Algoritmi numerici pentru analiza circuitelor electrice rezistive neliniare

Prof.dr.ing. Gabriela Ciuprina

Universitatea "Politehnica" Bucureşti, Facultatea de Inginerie Electrică,
Departamentul de Electrotehnică

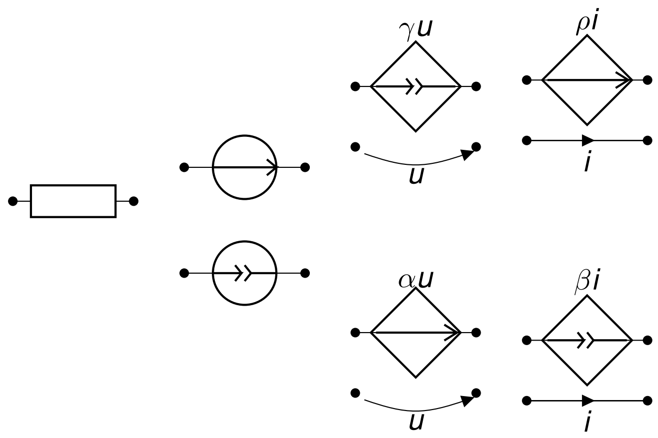
Suport didactic pentru disciplina *Algoritmi numerici*,
Facultatea de Inginerie Electrică, 2017-2018

Notes

Cuprins

Notes

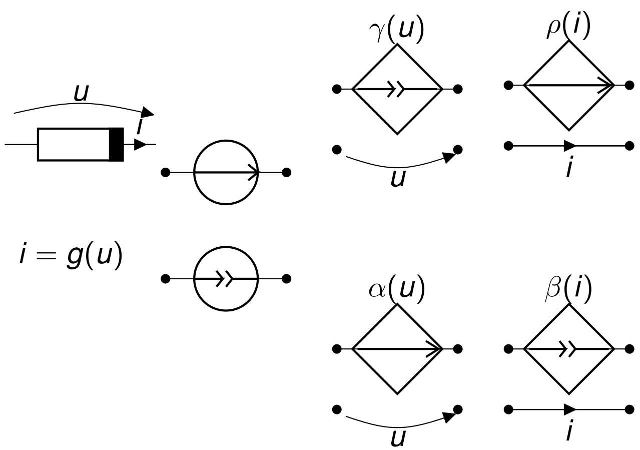
Elemente ideale - rezistive, liniare



Liniare

Notes

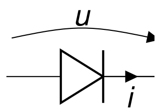
Elemente ideale - rezistive, neliniare



Neliniare

Notes

Elemente reale - rezistive, neliniare



$i = g(u)$

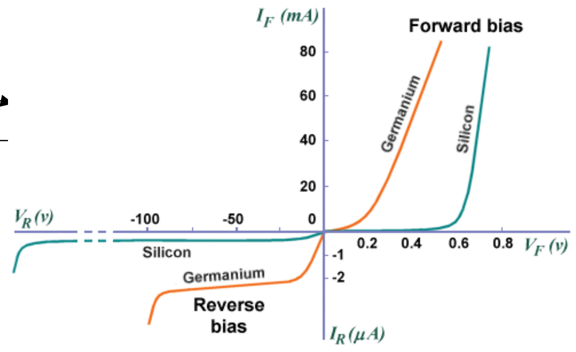


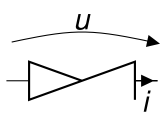
Figura este preluată de la

https://www.technologyuk.net/physics/

Notes

Introducere
Metoda nodală clasică
Descrierea caracteristicilor neliniare
Algoritmi
Elemente de circuit rezistive neliniare
Formularea problemei
Ecuaţii
Exemple

Elemente reale - rezistive, neliniare



$i = g(u)$

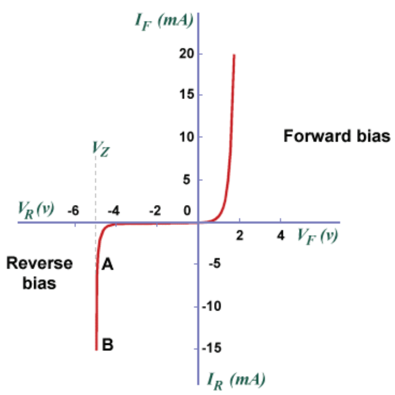


Figura este preluată de la

https://www.technologyuk.net/physics/

Notes

Analiza circuitelor electrice rezistive neliniare (c.c.)

Date:

- *Topologia circuitului* (graful circuitului) - poate fi descris:
  - geometric;
  - numeric (matrice topologice/ *netlist*);
- Pentru fiecare latură liniară k:
  - tipul laturii (R,SUCU,SICI,SICU,SUCI, SIT,SIC);
  - caracteristica constitutivă
    - R_k;
    - parametrul de transfer α_k, β_k, γ_k, ρ_k;
    - semnalul de comandă (curent/tensiune, latură/noduri);
    - parametrii surselor: (e_k, j_k)

Notes

Analiza circuitelor electrice rezistive neliniare (c.c.)

- Pentru fiecare latură neliniară k:
  - tipul laturii (Rn,SUCUn,SICIn,SICUn,SUCIn);
  - caracteristica constitutivă neliniaa
    - $f_k(i)$ dacă controlul este în curent sau $g_k(u)$ dacă controlul este în tensiune;
    - dependenţele $\alpha_k(u)$, $\beta_k(i)$, $\gamma_k(u)$, $\rho_k(i)$;
    - semnalul de comandă (curent/tensiune, latură/noduri);

Se cer: $i_k(t)$, $u_k(t)$, $k = 1, 2, \ldots, L$.

Notes

Ca la c.c. - cazul elementelor liniare

1. Kirchhoff I
2. Kirchhoff II
3. Ecuaţii constitutive pentru elementele rezistive liniare:
   - laturi de tip SRC, SRT;
   - laturi de tip SIC, SIT;
   - laturi de tip SUCU, SICI, SUCI, SICU - comandate liniar.

relaţii algebrice

DAR

Notes

Elementele rezistive neliniare

Ecuaţii constitutive pentru elementele rezistive neliniare:

- rezistoare neliniare;
- surse comandate neliniar;

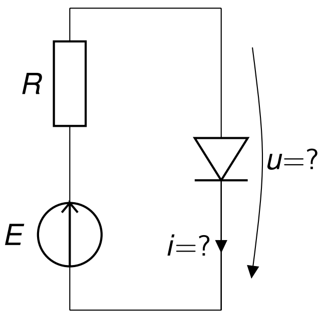
relaţii algebrice neliniare

Sistemul de rezolvat va fi un sistem algebric neliniar

Ce se întâmplă dacă surselor independente sunt variabile în timp?

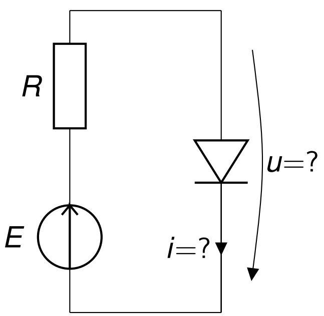
Notes

Exemplul 1



Notes

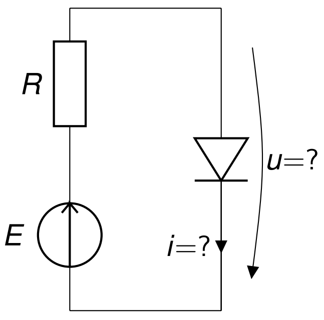
Exemplul 1



$$i = g(u)$$
$$i = \frac{E - u}{R}$$

Notes

Exemplul 1



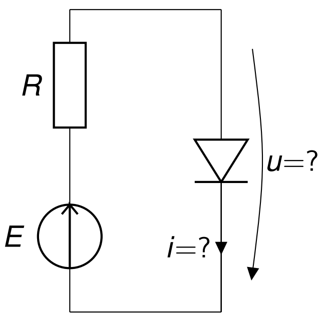
$$i = g(u)$$
$$i = \frac{E - u}{R}$$

$E = 1.25\text{V},\ R = 1.25\text{m}\Omega$

Notes

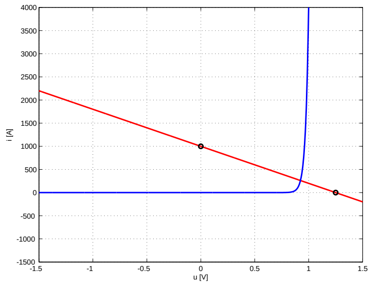
Introducere
Metoda nodală clasică
Descrierea caracteristicilor neliniare
Algoritmi
Elemente de circuit rezistive neliniare
Formularea problemei
Ecuaţii
Exemple

Exemplul 1



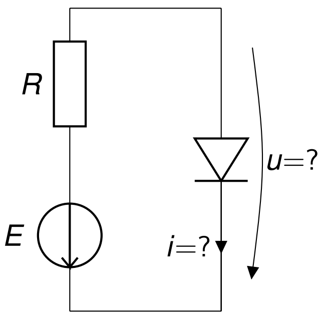
$$i = g(u)$$
$$i = \frac{E - u}{R}$$

$E = 1.25\text{V},\ R = 1.25\text{m}\Omega$



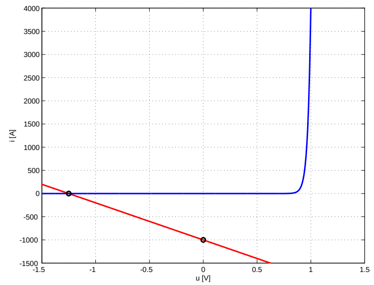
Notes

Exemplul 2



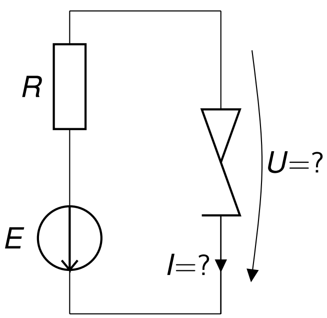
$$i = g(u)$$
$$i = \frac{-E-u}{R}$$

$E = 1.25\text{V},\ R = 1.25\text{m}\Omega$



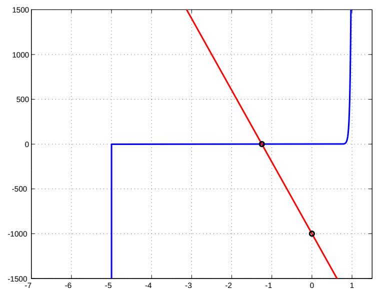
Notes

Exemplul 3 a)



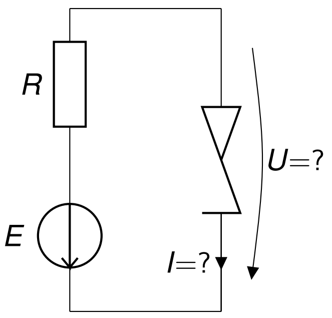
$$i = g(u)$$
$$i = \frac{-E-u}{R}$$

$E = 1.25\text{V},\ R = 1.25\text{m}\Omega$



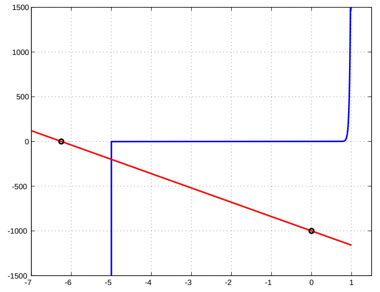
Notes

Exemplul 3 b)



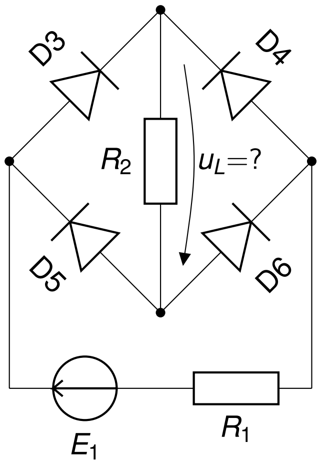
$$i = g(u)$$
$$i = \frac{-E-u}{R}$$

$E = 5 \cdot 1.25\text{V},\ R = 5 \cdot 1.25\text{m}\Omega$



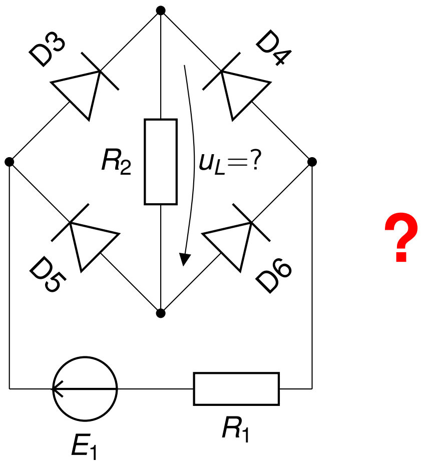
Notes

Exemplul 4



Notes

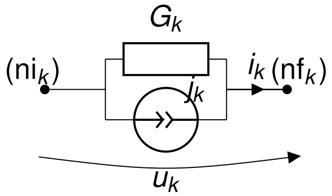
Exemplul 4



Notes

Laturi controlate în tensiune

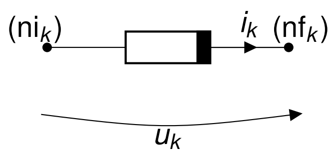
Cazul liniar (SRC)



$$i_k = G_k u_k + j_k$$

$\mathbf{i} = \mathbf{G}\mathbf{u} + \mathbf{j}$
$\mathbf{G} = \mathrm{diag}\{G_1, G_2, \ldots, G_L\}$
$\mathbf{G} \in \mathbb{R}^{L\times L}$
$\mathbf{u}, \mathbf{j}, \mathbf{i} \in \mathbb{R}^{L\times 1}$

Cazul neliniar



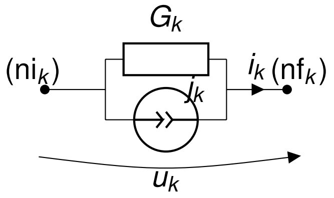
$$i_k = g_k(u_k)$$

$\mathbf{i} = \mathbf{G}(\mathbf{u})$
$\mathbf{G} = [g_1, g_2, \ldots, g_L]^T$
$\mathbf{G} : \mathbb{R}^L \to \mathbb{R}^L$
$\mathbf{u}, \mathbf{i} \in \mathbb{R}^{L\times 1}$

Notes

Laturi controlate în tensiune

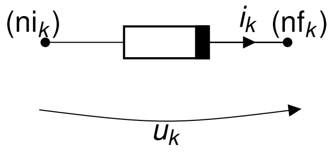
Cazul liniar (SRC)



$$i_k = G_k u_k + j_k$$

$\mathbf{i} = \mathbf{G}\mathbf{u} + \mathbf{j}$
$\mathbf{A}\mathbf{i} = \mathbf{0}$
$\mathbf{u} = \mathbf{A}^T\mathbf{V}$
$\mathbf{A}(\mathbf{G}\mathbf{A}^T\mathbf{V} + \mathbf{j}) = \mathbf{0}$

Cazul neliniar



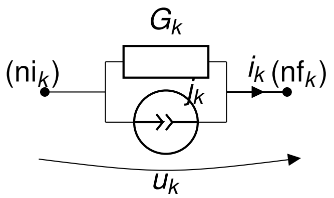
$$i_k = g_k(u_k)$$

$\mathbf{i} = \mathbf{G}(\mathbf{u})$
$\mathbf{A}\mathbf{i} = \mathbf{0}$
$\mathbf{u} = \mathbf{A}^T\mathbf{V}$
$\mathbf{A}(\mathbf{G}(\mathbf{A}^T\mathbf{V})) = \mathbf{0}$

Notes

Laturi controlate în tensiune

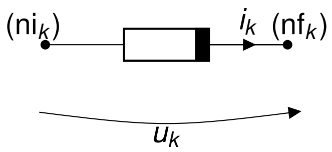
Cazul liniar (SRC)



$$i_k = G_k u_k + j_k$$

$\mathbf{i} = \mathbf{G}\mathbf{u} + \mathbf{j}$
$\mathbf{A}\mathbf{i} = \mathbf{0}$
$\mathbf{u} = \mathbf{A}^T\mathbf{V}$
$\mathbf{A}\mathbf{G}\mathbf{A}^T\mathbf{V} = -\mathbf{A}\mathbf{j}$

Cazul neliniar



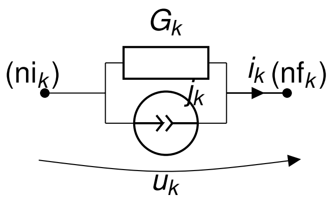
$$i_k = g_k(u_k)$$

$\mathbf{i} = \mathbf{G}(\mathbf{u})$
$\mathbf{A}\mathbf{i} = \mathbf{0}$
$\mathbf{u} = \mathbf{A}^T\mathbf{V}$
$\mathbf{A}\mathbf{G}(\mathbf{A}^T\mathbf{V}) = \mathbf{0}$

Notes

Laturi controlate în tensiune

Cazul liniar (SRC)

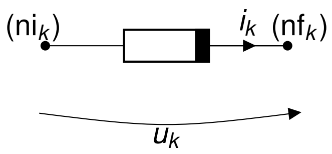


$$i_k = G_k u_k + j_k$$

$\mathbf{AGA}^T\mathbf{V} = -\mathbf{Aj}$

Sistem algebric liniar

Cazul neliniar



$$i_k = g_k(u_k)$$

$\mathbf{AG}(\mathbf{A}^T\mathbf{V}) = \mathbf{0}$

Sistem algebric neliniar

$\mathbf{F}(\mathbf{V}) = \mathbf{0}$ unde

$\mathbf{F}(\mathbf{V}) = \mathbf{AG}(\mathbf{A}^T\mathbf{V})$

$\mathbf{F} : \mathbb{R}^{(N-1)} \to \mathbb{R}^{(N-1)}$

Notes

Dioda semiconductoare

Modelul exponenţial (de exemplu modelul cu parametrii I_s şi u_T)

$$i(u) = I_s\left(e^{\frac{u}{u_T}} - 1\right)$$

unde $I_s \approx 10^{-6}\mathrm{A}$, $u_T \approx 25\mathrm{mV}$

Notes

Dioda semiconductoare

Modele liniare pe porţiuni (de exemplu - modelul cu parametrii u_p, G_d, G_i) definite prin cod

$$i(u) = \begin{cases} G_i u & \text{dacă} \quad u \leq u_p \\ G_d(u - u_p) + G_i u_p & \text{dacă} \quad u > u_p \end{cases}$$

Notes

Dioda semiconductoare

Modele liniare pe porţiuni - definite prin tabele de valori

Exemplu - modelul lpp cu parametrii u_p, G_d, G_i

u	0	u_p	$2u_p$
i	0	$G_i u_p$	$(G_i + G_d)u_p$

Notes

Newton

Iteraţii Newton:

- Ecuaţie: $f(x) = 0$

$$x^{(m+1)} = x^{(m)} - f(x^{(m)})/f'(x^{(m)})$$

sau

$$z = f(x^{(m)})/f'(x^{(m)}) \quad (1)$$
$$x^{(m+1)} = x^{(m)} + z \quad (2)$$

- Sistem: $\mathbf{F}(\mathbf{x}) = \mathbf{0}$

$$\mathbf{x}^{(m+1)} = \mathbf{x}^{(m)} - (\mathbf{F}'(\mathbf{x}^{(m)}))^{-1}\mathbf{F}(\mathbf{x}^{(m)})$$

sau

$$\mathbf{F}'(\mathbf{x}^{(m)})\mathbf{z} = \mathbf{F}(\mathbf{x}^{(m)}) \quad (3)$$
$$\mathbf{x}^{(m+1)} = \mathbf{x}^{(m)} + \mathbf{z} \quad (4)$$

Notes

Newton

În cazul circuitelor rezistive neliniare $\mathbf{F}(\mathbf{V}) = \mathbf{0}$ unde

$$\mathbf{F}(\mathbf{V}) = \mathbf{AG}(\mathbf{A}^T\mathbf{V})$$

Iteraţii Newton:

$$\mathbf{F}'(\mathbf{V}^{(m)})\mathbf{z} = -\mathbf{F}(\mathbf{V}^{(m)}) \quad (5)$$
$$\mathbf{V}^{(m+1)} = \mathbf{V}^{(m)} + \mathbf{z} \quad (6)$$

$$\mathbf{F}'(\mathbf{V}) = \mathbf{AG}'(\mathbf{A}^T\mathbf{V})\mathbf{A}^T$$

Notes

Newton

În cazul circuitelor rezistive neliniare $\mathbf{F}(\mathbf{V}) = \mathbf{0}$ unde

$$\mathbf{F}(\mathbf{V}) = \mathbf{A}\mathbf{G}(\mathbf{A}^T\mathbf{V})$$

Iteraţii Newton:

$$\mathbf{F}'(\mathbf{V}^{(m)})\mathbf{z} = -\mathbf{F}(\mathbf{V}^{(m)}) \quad (5)$$
$$\mathbf{V}^{(m+1)} = \mathbf{V}^{(m)} + \mathbf{z} \quad (6)$$

$$\mathbf{F}'(\mathbf{V}) = \mathbf{A}\mathbf{G}'(\mathbf{A}^T\mathbf{V})\mathbf{A}^T$$

- Calculul Jacobianului necesită evaluarea conductanţelor dinamice!

Notes

Introducere
Metoda nodală clasică
Descrierea caracteristicilor neliniare
Algoritmi
Metoda Newton
Idei de implementare
Preprocesare
Procesare

Newton

În cazul circuitelor rezistive neliniare $\mathbf{F}(\mathbf{V}) = \mathbf{0}$ unde

$$\mathbf{F}(\mathbf{V}) = \mathbf{A}\mathbf{G}(\mathbf{A}^T\mathbf{V})$$

Iteraţii Newton:

$$\mathbf{F}'(\mathbf{V}^{(m)})\mathbf{z} = -\mathbf{F}(\mathbf{V}^{(m)}) \quad (5)$$
$$\mathbf{V}^{(m+1)} = \mathbf{V}^{(m)} + \mathbf{z} \quad (6)$$

$$\mathbf{F}'(\mathbf{V}) = \mathbf{A}\mathbf{G}'(\mathbf{A}^T\mathbf{V})\mathbf{A}^T$$

- Calculul Jacobianului necesită evaluarea conductanţelor dinamice!
- Evaluarea conductanţelor dinamice depinde de modul în care au fost definite caracteristicile neliniare.

Notes

Semnificaţia iteraţiilor Newton

Iteraţii Newton:

$$\mathbf{F}'(\mathbf{V}^{(m)})\mathbf{z} = -\mathbf{F}(\mathbf{V}^{(m)}) \quad (7)$$

$$\mathbf{V}^{(m+1)} = \mathbf{V}^{(m)} + \mathbf{z} \quad (8)$$

$$\mathbf{F}(\mathbf{V}) = \mathbf{AG}(\mathbf{A}^T\mathbf{V})$$

$$\mathbf{F}'(\mathbf{V}) = \mathbf{AG}'(\mathbf{A}^T\mathbf{V})\mathbf{A}^T$$

$$\mathbf{AG}'(\mathbf{A}^T\mathbf{V}^{(m)})\mathbf{A}^T\mathbf{z} = -\mathbf{AG}(\mathbf{A}^T\mathbf{V}^{(m)}) \quad (9)$$

Liniare (SRC)

$$\mathbf{AGA}^T\mathbf{V} = -\mathbf{Aj}$$

Notes

Introducere
Metoda nodală clasică
Descrierea caracteristicilor neliniare
Algoritmi
Metoda Newton
Idei de implementare
Preprocesare
Procesare

Semnificaţia iteraţiilor Newton

Iteraţii Newton:

$$\mathbf{F}'(\mathbf{V}^{(m)})\mathbf{z} = -\mathbf{F}(\mathbf{V}^{(m)}) \quad (7)$$

$$\mathbf{V}^{(m+1)} = \mathbf{V}^{(m)} + \mathbf{z} \quad (8)$$

$$\mathbf{F}(\mathbf{V}) = \mathbf{AG}(\mathbf{A}^T\mathbf{V})$$

$$\mathbf{F}'(\mathbf{V}) = \mathbf{AG}'(\mathbf{A}^T\mathbf{V})\mathbf{A}^T$$

$$\mathbf{AG}'(\mathbf{A}^T\mathbf{V}^{(m)})\mathbf{A}^T\mathbf{z} = -\mathbf{AG}(\mathbf{A}^T\mathbf{V}^{(m)}) \quad (9)$$

Liniare (SRC)

$$\mathbf{AGA}^T\mathbf{V} = -\mathbf{Aj}$$

Semnificaţia relaţiei (9):

Notes

Semnificaţia iteraţiilor Newton

Iteraţii Newton:

$$\mathbf{F}'(\mathbf{V}^{(m)})\mathbf{z} = -\mathbf{F}(\mathbf{V}^{(m)}) \quad (7)$$

$$\mathbf{V}^{(m+1)} = \mathbf{V}^{(m)} + \mathbf{z} \quad (8)$$

$$\mathbf{F}(\mathbf{V}) = \mathbf{AG}(\mathbf{A}^T\mathbf{V})$$

$$\mathbf{F}'(\mathbf{V}) = \mathbf{AG}'(\mathbf{A}^T\mathbf{V})\mathbf{A}^T$$

$$\mathbf{AG}'(\mathbf{A}^T\mathbf{V}^{(m)})\mathbf{A}^T\mathbf{z} = -\mathbf{AG}(\mathbf{A}^T\mathbf{V}^{(m)}) \quad (9)$$

Liniare (SRC)

$$\mathbf{AGA}^T\mathbf{V} = -\mathbf{Aj}$$

Semnificaţia relaţiei (9):
La fiecare iteraţie se rezolvă un circuit liniar, poteţialele lui reprezintă corecţiile în iteraţiile Newton

Notes

Introducere | Metoda Newton
Metoda nodală clasică | Idei de implementare
Descrierea caracteristicilor neliniare | Preprocesare
Algoritmi | Procesare

Semnificaţia iteraţiilor Newton

Iteraţii Newton:

$$\mathbf{F}'(\mathbf{V}^{(m)})\mathbf{z} = -\mathbf{F}(\mathbf{V}^{(m)}) \quad (7)$$

$$\mathbf{V}^{(m+1)} = \mathbf{V}^{(m)} + \mathbf{z} \quad (8)$$

$$\mathbf{F}(\mathbf{V}) = \mathbf{AG}(\mathbf{A}^T\mathbf{V})$$

$$\mathbf{F}'(\mathbf{V}) = \mathbf{AG}'(\mathbf{A}^T\mathbf{V})\mathbf{A}^T$$

$$\mathbf{AG}'(\mathbf{A}^T\mathbf{V}^{(m)})\mathbf{A}^T\mathbf{z} = -\mathbf{AG}(\mathbf{A}^T\mathbf{V}^{(m)}) \quad (9)$$

Liniare (SRC)

$$\mathbf{AGA}^T\mathbf{V} = -\mathbf{Aj}$$

Semnificaţia relaţiei (9):
La fiecare iteraţie se rezolvă un circuit liniar, poteţialele lui reprezintă corecţiile în iteraţiile Newton

Circuit incremental

Notes

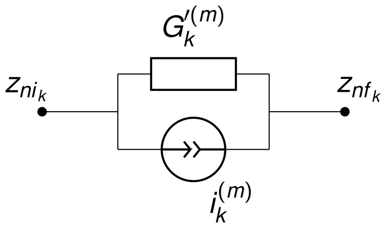
Circuite incrementale/liniarizate

Neliniar

$$\mathbf{AG}'(\mathbf{A}^T\mathbf{V}^{(m)})\mathbf{A}^T\mathbf{z} = -\mathbf{AG}(\mathbf{A}^T\mathbf{V}^{(m)})$$

Liniar

$$\mathbf{AGA}^T\mathbf{V} = -\mathbf{Aj}$$



$$z_{ni_k} = V_{ni_k}^{(m+1)} - V_{ni_k}^{(m)} \qquad z_{nf_k} = V_{nf_k}^{(m+1)} - V_{nf_k}^{(m)}$$

Notes

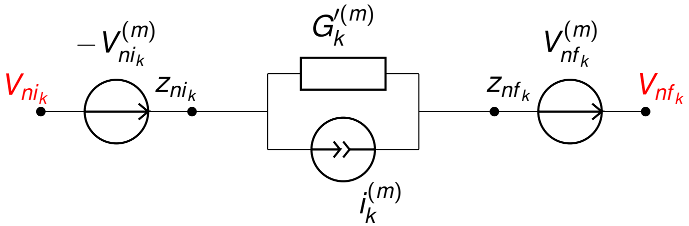
Circuite incrementale/liniarizate

Neliniar

$$\mathbf{AG}'(\mathbf{A}^T\mathbf{V}^{(m)})\mathbf{A}^T\mathbf{z} = -\mathbf{AG}(\mathbf{A}^T\mathbf{V}^{(m)})$$

Liniar

$$\mathbf{AGA}^T\mathbf{V} = -\mathbf{Aj}$$



$$z_{ni_k} = V_{ni_k}^{(m+1)} - V_{ni_k}^{(m)} \qquad z_{nf_k} = V_{nf_k}^{(m+1)} - V_{nf_k}^{(m)}$$

Notes

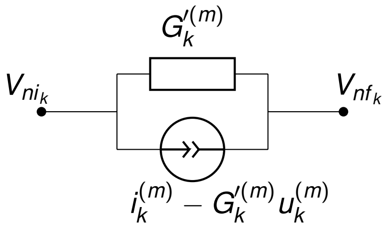
Circuite incrementale/liniarizate

Neliniar

$$\mathbf{AG}'(\mathbf{A}^T\mathbf{V}^{(m)})\mathbf{A}^T\mathbf{z} = -\mathbf{AG}(\mathbf{A}^T\mathbf{V}^{(m)})$$

Liniar

$$\mathbf{AGA}^T\mathbf{V} = -\mathbf{Aj}$$



Circuit liniarizat →
La fiecare iteraţie se rezolvă un circuit liniar, potenţialele lui reprezintă soluţiile noi în iteraţiile Newton

Notes

Algoritm - bazat pe asamblare de circuite

Ideea (nr. 1):
Se rezolvă o succesiune de circuite rezistive liniare (liniarizate).

it = 0
iniţializează soluţia **V**
repetă
    it = it + 1
    înlocuieşte elementele neliniare cu schemele lor *liniarizate*
    rezolvă circuitul rezistiv liniar şi calculează **Vn**
    actualizează soluţia **V** = **Vn**
    dacă it == itmax scrie mesaj de eroare
cât timp norma(**V** − **Vnou**) > toleranţa impusă şi it < itmax

Notes

Algoritm - bazat pe rezolvare de circuite

Ideea (nr. 2):
Se rezolvă o succesiune de circuite rezistive liniare (incrementale).

it = 0
iniţializează soluţia **V**
repetă
    it = it + 1
    înlocuieşte elementele neliniare cu schemele lor *incrementale*
    rezolvă circuitul rezistiv liniar şi calculează corecţiile **z**
    actualizează soluţia $\mathbf{V} = \mathbf{V} + \mathbf{z}$
    dacă it == itmax scrie mesaj de eroare
cât timp norma(**z**) > toleranţa impusă şi it < itmax

Notes

Algoritm - bazat pe operaţii cu matrice

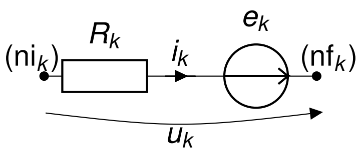
Ideea (nr. 3):
Se rezolvă o succesiune de sisteme algebricce liniare.

it = 0
asamblează matricea **A**
iniţializează soluţia **V**
repetă
    it = it + 1
    calculează conductanţele dinamice şi asamblează $\mathbf{G}'$
    rezolvă sistemul liniar $\mathbf{A}\mathbf{G}'\mathbf{A}^T\mathbf{z} = -\mathbf{A}\mathbf{i}$ şi calculează corecţiile **z**
    actualizează soluţia $\mathbf{V} = \mathbf{V} + \mathbf{z}$
    dacă it == itmax scrie mesaj de eroare
cât timp norma(**z**) > toleranţa impusă şi it < itmax

Notes

Cel mai simplu algoritm - pe ce ne bazăm

Primul algoritm scris pentru circuite rezistive liniare (crl) - laturi SRT



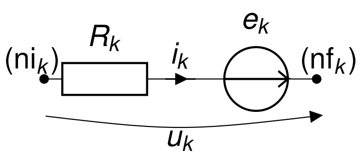

```
; declaratii date - varianta A
întreg N                    ; număr de noduri
întreg L                    ; număr de laturi
tablou întreg ni[L]         ; noduri iniţiale ale laturilor
tablou întreg nf[L]         ; noduri finale ale laturilor
tablou real R[L]            ; rezistenţe
tablou real e[L]            ; tensiuni electromotoare
```


Notes

Introducere | Metoda Newton
Metoda nodală clasică | Idei de implementare
Descrierea caracteristicilor neliniare | Preprocesare
Algoritmi | Procesare

Cel mai simplu algoritm - pe ce ne bazăm

Primul algoritm scris pentru circuite rezistive liniare (crl) - laturi SRT




```
; declaraţii date - varianta B
înregistrare circuit
        întreg N                    ; număr de noduri
        întreg L                    ; număr de laturi
        tablou întreg ni[L]         ; noduri iniţiale ale laturilor
        tablou întreg nf[L]         ; noduri finale ale laturilor
        tablou real R[L]            ; rezistenţe
        tablou real e[L]            ; tensiuni electromotoare
•
```


Notes

Cel mai simplu algoritm - pe ce ne bazăm

Să pp că avem la dispoziţie o procedură:

```
procedură nodal_crl(circuit,v)
; rezolvă un circuit rezistiv liniar cu metoda nodală
; date de intrare: structura circuit
; ieşire: valorile potenţialelor v în noduri, ultimul nod este de referinţă
...
retur
```

Obs: procedura cuprinde atât asamblarea sistemului de ecuaţii cât şi rezolvarea lui.

Notes

Cel mai simplu algoritm - ce e nou

- Admitem acum în plus, laturi rezistive neliniare, controlate în tensiune;

Vom presupune că există câte o procedură care poate, pentru orice latură neliniară, să întoarcă

- curentul prin latură pentru o tensiune dată ($i_k = g_k(u_k)$);
  Dacă curbele neliniare sunt date tabelar - aceasta presupune o interpolare).
- conductanţa dinamică a laturii, pentru o tensiune dată ($G'_k = g'_k(u_k)$).
  Dacă curbele neliniare sunt date tabelar - aceasta presupune o derivare numerică).

Notes

Cel mai simplu algoritm - etapa de preprocesare

```
funcţie citire_date ()
; declaraţii
...
citeşte circuit.N, circuit.L
pentru k = 1,circuit.L
        citeşte circuit.ni_k, circuit.nf_k
        citeşte circuit.tip_k      ; tipul poate fi "R" sau "n"
        dacă circuit.tip_k = "R"
                citeşte circuit.e_k, circuit.R_k
        •
citeşte tol                        ; toleranţă pentru procedura Newton
citeşte itmax                      ; numărul maxim de iteraţii admis
•
întoarce circuit
```


Notes

Introducere
Metoda nodală clasică
Descrierea caracteristicilor neliniare
Algoritmi
Metoda Newton
Idei de implementare
Preprocesare
Procesare

Cel mai simplu algoritm - etapa de preprocesare

funcţie citire_date ()
; declaraţii
...
citeşte circuit.N, circuit.L
pentru k = 1,circuit.L
citeşte circuit.ni_k, circuit.nf_k
citeşte circuit.tip_k ; tipul poate fi "R" sau "n"
dacă circuit.tip_k = "R"
citeşte circuit.e_k, circuit.R_k
•
citeşte tol ; toleranţă pentru procedura Newton
citeşte itmax ; numărul maxim de iteraţii admis
•
întoarce circuit

Dar partea neliniară?

Notes

Cel mai simplu algoritm - etapa de preprocesare

Variante - pentru partea neliniară:

```
funcţie g(u)
Is = 1e-12
Vt = 0.0278
întoarce Is*(exp(u/Vt)-1)
```

```
funcţie g(u)
nd = 3 ; numărul de puncte de discontinuitate
uval = .....
ival = ....
m = cauta(uval, ival, u)
întoarce ival(m) + (ival(m+1) - ival(m))/(uval(m+1)-uval(m))*(u - uval(m))
```

```
funcţie gder(u)
Is = 1e-12
Vt = 0.0278
întoarce Is*exp(u/Vt)/Vt
```

```
funcţie gder(u)
nd = 3 ; numărul de puncte de discontinuitate
uval = .....
ival = ....
m = cauta(uval, ival, u)
întoarce (ival(m+1) - ival(m))/(uval(m+1)-uval(m))
```

Is, Vt, nd, uval, ival - pot fi citite în etapa de preprocesare (şi pot fi diferite pentru diferitele elemente neliniare).

Notes

Algoritm - v3

```
procedură solve_crnl_v3(circuit,tol,itmax,V)
circuit - structură - parametru de intrare
tol, itmax - parametri de intrare, specifici procedurii Newton
V - vector - parametru de ieşire
....
asamblează matricea incidenţelor laturi noduri
A = 0; matrice de dimensiune N x L
pentru k = 1:L
        i = circuit.ni(k);
        j = circuit.nf(k);
        A(i,k) = 1;
        A(j,k) = -1;
•
A(N,:) = []; elimină ultima linie
V = 0; vector de dimensiune N-1
err = 0.01;
cor = 1;
itk = 0;
cât timp abs(norm(cor)) > err şi itk < itmax
        u = A^T * V
        solve_lin(Fder(u), -F(u), cor)        ; rezolvă sistemul liniar
                                              ; şi calculează corecţia
        itk = itk + 1;
        V = V + cor;
•
retur
```


Notes

Algoritm - v3

```
funcţie F(u)
....
G = 0 ; vector coloană de dimensiune L
pentru k = 1:L
        dacă circuit.tip(k) == "I"
                G(k) = (u(k) + circuit.e(k))/circuit.R(k)
        altfel
                G(k) = g(u(k))
        •
•
întoarce A * G
```

```
funcţie Fder(u)
....
Gd = 0 ; vector coloană de dimensiune L
pentru k = 1:L
        dacă circuit.tip(k) == "l"
                Gd(k) = 1/circuit.R(k)
        altfel
                Gd(k) = gder(u(k))
        •
•
Gder = diag(Gd)
întoarce A * Gder * A^T
```

Aici structura circuit şi matricea **A** sunt pp. globale, altfel trebuie date ca parametri.

Notes

Introducere | Metoda Newton
Metoda nodală clasică | Idei de implementare
Descrierea caracteristicilor neliniare | Preprocesare
Algoritmi | Procesare

Algoritm - v2

```
procedură solve_crnl_v2(circuit,tol,itmax,V)
circuit - structură - parametru de intrare
tol, itmax - parametri de intrare, specifici procedurii Newton
V - vector - parametru de ieşire
....
iniţializare
V = 0 ; vector de dimensiune N
err = 1
itk = 0
cât timp err > tol şi itk < itmax
        kit = kit + 1
        pentru k = 1:L
                dacă circuit.tip(k) == "n"
                        tens = V(circuit.ni(k)) - V(circuit.nf(k))
                        cond_din = gder(tens)
                        crt = g(tens)
                        circuit.R(k) = 1/cond_din
                        circuit.e(k) = circuit.R(k)*crt - tens
                •
        •
        nodal_crl(circuit,Vn)
        err = norma(Vn − V)
        V = Vn
•
retur
```

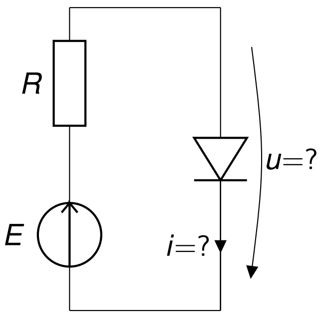

Notes

Algoritm - v1

```
procedură solve_crnl_v1(circuit,tol,itmax,V)
circuit - structură - parametru de intrare
tol, itmax - parametri de intrare, specifici procedurii Newton
V - vector - parametru de ieşire
....
iniţializare
V = 0 ; vector de dimensiune N
err = 1
itk = 0
cât timp err > tol şi itk < itmax
        kit = kit + 1
        pentru k = 1:L
                dacă circuit.tip(k) == "n"
                        tens = V(circuit.ni(k)) - V(circuit.nf(k))
                        cond_din = gder(tens)
                        crt = g(tens)
                        circuit.R(k) = 1/cond_din
                        circuit.e(k) = circuit.R(k)*crt
                •
        •
        nodal_crl(circuit,z)
        err = norma(z)
        V = V + z
•
retur
```

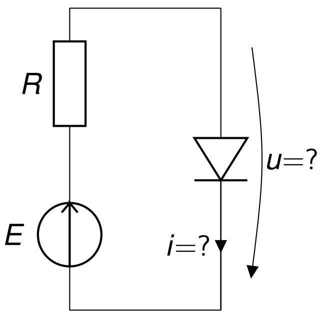

Notes

Exemplul 1 - rezultate



Notes

Exemplul 1 - rezultate

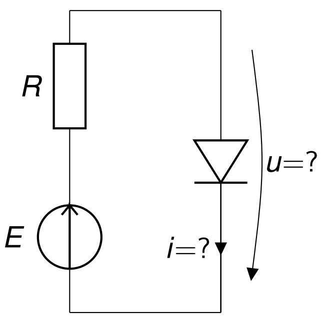


$$i = g(u)$$
$$i = \frac{E-u}{R}$$

Notes

Introducere
Metoda nodală clasică
Descrierea caracteristicilor neliniare
Algoritmi
Metoda Newton
Idei de implementare
Preprocesare
Procesare

Exemplul 1 - rezultate

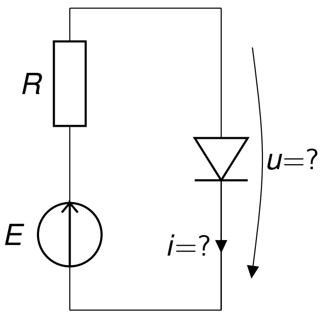


$$i = g(u)$$
$$i = \frac{E-u}{R}$$

$$E = 1.25\mathrm{V}, \; R = 1.25\mathrm{m}\Omega$$

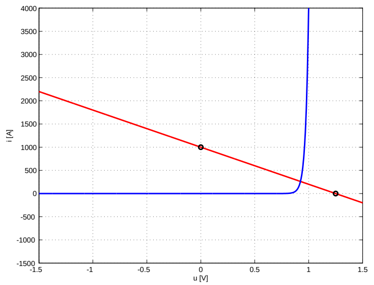
Notes

Exemplul 1 - rezultate



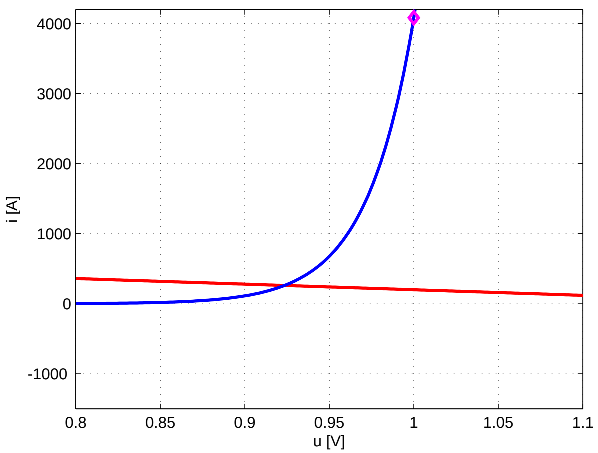
$$i = g(u)$$
$$i = \frac{E-u}{R}$$

$E = 1.25\mathrm{V},\ R = 1.25\mathrm{m}\Omega$



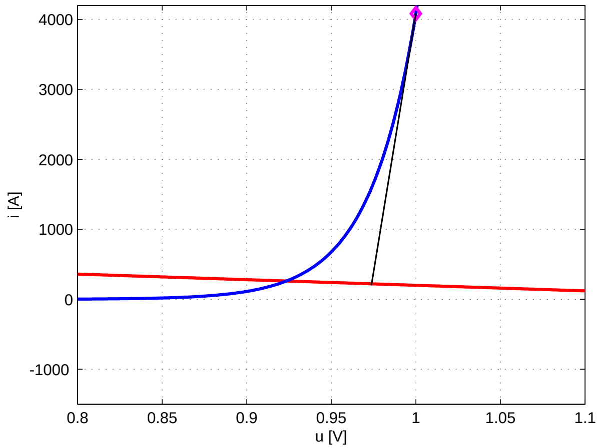
Notes

Exemplul 1 - rezultate



Notes

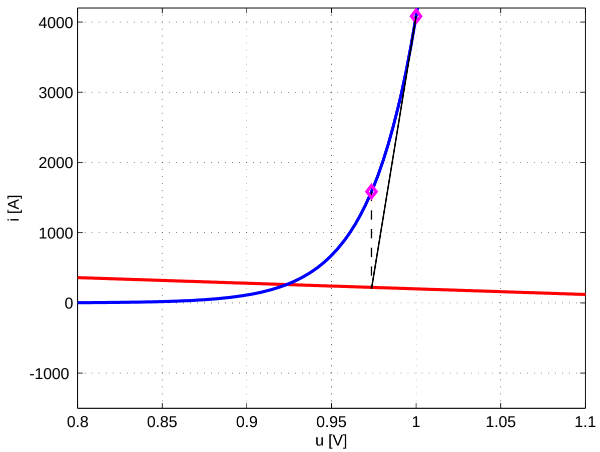
Exemplul 1 - rezultate



Notes

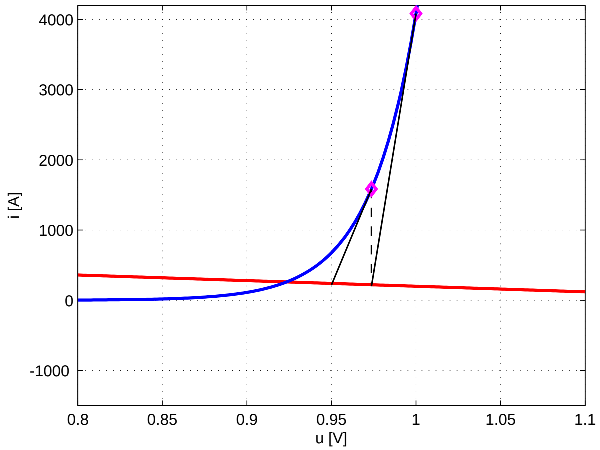
Introducere
Metoda nodală clasică
Descrierea caracteristicilor neliniare
Algoritmi
Metoda Newton
Idei de implementare
Preprocesare
Procesare

Exemplul 1 - rezultate



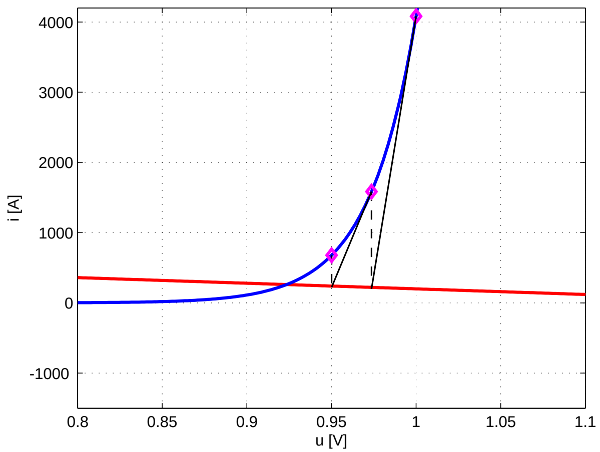
Notes

Exemplul 1 - rezultate



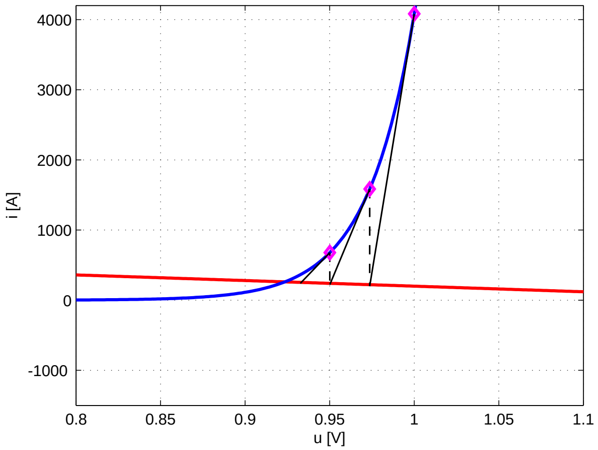
Notes

Exemplul 1 - rezultate



Notes

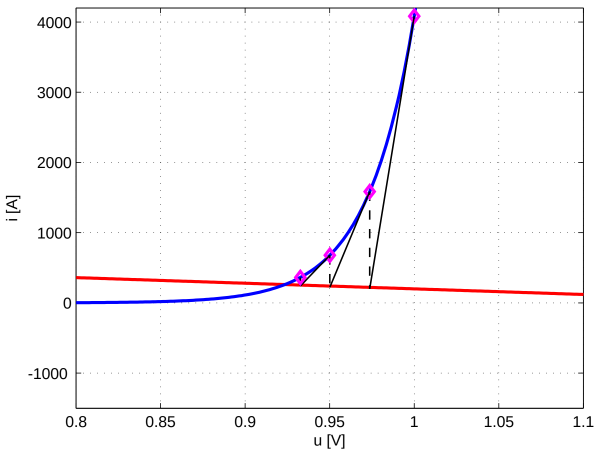
Exemplul 1 - rezultate



Notes

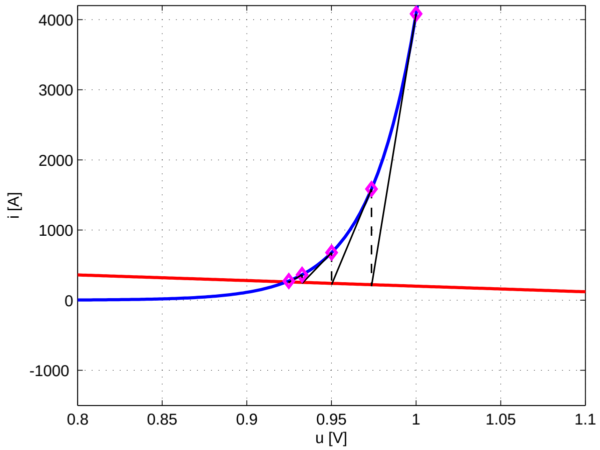
Introducere | Metoda Newton
Metoda nodală clasică | Idei de implementare
Descrierea caracteristicilor neliniare | Preprocesare
Algoritmi | Procesare

Exemplul 1 - rezultate



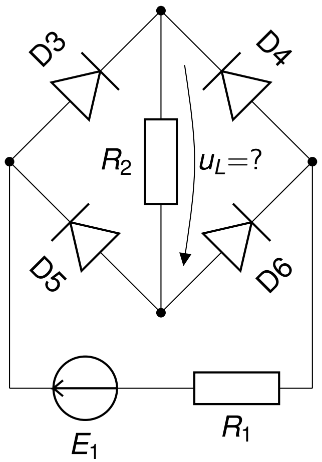
Notes

Exemplul 1 - rezultate



Notes

Exemplul 4 - rezultate

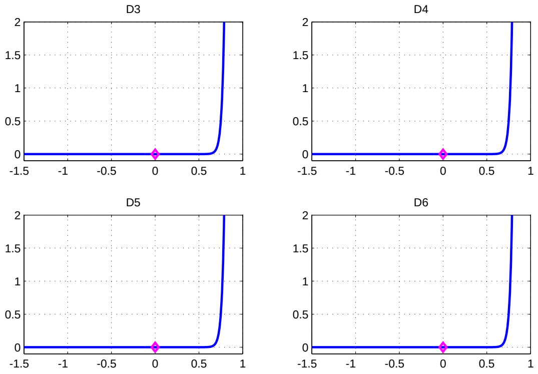


Notes

Exemplul 4 - rezultate

$E_1 = 2V$, $R_1 = 1\Omega$, $R_2 = 2\Omega$, 13 iteraţii pentru tol = 0.01

Numai iniţializarea şi ultimele patru sunt ilustrate.



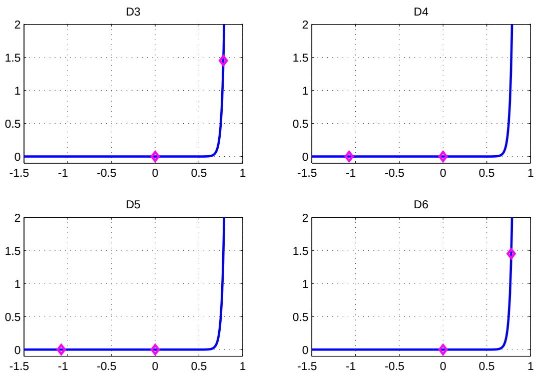
Notes

Introducere | Metoda Newton
Metoda nodală clasică | Idei de implementare
Descrierea caracteristicilor neliniare | Preprocesare
Algoritmi | Procesare

Exemplul 4 - rezultate

$E_1 = 2V$, $R_1 = 1\Omega$, $R_2 = 2\Omega$, 13 iteraţii pentru tol = 0.01

Numai iniţializarea şi ultimele patru sunt ilustrate.



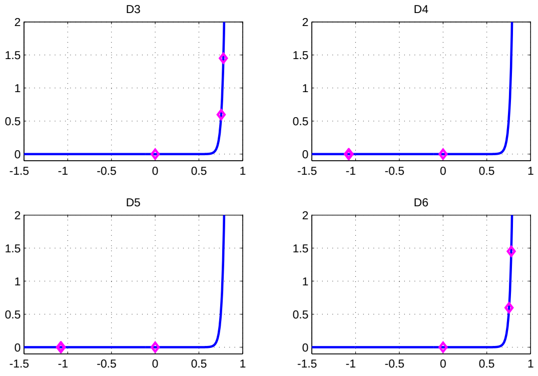
42/53
Gabriela Ciuprina | Analiza circuitelor electrice rezistive neliniare

Notes

Exemplul 4 - rezultate

$E_1 = 2V$, $R_1 = 1\Omega$, $R_2 = 2\Omega$, 13 iteraţii pentru tol = 0.01

Numai iniţializarea şi ultimele patru sunt ilustrate.

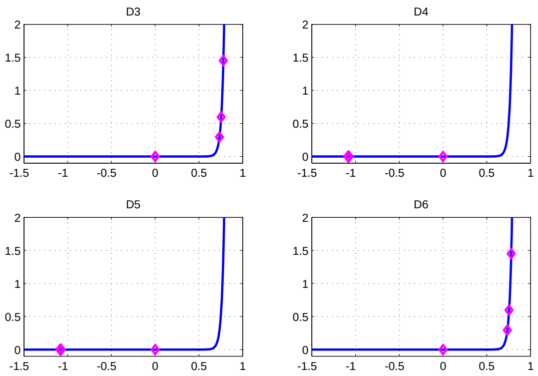


Notes

Exemplul 4 - rezultate

$E_1 = 2V$, $R_1 = 1\Omega$, $R_2 = 2\Omega$, 13 iteraţii pentru tol = 0.01

Numai iniţializarea şi ultimele patru sunt ilustrate.

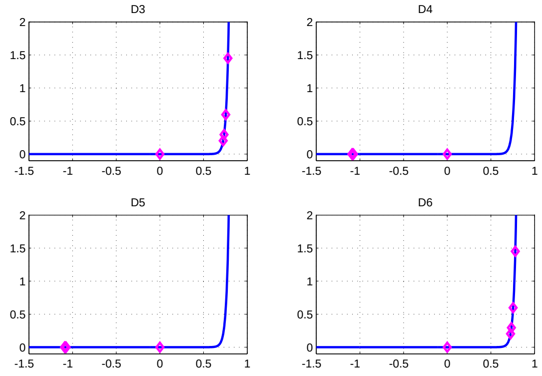


Notes

Exemplul 4 - rezultate

$E_1 = 2V$, $R_1 = 1\Omega$, $R_2 = 2\Omega$, 13 iteraţii pentru tol = 0.01

Numai iniţializarea şi ultimele patru sunt ilustrate.



Notes

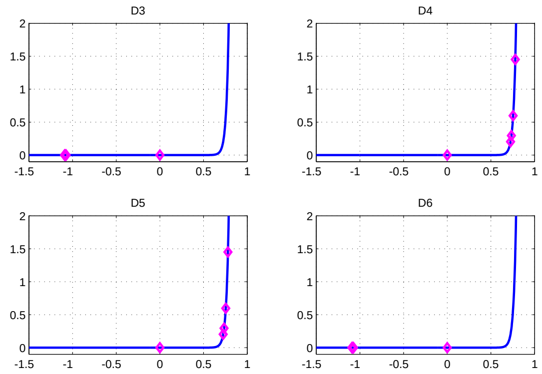
Introducere
Metoda nodală clasică
Descrierea caracteristicilor neliniare
Algoritmi

Metoda Newton
Idei de implementare
Preprocesare
Procesare

Exemplul 4 - rezultate

$E_1 = -2V$, $R_1 = 1\Omega$, $R_2 = 2\Omega$, 13 iteraţii pentru tol = 0.01

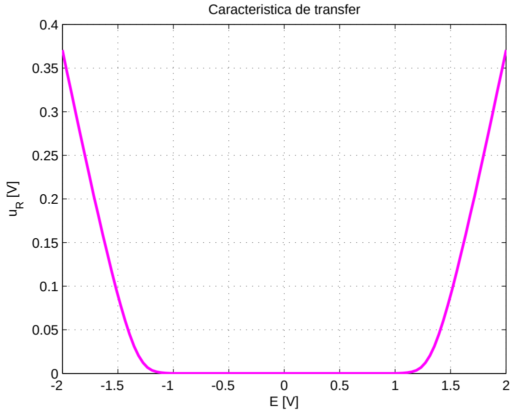
Numai iniţializarea şi ultimele patru sunt ilustrate.



Notes

Exemplul 4 - rezultate

$E_1 \in [-2, 2]\mathrm{V}$, $R_1 = 1\Omega$, $R_2 = 2\Omega$, $u_{R2} = ?$

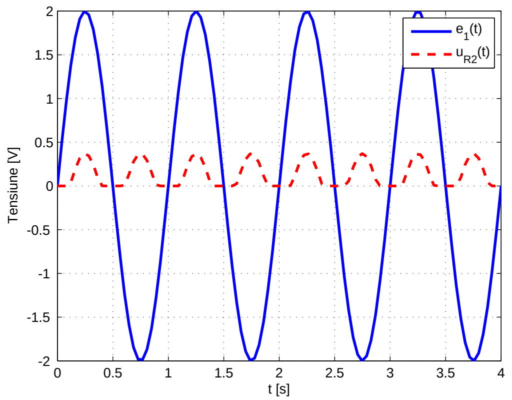


Notes

Exemplul 4 - rezultate

Sursa variabilă în timp? *Timpul are un caracter convențional. (Sistemul este algebric!)*

$e_1(t) = 2\sin(2\pi t)\mathrm{V}$, $R_1 = 1\Omega$, $R_2 = 2\Omega$, $u_{R2}(t) = ?$



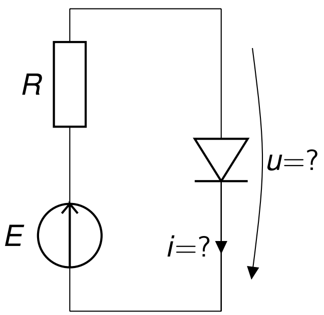
Notes

Concluzii

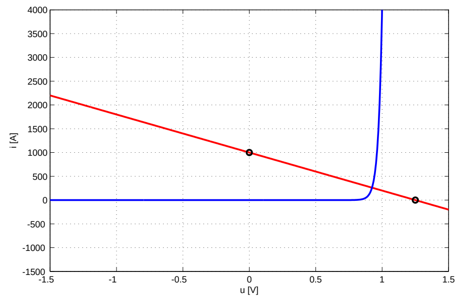
- Analiza circuitelor rezistive neliniare se reduce la o succesiune de rezolvări de sisteme algebrice liniare (care pot fi privite ca rezolvări de circuite rezistive liniare - incrementale sau liniarizate).
- Convergenţa procedurii depinde de iniţializare.
- Numărul de iteraţii depinde de iniţializare şi de eroarea impusă soluţiei.

Notes

Cazul caracteristicilor lpp

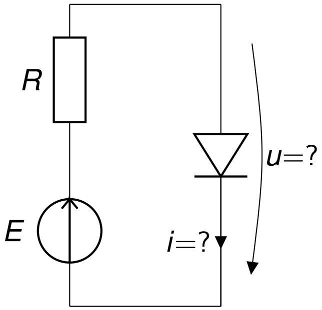


Aproximaţia lpp a caracteristicii diodei semiconductoare.

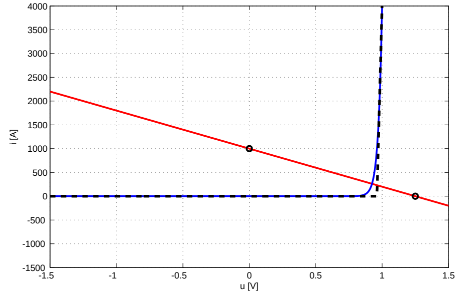


Notes

Cazul caracteristicilor lpp



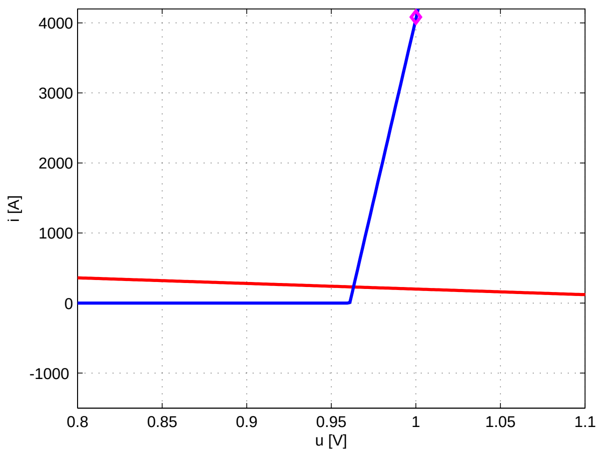
Aproximaţia lpp a caracteristicii diodei semiconductoare.



Notes

Cazul caracteristicilor lpp

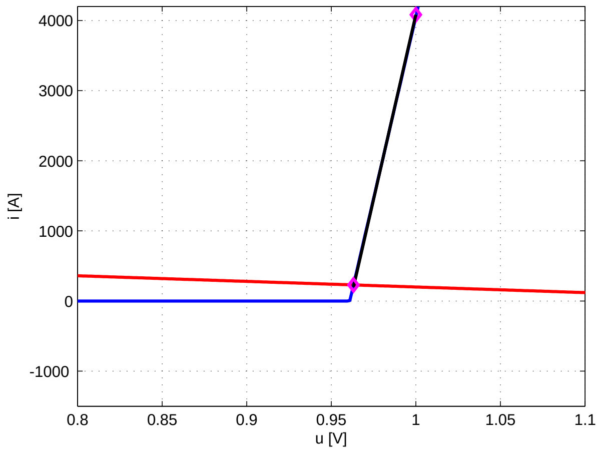
Iteraţii Newton - iniţializarea.



Notes

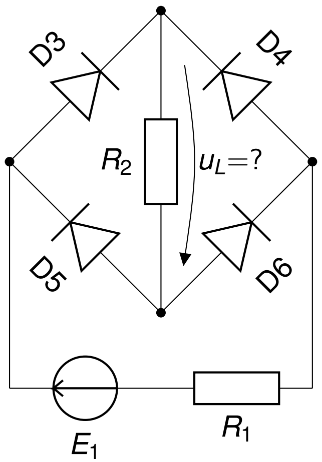
Cazul caracteristicilor lpp

Iteraţii Newton - iteraţia 1.



Notes

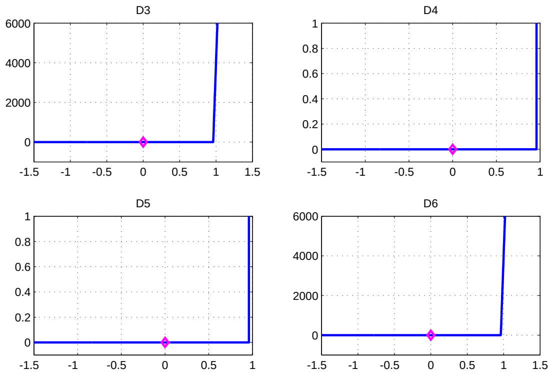
Cazul caracteristicilor lpp



Notes

Cazul caracteristicilor lpp

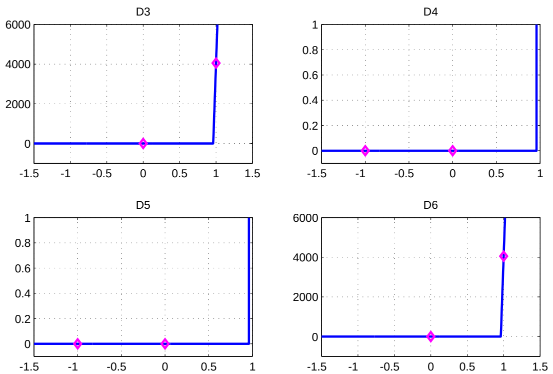
Iteraţii Newton - iniţializarea.



Notes

Cazul caracteristicilor lpp

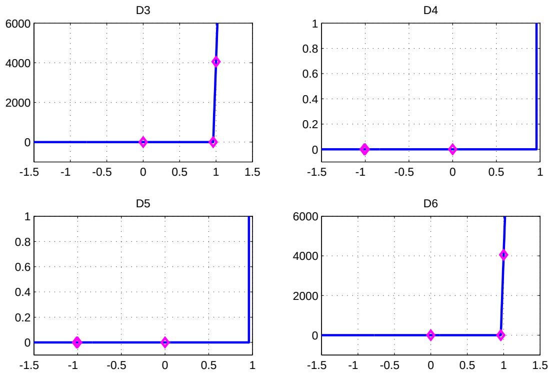
Iteraţii Newton - iteraţia 1.



Notes

Cazul caracteristicilor lpp

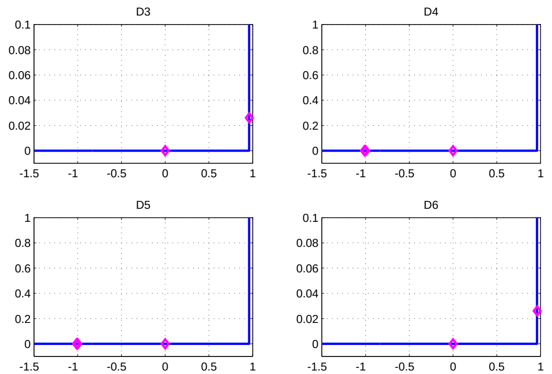
Iteraţii Newton - iteraţia 2.



Notes

Cazul caracteristicilor lpp

Iteraţii Newton - iteraţia 2 - *zoom in*.



Notes

Cazul caracteristicilor lpp

- Eroarea impusă nu influenţează prea mult numărul de iteraţii deoarece după determinarea corectă a segmentului în care se află PSF, eroarea impusă este satisfăcută la următoarea iteraţie.
- Dacă iniţializarea corespunde combinaţiei corecte de segmente, atunci se va face exact o singură iteraţie.
- Numărul maxim de iteraţii este egal cu numărul maxim de combinaţii de segmente.
- Există o variantă a metodei (cunoscută sub numele de metoda Katzenelson) în care la fiecare iteraţie se modifică un singur segment, cel corespunzător variaţiei maxime. Avantaj - convergenţa garantată.

Notes

Referinţe

- [Ioan98] D. Ioan et al., *Metode numerice în ingineria electrică*, Ed. Matrix Rom, Bucuresti, 1998. (Capitolul 17)
- [Chua75] Leon Chua, Pen-Min Lin, *Computer-Aided Analysis of Electronic Circuits*, Prentice-Hall,1975. (Capitolele 5 si 7)

Notes