y^2}(x_i, y_j) = \frac{-u_{i,j-1} + 2u_{i,j} - u_{i,j+1}}{h_y^2},$$

και

$$\frac{\partial u}{\partial x}(x_i, y_j) = \frac{u_{i+1,j} - u_{i-1,j}}{2h_x}, \quad \frac{\partial u}{\partial y}(x_i, y_j) = \frac{u_{i,j+1} - u_{i,j-1}}{2h_y}.$$

3. Γράφουμε το σύστημα των γραμμικών εξισώσεων σε μορφή πίνακα, $A * sol = f$, όπου ο πίνακας A είναι μορφής ζώνης, όπως φαίνεται στο Σχήμα 2.



Σχήμα 2. Μορφή πίνακα για πλέγμα 5×5 στο παραπάνω πεδίο $[0, 1] \times [0, 1]$.

4. Επιλύουμε το σύστημα με την κατάλληλη μέθοδο.

Ακολουθείστε τις παρακάτω υποδείξεις:

- Χρησιμοποιείτε τις συναρτήσεις `\`, `lu` του Matlab για την επίλυση γραμμικού συστήματος.
- Χρησιμοποιείτε τη συνάρτηση `elliptic` για να δημιουργήσετε τον πίνακα A και το δεξί μέλος f με κατάλληλα n_x, n_y, c_1, c_2, c .
- Χρησιμοποιείτε τη συνάρτηση `nonzero` για να σχεδιάσετε την δομή του πίνακα και να δείτε τη μη-μηδενική δομή του.
- Χρησιμοποιείτε τη συνάρτηση `plotsolution` για να σχεδιάσετε την λύση του συστήματος, δηλαδή την λύση της διαφορικής εξίσωσης.
- Χρησιμοποιείτε την $norm(A * sol - f)$ για να υπολογίσετε το σφάλμα της λύσης σας.
- Χρησιμοποιείτε τις `tic`, `toc` για να μετρήσετε το χρόνο εκτέλεσης.

- Χρησιμοποιείτε τη συνάρτηση `jacobi` για να λύσετε ένα γραμμικό σύστημα με την μέθοδο του Jacobi.
- Χρησιμοποιείτε τη συνάρτηση `cg` για να λύσετε ένα γραμμικό σύστημα με την μέθοδο των Συζυγών Κλίσεων (Conjugate Gradients).
- Χρησιμοποιείτε `Cntl-C` για να σταματήσετε *απέλειωτες* δουλειές του Matlab.
- Για να αποφύγετε την περιττή αντιγραφή/επικόλληση στο κύριο script σας, γράψτε συναρτήσεις (μια για κάθε μέθοδο `lu-run`, `luinc-run`, `jacobi-run`, `gs-run`, `cg-run`). Κάθε μια από αυτές θα:
 1. Δημιουργεί τα a, f .
 2. Σχεδιάζει δομή του πίνακα αν $n_x, n_y \leq 5$.
 3. Λύνει το σύστημα με την κατάλληλη μέθοδο κάθε φορά.
 4. Μετρά χρόνο εκτέλεσης και σφάλμα προσέγγισης της λύσης.
 5. Εμφανίζει με κατάλληλα σχόλια το χρόνο εκτέλεσης, το σφάλμα και τον αριθμό των επαναλήψεων (αν πρόκειται για επαναληπτική μέθοδο).
 6. Σχεδιάζει την λύση.
- Για κάθε άσκηση καλέστε την αντίστοιχη συνάρτηση όσες φορές χρειάζεται και γράψτε τις παρατηρήσεις σας και τα σχόλια σας γύρω από τις γραφικές παραστάσεις, τα σφάλματα, τους χρόνους των λύσεων και τον αριθμό επαναλήψεων. Το κείμενο σας να είναι σε LaTeX.

ΑΣΚΗΣΗ 1. α) Καλέστε την `lu-run` για να κάνετε παραγοντοποίηση με την `lu`, και την `\` για να λύσετε τα τριγωνικά συστήματα που προκύπτουν, αν το αρχικό προέρχεται από $n_x = 3, n_y = 10, c_1 = 0, c_2 = 0, c = 0$.

β) Όμοια για $n_x = 10, n_y = 3, c_1 = 0, c_2 = 0, c = 0$.

γ) Όμοια για $n_x = 5, n_y = 5, c_1 = 0, c_2 = 0, c = 0$.

δ) Όμοια για $n_x = 5, n_y = 5, c_1 = 100, c_2 = 0, c = 0$.

ε) Όμοια για $n_x = 5, n_y = 5, c_1 = 0, c_2 = 100, c = 0$.

στ) Όμοια για $n_x = 5, n_y = 5, c_1 = 0, c_2 = 0, c = 100$.

Τι παρατηρείτε ως προς την ακρίβεια της λύσης σας;

ΑΣΚΗΣΗ 2. α) Όμοια για $n_x = 10, n_y = 10, c_1 = 0, c_2 = 0$ και $c = 0$.

β) Όμοια για $n_x = 20, n_y = 20, c_1 = 0, c_2 = 0, c = 0$ και $c = 100$.

γ) Όμοια για $n_x = 30, n_y = 30, c_1 = 0, c_2 = 0, c = 0$ και $c = 100$.

δ) Όμοια για $n_x = 40, n_y = 40, c_1 = 0, c_2 = 0, c = 0$ και $c = 100$.

ε) Όμοια για $n_x = 50, n_y = 50, c_1 = 0, c_2 = 0, c = 0$ και $c = 100$.

Τι παρατηρείτε ως προς την ακρίβεια της λύσης σας και ως προς το χρόνο εκτέλεσης;

ΑΣΚΗΣΗ 3. Δοκιμάστε να λύσετε τα τελευταία 5 συστήματα, της άσκησης 2, χρησιμοποιώντας την βασική ιδιότητα του A , ότι είναι αραιός. Μελετήστε την συνάρτηση `sparse` του Matlab και χρησιμοποιείστε την για μετατρέψετε και να αποθηκεύσετε τον A σε αραιή μορφή. Επαναλάβετε τις εντολές για τη λύση του συστήματος, μόνο που τώρα θα χρησιμοποιήσετε την `luinc` αντί της `lu` μέσα από την `luinc-run`. Τι παρατηρείτε ως προς το χρόνο εκτέλεσης;

Λαμβάνοντας υπόψη σας την διάσπαση του πίνακα A σε $A = D - L - U$ και τις επαναληπτικές μεθόδους **Jacobi** και **Gauss-Seidel** για την επίλυση γραμμικών συστημάτων προχωρείστε στις δύο τελευταίες ασκήσεις, έχοντας σαν αρχικό διάνυσμα το $x_0 = \mathbf{0}$, $\epsilon = .5e - 5$ και $maxiter = n_x * n_y$. Οι αλγόριθμοι περιγράφονται στον παρακάτω πίνακα.

Σε μορφή εξισώσεων	Σε μορφή πινάκων
Επαναληπτική μέθοδο Jacobi	
<pre> for $k = 1, \dots, maxiter$ $x_i^{(k+1)} = \frac{b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k)} - \sum_{j=i+1}^n a_{ij} x_j^{(k)}}{a_{ii}}, \quad i = 1, \dots, n$ if ($\ x^{(k+1)} - x^{(k)}\ _2 < \epsilon$) stop end </pre>	<pre> for $k = 1, \dots, maxiter$ $Dx^{(k+1)} = (L + U)x^{(k)} + b$ if ($\ x^{(k+1)} - x^{(k)}\ _2 < \epsilon$) stop end </pre>
Επαναληπτική μέθοδο Gauss-Seidel	
<pre> for $k = 1, \dots, maxiter$ $x_i^{(k+1)} = \frac{b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^n a_{ij} x_j^{(k)}}{a_{ii}}, \quad i = 1, \dots, n$ if ($\ x^{(k+1)} - x^{(k)}\ _2 < \epsilon$) stop end </pre>	<pre> for $k = 1, \dots, maxiter$ $(D - L)x^{(k+1)} = Ux^{(k)} + b$ if ($\ x^{(k+1)} - x^{(k)}\ _2 < \epsilon$) stop end </pre>

ΑΣΚΗΣΗ 4. Χρησιμοποιώντας πάλι την ιδιότητα του A , ότι είναι αραιός, καλέστε τη `jacobi`, η οποία θα υλοποιεί την επαναληπτική μέθοδο Jacobi, μέσα από την `jacobi-run`.

Δοκιμάστε να λύσετε τα τελευταία 5 συστήματα από την άσκηση 2. Τι παρατηρείτε ως προς την ακρίβεια της λύσης σας και ως προς το χρόνο εκτέλεσης και πλήθος επαναλήψεων;

ΑΣΚΗΣΗ 5. Τροποποιήστε την `jacobi.m` ώστε να υλοποιήσετε την επαναληπτική μέθοδο Gauss-Seidel μέσα στην `gs.m` την οποία θα καλέσετε κλασσικά μέσα από την `gs-run`.

Δοκιμάστε να λύσετε τα ίδια τελευταία 5 συστήματα. Τι παρατηρείτε ως προς την ακρίβεια της λύσης σας και ως προς το χρόνο εκτέλεσης και πλήθος επαναλήψεων;

ΑΣΚΗΣΗ 6. Καλέστε την `cg.m` μέσα από την `cg-run` για να χρησιμοποιήσετε την μέθοδο Συζηγών Κλίσεων(Conjugate Gradients) στην επίλυση των γραμμικών συστημάτων.

Δοκιμάστε να λύσετε τα ίδια τελευταία 5 συστήματα. Τι παρατηρείτε ως προς την ακρίβεια της λύσης σας και ως προς το χρόνο εκτέλεσης και πλήθος επαναλήψεων;

Υπενθυμίζουμε ότι η μέθοδος των Συζηγών Κλίσεων(Conjugate Gradients) μπορεί να δοθεί από τον αλγόριθμο:

$\begin{aligned} &\text{Αρχικοποίηση: } x_0 = 0 \Rightarrow r_0 = b - Ax_0 \\ &\text{Λύσε: } D\tilde{r}_0 = r_0 \Rightarrow p_0 = \tilde{r}_0 \\ &\text{for } k = 0, \dots \\ &\quad \alpha_k = \frac{(\tilde{r}_k, r_k)}{(p_k, Ap_k)} \\ &\quad x_{k+1} = x_k + \alpha_k p_k \\ &\quad r_{k+1} = r_k - \alpha_k Ap_k \\ &\quad D\tilde{r}_{k+1} = r_{k+1} \\ &\quad \text{if } (r_{k+1}, r_{k+1}) \leq \epsilon \\ &\quad \quad \text{return} \\ &\quad \beta_k = \frac{(\tilde{r}_{k+1}, r_{k+1})}{(\tilde{r}_k, r_k)} \\ &\quad p_{k+1} = \tilde{r}_{k+1} + \beta_k p_k \\ &\text{end} \end{aligned}$	}	Μέθοδος Συζηγών Κλίσεων.
--	---	--------------------------

Γενική υπόδειξη: Για να μελετήσετε τα αποτελέσματά σας και να γράψετε τις παρατηρήσεις σας, συγκεντρώστε τα στους παρακάτω πίνακες:

n_x	n_y	c_1	c_2	c	χρόνος	σφάλμα
3	10	0	0	0		
10	3	0	0	0		
5	5	0	0	0		
5	5	100	0	0		
5	5	0	100	0		
5	5	0	0	100		
10	10	0	0	0		
20	20	0	0	0		
30	30	0	0	0		
40	40	0	0	0		
50	50	0	0	0		
10	10	0	0	100		
20	20	0	0	100		
30	30	0	0	100		
40	40	0	0	100		
50	50	0	0	100		

Πίνακας 1. Αποτελέσματα με χρήση της `lu`.

	$n_x = n_y$	c	χρόνος	επαναλήψεις	σφάλμα	c	χρόνος	επαναλήψεις	σφάλμα
<i>luinc</i>	10	0				100			
	20	0				100			
	30	0				100			
	40	0				100			
	50	0				100			
<i>Jacobi</i>	10	0				100			
	20	0				100			
	30	0				100			
	40	0				100			
	50	0				100			
<i>G - S</i>	10	0				100			
	20	0				100			
	30	0				100			
	40	0				100			
	50	0				100			
<i>CG</i>	10	0				100			
	20	0				100			
	30	0				100			
	40	0				100			
	50	0				100			

Πίνακας 2. Αποτελέσματα με χρήση λογισμικού για αραιούς πίνακες και επαναληπτικές μεθόδους.

Καλή επιτυχία.