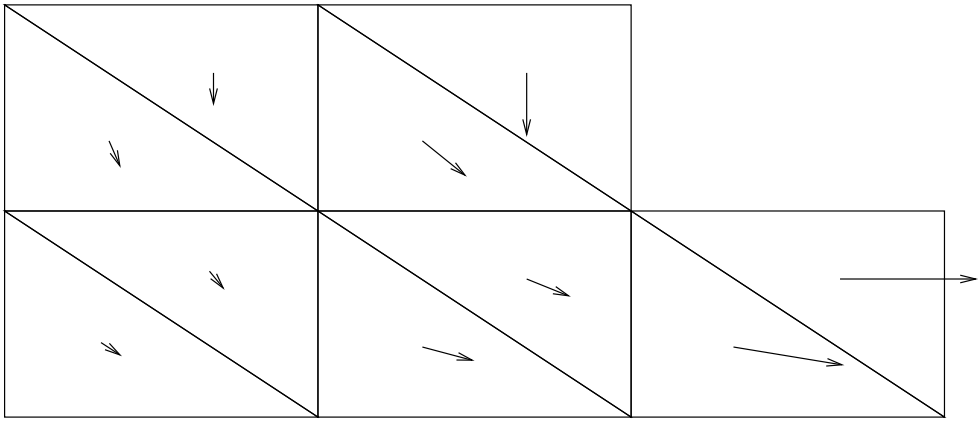
Irina Munteanu, Gabriela Ciuprina, F. M. G. Tomescu



Modelarea numerică a câmpului electromagnetic prin programe Scilab

Printech
Bucureşti, 2000

Irina Munteanu, Gabriela Ciuprina, F. M. G. Tomescu
Catedra de Electrotehnică, Universitatea "Politehnica" din Bucureşti

Modelarea numerică a câmpului electromagnetic prin programe Scilab

Printech
Bucureşti, 2000

Irina Munteanu, Gabriela Ciuprina, F. M. G. Tomescu

Modelarea numerică a câmpului electromagnetic prin programe Scilab.

Referenţi ştiinţifici: Prof. dr. ing. Daniel Ioan
Prof. dr. ing. Alexandru Morega

Printech, Bucureşti, 2000

Spl. Independenţei 313
77206 Bucureşti

Această carte a fost editată cu sprijinul CNCSU şi al Băncii Mondiale în cadrul proiectului CNCSU/36.

Cuprins

Introducere

Volumul de faţă reprezintă o dezvoltare a îndrumarului de laborator asociat cursului de *Modelare numerică a câmpului electromagnetic*, dedicat studenţilor anului IV al Facultăţii de Electrotehnică din Universitatea "Politehnica" din Bucureşti de la specializarea *Inginerie electrică asistată de calculator*.

Deşi lucrarea se adresează în primul rând studenţilor care urmează acest curs, el poate fi util tuturor studenţilor şi inginerilor care doresc să se familiarizeze cu conceptele de bază şi cu principalele metode numerice utilizate în analiza asistată de calculator a dispozitivelor electromagnetice.

Laboratorul de *Modelarea numerică a câmpului electromagnetic* este conceput în ideea urmăririi a două obiective. În primul rând, laboratorul trebuie să ilustreze metode, tehnici şi algoritmi de modelare numerică a câmpului electromagnetic, în acord cu subiectele tratate la curs. În acelaşi timp însă, în conformitate cu specializarea studenţilor, este urmărită crearea şi perfecţionarea abilităţilor de utilizare eficientă a calculatorului în practica inginerească. Acest al doilea obiectiv poate fi realizat prin iniţierea şi perfecţionarea studenţilor în utilizarea unui pachet de programe de mare eficienţă, cum este *Scilab*.

Cartea este structurată în şapte teme de laborator.

Prima temă reprezintă o iniţiere în sistemul de operare Linux, o variantă pentru PC-uri a sistemului de operare Unix. Deşi este mai puţin cunoscut decât Windows, Unix este sistemul de operare utilizat în întreaga lume pentru calcule ştiinţifice. Tema conţine o prezentare a mediului hardware de lucru în care studenţii îşi desfăşoară activitatea de laborator, o listă a principalelor comenzi Linux, precum şi instrucţiuni de utilizare pentru două utilitare de strictă necesitate, un editor de texte (`joe`) şi un program de poştă electronică (`elm`).

Tema a doua prezintă mediul de programare *Scilab* destinat efectuării de calcule matematice. *Scilab*, dezvoltat de Institutul Naţional Francez de Informatică (INRIA) este un emulator al binecunoscutului mediu Matlab. Tema descrie principalele tipuri de date şi instrucţiuni ale mediului *Scilab*, pe baza a numeroase exemple.

A treia temă reprezintă o trecere în revistă a conceptelor de bază necesare pentru înţelegerea metodelor numerice prezentate în următoarele teme. Sunt prezentate pe scurt aproximările polinomiale unidimensionale (de tip Lagrange, Hermite şi liniare pe porţiuni) şi bidimensionale (de tip Lagrange), precum şi variantele unidimensionale ale

metodelor diferenţelor finite şi reziduurilor ponderate.

Temele 4 – 7 prezintă principalele metode numerice de analiză a câmpului electromagnetic: metoda volumelor finite, diferenţelor finite, elementelor finite şi elementelor de frontieră. La fiecare dintre metode este descrisă în mod sugestiv ideea metodei, este prezentat modul de discretizare a ecuaţiilor câmpului electromagnetic în detaliu (până la formulele coeficienţilor sistemului final de ecuaţii) şi sunt prezentaţi algoritmii scrişi în pseudocod. De asemenea, fiecare temă conţine noţiuni de pre/postprocesare specifice metodei tratate. Algoritmii propuşi pot fi uşor implementaţi de către studenţi în limbajul *Scilab* şi testaţi pe configuraţii relativ simple, în regim staţionar.

Fiecare dintre teme conţine un număr mare de exerciţii, într-o ordine crescătoare a gradului de dificultate, ceea ce permite asimilarea mai rapidă a cunoştinţelor.

Contribuţia autorilor la realizarea lucrării este următoarea:

- Ş.l. univ. Gabriela Ciuprina a conceput şi tehnoredactat temele 2, 4, 5, 7 şi subcapitolele 3.1 – 3.3 din tema 3;
- Conf. univ. dr. ing. Irina Munteanu a conceput şi tehnoredactat temele 1, 6, subcapitolul 3.4 din tema 3 şi a realizat integrarea finală a temelor în volumul de faţă;
- Prof. univ. dr. ing. F. M. G. Tomescu a coordonat realizarea întregii lucrări şi a contribuit la concepţia temelor 3, 4, 6, 7.

Autorii mulţumesc proiectului TEMPUS 9122 şi proiectului Băncii Mondiale CNCSU/36, a căror generoasă finanţare a contribuit la dotarea sălii de calculatoare în care se desfăşoară laboratorul de *Modelare numerică a câmpului electromagnetic*.

De asemenea, autorii sunt recunoscători conducerii Facultăţii şi Catedrei de Electrotehnică din Universitatea "Politehnica" din Bucureşti, în special domnilor profesori Mihai Popescu, C. Ghiţă şi F. Hănţilă, pentru sprijinul acordat în vederea introducerii noii specializări şi în particular a acestui curs în planul de învăţământ. Nu în ultimul rând, mulţumiri i se cuvin domnului profesor Daniel Ioan, şeful Laboratorului de Metode Numerice, care a constituit punctul de plecare al întregii structuri de cercetare şi învăţământ în direcţia ingineriei electrice asistate de calculator.

Autorii sunt recunoscători tuturor celor care pot face comentarii şi sugestii în vederea îmbunătăţirii acestei cărţi, la una din adresele:

`irina@lmn.pub.ro`

`gabriela@lmn.pub.ro`

Capitolul 1

Iniţiere în Linux

Linux este o variantă a sistemului de operare Unix.

Tema 1 are ca scop familiarizarea cu mediul de lucru (intrarea în sistem, lucrul cu ferestre), cu câteva din principalele comenzi Linux precum şi cu două utilitare Linux: editorul de texte `joe` şi programul de poştă electronică `elm`.

Observaţie: Subcapitolele 1.1 – 1.4 sunt destinate cititorilor care lucrează în Laboratorul de Metode Numerice al Universităţii "Politehnica" din Bucureşti. Dacă nu este cazul Dvs., solicitaţi administratorului de reţea să vă indice regulile de acces în sistemul pe care lucraţi.

1.1 Mediul hardware de lucru

Laboratorul "Modelarea numerică a câmpului electromagnetic" se desfăşoară în sala EB–210 a catedrei de Electrotehnică din Politehnica bucureşteană. Dotarea hardware constă din:

- două servere marca Digital MicroVAX 3900 şi MicroVAX II;
- un server Linux de tip Pentium 75MHz;
- XTerminale marca Digital VAXStation 3100;
- reţea locală Ethernet 10Mb/s;
- conexiune Internet.

1.2 Intrarea în sistem

Pentru a putea lucra în acest sistem, fiecărui utilizator i se acordă:

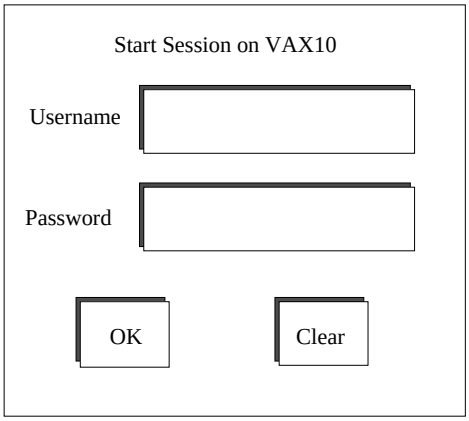
- un nume de cont (username);
- o parolă (password).

Accesul în sistem se realizează în două etape:

1. etapa de acces în sistemul Linux;
2. etapa de acces al utilizatorului la propriul său cont Linux.

Aceste două etape sunt descrise în cele ce urmează.

După pornirea reţelei de către administratorul de sistem, staţiile de lucru (XTerminale) afişează **fereastra de acces în reţea**, prin care se poate realiza conectarea la serverul Linux. Fereastra are aspectul:



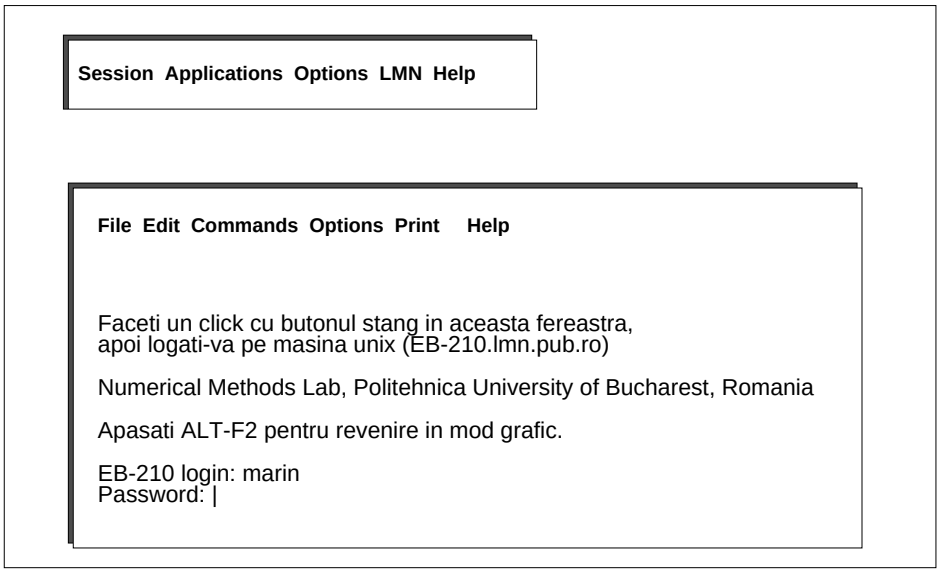
În această fereastră se introduc:

Username: `linux`

Password: `vax`

apoi se "apasă" (se mută cursorul mouse-ului şi se apasă butonul din stânga) butonul OK.

După câteva secunde se afişează pe rând două ferestre:



Fereastra din partea de jos a ecranului, numită **fereastra de acces a utilizatorului**, permite introducerea numelui de cont şi a parolei specifice fiecărui utilizator, astfel:

- se plasează cursorul în fereastra de acces a utilizatorului;
- se apasă butonul din stânga al mouse–ului (în limba engleză, "click");
- se introduc numele de cont şi parola:

  login: *username* < ↵ >

  password: *parola* < ↵ >

Aceste operaţii trebuie efectuate cât mai repede. Din motive de securitate, parola nu este afişată pe ecran.

Dacă numele şi parola sunt recunoscute de serverul Linux, după scurt timp pe ecran se afişează *mediul Linux*.

1.3 Lucrul în mediul Linux

La începutul unei sesiuni Linux se afişează o fereastră de lucru (XTerm) şi un grup de butoane (să-l numim în continuare Toolbox) în colţul din dreapta jos a ecranului.

Mutarea cursorului mouse–ului în fereastra de lucru are ca efect vizibil schimbarea culorii barei superioare a ferestrei. Se marchează astfel faptul că fereastra a devenit **fereastra activă** de lucru. În acest moment se pot da comenzi la promptul sistemului, `EB-210:~$`.

Butoanele mouse–ului au funcţii diferite, după cum cursorul se află plasat într-o fereastră sau în afara oricărei ferestre.

- **Cursorul mouse–ului este plasat într-o fereastră**
  - Butonul din stânga: permite *selectarea* unui text, astfel: se plasează cursorul la începutul textului, se apasă butonul din stânga şi se deplasează mouse–ul ţinând apăsat butonul din stânga. Textul astfel selectat îşi schimbă culoarea.
  - Butonul din mijloc: permite introducerea la promptul sistemului a textului selectat, literă cu literă, ca şi cum el ar fi introdus de utilizator de la tastatură.
  - Butonul din dreapta: se comportă similar cu butonul din stânga, dar permite selectarea unei porţiuni mai mari de text: se deplasează cursorul la începutul textului de selectat, se apasă butonul din stânga fără a deplasa mouse–ul, se deplasează mouse–ul la sfârşitul textului de selectat, se apasă butonul din dreapta.

EXERCIŢIUL 1:	Introduceţi textul "EB-210" urmat de < ↵ > (`Enter`) la promptul din fereastra de lucru. Faceţi aceeaşi operaţiune folosind butoanele mouse–ului. Comentaţi utilitatea acestei ultime tehnici. Observaţie: ignoraţi deocamdată mesajele de eroare afişate de sistemul de operare Linux – ele sunt, la acest exerciţiu, inofensive.

Cu excepţia acestui exerciţiu, NU IGNORAŢI NICIODATĂ MESAJELE DE EROARE afişate!!! Dacă nu ştiţi ce înseamnă, întrebaţi, dar NU LE IGNORAŢI!

- **Cursorul mouse–ului este plasat pe bordura unei ferestre**
  - Butonul din stânga: permite mutarea ferestrei;
  - Butonul din dreapta: permite "ridicarea" ferestrei la suprafaţă (deasupra altor ferestre de pe ecran) sau "ascunderea" ei (sub celelalte ferestre).
- **Cursorul mouse–ului este plasat în afara oricărei ferestre** (pe "fondul" ecranului")
  - Butonul din stânga: afişează un **meniu** din care se pot lansa diferite programe sau utilitare. Dintre acestea, deosebit de utile sunt: prima linie a meniului, care permite deschiderea unei noi ferestre de lucru; ultima linie a meniului, care permite ieşirea din sistem (nu încercaţi această opţiune chiar în acest moment!).
  - Butonul din mijloc: permite efectuarea de operaţii cu ferestrele de lucru: schimbarea dimensiunilor (resize), mutarea (move), închiderea (kill).
  - Butonul din dreapta: afisează o listă a ferestrelor deschise, şi permite deplasarea rapidă a cursorului în fereastra selectată.

EXERCIŢIUL 2:	Deschideţi o nouă fereastră. Folosind butonul din mijloc, schimbaţi-i dimensiunile. Mutaţi fereastra în colţul din stânga jos al ecranului. Închideţi fereastra nou creată.

Observaţie: Schimbarea dimensiunii unei ferestre la dimensiunea maximă (cât tot ecranul) se poate face şi apăsând al doilea buton (un pătrăţel) aflat în partea din dreapta sus a bordurii ferestrei.

Dacă la un moment dat sunt deschise mai multe ferestre, în fiecare dintre ele se pot da comenzi independente, ca şi cum utilizatorul ar avea la dispoziţie mai multe terminale (ecrane). Gestionarea unui număr mare de ferestre (care la un moment dat pot să se suprapună, unele dintre ele nefiind vizibile) este facilitată de:

- bara orizontală plasată în partea de jos a ecranului; ea conţine câte un buton pentru fiecare fereastră deschisă; apăsarea unui astfel de buton are ca efect mutarea cursorului în fereastra corespunzătoare şi selectarea acesteia ca fereastră activă;

- cele trei căsuţe intitulate Desktop din Toolbox (grupul de butoane plasate în colţul dreapta jos al ecranului); să notăm căsuţele cu numere de ordine: Desktop0, Desktop1, Desktop2. Ele pot fi privite ca porţiuni din "ecranul logic" pe care se pot plasa ferestre. Numai una din cele trei părţi ale acestui "ecran logic" este afişată, la un moment dat, pe ecranul fizic (la început, Desktop0).

EXERCIŢIUL 3:

a) Deschideţi încă două ferestre. Observaţi efectul asupra căsuţei Desktop0.
b) Deplasaţi-vă din fereastră în fereastră (activând astfel una sau alta dintre ferestre), în două moduri:

- deplasând cursorul mouse–ului;
- folosind bara orizontală din partea de jos a ecranului.

c) Mutaţi o fereastră în Desktop1 astfel:

- mutaţi fereastra spre dreapta până când o jumătate a ei nu mai este vizibilă (observaţi efectul asupra căsuţei Desktop1);
- deplasaţi cursorul mouse–ului pe căsuţa Desktop1 şi apăsaţi butonul din stânga al mouse–ului;
- poziţionaţi fereastra în Desktop1.

Comentaţi utilitatea celor trei desktop–uri.

1.4 Ieşirea din sistem

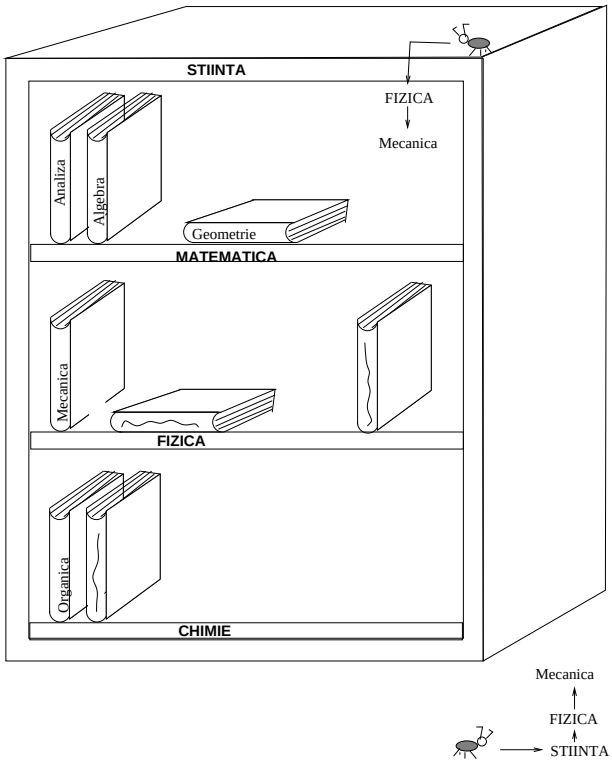
- Se închid pe rând toate ferestrele de lucru deschise, de exemplu folosind comanda `exit`;
- Se apasă butonul din stânga al mouse–ului şi se selectează ultima linie din meniu (`Exit Fvwm95`);
- La noul meniu apărut se răspunde (prin selectare cu mouse–ul) `Yes, really quit`;
- Se aşteaptă apariţia unei ferestre a reţelei, din care se selectează `Session` şi apoi `Exit`.

1.5 Câteva comenzi Linux de bază

Ca orice sistem de operare, Linux operează cu două concepte legate de stocarea informaţiei:

- fişiere: entităţi care conţin informaţii (text, programe);
- directoare: entităţi care permit gruparea şi ordonarea fişierelor.

Folosind similitudinea cu o bibliotecă, fişierele se aseamănă cu cărţile, în timp ce directoarele se aseamănă cu rafturile în care acestea sunt aşezate. La rândul lor, rafturile pot fi grupate în dulapuri de cărţi, care în Linux sunt tot directoare. Fiecare "dulap de cărţi", raft sau carte are un nume. Pentru a preciza unde se află o carte, trebuie precizate toate informaţiile care să permită găsirea cărţii: dulapul, raftul, titlul. Tot astfel în Linux trebuie precizată toată *calea* pe care se poate ajunge la un director sau fişier.



Dacă o furnică erudită ar vrea să ajungă până la cartea *Mecanică* din domeniul ştiinţelor exacte, şi dacă ea se află deja pe dulapul numit "Ştiinţă", nu mai este nevoie să i se precizeze şi dulapul, este suficient să i se indice raftul şi titlul exact al cărţii. Dacă furnica a reuşit să ajungă pe raftul care o interesează, *Fizica*, singura informaţie de care mai are nevoie este titlul cărţii. Aceste informaţii parţiale se numesc *relative* la poziţia în care se află furnica.

Dacă în schimb furnica se află la intrarea facultăţii, sau dacă rătăceşte prin dulapul marcat "Limbi străine", pentru a putea găsi cartea este nevoie să se precizeze toate informaţiile (dulap, raft, titlu), adică informaţii de natură *absolută*.

Pentru furnica aflată în sală, departe de dulapul de cărţi ştiinţifice, drumul ei ar fi descris în Linux de: `/STIINTA/FIZICA/Mecanica`,

în timp ce pentru furnica aflată deja în dulapul de ştiinţe, calea de mai sus poate fi prescurtată ca: `FIZICA/Mecanica`.

Observaţi lipsa primului caracter / (în engleză "slash") atunci când calea spre "cartea" (fişierul) de mecanică este dată relativ la poziţia curentă.

Numele fişierelor şi directoarelor pot conţine litere, cifre, caractere speciale: `_` (underscore sau caracter de subliniere), `-` (minus) şi `.` (punctul).

Exemplu: `program1, tema.pas.bak, tema-1, tema_2` sunt nume valide de fişiere sau de directoare.

Un nume format dintr-un singur punct semnifică *directorul curent*, oricare ar fi acesta. Directorul simbolizat `/`, cel care conţine toate celelalte directoare şi fişiere ale sistemului (şi care ar fi echivalent sălii de bibliotecă în care sunt plasate toate dulapurile cu cărţi), se numeşte *directorul rădăcină*.

Un director special este aşa–numitul director *home* al utilizatorului, simbolizat prin `$ HOME` sau prin $\sim$. La intrarea în sistem sunteţi plasaţi în directorul *home*.

Sunt deosebit de utile alte două notaţii: directorul *curent* este simbolizat cu `.` (punct); directorul *precedent* ierarhic, adică cel care **conţine directorul curent** este simbolizat cu `..` (de două ori punct).

1.5.1 Lucrul cu fişiere

- `ls` – afişează conţinutul (lista directoarelor şi fişierelor) directorului curent.

  Comanda `ls` poate fi apelată cu parametri, de exemplu:

  ```
  ls -l
  ```

  pentru afişarea conţinutului directorului curent, cu informaţii detaliate privitoare la drepturile de citire (`r`), scriere (`w`) sau execuţie (`x`), data şi ora ultimei modificări, dimensiunea fişierului, etc.
- `more` *nume_fişier* – listează conţinutul unui fişier.

  După lansarea `more`, sunt active următoarele comenzi:

  - *spaţiu* – avansare cu o pagină spre sfârşitul fişierului;
  - `b` – avansare cu o pagină spre începutul fişierului;
  - $<\hookleftarrow>$ – avansare cu un rând înainte;
  - `q` – ieşire din `more`.

EXERCIŢIUL 4:	a) Afişaţi lista fişierelor din directorul curent. Deschideţi o altă fereastră şi repetaţi comanda. Comentaţi. b) Afişaţi pe ecran, într-una din ferestrele de lucru, conţinutul fişierului `/usr/local/man/whatis`.

- `cp` *nume_fişier_existent* *nume_nou* – copiază fişierul *nume_fişier_existent* sub numele *nume_nou*.
- `mv` *nume_fişier_existent* *nume_nou* – redenumeşte fişierul *nume_fişier_existent*

  `mv` *nume_fişier_existent* *nume_director* – mută fişierul în directorul *nume_director*

- `rm` *nume_fişier_existent* – şterge fişierul.

EXERCIŢIUL 5:

a) Copiaţi fişierul `.cshrc` sub numele `test`. Listaţi conţinutul fişierului `test`.
Redenumiţi `test` în `test1`.
Listaţi conţinutul directorului curent. Ştergeţi fişierul `test1`.
b) Fără a le executa, încercaţi să prevedeţi efectul următoarelor instrucţiuni:

```
cp .bashrc temporar
mv temporar temp
rm temporar
rm temp
```

Executaţi aceste instrucţiuni şi observaţi efectul lor.

1.5.2 Lucrul cu directoare

- `cd` *nume_director* – schimbarea directorului curent

  Exemplu: Efectul următoarelor trei comenzi:

  ```
  cd /            schimbă directorul curent în directorul rădăcină
  cd stiinta      schimbă directorul curent în /stiinta (deplasare relativă)
  cd matematica
  ```

  este echivalent cu al comenzii:

  ```
  cd /stiinta/matematica
  ```

  Comanda `cd` fără nici un nume de director are ca efect mutarea în directorul *home*.

- `pwd` – afişează numele directorului curent (calea absolută)

- `mkdir` *nume_director* – crează un nou director. Dacă numele directorului nu este precedat de o cale, noul director este creat în cel curent.

  Exemplu:

  `mkdir modelare` – crează directorul `modelare` în directorul curent

  `mkdir /modelare` – crează directorul `modelare` în directorul rădăcină (de obicei, această comandă va genera un mesaj de eroare, deoarece nu aveţi drepturi de scriere în directorul rădăcină).

EXERCIŢIUL 6:

Mutaţi-vă în directorul Dvs. *home*:
`cd` $<\hookleftarrow>$
Listaţi conţinutul acestui director.

Creaţi două noi directoare, `modelare` şi `inform`, în care veţi lucra de acum înainte la laboratoarele de "Modelarea dispozitivelor electromagnetice" şi respectiv "Transmisiunea informaţiei".
Listaţi noul conţinut al directorului *home*.

În directorul `modelare`, creaţi un nou director, `tema1`.
Aţi reuşit?

- `rmdir` *nume_director* – şterge directorul.

  Dacă directorul conţine alte fişiere sau directoare, el nu poate fi şters cu comanda anterioară.

  Atenţie! Comenzile `rm` şi `rmdir` au efect definitiv! Un fişier sau director, odată şters, nu mai poate fi recuperat!

1.5.3 Schimbarea parolei

- `passwd` – permite schimbarea parolei.

  Se solicită introducerea vechii parole şi apoi a noii parole (de două ori, pentru verificarea introducerii corecte a noii parole).

  Parola trebuie să fie suficient de lungă (numărul minim de caractere variind de la sistem la sistem), să conţină atât litere cât şi caractere speciale (numere, semne de punctuaţie).

  Recomandări: Schimbaţi parola periodic. Nu ignoraţi mesajul afişat la intrarea în sistem, care vă poate avertiza în legătură cu data de expirare a parolei. Nu comunicaţi nimănui parola şi nu o notaţi.

1.5.4 Procese şi utilizatori

- `who` – afişează lista utilizatorilor conectaţi în sistem

  `w` – afişează lista utilizatorilor conectaţi în sistem şi comanda pe care aceştia o execută;

  `whoami` – "Cine sunt eu?" – afişează numele de cont al utilizatorului;

- `ps` – lista proceselor.

  Sub Linux, mai multe comenzi sau programe pot fi rulate "în paralel" (de exemplu, dacă aveţi două ferestre deschise puteţi da comenzi care vor fi executate în mod independent). Fiecare comandă în curs de execuţie se numeşte *proces* şi este identificată printr-un număr, numit PID (Process IDentifier).

Comanda `ps` fără nici un parametru afişează lista proceselor lansate de utilizatorul curent.

- `kill` *PID* – opreşte definitiv execuţia procesului cu numărul *PID*.

EXERCIŢIUL 7:	Listaţi procesele în curs de execuţie. Mutaţi-vă în directorul dvs. *home*. Într-o fereastră, daţi comanda: `more /usr/local/man/whatis` În altă fereastră, daţi comanda `ps` Comentaţi diferenţele faţă de rezultatul precedentei comenzi `ps`. Lucrând în a doua fereastră, opriţi execuţia procesului care corespunde comenzii `more usr/local/man/whatis`.

1.5.5 Ajutor!

Comanda care permite aflarea sintaxei unei comenzi, precum şi multe alte informaţii despre comandă, este:

- `man` *nume_comandă*

  După lansarea ei, sunt active comenzile de deplasare descrise la `more`.

- `man -k` *cuvânt_cheie* sau

  `apropos` *cuvânt_cheie* – caută cuvinte cheie în lista de comenzi Linux.

1.5.6 Diverse utilitare

- `date` – afişează data şi ora;
- `sort` *nume_fişier* – sortează în ordine alfabetică liniile din fişier şi afişează rezultatul pe ecran
- `clear` – şterge fereastra de lucru;
- `wc` *nume_fişier* – numără liniile, cuvintele şi caracterele din fişier şi afişează rezultatul pe ecran;
- `<Ctrl> C` sau `<Ctrl> D` – opreşte execuţia unei comenzi.

EXERCIŢIUL 8:

a) Folosind comanda `man` aflaţi care este efectul comenzii `cal`.

b) Copiaţi fişierul `.bashrc` din directorul *home* în directorul `~/modelare/tema1/`.
Comparaţi efectul comenzilor `sort .bashrc`
şi respectiv
`sort .bashrc > sortat`
(Indicaţie: încercaţi `ls` şi `more ...`).
Explicaţi care este efectul adăugării caracterului > urmat de un nume, la sfârşitul unei comenzi.

1.6 Editorul de texte `joe`

`joe` este un editor care emulează editorul MS–DOS *WordStar*, extrem de popular acum un deceniu.

Majoritatea comenzilor `joe` folosesc tasta `Ctrl`, care va fi marcată de acum înainte ca ↑. (Simbolul folosit în fişierele de Help ale editorului este ^.)

De exemplu, pentru a introduce comanda notată ↑A (se citeşte "control a"), se ţine apăsată tasta `Ctrl` şi apoi se apasă tasta A. Anumite comenzi necesită apăsarea simultană a tastelor `Ctrl` şi `Shift` pe lângă caracterul comenzii, ca de exemplu în comanda ↑`Shift` 6.

În alte comenzi, simbolul tastei `Ctrl` este urmat de două litere, ca în ↑KA. Această comandă se poate introduce fie ca ↑K, urmat de litera A, fie ca ↑K urmat de ↑A.

(Atenţie, în comenzile descrise mai sus "A" semnifică *tasta* A, şi nu litera mare A, care ar necesita apăsarea suplimentară a tastei `Shift`.)

Principalele comenzi ale editorului `joe`, grupate după funcţionalitatea lor, sunt prezentate mai jos.

- **Lansarea editorului**

  `joe` *nume_fişier* < ↵ >

- **Introducerea textului**

  Textul se introduce acum ca în orice editor. Nu este nevoie să se introducă <↵> la sfârşit de linie deoarece la depăşirea unui anumit număr de caractere pe o linie editorul face automat trecerea la linie nouă;

- **Deplasarea cursorului**

  Se poate face cu ajutorul săgeţilor ↑, ↓, ←, → sau folosind comenzile:

↑A	sau tasta `Home`[1]	– deplasare la începutul liniei
↑E		– deplasare la sfârşitul liniei
↑V	sau tasta `Next Screen`[1]	– deplasare cu un ecran spre sfârşitul fişierului
	sau tasta `Page Down`[1]	
↑U	sau tasta `Prev Screen`[1]	– deplasare cu un ecran spre începutul fişierului
	sau tasta `Page Up`[1]	

- **Help**

  Comanda de help este ↑KH.

  Ea afişează un meniu de help în partea de sus a ecranului. Se poate trece la "pagina" următoare de help cu ↑ [. (sau Escape[1] **.**), şi la pagina anterioară cu ↑ [, (sau Escape[1] **,**).

  Fereastra de help poate fi păstrată pe ecran, pentru referinţă, pe tot parcursul lucrului în `joe`.

- **Salvare şi ieşire din editor**

↑KS sau ↑KD	– salvare şi continuarea lucrului
↑KX	– salvare şi ieşire din `joe`
↑C	– ieşire fără salvare.

- **Editarea şi formatarea textului**

↑ `Shift` - (*minus*)	– undo
↑ `Shift` 6	– redo
Tasta `Backspace` sau `Del`[1]	– şterge caracterul dinaintea cursorului
↑D	– şterge caracterul de la cursor
↑KA	– centrare linie

- **Modificarea setărilor implicite**

  Se introduce comanda ↑T, urmat de o literă sau de deplasarea cursorului folosind săgeata dreapta →.

  Se pot modifica, printre altele:

  - Indentarea automată (Autoindent, implicit: OFF)
  - Trecerea automată pe linie nouă (Wordwrap, implicit: ON)
  - Crearea automată, la fiecare salvare, a unui fişier backup
  - Inserare caractere / scriere peste vechile caractere (Overtype, implicit: OFF),
  - . . ., etc.

- **Lucrul cu fişiere**

↑ KR	– Inserarea unui alt fişier în cel aflat în curs de editare
↑ KE	– Schimbarea fişierului care se editează

[1] Poate lipsi de pe unele tastaturi.

- **Căutări şi înlocuiri**
  - ↑ KF *text* < ↵ > – căutare, cu opţiunile:
    - `b` – căutare înapoi
    - `r` – căutare şi înlocuire
  - ↑ L – repetarea ultimei căutări
- **Lucrul cu blocuri de text**
  - ↑ KB – marcarea începutului unui bloc
  - ↑ KK – marcarea sfârşitului unui bloc
  - ↑ KM – mutarea blocului marcat la poziţia cursorului
  - ↑ KC – copierea blocului marcat la poziţia cursorului
  - ↑ KY – ştergerea blocului marcat
  - ↑ KB ↑ KK – "ascunderea" blocului marcat
  - ↑ TX ↑ KB (*deplasare cursor*) ↑ KK – marcarea unei coloane

EXERCIŢIUL 9:

În directorul ∼`/modelare/tema1` creaţi un fişier numit `tema`, folosind editorul `joe`.
Fişierul va conţine răspunsurile (pe scurt!) la exerciţiile 1–8. (Unele exerciţii nu solicită răspunsuri la întrebări, nici comentarii. Notaţi totuşi că aţi parcurs exerciţiul, de exemplu *Exerciţiul x: OK*.

1.7 Programul de poştă electronică `elm`

1.7.1 Apelare

```
elm
```

După intrarea în `elm` puteţi:

- să citiţi mesajele primite;
- să trimiteţi mesaje;
- să menţineţi o listă de "alias"-uri personale;
- să ştergeţi sau să mutaţi în alte căsuţe poştale mesajele primite;
- să modificaţi opţiunile programului `elm`;
- să ieşiţi din `elm`.

1.7.2 Ajutor!

Pentru "help" se dă comanda: `?` urmat de:

`nume_comandă` - afişează informaţii despre această comandă
`?` - afişează lista tuturor comenzilor disponibile

1.7.3 Citirea mesajelor primite

< ↵ > - citirea mesajului curent fără a afişa antetul (header-ul) mesajului

`h` - citirea mesajului curent, cu afişarea antetului (header)

`n` - setarea mesajului numarul n ca mesaj curent

`+` - avans un ecran

`-` - înapoi un ecran

`K` - plasarea cursorului pe mesajul anterior

`J` - plasarea cursorului pe mesajul următor

`k` sau ↑ - plasarea cursorului pe precedentul mesaj neşters

`j` sau ↓ - plasarea cursorului pe următorul mesaj neşters

În cursul citirii unui mesaj:

`<spaţiu>` - avansează un ecran

↑ - mesaj precedent

↓ - mesaj următor

1.7.4 Trimitere de mesaje

Se dă una din comenzile:

`m` - trimitere mesaj

`r` - răspuns expeditorului mesajului curent

`g` - răspuns expeditorului şi tuturor recipienţilor mesajului curent

`f` - "forward" mesaj curent (retransmitere)

`b` - "bounce" mesaj curent (retransmitere)

Programul solicită:

- adresantul mesajului – listă de adrese Internet completă sau aliasuri (la `m`, `f` sau `b`);
- subiect;
- copii către: (cc –carbon copy) - listă de adrese internet sau aliasuri, separate prin spaţii.

Se intră apoi într-un editor de text, în mod implicit `vi`. După comanda de scriere şi salvare a mesajului, utilizatorul poate:

`s` - să trimită mesajul

`e` - să editeze header-ul mesajului

`f` - să renunţe la trimiterea mesajului

Pentru crearea unor headere personalizate, se foloseşte fişierul

`$HOME/.elm/elmheaders`

Printre altele, în el se poate introduce linia:

`Return-Receipt-To:` *user*`@lmn.pub.ro`

utilă pentru a avea confirmarea de primire a mesajelor trimise.

1.7.5 Aliasuri personale

Se dă comanda:

`a`

În modul "alias" sunt disponibile comenzile:

- `n` - creare alias nou
- `a` - creare alias din adresa expeditorului mesajului curent

  Se solicită:
  - alias (nume scurt)
  - nume de familie (last name)
  - prenume (first name)
  - comentariu opţional
  - adresă de e-mail (sau listă de adrese, pentru a crea un grup)
  - <↵ >
- `?` - help pentru comenzile disponible în "alias"
- `d` - ştergerea unui alias
- `c` - modificare alias curent

<↵ > - verificarea adresei de e-mail pentru alias-ul curent

1.7.6 Ştergerea / mutarea mesajelor în altă căsuţă poştală

- `d` - şterge mesaj curent
- `s` - salvează mesaj curent în altă căsuţă poştală şi îl marchează ca "şters" (Deleted)
- `C` - salvează mesaj curent în altă casuţă poştală fără să îl marcheze ca "Deleted"
- `c` - schimbarea căsuţei poştale; se pot folosi următoarele prescurtări pentru calea de acces la un fişier:
  - fişier fis în directorul $HOME: ∼/fis
  - fişier fis în directorul Mail: =/fis

- fişierul de mesaje primite: <
- fişierul de mesaje trimise: >
- casuţă postală "principală": !

Sunt utile comenzile:

`t` - marcare mesaj curent pentru operaţii ulterioare (salvare, ştergere, etc.)

↑ `t` - marcarea tuturor mesajelor care îndeplinesc o condiţie, în vederea unor operaţii ulterioare

1.7.7 Modificarea opţiunilor

- în mod comandă: `o`
  - Schimbarea nivelului de utilizator: `u` - 0 - începător; 1 - familiar; 2 - expert
  - Afişare mail: `d` - builtin; (usr/bin/more)
  - Editor: `e` - /usr/bin/vi
  - Folder (unde se ţin căsuţele poştale ale utilizatorului):

    `f` - $HOME/Mail
  - Criteriul de sortare a mesajelor:

    `s` - received (în ordinea primirii, cel mai recent primul)

    R - reverse received (în ordinea primirii, cel mai recent ultimul)
  - Căsuţa unde se pun mesajele trimise:

    `o` - $HOME/mbox
    ($HOME/outbox) – trebuie definit copy = ON în fişierul elmrc
  - Programul de printare:

    `p`
  - Numele utilizatorului:

    `y`
  - Forma cursorului:

    `a` - OFF (bară video-invers); ON (săgeată)
  - Afişare numai numele persoanei în alias (Names Only):

    `n` - OFF (afişeaza şi adresa de Internet); ON (afişeaza numai numele)
- în fişierul ∼/.elm/elmrc:

  Fişierul conţine variabile grupate pe următoarele categorii:
  - formatarea afişărilor
  - dispozitive ("devices") utilizate
  - directoare şi fişiere de mesaje

– opţiuni/răspunsuri implicite

Toate opţiunile sunt date prin setarea unor variabile. Instructiunile sunt de forma: <variabilă> = <valoare>

Exemplu:

```
askcc = ON
editor = /usr/bin/vi
names = ON
```

EXERCIŢIUL 10:

a) Modificaţi opţiunea "Editor" implicită a programului `elm`, astfel încât editorul folosit să fie `/usr/bin/joe`
b) Trimiteţi fişierul creat la Exerciţiul 9, prin poştă electronică, la adresele: `irina@lmn.pub.ro` şi `gabriela@lmn.pub.ro`.

1.7.8 Ieşirea din `elm`

`x` - ieşire fără modificarea căsuţei poştale

`q` - ieşire cu modificarea căsuţei poştale

În concluzia acestei teme ...

Aţi parcurs prima temă, v-aţi familiarizat atât cu editorul de texte `joe` cât şi cu utilizarea poştei electronice.

Dacă sunteţi studentul Facultăţii de Electrotehnică şi utilizaţi această carte în cadrul laboratorului de Modelare numerică a câmpului electromagnetic: începând cu tema 2 răspunsurile la întrebări vor fi scrise **în fişiere** şi trimise cadrului didactic:

`irina@lmn.pub.ro` (Irina Munteanu)

sau

`gabriela@lmn.pub.ro` (Gabriela Ciuprina)

Capitolul 2

Iniţiere în Scilab

Scilab (Ψlab) este un mediu de programare sub sistemul de operare Unix care permite rezolvarea unor probleme tipice de matematică prin efectuarea de calcule matematice, trasarea de grafice, programare în limbaj specific. El emulează limbajul ***MATLAB***, având extensii suplimentare.

În cadrul laboratorului de Modelare numerică a câmpului electromagnetic el este folosit în rezolvarea numerică a unor probleme de câmp electromagnetic prin metoda elementelor de volum (tema 4), metoda diferenţelor finite (tema 5), metoda elementelor finite (tema 6), metoda elementelor de frontieră (tema 7).

Tema 2 are ca scop familiarizarea cu *Scilab* în vederea rezolvării temelor ulterioare.

2.1 Înainte de toate....

Una din componentele mediului *Scilab* este interpretorul care permite introducerea în mod interactiv a unor comenzi de la consolă şi executarea imediată a acestora.

• *Cum se lansează în execuţie interpretorul?* Lansarea interpretorului se face prin invocarea numelui său:

`scilab` $<\hookleftarrow>$

• *Cum se opreşte interpretorul?* La promptul *Scilab* `-->` se dă comanda:

`quit` $<\hookleftarrow>$

• *Există programe demostrative?* Apăsaţi butonul "Demos" şi apoi (de exemplu) "Introduction to SCILAB". Sunt vizualizate instrucţiunile utilizate precum şi rezultatele lor.

• *Ajutor !!.* Lista comenzilor şi semnificaţia lor se poate obţine apăsând butonul "Help". Fereastra de help este împărţită în două. În partea de jos sunt principalele capitole. În cadrul acestui laborator veţi avea nevoie de: "Scilab Programming", "Graphic Library", " Utilities and Elementary Functions" şi "Linear Algebra". Partea de sus conţine comenzi

legate de subiectul capitolului selectat. Selectarea unei astfel de comenzi are ca rezultat apariţia unei ferestre care descrie sintaxa comenzii, parametrii ei, modul de apelare.

Există şi help-on-line. La promptul *Scilab*-ului apăsaţi:

`help` $<\hookleftarrow>$

• *Nu ştii comanda care face ....?* O comandă utilă este comanda `apropos`, care caută cuvinte cheie în help-ul *Scilab*-ului.

EXERCIŢIUL 1:

a) Folosind comanda `help` aflaţi care este sintaxa comenzii `apropos`.
b) Folosind comanda `apropos` aflaţi care sunt comenzile cu care se pot trasa spectre de linii de câmp.

2.2 Variabile şi constante.

În rezolvarea temelor veţi folosi în exclusivitate matrice cu elemente reale. Un număr poate fi considerat o matrice cu un singur element.

• *Dimensiunea unei matrice* nu trebuie declarată explicit.

EXERCIŢIUL 2:

Care este efectul comenzii:

```
--> a(10,5) = 1
```

• *Introducerea unei matrice* se poate face natural astfel:

```
--> A = [  1  2  3
           4  5  6
           7  8  9  ]
```

Într-o scriere compactă, liniile matricei pot fi separate prin ";" astfel:

```
--> A = [1 2 3; 4 5 6; 7 8 9]
```

Pentru separarea elementelor unei linii se poate folosi caracterul blank (ca mai sus) sau virgula:

```
--> A = [1,2,3;4,5,6;7,8,9]
```

EXERCIŢIUL 3:

Ştiind că vectorii sunt un caz particular de matrice, care sunt comenzile cu care se va introduce un vector linie, respectiv coloană cu elementele 1, 2, 3?

EXERCIŢIUL 4: Comentaţi următoarea instrucţiune:

```
u = 12.4e-3
```

Observaţie: variabilele utilizate într-o sesiune de lucru ocupă memoria sistemului pe măsură ce sunt definite. Pentru a vizualiza lista variabilelor existente la un moment dat şi memoria disponibilă se foloseşte comanda `who`. Dacă se doreşte eliberarea memoriei de una, mai multe sau toate variabilele generate se foloseşte comanda `clear`.

EXERCIŢIUL 5: Executaţi instrucţiunea **who**. Ce reprezintă

```
%e  %pi  %f  %t  %eps  %inf   ?
```

2.3 Atribuiri şi expresii

Instrucţiunea de atribuire are sintaxa:

```
variabila = expresie
```

sau simplu:

```
expresie
```

în care `variabila` este numele unei variabile, iar `expresie` este un şir de operatori şi operanzi care respectă anumite reguli sintactice. În a doua formă, după evaluarea expresiei, rezultatul este atribuit variabilei predefinite `ans`.

• *Operatorii aritmetici*[1] recunoscuţi de ***Scilab*** sunt:

`+` adunare;
`-` scădere;
`*` înmulţire;
`/` împărţire la dreapta;
`\` împărţire la stânga;
`^` ridicare la putere.

Pentru transpunerea unei matrice se foloseşte operatorul "apostrof" ca în exemplul:

```
--> B = A'
```

în care matricea `B` se calculează ca transpusa matricei `A`.

[1] Operatorii aritmetici se aplică unor operanzi aritmetici, iar rezultatul este aritmetic.

Observaţii:

1. Adunarea şi scăderea pot fi efectuate:
   - între două matrice cu aceleaşi dimensiuni;
   - între o matrice şi un număr (caz în care numărul este adunat, respectiv scăzut din fiecare din elementele matricei).
2. Înmulţirea poate fi efectuată:
   - între două matrice dacă lungimea liniei primei matrice este egală cu lungimea coloanei celei de a doua matrice;
   - între un număr şi o matrice (caz în care numărul este înmulţit cu fiecare din elementele matricei);
   - între două matrice cu aceleaşi dimensiuni (element cu element), caz în care operatorul * este precedat de un punct, ca în exemplul:

     ```
     --> C = A .* B
     ```
3. Împărţirea matricelor poate fi făcută în mai multe feluri:
   - la dreapta (pentru matrice pătrate şi nesingulare):

     ```
     --> X = B / A
     ```

     echivalent cu:

     ```
     --> X = B * inv(A)
     ```

     sau cu:

     ```
     --> X = B * A^(-1)
     ```
   - la stânga:

     ```
     --> X = A \ B
     ```

     echivalent cu:

     ```
     --> X = inv(A) * B
     ```

     sau cu:

     ```
     --> X = A^(-1) * B
     ```

     Dacă A este o matrice dreptunghiulară de dimensiuni $m \times n$, iar b este un vector coloană cu m elemente, atunci împărţirea la stânga $x = A \setminus b$ calculează soluţia ecuaţiei $Ax = b$ în sensul celor mai mici pătrate.
   - împărţirea unei matrice la un număr (să îl notăm cu `u`):

     ```
     --> Y = A / u
     ```
   - împărţirea element cu element a matricelor de dimensiuni egale:

```
--> C = A ./ B
```

sau

```
--> C = A .\ B
```

4. Ridicarea la putere `A^p` se face astfel [2]:

   - pentru `p` întreg pozitiv: dacă matricea `A` este pătrată atunci `A` se înmulţeşte cu ea însăşi de `p` ori; dacă matricea `A` este dreptunghiulară atunci se ridică la puterea `p` fiecare element din matricea `A`
   - pentru `p` întreg negativ: dacă matricea `A` este pătrată atunci inversa ei se înmulţeşte cu ea însăşi de $-p$ ori; dacă matricea `A` este dreptunghiulară atunci se ridică la puterea `p` fiecare element din matricea `A`
   - pentru `p` număr real (dar nu întreg) pozitiv: dacă matricea `A` este pătrată atunci calculul se face cu ajutorul vectorilor şi valorilor proprii ale matricei; dacă matricea `A` este dreptunghiulară, atunci se ridică la puterea `p` fiecare element din matricea A.
   - pentru `p` număr real (dar nu întreg) negativ: dacă matricea `A` este pătrată atunci calculul se face cu ajutorul vectorilor şi valorilor proprii ale inversei matricei; dacă matricea `A` este dreptunghiulară, atunci se ridică la puterea `p` fiecare element din matricea A.

- *Operatorii de relaţie*[3] recunoscuţi de ***Scilab*** sunt:
  - `<` mai mic decât;
  - `<=` mai mic sau egal cu;
  - `>` mai mare decât;
  - `>=` mai mare sau egal cu;
  - `==` egal cu;
  - `~=` diferit de.

Aceştia permit testarea unor condiţii, rezultatul având valoarea `F` (fals) sau `T` (adevărat). Dacă operanzii sunt matrice de dimensiuni egale, atunci operaţiile logice se fac între elementele de pe aceleaşi poziţii, iar rezultatul este o matrice cu elementele `F` şi `T`.

- *Operatorii logici*[4] recunoscuţi de ***Scilab*** sunt:
  - `&` conjuncţia logică;
  - `|` disjuncţia logică;
  - `~` negaţia logică.

Dacă operanzii sunt matrice (logice) cu aceleaşi dimensiuni, atunci operaţia se face element cu element. Dacă unul din operanzi este o valoare logică, atunci acesta se combină cu fiecare din elementele celuilalt operand. Alte situaţii nu sunt permise.

[2] Nu sunt descrise toate situaţiile posibile.

[3] Operatorii de relaţie se aplică unor operanzi aritmetici iar rezultatul este logic.

[4] Operatorii logici se aplică unor operanzi logici iar rezultatul este logic.

• *Funcţii elementare.* Operanzii unor expresii pot fi şi apeluri de funcţii elementare (de exemplu trigonometrice), sau alte funcţii cunoscute. Aceste funcţii aplicate unei matrice acţionează asupra fiecărui element în mod independent. Lista lor poate fi inspectată apăsând butonul "Help" şi apoi "Utilities and Elementary Functions".

EXERCIŢIUL 6: Fie $A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$, $b = \begin{bmatrix} 5 \\ 6 \end{bmatrix}$ şi $v = \begin{bmatrix} 4 & 3 \end{bmatrix}$. Care sunt comenzile ***Scilab*** pentru rezolvarea ecuaţiei: $A\left(v^T + x\right) = b$

2.4 Generarea vectorilor şi matricelor

• *Vectorii ai căror elemente formează o progresie aritmetică* pot fi generaţi cu construcţia:

valin : pas : valfin

EXERCIŢIUL 7: Comentaţi rezultatele următoarelor instrucţiuni:

```
--> x = 1:10
--> y = 1:2:10
--> z = 1:3:10
--> t = 1:-3:10
--> w = -1:-3:-10
--> u = -1:-10
--> v = -10:-1
--> a = 10:-2:-3
```

• *Vectorii ai căror elemente formează o progresie geometrică* pot fi generaţi cu construcţia:

```
logspace(d1, d2, n)
```

EXERCIŢIUL 8: a) Care este semnificaţia mărimilor $d1$, $d2$, n din comanda `logspace` ?
b) Ce generează comanda `linspace` ?

• *Descrierea vectorilor şi matricelor pe blocuri.* Vectorii şi matricele pot fi descrişi pe blocuri, folosind notaţii de forma:

```
A = [X Y; U V];
```

cu semnificaţia $A = \begin{bmatrix} X & Y \\ U & V \end{bmatrix}$, în care X, Y, U, V sunt matrice sau vectori.

EXERCIŢIUL 9: Care este rezultatul comenzii:

```
--> A = [1:3 ; 1:2:7]
```

• *Referirea la elementele unei matrice.* Pentru a obţine valoarea unui element, se folosesc construcţii de forma $a(1,1), a(1,2)$. Se pot obţine valorile mai multor elemente prin construcţii de forma $a(u,v)$ unde u şi v sunt vectori. De exemplu a(2,1:3) reprezintă primele trei elemente din linia a doua a matricei a. Pentru a obţine toate elementele liniei 2 se scrie $a(2,:)$.

EXERCIŢIUL 10:

Fie $A = \begin{bmatrix} 1 & 10 & 100 & 1000 \\ 2 & 20 & 200 & 2000 \\ 3 & 30 & 300 & 3000 \end{bmatrix}$.

Notaţi rezultatele şi comentaţi următoarele comenzi:

```
--> A(0,1)
--> A(2,3)
--> A(:,3)
--> A(:,:)
--> A(3,:)
--> A(2,2:4)
--> A(2:3,2:4)
```

• *Generarea unor matrice particulare utile* se poate face cu ajutorul funcţiilor:

`eye` matrice cu elementele unitare pe diagonală şi nule în rest;
`zeros` matrice nulă;
`ones` matrice cu toate elementele unitare;
`rand` matrice cu elemente aleatoare în intervalul (0,1);
`diag` construieşte o matrice cu o anumită diagonală, sau extrage diagonala dintr-o matrice.

EXERCIŢIUL 11:

Comentaţi următoarele comenzi (unde A este matricea de la exerciţiul 10 iar $v = [\ 1 \quad 2 \quad 3 \quad 4 \quad 5\ ]$):

```
--> eye(3)
--> eye(3,3)
--> eye(3,4)
--> eye(A)
--> diag(A)
--> diag(v)
```

EXERCIŢIUL 12:

Comentaţi următoarele comenzi:

```
--> A = diag(1:3)
--> B = [A, eye(A); ones(A) zeros(A)]
--> C = diag(B)
--> D = C'*C
--> E = (D == 14)
```

• *Dimensiunile matricelor (vectorilor)* pot fi modificate în timpul execuţiei unui program. Pentru a obţine dimensiunea unei matrice X se foloseşte instrucţiunea:

```
[m, n] = size(X)
```

în care m reprezintă numărul de linii şi n numărul de coloane. Dimensiunea unui vector v se obţine cu:

```
length(v)
```

care are semnificaţia `max(size(v))`.

EXERCIŢIUL 13:

Definiţi o matrice oarecare B (de exemplu cu 3 linii şi 4 coloane). Executaţi şi comentaţi următoarele instrucţiuni:

```
--> [m,n] = size(B)
--> B = [B; zeros(1, n)]
--> B = [B zeros(m+1,1)]
```

• *Matricea vidă.* ***Scilab*** operează şi cu conceptul de matrice vidă, notată cu [] şi care este o matrice de dimensiune nulă, fără elemente. Aceasta se dovedeşte utilă în eliminarea unor linii sau coloane dintr-o matrice dată. De exemplu, instrucţiunea

```
--> B(:, [2 4]) = []
```

are ca efect eliminarea coloanelor 2 şi 4 din matricea B. În acest fel, dimensiunea unei matrice poate să şi scadă în timpul execuţiei unui program, nu numai să crească prin adăugarea de noi elemente.

2.5 Instrucţiuni grafice

Funcţia principală pentru reprezentări grafice este:

```
plot
```

Ea are diferite variante[5], printre care:

```
plot(x,y)
```

în care x este vectorul variabilei independente, iar y este vectorul variabilei dependente. Instrucţiunea:

```
plot(A)
```

[5] Încercaţi **apropos plot**

în care A este o matrice are ca efect reprezentarea grafică a variaţiei elementelor coloanelor matricei A în funcţie de indexul lor. Numărul de grafice este egal cu numărul de coloane.

Funcţiile auxiliare `xtitle` şi `xgrid` permit completarea graficului cu un titlu şi respectiv adăugarea unui rastru.

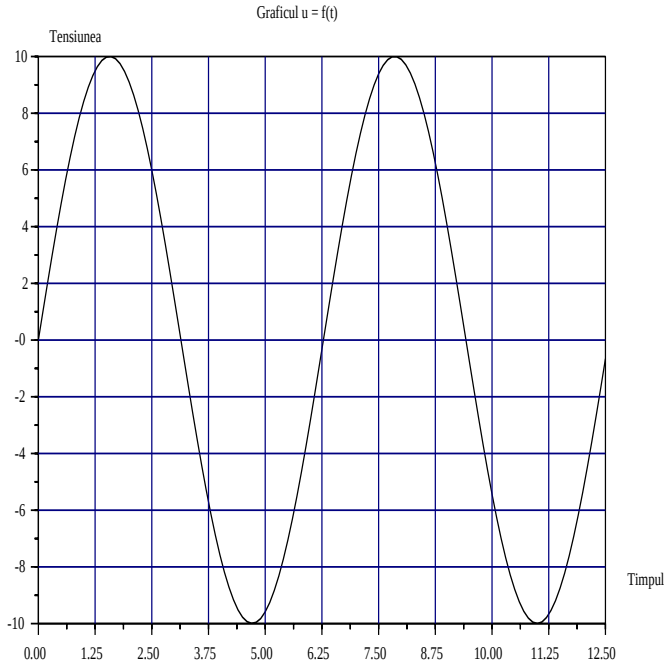


Fig. 2.1. Acest grafic trebuie obţinut la exerciţiul 14

EXERCIŢIUL 14:	Realizaţi graficul din figura 2.1. El reprezintă funcţia $y(t) = 10 * \sin(t)$ pentru $t \in [0, 4\pi]$. Puneţi titlul graficului, axelor, şi rastrul ca în figură.

2.6 Programare în *Scilab*

Scilab permite utilizarea unor structuri de control (decizii, cicluri, definiri de funcţii) ca orice limbaj de programare de nivel înalt. Se pot scrie programe în ***Scilab***, ca în orice limbaj de programare.

Avantajul folosirii ***Scilab*** constă, mai ales, în uşurinţa cu care se pot postprocesa rezultatele, ţinând seama de facilităţile grafice ale sale.

2.6.1 Editarea programelor

Până acum, aţi lucrat la consola ***Scilab***. Comenzile introduse pot fi scrise într-un fişier şi apoi "citite" cu comanda

```
exec
```

Un program în ***Scilab*** are următoarea structură posibilă:

```
// comentarii
//.....
instrucţiune;    // fără afişarea rezultatului
instrucţiune     // cu afişarea rezultatului
```

EXERCIŢIUL 15:

Scrieţi comenzile cu care aţi rezolvat exerciţiul 14, într-un fişier numit tema2.sci, din directorul ∼/modelare/tema2 şi apoi executaţi-l cu comanda:

```
--> exec('tema2.sci')
```

Observaţi ce se întâmplă dacă la sfârşitul fiecărei instrucţiuni adăugaţi terminatorul ";".

IMPORTANT: Comenzile necesare rezolvării temelor ce vor urma vor fi scrise în fişiere şi executate apoi cu comanda `exec`.

2.6.2 Operaţii de intrare/ieşire

• *Introducerea datelor*

Cea mai simplă metodă constă în utilizarea instrucţiunii de atribuire, ca în exemplul:

$a = 5$

În cazul unui program scris într-un fişier, este mai convenabil să se folosească funcţia `input`. Funcţia input se utilizează în atribuiri de forma:

variabila = `input`('text')

în care "variabila" este numele variabilei a cărei valoare va fi citită de la consolă, iar "text" este un şir de caractere ce va fi afişat, ajutând utilizatorul la identificarea informaţiei ce trebuie introdusă. De exemplu:

```
a = input('Introduceti valoarea variabilei a')
```

• *Inspectarea şi afişarea rezultatelor*

Pentru inspectarea valorilor variabilelor este suficient să fie invocat numele lor:

```
--> x
```

pentru ca interpretorul să afişeze valoarea lor.

Dacă se doreşte eliminarea afişării numelui variabilelor din faţa valorii sale, atunci se foloseşte funcţia `disp`:

```
--> disp(x)
```

Această funcţie poate fi folosită şi pentru afişarea textelor, de exemplu:

```
disp('Acest program calculeaza ceva ')
```

Formatul în care sunt afişate valorile numerice poate fi modificat de utilizator cu ajutorul instrucţiunii:

```
format
```

Operaţia de ieşire se poate realiza şi prin apelul funcţiei `printf` în instrucţiuni de forma:

`printf`('format',variabile)

în care "variabile" sunt variabilele care vor fi scrise în formatul corespunzător instrucţiunii, iar "format" este un şir de caractere ce descrie formatul de afişare. Sunt recunoscute următoarele construcţii, similare celor din limbajul C:

- `%f` scrierea numărului în format cu virgulă fixă;
- `%e` scrierea numărului în format cu exponent;
- `%g` scrierea numărului în formatul cel mai potrivit (%f sau %e).

Celelalte caractere întâlnite în şirul "format" sunt afişate ca atare, de exemplu:

```
printf(' Rezultatul este a = %g', a)
```

Afişarea rezultatelor se poate face şi grafic (vezi paragraful 5).

EXERCIŢIUL 16:	Scrieţi, într-un fişier, un program prietenos care va permite introducerea de la consola ***Scilab*** a două numere reale, va calcula suma lor, şi va afişa rezultatul în formatul cel mai potrivit

2.6.3 Structuri de control

• *Decizii*

Decizia simplă:

Pseudolimbaj	***Scilab***	Observaţii

dacă condiţie **atunci**
instrucţiuni
	`if` condiţie `then`
	instrucţiuni
	`end`
		"condiţie" este o expresie care este evaluată, iar dacă rezultatul este adevărat (T), atunci se execută "instrucţiuni", altfel se execută prima instrucţiune ce urmează după `end`.

Decizia cu alternativă:

Pseudolimbaj	***Scilab***	Observaţii

dacă condiţie **atunci**
instrucţiuni1
altfel
instrucţiuni2
	`if` condiţie `then`
	instrucţiuni1
	`else`
	instrucţiuni2
	`end`
		"condiţie" este o expresie care este evaluată, iar dacă rezultatul este adevărat (T), atunci se execută "instrucţiuni1", iar dacă rezultatul este fals (F), se execută "instrucţiuni2".

Decizia de tip selecţie:

Pseudolimbaj	***Scilab***	Observaţii

dacă condiţie1 **atunci**
instrucţiuni1
altfel dacă condiţie2
instrucţiuni2
altfel
instrucţiuni3
	`if` condiţie1 `then`
	instrucţiuni1
	`elseif` condiţie2
	instrucţiuni2
	`else`
	instrucţiuni3
	`end`
		Pot exista oricâte alternative de selecţie.

dacă expresie = expresie1
instrucţiuni1
altfel dacă expresie = expresie2
instrucţiuni2
altfel
instrucţiuni
	`select` expresie,
	`case` expresie1 `then`
	instrucţiuni1,
	`case` expresie2 `then`
	instrucţiuni2,
	`else` instrucţiuni,
	`end`
		Pot exista oricâte cazuri.
		"instrucţiuni1" sunt executate dacă expresie==expresie1, etc.

Observaţie: "instrucţiuni" pot fi scrise după `then`, pe aceeaşi linie. Cuvântul cheie `then` poate lipsi dacă "instrucţiuni" se scriu pe linie separată[6].

[6] Aceasta este sintaxa ***MATLAB***.

• *Cicluri*

Ciclul cu test iniţial:

Pseudolimbaj	***Scilab***	Observaţii
cât timp condiţie instrucţiuni	`while` condiţie instrucţiuni `end`	Se repetă corpul ciclului, adică "instrucţiuni", cât timp "condiţie" este adevărată. S-ar putea ca "instrucţiuni" să nu fie executate niciodată în timpul rulării programului.

Ciclul cu contor:

Ciclul cu contor are două forme, din care a doua este cea generală. Dacă "expresie" este o matrice, atunci "variabila" ia succesiv valorile coloanelor matricei. "instrucţiuni" nu sunt executate niciodată dacă vectorul "valin:pas:valfin" este incorect definit (vid) sau dacă "expresie" este matricea vidă.

Pseudolimbaj	***Scilab***	Observaţii
pentru contor = valin, valfin, pas instrucţiuni	`for` contor = valin : pas : valfin, instrucţiuni `end`	Forma a doua este cea generală.
	`for` variabila = expresie, instrucţiuni `end`	

Ieşirile forţate din cicluri se pot face cu instrucţiunea `break`.

EXERCIŢIUL 17: Scrieţi un program care să determine cel mai mare număr întreg n pentru care 10^n poate fi reprezentat în ***Scilab***. Indicaţie: folosiţi pentru testare constanta `%inf`.

2.6.4 Funcţii

Funcţiile sunt proceduri ***Scilab*** (termenii "macro", "funcţie" sau "procedură" au aceeaşi semnificaţie).

Definirea funcţiilor în fişiere

De obicei funcţiile sunt definite în fişiere şi "încărcate" în ***Scilab*** cu comanda `getf`. Un fişier conţinând o astfel de funcţie trebuie să înceapă astfel:

`function`[$y_1, \ldots, y_n$] = nume_funcţie ($x_1, \ldots, x_m$)

unde y_i sunt variabilele de ieşire, calculate în funcţie de variabilele de intrare x_j şi, eventual, de alte variabile existente în ***Scilab*** în momentul execuţiei funcţiei.

EXERCIŢIUL 18:

Editaţi un fişier numit "numefis.sci" cu următorul conţinut:

```
function [x,y] = combin(a,b)
        x = a + b
        y = a - b
```

şi un fişier numit "main.sci" cu următorul conţinut:

```
getf('numefis.sci')
a = input('Introduceti a');
b = input('Introduceti b');
[c,d] = combin(a,b)
printf(' Suma numerelor  a = %g si b = %g este
          a + b = %g',a,b,c);
printf(' Diferenta numerelor a = %g si b = %g
          este a - b  = %g',a,b,d);
```

Executaţi în ***Scilab*** comenzile din "main.sci":

```
--> exec('main.sci');
```

Comentaţi.

EXERCIŢIUL 19:

Scrieţi (într-un fişier) o funcţie care să calculeze produsul scalar a doi vectori. Scrieţi un program ***Scilab*** care să permită introducerea de la tastatură a doi vectori de dimensiune egală şi care să afişeze produsul lor scalar.

Definirea "on-line" a funcţiilor

Funcţiile se pot defini şi "on-line" cu comanda `deff`.

EXERCIŢIUL 20:

a) Cu ajutorul comenzii `help` aflaţi ce face comanda `fcontour`.
b) Executaţi şi comentaţi următoarele comenzi ***Scilab***:

```
--> deff('[x] = cerc(u,v)','x = sqrt(u^2+v^2)')
--> fcontour(-1:0.1:1,-1:0.1:1,cerc,4)
```

c) Definiţi "on-line" funcţia "combin" din exerciţiul 18.

2.7 Exerciţiul final - câmpul unei sarcini punctiforme situate în vid

Acest exerciţiu urmăreşte exersarea facilităţilor de postprocesare ale programului, necesare temelor viitoare.

EXERCIŢIUL 21:

Fie o sarcină punctiformă $q = 10^{-10}$ C, situată în vid ($\varepsilon_0 = 8.8 \cdot 10^{-12}$ F/m), în punctul de coordonate $(x, y, z) = (0, 0, 0)$.
Să se vizualizeze, în planul $z = 0$, linii echipotenţiale şi vectori câmp electric cu ajutorul comenzilor `contour` şi `champ`. Pentru aceasta, se vor determina valorile potenţialului şi ale componentelor câmpului într-o mulţime discretă de puncte din spaţiu, plasate în nodurile unei grile generate de un vector de abscise şi un vector de ordonate.
Se recomandă parcurgerea următorilor paşi:

- Se va scrie o funcţie care calculează, într-un punct oarecare, potenţialul unei sarcini punctuale situată într-un punct oarecare din spaţiu;
- Se va scrie o funcţie care calculează, într-un punct oarecare, vectorul componentelor câmpului electric al unei sarcini punctuale, situată într-un punct oarecare din spaţiu;
- Se va scrie un program principal, care să permită introducerea datelor problemei de la tastatură şi care să afişeze linii echipotenţiale şi vectori câmp într-un anumit domeniu din planul $z = 0$.

Figurile 2.2, 2.3, 2.4 prezintă rezultatele unui astfel de program în cazul în care grila a fost extrem de rară, generată cu ajutorul vectorilor:

```
d = 0.5
xmin = -d; xmax = d
ymin = -d; ymax = d
pasx = 2*d/3
pasy = 2*d/3

abscise = xmin:pasx:xmax
ordonate = ymin:pasy:ymax
```

Observaţie: în cazul spectrelor desenate cu comanda `champ`, vectorii sunt desenaţi (translataţi) astfel încât mijlocul lor coincide cu punctul de calcul (de aceea, figura 2.2 apare oarecum ciudat).

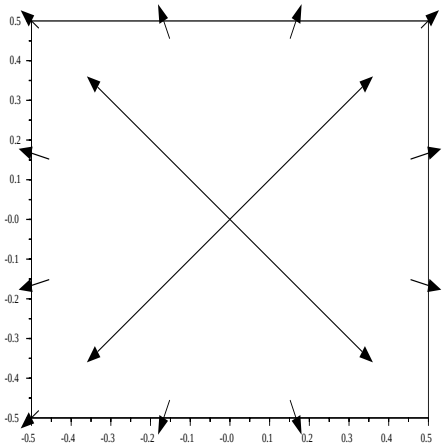


Fig. 2.2. Efectul comenzii **champ**: vectorii au o lungime proporţională cu valoarea modulului lor

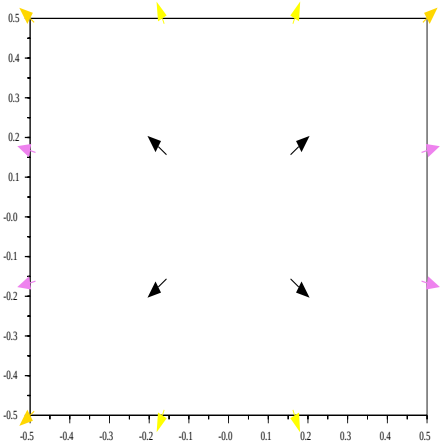


Fig. 2.3. Efectul comenzii **champ1**: vectorii au lungimi egale şi sunt coloraţi conform valorii modulului lor

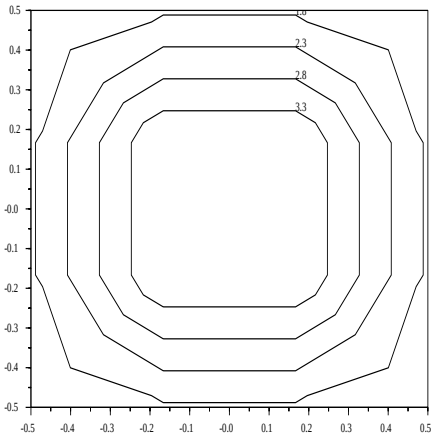


Fig. 2.4. Efectul comenzii **contour**; curbele au această formă datorită grilei extrem de rare folosite

EXERCIŢIUL 22:

a) Observaţi şi comentaţi alura echipotenţialelor pentru diferite grade de fineţe ale grilei folosite.
b) Cum trataţi cazul în care unul din punctele grilei coincide cu punctul în care se află sarcina?
c) Scrieţi un program care să genereze o grilă adaptată problemei, astfel încât liniile echipotenţiale să aibă racordări cât mai "dulci".

Capitolul 3

Concepte de bază.

Această temă urmăreşte evidenţierea conceptelor fundamentale folosite de metoda diferenţelor finite şi de metoda reziduurilor ponderate.

Metoda diferenţelor finite constă în rezolvarea ecuaţiei diferenţiale a problemei în punctele unei reţele de discretizare, folosind aproximarea derivatelor prin diferenţe finite.

Metoda reziduurilor ponderate – în particular, implementarea ei cu elemente finite – constă în rezolvarea unei formulări integrale (slabe) a ecuaţiei problemei, folosind aproximarea polinomială pe porţiuni a soluţiei.

Ambele metode pot fi studiate în termenii teoriei clasice a aproximărilor polinomiale.

3.1 Aproximări polinomiale

3.1.1 Cazul unidimensional

• *Interpolarea polinomială* ***Lagrange*** este una din cele mai cunoscute aproximări pentru o funcţie $f : [a, b] \to \mathbb{R}$, cunoscută într-un număr de puncte din $\Omega = [a, b]$. Reamintim următoarea teoremă:

TEOREMA 1 *Fie o diviziune a intervalului* $[a, b]$ *în* n *intervale definite prin punctele:*

$a = x_0 < x_1 < \ldots < x_n = b$

şi fie o mulţime de numere $\{f(x_i)\}_{i=0}^n$. *Atunci există un polinom unic* $P_n(x)$ *care poate fi exprimat în funcţie de numerele* $\{f(x_i)\}_{i=0}^n$ *şi de polinoamele de bază:*

$$l_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j} \tag{3.1}$$

şi anume:

$$P_n(x) = \sum_{i=0}^{n} l_i(x) f(x_i) \tag{3.2}$$

astfel încât $P_n(x_i) = f(x_i)$. *Mai mult, dacă funcţia* f *este de clasă* $C^{n+1}[a,b]$, *atunci eroarea este:*

$$E_n(x) = f(x) - P_n(x) = \frac{w_n(x)}{(n+1)!}\frac{\mathrm{d}^{n+1}f}{\mathrm{d}x^{n+1}}(\xi) \tag{3.3}$$

unde $x_0 < \xi < x_n$, $w_n(x) = (x-x_0)(x-x_1)\cdots(x-x_n)$.

Observaţie: în cazul unei reţele echidistante, eroarea este $E_n(x) = Kh^{n+1}$, unde K este o constantă independentă de pasul h.

EXERCIŢIUL 1:

a) Ce grad are polinomul Lagrange $l_i(x)$ dat de (3.1) ?
b) Ce valori ia un polinom Lagrange în nodurile grilei căreia îi este asociat?

EXERCIŢIUL 2:

În directorul ∼/modelare/tema3, editaţi un fişier numit "lagrange.sci" care să conţină definiţia unei funcţii de tipul:

```
function [l] = lagrange(x,var,i)
```

unde `x` este un vector care reprezintă diviziunea intervalului $[a,b]$, `i` reprezintă indicele nodului căruia îi este asociat polinomul Lagrange, iar `var` reprezintă un vector de abscise, cu valori între a şi b, în care va fi evaluat polinomul Lagrange asociat nodului `i`. Funcţia întoarce un vector linie, cu valorile polinomului Lagrange $l_i(x)$ evaluat în punctele vectorului linie `var`.

Indicaţie: iată un pseudocod posibil pentru această funcţie:

```
funcţie [lgr] = lagrange(vector x,var,  întreg i)
        ;calculează numărul de intervale ale diviziunii
        n = .....
        ;calculează lungimea vectorului absciselor
        m = .....
        pentru k = 1, m
                lgr(1,k) = 1;
                pentru j = 1, (n + 1)
                        dacă j ≠ i atunci
                                lgr(1,k) = lgr(1,k)*(var(k)-x(j))/(x(i)-x(j))
```

EXERCIŢIUL 3:

În directorul ∼/modelare/tema3, editaţi un fişier numit "ex3.sci" cu următorul conţinut:

```
clear;
getf('lagrange.sci');

x = [1 2 4]   // diviziunea
pasx = 0.1
n = length(x) - 1;
abscise = x(1): pasx: x(n+1);

xsetech([0,0,0.5,0.5])
ordonate1 = lagrange(x,abscise,1);
plot2d(abscise',ordonate1');

xsetech([0,0.5,0.5,0.5])
ordonate2 = lagrange(x,abscise,2);
plot2d(abscise',ordonate2');

xsetech([0.5,0,0.5,0.5])
ordonate3 = lagrange(x,abscise,3);
plot2d(abscise',ordonate3');
```

Executaţi acest program şi comentaţi-l.

EXERCIŢIUL 4:

Adăugaţi la fişierul "ex3.sci" comenzile:

```
xset("window",1);
xsetech([0,0,1,1])
plot2d([abscise;abscise;abscise]',
       [ordonate1;ordonate2;ordonate3]');
```

a) Ce efect au aceste comenzi?
b) Ce reprezintă figura 3.1?

EXERCIŢIUL 5:

Figura 3.2 reprezintă funcţiile Lagrange asociate unei anumite diviziuni a intervalului $[a, b]$.
a) Care este intervalul şi cum este realizată diviziunea lui?
b) Copiaţi fişierul "ex3.sci" în fişierul "ex5.sci". Modificaţi-l pe acesta din urmă astfel încât să obţineţi graficul din figura 3.2.

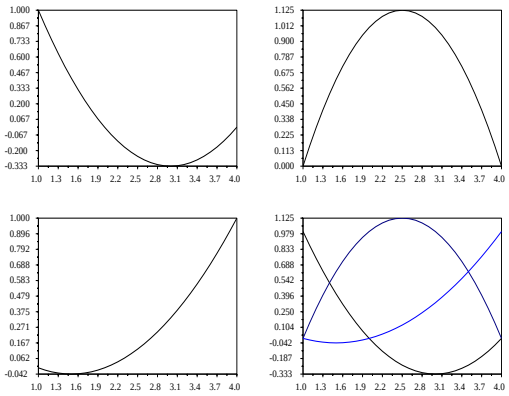


Fig. 3.1. Vezi exerciţiul 4

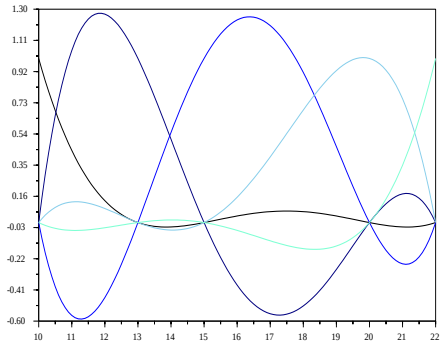


Fig. 3.2. Vezi exerciţiul 5

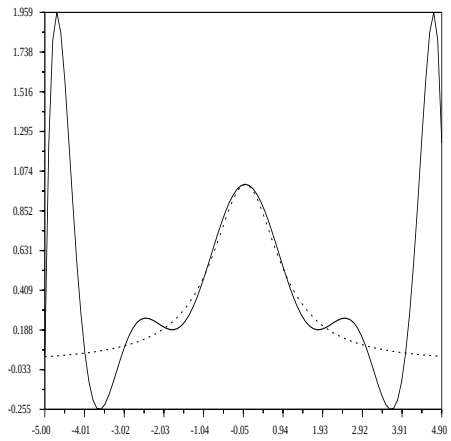
Până acum aţi văzut cum arată funcţiile de bază ale interpolării Lagrange. Să vedem acum cum arată polinomul de interpolare.

EXERCIŢIUL 6:

Scrieţi un fişier "ex6.sci" cu următorul conţinut:

```
clear;
getf('lagrange.sci');
x = [-5 -4 -3 -2 -1 0 1 2 3 4 5];   //diviziunea
y = (x^2+1)^(-1);
pasx = 0.1
n = length(x) - 1;
abscise = x(1): pasx: x(n+1);
ordonate_adevarate = (abscise^2+1)^(-1);
m = length(abscise);
L = zeros(n+1,m);
for i = 1:n+1,
        L(i,:) = lagrange(x,abscise,i);
end
ordonate = L'*y';
plot2d([abscise;abscise]',
       [ordonate ordonate_adevarate']);
```

a) Comentaţi acest program. Executaţi-l.
b) Ce reprezintă figura 3.3? Încercaţi să daţi o explicaţie pentru acest fenomen (numit fenomenul Runge). Cum ar putea fi evitat acest fenomen?



EXERCIŢIUL 7:

a) Copiaţi fişierul "ex6.sci" în "ex7.sci".
b) Modificaţi programul astfel încât în fereastra grafică să se afişeze în partea de sus acelaşi grafic ca la exerciţiul 6, iar în partea de jos să se deseneze graficul erorii.

Fig. 3.3. Vezi exerciţiul 6

• *Interpolarea polinomială* ***Hermite***[1] construieşte un polinom de interpolare a unei funcţii $f : [a,b] \to \mathbb{R}$, cunoscută într-un număr de puncte din $\Omega = [a,b]$. În plus faţă de interpolarea Lagrange, este cunoscută şi derivata funcţiei în "nodurile" diviziunii (grilei). Reamintim următoarea teoremă:

TEOREMA 2 *Fie o diviziune a intervalului* $[a,b]$ *în* n *intervale definite prin punctele:*

$a = x_0 < x_1 < \ldots < x_n = b$

şi fie o mulţime de perechi de numere $\{f(x_i), \frac{\mathrm{d}f}{\mathrm{d}x}(x_i)\}_{i=0}^n$. *Atunci există un polinom unic* $H_n(x)$ *care poate fi exprimat în funcţie de numerele* $\{f(x_i), \frac{\mathrm{d}f}{\mathrm{d}x}(x_i)\}_{i=0}^n$ *şi de polinoamele de bază:*

$$h_i^0(x) = \frac{Q_i(x)}{Q_i(x_i)}\left[1 - \frac{\frac{\mathrm{d}Q_i}{\mathrm{d}x}(x_i)}{Q_i(x_i)}(x - x_i)\right] \tag{3.4}$$

$$h_i^1(x) = \frac{Q_i(x)}{Q_i(x_i)}(x - x_i) \tag{3.5}$$

unde $Q_i(x) = (l_i(x))^2$, *şi anume:*

$$H_n(x) = \sum_{i=0}^{n}\left[h_i^0(x)f(x_i) + h_i^1(x)\frac{\mathrm{d}f}{\mathrm{d}x}(x_i)\right] \tag{3.6}$$

de grad cel mult $2n+1$ *astfel încât* $H_n(x_i) = f(x_i)$ *şi* $\frac{\mathrm{d}H_n}{\mathrm{d}x}(x_i) = \frac{\mathrm{d}f}{\mathrm{d}x}(x_i)$. *Mai mult, dacă funcţia* f *este de clasă* $C^{2(n+1)}[a,b]$, *atunci eroarea este:*

$$E_n(x) = f(x) - H_n(x) = \frac{(w_n(x))^2}{(2n+2)!}\frac{\mathrm{d}^{2(n+1)}f}{\mathrm{d}x^{2(n+1)}}(\xi) \tag{3.7}$$

unde $x_0 < \xi < x_n$, $w_n(x) = (x - x_0)(x - x_1)\cdots(x - x_n)$.

[1]Partea legată de interpolarea Hermite (exerciţiile 8 ÷ 11 şi 18) este facultativă.

EXERCIŢIUL 8:

În directorul ∼/modelare/tema3, editaţi un fişier numit "hermite.sci" care să conţină definiţia funcţiei:

```
function [h0, h1] = hermite(x,var,i)
```

unde `x` este un vector care reprezintă diviziunea intervalului $[a, b]$, `i` reprezintă indicele nodului căruia îi sunt asociate polinoamele Hermite, iar `var` reprezintă un vector de abscise, cu valori între între a şi b, în care se va evalua polinomul Hermite h_i^0, respectiv h_i^1, asociat nodului `i`. Funcţia întoarce doi vectori linie, cu valorile polinomului Hermite $h_i^0(x)$, respectiv $h_i^1(x)$, evaluat în punctele vectorului linie `var`.

EXERCIŢIUL 9:

În directorul ∼/modelare/tema3, editaţi un fişier numit "ex9.sci" care să conţină comenzile ***Scilab*** pentru a desena într-o fereastră polinoamele Lagrange în partea de sus şi polinoamele Hermite în partea de jos.

EXERCIŢIUL 10:

Scrieţi un program care să realizeze interpolarea Hermite a funcţiei lui Runge $1/(1+x^2)$. Programul va afişa pe acelaşi grafic funcţia reală, rezultatul interpolării Lagrange şi rezultatul interpolării Hermite.

EXERCIŢIUL 11:

Modificaţi programul de la exerciţiul 10 astfel încât în partea de jos a ferestrei grafice să fie desenate pe acelaşi grafic eroarea interpolării Lagrange şi eroarea interpolării Hermite.

• *Interpolări polinomiale pe porţiuni.* Interpolările prezentate până acum au folosit un singur polinom care să aproximeze funcţia pe **întreg** domeniul dorit. Ele se numesc de aceea *interpolări globale.* Aţi văzut că prin creşterea fineţei grilei de discretizare a domeniului, nu este îmbunătăţită eroarea de aproximare (uneori, dimpotrivă, apar fenomene nedorite chiar în cazul în care funcţia de aproximat are o alură cuminte, ca funcţia lui Runge). Dacă funcţia de aproximat are un relief foarte accidentat, atunci de asemenea aproximările polinomiale globale nu sunt precise.

Din aceste motive, mult mai necesare pentru rezolvarea numerică a problemelor de câmp electromagnetic sunt interpolările pe porţiuni.

EXERCIŢIUL 12:

a) Folosind funcţia `lagrange` pe care aţi construit-o la exerciţiul 2, scrieţi un program care să deseneze cele două polinoame Lagrange asociate diviziunii `[a b]`.
b) Comentaţi figura 3.4.

Să considerăm acum problema aproximării unei funcţii pe un interval $[a, b]$ divizat astfel: $a = x_0 < x_1 < x_2 = b$. Să notăm cu $0, 1, 2$ nodurile grilei, asociate respectiv coordo-

natelor x_0, x_1, x_2. În locul unei interpolări globale, care să folosească cele trei polinoame Lagrange asociate diviziunii $[x_0\, x_1\, x_2]$, vom face o interpolare pe porţiuni, în care vom folosi polinoamele Lagrange asociate diviziunii $[x_0\, x_1]$ şi polinoamele Lagrange asociate diviziunii $[x_1\, x_2]$. Astfel, pe intervalul $[x_0, x_1]$ funcţia f este aproximată prin:

$$P(x) = l_0(x)f(x_0) + l_1(x)f(x_1) \tag{3.8}$$

unde l_0 şi l_1 sunt funcţiile Lagrange asociate nodului 0 şi respectiv 1 şi diviziunii $[x_0\, x_1]$, iar pe intervalul $[x_1, x_2]$ funcţia f este aproximată prin:

$$P(x) = l_1(x)f(x_1) + l_2(x)f(x_2) \tag{3.9}$$

unde l_1 şi l_2 sunt funcţiile Lagrange asociate nodului 1 şi respectiv 2 şi diviziunii $[x_1\, x_2]$.

EXERCIŢIUL 13: Comentaţi figura 3.5.

EXERCIŢIUL 14:

a) În fişierul "ex14a.sci" scrieţi un program care să realizeze interpolarea funcţiei e^x pe diviziunea $[x_0\ x_1] = [0\ \ 1]$, cu folosirea formulei (3.8). Reprezentaţi grafic cei doi termeni ai sumei (3.8). Într-o altă fereastră reprezentaţi polinomul P de interpolare şi funcţia e^x.
b) Copiaţi fişierul "ex14a.sci" în "ex14b.sci".
Modificaţi fişierul "ex14b.sci" astfel încât să se realizeze interpolarea funcţiei e^x pe diviziunea [0 1 2], astfel: pe intervalul $[0, 1]$ să fie folosită relaţia (3.8), iar pe intervalul $[1, 2]$ să fie folosită relaţia (3.9).

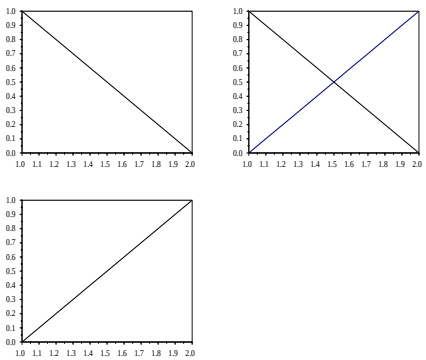


Fig. 3.4. Vezi exerciţiul 12

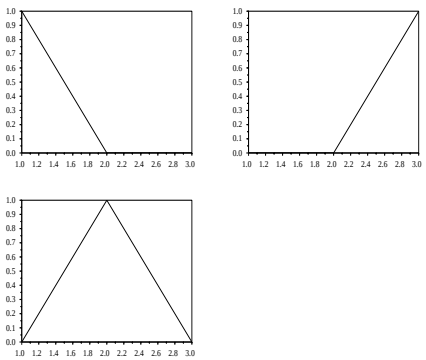


Fig. 3.5. Vezi exerciţiul 13

EXERCIŢIUL 15:

La fişierul "lagrange.sci" construit la exerciţiul 2, adăugaţi o funcţie de tipul:

```
function [lgr] = lagrange_portiuni(x,var,i)
```

unde `x` este un vector care reprezintă diviziunea intervalului $[a, b]$, `i` reprezintă indicele nodului căruia îi este asociat polinomul Lagrange definit pe porţiuni, iar `var` reprezintă un vector de abscise între a şi b în care va fi evaluat polinomul Lagrange pe porţiuni asociat nodului `i`. Funcţia întoarce un vector linie, cu valorile polinomului Lagrange pe porţiuni $l_i(x)$ evaluat în punctele vectorului linie `var`. (Indicaţie: în definiţia acestei funcţii apelaţi funcţia `lagrange`.)

EXERCIŢIUL 16:

a) Scrieţi într-un fişier "ex16.sci" un program care să deseneze polinoamele Lagrange pe porţiuni, asociate unei diviziuni neuniforme.
b) Comentaţi figura 3.6.

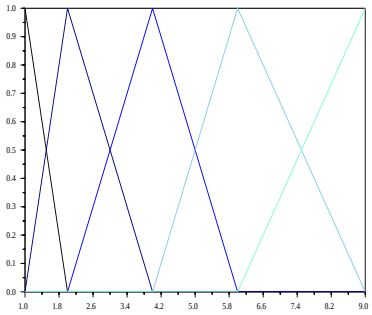


Fig. 3.6. Vezi exerciţiul 16

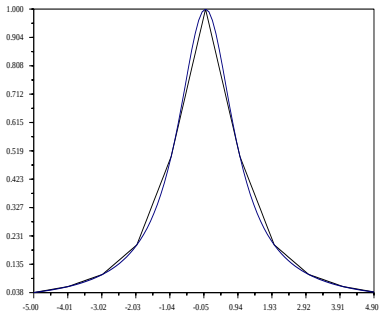


Fig. 3.7. Interpolarea Lagrange pe porţiuni a funcţiei lui Runge

EXERCIŢIUL 17:

Copiaţi fişierul "ex6.sci" în "ex17.sci". Înlocuiţi apelul funcţiei `lagrange` cu `lagrange_portiuni`. Executaţi acest nou program. Rezultatul ar trebui să arate ca în figura 3.7. Comentaţi rezultatul, comparându-l cu rezultatul exerciţiului 6.

EXERCIŢIUL 18:

a) Explicaţi cum credeţi că se va face interpolarea Hermite pe porţiuni;
b) Încercaţi să faceţi o comparaţie între interpolarea Lagrange şi interpolarea Hermite.

3.1.2 Cazul bidimensional

Teoria generală a interpolării în două sau mai multe dimensiuni este mult mai dificilă decât cea unidimensională.

Iată cum poate fi extinsă *interpolarea Lagrange* în cazul bidimensional.

Fie o funcţie $f : \Omega \to \mathbb{R}$, unde Ω este un domeniu bidimensional $[a, b] \times [c, d]$, cunoscută într-un număr de puncte din Ω. Presupunem că funcţia este cunoscută în nodurile unei grile date de diviziunea pe Ox:

$$a = x_0 < x_1 < \ldots < x_n = b$$

şi de diviziunea pe Oy:

$$c = y_0 < y_1 < y_2 < \ldots y_m = d.$$

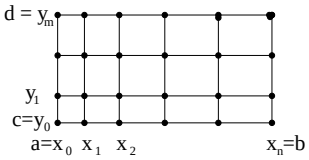


Fig. 3.8. Grila de discretizare a lui Ω

Păstrând fixă una din variabilele independente, vom interpola unidimensional funcţia de-a lungul celeilalte variabile. De exemplu, pentru un y_j fixat al diviziunii pe Oy rezultă:

$$f(x, y_j) = \sum_{i=0}^{n} l_i(x) f(x_i, y_j) + E_n(x, y_j). \tag{3.10}$$

Interpolând apoi după y, rezultă:

$$f(x, y) = \sum_{i=0}^{n} \sum_{j=0}^{m} l_i(x) l_j(y) f(x_i, y_j) + E_{m,n}(x, y). \tag{3.11}$$

În consecinţă, funcţiile Lagrange asociate unei grile bidimensionale sunt:

$$l_{ij}(x, y) = l_i(x) l_j(y) = \left(\prod_{\substack{k=0 \\ k \neq i}}^{n} \frac{x - x_k}{x_i - x_k} \right) \left(\prod_{\substack{k=0 \\ k \neq j}}^{m} \frac{y - y_k}{y_j - y_k} \right). \tag{3.12}$$

EXERCIŢIUL 19:

Copiaţi fişierul "lagrange.sci" în "lagrange2D.sci". Modificaţi funcţia `lagrange` în `lagrange2D`, cu următorii parametri:

```
function [lgr] = lagrange2D(x,y,varx,vary,i,j)
```

unde `x` şi `y` sunt vectori care reprezintă diviziunea intervalului $[a, b]$, respectiv $[c, d]$, `i` şi `j` definesc în mod univoc nodul căruia îi este asociat polinomul Lagrange, iar `varx` şi `vary` reprezintă un vector de abscise între a şi b, respectiv un vector de ordonate între c şi d, care definesc punctele în care va fi evaluat polinomul Lagrange asociat nodului respectiv din grila bidimensionlă. Funcţia întoarce o matrice care conţine valorile polinomului Lagrange $l_{i,j}(x, y)$ evaluat în punctele determinate de abscisele `varx` şi ordonatele `vary`.

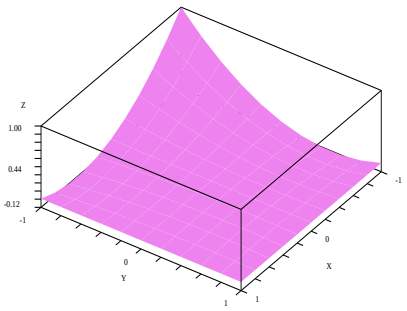
EXERCIŢIUL 20:

Editaţi un fişier "ex20.sci" cu următorul conţinut:

```
clear;
getf('lagrange2D.sci');

x = [-1 0 1]
y = [-1 0 1]
pasx = 0.2;
pasy = 0.2
n1 = length(x);
n2 = length(y);
abscise = x(1):pasx:x(n1);
ordonate = y(1):pasy:y(n2);
[cote] = lagrange2D(x,y,abscise,ordonate,2,3);
plot3d(abscise,ordonate,cote);
```

Executaţi şi comentaţi acest program.



EXERCIŢIUL 21:

a) Ce reprezintă figura 3.9?
b) Scrieţi un program care să deseneze într-o fereastră grafică a ***Scilab***-ului, cele 9 funcţii Lagrange asociate diviziunii de la exerciţiul 20.

Fig. 3.9. Vezi exerciţiul 21

EXERCIŢIUL 22:

Scrieţi expresia polinomului de interpolare Lagrange pentru funcţia $\exp(x+y)$, în domeniul $[-1,1]\times[-1,1]$, pentru o grilă uniformă pe ambele axe, având în total 9 noduri.

EXERCIŢIUL 23:

Folosind drept model cazul unidimensional, extindeţi interpolarea Lagrange pe porţiuni la cazul bidimensional.

3.2 Clasificarea sistemelor de ecuaţii cu derivate parţiale

Din punct de vedere matematic, rezolvarea problemelor de câmp electromagnetic implică rezolvarea unor sisteme de ecuaţii cu derivate parţiale[2].

[2]În limba engleză se prescurtează PDE - "Partial Differential Equations".

Acestea se clasifică în trei categorii:

- **hiperbolice** – exemplu tipic:
$$\frac{\partial^2 u}{\partial t^2} = v^2 \frac{\partial^2 u}{\partial x^2} \tag{3.13}$$
Aceasta este ecuaţia undelor, v reprezentând viteza de propagare.
- **parabolice** – exemplu tipic:
$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left(D\frac{\partial u}{\partial x}\right) \tag{3.14}$$
Aceasta este ecuaţia de difuzie, D reprezentând coeficientul de difuzie.
- **eliptice**: – exemplu tipic:
$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = \rho(x, y) \tag{3.15}$$
Aceasta este ecuaţia Poisson, ρ reprezintă densitatea sursei. Dacă $\rho = 0$, ecuaţia se numeşte Laplace.

Ecuaţia (3.15) reprezintă ecuaţia satisfăcută de o funcţie staţionară $u(x, y)$, definită pe o regiune de interes. Rezolvarea acestei probleme necesită cunoaşterea valorilor funcţiei (sau ale derivatei acesteia) pe frontiera domeniului.

EXERCIŢIUL 24:

Descrieţi frontiera următoarelor domenii:
a) un interval $[a, b]$ al axei reale;
b) un disc plan de rază r;
c) o placă subţire de formă pătrată;
d) o sferă de rază r;
e) un cilindru de rază r şi de înălţime H.
Dacă un domeniu Ω aparţine spaţiului n–dimensional, $\Omega \subset \mathbb{R}^n$, cărui spaţiu îi aparţine frontiera sa $\partial\Omega$?

Ecuaţiile (3.13) şi (3.14) descriu evoluţii în timp. Ele reprezintă probleme Cauchy care necesită în plus cunoaşterea valorilor iniţiale.

Există şi alte tipuri de ecuaţii diferenţiale. Pentru ilustrarea metodei diferenţelor finite şi a elementelor finite se va folosi în cele ce urmează o ecuaţie mult mai simplă decât oricare din tipurile prezentate mai sus (vezi problema 1 din paragraful 3.3.3).

3.3 Aproximarea cu diferenţe finite

3.3.1 Ideea metodei. Formule de aproximare în cazul unidimensional.

Ideea principală din spatele metodei diferenţelor finite este aceea de a aproxima derivatele care apar în ecuaţii cu ajutorul unei mulţimi de valori ale funcţiei dintr-un număr de puncte numite "noduri".

Cea mai populară metodă de a genera aceste aproximaţii este prin intermediul seriilor Taylor. Există şi alte posibilităţi de a genera aceste aproximaţii, de exemplu folosind formule de interpolare.

Vom reaminti câteva rezultate pentru cazul aproximării derivatelor unei funcţii unidimensionale $f : [a, b] \to \mathbb{R}$.

Vom nota cu x_i punctele diviziunii intervalului $[a, b]$, $f_i = f(x_i)$, cu $Df_i = \frac{\partial f}{\partial x}(x_i)$ şi cu $D^2 f_i = \frac{\partial^2 f}{\partial x^2}(x_i)$.

Cazul unei diviziuni uniforme: $x_{i+1} - x_i = h$

- Aproximarea derivatelor cu o eroare de ordinul[3] h:

$$\text{(progresivă)} \quad Df_i = \frac{f_{i+1} - f_i}{h} + O(h) \tag{3.16}$$

$$\text{(regresivă)} \quad Df_i = \frac{f_i - f_{i-1}}{h} + O(h) \tag{3.17}$$

- Aproximarea derivatelor (centrate) cu o eroare de ordinul h^2:

$$Df_i = \frac{f_{i+1} - f_{i-1}}{2h} + O(h^2) \tag{3.18}$$

$$D^2 f_i = \frac{f_{i+1} - 2f_i + f_{i-1}}{h^2} + O(h^2) \tag{3.19}$$

- Aproximarea derivatelor (centrate) cu o eroare de ordinul h^4:

$$Df_i = \frac{f_{i-2} - 8f_{i-1} + 8f_{i+1} - f_{i+2}}{12h} + O(h^4) \tag{3.20}$$

$$D^2 f_i = \frac{-f_{i-2} + 16f_{i-1} - 30f_i + 16f_{i+1} - f_{i+2}}{12h^2} + O(h^4) \tag{3.21}$$

Cazul unei diviziuni neuniforme: $x_{i+1} - x_i = \alpha h$, $x_i - x_{i-1} = \beta h$

- Aproximarea derivatei de ordinul unu cu o eroare de ordinul h^2:

$$Df_i = \frac{\beta^2 f_{i+1} + (\alpha^2 - \beta^2) f_i - \alpha^2 f_{i-1}}{\alpha\beta(\alpha + \beta)h} + O(h^2) \tag{3.22}$$

- Aproximarea derivatei de ordinul doi cu o eroare de ordinul[4] h:

$$D^2 f_i = \frac{\beta f_{i+1} - (\alpha + \beta) f_i - \alpha f_{i-1}}{\alpha\beta(\alpha + \beta)\frac{h^2}{2}} + O(h) \tag{3.23}$$

[3]Este posibilă doar pentru derivatele de ordinul unu.

[4]Derivata de ordinul doi nu poate fi aproximată cu o eroare de ordinul h^2 dacă sunt folosite numai punctele x_{i-1}, x_i şi x_{i+1}.

3.3.2 Algoritmul metodei

Aplicarea metodei diferenţelor finite pentru rezolvarea oricărei ecuaţii diferenţiale presupune realizarea următorilor paşi:

- **Pasul 1:** Se alege o schemă de diferenţe finite pentru aproximarea derivatelor din ecuaţii şi se rescrie ecuaţia ca ecuaţie cu diferenţe finite.
- **Pasul 2:** Se stabileşte grila de discretizare şi se scrie ecuaţia discretizată pentru fiecare nod.
- **Pasul 3:** Se rezolvă sistemul de ecuaţii pentru determinarea valorilor necunoscute în nodurile grilei.
- **Pasul 4:** Calculul altor valori, în afara celor din noduri, se face prin interpolare.

Se obişnuieşte să se spună că paşii 1 şi 2 reprezintă etapa de *preprocesare*, pasul 3 reprezintă *rezolvarea*, iar pasul 4 *postprocesarea*.

3.3.3 Exerciţii

Problema 1:

Newton a arătat că, în unele cazuri, viteza de răcire a unui corp (derivata temperaturii în raport cu timpul) este proporţională cu diferenţa dintre temperatura T a obiectului şi temperatura T_e a mediului înconjurător:

$$\frac{\partial T}{\partial t} + k\left[T(t) - T_e\right] = 0 \tag{3.24}$$

unde k este un factor de proporţionalitate şi t este timpul.

Se consideră o placă infinit extinsă în direcţiile axelor Oy şi Oz, de grosime foarte mică, astfel încât temperatura este instantaneu uniformă în interiorul ei. Placa este brusc introdusă într-un mediu cu temperatură constantă.

a) Folosind metoda diferenţelor finite, să se calculeze variaţia în timp a temperaturii în intervalul $[0, 1]$, presupunând $k = 2$ şi $T_e = 0.5$. Condiţia iniţială este $T(0) = 1$.

b) Să se compare rezultatul numeric cu rezultatul exact:

$$T(t) = T(0)e^{-kt} + T_e(1 - e^{-kt}). \tag{3.25}$$

EXERCIŢIUL 25:

Scrieţi un fişier "problema1.sci" cu următorul conţinut:

```
clear;   // este bine ca aceasta comanda sa fie
         // prima linie din orice fisier
         // scris in Scilab

getf('grid_1D.sci');
getf('prep_mdf.sci');

disp(' Tema3 - problema 1');
// datele problemei:
Te = 0.5;
k = 2;
a = 0;
b = 1;
initial = 1;

// ----------- preprocesare -----------
[n, deltat] = grid_uniform(a,b);
[M,p] = asambleaza(k, Te, initial, deltat, n)
```

EXERCIŢIUL 26:

a) Scrieţi un fişier "grid_1D.sci" pentru definirea unei funcţii `grid_uniform`, care să aibă ca parametri de intrare capetele intervalului $[a, b]$, să ceară utilizatorului numărul `n` de segmente în care va fi împărţit acest interval, şi să întoarcă acest număr şi pasul de discretizare `deltat`.
b) Testaţi corectitudinea acestei funcţii astfel: în fişierul "problema1.sci" comentaţi linia cu al doilea `getf` precum şi ultima linie, şi apoi executaţi acest program.

EXERCIŢIUL 27:

Deduceţi forma discretizată a ecuaţiei de rezolvat, folosind pentru aproximarea derivatei de ordinul unu o schemă cu diferenţe progresive (formula (3.16)).

EXERCIŢIUL 28:

Scrieţi un fişier "prep_mdf.sci" pentru definirea funcţiei `asambleaza`. Aceasta va avea ca parametri de intrare datele problemei şi date despre grila de discretizare. Parametrii de ieşire vor fi matricea coeficienţilor `M` şi vectorul termenilor liberi `p`.

Indicaţie: iată un pseudocod posibil:

```
funcţie [M, p] = asambleaza(real k,Te,initial,deltat,  întreg n)
        ; crează o matrice M cu elemente nule
          . . . . . .
        ; crează un vector p cu elemente nule
          . . . . . .
        pentru i = 1, n
                M(i+1,i+1) = 1,
                M(i+1,i) = k*deltat – 1,
                p(i+1) = k*Te*deltat,
        ; impunerea condiţiilor iniţiale
        M(1,1) = 1;
        p(1) = initial;
retur
```

EXERCIŢIUL 29:

Completaţi fişierul "problema1.sci" cu:

```
// ------------------ rezolvare ------------
T = inv(M)*p
```

Verificaţi corectitudinea funcţiei `asambleaza`: pentru o împărţire în 4 segmente rezultatul trebuie să fie: `T = [ 1, 0.75, 0.625, 0.5625, 0.53125 ]` .

EXERCIŢIUL 30:

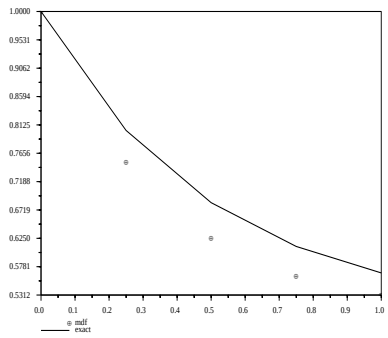
Cum este matricea `M`:
a) rară sau plină?
b) simetrică sau nesimetrică?
c) singulară sau nesingulară?
d) bine sau prost condiţionată (folosiţi comanda `cond`)?

EXERCIŢIUL 31:

Completaţi fişierul "problema1.sci" cu:

```
// -------- postprocesare --------
deff('[solutie] = Texact(t)',
      'solutie=(initial-Te)*exp(-k*t)+Te');
for i = 1:n+1,
      t(i) = a + (i-1)*deltat;
end
Tex = Texact(t);
plot2d([t t],[T Tex],[3 -1],'121','mdf@exact');
```

Ce reprezintă figura 3.10?



EXERCIŢIUL 32: Se poate evita folosirea matricei M? Propuneţi o altă metodă de implementare (scrieţi un pseudocod).

Fig. 3.10. Vezi exerciţiul 32

3.4 Metoda reziduurilor ponderate

3.4.1 Ideea metodei. Funcţii de bază în cazul unidimensional.

Ideea metodei

Există două abordări posibile atunci când se doreşte aplicarea unei metode variaţionale pentru rezolvarea unei ecuaţii cu derivate parţiale. Să notăm această ecuaţie cu:

$$LT = f, \tag{3.26}$$

unde L este un operator convenabil ataşat problemei de rezolvat. Pentru buna formulare a problemei, trebuie date şi condiţiile pe frontiera domeniului $\mathcal{D}$ al funcţiei T, precum şi condiţiile iniţiale dacă este cazul.

O abordare este cunoscută sub numele de ***metoda Rayleigh-Ritz*** şi este metoda variaţională clasică. Ideea acestei metode este următoarea:

- Ecuaţiei satisfăcute de funcţia necunoscută T i se asociază o anumită *funcţională* $\mathcal{F}(T)$ al cărei punct de minim coincide cu soluţia ecuaţiei diferenţiale.
- Se alege un sistem complet de *funcţii de bază*, liniar independente, Ψ_j, $j = 1, \ldots, n$ iar soluţia exactă este aproximată ca:

$$T \approx \widetilde{T} = \sum_{j=1}^{n} c_j \Psi_j. \tag{3.27}$$

- Constantele c_j se determină din condiţia de minimizare a funcţionalei $\mathcal{F}(\widetilde{T})$.

Această metodă nu poate fi utilizată decât dacă este găsită expresia funcţionalei, lucru posibil pentru multe ecuaţii care descriu fenomene staţionare. În caz contrar, se foloseşte metoda reziduurilor ponderate.

O altă abordare este ***metoda reziduurilor ponderate*** care poate fi descrisă pe scurt astfel:

- Dacă $\widetilde{T}$ este o soluţie aproximativă, atunci se defineşte *reziduul*:

$$R(\widetilde{T}) = L\,\widetilde{T} - f. \tag{3.28}$$

- Ca la metoda Rayleigh-Ritz, se alege un sistem complet de funcţii de bază liniar independente, Ψ_j, $j = 1, \ldots, n$ iar soluţia exactă este aproximată ca:

$$\widetilde{T} = \sum_{j=1}^{n} c_j \Psi_j. \tag{3.29}$$

- În general, reziduul (3.28) are valoarea zero doar pentru soluţia exactă. Cum $\widetilde{T}$ este doar o aproximaţie a acesteia, reziduul este în general nenul şi constituie o măsură a erorii introduse de aproximarea funcţiei T prin relaţia (3.29).

  Pentru determinarea constantelor c_j se impune condiţia de anulare a reziduului *în medie* pe întregul domeniu al problemei. Media este definită sub o formă generală ca integrala pe domeniu a funcţiei reziduu (3.28) înmulţită cu anumite funcţii w_i numite *funcţii pondere*:

$$\int_\Omega R(\widetilde{T}) w_i \,\mathrm{d}\Omega = 0 \qquad i = 1, \ldots, n. \tag{3.30}$$

  Relaţia (3.30) extinde noţiunea binecunoscută de proiecţie a unui vector pe o axă, la cea de proiecţie a unei funcţii R pe altă funcţie w_i, ambele având acelaşi domeniu de definiţie Ω. De aceea se spune că metoda reziduurilor ponderate este o *metodă de proiecţie*.

  În metoda reziduurilor ponderate, relaţia (3.30) se scrie pentru cele n funcţii pondere, rezultând astfel un sistem de n ecuaţii din care pot fi calculaţi cei n coeficienţi necunoscuţi c_j.

  În funcţie de modul în care se aleg funcţiile pondere, metoda are mai multe variante:

  - *Metoda colocaţiei*: funcţiile pondere sunt funcţii Dirac; prin (3.30) se impune deci anularea reziduului în anumite puncte din domeniu;
  - *Metoda celor mai mici pătrate*: funcţiile pondere coincid cu funcţia reziduu;
  - *Metoda Galerkin*: funcţiile pondere coincid cu funcţiile de bază. Aceasta este cea mai des folosită metodă şi de aceea o vom folosi în exerciţiile ce urmează.

Funcţii de bază în cazul unidimensional

Cele mai folosite funcţii de bază provin din aproximări polinomiale pe porţiuni. Cazul cel mai simplu este cel în care polinoamele sunt *de gradul 1.*

Să considerăm diviziunea x_i, cu $i = 1, \ldots, n+1$ a intervalului $[a, b]$ pe care este definită funcţia necunoscută T. În paragraful 3.1.1 aţi văzut cum se face interpolarea polinomială pe porţiuni în *cazul unidimensional.* Graficele polinoamelor de interpolare Lagrange pe porţiuni (fig. 3.5) arată că, pentru un nod, există două polinoame care iau valoarea 1 în

nod şi zero în nodurile vecine. Aceste două polinoame definesc funcţia de bază asociată nodului i:

$$\Psi_i(x) = \begin{cases} \frac{x-x_{i-1}}{x_i-x_{i-1}} & x \in [x_{i-1}, x_i] \\ \frac{x-x_{i+1}}{x_i-x_{i+1}} & x \in [x_i, x_{i+1}] \\ 0 & x \notin [x_{i-1}, x_{i+1}] \end{cases} \tag{3.31}$$

pentru $i \neq 1$ şi $i \neq n+1$, respectiv

$$\Psi_1(x) = \begin{cases} \frac{x-x_2}{x_1-x_2} & x \in [x_1, x_2] \\ 0 & x \notin [x_1, x_2] \end{cases} \tag{3.32}$$

$$\Psi_{n+1}(x) = \begin{cases} \frac{x-x_n}{x_{n+1}-x_n} & x \in [x_n, x_{n+1}] \\ 0 & x \notin [x_n, x_{n+1}] \end{cases} \tag{3.33}$$

Observăm că funcţia de bază[5] asociată nodului i este diferită de zero numai în intervalele adiacente nodului i: $(x_{i-1}, x_i]$ şi $[x_i, x_{i+1})$. Fiecare interval $[x_i, x_{i+1}]$ se numeşte *element finit*. De aceea, această metodă mai este cunoscută şi sub numele de *metoda elementelor finite*.

Cu alegerea (3.31) $\div$ (3.33) a funcţiilor de bază, necunoscutele c_j din relaţia (3.29) reprezintă chiar valorile funcţiei necunoscute T în nodurile reţelei de discretizare, $c_j = T_j$:

$$\tilde{T} = \sum_{j=1}^{n+1} T_j \Psi_j. \tag{3.34}$$

În consecinţă, necunoscutele problemei sunt chiar valorile T_j ale funcţiei în nodurile reţelei de discretizare.

EXERCIŢIUL 33: Demonstraţi afirmaţia precedentă.

În cazul bidimensional elementele finite sunt de obicei triunghiuri sau patrulatere, iar în cazul tridimensional sunt de obicei tetraedre sau prisme (triunghiulare sau patrulatere).[6]

Funcţiile de bază Ψ_i pot fi polinoame de grad mai mare decât 1, construite tot cu ajutorul polinomului Lagrange $l_i(x)$ (vezi teorema 1). În practică se folosesc funcţii de bază de grad cel mult 3.

3.4.2 Algoritmul metodei

Aplicarea metodei reziduurilor ponderate (elementelor finite) pentru rezolvarea oricărei ecuaţii diferenţiale presupune parcurgerea următorilor paşi:

[5] Funcţiei de bază asociată unui nod i se mai spune şi *funcţie de formă* ("shape function") sau *funcţie "pălărie"* ("hat function"). Această din urmă denumire provine de la forma acestei funcţii în cazul bidimensional.

[6] În limba engleză se spune că discretizarea se face cu o reţea "mesh" în cazul metodei elementelor finite şi cu o grilă "grid" in cazul metodei diferenţelor finite.

- **Pasul 1:** Se aleg funcţiile de bază. Acesta este cel mai important pas pentru că de această alegere depinde acurateţea aproximării necunoscutei.
- **Pasul 2:** Se aproximează funcţia necunoscută printr-o relaţie de tip (3.34). Dacă funcţiile de bază au fost alese ca fiind funcţii polinomiale pe porţiuni, atunci acest pas presupune împărţirea domeniului pe care este definită funcţia necunoscută în elemente finite.
- **Pasul 3:** Se aleg funcţiile pondere.
- **Pasul 4:** Se asamblează sistemul de ecuaţii.
- **Pasul 5:** Se rezolvă sistemul de ecuaţii determinând coeficienţii necunoscuţi (valorile necunoscutei în nodurile reţelei de discretizare, în cazul variantei Galerkin).
- **Pasul 6:** Se calculează alte mărimi de interes asociate problemei.

Paşii 1÷4 reprezintă etapa de *preprocesare*, pasul 5 este etapa de *rezolvare* iar pasul 6 reprezintă *postprocesarea*.

3.4.3 Forma discretizată a ecuaţiei de rezolvat

Vom rezolva acum problema 1 cu metoda elementelor finite în **varianta Galerkin**, cu **funcţii de bază liniare**. Aceasta este echivalent cu:

- alegerea la **pasul 1** a funcţiilor liniare pe porţiuni (3.31) ca funcţii de bază;
- discretizarea, la **pasul 2**, a domeniului de definiţie $\Omega = [a, b]$ al necunoscutei, în n subintervale (elemente finite $[x_i, x_{i+1}]$, cu $i = 1, \ldots, n$); pentru simplitate, se va considera diviziunea echidistantă, în care toate elementele finite au aceeaşi lungime;
- construirea funcţiei aproximate conform relaţiei (3.34);
- alegerea, la **pasul 3**, a funcţiilor pondere ca fiind aceleaşi ca funcţiile de bază (conform metodei Galerkin).

Cu aceste alegeri, principala dificultate a preprocesării o constituie **pasul 4** al algoritmului, şi anume asamblarea sistemului de ecuaţii.

Asamblarea sistemului de ecuaţii

Sub acest nume este cunoscută operaţia de construire a sistemului de ecuaţii (3.30), cu $i = 1, \ldots, n + 1$.

Vom renota variabila t cu x şi vom folosi, pentru simplificarea scrierii, notaţia $C = kT_e$. Ecuaţia de rezolvat (3.24) devine:

$$T'(x) + kT(x) = C, \tag{3.35}$$

în care dependenţa de x va fi, în cele ce urmează, subînţeleasă.

Reziduul definit de relaţia (3.28) are, pentru această problemă, forma:

$$R(T) = T' + kT - C. \tag{3.36}$$

Proiecţia reziduului pe funcţiile pondere, dată de (3.30), se particularizează sub forma (cu $w_i = \Psi_i$):

$$\int_\Omega (T' + kT - C)\Psi_i \, \mathrm{d}\Omega = 0. \tag{3.37}$$

Odată explicitată integrala pe domeniul de definiţie al necunoscutei, această relaţie va reprezenta ecuaţia cu numărul i dintr-un sistem de $n+1$ ecuaţii cu $n+1$ necunoscute T_j — valorile temperaturii în nodurile reţelei de discretizare.

Se observă că relaţia (3.37) conţine termenul $\int_\Omega T'\Psi_i \, \mathrm{d}\Omega$. O **etapă esenţială** în rezolvarea oricărei ecuaţii diferenţiale prin metoda elementelor finite este **transformarea** acestui termen astfel încât nu necunoscuta T, ci funcţia de bază Ψ_i pe care se face proiecţia, să apară derivată. Această transformare se poate face folosind *formula de integrare prin părţi*:

$$\int_{\Omega=[a,b]} T'\Psi_i \, \mathrm{d}\Omega = -\int_\Omega T\Psi_i' \mathrm{d}\Omega + \int_\Omega (T\Psi_i)' \mathrm{d}\Omega = -\int_\Omega T\Psi_i' \mathrm{d}\Omega + (T\Psi_i)|_a^b \,. \tag{3.38}$$

Relaţia (3.37) devine:

$$-\int_\Omega T\Psi_i' \mathrm{d}\Omega + \int_\Omega kT\Psi_i \, \mathrm{d}\Omega - \int_\Omega C\Psi_i \, \mathrm{d}\Omega + (T\Psi_i)|_a^b = 0. \tag{3.39}$$

Cum domeniul Ω este împărţit în elementele finite $e = 1, \ldots, n$, integralele pe Ω pot fi scrise ca sume de integrale pe elementele finite:

$$\sum_{e=1}^{n} \left(-\int_e T\Psi_i' \mathrm{d}\Omega + \int_e kT\Psi_i \, \mathrm{d}\Omega - \int_e C\Psi_i \, \mathrm{d}\Omega \right) + (T\Psi_i)|_a^b = 0. \tag{3.40}$$

Se observă că ultimul termen din relaţia (3.40) nu mai este o integrală pe întregul domeniu Ω, ci este o expresie definită pe frontiera $\partial\Omega = \{a, b\}$ a acestuia.

În consecinţă, pot fi identificate două *tipuri* de contribuţii la ecuaţiile (3.40):

— contribuţia elementelor finite;
— contribuţia frontierei.

Aceste două tipuri de contribuţii vor fi analizate separat în cele ce urmează.

Determinarea contribuţiilor elementelor finite

Necunoscuta T va fi aproximată conform relaţiei (3.34), în care j variază de la 1 la $n+1$, deoarece reţeaua de discretizare conţine $n+1$ noduri.

În relaţia (3.40) scrisă pentru nodurile interioare domeniului ($i = 2, \ldots, n$), termenul $(T\Psi_i)|_a^b$ nu intervine, deoarece $\Psi_i(a) = \Psi_i(b) = 0$. De aceea, vom elimina acest termen în cele ce urmează.

Astfel, relaţia (3.40) (în care se consideră nulă contribuţia frontierei) devine:

$$\sum_{e=1}^{n}\left(-\int_e\sum_{j=1}^{n+1}T_j\Psi_j\Psi_i'\mathrm{d}\Omega+\int_e k\sum_{j=1}^{n+1}T_j\Psi_j\Psi_i\,\mathrm{d}\Omega-\int_e C\Psi_i\,\mathrm{d}\Omega\right)=0, \tag{3.41}$$

sau, interschimbând poziţia sumelor după e şi după j în primii doi termeni integrali şi ţinând cont de faptul că T_j sunt coeficienţi necunoscuţi constanţi, deci pot fi scoşi în afara integralelor:

$$\sum_{j=1}^{n+1}T_j\left(\sum_{e=1}^{n}\left(-\int_e\Psi_j\Psi_i'\mathrm{d}\Omega+k\int_e\Psi_j\Psi_i\,\mathrm{d}\Omega\right)\right)=\sum_{e=1}^{n}\int_e C\Psi_i\,\mathrm{d}\Omega. \tag{3.42}$$

Să reamintim faptul că relaţia (3.42) este ecuaţia cu numărul i din sistemul de ecuaţii cu necunoscutele T_j. Se constată că membrul stâng al relaţiei (3.42) este o combinaţie liniară a necunoscutelor T_j, în timp ce membrul drept nu conţine necunoscute. Ecuaţia (3.42) va avea deci, după calculul integralelor care intervin, forma standard a unei ecuaţii dintr-un sistem de ecuaţii algebrice liniare:

$$\sum_{j=1}^{n+1}T_j m_{ij}=p_i. \tag{3.43}$$

Asamblarea sistemului de ecuaţii constă în calculul coeficienţilor m_{ij} şi al termenilor liberi p_i. Comparând (3.42) cu (3.43) se constată că aceşti coeficienţi au, în cazul metodei elementelor finite, expresiile:

$$m_{ij}=\sum_{e=1}^{n}\left(-\int_e\Psi_j\Psi_i'\mathrm{d}\Omega+k\int_e\Psi_j\Psi_i\,\mathrm{d}\Omega\right)=\sum_{e=1}^{n}(-\mathcal{I}_1+k\mathcal{I}_2)=\sum_{e=1}^{n}m_{ij}^e, \tag{3.44}$$

dacă $i\neq j$, şi respectiv

$$m_{ii}=\sum_{e=1}^{n}\left(-\int_e\Psi_i\Psi_i'\mathrm{d}\Omega+k\int_e\Psi_i\Psi_i\,\mathrm{d}\Omega\right)=\sum_{e=1}^{n}(-\mathcal{I}_1{}'+k\mathcal{I}_2')=\sum_{e=1}^{n}m_{ii}^e, \tag{3.45}$$

iar termenul liber are expresia:

$$p_i=\sum_{e=1}^{n}\int_e C\Psi_i\,\mathrm{d}\Omega=\sum_{e=1}^{n}C\mathcal{I}_3=\sum_{e=1}^{n}p_i^e. \tag{3.46}$$

Calculul expresiilor m_{ij}, m_{ii} şi p_i este mult simplificat de observaţia făcută anterior, aceea că funcţia de formă (de bază) Ψ_i este nulă în toate elementele finite $e=[x_k,x_{k+1}]$ care nu conţin nodul i ($i\neq k$ şi $i\neq k+1$), după cum rezultă din însăşi definiţia (3.31) a funcţiei Ψ_i.

Ca urmare:
$\mathcal{I}_1$ este nenulă doar pentru elementele e care conţin nodul j ($e\ni j$)
$\mathcal{I}_2$ este nenulă doar pentru elementele e care conţin nodurile i şi j ($e\ni j$ şi $e\ni i$)

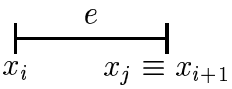
$\mathcal{I}_1'$ este nenulă doar pentru elementele e care conţin nodul i ($e \ni i$)
$\mathcal{I}_2'$ este nenulă doar pentru elementele e care conţin nodul i ($e \ni i$).

De aceea, în (3.44) suma pentru toate elementele e se reduce la $\sum_{e \ni i \text{ şi } e \ni j}$, iar în (3.45) şi (3.46) doar la $\sum_{e \ni i}$.

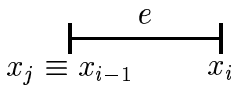
EXERCIŢIUL 34:

a) Ce grad are polinomul Ψ_i?
b) Câte elemente contribuie, în cazul problemei unidimensionale de faţă, la coeficientul m_{ij}? Dar la coeficientul m_{ii}?

Să calculăm contribuţia unui element finit e la coeficientul m_{ij} al matricei, contribuţie notată cu m_{ij}^e în relaţia (3.44). Conform observaţiilor de mai sus, m_{ij}^e e nenulă dacă elementul e conţine nodurile i şi j. În concluzie, pot apărea următoarele două situaţii (se va nota cu h dimensiunea[7] elementului finit):



Formulele (3.31) se particularizează ca:
$\Psi_i = \frac{x-x_j}{x_i-x_j} = -\frac{x-x_j}{h}$
$\Psi_j = \frac{x-x_i}{x_j-x_i} = \frac{x-x_i}{h}$
Derivata funcţiei de bază Ψ_i este:
$\Psi_i' = -\frac{1}{h}$
$\mathcal{I}_1 = \int_e \Psi_j \Psi_i' \mathrm{d}\Omega = \int_{x_i}^{x_j} -\frac{x-x_i}{h^2}\,\mathrm{d}x = -\frac{1}{2}$
$\mathcal{I}_2 = \int_e \Psi_j \Psi_i\,\mathrm{d}\Omega = \int_{x_i}^{x_j} -\frac{(x-x_i)(x-x_j)}{h^2}\,\mathrm{d}x = \frac{h}{6}$
$\mathcal{I}_3 = \int_e \Psi_i\,\mathrm{d}\Omega = \int_{x_i}^{x_j} -\frac{x-x_j}{h}\,\mathrm{d}x = \frac{h}{2}$
$\mathcal{I}_1' = \int_e \Psi_i \Psi_i' \mathrm{d}\Omega = \int_{x_i}^{x_j} \frac{x-x_j}{h^2}\,\mathrm{d}x = -\frac{1}{2}$
$\mathcal{I}_2' = \int_e \Psi_i^2\,\mathrm{d}\Omega = \int_{x_i}^{x_j} \frac{(x-x_j)^2}{h^2}\,\mathrm{d}x = \frac{h}{3}$



Formulele (3.31) se particularizează ca:
$\Psi_i = \frac{x-x_j}{x_i-x_j} = \frac{x-x_j}{h}$
$\Psi_j = \frac{x-x_i}{x_j-x_i} = -\frac{x-x_i}{h}$
Derivata funcţiei de bază Ψ_i este:
$\Psi_i' = \frac{1}{h}$
$\mathcal{I}_1 = \int_e \Psi_j \Psi_i' \mathrm{d}\Omega = \int_{x_j}^{x_i} \frac{x-x_i}{h^2}\,\mathrm{d}x = \frac{1}{2}$
$\mathcal{I}_2 = \int_e \Psi_j \Psi_i\,\mathrm{d}\Omega = \int_{x_j}^{x_i} -\frac{(x-x_i)(x-x_j)}{h^2}\,\mathrm{d}x = \frac{h}{6}$
$\mathcal{I}_3 = \int_e \Psi_i\,\mathrm{d}\Omega = \int_{x_j}^{x_i} \frac{x-x_j}{h}\,\mathrm{d}x = \frac{h}{2}$
$\mathcal{I}_1' = \int_e \Psi_i \Psi_i' \mathrm{d}\Omega = \int_{x_j}^{x_i} \frac{x-x_j}{h^2}\,\mathrm{d}x = \frac{1}{2}$
$\mathcal{I}_2' = \int_e \Psi_i^2\,\mathrm{d}\Omega = \int_{x_j}^{x_i} -\frac{(x-x_j)^2}{h^2}\,\mathrm{d}x = \frac{h}{3}$

EXERCIŢIUL 35:

a) Demonstraţi formulele de calcul pentru integralele $\mathcal{I}_1$, $\mathcal{I}_2$, $\mathcal{I}_3$, $\mathcal{I}_1'$, $\mathcal{I}_2'$.
Indicaţie: Folosiţi interpretarea geometrică a integralei, pentru a calcula rapid o parte dintre aceste expresii.
b) Cum puteţi reţine uşor semnele care apar în formulele acestor integrale?

În concluzie, un element finit $e = [x_i, x_j]$ (sau $e = [x_j, x_i]$) contribuie la coeficienţii liniei i din matricea sistemului cu valorile:

$$m_{ij}^e = -\mathcal{I}_1 + k\mathcal{I}_2 = \begin{cases} 1/2 + kh/6, & i,j \in e, \quad j > i \\ -1/2 + kh/6, & i,j \in e, \quad j < i \\ 0, & i \notin e \text{ sau } j \notin e \end{cases} \tag{3.47}$$

[7] în cazul de faţă, lungimea

$$m_{ii}^e = -\mathcal{I}_1{}' + k\mathcal{I}_2{}' = \begin{cases} 1/2 + kh/3, & i,j \in e, \quad j > i \\ -1/2 + kh/3, & i,j \in e, \quad j < i \\ 0, & i \notin e \text{ sau } j \notin e \end{cases} \tag{3.48}$$

$$p_i^e = C\mathcal{I}_3 = \begin{cases} Ch/2, & i \in e \\ 0, & i \notin e \end{cases} \tag{3.49}$$

În relaţiile de mai sus, $h = |x_i - x_j|$.

Adunând contribuţiile tuturor elementelor e se obţin (conform relaţiilor (3.44) – (3.46)) coeficienţii sistemului de ecuaţii:

$$m_{ij} = \sum_e m_{ij}^e, \quad m_{ii} = \sum_e m_{ii}^e, \quad p_i = \sum_e p_i^e. \tag{3.50}$$

Determinarea contribuţiilor frontierei

Să notăm cu $\mathcal{E}$ expresia ultimului termen din relaţia (3.40):

$$\mathcal{E} = (T\Psi_i)\big|_a^b = T(b)\Psi_i(b) - T(a)\Psi_i(a). \tag{3.51}$$

În punctele de pe frontiera domeniului Ω (plasate la capetele intervalului), singurele funcţii de formă nenule sunt $\Psi_1(a) = 1$ şi $\Psi_{n+1}(b) = 1$. În consecinţă, $\mathcal{I}_4$ va avea valori nenule doar pentru prima şi ultima ecuaţie din sistem:

$$\mathcal{E} = \begin{cases} -T(a) = -T_1, & i = 1 \\ T(b) = T_{n+1}, & i = n+1 \\ 0, & 2 \le i \le n \end{cases} . \tag{3.52}$$

În ce priveşte prima ecuaţie din sistem, valoarea $T_1 = T(0)$ reprezintă condiţia iniţială a ecuaţiei de rezolvat, care va fi impusă direct:

$$m_{11} = 1; \quad p_1 = T_1; \quad m_{1j} = 0 \text{ pentru } j \neq 1. \tag{3.53}$$

De aceea, singura parte a frontierei care va contribui la coeficienţii sistemului este ultimul nod, $x_{n+1} = b$.

Cum valoarea T_{n+1} este necunoscută, $\mathcal{E}$ va contribui la coeficientul diagonal $m_{n+1,n+1}$ al liniei $n+1$ din sistemul de ecuaţii, cu valoarea contribuţiei de frontieră

$$m_{n+1,n+1}^{\ f} = 1, \tag{3.54}$$

iar coeficientul diagonal al ultimei linii are expresia

$$m_{n+1,n+1} = m_{n+1,n+1}^{\ f} + \sum_e m_{n+1,n+1}^e,$$

unde al doilea termen reprezintă suma contribuţiilor elementelor finite care conţin nodul $n+1$ (calculate cu relaţia (3.48) cazul $j < i$).

EXERCIŢIUL 36:

Până acum, aţi văzut care este contribuţia unui element e la coeficienţii **liniei** i din matricea sistemului de ecuaţii.
a) La câţi coeficienţi ai matricei (eventual plasaţi pe alte linii) contribuie elementul e?
b) Care sunt expresiile acestor contribuţii?
Indicaţie: Scrieţi o relaţie asemănătoare cu (3.42), corespunzătoare liniei j din matrice.

EXERCIŢIUL 37:

Cum se modifică raţionamentele de până acum dacă diviziunea intervalului $[a, b]$ nu este uniformă?

Observaţia 1

Să considerăm un nod i **interior** domeniului $\Omega = [a, b]$ (deci cu $2 \leq i \leq n$). Să scriem ecuaţia (3.43) corespunzătoare acestui nod. Ştim deja că o mare parte din coeficienţii m_{ij} ai liniei i sunt nuli, mai precis relaţia (3.43) are forma:

$$m_{i,i-1}T_{i-1} + m_{ii}T_i + m_{i,i+1}T_{i+1} = p_i. \tag{3.55}$$

Pentru o reţea de discretizare echidistantă, lungimea h este aceeaşi pentru oricare element finit. Folosind relaţiile (3.47) $\div$ (3.49) şi adunând conform (3.50) contribuţiile celor două elemente finite care contribuie la coeficienţi, se obţine:

$$\left(-\frac{1}{2} + \frac{kh}{6}\right)T_{i-1} + \left(-\frac{1}{2} + \frac{kh}{3} + \frac{1}{2} + \frac{kh}{3}\right)T_i + \left(\frac{1}{2} + \frac{kh}{6}\right)T_{i+1} = \frac{Ch}{2} + \frac{Ch}{2}, \tag{3.56}$$

sau

$$\left(-\frac{1}{2} + \frac{kh}{6}\right)T_{i-1} + 2\frac{kh}{3}T_i + \left(\frac{1}{2} + \frac{kh}{6}\right)T_{i+1} = Ch. \tag{3.57}$$

EXERCIŢIUL 38:

a) Prin raţionamente similare, deduceţi ecuaţia corespunzătoare nodului $i = n + 1$.
b) Ce structură va avea matricea M a sistemului de ecuaţii rezultat?

Împărţind relaţia (3.57) la lungimea h a intervalului şi regrupând termenii, se obţine:

$$\frac{T_{i+1} - T_{i-1}}{2h} + k\frac{T_{i-1} + 4T_i + T_{i+1}}{6} = C. \tag{3.58}$$

Observaţia 2

Aceeaşi formulă (3.58) se obţine dacă se discretizează ecuaţia

$$T'(x_i) + kT(x_i) = C \tag{3.59}$$

în nodul $x = x_i$, astfel:

- se foloseşte formula de diferenţe centrate (3.18) pentru discretizarea derivatei $T'(x_i)$;
- se consideră $T(x)$ interpolat printr-un polinom Lagrange $l_2(x)$ (de gradul doi), pe diviziunea $[x_{i-1}\ x_i\ x_{i+1}]$;
- se adoptă pentru $T(x_i)$ valoarea medie a funcţiei $l_2(x)$ pe intervalul $[x_{i-1}, x_{i+1}]$, dată de:

$$T(x_i) = l_{2_{mediu}}[x_{i-1}, x_{i+1}] = \frac{1}{x_{i+1} - x_{i-1}} \int_{x_{i-1}}^{x_{i+1}} l_2(x)\,\mathrm{d}x.$$

EXERCIŢIUL 39:

a) Demonstraţi afirmaţia precedentă.
b) Se ştie de la cursul de metode numerice că aproximarea integralei unei funcţii $T(x)$ prin integrala polinomului Lagrange de gradul II are ordinul de precizie $O(h^2)$:

$$\int_{x_{i-1}}^{x_{i+1}} T(x)\,\mathrm{d}x = \int_{x_{i-1}}^{x_{i+1}} l_2(x)\,\mathrm{d}x + O(h^2).$$

Folosind această relaţie şi concluzia punctului a), estimaţi precizia schemei de elemente finite (3.57) (echivalentă cu (3.58)).

3.4.4 Exerciţii

Sunt acum disponibile toate "cărămizile" necesare scrierii algoritmului care rezolvă problema 1 prin metoda elementelor finite.

EXERCIŢIUL 40:

Completaţi fişierul "problema1.sci" cu liniile:

```
getf('prep_mef.sci');
. . .
[M1,p1] = asambleaza_mef(k,Te,initial,deltat,n)
```

b) Scrieţi un fişier "prep_mef.sci" pentru definirea funcţiei `asambleaza_mef`. Aceasta va avea aceiaşi parametri ca şi funcţia `asambleaza`, dar asamblarea matricei se va face, de această dată, conform metodei elementelor finite.

Indicaţie: iată un pseudocod posibil:

```
funcţie [M, p] = asambleaza_mef (real k,Te,initial,deltat,  întreg n)
        ; crează o matrice M şi un vector p cu elemente nule
        . . . . . .
        h = deltat
        C = k * Te
        pentru e = 1, n
                i = e,          ; nodurile elem. e
                j = e+1,
                ; contribuţii la linia i
                M(i,j) = M(i,j) + 1/2 + k*h/6,
                M(i,i) = M(i,i) + 1/2 + k*h/3,
                p(i) = p(i) + C*h/2,
                ; contribuţii la linia j
                 . . . . . .
        ; contribuţia frontierei la coeficientul diagonal al ultimei linii
        M(n+1,n+1) = M(n+1,n+1) + 1;
        ; impunerea condiţiilor iniţiale
        M(1,1) = 1;
        . . . . . ;  anularea celorlalte elemente nenule ale liniei 1
        p(1) = initial;
retur
```

Observaţie. Algoritmul propus are structura clasică a unui algoritm bazat pe metoda elementelor finite:

- Se parcurg elementele finite $e = 1, n$
  - Se identifică nodurile elementului finit curent (i, j).
  - Se adaugă contribuţia elementului finit curent la coeficienţii matricei.

O alternativă (rar folosită în practică) ar fi următoarea:

- Se parcurg nodurile $i = 1, n + 1$
  - Se identifică nodurile j şi elementele e vecine nodului curent i.
  - Se calculează contribuţia elementelor vecine la coeficienţii m_{ij}.

EXERCIŢIUL 41:

Completaţi fişierul "problema1.sci" cu:

```
// -------- rezolvare prin m.e.f. ---------
T1 = inv(M1)*p1
```

Verificaţi corectitudinea funcţiei `asambleaza_mef`: pentru o împărţire în 4 segmente rezultatul trebuie să fie:

```
T1 = [ 1, 0.801961, 0.6845933, 0.6102050,
       0.5688781 ].
```

EXERCIŢIUL 42:

Cum este matricea `M1`:
a) rară sau plină?
(comparaţi cu "previziunea" făcută la exerciţiul 38)
b) simetrică sau nesimetrică?
c) singulară sau nesingulară?
d) bine sau prost condiţionată?

EXERCIŢIUL 43:

Înlocuiţi ultima linie din fişierul "problema1.sci" cu:

```
plot2d([t t t],[T T1 Tex],[3 2 -1],'121', ...
       'mdf@mef@exact');
```

Executaţi programul şi comentaţi graficul obţinut.

EXERCIŢIUL 44:

a) Calculaţi valorile erorii absolute în nodurile reţelei:

$$er(i) = T(i) - Tex(i), \quad i = 1, n+1$$
$$er1(i) = T_1(i) - Tex(i), \quad i = 1, n+1$$

Reprezentaţi grafic aceste erori.
b) Calculaţi eroarea Cebîşev a soluţiilor aproximative T şi T_1 pe întregul interval de definiţie. Această eroare este definită ca:

$$e = \max_{i=1,n+1} |er(i)|, \quad e1 = \max_{i=1,n+1} |er1(i)|.$$

c) Cum pot fi reduse erorile fără a modifica algoritmii? Prin încercări succesive, determinaţi în ce condiţii eroarea absolută Cebîşev scade la ambele metode sub $\varepsilon = 0.01$.
d) Comparaţi cele două metode (metoda diferenţelor finite de ordinul I şi metoda elementelor finite în varianta Galerkin cu funcţii de bază liniare) din punctul de vedere al preciziei soluţiei.

EXERCIŢIUL 45:

a) Comparaţi cele două metode din punctul de vedere al timpului de calcul. Indicaţie: Pentru determinarea experimentală a timpului de calcul folosiţi instrucţiunea `timer`.
b) Cum ar putea fi redus timpul de calcul la metoda elementelor finite?

EXERCIŢIUL 46:

a) Poate fi evitată folosirea matricei `M` în cazul metodei elementelor finite? Justificaţi.

b) Propuneţi o variantă a algoritmului de asamblare a matricei M prin metoda elementelor finite, care să implementeze direct relaţia (3.57).

Capitolul 4

Metoda volumelor finite

Scopul acestei teme este să vă călăuzească pas cu pas în implementarea unui program ***Scilab*** care să rezolve, prin metoda volumelor finite, o problemă plan-paralelă de regim electrocinetic staţionar.

4.1 Enunţul problemei

Fie un conductor omogen, de conductivitate σ, situat într-un mediu perfect izolant. Conductorul are o dimensiune după axa Oz mult mai lungă decât celelalte două. Figura 4.1 reprezintă o secţiune perpendiculară pe direcţia dimensiunii foarte mari. Această secţiune are forma literei L, de dimensiuni a, b, A, B. Conductorul are două borne supraconductoare, una aflată la potenţial V_0 şi cealaltă aflată la potenţial nul. Se cer:

a) Să se reprezinte liniile echipotenţiale;

b) Să se reprezinte spectrul liniilor de curent;

c) Să se calculeze rezistenţa pe unitatea de lungime (în direcţia axei Oz) a acestui rezistor.

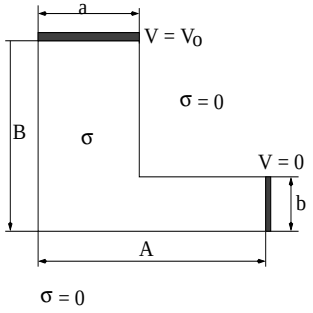


Fig. 4.1. Conductor în formă de L

TEMA de laborator:	Să se scrie un program ***Scilab*** care să rezolve, folosind metoda volumelor finite, problema enunţată mai sus. Programul va permite introducerea de la consolă a datelor geometrice şi de material, precum şi a unor parametri care să permită utilizarea unor grile de discretizare mai dese sau mai rare, uniforme sau neuniforme.

Paşii necesari rezolvării acestei teme sunt schiţaţi în cele ce urmează.

4.2 Formularea corectă a problemei

Înainte de orice, o problemă de câmp electromagnetic trebuie corect formulată, în conformitate cu teorema de unicitate.

Problema fiind de regim electrocinetic staţionar, formularea se va face în V – potenţial electrocinetic, unde $\mathbf{E} = -\text{grad}\, V$, $\mathbf{E}$ fiind intensitatea câmpului electric.

Ecuaţia de ordinul doi satisfăcută de potenţialul electrocinetic este:

$$-\text{div}\,(\sigma\,\text{grad}\, V) = 0 \tag{4.1}$$

unde $V : \mathcal{D} \to R$ este potenţialul definit pe domeniul bidimensional de forma literei L: $\mathcal{D}$ = MNPQRS (figura 4.2).

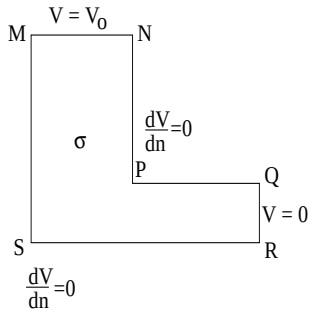


Fig. 4.2. Domeniul de calcul

Pentru buna formulare a problemei, trebuie impuse pentru potenţial condiţiile de frontieră. Frontierele supraconductoare (MN şi QR) au potenţial constant, şi în consecinţă pe ele trebuie impuse condiţii Dirichlet în concordanţă cu valorile potenţialului. Pe frontierele de lângă izolant (NPQ şi RSM), trebuie impuse condiţii Neumann nule[1].

Condiţiile de frontieră impuse potenţialului sunt:

$$V = V_0 \quad \text{pe} \quad \text{MN} \quad \text{(Dirichlet)} \tag{4.2}$$

$$V = 0 \quad \text{pe} \quad \text{QR} \quad \text{(Dirichlet)} \tag{4.3}$$

$$\frac{\partial V}{\partial n} = 0 \quad \text{pe} \quad \text{NPQ} \cup \text{RSM} \quad \text{(Neumann)} \tag{4.4}$$

4.3 Ideea metodei. Ecuaţii finale.

Metoda volumelor finite este o metodă numerică şi, în consecinţă, ea nu poate determina valorile potenţialului în toate punctele (o infinitate) domeniului de calcul. Valorile potenţialului sunt determinate (cu aproximaţie) într-un număr finit de puncte, situate în nodurile unei grile de discretizare. Dacă notăm cu n numărul de noduri ale grilei, aceasta înseamnă că vom calcula n valori ale potenţialului (problema are n necunoscute).

La orice metodă numerică de calcul al câmpului trebuie precizate două lucruri foarte importante:

- *Cum se aproximează funcţia necunoscută* în orice punct din domeniul de studiu în funcţie de cele n necunoscute?
- *Cum se aproximează ecuaţia* satisfăcută de necunoscuta principală a problemei? Scrierea acestei ecuaţii aproximative în cele n noduri va genera un sistem de n ecuaţii cu n necunoscute.

[1] În izolant nu există curent de conducţie, în consecinţă $\mathbf{J} \cdot \mathbf{n} = 0$ la frontiera cu izolantul, deci $-\sigma \text{grad}\, V \cdot \mathbf{n} = 0$, echivalent cu $-\sigma \frac{\partial V}{\partial n} = 0$ pe acele frontiere.

În cele ce urmează vom reaminti raţionamentul acestei metode pentru cazul regimului electrostatic, deoarece ecuaţia de ordinul doi asociată lui este mai generală decât cea asociată regimului electrocinetic staţionar, pentru că are sens fizic forma ei neomogenă:

$$-\text{div}\,(\varepsilon\,\text{grad}\,V) = \rho \tag{4.5}$$

unde $V : \mathcal{D} \to R$, ε este permitivitatea mediului, ρ este densitatea de volum a sarcinii electrice. Condiţiile de frontieră pot fi Dirichlet $V(M) = f$, unde f este cunoscută pentru o parte a frontierei domeniului, sau Neumann $\varepsilon\,\partial V/\partial n = g$, unde g este cunoscută pentru restul frontierei. Vom presupune că în domeniul de calcul nu există distribuţii superficiale de sarcină.

Cel mai simplu raţionament se face pe o grilă de discretizare carteziană, tip potrivit pentru problema enunţată mai sus.

4.3.1 Aproximarea câmpului electric

Pentru deducerea sistemului de ecuaţii, nu este nevoie decât să precizăm cum se aproximează intensitatea câmpului electric.

Pe o muchie a grilei de discretizare, intensitatea câmpului electric după direcţia dată de muchia respectivă se consideră constantă şi egală cu diferenţa de potenţial dintre nodurile muchiei raportată la lungimea muchiei. Această valoare se atribuie şi câte unei jumătăţi din fiecare celulă ce conţine muchia respectivă.

De exemplu, în figura 4.3, componenta lui **E** după direcţia **u** este aproximată prin $(V_0 - V_B)/\|OB\|$ în toată zona haşurată.

4.3.2 Aproximarea ecuaţiilor

Deducerea ecuaţiilor metodei volumelor finite se face scriind legea fluxului electric pe o suprafaţă închisă Σ ce înconjoară un nod al grilei de discretizare (figura 4.4).

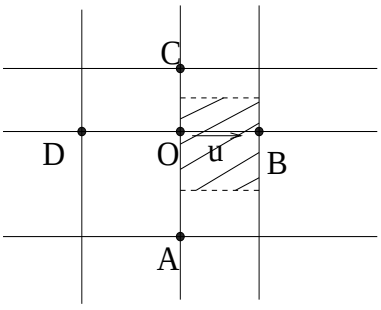


Fig. 4.3. Aproximarea câmpului

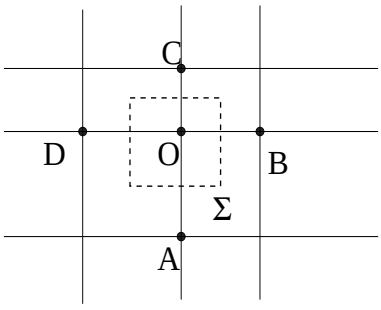


Fig. 4.4. Suprafaţa Σ pentru un nod interior

Tratarea nodurilor de pe frontieră se face în funcţie de tipul condiţiei de frontieră şi de forma frontierei. Pentru nodurile de pe frontiera Dirichlet, ecuaţia se înlocuieşte cu

simpla atribuire a valorii potenţialului nodului respectiv. Ecuaţia asociată nodurilor de pe frontiera Neumann are o formă specifică, în funcţie de poziţia nodului pe frontiera Neumann (pe o frontieră dreaptă sau într-un "colţ").

Ecuaţia asociată unui nod interior:

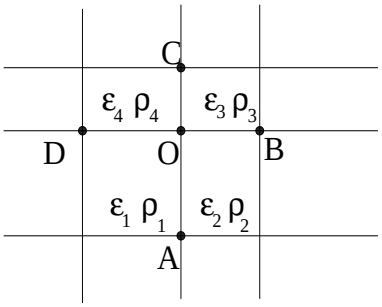


Fig. 4.5. Notaţii pentru un nod interior

Ecuaţia de ordin doi, discretizată, scrisă pentru punctul interior O (figura 4.5) este dată de relaţia (4.6) pentru regimul electrostatic. În regim electrocinetic staţionar, ε se înlocuieşte cu σ şi ρ cu zero.

Observaţie: stabilirea grilei pentru metoda volumelor finite se face astfel încât putem presupune că în fiecare celulă a grilei materialul este omogen (permitivitatea, conductivitatea sunt constante), şi, de asemenea, sursele de câmp (densitatea de volum a sarcinii) sunt constante în fiecare celulă.

În scrierea ecuaţiei asociată nodului O intervin numai valorile necunoscutei în nodurile A, B, C, D, proprietăţile de material şi sursele asociate celulelor înconjurătoare (notate cu indicii 1,2,3 şi 4), precum şi dimensiunile segmentelor grilei care concură în punctul O (segmentele $\|OA\| = h_A, \|OB\| = h_B, \|OC\| = h_C, \|OD\| = h_D$).

$$\begin{aligned} &V_O\left(\frac{\varepsilon_1 h_D + \varepsilon_2 h_B}{h_A} + \frac{\varepsilon_2 h_A + \varepsilon_3 h_C}{h_B} + \frac{\varepsilon_3 h_B + \varepsilon_4 h_D}{h_C} + \frac{\varepsilon_4 h_C + \varepsilon_1 h_A}{h_D}\right) - \\ &-V_A\frac{\varepsilon_1 h_D + \varepsilon_2 h_B}{h_A} - V_B\frac{\varepsilon_2 h_A + \varepsilon_3 h_C}{h_B} - V_C\frac{\varepsilon_3 h_B + \varepsilon_4 h_D}{h_C} - V_D\frac{\varepsilon_4 h_C + \varepsilon_1 h_A}{h_D} = \\ &= \rho_1\frac{h_A h_D}{2} + \rho_2\frac{h_B h_A}{2} + \rho_3\frac{h_C h_B}{2} + \rho_4\frac{h_D h_C}{2} \end{aligned} \tag{4.6}$$

În cazul problemei de electrocinetică, deoarece mediul este considerat omogen (σ constant în domeniul $\mathcal{D}$), ecuaţia (4.6) se scrie mai simplu, ca în relaţia (4.7).

$$\boxed{\begin{aligned} &V_O\left(\frac{h_D + h_B}{h_A} + \frac{h_A + h_C}{h_B} + \frac{h_B + h_D}{h_C} + \frac{h_C + h_A}{h_D}\right) - \\ &-V_A\frac{h_D + h_B}{h_A} - V_B\frac{h_A + h_C}{h_B} - V_C\frac{h_B + h_D}{h_C} - V_D\frac{h_C + h_A}{h_D} = 0 \end{aligned}} \tag{4.7}$$

EXERCIŢIUL 1: Deduceţi relaţia (4.6).

Ecuaţia asociată unui nod pe frontieră dreaptă:

Ecuaţia de ordin doi, discretizată, scrisă pentru punctul pe frontieră O (figura 4.6) este dată de relaţia (4.8) pentru regimul electrostatic. Pe frontiera dreaptă AOC, vom nota $g = -\varepsilon \frac{\partial V}{\partial n}$ (condiţie de frontieră Neumann, care provine din cunoaşterea componentei normale a inducţiei electrice).

În scrierea ecuaţiei asociată nodului O, intervin numai valorile necunoscutei în nodurile A, B, şi C, proprietătile de material şi sursele asociate celulelor înconjurătoare (notate cu indicii 2 şi 3), precum şi dimensiunile segmentelor grilei care concură în punctul O (segmentele $\|OA\|$, $\|OB\|$ şi $\|OC\|$).

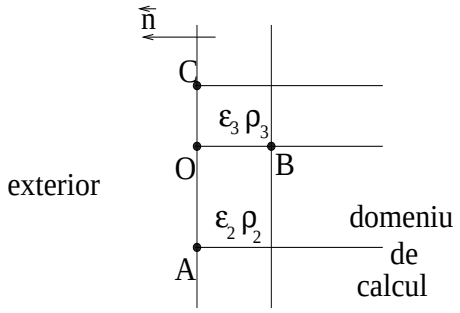


Fig. 4.6. Notaţii pentru un nod pe frontieră dreaptă

$$V_O\left(\frac{\varepsilon_2 h_B}{h_A}+\frac{\varepsilon_2 h_A+\varepsilon_3 h_C}{h_B}+\frac{\varepsilon_3 h_B}{h_C}\right)-V_A\frac{\varepsilon_2 h_B}{h_A}-V_B\frac{\varepsilon_2 h_A+\varepsilon_3 h_C}{h_B}-V_C\frac{\varepsilon_3 h_B}{h_C}=$$
$$=\rho_2\frac{h_B h_A}{2}+\rho_3\frac{h_C h_B}{2}-g_{OA}h_A-g_{OC}h_C \qquad (4.8)$$

În relaţia (4.8), g_{OA} este condiţia de frontieră Neumann asociată segmentului OA, iar g_{OC} este condiţia de frontieră asociată segmentului OC.

În cazul problemei de electrocinetică în care condiţiile de frontieră Neumann sunt nule, pentru un nod O situat pe frontiera Neumann dreaptă ecuaţia se scrie mai simplu, ca în relaţia (4.9).

$$V_O\left(\frac{h_B}{h_A}+\frac{h_A+h_C}{h_B}+\frac{h_B}{h_C}\right)-V_A\frac{h_B}{h_A}-V_B\frac{h_A+h_C}{h_B}-V_C\frac{h_B}{h_C}=0 \qquad (4.9)$$

Ecuaţia asociată unui nod pe frontieră - colţ exterior:

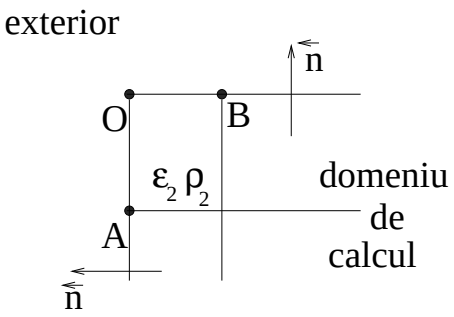


Fig. 4.7. Notaţii pentru un nod pe frontieră colţ exterior

Ecuaţia de ordin doi, discretizată, scrisă pentru punctul pe frontieră O (figura 4.7) este dată de relaţia (4.10) pentru regimul electrostatic. Pe frontiera AOB, vom nota $g = -\varepsilon \frac{\partial V}{\partial n}$ (condiţie de frontieră Neumann, care provine din cunoaşterea componentei normale a inducţiei electrice).

În scrierea ecuaţiei asociată nodului O intervin numai valorile necunoscutei în nodurile A şi B, proprietatea de material şi sursa asociată celulei vecine (notată cu indicele 2), precum şi dimensiunile segmentelor grilei care concură în punctul O (segmentele OA şi OB).

$$V_O\left(\frac{\varepsilon_2 h_B}{h_A}+\frac{\varepsilon_2 h_A}{h_B}\right)-V_A\frac{\varepsilon_2 h_B}{h_A}-V_B\frac{\varepsilon_2 h_A}{h_B}=\rho_2\frac{h_B h_A}{2}-g_{OA}h_A-g_{OB}h_B \quad (4.10)$$

În relaţia (4.10), g_{OA} reprezintă condiţia de frontieră asociată segmentului OA şi g_{OB} cea asociată segmentului OB.

În cazul problemei de electrocinetică, pentru un nod O situat în colţul exterior al frontierei Neumann (punctul S din figura 4.2), ştiind că condiţiile de frontieră Neumann sunt nule, ecuaţia se scrie mai simplu, ca în relaţia (4.11).

$$V_O\left(\frac{h_B}{h_A}+\frac{h_A}{h_B}\right)-V_A\frac{h_B}{h_A}-V_B\frac{h_A}{h_B}=0 \quad (4.11)$$

EXERCIŢIUL 2: Deduceţi relaţia (4.10).

Ecuaţia asociată unui nod pe frontieră - colţ interior:

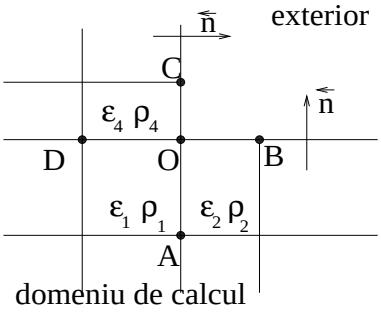


Fig. 4.8. Notaţii pentru un nod pe frontieră colţ interior

Ecuaţia de ordin doi, discretizată, scrisă pentru punctul pe frontieră O (figura 4.8) este dată de relaţia (4.12) pentru regimul electrostatic. Pe frontiera BOC, vom nota $g = -\varepsilon \frac{\partial V}{\partial n}$ (condiţie de frontieră Neumann, care provine din cunoaşterea componentei normale a inducţiei electrice).

În scrierea ecuaţiei asociată nodului O, intervin numai valorile necunoscutei în nodurile A, B, C şi D, proprietăţile de material şi sursele asociate celulelor vecine nodului O (notate cu indicii 1, 2 şi 4), precum şi dimensiunile segmentelor grilei care concură în punctul O (segmentele OA, OB, OC şi OD).

$$\begin{aligned} &V_O\left(\frac{\varepsilon_1 h_D+\varepsilon_2 h_B}{h_A}+\frac{\varepsilon_2 h_A}{h_B}+\frac{\varepsilon_4 h_D}{h_C}+\frac{\varepsilon_4 h_C+\varepsilon_1 h_A}{h_D}\right)- \\ &-V_A\frac{\varepsilon_1 h_D+\varepsilon_2 h_B}{h_A}-V_B\frac{\varepsilon_2 h_A}{h_B}-V_C\frac{\varepsilon_4 h_D}{h_C}-V_D\frac{\varepsilon_4 h_C+\varepsilon_1 h_A}{h_D}= \\ &=\rho_1\frac{h_A h_D}{2}+\rho_2\frac{h_B h_A}{2}+\rho_4\frac{h_D h_C}{2}-g_{OB}h_B-g_{OC}h_C \end{aligned} \tag{4.12}$$

În cazul problemei de electrocinetică cu condiţii de frontieră Neumann nule, pentru un nod O situat în colţul interior al frontierei Neumann (punctul P din figura 4.2) ecuaţia se scrie mai simplu, ca în relaţia (4.13).

$$\begin{aligned} &V_O\left(\frac{h_D+h_B}{h_A}+\frac{h_A}{h_B}+\frac{h_D}{h_C}+\frac{h_C+h_A}{h_D}\right)- \\ &-V_A\frac{h_D+h_B}{h_A}-V_B\frac{h_A}{h_B}-V_C\frac{h_D}{h_C}-V_D\frac{h_C+h_A}{h_D}=0 \end{aligned} \tag{4.13}$$

Pe scurt:

Domeniul de calcul se discretizează cu ajutorul unei grile cu n noduri. Pentru fiecare nod care nu se află pe o frontieră Dirichlet se va scrie o ecuaţie corespunzătoare (de tipul (4.7), (4.9), (4.11) sau (4.13)), iar pentru nodurile Dirichlet se impun valorile respective. Rezultă astfel un sistem de n ecuaţii cu n necunoscute, scris matriceal:

```
M x = p
```

4.4 Implementarea metodei în *Scilab*

Programul implementat va avea (ca orice program ce implementează o metodă numerică de rezolvare a unei probleme de câmp) trei părţi importante:

- *Preprocesarea:* constă în construirea structurilor de date asociate problemei, a matricei coeficienţilor `A` şi a vectorului termenilor liberi `b`.
- *Rezolvarea:* constă în rezolvarea sistemului şi aflarea vectorului necunoscut `x`.
- *Postprocesarea:* constă în calculul mărimilor de interes.

În cele ce urmează, vom parcurge etapele scrierii unui astfel de program.

4.4.1 Preprocesarea

Următorul exerciţiu sugerează faptul că un program trebuie să fie cât mai clar scris astfel încât să poată fi uşor înţeles şi eventual depanat. Programul principal trebuie să reflecte paşii principali ai algoritmului. Toate detaliile se vor executa prin funcţii care vor fi apelate din programul principal.

Prima funcţie apelată de programul principal trebuie să permită introducerea în mod interactiv a datelor problemei, astfel încât acestea să poate fi modificate uşor de un utilizator al programului, fără să fie nevoie de modificarea codului sursă.

EXERCIŢIUL 3:

În directorul ∼/modelare/tema4 scrieţi un fişier "main.sci" cu următorul conţinut:

```
clear;   // este bine ca aceasta comanda sa fie
         // prima linie din orice fisier
         // scris in Scilab

getf('citire_date.sci');
getf('grid.sci');
getf('mvf.sci');

disp('Metoda volumelor finite
      conductor in forma de L');

// --------------- preprocesare ----------------

// citirea si validarea datelor problemei
[a,b,A,B,sigma,V0] = date();

// alegerea grilei
[x,y,NA,Na,Nb,NBb] = grid(a,b,A,B);

// asamblarea matricei coeficientilor M si
// asamblarea vectorului termenilor liberi p
[M, p] = asambleaza(x,y,sigma,V0,NA,Na,Nb,NBb);
```

A doua funcţie apelată din `main` construieşte doi vectori `x` şi `y` având semnificaţia din figura 4.9. Vectorul `x` reprezintă vectorul absciselor. Valoarea `x(1)` poate fi aleasă zero, şi atunci ultima componentă a vectorului trebuie să aibă valoarea A. De asemenea, printre componente trebuie să fie şi valoarea a. Un raţionament similar se face şi pentru vectorul `y`. Funcţia `grid` mai întoarce şi alte numere utile: `NA` – numărul de noduri care se află pe segmentul RS, `Na` – numărul de noduri care se află pe segmenul MN, `Nb` – numărul de noduri care se află pe segmenul QR şi `NBb` – numărul de noduri care se află pe segmenul NP, minus 1.

EXERCIŢIUL 4:

a) Scrieţi, într-un fişier `citire_date.sci` o funcţie cu numele `date`, care să nu aibă nici un parametru de intrare, şi care să întoarcă datele problemei (`a,b,A,B,sigma,V0`). Această funcţie va cere de la consola ***Scilab*** introducerea datelor şi va face verificarea lor (nu va permite de exemplu introducerea unor valori `a, b` negative sau zero, şi nu va permite introducerea unor valori `A, B` mai mici sau egale cu `a`, respectiv `b`).
b) Testaţi corectitudinea funcţiei astfel: în programul principal comentaţi liniile de sub apelul funcţiei `date`, şi executaţi acest program cu comanda `exec`.

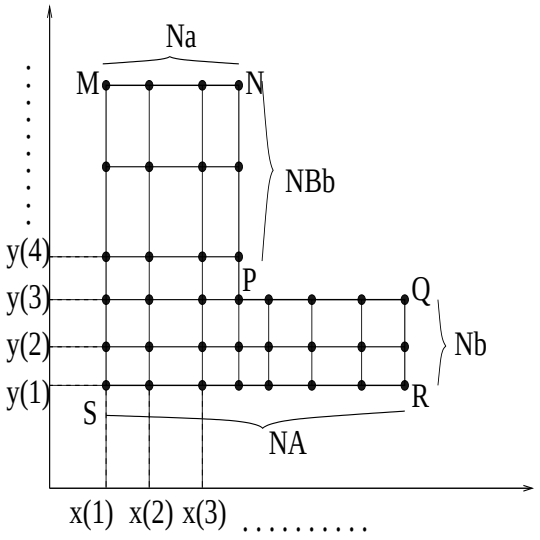


Fig. 4.9. Evidenţierea vectorilor x şi y

EXERCIŢIUL 5:

a) Scrieţi, într-un fişier `grid.sci` o funcţie cu numele `grid`, care să aibă drept parametri de intrare datele geometrice şi care să întoarcă doi vectori `x` şi `y` care reprezintă vectorii absciselor şi ordonatelor care generează grila de discretizare (figura 4.9). Funcţia va cere de la consola **Scilab** introducerea numărului de segmente în care este împărţit segmentul MN, numărul de segmente în care este împărţit segmentul QR, apoi NP şi PQ. Într-o primă etapă, realizaţi o discretizare uniformă a tuturor acestor segmente, folosind instrucţiunea `linspace`.
b) Testaţi corectitudinea funcţiei astfel: în programul principal ştergeţi comentariul liniei corespunzătoare apelului funcţiei `grid` şi executaţi programul cu comanda `exec`.

Urmează "miezul" algoritmului, asamblarea sistemului de ecuaţii în conformitate cu metoda volumelor finite. Vom nota cu `M` matricea coeficienţilor şi cu `p` vectorul termenilor liberi. Nodurile grilei de discretizare le vom numerota de la stânga la dreapta şi de jos în sus, aşa cum se arată în figura 4.10.

Următorul pseudocod detaliază aproape complet modul în care trebuie definită funcţia ce realizează asamblarea matricei coeficienţilor şi a vectorului termenilor liberi.

EXERCIŢIUL 6: Verificaţi şi completaţi pseudocodul funcţiei care realizează asamblarea matricei `M` şi a vectorului `p`.

EXERCIŢIUL 7: Scrieţi un fişier `mvf.sci` care să conţină definiţia funcţiei `asambleaza`, precum şi, eventual, alte funcţii pe care aceasta le apelează.

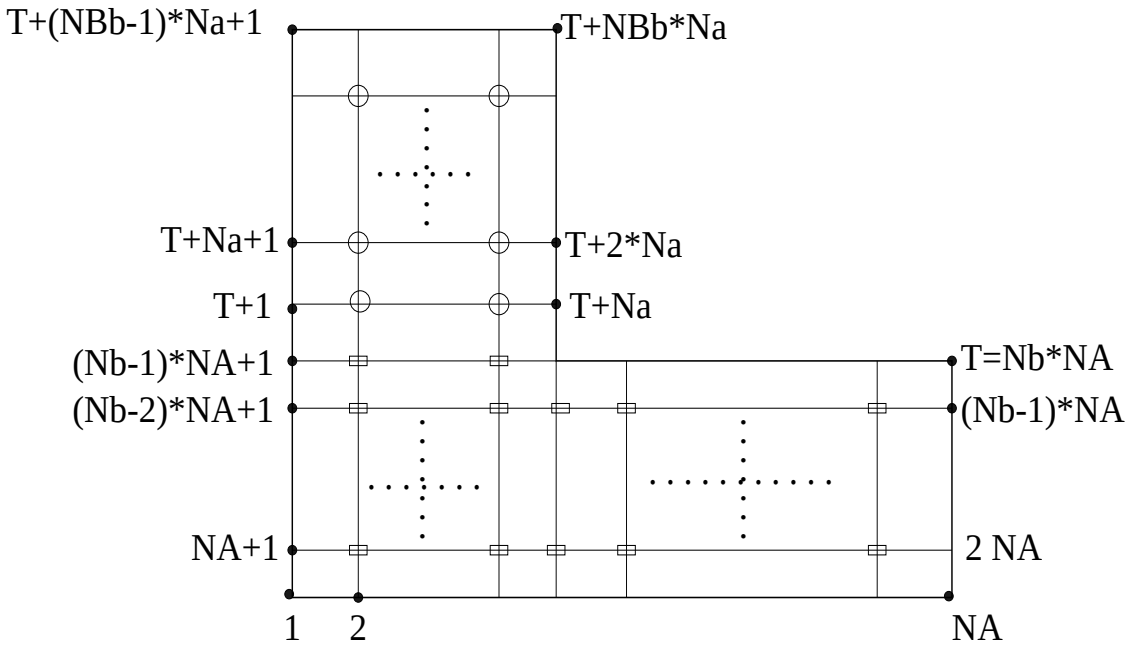


Fig. 4.10. Modul în care se face numerotarea nodurilor

```
funcţie [matrice M, vector p] =                    ; parametri de ieşire
        asambleaza(vector x, vector y, real sigma, V0, NA,Na,Nb,NBb)
; tratarea nodurilor interioare marcate cu dreptunghi în figura 4.10
pentru i = 1, Nb–2
    pentru j = i*NA+2,  (i+1)*NA–1          ;j este numărul nodului
        indicex = j – i*NA
        indicey = i + 1
        OA = y(indicey) – y(indicey–1)
        OB = x(indicex+1) – x(indicex)
        OC = y(indicey+1) – y(indicey)
```

```
                OD = x(indicex) – x(indicex–1)
                ;calculează linia j a matricei M conform relației (5.9)
                .....
i = Nb – 1                                          ;rândul scurt
pentru j = i*NA+2,  i*NA + Na – 1                   ;j este numărul nodului
        indicex = j – i*NA
        indicey = i + 1
        OA = y(indicey) – y(indicey–1)
        OB = x(indicex+1) – x(indicex)
        OC = y(indicey+1) – y(indicey)
        OD = x(indicex) – x(indicex–1)
        ;calculează linia j a matricei M conform relației (5.9)
        .....
; tratarea nodurilor interioare marcate cu cerculeț în figura 4.10
T = Nb*NA
pentru i = 0, NBb–2
        pentru j = T+ i*Na+2,  T+(i+1)*Na–1         ;j este numărul nodului
                indicex = j – T – i*Na
                indicey = Nb + i + 1
                OA = y(indicey) – y(indicey–1)
                OB = x(indicex+1) – x(indicex)
                OC = y(indicey+1) – y(indicey)
                OD = x(indicex) – x(indicex–1)
                ;calculează linia j a matricei M conform relației (5.9)
                .....
;tratarea condițiilor de frontieră
;segmentul MN (Dirichlet, V = V0)
pentru i = 1, Na
        j = T+ (NBb – 1)*Na + i                     ;j este indicele global al nodului
        ;anulează termenii nediagonali ai liniei j ai matricei M,
        ;în afară de termenul diagonal căruia i se atribuie valoarea 1
        .....
        ;atribuie valoarea V0 termenului j al vectorului p
        .....
;segmentul QR (Dirichlet, V = 0)
pentru i = 1, Nb
        j = i*NA                                    ;j este indicele global al nodului
        ;anulează termenii nediagonali ai liniei j ai matricei M,
        ;în afară de termenul diagonal căruia i se atribuie valoarea 1
        .....
        ;atribuie valoarea zero termenului j al vectorului p
        .....
```

```
;segmentul NP (Neumann, zero)
pentru i = 1, NBb – 1
        j = T + i*Na                                ;j este indicele global al nodului
        indicex = Na
        indicey = Nb + i
        OA = y(indicey) – y(indicey–1)
        OB = x(indicex) – x(indicex–1)
        OC = y(indicey+1) – y(indicey)
        ;calculează linia j a matricei M conform relației (4.9)
        .....
;segmentul PQ (Neumann, zero)
pentru i = 1, NA – Na – 1
        j = (Nb – 1)*NA + Na + i                    ;j este indicele global al nodului
        indicex = Na + i
        indicey = Nb
        OA = x(indicex) – x(indicex–1)
        OB = y(indicey) – y(indicey–1)
        OC = x(indicex+1) – x(indicex)
        ;calculează linia j a matricei M conform relației (4.9)
        .....
;segmentul RS (Neumann, zero)
pentru i = 2, NA – 1
        j = i                                       ;j este indicele global al nodului
        indicex = i
        indicey = 1
        OA = x(indicex) – x(indicex–1)
        OB = y(indicey+1) – y(indicey)
        OC = x(indicex+1) – x(indicex)
        ;calculează linia j a matricei M conform relației (4.9)
        .....
;segmentul SM (Neumann, zero)
pentru i = 1, Nb – 1
        j = i*NA + 1                                ;j este indicele global al nodului
        indicex = 1
        indicey = i + 1
        OA = y(indicey) – y(indicey–1)
        OB = x(indicex+1) – x(indicex)
        OC = y(indicey+1) – y(indicey)
        ;calculează linia j a matricei M conform relației (4.9)
        .....
pentru i = 0, NBb – 2
        j = T + i*Na + 1                            ;j este indicele global al nodului
        indicex = 1
        indicey = Nb + i + 1
        OA = y(indicey) – y(indicey–1)
```

```
        OB = x(indicex+1) – x(indicex)
        OC = y(indicey+1) – y(indicey)
        ;calculează linia j a matricei M conform relației (4.9)
        .....
;punctul P (Neumann, zero)
j = (Nb – 1)*NA + Na                          ;j este indicele global al nodului
indicex = Na
indicey = Nb
OA = y(indicey) – y(indicey–1)
OB = x(indicex+1) – x(indicex)
OC = y(indicey+1) – y(indicey)
OD = x(indicex) – x(indicex–1)
;calculează linia j a matricei M conform relației (4.13)
.....
;punctul S (Neumann, zero)
j = 1          ;j este indicele global al nodului
indicex = 1
indicey = 1
OA = x(indicex+1) – x(indicex)
OB = y(indicey+1) – y(indicey)
;calculează linia j a matricei M conform relației (4.11)
.....
```

EXERCIŢIUL 8:

Vizualizaţi matricea `M` (evident, la consola **Scilab**). Ce proprietăţi are această matrice?
a) Este simetrică sau nesimetrică?
b) Este singulară sau nesingulară?
c) Este rară sau plină?
d) Este bine sau prost condiţionată? (folosiţi comanda `cond`)
e) Cât de mare este banda matricei? Ce determină forma şi dimensiunile benzii matricei?
f) Cum puteţi folosi comanda `grayplot` pentru a vizualiza această matrice?

EXERCIŢIUL 9:

a) Aflaţi (şi descrieţi cum aţi procedat) care este numărul maxim de noduri pe care îl poate avea grila de discretizare.
b) Sugeraţi îmbunătăţiri ale pseudocodului propus.
c) Sugeraţi alte modalităţi de a preprocesa această problemă.

4.4.2 Rezolvarea

Rezolvarea sistemului de ecuaţii este extrem de simplă.

EXERCIŢIUL 10:

Adăugaţi la fişierul "main.sci" următoarele rânduri:

```
// --------------- rezolvare ----------------

// rezolvarea sistemului de ecuatii
v = inv(M)*p
```

Cu ce alte instrucţiuni ***Scilab*** puteaţi rezolva sistemul?

4.4.3 Postprocesarea

Potenţialul electric este necunoscuta principală a problemei (aflată direct din rezolvarea sistemului de ecuaţii).

Reprezentarea liniilor echipotenţiale este ceva mai delicată pentru domeniul studiat, deoarece comanda `contour` reprezintă linii echipotenţiale numai pe domenii dreptunghiulare.

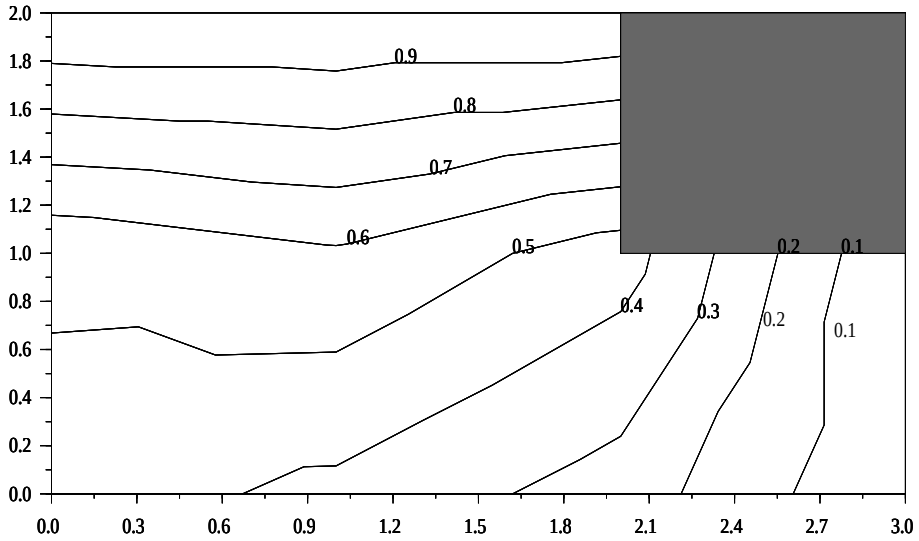


Fig. 4.11. Linii echipotenţiale pentru conductorul în formă de L.

O soluţie este următoarea: se construiesc structuri de date pentru această comandă pentru un domeniu dreptunghiular de dimensiuni `A` şi `B` şi apoi se ignoră colţul fără semnificaţie. Această metodă a fost folosită pentru a trasa liniile echipotenţiale din figura 4.11 (cazul $a = 2, A = 3, b = 1, B = 2$ şi o reţea extrem de rară, cu $x = [0\,1\,2\,3]$ şi $y = [0\,1\,2]$). Pentru acoperirea zonei care nu interesează a fost folosită comanda `xrects`.

	Completaţi fişierul "main.sci" cu:
	`// -------------- postprocesare -------------`
	`exec('postproc.sci');`

EXERCIŢIUL 11:	Scrieţi, într-un fişier numit "postproc.sci" comenzile necesare trasării liniilor echipotenţiale şi "ascunderii" zonei care nu interesează. Studiaţi influenţa valorilor "fantomă" asupra alurii liniilor echipotenţiale din zona de interes.

Calitatea unui algoritm este apreciată prin:

- acurateţea soluţiei obţinute;
- ordinul de complexitate:
  - eficienţa spaţială (memoria necesară datelor);
  - eficienţa temporală.

Eficienţa spaţială şi temporală pot fi estimate teoretic prin numărarea operaţiilor efectuate în algoritm. Necesarul de memorie şi timpul de calcul se exprimă de obicei în funcţie de parametri care determină modificarea numărului de operaţii din algoritm (în cazul problemei studiate, acest parametru este dimensiunea grilei de discretizare).

EXERCIŢIUL 12:	a) Estimaţi teoretic ordinul de complexitate al algoritmului în funcţie de numărul de noduri ale grilei de discretizare. b) Determinaţi experimental ordinul de complexitate. Pentru necesarul de memorie folosiţi comanda `who`, iar pentru determinarea timpului de calcul folosiţi instrucţiunea `timer`.

EXERCIŢIUL 13:	Rezolvaţi problema cu ajutorul metodei aproximării liniilor de câmp şi comparaţi rezultatele cu cele numerice.

Calculul altor mărimi locale (câmp electric, densitate de curent) sau globale (curent, rezistenţă) presupun scrierea de funcţii care au drept parametri de intrare grila de discretizare, proprietăţile de material şi valorile potenţialului.

EXERCIŢIUL 14:

a) Propuneţi şi implementaţi o metodă de calcul a câmpului electric. Reprezentaţi apoi spectrul câmpului electric.
b) Propuneţi şi implementaţi o metodă de calcul a densităţii de curent. Reprezentaţi apoi spectrul densităţii de curent.
c) Propuneţi şi implementaţi o metodă de calcul a curentului.
d) Propuneţi şi implementaţi o metodă de calcul a rezistenţei.
e) Propuneţi o metodă de reprezentare a liniilor câmpului electric.

Capitolul 5

Metoda diferenţelor finite

Ideea metodei diferenţelor finite şi algoritmul ei au fost descrise în tema 3 (paragrafele 3.3.1 şi 3.3.2). Detalierea ei a fost făcută pentru o problemă în care funcţia necunoscută era definită pe un domeniu unidimensional (vezi paragraful 3.3.3).

Această temă urmăreşte detalierea metodei diferenţelor finite în cazul în care domeniul de definiţie al funcţiei este bidimensional. Pentru exemplificare vom considera problema conductorului de forma literei L, problemă enunţată şi formulată în tema 4 (paragrafele 4.1 şi 4.2).

TEMA de laborator:	a) Să se scrie un program ***Scilab*** care să rezolve, folosind metoda diferenţelor finite, problema enunţată şi formulată la tema 4. b) Să se compare rezultatele metodei diferenţelor finite cu rezultatele metodei volumelor finite.

5.1 Metoda diferenţelor finite în cazul bidimensional

Singura deosebire faţă de cazul unidimensional este aceea că grila de discretizare este una bidimensională, iar în loc de aproximarea derivatelor în raport cu o singură variabilă, acum sunt aproximate derivate parţiale.

Vom considera din nou cazul mai general al regimului electrostatic, descris de ecuaţia de ordinul doi neomogenă:

$$-\mathrm{div}\,(\varepsilon\,\mathrm{grad}\,V) = \rho, \tag{5.1}$$

unde $V : \mathcal{D} \to R$, ε este permitivitatea mediului, ρ este densitatea de volum a sarcinii electrice. Condiţiile de frontieră pot fi Dirichlet $V(M) = f$, unde f este cunoscută pentru o parte a frontierei domeniului, sau Neumann $-\varepsilon\,\partial V/\partial n = g$, unde g este cunoscută pentru restul frontierei. Vom presupune că în domeniul de calcul nu există distribuţii superficiale de sarcină.

Pentru a face această prezentare mai uşor de înţeles vom presupune mediul de calcul omogen din punct de vedere electric (ε constant în domeniul $\mathcal{D}$), şi atunci ecuaţia de gradul doi devine:

$$-\text{div}\,(\text{grad}\,V) = \frac{\rho}{\varepsilon}. \tag{5.2}$$

În cazul bidimensional $V = V(x, y)$, şi relaţia (5.2) devine:

$$-\left(\frac{\partial^2 V}{\partial x^2} + \frac{\partial^2 V}{\partial y^2}\right) = \frac{\rho}{\varepsilon}, \tag{5.3}$$

unde $\rho = \rho(x, y)$ iar ε este constantă în domeniu.

Deducerea ecuaţiilor pentru o problemă de regim electrocinetic se face înlocuind ε cu σ şi ρ cu 0.

Vom adopta şi de data aceasta o grilă cartezianăă, pentru care raţionamentul este cel mai simplu. O astfel de grilă este adaptată şi geometriei problemei de rezolvat.

5.1.1 Aproximarea potenţialului şi a derivatelor sale

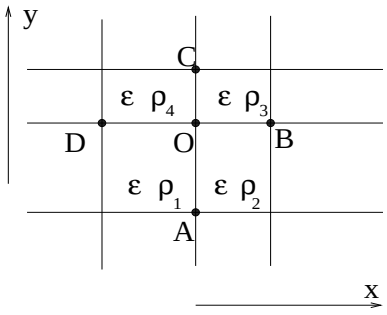


Fig. 5.1. Notaţii pentru un nod interior

Potenţialul într-un punct este aproximat prin dezvoltarea sa în serie Taylor în jurul unor noduri ale grilei, din care sunt reţinuţi inclusiv termenii de ordin doi. Cu notaţiile din figura 5.1 şi $\|OA\| = h_A$, $\|OB\| = h_B$, $\|OC\| = h_C$, $\|OD\| = h_D$, rezultă (pentru nodurile D, O, B situate în această ordine pe axa x):

$$\begin{aligned} V_B &= V_O + h_B \frac{\partial V}{\partial x}(O) + \frac{h_B^2}{2}\frac{\partial^2 V}{\partial x^2}(O) + O(h_B^3) \\ V_D &= V_O - h_D \frac{\partial V}{\partial x}(O) + \frac{h_D^2}{2}\frac{\partial^2 V}{\partial x^2}(O) + O(h_D^3) \end{aligned}$$

Eliminând derivatele de ordinul doi din relaţiile de mai sus este obţinută aproximarea derivatei de ordinul unu a potenţialului:

$$\frac{\partial V}{\partial x}(O) \approx \frac{h_D^2 V_B + (h_B^2 - h_D^2)V_O - h_B^2 V_D}{h_B h_D (h_B + h_D)} \tag{5.4}$$

Prin eliminarea derivatelor de ordinul întâi este obţinută aproximaţia pentru derivata de ordinul doi:

$$\frac{\partial^2 V}{\partial x^2}(O) \approx \frac{h_D V_B - (h_B + h_D)V_O + h_B V_D}{\frac{1}{2} h_B h_D (h_B + h_D)} \tag{5.5}$$

Relaţii similare pot fi deduse şi pentru derivatele parţiale după y:

$$\frac{\partial V}{\partial y}(O) \approx \frac{h_A^2 V_C + (h_C^2 - h_A^2)V_O - h_C^2 V_A}{h_C h_A (h_C + h_A)} \tag{5.6}$$

$$\frac{\partial^2 V}{\partial y^2}(O) \approx \frac{h_A V_C - (h_C + h_A)V_O + h_C V_A}{\frac{1}{2} h_C h_A (h_C + h_A)} \tag{5.7}$$

5.1.2 Aproximarea ecuaţiilor

Forma discretizată (aproximativă) a ecuaţiei potenţialului este obţinută aproximând prin diferenţe finite ecuaţia (5.3). Forma specifică depinde de tipul nodului: interior sau pe frontiera Neumann. Nodurile de pe frontiera Dirichlet nu ridică probleme, ecuaţia asociată unui astfel de nod constând în simpla atribuire a valorii potenţialului nodului respectiv.

Ecuaţia asociată unui nod interior:

Ecuaţia aproximativă pentru un nod interior se deduce înlocuind derivatele parţiale după x şi după y cu formulele (5.5) şi (5.7). Densitatea de sarcină în punctul O este aproximată cu media ei ponderată în acest punct:

$$\rho_O = \frac{\rho_1 h_A h_D + \rho_2 h_B h_A + \rho_3 h_C h_B + \rho_4 h_D h_C}{h_A h_D + h_B h_A + h_C h_B + h_D h_C} \tag{5.8}$$

Ecuaţia aproximativă a problemei electrostatice pentru un nod interior este atunci:

$$\begin{aligned} &V_O \left(\frac{1}{h_B h_D} + \frac{1}{h_A h_C} \right) - V_A \frac{1}{h_A (h_A + h_C)} - V_B \frac{1}{h_B (h_B + h_D)} - \\ &- V_C \frac{1}{h_C (h_A + h_C)} - V_D \frac{1}{h_D (h_B + h_D)} = \frac{\rho_O}{2\varepsilon} \end{aligned} \tag{5.9}$$

În consecinţă, pentru problema de electrocinetică considerată, ecuaţia pentru un nod interior este:

$$\boxed{\begin{aligned} &V_O \left(\frac{1}{h_B h_D} + \frac{1}{h_A h_C} \right) - V_A \frac{1}{h_A (h_A + h_C)} - V_B \frac{1}{h_B (h_B + h_D)} - \\ &- V_C \frac{1}{h_C (h_A + h_C)} - V_D \frac{1}{h_D (h_B + h_D)} = 0. \end{aligned}} \tag{5.10}$$

EXERCIŢIUL 1: Deduceţi relaţia (5.9). Comparaţi această relaţie cu ecuaţia aproximativă pentru un nod interior obţinută la metoda volumelor finite, particularizată pentru un mediu omogen.

Ecuaţia asociată unui nod pe frontieră dreaptă:

În cazul unui punct pe frontieră O (figura 5.2), să notăm cu $g = -\varepsilon \frac{\partial V}{\partial n}$ condiţia de frontieră Neumann şi cu g_O valoarea ei medie în punctul O:

$$g_O = \frac{g_{OA} h_A + g_{OC} h_C}{h_A + h_C} \tag{5.11}$$

unde g_{OA} este condiţia de frontieră asociată segmenului OA iar g_{OC} este condiţia de frontieră asociată segmentului OC.

Pentru deducerea ecuaţiei aproximative pentru punctul O, vom considera un nod "fantomă" D_1, situat în exterior, la o distanţă $h_D = \|OD_1\|$ de punctul O. Pentru simplitate putem considera $h_D = h_B$.

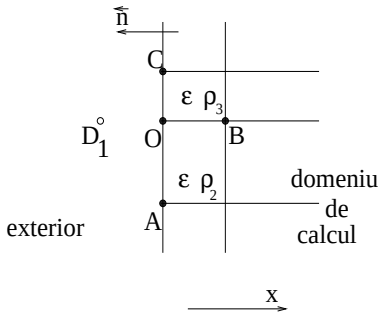


Fig. 5.2. Notaţii pentru un nod pe frontieră dreaptă

Din condiţia de frontieră Neumann, deducem valorea potenţialului fantomă în funcţie de valoarea acestei condiţii. Pentru cazul din figură (normala exterioară în sens contrar axei x) rezultă:

$$\frac{\partial V}{\partial x} = \frac{g}{\varepsilon} \tag{5.12}$$

Scriind ecuaţia aproximativă (5.4) pentru condiţia de frontieră Neumann, rezultă că:

$$V_{D_1} = V_B - 2h_B \frac{g_O}{\varepsilon} \tag{5.13}$$

Pentru nodul O se scrie acum o ecuaţie de tipul (5.9), ca pentru un nod interior şi înlocuind expresia (5.13) rezultă ecuaţia finală (5.14):

$$V_O \left(\frac{1}{h_B^2} + \frac{1}{h_A h_C} \right) - V_A \frac{1}{h_A(h_A + h_C)} - V_B \frac{1}{h_B^2} - V_C \frac{1}{h_C(h_A + h_C)} = \frac{\rho_O}{2\varepsilon} - \frac{g_O}{\varepsilon h_B} \tag{5.14}$$

unde ρ_O este valoarea medie a densităţii de sarcină în punctul O:

$$\rho_O = \frac{\rho_2 h_A h_B + \rho_3 h_B h_C}{h_A h_B + h_B h_C}. \tag{5.15}$$

În cazul regimului electrocinetic, ecuaţia asociată unui nod pe o frontieră dreaptă, corespunzătoare unei condiţii Neumann omogene (nule), este:

$$V_O \left(\frac{1}{h_B^2} + \frac{1}{h_A h_C} \right) - V_A \frac{1}{h_A(h_A + h_C)} - V_B \frac{1}{h_B^2} - V_C \frac{1}{h_C(h_A + h_C)} = 0. \tag{5.16}$$

EXERCIŢIUL 2: Deduceţi relaţia (5.14). Comparaţi această relaţie cu ecuaţia aproximativă pentru un nod pe frontieră dreptă obţinută la metoda volumelor finite.

Ecuaţia asociată unui nod pe frontieră - colţ exterior:

Nodurile situate în colţuri reprezintă o problemă pentru metoda diferenţelor finite. Aceasta deoarece pentru un colţ nu poate fi definită direcţia normalei[1]. Dacă notăm $g = -\varepsilon \frac{\partial V}{\partial n}$, şi notăm cu g_{OA} valoarea lui g pe segmentul OA (fără capătul O) şi cu g_{OB} valoarea lui g pe segmentul OB (fără capătul O), atunci vom presupune că în nodul O (figura 5.3) este cunoscută derivata după o direcţie care rezultă din prelungirea în O a valorilor funcţiei g.

Pentru deducerea ecuaţiei vom considera două noduri "fantomă" C_1 şi D_1, situate în exterior, la distanţele $h_D = \|OD_1\|$ şi $h_C = \|OC_1\|$ de punctul O. Pentru simplitate putem considera $h_D = h_B$ şi $h_C = h_A$.
Din condiţia de frontieră Neumann pe segmentul OB, scrisă în nodul O, putem deduce valoarea potenţialului fantomă C_1:

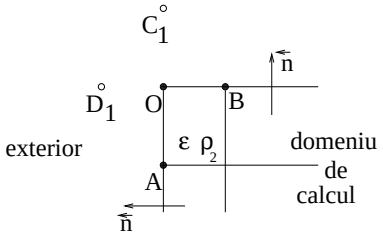


Fig. 5.3. Notaţii pentru un nod pe frontieră colţ exterior

$$V_{C_1} = V_A - 2h_A \frac{g_{OB}}{\varepsilon}. \qquad (5.17)$$

Din condiţia de frontieră Neumann pe segmenul OA, scrisă în nodul O, putem deduce valoarea potenţialului fantomă D_1:

$$V_{D_1} = V_B - 2h_B \frac{g_{OA}}{\varepsilon}. \qquad (5.18)$$

Pentru nodul O se scrie acum o ecuaţie de tipul (5.9), ca pentru un nod interior, şi înlocuind expresiile (5.17) şi (5.18) rezultă ecuaţia finală (5.19):

$$V_O \left(\frac{1}{h_A^2} + \frac{1}{h_B^2} \right) - V_A \frac{1}{h_A^2} - V_B \frac{1}{h_B^2} = \frac{\rho_2}{2\varepsilon} - \frac{g_{OB}}{\varepsilon h_A} - \frac{g_{OA}}{\varepsilon h_B} \qquad (5.19)$$

În cazul regimului electrocinetic, ecuaţia asociată unui nod situat într-un colţ exterior, pentru condiţii Neumann nule, este:

$$\boxed{V_O \left(\frac{1}{h_A^2} + \frac{1}{h_B^2} \right) - V_A \frac{1}{h_A^2} - V_B \frac{1}{h_B^2} = 0} \qquad (5.20)$$

EXERCIŢIUL 3: Deduceţi relaţia (5.19). Comparaţi această relaţie cu ecuaţia aproximativă pentru un nod pe un colţ exterior obţinută la metoda volumelor finite.

[1] În realitate, colţul reprezintă o modelare brutală a realităţii. Trecerea de la o suprafaţă la alta se face prin racordări

Ecuaţia asociată unui nod pe frontieră - colţ interior:

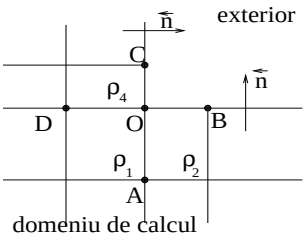


Fig. 5.4. Notaţii pentru un nod pe frontieră colţ interior

Raţionamentul pentru un astfel de colţ este mai dificil de făcut în cazul metodei diferenţelor finite deoarece punctele "fantomă" pe care am încerca să le folosim nu cad în domeniul exterior ci chiar pe frontiera domeniului de calcul şi, în consecinţă, restricţiile asociate lor ar putea intra în contradicţie cu restricţii legate de forma ecuaţiei pentru acele puncte.

De aceea, convenim ca direcţia normalei în punctul O să fie dată de direcţia diagonalei celulei care conţine punctele O, D şi A (figura 5.4). Ne vom imagina şi acum două puncte "fantomă" B_1 şi C_1 plasate astfel încât $\|OB_1\| = \|OD\| = h_D$ şi $\|OC_1\| = \|OA\| = h_A$. În scrierea relaţiei derivatei pentru punctele A, O şi C_1 vom considerea că nu toată condiţia g_{OB} intervine ci numai o parte din ea, şi anume $g_{OB}\sin\alpha$. Similar, în scrierea derivatei pentru punctele D, O şi B_1 vom considera că nu intervine toată condiţia g_{OC} ci numai o parte din ea şi anume $g_{OC}\cos\alpha$. Raţionând similar ca în cazul colţului exterior, rezultă că:

$$V_{C_1} = V_A - 2h_A\frac{g_{OB}\sin\alpha}{\varepsilon} = V_A - 2h_A\frac{g_{OB}}{\varepsilon}\frac{h_A}{\sqrt{h_A^2+h_D^2}} \tag{5.21}$$

$$V_{B_1} = V_D - 2h_D\frac{g_{OC}\cos\alpha}{\varepsilon} = V_D - 2h_D\frac{g_{OC}}{\varepsilon}\frac{h_D}{\sqrt{h_A^2+h_D^2}} \tag{5.22}$$

Pentru punctele O, A, B_1, C_1, D se scrie o relaţie de tipul celei pentru un nod interior şi se substituie apoi expresiile de mai sus pentru nodurile "fantomă", rezultând ecuaţia finală:

$$V_O\left(\frac{1}{h_A^2}+\frac{1}{h_D^2}\right) - V_A\frac{1}{h_A^2} - V_D\frac{1}{h_D^2} = \frac{\rho_O}{2\varepsilon} - \frac{g_{OB}+g_{OC}}{\varepsilon\sqrt{h_A^2+h_D^2}} \tag{5.23}$$

unde pentru ρ_O se poate lua valoarea:

$$\rho_O = \frac{\rho_1 h_A h_D + \rho_2 h_B h_A + \rho_4 h_D h_C}{h_A h_D + h_B h_A + h_D h_C} \tag{5.24}$$

În cazul regimului electrocinetic, ecuaţia asociată unui nod situat într-un colţ interior, pentru condiţii Neumann nule, este:

$$\boxed{V_O\left(\frac{1}{h_A^2}+\frac{1}{h_D^2}\right) - V_A\frac{1}{h_A^2} - V_D\frac{1}{h_D^2} = 0} \tag{5.25}$$

EXERCIŢIUL 4: Deduceţi relaţia (5.23). Comparaţi această relaţie cu ecuaţia aproximativă pentru un nod pe un colţ interior obţinută la metoda volumelor finite.

5.2 Implementarea metodei în *Scilab*

Din punct de vedere al implementării metodei în ***Scilab***, există foarte multe asemănări cu metoda volumelor finite. Singurul lucru care se modifică este modul în care este asamblat sistemul de ecuaţii.

EXERCIŢIUL 5:	Din directorul ∼/modelare/tema4 copiaţi în directorul ∼/modelare/tema5 fişierele: "main.sci", "mvf.sci", "citire_date.sci", "grid.sci", "postprocesare.sci". Modificaţi funcţia `asambleaza` din fişierul "mvf.sci" astfel încât ea să asambleze sistemul de ecuaţii în conformitate cu metoda diferenţelor finite.

EXERCIŢIUL 6:	Comparaţi rezultatele numerice cu cele obţinute la metoda volumelor finite. (Indicaţie: pentru a vizualiza simultan rezultatele numerice, puteţi folosi în acelaşi timp două sesiuni ***Scilab***.)

EXERCIŢIUL 7:	a) Deduceţi ecuaţia metodei diferenţelor finite pentru un punct interior în cazul în care mediul nu este omogen din punct de vedere electric (în regim electrostatic). b) Comparaţi metoda diferenţelor finite cu metoda volumelor finite.

Capitolul 6

Metoda elementelor finite

Ideea metodei elementelor finite şi algoritmul ei au fost descrise în tema 3 (paragraful 3.4). Detalierea ei a fost făcută pentru o problemă în care funcţia necunoscută era definită pe un domeniu unidimensional (vezi paragrafele 3.4.3 şi 3.4.4 de la tema 3).

Această temă urmăreşte detalierea metodei elementelor finite în cazul în care domeniul de definiţie al funcţiei este bidimensional. Pentru exemplificare vom considera problema conductorului de forma literei L, problemă enunţată şi formulată în tema 4 (paragrafele 4.1 şi 4.2).

TEMA de laborator:	a) Să se scrie un program ***Scilab*** care să rezolve, folosind metoda elementelor finite, problema enunţată şi formulată la tema 4. b) Să se compare rezultatele metodei elementelor finite cu rezultatele obţinute prin metoda diferenţelor finite şi respectiv volumelor finite.

6.1 Metoda elementelor finite în cazul bidimensional

Vom considera din nou cazul mai general al regimului electrostatic, descris de ecuaţia de ordinul doi neomogenă:

$$-\text{div}\,(\varepsilon\,\text{grad}\,V) = \rho, \tag{6.1}$$

unde $V : \Omega \to R$, ε este permitivitatea mediului, ρ este densitatea de volum a sarcinii electrice. Domeniul Ω este în general neomogen din punct de vedere electric. Vom presupune că în domeniul de calcul nu există distribuţii superficiale de sarcină.

Condiţiile de frontieră sunt:

Dirichlet: $V = f$, unde f este cunoscută pe o parte $\mathcal{F}_D$ a frontierei $\partial\Omega$ a domeniului;

Neumann: $-\varepsilon\,\partial V/\partial n = g$, unde g este cunoscută pe restul frontierei, $\mathcal{F}_N$.

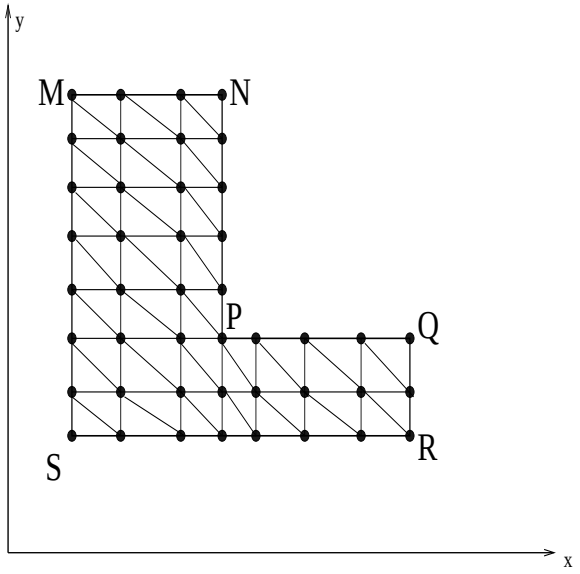


Fig. 6.1. Reţeaua de discretizare a domeniului problemei

Deducerea ecuaţiilor pentru o problemă de regim electrocinetic se face înlocuind ε cu σ şi ρ cu 0.

Discretizarea domeniului problemei

În cazul domeniilor bidimensionale, reţeaua de discretizare este mai complicată decât în cazul unidimensional. Elementele finite (segmente în cazul unidimensional) vor fi de această dată triunghiuri. Acest tip de element finit este deseori folosit, deoarece permite discretizarea mai uşoară a domeniilor de formă complicată. Alternativ, se pot folosi elemente finite patrulatere.

În cazul problemei noastre, vom discretiza domeniul de calcul astfel: fiecare dreptunghi din grila de discretizare folosită la metoda diferenţelor finite va fi împărţit în două triunghiuri, conform figurii 6.1.

În cazul domeniilor neomogene, se presupune că parametrul de material ε (respectiv σ în cazul electrocinetic) este constant în interiorul fiecărui element finit. Sursele (ρ în cazul electrostatic) se presupun de asemenea constante pe elemente.

6.1.1 Aproximarea potenţialului

În fiecare element triunghiular, potenţialul electric va fi aproximat ca o combinaţie liniară de funcţii de bază:

$$\tilde{V} = \sum_{j=1}^{nn} V_j \Psi_j, \tag{6.2}$$

unde atât $\widetilde{V}$ cât şi Ψ_j sunt funcţii de x şi de y, V_j sunt valorile potenţialului în nodurile reţelei de discretizare, iar nn este numărul acestor noduri.

Cazul cel mai simplu este cel în care funcţiile de bază sunt liniare pe porţiuni. Ca şi în cazul unidimensional (paragraful 3.4.1), funcţiile de bază Ψ_j vor fi polinoame Lagrange pe triunghi, cu valoarea 1 în nodul j şi 0 în celelalte noduri.

EXERCIŢIUL 1: Relaţia (6.2) scrisă pentru un punct (x, y) fixat reprezintă o sumă de nn termeni. Care dintre aceştia sunt nenuli?

Pentru a determina funcţiile de bază liniare pe triunghi se procedează după cum urmează. Fie un element finit triunghiular având nodurile j, k, l ordonate în sens trigonometric (figura 6.2). Funcţia de bază asociată nodului j variază liniar:

$$\Psi_j = a_j x + b_j y + c_j. \tag{6.3}$$

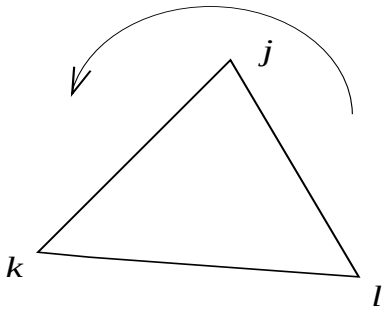


Fig. 6.2. Notaţii pentru nodurile unui element triunghiular

Această funcţie ia valoarea 1 în nodul j şi valoarea 0 în nodurile k şi l:

$$\begin{aligned} \Psi_j(x_j, y_j) &= a_j x_j + b_j y_j + c_j = 1 \\ \Psi_j(x_k, y_k) &= a_j x_k + b_j y_k + c_j = 0 \\ \Psi_j(x_l, y_l) &= a_j x_l + b_j y_l + c_j = 0 \end{aligned} \tag{6.4}$$

Prin rezolvarea sistemului (6.4) rezultă valorile coeficienţilor necunoscuţi:

$$\begin{aligned} a_j &= \frac{y_k - y_l}{\Delta} \\ b_j &= -\frac{x_k - x_l}{\Delta} \\ c_j &= \frac{x_k y_l - x_l y_k}{\Delta}, \end{aligned} \tag{6.5}$$

unde $\Delta = 2A$ este determinantul matricei sistemului, egal cu de două ori aria triunghiului:

$$\Delta = \begin{vmatrix} x_j & y_j & 1 \\ x_k & y_k & 1 \\ x_l & y_l & 1 \end{vmatrix}. \tag{6.6}$$

EXERCIŢIUL 2:

a) Deduceţi formulele (6.5).
b) Prin similitudine cu formulele (6.5), deduceţi expresiile funcţiilor de bază Ψ_k şi Ψ_l asociate nodurilor k şi l ale elementului triunghiular.

Funcţia de bază asociată nodului j este nenulă doar în triunghiurile care conţin nodul j şi are alura din figura 6.3.

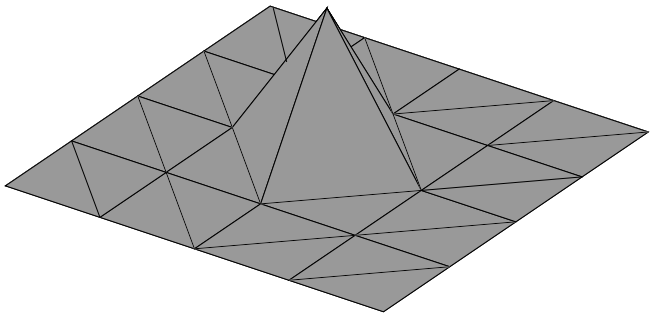


Fig. 6.3. Funcţia de formă asociată unui nod

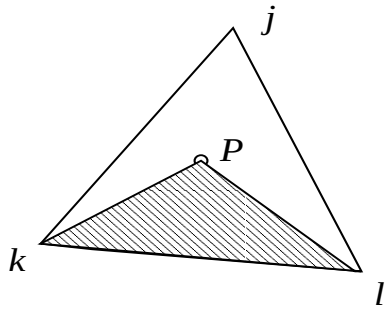


Fig. 6.4. În punctul P, funcţia de bază asociată nodului j este raportul dintre ariile triunghiurilor Pkl şi jkl.

Cele trei funcţii de bază asociate nodurilor triunghiului se mai numesc şi *coordonate areale*, deoarece în orice punct P din triunghi

$$\Psi_j(x,y) = \Psi_j(P) = \frac{A_{Pkl}}{A}, \tag{6.7}$$

cu A_{Pkl} aria triunghiului format din punctele P, k, l (figura 6.4).

Elementele finite pe care potenţialul se aproximează folosind funcţii de bază de tipul (6.3) se numesc elemente finite triunghiulare *liniare* sau *de ordinul 1.*

EXERCIŢIUL 3:

Arătaţi că în orice punct din triunghi avem:

$$\Psi_j(x,y) + \Psi_k(x,y) + \Psi_l(x,y) = 1.$$

Justificaţi de ce se spune că cele trei funcţii de bază formează o *partiţie a unităţii.*
Indicaţie: Pentru deducerea relaţiei puteţi folosi semnificaţia geometrică a funcţiilor Ψ_i.

6.1.2 Asamblarea sistemului de ecuaţii

Forma sistemului de ecuaţii asociat metodei Galerkin

Reziduul ecuaţiei (6.1) are forma $R(V) = -\text{div}\,(\varepsilon\,\text{grad}\,V) - \rho$. Folosirea metodei Galerkin (cu funcţiile pondere aceleaşi cu funcţiile de bază) conduce la:

$$\int_\Omega \left(-\text{div}\,(\varepsilon\,\text{grad}\,V) - \rho\right)\Psi_i\,\mathrm{d}\Omega = 0, \quad i = 1, \ldots, n, \tag{6.8}$$

în care n este numărul de potenţiale necunoscute.

Observaţie. Numărul n al funcţiilor pondere este diferit de numărul total de noduri ale reţelei de discretizare nn: funcţiile pondere sunt asociate numai nodurilor al căror potenţial este **necunoscut**, respectiv, nodurilor interioare şi celor de pe frontiera Neumann. Se constată că funcţiile pondere astfel alese $\Psi_i, i = 1, \ldots, n$ satisfac $\Psi_i = 0$ în nodurile frontierei Dirichlet.

Similar cu cazul unidimensional (tema 3, paragraful 3.4.3), o etapă esenţială o constituie aplicarea *formulei de integrare prin părţi* pentru transformarea primului termen al integralei. În cazul integralelor multiple (în cazul de faţă, bidimensionale), formula de integrare prin părţi este bazată pe formula Gauss–Ostrogradski:

$$\int_\Omega \text{div}\,\mathbf{G}\,\mathrm{d}\Omega = \oint_{\mathcal{F}=\partial\Omega} \mathbf{G}\cdot\mathbf{n}\,\mathrm{d}\mathcal{F}. \tag{6.9}$$

În cazul problemei noastre $\mathbf{G} = \varepsilon\,\text{grad}\,V\,\Psi_i$, iar formula de integrare prin părţi devine

$$\int_\Omega -\text{div}\,(\varepsilon\,\text{grad}\,V)\Psi_i\,\mathrm{d}\Omega = \int_\Omega \varepsilon\,\text{grad}\,V\,\cdot\,\text{grad}\,\Psi_i\,\mathrm{d}\Omega - \oint_{\partial\Omega} \varepsilon\,\Psi_i\,\text{grad}\,V\cdot\mathbf{n}\,\mathrm{d}\mathcal{F}, \tag{6.10}$$

EXERCIŢIUL 4: Deduceţi formula (6.10).
Indicaţie: Notaţi $\mathbf{F} = \varepsilon\,\text{grad}\,V$, $U = \Psi_i$, $\mathbf{G} = U\mathbf{F}$.
Folosiţi, în deducerea relaţiei, formula pentru $\text{div}\,(U\mathbf{F})$.

Cu folosirea relaţiei (6.10), ecuaţia (6.8) devine

$$\int_\Omega \varepsilon\,\text{grad}\,V\cdot\text{grad}\,\Psi_i\,\mathrm{d}\Omega = \int_\Omega \rho\Psi_i\,\mathrm{d}\Omega + \oint_{\partial\Omega} \varepsilon\,\Psi_i\,\text{grad}\,V\cdot\mathbf{n}\,\mathrm{d}\mathcal{F}, \tag{6.11}$$

sau, folosind aproximarea potenţialului V prin relaţia (6.2):

$$\int_\Omega \varepsilon\,\text{grad}\left(\sum_{j=1}^{nn} V_j\Psi_j\right)\cdot\text{grad}\,\Psi_i\,\mathrm{d}\Omega = \int_\Omega \rho\Psi_i\,\mathrm{d}\Omega + \oint_{\partial\Omega} \varepsilon\,\Psi_i\,\text{grad}\,V\cdot\mathbf{n}\,\mathrm{d}\mathcal{F}. \tag{6.12}$$

Valorile V_j nu depind de coordonatele spaţiale x şi y, deci pot fi scoase în afara integralei:

$$\sum_{j=1}^{nn} V_j \int_\Omega \varepsilon\,\text{grad}\,\Psi_i\cdot\text{grad}\,\Psi_j\,\mathrm{d}\Omega = \int_\Omega \rho\Psi_i\,\mathrm{d}\Omega + \oint_{\partial\Omega} \varepsilon\,\Psi_i\,\text{grad}\,V\cdot\mathbf{n}\,\mathrm{d}\mathcal{F}, \quad i = 1, \ldots, n. \tag{6.13}$$

Relaţiile (6.13) scrise pentru $i = 1, \ldots, n$ reprezintă un sistem de ecuaţii algebrice liniare de forma:

$$\sum_{j=1}^{nn} V_j m_{ij} = p_i, \quad i = 1, \ldots, n. \tag{6.14}$$

Deşi suma se extinde pentru $j = 1, \ldots, nn$, cu $nn > n$, doar un număr de n potenţiale V_j sunt necunoscute.

Integralele pe domeniul Ω al problemei se pot scrie ca sume de integrale pe elementele finite, în consecinţă coeficienţii matricei sistemului, m_{ij}, precum şi prima integrală din membrul drept se pot obţine însumând contribuţiile elementelor finite. În plus, la termenii liberi va mai exista o contribuţie datorată integralei pe frontiera $\partial\Omega$.

Determinarea contribuţiei elementelor finite

Comparând relaţiile (6.13) şi (6.14), se constată că un coeficient m_{ij} al matricei sistemului este dat de expresia:

$$m_{ij} = \int_\Omega \varepsilon \operatorname{grad} \Psi_i \cdot \operatorname{grad} \Psi_j \, \mathrm{d}\Omega = \sum_{e=1}^{ne} \int_e \varepsilon \operatorname{grad} \Psi_i \cdot \operatorname{grad} \Psi_j \, \mathrm{d}\Omega = \sum_{e=1}^{ne} m_{ij}^e, \tag{6.15}$$

în care s-a notat cu ne numărul de elemente finite în care a fost divizat domeniul problemei. Similar cu cazul unidimensional, un element finit e va contribui la coeficientul m_{ij} numai dacă nodurile i şi j aparţin elementului finit e.

Contribuţia unui element finit la coeficienţii matricei se calculează astfel:

$$m_{ij}^e = \int_e \varepsilon \operatorname{grad} \Psi_i \cdot \operatorname{grad} \Psi_j \, \mathrm{d}\Omega = \begin{cases} \varepsilon(a_i a_j + b_i b_j)A, & i, j \in e; \quad j \neq i; \\ \varepsilon(a_i^2 + b_i^2)A, & i, j \in e; \quad j = i; \\ 0, & i, j \notin e. \end{cases} \tag{6.16}$$

În relaţiile (6.16) A este aria elementului finit e.

EXERCIŢIUL 5:
a) Deduceţi relaţiile (6.16).
b) La care coeficienţi ai sistemului contribuie un element finit triunghiular $e = \{i, j, k\}$?

Un element finit contribuie şi la termenul liber p_i dacă nodul i aparţine elementului. Comparând relaţiile (6.13) şi (6.14), se constată că partea din coeficientul p_i al termenului liber datorată integralei pe elementele finite este dată de expresia:

$$p_i^e = \int_e \rho \Psi_i \, \mathrm{d}\Omega = \begin{cases} \frac{\rho A}{3}, & i \in e \\ 0, & i \notin e. \end{cases} \tag{6.17}$$

EXERCIŢIUL 6:

Deduceţi relaţia (6.17).
Indicaţie: Puteţi folosi interpretarea geometrică a integralei.

EXERCIŢIUL 7:

O binecunoscută formulă de integrare pe triunghiul $e = \{i, j, k\}$ a coordonatelor areale (funcţiilor Ψ_i) este *formula Holland–Bell*:

$$\int_e \Psi_i^m \Psi_j^n \Psi_k^p \, \mathrm{d}\Omega = 2A \frac{m!n!p!}{(m+n+p+2)!}. \qquad (6.18)$$

a) Verificaţi valabilitatea formulei, în cazul integralei calculate la exerciţiul 6.
b) Calculaţi integralele: $\int_e \Psi_i^2 \, \mathrm{d}\Omega$, $\int_e \Psi_i^2 \Psi_j \, \mathrm{d}\Omega$, $\int_e \Psi_i \Psi_j^2 \, \mathrm{d}\Omega$.

Determinarea contribuţiei frontierei

Cum domeniul Ω este discretizat în elemente finite triunghiulare, frontiera sa va fi aproximată prin segmente de dreaptă. În figura 6.5 este reprezentat un element finit vecin frontierei, segmentul prin care este aproximată frontiera, precum şi sensurile de referinţă pentru normala $\mathbf{n}$ la segmentul de frontieră, versorul tangent $\mathbf{t}$ şi elementul de linie $\mathbf{ds}$.

Aproximarea condiţiilor de frontieră.

Odată cu discretizarea domeniului şi a ecuaţiei satisfăcute de potenţial, este necesară şi discretizarea (aproximarea) condiţiilor de frontieră, după cum urmează.

Deoarece potenţialul V a fost presupus liniar pe fiecare element finit, el va avea variaţie liniară şi pe direcţiile perpendiculare pe segmentele prin care este aproximată frontiera. În consecinţă, $\partial V/\partial n$ va fi constant pe orice segment cu condiţie de frontieră Neumann. Funcţia g reprezentând condiţia de frontieră **Neumann** va fi considerată, prin urmare, ca având câte o valoare constantă pe fiecare din segmentele de dreaptă care aproximează frontiera.

Discretizarea funcţiei f, reprezentând condiţia de frontieră **Dirichlet**, va fi făcută prin considerarea unui set finit de valori ale acesteia, $V_i = f(x_i, y_i)$ în nodurile i aparţinând

frontierei. (Între aceste noduri, variaţia potenţialului este liniară.)

EXERCIŢIUL 8: Pentru ce clasă de funcţii f discretizarea de mai sus nu introduce nici o aproximare?

Tratarea condiţiilor de frontieră Neumann.

Contribuţia frontierei la termenul liber este reprezentată de cea de a doua integrală din membrul drept al relaţiei (6.13):

$$\mathcal{I} = \oint_{\mathcal{F}=\partial\Omega} \varepsilon \, \Psi_i \, \mathrm{grad}\, V \cdot \mathbf{n} \, \mathrm{d}\mathcal{F}. \tag{6.19}$$

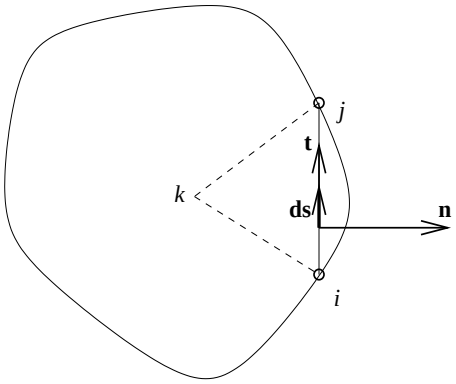


Fig. 6.5. Element triunghiular vecin frontierei

Având în vedere partiţionarea frontierei $\partial\Omega$ în cele două părţi $\mathcal{F}_D$ (pe care este impusă condiţia Dirichlet) şi $\mathcal{F}_N$ (pe care este impusă condiţia Neumann), integrala anterioară poate fi scrisă ca o sumă de două integrale:

$$\mathcal{I} = \mathcal{I}_D + \mathcal{I}_N = \int_{\mathcal{F}_D} \varepsilon \, \Psi_i \, \mathrm{grad}\, V \cdot \mathbf{n} \, \mathrm{d}\mathcal{F} + \int_{\mathcal{F}_N} \varepsilon \, \Psi_i \, \mathrm{grad}\, V \cdot \mathbf{n} \, \mathrm{d}\mathcal{F}. \tag{6.20}$$

Conform observaţiilor referitoare la relaţia (6.8), pe frontiera Dirichlet $\mathcal{F}_D$ funcţiile pondere satisfac $\Psi_i = 0$, în consecinţă prima integrală, $\mathcal{I}_D$, este nulă.

Pe porţiunea $\mathcal{F}_N$ a frontierei se cunoaşte

$$g = -\varepsilon \frac{\partial V}{\partial n}.$$

Integrala $\mathcal{I}_N$ se scrie:

$$\mathcal{I}_N = \int_{\mathcal{F}_N} \Psi_i \, \varepsilon \, \mathrm{grad}\, V \cdot \mathbf{n} \, \mathrm{d}\mathcal{F} = \int_{\mathcal{F}_N} \Psi_i \, \varepsilon \frac{\partial V}{\partial n} \, \mathrm{d}\mathcal{F} = -\int_{\mathcal{F}_N} \Psi_i \, g \, \mathrm{d}\mathcal{F}. \tag{6.21}$$

Integrala $\mathcal{I}_N$ este nenulă numai dacă i este un nod de pe frontieră, deci la ea vor contribui toate elementele care au o latură $[i\ j]$ pe frontieră. Să notăm cu l o astfel de latură.

Integrala $\mathcal{I}_N$ se scrie ca o sumă a contribuţiilor laturilor de pe frontiera Neumann:

$$\mathcal{I}_N = \sum_l \mathcal{I}_N^e = \sum_l \left(-\int_i^j \Psi_i \, g \, \mathrm{d}s \right). \tag{6.22}$$

Latura de frontieră l cu nodurile i şi j va contribui deci la termenul liber p_i cu valoarea:

$$p_i^l = -\int_i^j \Psi_i \, g \, \mathrm{d}s = -g \int_i^j \Psi_i \, \mathrm{d}s = -\frac{gh}{2}, \tag{6.23}$$

unde h este lungimea segmentului $[i\ j]$ şi s-a ţinut cont de faptul că g este constant pe segment.

EXERCIŢIUL 9:

a) Demonstraţi relaţia (6.23).
b) Ce expresie are p_i^l dacă i este nodul **final** al laturii?
c) La care termeni liberi contribuie o latură $[i\ j]$ plasată pe frontiera Neumann?

După cum se constată, în cazul general (când $g \neq 0$), formularea Galerkin include în mod implicit condiţiile de frontieră Neumann. De aceea, condiţiile de frontieră Neumann se numesc condiţii de frontieră *naturale*.

Observaţie: Contribuţiile de tip (6.23) sunt zero în cazul condiţiilor de frontieră Neumann omogene $g = 0$.

Tratarea condiţiilor de frontieră Dirichlet.

Se poate demonstra că sistemul de ecuaţii (6.14), ai cărui coeficienţi sunt complet determinaţi de relaţiile (6.15)— (6.17) şi (6.23), corespunde discretizării problemei (6.1), cu condiţiile de frontieră Neumann $-\varepsilon \, \partial V/\partial n = g$ pe $\mathcal{F}_N$ şi cu condiţii Dirichlet **omogene** $V = 0$ pe $\mathcal{F}_D$.

Impunerea unor condiţii de frontieră Dirichlet nenule, $V = f$ pe $\mathcal{F}_D$ se poate face în mod simplu prin scrierea unui număr suplimentar de ecuaţii, de forma

$$1 \cdot V_i = f_i, \quad i = n+1, \ldots, nn, \tag{6.24}$$

deci cu $m_{ii} = 1$, $m_{ij} = 0$, $\quad j \neq i$, $p_i = f_i$.

Observaţia 1. Această tehnică are două dezavantaje: mărirea dimensiunii sistemului şi distrugerea simetriei matricei M, în schimb, algoritmul de generare a sistemului de ecuaţii rezultă mai simplu. Alternativa care nu are aceste dezavantaje dar complică algoritmul metodei este de a identifica nodurile de potenţial cunoscut în relaţia (6.14), contribuţia lor fiind trecută în membrul drept:

$$\sum_{j=1}^{n} V_j m_{ij} = p_i - \sum_{j=n+1}^{nn} V_j m_{ij}, \quad i = 1, \ldots, n. \tag{6.25}$$

Observaţia 2. Condiţiile de frontieră Dirichlet nu intervin în formularea Galerkin ci trebuie impuse separat, în mod explicit. Din acest motiv, ele se numesc condiţii de frontieră *esenţiale*.

6.2 Implementarea metodei în *Scilab*

Pentru problema considerată, implementarea metodei elementelor finite se poate face similar cu propusă în temele anterioare pentru metoda volumelor finite sau diferenţelor finite (evident, cu alt mod de calcul al coeficienţilor sistemului de ecuaţii). Totuşi, se va prefera implementarea ei într-o formă mai compactă, care se aseamănă cu implementările metodei în programele performante de calcul.

EXERCIŢIUL 10:

Din directorul ∼/modelare/tema4 copiaţi în directorul ∼/modelare/tema6 fişierele: "main.sci", "citire_date.sci", "grid.sci", "postprocesare.sci".
Modificaţi fişierul "main.sci" astfel încât el să aibă conţinutul:

```
clear;

getf('citire_date.sci'); getf('grid.sci');
getf('coord_nod.sci'); getf('gen_elem.sci');
getf('impune_cf.sci'); getf('mef.sci');

disp('Metoda elementelor finite
        conductor in forma de L');

// --------------- preprocesare ----------------
// citirea si validarea datelor problemei
[a,b,A,B,sigma,V0] = date();

// alegerea grilei
[x,y,NA,Na,Nb,NBb] = grid(a,b,A,B);

// generarea retelei de discretizare:
//   coordonate noduri, elemente finite
[nn,xn,yn] = coord_nod(NA,Na,Nb,NBb, x,y);
[ne,elem,ct,sursa] = gen_elem(NA,Na,Nb,NBb,sigma);

// generarea listei de conditii de frontiera
[ncf, cf] = impune_cf(NA,Na,Nb,NBb, V0, 0);

// asamblarea matricei coeficientilor M
// si a vectorului termenilor liberi p
[M,p]=asambleaza(nn,ne,ncf,xn,yn,elem,ct,sursa,cf);

// ----------------- rezolvare -----------------
v = inv(M)*p

// ----------------- postprocesare -------------
exec('postprocesare.sci');
```

Fişierele `citire_date.sci`, `grid.sci` şi `postprocesare.sci` sunt identice cu cele folosite în metoda volumelor finite. Se vor crea fişiere noi care vor conţine funcţii pentru: generarea listei de elemente, generarea listei de coordonate ale nodurilor, impunerea condiţiilor de frontieră.

Funcţia `asambleaza`, utilizată în generarea automată a sistemului de ecuaţii, are forma specifică metodei elementelor finite. Ea va fi definită în fişierul "mef.sci".

6.2.1 Generarea structurilor de date

Prima nouă funcţie apelată trebuie să genereze tablourile coordonatelor nodurilor, `xn` şi `yn`, precum şi numărul total de noduri, `n`.

EXERCIŢIUL 11:	a) Scrieţi, într-un fişier "coord_nod.sci" o funcţie cu numele `coord_nod`, având ca parametri de intrare vectorii `x` şi `y` (coordonatele grilei, vezi fig. 4.9, tema 4) şi care să întoarcă numărul total de noduri `nn` şi doi vectori `xn, yn` care conţin coordonatele nodurilor numerotate conform figurii 4.10, tema 4. b) Verificaţi corectitudinea funcţiei, comentând liniile din programul `main.sci` care nu intervin.

Indicaţie: Iată un pseudocod posibil:

```
funcţie [întreg nn, vector xn, yn] = coord_nod(întreg NA,Na,Nb,NBb, vector x,y)
        ; numar de noduri
        nn = NA*Nb + NBb*Na

        ; noduri in dreptunghiul de jos
        temp = ones(NA, Nb)
        xn = diag(x) * temp, xn = xn(:)
        yn = temp*diag( y(1:Nb) ), yn = yn(:)

        ; noduri in dreptunghiul de sus
        temp = diag( x(1:Na) ) * ones(Na, NBb)
        xn = [xn; temp(:)]'
        temp = ones(Na, NBb) * diag( y(Nb+1:Nb+NBb) )
        yn = [yn; temp(:)]'
retur
```

Pentru memorarea datelor referitoare la elementele finite, structurile de date propuse sunt următoarele (figura 6.6):

- Pentru memorarea nodurilor elementelor finite, se poate folosi o matrice cu ne linii şi 3 coloane; elementele liniei i a matricei au semnificaţia numerelor celor trei noduri ale elementului finit triunghiular i. Vom nota această matrice cu elem.

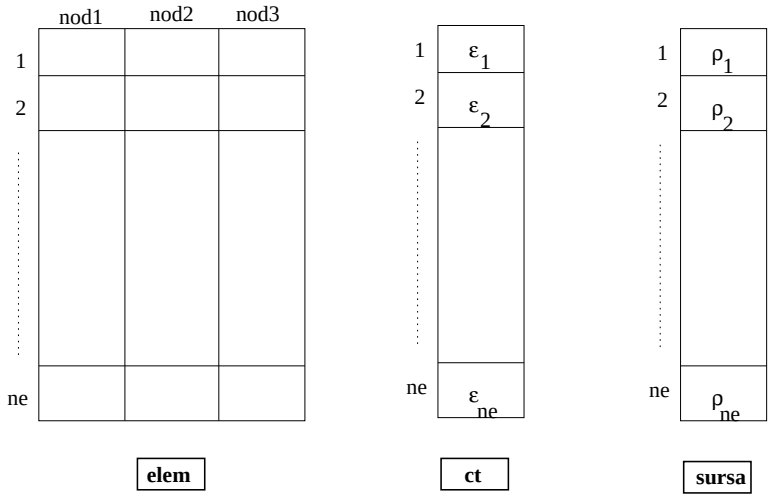


Fig. 6.6. Structuri de date asociate elementelor finite (exemplu pentru o problemă de electrostatică)

- Pentru memorarea constantelor de material ale elementelor finite (presupuse, reamintim, ca având aceeaşi valoare în orice punct din triunghi) se poate folosi un vector cu ne linii, al cărui element i are semnificaţia valorii constantei (de exemplu ε sau σ) a elementului finit i. Această structură de date este potrivită pentru cazul problemelor neomogene. O vom nota cu ct.
- Pentru memorarea surselor (de exemplu ρ) asociate elementelor finite se poate folosi de asemenea un vector cu ne linii. O vom nota sursa.

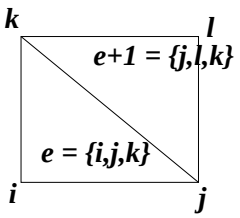


Fig. 6.7. Generarea a două elemente finite, pornind de la un dreptunghi al grilei

Elementele finite triunghiulare vor fi construite, după cum s-a arătat anterior, divizând fiecare dreptunghi al grilei construite la temele 4 şi 5 în câte două triunghiuri (figura 6.7). Se constată că în cazul problemei noastre elementele se pot genera în mod elegant folosind o matrice ajutătoare temp care conţine, pe linii şi coloane, numerele nodurilor, reproducând parţial structura grilei. Matricea ajutătoare se va construi separat pentru fiecare porţiune dreptunghiulară a domeniului de calcul. Figura 6.8 exemplifică structura matricei pentru dreptunghiul inferior al domeniului problemei.

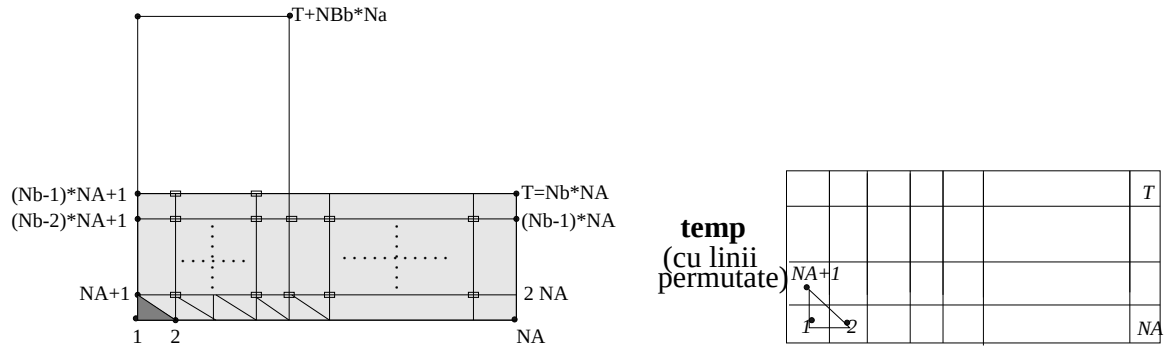


Fig. 6.8. Tehnica de generare a listei de elemente finite, pe baza unei matrice ajutătoare

EXERCIŢIUL 12:

a) Scrieţi, într-un fişier "gen_elem.sci" o funcţie cu numele `gen_elem`, cu parametrii de intrare: `NA, Na, Nb, NBb, sigma` şi care întoarce:
- `ne` – numărul de elemente finite triunghiulare;
- `elem` – lista elementelor finite;
- `ct` – vectorul constantelor de material ale elementelor finite (permitivităţi în cazul electrostatic, respectiv conductivităţi, în cazul problemei de faţă);
- `sursa` – vectorul surselor asociate elementelor finite (ρ în cazul electrostatic).

b) În acelaşi fişier, scrieţi o funcţie `element` care adaugă două noi elemente, conform figurilor 6.7 şi 6.8. Funcţia va avea parametrii de intrare: linia `i` şi coloana `j` din matricea ajutătoare, matricele `elem, temp`, precum şi numărul curent de elemente, `e`. Parametrii de ieşire vor fi `elem, e`.

c) Testaţi corectitudinea funcţiilor `gen_elem` şi `element`.

Indicaţie: Iată un pseudocod posibil (adaptat facilităţilor oferite de *Scilab*).

```
funcţie [întreg ne, matrice elem, vector ct,sursa]
                    = gen_elem (întreg NA,Na,Nb,NBb, real sigma)
    ; Elemente
    ; dreptunghiul de jos
    T = NA * Nb
    temp = 1:T
    temp = matrix(temp, NA, Nb)'

    e = 0
    elem = [ ]
```

```
        pentru i=1:Nb-1,                ; linii, de jos in sus
                pentru j=1:NA-1,        ; coloane, de la stanga la dreapta
                        ; adauga doua elemente
                        [elem,e]=element(i,j,elem,temp,e)

        ;dreptunghiul de sus
        temp = [T+1:T+NBb*Na]
        temp = [(Nb-1)*NA+1:(Nb-1)*NA+Na, temp]
        temp = matrix(temp, Na, NBb+1)'

        pentru i=1:NBb,                 ; linii, de jos in sus
                pentru j=1:Na-1,        ; coloane, de la stanga la dreapta
                        ; adauga doua elemente
                        [elem,temp,e]=element(i,j,elem,temp,e)
        ;Verificare numar elemente
        dacă (e == 2*( (NA-1)*(Nb-1) + (Na-1)*NBb))
                scrie "Numar elemente calculat corect."
        altfel
                scrie "Eroare. Numar incorect de elemente!"

        ; Genereaza vectorul surselor si al constantelor de material
        ct = sigma * ones(1:e)
        sursa = 0 * ones(1:e)
retur
```

Condiţiile de frontieră vor fi memorate într-o matrice `cf` cu 4 coloane şi cu un număr de linii egal cu numărul de condiţii de frontieră (numărul de noduri Dirichlet plus numărul de segmente Neumann) (figura 6.9). Semnificaţiile coloanelor matricei `cf` sunt următoarele:

• prima coloană conţine tipul condiţiei (1 – Dirichlet, 2 – Neumann);

• a doua şi a treia coloană conţin numere de noduri: nodul cu condiţie Dirichlet, respectiv nodul iniţial şi final al laturii cu condiţie Neumann (coloana a treia nu are semnificaţie în cazul condiţiei Dirichlet);

• coloana a patra conţine valoarea condiţiei de frontieră (valoarea funcţiei f sau g în nodul/pe latura de frontieră respectivă).

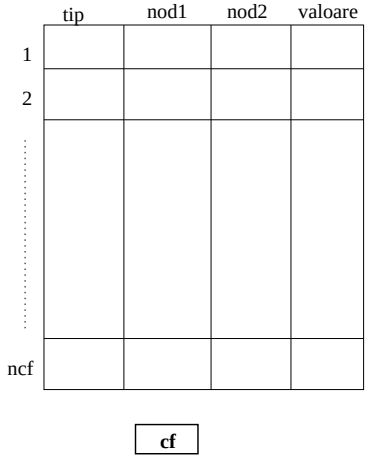


Fig. 6.9. Structuri de date asociate condiţiilor de frontieră

EXERCIŢIUL 13:

a) Scrieţi, într-un fişier "impune_cf.sci" o funcţie cu numele `impune_cf` care primeşte ca parametri `NA, Na, Nb, NBb, V0, gN` şi returnează: numărul `ncf` de elemente cu condiţii de frontieră şi matricea `cf`.
b) Verificaţi corectitudinea funcţiei `impune_cf`.

Indicaţie: Iată un pseudocod posibil.

```
funcţie [întreg ncf, matrice cf] = impune_cf(întreg NA,Na,Nb,NBb,real V0,gN)
        T = NA*Nb
        dir = 1
        neum = 2
        tip = 1, nod1 = 2, nod2 = 3, val = 4
        ncf = 0; numar laturi cu conditie de frontiera
        cf = [ ]

        ; impune Dirichlet
        pentru i=1, Nb
                ncf = ncf+1
                cf(ncf, tip) = dir
                cf(ncf,nod1) = NA*i
                cf(ncf, val) = 0
        pentru i=T+(NBb-1)*Na+1, T+NBb*Na
                ncf = ncf+1
                cf(ncf, tip) = dir
                cf(ncf,nod1) = i
                cf(ncf, val) = V0

        ; impune Neumann
        ; nu e necesar in problema de fata deoarece cond Neumann e zero
        dacă gN <> 0 atunci
        pentru i=1, NA-1                        ; conditie Neumann nula pe SR
                ncf = ncf+1
                cf(ncf, tip) = neum
                cf(ncf,nod1) = i                ; Neumann pe segmentul [i i+1]
                cf(ncf,nod2) = i+1
                cf(ncf,val) = gN
        . . . . . .
retur
```

6.2.2 Asamblarea matricei

Asamblarea matricei va fi făcută prin parcurgerea elementelor finite şi adăugarea contribuţiilor lor la matricea sistemului şi la termenii liberi, conform formulelor (6.16), (6.17).

EXERCIŢIUL 14: În fişierul "mef.sci" scrieţi o funcţie `calcul_abc` având ca parametri de intrare coordonatele celor trei noduri ale unui element şi care calculează vectorii `a, b, c` cu formulele (6.5).

Indicaţie: Iată un pseudocod posibil.

```
funcţie [vector a, b, c] = calcul_abc(vector x, y)
; calculeaza a(i), b(i), c(i) pentru elementul curent

        delta = det([x', y', ones(3,1)])
        pentru i=1:3,
                a(i) = ( y(2)-y(3) )/delta
                b(i) = -( x(2)-x(3) )/delta
                c(i) = ( x(2)*y(3) - x(3)*y(2) )/delta
                ;permuta x, y
                x = [x(2:3), x(1)]
                y = [y(2:3), y(1)]
retur
```

EXERCIŢIUL 15: Adăugaţi, în fişierul "mef.sci", două funcţii care calculează contribuţiile unui element finit la termenul m_{ij} al matricei, respectiv la termenul liber p_i. Se presupune că determinantul $\Delta = 2A$ este calculat în alt modul al programului.

```
funcţie [s] = m_ij_elem(e, i, j)
; calculeaza contributia elementului e la coeficientul m(i,j)
        s = ct(e) * (a(i)*a(j) + b(i)*b(j)) *delta/2
retur
funcţie [s] = p_i_elem(e, i) ; calculeaza contributia elementului e la termenul liber p(i)
        s = sursa(e) * delta/6
retur
```

EXERCIŢIUL 16:

Adăugaţi, în fişierul "mef.sci", funcţia `asambleaza`, care generează automat sistemul de ecuaţii. Funcţia va avea ca parametri de intrare datele referitoare la reţeaua de discretizare şi la problemă (`nn,ne,ncf,xn,yn,elem, ct, sursa, cf`) şi va returna matricea sistemului `M` şi termenul liber `p`.

```
funcţie [matrice M,vector p] = asambleaza(întreg nn,ne,ncf, vector xn,yn,
                matrice elem, vector ct, sursa, matrice cf)
        ; asambleaza matricea sistemului

        M = zeros(nn, nn), p = zeros(nn, 1)
        ; contributiile elementelor
        pentru e=1:ne,
                nod = elem(e,:)
                x = xn(nod)
                y = yn(nod)
                delta = det([x', y', ones(3,1)])        ; delta = 2 * aria triunghiului
                [a, b, c] = calcul_abc(x, y)

                pentru i=1:3,           ; parcurge nodurile elementului
                        n1 = nod(i)
                        pentru j=1:3,
                                n2 = nod(j)
                                M(n1,n2) = M(n1,n2) + m_ij_elem(e,i,j)
                        p(n1) = p(n1) + sursa(e) * delta/6
        ; contributiile conditiilor de frontiera
        dir = 1, neum = 2
        tip = 1, nod1 = 2, nod2 = 3, val = 4        ;indici in matricea cf
        pentru i=1:ncf,
                tipcf = cf(i, tip)
                n1 = cf(i, nod1), n2 = cf(i, nod2)    ; nodurile laturii
                dacă (tipcf == dir) atunci          ; Dirichlet
                        M(n1, :) = zeros(M(n1, :))
                        M(n1, n1) = 1
                        p(n1) = cf(i, val)
                dacă (tipcf == neum)                ; Neumann
                        h = sqrt( (xn(n2)-xn(n1))^2 + (yn(n2)-yn(n1))^2 ) ; lungimea laturii
                        p(n1) = p(n1) - cf(i,val) * h/2
                        p(n2) = p(n2) - cf(i,val) * h/2
retur
```

EXERCIŢIUL 17:	Comparaţi rezultatele numerice cu cele obţinute la metoda volumelor finite. (Indicaţie: pentru a vizualiza simultan rezultatele numerice, puteţi folosi în acelaşi timp două sesiuni ***Scilab***.)

EXERCIŢIUL 18:	Comparaţi metoda elementelor finite cu metodele diferenţelor finite, respectiv volumelor finite.

EXERCIŢIUL 19:	Comparaţi matricea M obţinută în cazul metodei elementelor finite cu cea obţinută la metoda volumelor finite. Indicaţie: Imaginaţi-vă că aţi scala prin înmulţire cu 2 toţi coeficienţii de pe liniile matricei de elemente finite care nu corespund unor noduri cu condiţie de frontieră Dirichlet.

EXERCIŢIUL 20:	Funcţiile apelate de programul principal de elemente finite sunt: `date`, `grid`, `coord_nod`, `gen_elem`, `impune_cf`, `asambleaza`, `postprocesare`. Care dintre aceste funcţii **nu** trebuie rescrise dacă se doreşte rezolvarea altei probleme (se modifică geometria problemei, sursele, parametrii de material şi condiţiile de frontieră)?

6.2.3 Exerciţii propuse

Exerciţiile care urmează sunt facultative[1].

Verificarea preciziei aproximării condiţiei Neumann

În cazul problemei de faţă, teoretic, pe laturile de pe frontiera Neumann $\partial V/\partial n = 0$. Verificaţi cât de bine este aproximată această condiţie, rezolvând exerciţiile 21, 22.

EXERCIŢIUL 21:	a) Cum variază $\operatorname{grad} V$ într-un element finit? Care este expresia lui? b) Calculaţi, pentru o latură de pe frontieră, expresia $\partial V/\partial n = \operatorname{grad} V \cdot \mathbf{n}$. Indicaţie: Scrieţi pe componente versorul $\mathbf{n} = n_x\mathbf{i} + n_y\mathbf{j}$ şi apoi efectuaţi produsul scalar $\operatorname{grad} V \cdot \mathbf{n}$.

[1] Le puteţi rezolva după ce aţi terminat tema următoare.

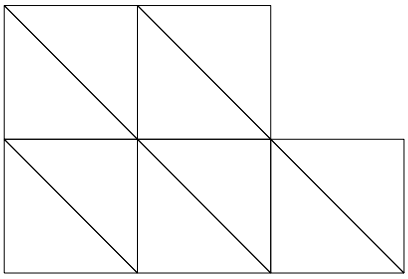


Fig. 6.10. Reţeaua de discretizare, reprezentată folosind *Scilab*, în cazul Na = 3, Nb = 2, NBb = 1, NA = 4

EXERCIŢIUL 22:

a) Scrieţi un algoritm care calculează grad $V \cdot \mathbf{n}$ pentru o latură de pe frontiera Neumann.
b) Aplicaţi algoritmul scris la punctul a) pentru a obţine un vector de valori ale derivatei după normală a potenţialului pe toate laturile de pe frontiera Neumann. (Indicaţie: puteţi utiliza structura de date cf.)
c) Verificaţi dacă $\partial V/\partial n$ este zero pe fiecare segment. Dar în medie, pe toate segmentele frontierei Neumann?

Noi elemente de postprocesare

În etapa de postprocesare este util să se reprezinte grafic nu numai curbele de potenţial constant ci şi alte elemente, cum ar fi reţeaua de discretizare sau vectorii câmpului electric.

EXERCIŢIUL 23:

a) Folosind comanda `help` aflaţi care este efectul şi care sunt parametrii comenzii `xpolys`.
b) Concepeţi un algoritm care permite reprezentarea grafică a reţelei de elemente finite triunghiulare utilizată. Indicaţie: Puteţi folosi structurile deja existente `elem, xn, yn`.
c) Adăugaţi în fişierul "postprocesare.sci" instrucţiunile algoritmului propus la punctul anterior. Verificaţi corectitudinea algoritmului: pentru parametrii de intrare Na = 3, Nb = 2, NBb = 1, NA = 4, rezultatul ar trebui să fie similar cu cel din figura 6.10.

Vectorii intensităţii câmpului electric pot fi reprezentaţi grafic, dacă domeniul are formă dreptunghiulară, folosind comanda `champ` sau `fchamp`. În cazul unui domeniu de formă oarecare însă, aceste comenzi nu sunt potrivite. Se poate folosi în schimb facilitatea *Scilab* de reprezentare a unor săgeţi.

În cazul reprezentării câmpului electric, lungimile acestor săgeţi ar trebui să fie proporţionale cu modulul intensităţii câmpului electric, iar proiecţiile săgeţilor pe axele Ox şi Oy ar trebui să fie proporţionale cu componentele E_x şi respectiv E_y.

EXERCIŢIUL 24:

a) Folosind comanda `help` aflaţi care este sintaxa comenzii `xarrows`.
b) Comentaţi efectul următoarelor comenzi *Scilab*:

```
xbasc();
rect = [0 0 1 1];
plot2d(1,1,[-1],"213"," ",rect);  //comentati

// trei noduri
x1=0; x2=1; x3=0;
y1=0; y2=0; y3=1;
xx = [x1 x2 x3 x1];
yy = [y1 y2 y3 y1];
xpolys(xx', yy');  //comentati

scala = 0.3; Ex = 1; Ey = 1;
xc = (x1+x2+x3)/3;
yc = (y1+y2+y3)/3;  //comentati
nx = [xc; xc+Ex*scala];
ny = [yc; yc+Ey*scala];
xarrows(nx, ny);  //comentati
```

EXERCIŢIUL 25:

a) Folosind relaţia cunoscută $\mathbf{E} = -\text{grad}\, V$, determinaţi componentele E_x şi E_y ale intensităţii câmpului electric în centrele de greutate ale elementelor finite. Folosiţi pentru memorarea acestor valori doi vectori, `Ex` şi `Ey`.
b) Utilizaţi comenzile `xpolys` şi `arrows` pentru a reprezenta grafic vectorii intensităţii câmpului electric în centrele de greutate ale elementelor finite. (În cazul Na = 3, Nb = 2, NBb = 1, NA = 4 rezultatul ar trebui să arate ca în figura 6.11.)

Variante ale metodei elementelor finite

Exerciţiile următoare vă propun să construiţi variante ale metodei elementelor finite, aşa cum a fost ea prezentată în tema de faţă.

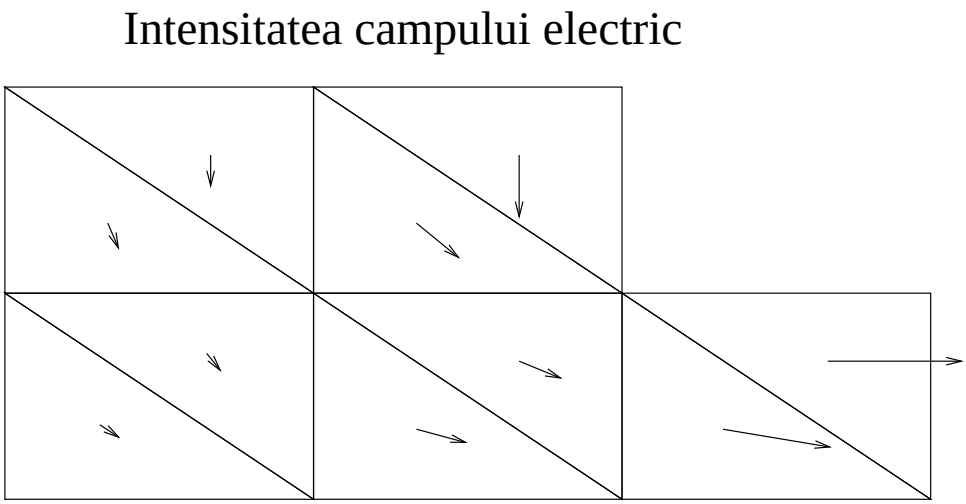


Fig. 6.11. Vectorii intensităţii câmpului electric, reprezentaţi folosind *Scilab*, în cazul Na = 3, Nb = 2, NBb = 1, NA = 4

EXERCIŢIUL 26:	Modificaţi algoritmul astfel încât să folosească elemente finite patrulatere.

EXERCIŢIUL 27:	Modificaţi algoritmul astfel încât să permită rezolvarea problemelor de regim staţionar axisimetrice (2,5 D), folosind o reţea de discretizare bidimensională (2D).

Indicaţie: Configuraţiile axisimetrice se obţin prin rotirea în jurul axei $r = 0$ a unui domeniu bidimensional. Diferenţa faţă de cazul plan–paralel intervine la calculul integralelor pe domeniu, respectiv pe frontieră. Este recomandabil să lucraţi în coordonate cilindrice (după efectuarea calculelor, coordonatele cilindrice r şi z pot fi renotate cu x şi respectiv y, pentru a putea refolosi cât mai mult din algoritmul scris în cazul plan–paralel).

Capitolul 7

Metoda elementelor de frontieră

Această temă[1] urmăreşte detalierea metodei elementelor de frontieră în cazul unei probleme plan-paralele de regim electrocinetic staţionar.

Pentru exemplificare vom considera problema conductorului de forma literei L, problemă enunţată şi formulată în tema 4 (paragrafele 4.1 şi 4.2).

TEMA de laborator:	a) Să se scrie un program ***Scilab*** care să rezolve, folosind metoda elementelor de frontieră, problema enunţată şi formulată la tema 4. b) Să se compare rezultatele metodei elementelor de frontieră cu rezultatele metodelor volumelor finite, diferenţelor finite şi elementelor finite.

7.1 Ideea metodei. Ecuaţii integrale.

Spre deosebire de metodele abordate până acum, în care erau aproximate ecuaţiile câmpului electromagnetic, în metoda elementelor de frontieră se aproximează condiţiile de frontieră, ecuaţia de ordinul doi asociată domeniului de calcul fiind satisfăcută exact.

Ideea metodei elementelor de frontieră constă în exploatarea unei formule integrale pentru necunoscuta principală a problemei (potenţialul).

O astfel de formulă integrală conţine o integrală pe domeniu şi integrale pe frontiera domeniului. Integrala pe domeniu cuprinde sursele interioare de câmp (de exemplu distribuţii de sarcină în cazul regimului electrostatic). Integralele pe frontiera domeniului au integranzi care conţin expresii de tipul condiţiilor de frontieră Dirichlet **şi** Neumann. Dacă în fiecare punct al frontierei s-ar cunoaşte atât condiţiile Dirichlet cât şi condiţiile Neumann atunci cu o astfel de formulă se poate calcula potenţialul în orice punct din domeniul de interes.

[1] Aţi parcurs până acum cele 6 teme anterioare? Felicitări! Drept recompensă, această temă cuprinde codurile complete ***Scilab*** ale programelor de implementat.

Conform teoremei de unicitate, condiţiile de frontieră sunt Dirichlet **sau** Neumann. Pentru deducerea celorlalte valori de pe frontieră, necesare calculului potenţialului în orice punct din domeniu, se foloseşte aceeaşi formulă integrală.

Avantajul acestei metode este acela că ea poate fi aplicată şi pentru domenii nemărginite[2], lucru nepermis în cazul metodelor învăţate până acum. În cazul metodelor volumelor finite, diferenţelor finite, elementelor finite, domeniul de calcul trebuie să fie mărginit. Problema de regim electrocinetic studiată este o problemă în care mărginirea domeniului de calcul a fost ceva natural, impus de condiţiile mărginirii cu un mediu izolant.

EXERCIŢIUL 1:	Cum alegeţi domeniul de calcul pentru metoda volumelor finite (sau diferenţelor finite sau elementelor finite) în cazul în care doriţi să calculaţi câmpul electrostatic al unui condensator plan cu armături dreptunghiulare, având una din dimensiuni mult mai mare decât celelalte două ?

Să ne amintim acum câteva cunoştinţe de matematică, necesare înţelegerii paragrafelor următoare.

• **Distribuţia Dirac** are următoarea *definiţie* în cazul unidimensional:

$$\delta(x-x_0)=\begin{cases}\infty & x=x_0\\ 0 & x\neq x_0\end{cases}\qquad \int_{-\infty}^{\infty}\delta(x-x_0)\,\mathrm{d}x=1 \tag{7.1}$$

În cazul bidimensional:

$$\delta(x-x_0,y-y_0)=\begin{cases}\infty & x=x_0 \text{ şi } y=y_0\\ 0 & x\neq x_0 \text{ sau } y\neq y_0\end{cases}\qquad \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\delta(x-x_0,y-y_0)\,\mathrm{d}x\mathrm{d}y=1 \tag{7.2}$$

În cazul tridimensional:

$$\delta(x-x_0,y-y_0,z-z_0)=\begin{cases}\infty & x=x_0 \text{ şi } y=y_0 \text{ şi } z=z_0\\ 0 & x\neq x_0 \text{ sau } y\neq y_0 \text{ sau } z\neq z_0\end{cases}$$

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\delta(x-x_0,y-y_0,z-z_0)\,\mathrm{d}x\mathrm{d}y\mathrm{d}z=1 \tag{7.3}$$

Dacă notăm cu P punctul "fix" de coordonată x_0 în cazul unidimensional, respectiv punctul de coordonate (x_0,y_0) în 2D şi (x_0,y_0,z_0) în 3D şi cu Q punctul "variabil", de coordonată x în cazul unidimensional, respectiv punctul de coordonate (x,y) în 2D şi (x,y,z) în 3D, atunci, relaţiile (7.1), (7.2), (7.3) pot fi scrise mai compact astfel:

$$\delta(Q-P)=\begin{cases}\infty & Q\equiv P\\ 0 & Q\not\equiv P\end{cases}\qquad \int_{\Omega_\infty}\delta(Q-P)\,\mathrm{d}\omega_Q=1 \tag{7.4}$$

În relaţia (7.4) am notat cu Ω_∞ spaţiul infinit (de dimensiune 1, 2 sau 3) şi cu $\mathrm{d}\omega_Q$ elementul de lungime, arie sau volum:

$$\mathrm{d}\omega_Q=\begin{cases}\mathrm{d}l_Q=\mathrm{d}x & \text{1D}\\ \mathrm{d}A_Q=\mathrm{d}x\mathrm{d}y & \text{2D}\\ \mathrm{d}v_Q=\mathrm{d}x\mathrm{d}y\mathrm{d}z & \text{3D}\end{cases} \tag{7.5}$$

[2] "Open boundary"

O proprietate importantă a distribuţiei Dirac este aceea de *filtrare*. Dacă f este o funcţie definită pe un domeniu infinit (1, 2 sau 3 dimensional), continuă în $x = x_0$ (sau $x = x_0$, $y = y_0$ sau $x = x_0$, $y = y_0$, $z = z_0$), atunci:

$$\text{1D}: \quad \int_{-\infty}^{\infty} f(x)\delta(x - x_0)\,\mathrm{d}x = f(x_0) \tag{7.6}$$

$$\text{2D}: \quad \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\delta(x - x_0, y - y_0)\,\mathrm{d}x\mathrm{d}y = f(x_0, y_0) \tag{7.7}$$

$$\text{3D}: \quad \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y,z)\delta(x - x_0, y - y_0, z - z_0)\,\mathrm{d}x\mathrm{d}y\mathrm{d}z = f(x_0, y_0, z_0) \tag{7.8}$$

Proprietatea de filtrare se poate scrie compact astfel:

$$\int_{\Omega_\infty} f(Q)\delta(Q - P)\,\mathrm{d}\omega_Q = f(P) \tag{7.9}$$

Iată un *exemplu* de distribuţie Dirac în cazul tridimensional. Să ne imaginăm o sarcină punctuală (sferă de rază infinit mică) cu o valoare egală cu 1 Coulomb, situată în spaţiul infinit tridimensional în punctul de coordonate $P(x_0, y_0, z_0)$. Atunci, densitatea de volum a sarcinii electrice este într-un punct oarecare $Q(x, y, z)$ din spaţiu:

$$\rho(Q) = \begin{cases} \infty & Q \equiv P \\ 0 & Q \not\equiv P \end{cases} \tag{7.10}$$

Este evident că:

$$\int_{\Omega_\infty} \rho(Q)\,\mathrm{d}v_Q = 1 \quad [\mathrm{C}] \tag{7.11}$$

Un exemplu de distribuţie Dirac în cazul bidimensional se poate construi astfel. Să ne imaginăm un fir rectiliniu, infinit lung (cilindru infinit, de rază infinit mică), electrizat, astfel încât sarcina totală pe unitatea de lungime este 1 Coulomb. Firul este situat în spaţiul infinit tridimensional pe o dreaptă paralelă cu axa z, ce taie planul XOY în punctul de coordonate $P(x_0, y_0, 0)$. Calculăm din nou densitatea de volum a sarcinii electrice într-un punct oarecare $Q(x, y, z)$:

$$\rho(Q) = \begin{cases} \infty & x = x_0 \text{ şi } y = y_0 \\ 0 & x \neq x_0 \text{ sau } y \neq y_0 \end{cases} \tag{7.12}$$

Observăm că $\rho(Q)$ nu depinde de z. Să integrăm densitatea de sarcină pe un volum tridimensional $\mathcal{V}$ ce cuprinde o lungime de fir $l = 1$ m cuprinsă de exemplu între cotele $z = 0$ şi $z = l$. Rezultă:

$$\int_{\mathcal{V}} \rho(Q)\,\mathrm{d}v_Q = 1 \quad [\mathrm{C}] \tag{7.13}$$

Dacă notăm cu Ω_∞ un domeniu infinit bidimensional (corespunzător planului $z = 0$) atunci $\mathcal{V} = \Omega_\infty \times [0, l]$, iar relaţia (7.13) devine:

$$\int_{\Omega_\infty} \left(\int_0^l \rho(Q)\,\mathrm{d}z \right) \mathrm{d}x\mathrm{d}y = 1 \quad [\mathrm{C}] \tag{7.14}$$

Deoarece ρ nu depinde de z, rezultă că:

$$\int_{\Omega_\infty} \rho(x,y,0)\,\mathrm{d}x\mathrm{d}y = 1 \quad [\mathrm{C/m}] \tag{7.15}$$

Renotând P ca fiind punctul de coordonate (x_0, y_0) din planul XOY, Q ca fiind punctul de coordonate (x, y) din planul XOY şi $\rho(Q)$ ca fiind restricţia densităţii de volum la acelaşi plan, rezultă că:

$$\int_{\Omega_\infty} \rho(Q)\,\mathrm{d}A_Q = 1 \quad [\mathrm{C/m}] \tag{7.16}$$

unde $\rho(Q)$ este dată de relaţiile (7.10) şi (7.16).

• **Funcţia Green** se *defineşte* astfel.

Fie A un operator diferenţial cu condiţii de frontieră nule pe frontiera domeniului $\partial\Omega$. Vom spune că funcţia de două "variabile" $G(P,Q) : \Omega \to \mathbb{R}$ este funcţia Green a operatorului A dacă soluţia ecuaţiei diferenţiale neomogene $AV = f$ cu condiţii de frontieră nule se scrie sub forma integrală:

$$V(P) = \int_\Omega G(P,Q) f(Q)\,\mathrm{d}v_Q \tag{7.17}$$

Funcţia Green are avantajul că, odată cunoscută, permite determinarea rapidă, printr-o integrală, a soluţiei ecuaţiei neomogene pentru diferiţi termeni liberi f. Operatorul integral definit de funcţia Green este, în consecinţă, inversul operatorului diferenţial considerat.

Aplicând operatorul A soluţiei scrisă sub forma (7.17) şi presupunând că aplicarea operatorului şi integrarea sunt inversabile, rezultă:

$$AV(P) = \int_\Omega [AG(P,G)] f(Q)\,\mathrm{d}v_Q = f(P), \tag{7.18}$$

relaţie satisfăcută de soluţia ecuaţiei $Au = f$ pentru orice termen liber f pentru care există soluţie. Acest lucru se întâmplă dacă $AG(P,Q)$ filtrează valoarea în P a funcţiei $f(Q)$ deci dacă:

$$AG(P,Q) = \delta(Q - P) \tag{7.19}$$

În consecinţă, funcţia Green $G(P,Q)$ este soluţia, în condiţii de frontieră nule, a ecuaţiei constituită cu operatorul A, în care sursa internă este funcţia generalizată Dirac, cu suportul în Ω (punctul P este în interiorul domeniului Ω).

7.1.1 Formula celor trei potenţiale

Vom considera din nou problema de regim electrostatic descrisă de ecuaţia de gradul doi:

$$-\mathrm{div}\,(\varepsilon\,\mathrm{grad}\,V) = \rho \tag{7.20}$$

unde $V : \mathcal{D} \to \mathbb{R}$. Un punct oarecare din $\mathcal{D}$ va fi notat cu Q şi el va avea coordonatele (x, y) într-o problemă plan-paralelă şi (x, y, z) într-o problemă tridimensională. Ecuaţia (7.20) este scrisă pentru un punct oarecare Q:

$$-\mathrm{div}\,(\varepsilon(Q)\,\mathrm{grad}\,V(Q)) = \rho(Q) \tag{7.21}$$

Funcţia Green asociată acestei probleme va satisface:

$$-\text{div}\ [\varepsilon(Q)\,\text{grad}\,G(P,Q)] = \delta(Q-P) \tag{7.22}$$

Să considerăm relaţiile:

$$\begin{aligned}\text{div}\ [G(P,Q)\varepsilon(Q)\,\text{grad}\,V(Q)] &= \varepsilon(Q)\text{grad}\,V(Q)\,\text{grad}\,G(P,Q) + \\ &+ G(P,Q)\,\text{div}\ [\varepsilon(Q)\,\text{grad}\,V(Q)] \qquad (7.23)\\ \text{div}\ [V(Q)\varepsilon(Q)\,\text{grad}\,G(P,Q)] &= \varepsilon(Q)\text{grad}\,V(Q)\,\text{grad}\,G(P,Q) + \\ &+ V(Q)\,\text{div}\ [\varepsilon(Q)\,\text{grad}\,G(P,Q)] \qquad (7.24)\end{aligned}$$

Dacă scădem relaţiile (7.23) şi (7.24), apoi integrăm pe $\mathcal{D}$ şi ţinem seama de relaţiile (7.20) şi (7.22), de proprietatea de filtrare a funcţiei Dirac, şi de formula Gauss-Ostrogradski, rezultă:

$$\begin{aligned}V(P) &= \int_{\mathcal{D}} G(P,Q)\rho(Q)\,\mathrm{d}v_Q + \oint_{\partial\mathcal{D}} G(P,Q)\varepsilon(Q)\frac{\partial V}{\partial n_Q}\,\mathrm{d}A_Q - \\ &- \oint_{\partial\mathcal{D}} V(Q)\varepsilon(Q)\,\text{grad}\,G(P,Q)\cdot\mathbf{n}_Q\,\mathrm{d}A_Q \end{aligned} \tag{7.25}$$

În cazul unei probleme plan-paralele, relaţia (7.25) se scrie astfel:

$$\begin{aligned}V(P) &= \int_{\mathcal{S}} G(P,Q)\rho(Q)\,\mathrm{d}A_Q + \oint_{\partial\mathcal{S}} G(P,Q)\varepsilon(Q)\frac{\partial V}{\partial n_Q}\,\mathrm{d}l_Q - \\ &- \oint_{\partial\mathcal{S}} V(Q)\varepsilon(Q)\,\text{grad}\,G(P,Q)\cdot\mathbf{n}_Q\,\mathrm{d}l_Q \end{aligned} \tag{7.26}$$

EXERCIŢIUL 2:

a) Deduceţi formula (7.25).
b) Proprietatea de filtrare presupune integrare pe un domeniu infinit. Cum justificaţi aplicarea ei pe un domeniu mărginit?

Cazul mediilor omogene

Dacă mediul este omogen ($\varepsilon(Q) = \varepsilon = \text{constant}$), funcţia Green este uşor de găsit. Ea satisface în domeniul de calcul relaţia:

$$-\text{div}\ [\text{grad}\,G(P,Q)] = \frac{\delta(Q-P)}{\varepsilon} \tag{7.27}$$

În consecinţă, funcţia Green pentru un mediu omogen poate fi aleasă ca fiind potenţialul electrostatic al unei sarcini punctiforme (în cazul tridimensional) sau al unui fir infinit lung încărcat cu sarcină (în cazul bidimensional):

$$\text{3D}: \qquad G(P,Q) = \frac{1}{4\pi\varepsilon R_{PQ}} \tag{7.28}$$

$$\text{2D}: \qquad G(P,Q) = \frac{1}{2\pi\varepsilon}\ln\frac{1}{R_{PQ}} \tag{7.29}$$

Gradientul (în Q) acestei funcţii este:

$$3\mathrm{D}: \qquad \operatorname{grad} G(P,Q) = \frac{1}{4\pi\varepsilon}\frac{\mathbf{R}_{PQ}}{R_{PQ}^3} \tag{7.30}$$

$$2\mathrm{D}: \qquad \operatorname{grad} G(P,Q) = \frac{1}{2\pi\varepsilon}\frac{\mathbf{R}_{PQ}}{R_{PQ}^2} \tag{7.31}$$

unde am notat cu $\mathbf{R}_{PQ}$ vectorul care uneşte punctul sursă P (unde se află concentrată sarcina) cu punctul de observaţie Q oarecare, având originea în Q şi vârful în P.

Prin urmare, relaţiile (7.25) şi (7.26) devin:

$$\begin{aligned} 3\mathrm{D}: \quad 4\pi V(P) &= \frac{1}{\varepsilon}\int_{\mathcal{D}} \frac{1}{R_{PQ}}\rho(Q)\,\mathrm{d}v_Q + \oint_{\partial\mathcal{D}} \frac{1}{R_{PQ}}\frac{\partial V}{\partial n_Q}\,\mathrm{d}A_Q - \\ &- \oint_{\partial\mathcal{D}} V(Q)\frac{\mathbf{R}_{PQ}\cdot\mathbf{n}_Q}{R_{PQ}^3}\,\mathrm{d}A_Q \end{aligned} \tag{7.32}$$

$$\begin{aligned} 2\mathrm{D}: \quad 2\pi V(P) &= \frac{1}{\varepsilon}\int_{\mathcal{S}} \ln\left(\frac{1}{R_{PQ}}\right)\rho(Q)\,\mathrm{d}A_Q + \oint_{\partial\mathcal{S}} \ln\left(\frac{1}{R_{PQ}}\right)\frac{\partial V}{\partial n_Q}\,\mathrm{d}l_Q - \\ &- \oint_{\partial\mathcal{S}} V(Q)\frac{\mathbf{R}_{PQ}\cdot\mathbf{n}_Q}{R_{PQ}^2}\,\mathrm{d}l_Q \end{aligned} \tag{7.33}$$

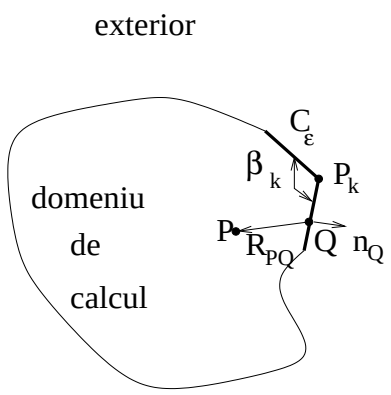


Fig. 7.1. În scrierea ecuaţiei integrale în P_k intervine β_k.

În toate relaţiile de până acum, P este un punct interior domeniului $\mathcal{D}$. Dacă s-ar cunoaşte valorile potenţialului **şi** ale derivatei lui după normală în orice punct de pe frontiera domeniului, atunci cu relaţiile (7.32) sau (7.33) se poate calcula potenţialul în orice punct din interiorul domeniului.

Condiţiile de frontieră impun însă potenţialul **sau** derivata lui după normală în orice punct al frontierei.

Primul pas care trebuie făcut este de a calcula mai întâi potenţialul în punctele frontierei pe care se dau condiţii Neumann şi derivata lui după normală în punctele frontierei pe care se impun condiţii Dirichlet.

Pentru aceasta, relaţiile (7.32) şi (7.33) sunt scrise pentru un punct P de pe frontieră. Când P se apropie de frontieră, integrala a treia din relaţiile (7.32) şi (7.33) este improprie. În *cazul bidimensional* se demonstrează că relaţia (7.33) devine:

$$\begin{aligned} \beta_k V(P_k) &= \frac{1}{\varepsilon}\int_{\mathcal{S}} \ln\left(\frac{1}{R_{P_kQ}}\right)\rho(Q)\,\mathrm{d}A_Q + \oint_{\partial\mathcal{S}} \ln\left(\frac{1}{R_{P_kQ}}\right)\frac{\partial V}{\partial n_Q}\,\mathrm{d}l_Q - \\ &- \oint_{\partial\mathcal{S}} V(Q)\frac{\mathbf{R}_{P_kQ}\cdot\mathbf{n}_Q}{R_{P_kQ}^2}\,\mathrm{d}l_Q \end{aligned} \tag{7.34}$$

unde β_k (vezi figura 7.1) este unghiul interior domeniului de calcul, în punctul P_k.

În *cazul regimului electrocinetic* staţionar (2D), adică în cazul absenţei densităţii de volum a surselor, ecuaţiile integrale devin:

- Pentru un punct interior P:

$$2\pi V(P) = \oint_{\partial\mathcal{S}} \ln\left(\frac{1}{R_{PQ}}\right) \frac{\partial V}{\partial n_Q} \mathrm{d}l_Q - \oint_{\partial\mathcal{S}} V(Q) \frac{\mathbf{R}_{PQ} \cdot \mathbf{n}_Q}{R_{PQ}^2} \mathrm{d}l_Q \quad (7.35)$$

- Pentru un punct P_k situat pe frontieră:

$$\beta_k V(P_k) = \oint_{\partial\mathcal{S}} \ln\left(\frac{1}{R_{P_kQ}}\right) \frac{\partial V}{\partial n_Q} \mathrm{d}l_Q - \oint_{\partial\mathcal{S}} V(Q) \frac{\mathbf{R}_{P_kQ} \cdot \mathbf{n}_Q}{R_{P_kQ}^2} \mathrm{d}l_Q \quad (7.36)$$

unde β_k este unghiul interior al frontierei, în P_k.

În cele ce urmează vom studia cazul regimului electrocinetic staţionar.

7.1.2 Aproximarea potenţialului şi a derivatei sale pe frontiera domeniului

Relaţia (7.36) este cea care se discretizează în vederea obţinerii celorlalte valori necesare pe frontieră: derivatele după normală ale potenţialului în punctele frontierei Dirichlet şi valorile potenţialului în punctele frontierei Neumann. Relaţia (7.35) este cea care se discretizează în vederea postprocesării: calculul potenţialului în orice punct din domeniu.

Pentru aceasta, frontiera domeniului $\partial\mathcal{D}$ se împarte în *elemente de frontieră*: segmente în cazul 2D şi triunghiuri (de exemplu) în cazul 3D.

EXERCIŢIUL 3: Ce diferenţe apar în cazul regimului electrostatic?

Vom aproxima potenţialul V pe fiecare element de frontieră ca având o variaţie liniară, iar derivata lui după normală ca fiind constantă.

7.1.3 Aproximarea condiţiilor de frontieră

Formulele de aproximare

Să notăm cu ne numărul *elementelor de frontieră* şi cu n numărul nodurilor.

EXERCIŢIUL 4: Ce relaţie există între ne şi n în cazul unei frontiere unidimensionale simplu conexe (cazul problemei de rezolvat)?

În cazul bidimensional un element de frontieră (notat generic cu e) reprezintă un segment $[P_i, P_j]$, unde $i, j = 1, \ldots, n$. În consecinţă, aproximaţia relaţiei (7.36) este:

$$\beta_k V(P_k) = \sum_{e=1}^{ne} \int_e \ln\left(\frac{1}{R_{P_k Q}}\right) \frac{\partial V}{\partial n_Q} \, \mathrm{d}l_Q - \sum_{e=1}^{ne} \int_e V(Q) \frac{\mathbf{R}_{P_k Q} \cdot \mathbf{n}_Q}{R^2_{P_k Q}} \, \mathrm{d}l_Q \tag{7.37}$$

Vom nota cu $V_i = V(P_i)$ potenţialul unui nod i şi cu $V'_e = \left(\frac{\partial V}{\partial n}\right)\Big|_e$ valoarea derivatei după normală asociată unui element e.

Potenţialul într-un punct Q de pe segmentul $[P_i, P_j]$, situat la distanţa $l_{P_i Q}$ de P_i (figura 7.2) se poate scrie în funcţie de potenţialele $V_i = V(P_i)$ şi $V_j = V(P_j)$ ca fiind:

$$V(Q) = V_i \left(1 - \frac{l_{P_i Q}}{L_e}\right) + V_j \frac{l_{P_i Q}}{L_e} \tag{7.38}$$

unde am notat cu L_e lungimea segmentului $[P_i, P_j]$.

Ţinând seama de aproximările şi notaţiile de mai sus, relaţia (7.37) se poate scrie astfel:

$$\beta_k V(P_k) = \sum_{e=1}^{ne} g(P_k, P_i, P_j) V'_e - \sum_{e=1}^{ne} h1(P_k, P_i, P_j) V_i - \sum_{e=1}^{ne} h2(P_k, P_i, P_j) V_j \tag{7.39}$$

unde am notat:

$$g(P, P_i, P_j) = \int_{P_i}^{P_j} \ln\left(\frac{1}{R_{PQ}}\right) \mathrm{d}l_Q \tag{7.40}$$

$$h2(P, P_i, P_j) = \int_{P_i}^{P_j} \frac{\mathbf{R}_{PQ} \cdot \mathbf{n}_Q}{R^2_{PQ}} \frac{l_{P_i Q}}{L_e} \, \mathrm{d}l_Q \tag{7.41}$$

$$h3(P, P_i, P_j) = \int_{P_i}^{P_j} \frac{\mathbf{R}_{PQ} \cdot \mathbf{n}_Q}{R^2_{PQ}} \, \mathrm{d}l_Q \tag{7.42}$$

$$h1(P, P_i, P_j) = h3(P, P_i, P_j) - h2(P, P_i, P_j) \tag{7.43}$$

Relaţia (7.39) se scrie pentru:

- toate nodurile interioare frontierei Neumann (caz în care P_k este plasat chiar în nod);
- toate segmentele Dirichlet – cele care au ambele noduri Dirichlet – (caz în care P_k este plasat în mijlocul segmentului respectiv).

La aceste ecuaţii se adaugă:

- ecuaţii asociate nodurilor de pe frontiera Dirichlet, care constau în simpla atribuire a potenţialului nodului respectiv;
- ecuaţii asociate segmentelor Neumann, care constau în simpla atribuire a valorii derivatei după normală.

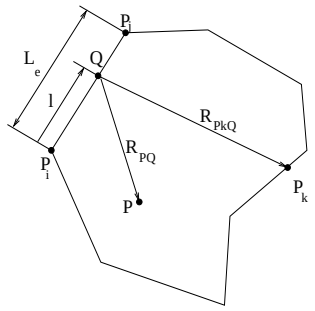


Fig. 7.2. Notaţii relative la un element de frontieră

Se obţine astfel un sistem de $n + ne$ ecuaţii cu $n + ne$ necunoscute din a cărui rezolvare rezultă valorile potenţialelor în toate nodurile frontierei şi valorile derivatei după normală pe toate elementele frontierei.

Calculul analitic al integralelor

Valorile $g(P, P_i, P_j)$, $h1(P, P_i, P_j)$ şi $h2(P, P_i, P_j)$ unde i şi j sunt nodurile elementului e se pot calcula analitic.

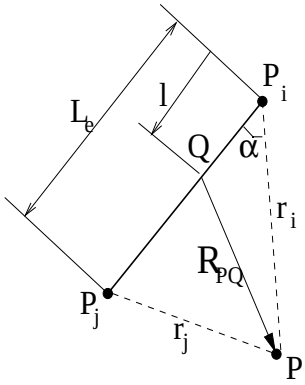


Fig. 7.3. Notaţii folosite în calculul integralei g

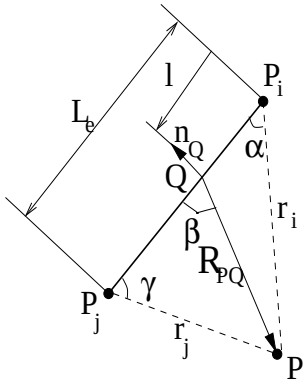


Fig. 7.4. Notaţii folosite în calculul integralelor $h2$ şi $h3$

• **Integrala** $g(P, P_i, P_j)$

Am văzut că:

$$g(P, P_i, P_j) = \int_{P_i}^{P_j} \ln\left(\frac{1}{R_{PQ}}\right) \mathrm{d}l_Q \tag{7.44}$$

Cu notaţiile din figura 7.3 rezultă că:

$$R_{PQ} = \sqrt{r_i^2 + l^2 - 2 r_i l \cos\alpha} \quad \text{unde} \quad 2 r_i L_e \cos\alpha = r_i^2 + L_e^2 - r_j^2 \tag{7.45}$$

Făcând schimbarea de variabilă $x = l/L_e$, integrala g se mai scrie:

$$g(P, P_i, P_j) = -\frac{L_e}{2} \int_0^1 \ln\left[r_i^2 + x^2 L_e^2 - x(r_i^2 + L_e^2 - r_j^2)\right] \mathrm{d}x \tag{7.46}$$

Această integrală se poate calcula analitic, rezultatul fiind:

$$\begin{aligned} g(P, P_i, P_j) \;=\; & -\frac{L}{2}\left(-2 + \frac{\left(-L^4 + 2L^2 ri^2 - ri^4 + 2L^2 rj^2 + 2 ri^2 rj^2 - rj^4\right) \operatorname{ArcTanh}\left(\frac{-L^2 + ri^2 - rj^2}{\sqrt{L^4 - 2L^2 ri^2 + ri^4 - 2L^2 rj^2 - 2 ri^2 rj^2 + rj^4}}\right)}{L^2 \sqrt{L^4 - 2L^2 ri^2 + ri^4 - 2L^2 rj^2 - 2 ri^2 rj^2 + rj^4}} + \right. \\ & + \frac{\sqrt{L^4 - 2L^2 ri^2 + ri^4 - 2L^2 rj^2 - 2 ri^2 rj^2 + rj^4} \operatorname{ArcTanh}\left(\frac{L^2 + ri^2 - rj^2}{\sqrt{L^4 - 2L^2 ri^2 + ri^4 - 2L^2 rj^2 - 2 ri^2 rj^2 + rj^4}}\right)}{L^2} + \\ & \left. + \frac{\left(L^2 + ri^2 - rj^2\right) \log(ri^2)}{2L^2} + \log(rj^2) + \frac{\left(-L^2 - ri^2 + rj^2\right) \log(rj^2)}{2L^2} \right) \end{aligned} \tag{7.47}$$

În relaţia (7.47) am notat $L = L_e$, $ri = r_i$ distanţa de la punctul P la punctul P_i şi cu $rj = r_j$ distanţa de la punctul P la punctul P_j.

Integrala g, scrisă ca în relaţia (7.46) nu este convergentă în cazul în care P este un punct interior segmentului $[P_i, P_j]$. În acest caz se demostrează uşor că ea are expresia:

$$g(P, P_i, P_j) = -r_i \ln(r_i) - r_j \ln(r_j) + L_e \tag{7.48}$$

EXERCIŢIUL 5: Demonstraţi relaţia (7.48).

• **Integrala** $h2(P, P_i, P_j)$

Am văzut că:

$$h2(P, P_i, P_j) = \int_{P_i}^{P_j} \frac{\mathbf{R}_{PQ} \cdot \mathbf{n}_Q}{R_{PQ}^2} \frac{l_{P_iQ}}{L_e} \, \mathrm{d}l_Q \tag{7.49}$$

$$h3(P, P_i, P_j) = \int_{P_i}^{P_j} \frac{\mathbf{R}_{PQ} \cdot \mathbf{n}_Q}{R_{PQ}^2} \, \mathrm{d}l_Q \tag{7.50}$$

Cu notaţiile din figura 7.4 rezultă că:

$$\mathbf{R}_{PQ} \cdot \mathbf{n}_Q = -R_{PQ} \sin\beta = -r_j \sin\gamma = -\frac{2A_{PP_iP_j}}{L_e} \tag{7.51}$$

unde am notat cu $A_{PP_iP_j}$ aria triunghiului determinat de punctele P, P_i şi P_j. Observăm că mărimea $\mathbf{R}_{PQ} \cdot \mathbf{n}_Q$ este constantă atunci când Q parcurge elementul $[P_iP_j]$.

Făcând schimbarea de variabilă $x = l/L_e$, integralele $h2$ şi $h3$ se mai scriu:

$$h2(P, P_i, P_j) = -2A_{PP_iP_j} \int_0^1 \frac{x}{r_i^2 + x^2 L_e^2 - x(r_i^2 + L_e^2 - r_j^2)} \, \mathrm{d}x \tag{7.52}$$

$$h3(P, P_i, P_j) = -2A_{PP_iP_j} \int_0^1 \frac{1}{r_i^2 + x^2 L_e^2 - x(r_i^2 + L_e^2 - r_j^2)} \, \mathrm{d}x \tag{7.53}$$

Aceste integrale se pot calcula analitic, rezultatele fiind:

$$h2(P, P_i, P_j) = -2\,A \left(\frac{\left(L^2 + ri^2 - rj^2\right) \operatorname{ArcTanh}\left(\frac{-L^2+ri^2-rj^2}{\sqrt{L^4-2\,L^2\,ri^2+ri^4-2\,L^2\,rj^2-2\,ri^2\,rj^2+rj^4}}\right)}{L^2 \sqrt{L^4 - 2\,L^2\,ri^2 + ri^4 - 2\,L^2\,rj^2 - 2\,ri^2\,rj^2 + rj^4}} - \right.$$
$$\left. - \frac{\left(L^2 + ri^2 - rj^2\right) \operatorname{ArcTanh}\left(\frac{L^2+ri^2-rj^2}{\sqrt{L^4-2\,L^2\,ri^2+ri^4-2\,L^2\,rj^2-2\,ri^2\,rj^2+rj^4}}\right)}{L^2 \sqrt{L^4 - 2\,L^2\,ri^2 + ri^4 - 2\,L^2\,rj^2 - 2\,ri^2\,rj^2 + rj^4}} - \frac{\log(ri^2)}{2\,L^2} + \frac{\log(rj^2)}{2\,L^2} \right) \tag{7.54}$$

$$h3(P, P_i, P_j) = \frac{-4\,A \left(\operatorname{ArcTanh}\left(\frac{-L^2+ri^2-rj^2}{\sqrt{L^4-2\,L^2\,ri^2+ri^4-2\,L^2\,rj^2-2\,ri^2\,rj^2+rj^4}}\right) - \operatorname{ArcTanh}\left(\frac{L^2+ri^2-rj^2}{\sqrt{L^4-2\,L^2\,ri^2+ri^4-2\,L^2\,rj^2-2\,ri^2\,rj^2+rj^4}}\right) \right)}{\sqrt{L^4 - 2\,L^2\,ri^2 + ri^4 - 2\,L^2\,rj^2 - 2\,ri^2\,rj^2 + rj^4}} \tag{7.55}$$

În relaţiile (7.54) şi (7.55) am notat $L = L_e$, $ri = r_i$ distanţa de la punctul P la punctul P_i, $rj = r_j$ distanţa de la punctul P la punctul P_j şi cu A aria triunghiului determinat de punctele P, P_i şi P_j.

Observaţii:

1. În cazul degenerat în care punctele P, P_i, P_j sunt coliniare integralele $h2$ şi $h3$ sunt nule.
2. Relaţiile deduse pentru $h2$ şi $h3$ sunt valabile dacă normala $\mathbf{n}_Q$ şi vectorul $\mathbf{R}_{PQ}$ formează un unghi obtuz (cazul desenat în figura 7.4). În caz contrar, dacă normala $\mathbf{n}_Q$ şi vectorul $\mathbf{R}_{PQ}$ formează un unghi ascuţit, relaţiile deduse pentru integralele $h2$ şi $h3$ trebuie înmulţite cu -1.

7.1.4 Calculul repartiţiei potenţialului

După determinarea tuturor valorilor şi derivatelor după normală, calculul potenţialului în oricare punct se va face astfel:

$$V(P) = \frac{1}{2\pi}\left(\sum_{e=1}^{ne} g(P,P_i,P_j)V'_e - \sum_{e=1}^{ne} h1(P,P_i,P_j)V_i - \sum_{e=1}^{ne} h2(P,P_i,P_j)V_j\right) \qquad (7.56)$$

unde i şi j sunt nodurile unui element e.

7.2 Implementarea metodei în *Scilab*

7.2.1 Implementarea funcţiilor care calculează integrale

Mai întâi vom implementa funcţiile g, $h1$ şi $h2$, necesare atât asamblării sistemului de ecuaţii cât şi postprocesării problemei.

EXERCIŢIUL 6:

a) În directorul ∼/modelare/tema7 creaţi un director "integrale".
b) În directorul ∼/modelare/tema7/integrale scrieţi un fişier "test.sci" cu următorul conţinut:

```
clear;
getf('functii_gh.sci');
P = [3 5];
Pi = [3 0];
Pj = [3 1];
[g,h1,h2] = calcul_integrale(P,Pi,Pj);
```

EXERCIŢIUL 7:

a) Cu ajutorul comenzii `help` descrieţi sintaxa şi efectul comenzii `integrate`.
b) În directorul ∼/modelare/tema7/integrale scrieţi un fişier "functii_gh.sci" cu următorul conţinut:

```
function [g,h1,h2] = calcul_integrale(P,Pi,Pj)
x = P(1)
y = P(2)
xi = Pi(1)
yi = Pi(2)
xj = Pj(1)
yj = Pj(2)
ri = sqrt((x-xi)^2 + (y-yi)^2)
rj = sqrt((x-xj)^2 + (y-yj)^2)
L = sqrt((xi-xj)^2 + (yi-yj)^2)
M = [1 P; 1 Pi; 1 Pj]
A = 0.5*abs(det(M));
if( (((x-xi)*(x-xj)<0) & ((y-yi)*(y-yj)<0) & ...
    ((x-xi)*(yj-yi)==(y-yi)*(xj-xi))) |...
    ( (xi == xj) & (x == xi) & ...
      ((y-yi)*(y-yj)<0) ) | ...
    ( (yi == yj) & (y == yi) & ...
      ((x-xi)*(x-xj)<0) ) ),
                g = -ri*log(ri) - rj*log(rj) + L;
else
  g = -L/2*integrate('log(ri^2+x^2*L^2 - ...
           x*(ri^2+L^2-rj^2))','x',0,1);
end;
if (A == 0),
    h2 = 0;
else
    h2 = -2*A*integrate('x/(ri^2+x^2*L^2-...
             x*(ri^2+L^2-rj^2))','x',0,1);
    if( (yj-yi)*(x-xi)-(xj-xi)*(y-yi)>0 ),
             h2 = -1*h2;
    end,
end;
if (A == 0),
        h3 = 0;
else
        h3 = -2*A*integrate('1/(ri^2+x^2*L^2-...
                 x*(ri^2+L^2-rj^2))','x',0,1);
     if((yj-yi)*(x-xi)-(xj-xi)*(y-yi)>0),
                 h3 = -1*h3;
     end,
end;
h1 = h3 - h2;
return
```

EXERCIŢIUL 8:

a) Comentaţi fiecare instrucţiune a funcţiei `calcul_integrale`. Cum este aleasă normala la un element?
b) Testaţi corectitudinea funcţiei implementate prin executarea programului "test.sci". Rezultatul trebuie să fie: $g = -1.5020121$, $h1 = 0$, $h2 = 0$. Pentru `P = [7 5]` rezultatele trebuie să fie: $g = -1.7950872$, $h1 = 0.0530434$, $h2 = 0.0576138$.

7.2.2 Implementarea structurilor de date

EXERCIŢIUL 9:

a) Din directorul ∼/modelare/tema4 copiaţi în directorul ∼/modelare/tema7 fişierele "citire_date.sci" şi "grid.sci".
b) În directorul ∼/modelare/tema7 editaţi un fişier "main.sci" cu următorul conţinut:

```
clear;
getf('citire_date.sci');
getf('grid.sci');
getf('noduri.sci');
getf('elemente.sci');
getf('bem.sci');
getf('integrale/functii_gh.sci');

disp('Metoda elementelor de frontiera - ...
              conductor in forma de L');
// --------------- preprocesare ----------------
// citirea si validarea datelor problemei
[a,b,A,B,sigma,V0] = date();
// discretizare frontiera
[x,y,N_A,N_a,N_b,N_Bb] = grid(a,b,A,B);
// generarea structurilor de date
[noduri] = nodes(x,y,V0,N_A,N_a,N_b,N_Bb);
[elemente] = elements(x,y,V0,N_A,N_a,N_b,N_Bb);

// asamblarea matricei coeficientilor M si
// a vectorului termenilor liberi p
[M, p] = asambleaza(noduri, elemente);

// ----------------- rezolvare -----------------
v = inv(M)*p;

// ----------------- postprocesare -------------
exec('postprocesare.sci');
```

Funcţia `date` şi funcţia `grid` sunt cele definite în fişierele "citire_date.sci" şi "grid.sci". Aceste funcţii, precum şi notaţiile a, A, b, B, N_A, N_a, N_b, N_Bb au fost explicate în detaliu la tema 4.

Ne vom concentra acum atenţia asupra structurilor de date specifice metodei elementelor de frontieră. Pentru a construi aceste structuri de date, ne vom alege mai întâi un mod de a numerota nodurile şi elementele de frontieră. Vom parcurge frontiera în sens trigonometric, începând din punctul S, aşa cum se arată în figura 7.5.

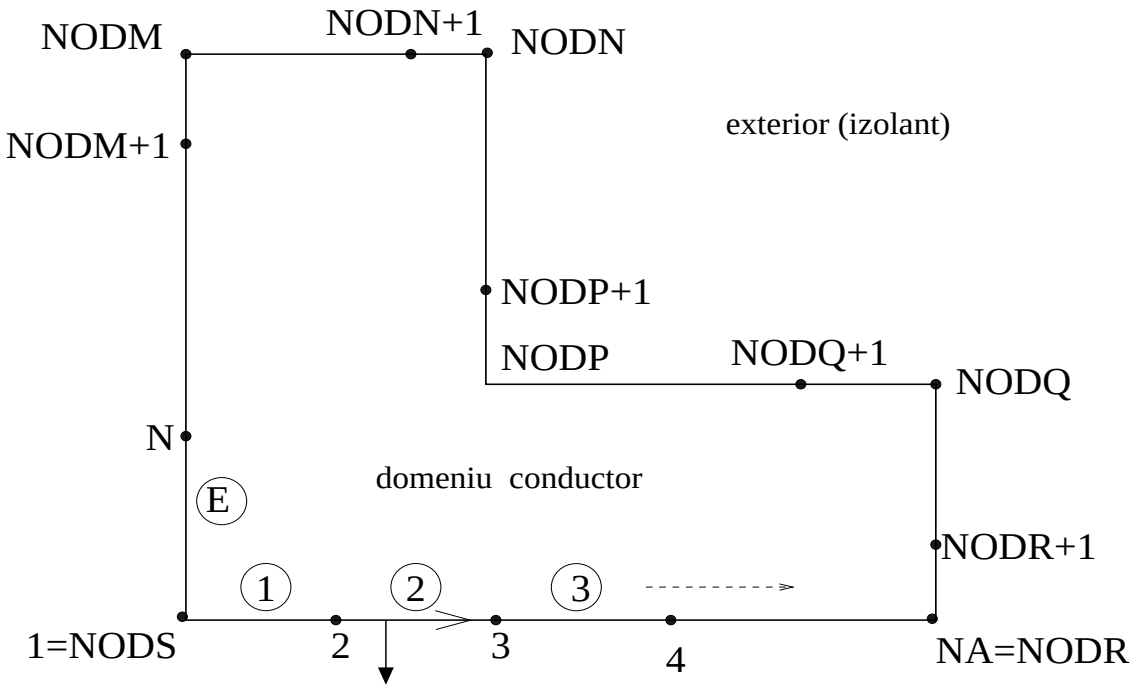


Fig. 7.5. Modul în care se face numerotarea nodurilor şi a elementelor (N este numărul de noduri şi $E = ne$ este numărul de elemente). Numerele încercuite reprezintă elemente, de exemplu: elementul 2 are nodul iniţial 2 şi nodul final 3. Prin definiţie, normala la un element este exterioară domeniului.

Informaţiile legate de noduri vor fi memorate în matricea `noduri`. Aceasta este o matrice cu cinci coloane şi cu un număr de linii egal cu numărul de noduri. Linia i a acestei matrice corespunde nodului numărul i, şi ea conţine următoarele informaţii (vezi şi figura 7.6):

1. Pe prima coloană se memorează tipul nodului (Dirichlet sau Neumann).
2. A doua coloană conţine abscisa nodului.
3. A treia coloană conţine ordonata nodului.
4. A patra coloană conţine valoarea condiţiei de frontieră. Acest număr are semnificaţie doar dacă nodul este Dirichlet.
5. A cincea coloană conţine unghiul interior al domeniului în acel nod. Această informaţie va fi utilă la preprocesare doar dacă nodul este de tip Neumann.

linia 1	20	0	0	0	π/2
linia N	20	0		0	π

Fig. 7.6. Matricea nodurilor: 10 = nod Dirichlet, 20 = nod Neumann

linia 1	20	1	2	0
linia E	20	N	1	0

Fig. 7.7. Matricea elementelor: 10 = segment Dirichlet, 20 = segment Neumann

Informaţiile legate de elemente vor fi memorate în matricea `elemente`. Aceasta este o matrice cu patru coloane şi cu un număr de linii egal cu numărul de elemente. Linia i a acestei matrice corespunde elementului numărul i, şi ea conţine următoarele informaţii (vezi şi figura 7.7):

1. Pe prima coloană se memorează tipul elementului (Dirichlet sau Neumann).
2. A doua coloană conţine numărul nodului iniţial.
3. A treia coloană conţine numărul nodului final.
4. A patra coloană conţine valoarea condiţiei de frontieră. Acest număr are semnificaţie doar dacă elementul este de tip Neumann.

Observaţie: Practic nu este nevoie de o orientare a elementului, dar, pentru uşurinţa scrierii programului, vom construi această matrice astfel încât un element este parcurs de la nodul iniţial spre nodul final, iar normala lui este exterioară domeniului (vezi figura 7.5, elementul 2).

EXERCIŢIUL 10:

În directorul ∼/modelare/tema7 editaţi un fişier "noduri.sci" cu următorul conţinut:

```
function [noduri]=nodes(x,y,V0,N_A,N_a,N_b,N_Bb)
NODS = 1;
NODR = N_A;
NODQ = NODR + N_b -1;
NODP = NODQ + N_A - N_a;
NODN = NODP + N_Bb;
NODM = NODN + N_a - 1;
NR_NODURI = NODM + N_b + N_Bb - 2;
tip = 1;
Dirichlet = 10; Neumann = 20;
x_nod = 2; y_nod = 3;
```

Continuarea exerciţiului pe pagina următoare

EXERCIŢIUL 10:

```
CF_nod = 4;  beta_nod = 5;
noduri = zeros(NR_NODURI,5);
for i = NODS : NODR-1,
        noduri(i,tip) = Neumann,
        j = i - NODS + 1,
        noduri(i,x_nod) = x(j),
        noduri(i,y_nod) = y(1),
end;
for i = NODR : NODQ
        noduri(i,tip) = Dirichlet,
        j = i - NODR + 1,
        noduri(i,x_nod) = x(N_A),
        noduri(i,y_nod) = y(j),
        noduri(i,CF_nod) = 0,
end;
for i = NODQ+1 : NODP
        noduri(i,tip) = Neumann,
        j = N_A - (i - NODQ) ,
        noduri(i,x_nod) = x(j),
        noduri(i,y_nod) = y(N_b),
end;
for i = NODP+1 : NODN-1
        noduri(i,tip) = Neumann,
        j = N_b + i - NODP ,
        noduri(i,x_nod) = x(N_a),
        noduri(i,y_nod) = y(j),
end;
for i = NODN : NODM
        noduri(i,tip) = Dirichlet,
        j = N_a - (i - NODN);
        noduri(i,x_nod) = x(j),
        noduri(i,y_nod) = y(N_b + N_Bb),
        noduri(i,CF_nod) = V0,
end;
for i = NODM+1 : NR_NODURI
        noduri(i,tip) = Neumann,
        j = N_b + N_Bb - (i - NODM),
        noduri(i,x_nod) = x(1),
        noduri(i,y_nod) = y(j),
end;
for i = 1:NR_NODURI,
        noduri(i,beta_nod) = %pi,
end;
noduri(NODS,beta_nod) = %pi/2;
noduri(NODP,beta_nod) = 3*%pi/2;
noduri(NODR,beta_nod) = %pi/2;
noduri(NODQ,beta_nod) = %pi/2;
noduri(NODN,beta_nod) = %pi/2;
noduri(NODM,beta_nod) = %pi/2;
return
```

EXERCIŢIUL 11:

a) Comentaţi fiecare instrucţiune a funcţiei `nodes`.
b) Testaţi corectitudinea funcţiei pentru cazul foarte simplu în care `x = [0 1 2 3]` şi `y = [0 1 2]`.

EXERCIŢIUL 12:

În directorul ∼/modelare/tema7 editaţi un fişier "elemente.sci" cu următorul conţinut:

```
function [elemente]=elements(x,y,V0,N_A,N_a,...
                                        N_b,N_Bb)
NODS = 1;
NODR = N_A;
NODQ = NODR + N_b -1;
NODP = NODQ + N_A - N_a;
NODN = NODP + N_Bb;
NODM = NODN + N_a - 1;
NR_NODURI = NODM + N_b + N_Bb - 2;
tip = 1;
Dirichlet = 10; Neumann = 20;
nod_initial = 2;
nod_final = 3;
CF_segment = 4;
NR_ELEMENTE = NR_NODURI;
elemente = zeros(NR_ELEMENTE,4);
for e = 1:NR_ELEMENTE,
        elemente(e,nod_initial) = e,
        elemente(e,nod_final) = e+1,
end;
elemente(NR_ELEMENTE,nod_final) = 1;
for e = 1: NODR-1,
        elemente(e,tip) = Neumann,
        elemente(e,CF_segment) = 0,
end;
for e = NODR: NODQ-1,
        elemente(e,tip) = Dirichlet,
end;
for e = NODQ: NODN-1,
        elemente(e,tip) = Neumann,
        elemente(e,CF_segment) = 0,
end;
for e = NODN: NODM-1,
        elemente(e,tip) = Dirichlet,
end;
for e = NODM: NR_ELEMENTE,
        elemente(e,tip) = Neumann,
        elemente(e,CF_segment) = 0,
end;
return;
```

EXERCIŢIUL 13:

a) Comentaţi fiecare instrucţiune a funcţiei `elements`.
b) Testaţi corectitudinea funcţiei pentru cazul foarte simplu în care `x = [0 1 2 3]` şi `y = [0 1 2]`.

7.2.3 Asamblarea sistemului de ecuaţii

Sistemul de ecuaţii este de forma `M v = p`, unde `M` este o matrice pătrată de dimensiune egală cu numărul de noduri n plus numărul de elemente ne. Vectorul necunoscutelor `v` este un vector coloană în care primele n linii corespund potenţialelor nodurilor iar următoarele ne linii corespund derivatelor după normală pe elementele de frontieră.

EXERCIŢIUL 14:

În directorul ∼/modelare/tema7 editaţi un fişier "bem.sci" cu următorul conţinut:

```
function [M, p] = asambleaza(noduri, elemente)

temp = size(noduri);
NR_NODURI = temp(1,1);

temp = size(elemente);
NR_ELEMENTE = temp(1,1);

nr_nec = NR_NODURI + NR_ELEMENTE;
M = zeros(nr_nec,nr_nec);
p = zeros(nr_nec,1);
Dirichlet = 10;
Neumann = 20;
for i = 1:NR_NODURI,
    if (noduri(i,1) == Dirichlet),
          M(i,i) = 1;
              p(i) = noduri(i,4);
          else
              Pk = [noduri(i,2) noduri(i,3)];
              betak = noduri(i,5);
              M(i,i) = betak;
              for e = 1:NR_ELEMENTE,
              M(i,:) = asambleaza_linie(M(i,:), ...
                        e, Pk,elemente, noduri,...
                                NR_NODURI);
          end,
    end,
end;
```

Continuarea exerciţiului pe pagina următoare

EXERCIŢIUL 14:

```
for i = 1:NR_ELEMENTE,
        j = NR_NODURI + i;
        if (elemente(i,1) == Neumann),
                M(j,j) = 1;
                p(j) = elemente(i,4);
        else
                nod0 = elemente(i,2);
                nod1 = elemente(i,3);
                Pk(1,1) = (noduri(nod0,2)+...
                           noduri(nod1,2))/2;
                Pk(1,2) = (noduri(nod0,3)+...
                           noduri(nod1,3))/2;
                betak = %pi;
                M(j,nod0) = betak/2;
                M(j,nod1) = betak/2;
                for e = 1:NR_ELEMENTE,
                  M(j,:) = asambleaza_linie(...
                               M(j,:),e,Pk,...
                               elemente,...
                               noduri,...
                               NR_NODURI);
                end,
        end,
end;
return;

function v = asambleaza_linie(vector,e,Pk,...
                    elemente,noduri,NR_NODURI)
v = vector;
nodi = elemente(e,2);
nodj = elemente(e,3);
Pi = [noduri(nodi,2) noduri(nodi,3)]
Pj = [noduri(nodj,2) noduri(nodj,3)]
[g h1 h2] = calcul_integrale(Pk,Pi,Pj);
v(nodi) = v(nodi) + h1;
v(nodj) = v(nodj) + h2;
v(NR_NODURI+e) = v(NR_NODURI+e) - g;
return
```

EXERCIŢIUL 15:

a) Comentaţi fiecare instrucţiune din exerciţiul 14.
b) Verificaţi corectitudinea programului implementat până acum: pentru `x = [0 1 2 3]` şi `y = [0 1 2]`, vectorul v are componentele:

```
v' = [.7867020 .7086116 .4440148 0 0 .5745304...
1 1 1 .8433378 0 0 0 -.5005025 0 0 .2941569...
.1800088 0 0]
```

7.2.4 Postprocesarea

Având valorile potenţialului şi derivatei lui după normală în oricare punct al frontierei, putem calcula potenţialul V oriunde în interiorul domeniului cu formula (7.56). Cu o formulă similară (în care împărţirea se face la β şi nu la 2π) se poate calcula potenţialul şi în orice punct de pe frontieră.

Pentru trasarea liniilor echipotenţiale vom calcula potenţialul V în punctele unei grile determinate de doi vectori `abscise` şi `ordonate`.

EXERCIŢIUL 16:

În directorul ∼/modelare/tema7 editaţi un fişier "post-procesare.sci" cu următorul conţinut:

```
getf('unghi.sci');
getf('calcul_v.sci');

nrsegm_x = 6;      nrsegm_y = 4;
pasx = A/nrsegm_x; pasy = B/nrsegm_y;
abscise = 0:pasx:A;
ordonate = 0:pasy:B;
NX = length(abscise); NY = length(ordonate);
matrice_pot = zeros(NX,NY);

for i = 1: NX,
       for j = 1: NY,
              val_x = abscise(i);
              val_y = ordonate(j);
              Pk = [val_x val_y];
              val = calcul_potential(Pk,...
                        elemente,noduri,v);
              beta = calcul_unghi(val_x,...
                           val_y,a,A,b,B);
              if beta == 0,
                  matrice_pot(i,j) = 0;
              else
                  matrice_pot(i,j) = val/beta;
              end;
       end,
end
if (A > B)
     xsetech([0,0,1,B/A]);
else
     xsetech([0,0,A/B,1]);
end
contour(abscise,ordonate,matrice_pot,10);
xrects([2;2;1;1],4);
```

EXERCIŢIUL 17:

În directorul ∼/modelare/tema7 editaţi un fişier "unghi.sci" cu următorul conţinut:

```
function beta = calcul_unghi(valx,valy,a,A,b,B)
beta = 2*%pi;
if valy == 0,
        if ((valx == 0)|(valx == A)),
                beta = %pi/2;
        else
                beta = %pi;
        end,
end;
if valy == b,
        if ((valx ==0)|((valx>a)&(valx<A))),
                beta = %pi;
        elseif valx == a,
                beta = 3*%pi/2;
        elseif valx == A,
                beta = %pi/2;
        end,
end;
if valy == B,
        if ((valx == 0) | (valx == a)),
                beta = %pi/2;
        elseif ((valx > 0)&(valx < a)),
                beta = %pi;
        else
                beta = 0;
        end,
end;
if valx == 0,
        if ((valy > 0)& (valy < B)),
                beta = %pi;
        end,
end;
if valx == a
        if ((valy > b) & (valy < B)),
                beta = %pi;
        end,
end;
if valx == A,
        if ((valy > 0) & (valy < b)),
                beta = %pi;
        end,
end;
if (((valx>a)&(valy>b))|(valx>A)|...
    (valx<0)|(valy<0)|(valy>B)),
        beta = 0;
end;
return
```

EXERCIŢIUL 18:

În directorul ∼/modelare/tema7 editaţi un fişier "calcul_v.sci" cu următorul conţinut:

```
function val = calcul_potential(Pk,...
                      elemente,noduri,v)
temp = size(noduri);
NR_NODURI = temp(1,1);
temp = size(elemente);
NR_ELEMENTE = temp(1,1);
val = 0;
for e = 1:NR_ELEMENTE,
        nodi = elemente(e,2);
        nodj = elemente(e,3);
        Pi = [noduri(nodi,2) noduri(nodi,3)]
        Pj = [noduri(nodj,2) noduri(nodj,3)]
        [g h1 h2] = calcul_integrale(Pk,Pi,Pj)
        val = val + g*v(NR_NODURI+e);
        val = val - h1*v(nodi);
        val = val - h2*v(nodj);
end
// val trebuie impartita la unghi
return
```

EXERCIŢIUL 19:

a) Comentaţi fiecare instrucţiune a funcţiei `calcul_unghi`.
b) Comentaţi fiecare instrucţiune a funcţiei `calcul_potential`.

Programul este acum complet. Executându-l ar trebui să obţineţi figura 7.8.

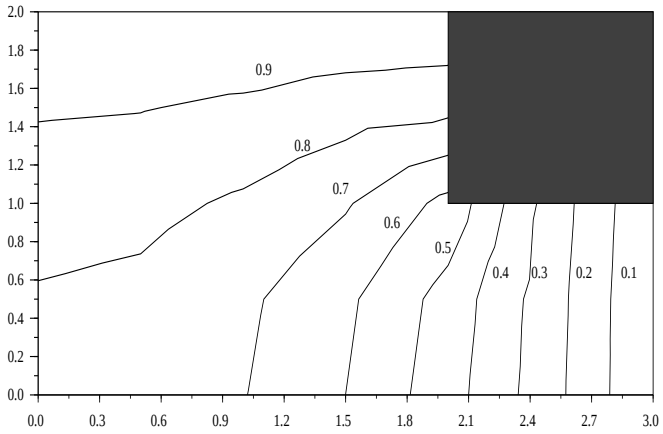


Fig. 7.8. Linii echipotenţiale

EXERCIŢIUL 20:	a) Calculaţi potenţialul (cu ajutorul funcţiei `calcul_v`) în nodurile Dirichlet. Ce observaţi? b) Calculaţi potenţialul (cu ajutorul funcţiei `calcul_v`) în mijloacele segmentelor Dirichlet. Comentaţi.

EXERCIŢIUL 21:	a) Studiaţi proprietăţile matricei M. b) Analizaţi complexitatea algoritmului din punct de vedere al timpului de calcul şi a necesarului de memorie.

EXERCIŢIUL 22:	Comparaţi rezultatele numerice ale acestei metode cu rezultatele obţinute în celelalte metode învăţate (volume finite, diferenţe finite, elemente finite).

Bibliografie

[1] ***. The Linux encyclopedia.
`http://www-ocean.tamu.edu/ baum/linuxlist.html/`.

[2] ***. Scilab: A free CACSD package by INRIA.
`http:///www-rocq.inria.fr/scilab/`.

[3] ***. *Drx. Linux. The Linux Documentation Project*. LSL, 1994.

[4] K. J. Binns, P. J. Lawrenson şi C. W. Trowbridge. *The Analytical and Numerical Solution of Electric and Magnetic Fields*. John Wiley & Sons, West Sussex, England, 1992.

[5] J. F. Botha şi G. F. Pinder. *Fundamental Concepts in the Numerical Solution of Differential Equations*. John Wiley and Sons, Inc., New York, 1983.

[6] I. Holland şi K. Bell, editori. *Finite Element Methods in Stress Analysis*. Tapir, Trondheim, Norway, 1969.

[7] I. F. Hănţilă. *Calculul câmpului electromagnetic cu ajutorul calculatorului*. PS-CAEE, Bucureşti, Romania, 1993.

[8] Kamran Husain şi Tim Parker. *Linux Unleashed*. Sams Publishing, Indianopolis, 1995.

[9] Daniel Ioan. *Metode pentru calculul cîmpului electromagnetic*. Institutul Politehnic Bucureşti, 1988.

[10] Daniel Ioan, Irina Munteanu, Bogdan Ionescu, Mihai Popescu, Radu Popa, Mihai Lăzărescu şi Gabriela Ciuprina. *Metode numerice în ingineria electrică*. MATRIX ROM, Bucureşti, România, 1998.

[11] J.R.Brauer, G.E. Barron şi Nancy J. Lambert. *What Every Engineer Should Know About Finite Element Analysis*. Marcel Dekker Inc., New York, 1988.

[12] K. S. Kunz şi R. J. Luebbers. *The Finite Difference Time Domain Method for Electromagnetics*. CRC Press, Boca Raton, FL, 1993.

[13] C. I. Mocanu. *Teoria câmpului electromagnetic*. Editura Didactică şi Pedagogică, Bucureşti, 1981.

[14] Irina Munteanu. *Calculul numeric al câmpului electromagnetic. Contribuţii la analiza câmpului electromagnetic general variabil prin metoda volumelor finite*. Teză de doctorat, Universitatea "Politehnica" Bucureşti, Romania, 1997.

[15] William H. Press, Saul A. Teukolsky, William T. Vetterling şi Brian P. Flannery. *Numerical Recipes in C: the art of scientific computing*. Cambridge University Press, Cambridge, UK, 1992.

[16] M. N. O. Sadiku. *Numerical Techniques in Electromagnetics*, pp. 179–204. CRC Press, Boca Raton, 1992.

[17] P. P. Silvester şi R. L. Ferrari. *Finite Elements for Electrical Engineers*. Cambridge Universtity Press, Cambridge, 1991.

[18] A. Taflove. *Computational Electrodynamics: The Finite-Difference Time-Domain Method*. Artech House, Boston, MA, 1995.

[19] A. Taflove, editor. *Advances in Computational Electrodynamics: The Finite-Difference Time-Domain Method*. Artech House, Boston, MA, 1998.

[20] O. C. Zienkiewicz şi R. L. Taylor. *The Finite Element Method*, vol. 1. McGraw Hill, London, 1989.

Anexa A

Cum se desfăşoară laboratorul *Modelarea numerică a câmpului electromagnetic*?

Pe parcursul laboratorului veţi parcurge un număr de teme. Laboratorul se bazează pe **lucru individual**, cadrul didactic stându-vă la dispoziţie pentru răspunsurile la întrebări.

- Pentru a folosi la maximum timpul de lucru în faţa calculatorului, este bine să citiţi tema de laborator înainte de începerea orei.
- Parcurgeţi fiecare temă în ritmul Dumneavoastră.
- Pe parcursul temei veţi întâlni exerciţii marcate astfel:

EXERCIŢIUL 1: Citiţi şi comentaţi acest text.

  Notaţi la fiecare exerciţiu rezultatul, sau comentariul cerut.

- "Referatul de laborator" constă în răspunsul la aceste exerciţii şi trebuie predat **la sfârşitul orei**. Începând cu tema a doua, răspunsurile la exerciţii vor fi scrise direct într-un fişier şi trimise prin poştă electronică cadrului didactic.
- Laboratorul în care veţi efectua lucrările prezentate în acest volum face parte din Laboratorul de Metode Numerice (LMN) din Catedra de Electrotehnică, Facultatea de Electrotehnică din Universitatea "Politehnica" Bucureşti.

  Intrând în acest laborator vă angajaţi să respectaţi regulamentul de funcţionare al LMN, prezentat în Anexa B.

	Citiţi acum, cu atenţie, Regulamentul de funcţionare din Anexa B şi întrebaţi cadrul didactic cu privire la orice nelămurire.
EXERCIŢIUL 2:	Veţi semna după aceea o "Fişă de deschidere de cont", iar cadrul didactic vă va comunica numele de utilizator şi parola Dvs. Din acest moment puteţi începe să lucraţi la Tema 1.

Anexa B

Regulamentul Laboratorului de Metode Numerice

Instrucţiuni de respectat la folosirea reţelei de calculatoare locale din Laboratorul de Metode Numerice (`lmn.pub.ro`)

Oricine intră în laboratorul de metode numerice şi are deschis un cont activ pe reţeaua locală de calculatoare trebuie să cunoască următoarele reguli.

1. La fiecare intrare in sistem (login time) veţi primi diferite mesaje de care trebuie sa ţineti cont. Acestea sunt scrise de către administratorul de sistem şi au forma:

   ```
   ##########################################
   Incercati sa stergeti din fisierele care nu
   mai sunt necesare
   ##########################################
   ```

2. Încercati să vă ţineţi cât mai actuale datele personale cu ajutorul comenzii chfn şi a fisierului .plan
3. Accesul cu diskete este total interzis. Orice problemă care cere introducerea unei diskete din exteriorul laboratorului va fi soluţionată numai de administrator.
4. Accesul fizic la calculatoare se va face numai prin intermediul tastaturii şi a mouse-ului. Nu acţionati nici un alt buton. Nu deconectaţi/conectaţi cabluri.
5. Întrucât aveţi şi acces direct la reţeaua Internet va trebui sa fiţi un bun locuitor al acesteia. Este interzis accesul în orice calculator din lume dacă nu aveţi un cont personal. Orice încercare de pătrundere frauduloasă în orice sistem sau de producere a oricăror daune asupra datelor din alte sisteme se va pedepsi foarte drastic.
6. Parola este personală şi este interzisă folosirea contului de către orice altă persoană în afară de posesorul de drept al acestuia. Nu incredintaţi parola nimănui altcuiva.

Schimbaţi parola cel puţin lunar cu ajutorul comenzii `passwd`.

Alegeţi o parolă bună, în primul rând foarte diferită de numele de cont. Nici data de naştere, numele mic, numele de familie nu constituie parole bune.

Parola trebuie să aibă 6-8 caractere, litere şi cifre.

7. Este interzisă înlesnirea accesului altor persoane în contul propriu via .rhosts.
8. Încercaţi să vă rezolvaţi singuri problemele prin metode soft care nu cer drepturi de root. Când sunteţi siguri că nu se poate, scrieţi o hârtie cu datele amănunţite ale problemei, cum aţi încercat s-o rezolvaţi şi numele dumneavoastră de cont. Mai sigur prin mail, la adresa:

   `help@lmn.pub.ro`
9. Orice resursă costă. De aceea, nu aveţi dreptul să folosiţi imprimanta decât pentru a imprima lucrările, referatele, etc. realizate de Dvs. în cadrul activităţilor de laborator sau de elaborare a lucrării de diplomă. Este interzisă imprimarea documentelor de pe Internet, a cursurilor, etc. fără aprobarea şefului laboratorului, prof. Daniel Ioan.

Metodele numerice pentru analiza câmpului electromagnetic reprezintă una din "pietrele de încercare" ale oricărui student în ingineria electrică, prin varietatea şi complexitatea lor teoretică şi algoritmică.

Modelarea numerică a câmpului electromagnetic prin programe Scilab reprezintă un bun punct de plecare în înţelegerea şi aprofundarea principalelor metode numerice de calcul al câmpului electromagnetic: volume finite, diferenţe finite, reziduuri ponderate – în particular implementarea acestora prin elemente finite şi elemente de frontieră.

Elementele teoretice expuse sunt cele esenţiale pentru dezvoltarea unui algoritm, iar exemplele de caz, deşi de o complexitate geometrică simplă, sunt ilustrative pentru metodele prezentate. Algoritmii propuşi în carte sunt cele mai simple forme ale celor utilizaţi în pachetele profesionale, scopul fiind de a uşura întelegerea principiului metodelor, şi nu de a demonstra cele mai avansate tehnici de implementare a acestora.

Scilab a fost ales ca mediu pentru implementarea algoritmilor, pentru două motive: asemănarea limbajului cu binecunoscutul Matlab şi faptul că programul se încadrează în categoria "free software", fiind astfel accesibil oricărui posesor de calculator.