Αριθμητικη Επιλυση Μερικων Διαφορικων Εξισωσεων

1ο Εργαστηριο

9 Οκτωβρίου 2015

1 Πεπερασμένες Διαφορές

Έστω το πρόβλημα συνοριακών τιμών

$$\begin{cases} -u''(x) + q(x)u(x) = f(x), \quad x \in [a,b] \\ u(a) = u(b) = 0 \end{cases} \tag{1}$$

όπου $q(x) \geq 0$ για κάθε $x \in [a,b]$. Θεωρούμε ομοιόμορφο διαμερισμό του διαστήματος $[a,b]$ με βήμα $h = (b-a)/N$, τα σημεία x_i του οποίου δίνονται από τη σχέση

$$x_i = a + (i-1)h, \ i = 1, 2, \ldots, N+1,$$

όπου ξεκινάμε την αρίθμηση από 1 και όχι από 0, γιατί έχουμε υπ΄ όψιν την υλοποίηση της μεθόδου στο Matlab. Σκοπός μας είναι να υλοποιήσουμε μία αριθμητική μέθοδο η οποία να υπολογίζει μία προσεγγιστική λύση της (1) στα σημεία x_i της διαμέρισης, τις οποίες θα συμβολίσουμε με U_i, όπου $U_i \approx u(x_i), \ i = 1, \ldots, N+1$.

Η μέθοδος που θα χρησιμοποιήσουμε εδώ για τον υπολογισμό της προσεγγιστικής λύσης ονομάζεται *μέθοδος πεπερασμένων διαφορών* και βασίζεται στην προσέγγιση της παραγώγου από πηλίκα διαφορών, τα οποία προέρχονται από τα αναπτύγματα Taylor. Μια προσέγγιση της πρώτης παραγώγου της u στο σημείο x_i, είναι η

$$u'(x_i) \approx \frac{u\left(x_i + \frac{h}{2}\right) - u\left(x_i - \frac{h}{2}\right)}{h}, \tag{2}$$

η οποία είναι μία προσέγγιση βασισμένη στις εξισώσεις κεντρικών διαφορών. Χρησιμοποιώντας την ίδια τεχνική όπως στην (2), μια προσέγγιση της δεύτερης παραγώγου με τη χρήση κεντρικών διαφορών είναι η

$$u''(x_i) \approx \frac{u'\left(x_i + \frac{h}{2}\right) - u'\left(x_i - \frac{h}{2}\right)}{h}. \tag{3}$$

Με αντικατάσταση και λίγες πράξεις εύκολα επιβεβαιώνουμε ότι η δεύτερη παράγωγος μπορεί να εκφραστεί συναρτήσει της u ως

$$u''(x_i) \approx \frac{u(x_i+h) - 2u(x_i) + u(x_i-h)}{h^2}$$

$$= \frac{u(x_{i+1}) - 2u(x_i) + u(x_{i-1})}{h^2} \tag{4}$$

Έτσι λοιπόν, για ένα σημείο $x_i,\ i = 2, \ldots, N$, μπορούμε να αντικαταστήσουμε τον όρο της δεύτερης παραγώγου στην εξίσωση (1) με την έκφραση που δίνεται από την (4) και λαμβάνουμε

$$-\frac{u(x_{i+1}) - 2u(x_i) + u(x_{i-1})}{h^2} + q(x_i)u(x_i) = f(x_i),$$

ή ισοδύναμα

$$-\frac{u(x_{i+1}) - (2 + h^2 q(x_i))u(x_i) + u(x_{i-1})}{h^2} = f(x_i).$$

Αντικαθιστώντας με τις ζητούμενες ποσότητες U_i, καταλήγουμε στο *διακριτό σχήμα κεντρικών διαφορών*

$$-\frac{1}{h^2}\left(U_{i+1} - (2 + h^2 q(x_i))U_i + U_{i-1}\right) = f(x_i)$$

Εδώ πρέπει να σημειώσουμε πως ο λόγος για τον οποίον παραλείπουμε τα σημεία x_1 και x_{N+1} είναι ότι αν προσπαθήσουμε να υπολογίσουμε την προσέγγιση του u'' σε αυτά τα σημεία, θα χρειαστεί να υπολογίσουμε την τιμή $u(x_0)$ για την προσέγγιση στο x_1 και την $u(x_{N+2})$ για το x_{N+1}. Όμως τα σημεία x_0 και x_{N+2} δεν ανήκουν στη διαμέρισή μας, επομένως θα πρέπει να χειριστούμε το πρώτο και το τελευταίο σημείο ξεχωριστά. Στην περίπτωσή μας, οι συνοριακές συνθήκες μας διευκολύνουν, μιας και για $i = 1$ έχουμε

$$U_1 = u(a) = 0$$

και για το τελευταίο σημείο έχουμε επίσης

$$U_{N+1} = u(b) = 0,$$

οπότε δεν χρειάζεται να υπολογίσουμε τις προσεγγίσεις στα άκρα του διαστήματος.
Συνοψίζοντας λοιπόν, το αριθμητικό σχήμα κεντρικών πεπερασμένων διαφορών δίνεται από τις εξισώσεις

$$\begin{cases} U_1 & = 0 \\ -U_{i+1} + \left(2 + h^2 q(x_i)\right) U_i - U_{i-1} & = h^2 f(x_i), \quad i = 2, \ldots, N \\ U_{N+1} & = 0 \end{cases} \tag{5}$$

Είναι εύκολο να γράψει κανείς την εξίσωση (5) ως ένα γραμμικό σύστημα

$$\mathbb{K}U = F,$$

όπου στην συγκεκριμένη περίπτωση θα έχει τη μορφή

$$\underbrace{\begin{bmatrix} -1 & (2+h^2q(x_2)) & -1 & 0 & \cdots & 0 \\ 0 & -1 & (2+h^2q(x_3)) & -1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & -1 & (2+h^2q(x_N)) & -1 \end{bmatrix}}_{\mathbb{K}} \underbrace{\begin{bmatrix} U_2 \\ U_3 \\ \vdots \\ U_N \end{bmatrix}}_{U} = h^2 \underbrace{\begin{bmatrix} f(x_2) \\ f(x_3) \\ \vdots \\ f(x_N) \end{bmatrix}}_{F}$$

Είναι πλέον ξεκάθαρο ότι αν μας δοθούν οι συναρτήσεις $q(x)$ και $f(x)$ και ξέρουμε πώς να διακριτοποιήσουμε το διάστημα $[a, b]$, είμαστε σε θέση να υπολογίσουμε μια προσέγγιση της λύσης της (1) με τη μέθοδο πεπερασμένων διαφορών. Αυτό θα γίνει πιο σαφές στο παρακάτω παράδειγμα.

1.1 Υλοποίηση σε Matlab

Θα θεωρήσουμε το πρόβλημα

$$\begin{cases} -u''(x) + u(x) = \sin(\pi x), & x \in [0, 1], \\ u(0) = u(1) = 0 \end{cases} \tag{6}$$

δηλαδή έχουμε επιλέξει $q(x) = 1$ και $f(x) = \sin(\pi x)$. Για να υλοποιήσουμε τη μέθοδο στη Matlab, θα πρέπει αρχικά να κατασκευάσουμε με κάποιον τρόπο τις συναρτήσεις $q(x)$ και $f(x)$. Αυτό μπορεί να γίνει εύκολα με τη δημιουργία δύο αρχείων με τα αντίστοιχα ονόματα, μέσα στα οποία θα δηλώσουμε τον τύπο της κάθε συνάρτησης. Για να δημιουργήσουμε το αρχείο `f.m` μπορούμε να γράψουμε στη Matlab την εντολή

```
>> edit f.m
```

και θα ανοίξει ο επεξεργαστής κειμένου όπου μπορούμε να γράψουμε τον κώδικα της συνάρτησης. Συγκεκριμένα, για τη συνάρτηση f ο κώδικας είναι ο παρακάτω.

Αρχείο: f.m

```
function v = f(x)
    v = sin(pi*x);
end
```

Με τον ίδιο τρόπο δημιουργούμε και το αρχείο `q.m` στο οποίο θα ορίσουμε τη σταθερή συνάρτηση $q(x) = 1$.

Αρχείο: q.m

```
function v = q(x)
    v = ones(size(x));
end
```

Εδώ πρέπει να προσέξει κανείς την ιδιαίτερη σύνταξη της εντολής που χρησιμοποιήσαμε. Το `x` που λαμβάνουμε σαν είσοδο μπορεί να είναι είτε αριθμός, είτε διάνυσμα. Σε οποιαδήποτε περίπτωση, η συνάρτηση `q` όπως την επιλέξαμε πρέπει να μας επιστρέφει 1. Στην περίπτωση που το `x` είναι αριθμός, περιμένουμε να πάρουμε ως αποτέλεσμα έναν αριθμό, αλλά όταν το `x` είναι διάνυσμα, η `q` θα έπρεπε να μας επιστρέφει μια τιμή για κάθε στοιχείο του `x`, και αυτή η τιμή θα είναι πάντα 1. Επομένως αρκεί να επιστρέψουμε ένα σύνολο από μονάδες τα οποία έχουν τη διάσταση του `x` (αυτό λειτουργεί καλά ακόμα και όταν το `x` είναι απλά αριθμός).
Τώρα είμαστε έτοιμοι να δημιουργήσουμε το κυρίως αρχείο στο οποίο θα υλοποιήσουμε τη μέθοδό μας. Θα ονομάσουμε το αρχείο `fd1d.m` και θα ορίσουμε μια συνάρτηση με το ίδιο όνομα στο εσωτερικό του. Η συνάρτηση αυτή θα δέχεται σαν είσοδο τις παραμέτρους `a`, `b` και `N`, και θα επιστρέφει σαν έξοδο το διάνυσμα με τις προσεγγίσεις `U`, υπολογισμένο με τη μέθοδο πεπερασμένων διαφορών που περιγράψαμε, καθώς και την διακριτοποίηση `x` του διαστήματος $[a, b]$. Αρχικά λοιπόν το αρχείο έχει αυτή τη μορφή:

Αρχείο: fd1d.m

```
function [U, x] = fd1d(a, b, N)
```

Μπορούμε τώρα να ορίσουμε μια διακριτοποίηση του διαστήματος $[a, b]$ ως εξής

Αρχείο: fd1d.m

```
h = (b-a)/N; % Compute the step size
x = a:h:b;   % Compute the discretization of the interval [a, b]
```

Πρέπει να τονίσουμε ότι η παράμετρος N ορίζει το πλήθος των υποδιαστημάτων του $[a, b]$, το οποίο σημαίνει ότι το διάνυσμα `x` θα έχει $N + 1$ σημεία, και όχι N.
Ως επόμενο βήμα, θα κατασκευάσουμε τον πίνακα του γραμμικού συστήματος που περιγράψαμε στην προηγούμενη ενότητα. Πρέπει να θυμηθούμε ότι ο πίνακας έχει διάσταση κατά 2 μικρότερη από τη διάσταση της διακριτοποίησης, καθώς οι τιμές στα δύο άκρα του διαστήματος είναι γνωστές και θα χειριστούν ξεχωριστά. Ο πίνακας $\mathbb{K}$ λοιπόν έχει διάσταση $(N-1)\times(N-1)$, και επίσης είναι τριδιαγώνιος και συμμετρικός, όπως είδαμε. Μπορούμε να εκμεταλλευτούμε την ειδική μορφή του πίνακα και να τον κατασκευάσουμε πιο εύκολα με τις δυνατότητες που μας προσφέρει η Matlab. Αρχικά, ας παρατηρήσουμε ότι τα διαγώνια στοιχεία του $\mathbb{K}$ είναι $N-1$ στο πλήθος, ενώ τα στοιχεία πάνω και κάτω από τη διαγώνιο είναι $N-2$ και είναι όλα ίσα με -1.
Εύκολα αντιλαμβάνεται κανείς ότι αρκούν 2 μόλις διανύσματα για να κατασκευάσουμε τον πίνακα $\mathbb{K}$. Θα κατασκευάσουμε το διάνυσμα που περιέχει τα διαγώνια στοιχεία, καθώς και ένα διάνυσμα με το κατάλληλο μήκος το οποίο θα περιέχει τα στοιχεία -1, και χρησιμοποιώντας την εντολή `diag` θα δημιουργήσουμε τον $\mathbb{K}$. Η διαδικασία είναι η εξής:

Αρχείο: fd1d.m

```
dd = 2 + h*h*q(x(2:end-1));      % Compute the diagonal values
d1 = -1 * ones(N-2, 1);          % Compute the -1 vector
K = diag(d1, -1) + diag(dd) + diag(d1, 1); % Construct K
```

Παρατηρήστε ότι για τον υπολογισμό των τιμών $q(x)$ δεν χρησιμοποιήσαμε ολόκληρο το διάνυσμα x αλλά μόνο από το δεύτερο μέχρι και το προτελευταίο στοιχείο. Η λέξη `end` μπορεί να χρησιμοποιηθεί μονάχα για να δείξει στοιχείο σε διάνυσμα ή πίνακα, και δείχνει το τελευταίο στοιχείο, επομένως το `end-1` δείχνει το προτελευταίο. Ο λόγος για τον οποίο πήραμε τιμές μόνο στα συγκεκριμένα στοιχεία είναι και πάλι επειδή θα χειριστούμε τις τιμές στα άκρα με ξεχωριστό τρόπο.
Το μόνο που μας λείπει πλέον είναι το διάνυσμα του δεξιού μέλους, και για να το κατασκευάσουμε δίνουμε τις εξής εντολές.

Αρχείο: fd1d.m

```
F = h*h*f(x(2:end-1));  % Compute the values of the right-hand side
F = F(:);               % Take care of orientation
```

Και εδώ υπολογίζουμε τις τιμές της συνάρτησης f μόνο στα κατάλληλα στοιχεία του `x`, και επίσης πρέπει να φροντίσουμε το διάνυσμα να έχει τις κατάλληλες διαστάσεις. Θυμηθείτε από τη Γραμμική Άλγεβρα ότι για να ορίζεται ο πολλαπλασιασμός πίνακα με διάνυσμα, πρέπει οι διαστάσεις να συμφωνούν, και μάλιστα το διάνυσμα πρέπει να είναι σε μορφή στήλης. Αν έχουμε ένα οποιοδήποτε διάνυσμα y στη Matlab, αλλά δεν ξέρουμε αν είναι γραμμή ή στήλη, η εντολή $y(:)$ το επιστρέφει σαν διάνυσμα στήλη.
Για να λύσουμε λοιπόν το γραμμικό σύστημα αρκεί να δώσουμε την εντολή

Αρχείο: fd1d.m

```
U = K \ F;  % Solve the linear system
```

και στο U έχουμε τις προσεγγίσεις τις λύσης στα σημεία $x_2, x_3, \ldots, x_N$. Πρέπει να επιβάλλουμε τις συνθήκες στα άκρα, και ο τρόπος για να το κάνουμε είναι ο εξής.

Αρχείο: fd1d.m

```
U = [0; U; 0];  % Impose the boundary conditions
```

Επειδή το διάνυσμα U είναι σε μορφή στήλης, πρέπει να προσέξουμε τη σύνταξη της εντολής η οποία απλά προσθέτει ένα μηδενικό στην αρχή και στο τέλος του U.
Αυτό ήταν! Ολόκληρος ο κώδικας ακολουθεί στο επόμενο πλαίσιο.

Αρχείο: fd1d.m

```
function [U, x] = fd1d(a, b, N)

h = (b-a)/N; % Compute the step size
x = a:h:b;   % Compute the discretization of the interval [a, b]

dd = 2 + h*h*q(x(2:end-1));      % Compute the diagonal values
d1 = -1 * ones(N-2, 1);          % Compute the -1 vector
K = diag(d1, -1) + diag(dd) + diag(d1, 1); % Construct K

F = h*h*f(x(2:end-1));  % Compute the values of the right-hand side
F = F(:);               % Take care of orientation

U = K \ F;       % Solve the linear system
U = [0; U; 0];   % Impose the boundary conditions

end
```

Αν τώρα στη Matlab εκτελέσουμε την εντολή

```
>> [sol, x] = fd1d(0, 1, 100);
```

θα πάρουμε πίσω το διάνυσμα sol διάστασης 101×1 το οποίο περιέχει τις προσεγγίσεις της πραγματικής λύσης του προβλήματος, καθώς επίσης και το διάνυσμα `x` επίσης διάστασης 101×1 το οποιο περιέχει τα σημεία της διακριτοποίησης του $[a, b]$. Πρέπει όμως με κάποιον τρόπο να ελέγξουμε πόσο σωστά λειτουργεί ο αλγόριθμός μας, και για αυτό το σκοπό θα δημιουργήσουμε τη συνάρτηση exact η οποία θα μας δίνει τις τιμές για την ακριβή λύση του προβλήματος. Σημειώνουμε ότι η ακριβής λύση του προβλήματος (6) είναι η συνάρτηση

$$u(x) = \frac{\sin(\pi x)}{1 + \pi^2}.$$

Δημιουργούμε το αρχείο exact.m και γραφουμε σε αυτό

Αρχείο: exact.m

```
function v = exact(x)
    v = sin(pi*x)/(1+pi*pi);
end
```

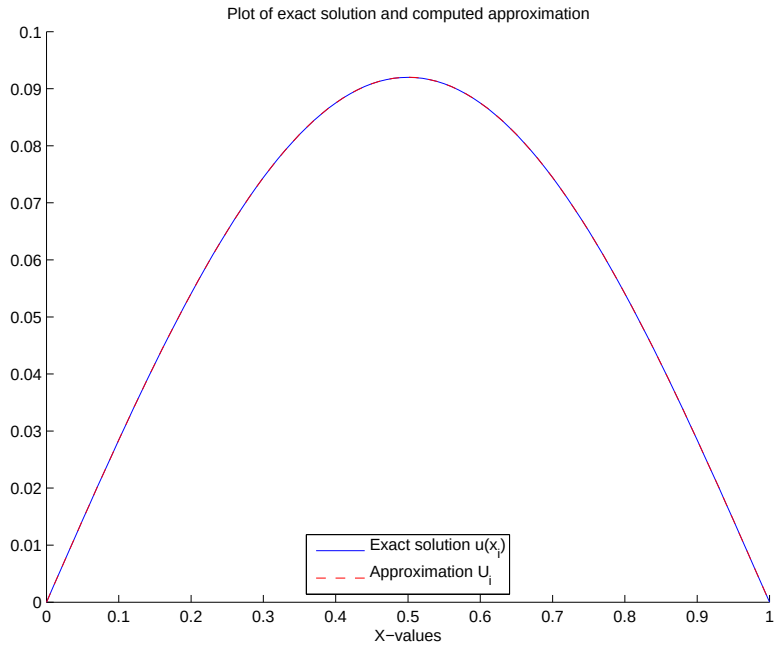
Μπορούμε τώρα να χρησιμοποιήσουμε την διακριτοποίηση `x` για να υπολογίσουμε τις τιμές της ακριβούς λύσης στα ίδια σημεία στα οποία υπολογίσαμε προσεγγίσεις, έτσι ώστε να μπορούμε να τα συγκρίνουμε άμεσα. Εκτελούμε την εντολή

```
>> ex = exact(x);
```

και στο διάνυσμα `ex`, διάστασης 101×1, έχουμε τις τιμές τις ακριβούς λύσης. Μπορούμε τώρα να συγκρίνουμε τα αποτελέσματα δημιουργώντας ένα κοινό γράφημα.

```
>> plot(x, ex, 'b');
>> hold all
>> plot(x, sol, 'r--');
>> legend('Exact solution', 'Approximation')
```

Πρέπει να προκύπτει ένα γράφημα στο οποίο η ακριβής λύση και η προσέγγιση σχεδόν δεν ξεχωρίζουν με το μάτι, όπως φαίνεται στο Σχήμα 1.



Σχήμα 1: Ακριβής λύση της (6) (μπλε γραμμή) και η προσεγγιστική λύση U (κόκκινη γραμμή), για $N = 100$.

Μπορούμε όμως να υπολογίσουμε τη διαφορά ή αλλιώς το σφάλμα της προσέγγισής μας μέσω του υπολογισμού της νόρμας της διαφοράς.

```
>> norm(ex' - sol, inf)
ans =
    6.8708e-06
```

Η δεύτερη παράμετρος στην εντολή norm ορίζει ποιά νόρμα θέλουμε να υπολογιστεί. Οι επιλογές που έχουμε είναι η νόρμα 1, 2 και ∞ (την οποία έχουμε διαλέξει εδώ).

Η ποσότητα της νόρμας είναι μια καλή ένδειξη για την ποιότητα της προσέγγισής μας, αλλά αυτό που μας ενδιαφέρει ακόμα περισσότερο είναι να ελέγξουμε την *τάξη ακρίβειας* ή *τάξη σύγκλισης* της μεθόδου. Η τάξη σύγκλισης είναι ένας αριθμός που χαρακτηρίζει την ίδια τη μέθοδο και μας δίνει πληροφορίες για την ποιότητα της προσέγγισης ανάλογα με το βήμα h που θα διαλέξουμε. Μια απλή περιγραφή της έννοιας είναι η εξής: υποθέστε ότι έχουμε εκτελέσει τη μέθοδό μας με

N σημεία, και έχουμε πάρει μια προσέγγιση η οποία έχει σφάλμα ε_N. Αν εκτελέσουμε ξανά τη μέθοδο, αυτή τη φορά με $\tilde{N} \neq N$ σημεία, θα πάρουμε ένα δεύτερο σφάλμα $\varepsilon_{\tilde{N}}$. Τι σχέση έχουν οι ποσότητες αυτές;

Όπως ξέρουμε γενικά, λέμε ότι μια αριθμητική μέθοδος έχει τάξη ακρίβειας p όταν υπάρχει κάποια σταθερά C ανεξάρτητη από το h και το N έτσι ώστε να ισχύει

$$\varepsilon = Ch^p$$

Για λόγους απλότητας θα υποθέσουμε ότι είναι $\tilde{N} = 2N$, επομένως αν h είναι το βήμα διακριτοποίησης που αντιστοιχεί στο N, τότε το βήμα διακριτοποίησης για το $\tilde{N}$ είναι $h/2$. Θεωρούμε τώρα το πηλίκο των σφαλμάτων και έχουμε

$$\frac{\varepsilon_N}{\varepsilon_{2N}} \approx \frac{Ch^p}{C\left(\frac{h}{2}\right)^p} = 2^p \Rightarrow p \approx \log_2\left(\frac{\varepsilon_N}{\varepsilon_{2N}}\right) = \frac{\log\left(\frac{\varepsilon_N}{\varepsilon_{2N}}\right)}{\log(2)}$$

Εδώ $\log_2$ συμβολίζει το λογάριθμο με βάση 2, ενώ log είναι ο λογάριθμος στην αγαπημένη σας βάση (θυμηθείτε τους κανόνες αλλαγής βάσης λογαρίθμων). Επίσης, αυτή είναι μια ειδική περίπτωση στην οποία $\tilde{N} = 2N$. Στη γενική του μορφή, ο τύπος είναι

$$p \approx \frac{\log\left(\frac{\varepsilon_N}{\varepsilon_{\tilde{N}}}\right)}{\log\left(\frac{\tilde{N}}{N}\right)}$$

Τώρα λοιπόν, αφού ξέρουμε πώς να εκτιμήσουμε αριθμητικά την τάξη ακρίβειας της μεθόδου πεπερασμένων διαφορών, μπορούμε να την προσεγγίσουμε. Αρχικά θα πρέπει να υπολογίσουμε δύο προσεγγίσεις της λύσης, με διαφορετικό πλήθος σημείων. Ακολουθεί ενδεικτικός κώδικας με το οποίο μπορεί να γίνει αυτό.

Αρχείο: fd1dorder.m

```
% Number of intervals for each execution
N1 = 100;
N2 = 200;

% Find error with N1+1 points
[U1, x1] = fd1d(0, 1, N1);
ex1 = exact(x1);
err1 = norm(ex1' - U1, inf)

% Find error with N2+1 points
[U2, x2] = fd1d(0, 1, N2);
ex2 = exact(x2);
err2 = norm(ex2' - U2, inf);
```

Έτσι, σύμφωνα με όσα είπαμε, μια εκτίμηση για την τάξη ακρίβειας της μεθόδου είναι η ποσότητα

Αρχείο: fd1dorder.m

```
p = log(err1/err2)/log(N2/N1);
fprintf(1, 'Estimated method order: %g \n', p);
```

Η τελευταία εντολή απλά αναλαμβάνει την εμφάνιση κατάλληλου μηνύματος στην οθόνη και στην περίπτωσή μας, θα τυπώσει

```
Estimated method order:  2.00005
```

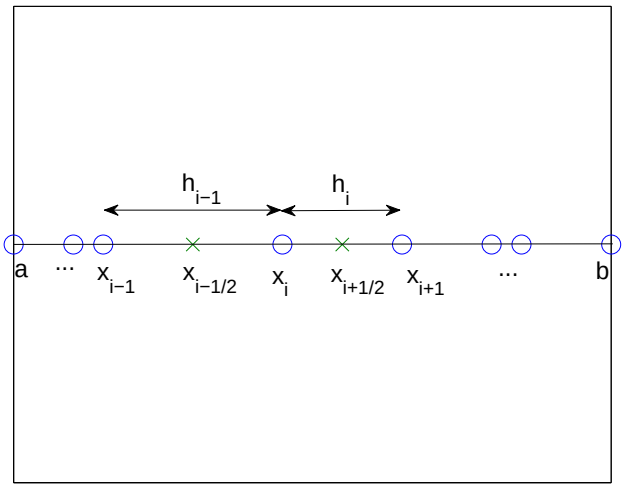
κάτι που μας υποδεικνύει πως η τάξη ακρίβειας της μεθόδου μας είναι 2.

2 Μη-ομοιόμορφος διαμερισμός

Έστω το πρόβλημα δύο σημείων:

$$\begin{cases} -u'' + qu = f, & x \in [a,b] \\ u(a) = u(b) = 0, \end{cases} \tag{7}$$

αλλά τώρα θα θεωρήσουμε ένα διαμερισμό του $[a,b]$, $a = x_1 < x_2 < \ldots < x_N < x_{N+1} = b$, που δεν θα είναι κατ΄ ανάγκην ομοιόμορφος. Θα συμβολίζουμε $h_i := x_{i+1} - x_i$, $i = 1, 2, \ldots, N$, $U_i \approx u(x_i)$, και με $x_{i-1/2}$, $x_{i+1/2}$, τα μέσα των διαστημάτων $[x_{i-1}, x_i]$, $[x_i, x_{i+1}]$, αντίστοιχα (βλ. Σχήμα 2).



Σχήμα 2: Αναπαράσταση του μη-ομοιόμορφου διαμερισμού.

Τότε η προσέγγιση της δευτέρας παραγώγου $u''(x_i)$ μπορεί να γίνει ως εξής:

$$u''(x_i) = (u'(x_i))' \approx \frac{u'(x_{i+1/2}) - u'(x_{i-1/2})}{\frac{h_i}{2} + \frac{h_{i-1}}{2}} \approx \frac{\frac{u(x_{i+1})-u(x_i)}{h_i} - \frac{u(x_i)-u(x_{i-1})}{h_{i-1}}}{\frac{1}{2}(h_i + h_{i-1})}.$$

Έτσι καταλήγουμε στο ακόλουθο σχήμα πεπερασμένων διαφορών για την επίλυση του (7):

$$-\frac{2}{h_{i-1}+h_i}\left(\frac{U_{i+1}-U_i}{h_i}-\frac{U_i-U_{i-1}}{h_{i-1}}\right)+q(x_i)U_i=f(x_i),\quad i=2,3,\ldots,N, \tag{8}$$

όπου λόγω των συν. συνθηκών Dirichlet έχουμε ότι $U_1=U_{N+1}=0$.
Πολλαπλασιάζουμε την (8) με $\frac{h_{i-1}+h_i}{2}$ και καταλήγουμε στην ισοδύναμη μορφή:

$$-\frac{1}{h_{i-1}}U_{i-1}+\left(\frac{1}{h_{i-1}}+\frac{1}{h_i}+\frac{h_{i-1}+h_i}{2}q(x_i)\right)U_i-\frac{1}{h_i}U_{i+1}=\frac{h_{i-1}+h_i}{2}f(x_i), \tag{9}$$

για $i=2,3,\ldots,N$, με $U_1=U_{N+1}=0$. (Παρατηρήστε ότι αν $h_i=h:=(b-a)/N$, για $i=1,2,\ldots,N$, δηλαδή για ομοιόμορφο διαμερισμό, καταλήγουμε στο γνωστό μας σχήμα πεπερασμένων διαφορών με τρία σημεία.)
Η (9) μπορεί επίσης να γραφεί στη μορφή του ακόλουθου γραμμικού συστήματος:

$$\mathbb{K}U=F, \tag{10}$$

όπου

$$\mathbb{K}=\begin{pmatrix} \frac{1}{h_1}+\frac{1}{h_2}+\frac{h_1+h_2}{2}q(x_2) & -\frac{1}{h_2} & 0 & \ldots & 0 \\ -\frac{1}{h_2} & \frac{1}{h_2}+\frac{1}{h_3}+\frac{h_2+h_3}{2}q(x_3) & -\frac{1}{h_3} & 0 & 0 \\ & & & & \\ & & \ddots & \ddots & \ddots \\ & & & & \\ 0 & \ldots & 0 & -\frac{1}{h_{N-1}} & \frac{1}{h_{N-1}}+\frac{1}{h_N}+\frac{h_{N-1}+h_N}{2}q(x_N) \end{pmatrix},$$

$U=(U_2,\ldots,U_N)^T$ και $F=\left(\frac{h_1+h_2}{2}f(x_2),\frac{h_2+h_3}{2}f(x_3),\ldots,\frac{h_{N-1}+h_N}{2}f(x_N)\right)^T$.

Ένα παράδειγμα. Θεωρούμε το πρόβλημα (7) με $[a,b]=[0,4]$, $q(x):=4$,
$f(x):=16\pi\left(\pi\sin(4\pi x)+\cos(4\pi x)\right)\mathrm{e}^{-2x}$. Τότε η ακριβής λύση είναι: $u(x)=\sin(4\pi x)\,\mathrm{e}^{-2x}$.
Όπως και στο προηγούμενο παράδειγμα, δημιουργούμε τα απαραίτητα αρχεία στα οποία ορίζουμε τις αντίστοιχες συναρτήσεις.

Αρχείο: q.m

```
function v = q(x)
    v = 4.0*ones(size(x));
end
```

Αρχείο: f.m

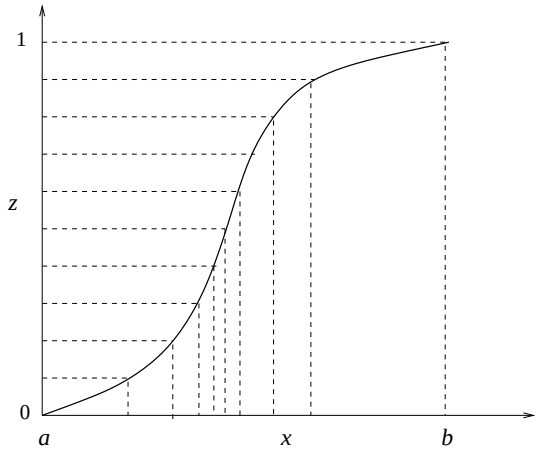
```
function v = f(x)
    v = 16*pi*(pi*sin(4*pi*x) + cos(4*pi*x)).*exp(-2*x);
end
```

Αρχείο: exact.m

```
function v = exact(x)
    v = sin(4*pi*x).*exp(-2*x);
end
```

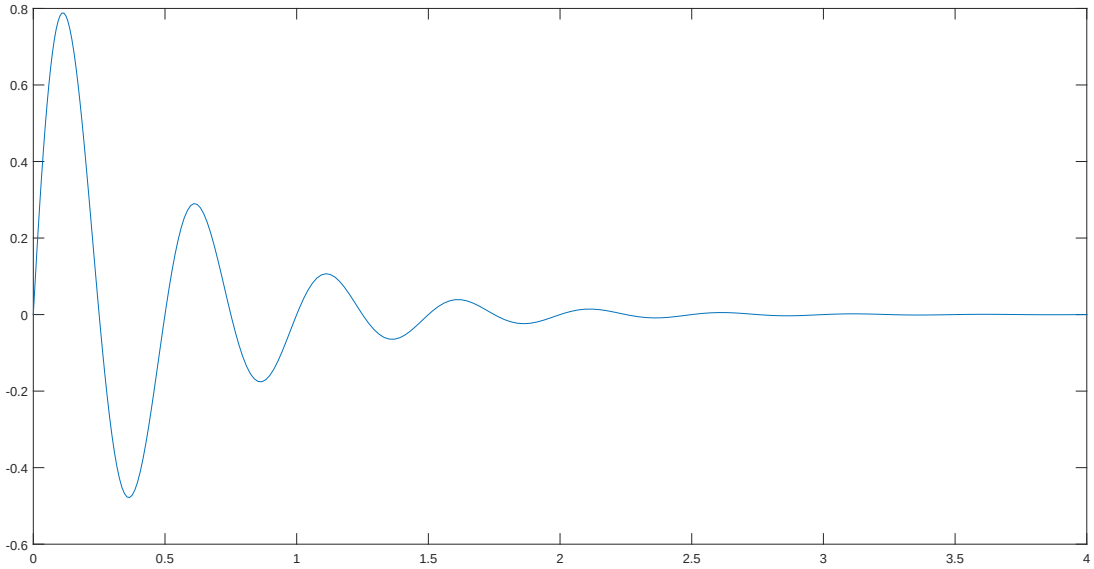
Προσέξτε τη χρήση του τελεστή «τελεία» ο οποίος αντί για πολλαπλασιασμό διανυσμάτων με την κλασσική έννοια προκαλεί πολλαπλασιασμό κατά συνιστώσα. Αν η μεταβλητή εισόδου x είναι ένα διάνυσμα, είναι απαραίτητο να λάβουμε υπόψη μας τέτοιες λεπτομέρειες. Στην περίπτωση που το x είναι απλά ένας αριθμός, ο τελεστής «τελεία» δεν αλλάζει τίποτα.

Προκειμένου να προχωρήσουμε, θα πρέπει να κατασκευάσουμε έναν μη ομοιόμορφο διαμερισμό του διαστήματος στο οποίο θα δουλέψουμε. Ένας τρόπος είναι ο ακόλουθος: Ξεκινάμε από έναν ομοιόμορφο διαμερισμό του [0,1], $z_i = (i-1)h$, $i = 1, 2, \ldots, N+1$, όπου $h = 1/N$, και στη συνέχεια χρησιμοποιούμε κάποια κατάλληλη απεικόνιση $\mathcal{X}(z)$ για να ορίσουμε τους κόμβους του μη ομοιόμορφου διαμερισμού $x_i := \mathcal{X}(z_i)$. Στο Σχήμα 3 έχουμε τοποθετήσει στον κάθετο άξονα τα z και στον οριζόντιο τα x. (Με αυτή την επιλογή αξόνων στο Σχήμα 3 ουσιαστικά δίνεται η γραφική παράσταση της αντίστροφης συνάρτησης $z = \mathcal{X}^{-1}(x)$.) Παρατηρήστε πώς τα ομοιόμορφα κατανεμημένα σημεία στον άξονα των z αντιστοιχούν στα μη ομοιόμορφα κατανεμημένα σημεία του άξονα των x.



Σχήμα 3

Για να καταλάβουμε τι πρέπει να κάνουμε, παραθέτουμε τη γραφική παράσταση της ακριβους λύσης του προβλήματος στο διάστημα [0, 4].

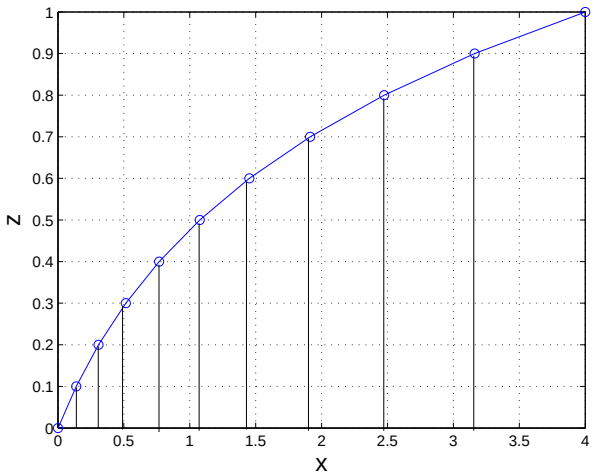


Σχήμα 4

Από το Σχήμα 4 γίνεται σαφές ότι η συνάρτηση που μελετάμε παρουσιάζει περισσότερες αλλαγές στο αριστερό κομμάτι του άξονα, κοντά στο 0. Κοντά στο 4, από την άλλη, η συμπεριφορά της είναι πολύ πιο ομαλή. Αυτό σημαίνει ότι θα ήταν ιδανικό να πάρουμε περισσότερα σημεία στο αριστερό κομμάτι του άξονα των x, και λιγότερα στο δεξί. Αυτό μπορεί να επιτευχθεί π.χ. με τη βοήθεια της συνάρτησης

$$\mathcal{X}(z) = (b-a)\frac{\mathrm{e}^{kz}-1}{\mathrm{e}^{k}-1} + a, \quad z \in [0,1], \tag{11}$$

όπου k είναι κατάλληλος συντελεστής ενίσχυσης της πύκνωσης. Στο Σχήμα 5 απεικονίζεται ο τρόπος λειτουργίας της (11) στο διάστημα [0,4] με $k = 2$.



Σχήμα 5

Για την υλοποίηση της λογικής αυτής θα χρησιμοποιήσουμε την έτοιμη συνάρτηση xgrid το περιεχόμενο της οποίας ακολουθεί.

Αρχείο: xgrid.m

```
function x = xgrid(ax,bx,m,gridchoice)
%
% Specify grid points in space for solving a 2-point boundary value problem
% or time-dependent PDE in one space dimension.
%
% Grid has m-1 interior points on the interval [ax, bx].
% gridchoice specifies the type of grid (see below).
% m+1 grid points (including boundaries) are returned in x.
%
% This file is a modified version of the original function taken from
% http://www.amath.washington.edu/~rjl/fdmbook/  (2007)

z = linspace(0, 1, m+1)';   % uniform grid in [0,1]

switch gridchoice

  case 'uniform'
     x = ax + (bx-ax)*z;

  case 'rtlayer'
     % Clustered near right boundary
     x =  ax + (bx-ax) * ((exp(-2*z) - 1)/(exp(-2)-1));

  case 'ltlayer'
     % Clustered near left boundary
     x =  ax + (bx-ax) * ((exp(2*z) - 1)/(exp(2)-1));

  case 'random'
     x = ax + (bx-ax)*sort(rand(m+1,1));
     x(1) = ax;
     x(m+1) = bx;

  case 'chebyshev'
     % Chebyshev extreme points
     x = ax + (bx-ax) * 0.5*(1 + cos(pi*(1-z)));
  end
```

Η συνάρτηση υποστηρίζει τη δημιουργία ομοιόμορφων και μη διαμερισμών στο ζητούμενο διάστημα $[ax, bx]$ με m υποδιαστήματα. Η παράμετρος gridchoice ορίζει τον τύπο του διαμερισμού, και οι τιμές που μπορούμε να δώσουμε είναι:

- 'uniform': ομοιόμορφος διαμερισμός
- 'rtlayer': μη ομοιόμορφος διαμερισμός με συγκέντρωση δεξιά
- 'ltlayer': μη ομοιόμορφος διαμερισμός με συγκέντρωση αριστερά
- 'random': τυχαίος διαμερισμός

- 'chebyshev': διαμερισμός τύπου Chebyshev (πυκνός στα άκρα και αραιός στη μέση)

Στην περίπτωσή μας λοιπόν η κατάλληλη χρήση της συνάρτησης είναι με την εντολή

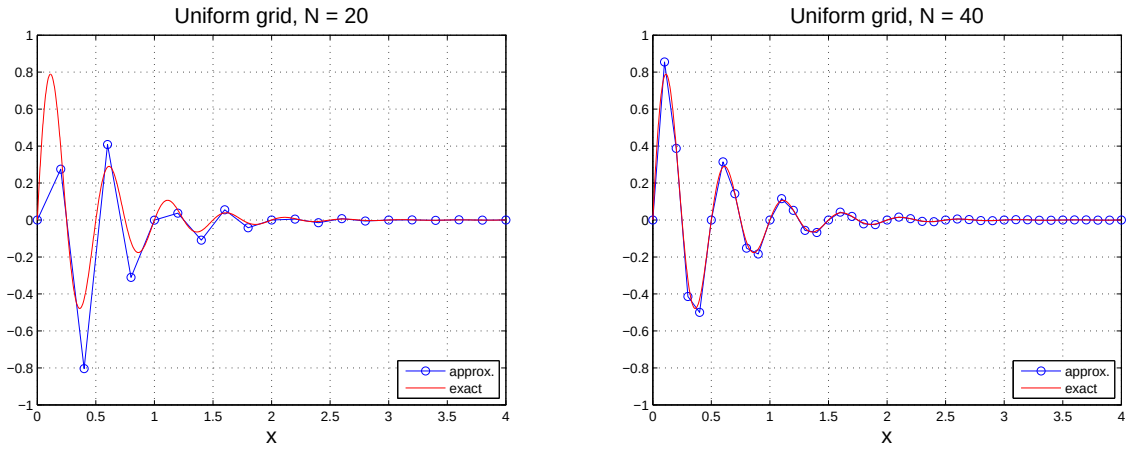
```
>> x = xgrid(0, 4, N, 'ltlayer');
```

όπου N το πλήθος των υποδιαστημάτων που θέλουμε στο διάστημα $[0, 4]$.
Μπορούμε τώρα να περάσουμε στην υλοποίηση του κεντρικού αρχείου για το παράδειγμά μας, το οποίο όπως και πριν έχουμε ονομάσει fd1d.m

Αρχείο: fd1d.m

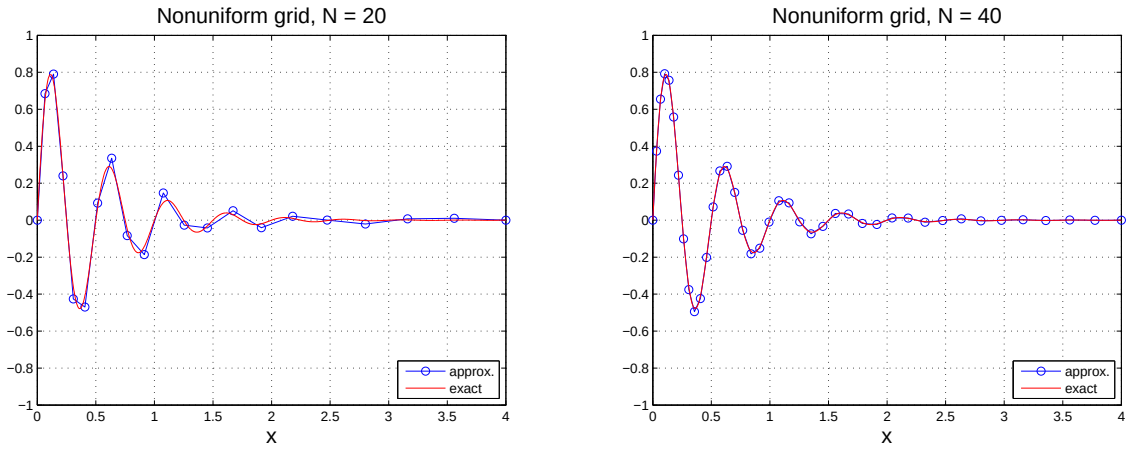
```
function [U, x] = fd1d(a, b, N, grid_type)

    % Compute the discretization of the interval [a, b]
    x = xgrid(a, b, N, grid_type);

    % Compute the step sizes
    h = zeros(N, 1);
    for i = 1:N
        h(i) = x(i+1) - x(i);
    end

    % Compute diagonal entries for K
    dd = zeros(N-1, 1);
    for i = 1:N-1
       dd(i) = 1/h(i) + 1/h(i+1) + (h(i)+h(i+1))*q(x(i+1))/2;
    end

    % Compute off-diagonal entries for K
    d1 = zeros(N-2, 1);
    for i = 1:N-2
       d1(i) = -1/h(i+1);
    end

    % Compute right-hand side
    F = zeros(N-1, 1);
    for i = 1:N-1
       F(i) = (h(i)+h(i+1))*f(x(i+1))/2;
    end

    % Construct the coefficients matrix A
    K = diag(d1, -1) + diag(dd) + diag(d1, 1);

    % Solve the linear system and impose boundary conditions
    U = K \ F;
    U = [0; U; 0];
end
```

Όπως μπορείτε να παρατηρήσετε, αυτή η μορφή του αρχείου είναι αρκετά διαφορετική από αυτήν που είχαμε στο προηγούμενο παράδειγμα. Η συνάρτηση δέχεται ένα παραπάνω όρισμα το οποίο θα μας επιρέψει να συγκρίνουμε τα αποτελέσματα της μεθόδου χρησιμοποιώντας ομοιόμορφο και μη-ομοιόμορφο διαμερισμό. Επιπλέον, οι τιμές h_i υπολογίζονται επαναληπτικά, και η κατασκευή του πίνακα $\mathbb{K}$ επίσης απαιτεί λίγη παραπάνω δουλειά. Παρατηρήστε όμως ότι το κομμάτι της επίλυσης και της επιβολής των συνοριακών συνθηκών έχει μείνει αμετάβλητο.
Στο Σχήμα 6 απεικονίζονται η ακριβής λύση και η προσεγγιστική λύση με τη μέθοδο πεπερασμένων διαφορών για ομοιόμορφο διαμερισμό του [0,4] α) με $N = 20$ και β) με $N = 40$ υποδιαστήματα. Όπως ήταν αναμενόμενο, ακόμα και με διπλάσιο πλήθος σημείων δεν πετυχαίνουμε κάτι πολύ καλύτερο, και αυτό οφείλεται στη φύση του προβλήματος που λύνουμε. Πρέπει να φροντίσουμε λοιπόν ώστε ο (ομοιόμορφος) διαμερισμός μας να είναι αρκετά λεπτός έτσι ώστε να να περιέχονται αρκετά σημεία στο διάστημα που η λύση μας μεταβάλλεται απότομα, το οποίο σημαίνει ότι θα κάνουμε περισσότερες πράξεις και στο διάστημα στο οποίο δεν χρειάζεται.



Σχήμα 6: Ομοιόμορφος διαμερισμός, $N = 20$ (αριστερά) και $N = 40$ (δεξιά).

Χρησιμοποιώντας τώρα το μη ομοιόμορφο διαμερισμό, αναδιατάσσουμε τους κόμβους της διαμέρισης ώστε να περιέχονται περισσότερα σημεία στο «δύσκολο» διάστημα και λιγότερα στο «εύκολο». Στο Σχήμα 7 απεικονίζονται η ακριβής λύση και η προσεγγιστική λύση με τη μέθοδο πεπερασμένων διαφορών για μη ομοιόμορφο διαμερισμό του [0,4], α) με $N = 20$ και β) με $N = 40$ υποδιαστήματα. Παρατηρήστε ότι οι κόμβοι πυκνώνουν όσο πλησιάζουμε προς το αριστερό άκρο του διαστήματος, και ότι τώρα η προσέγγισή μας είναι πολύ καλύτερη από την αντίστοιχη για ομοιόμορφο διαμερισμό.



Σχήμα 7: Μη Ομοιόμορφος διαμερισμός, $N = 20$ (αριστερά) και $N = 40$ (δεξιά).

Στον ακόλουθο πίνακα δίνεται το σχετικό σφάλμα

$$\mathcal{E}(N) = \frac{\max\limits_{1\le i\le N+1} |u(x_i) - U_i|}{\max\limits_{1\le i\le N+1} |u(x_i)|},$$

για ομοιόμορφο και μη ομοιόμορφο διαμερισμό του [0,4] σε N υποδιαστήματα.

$\mathcal{E}(N)$	Ομοιόμορφος διαμ.	Μη ομοιόμορφος διαμ.
$N = 20$	0.8811	0.0890
$N = 40$	0.1174	0.0230
$N = 80$	0.0283	0.0056
$N = 160$	0.0071	0.0014

Παρατήρηση: Φυσικά αυτό που θα επιθυμούσε κανείς από μια αριθμητική μέθοδο θα ήταν η επιλογή των κόμβων να γίνεται «αυτόματα» και χωρίς εκ των προτέρων γνώση της συμπεριφοράς της λύσης. Κάτι τέτοιο ξεφεύγει από τους σκοπούς αυτού του μαθήματος.