

# Rezolvarea sistemelor de ecuații algebrice neliniare

Prof.dr.ing. Gabriela Ciuprina

Universitatea "Politehnica" București, Facultatea de Inginerie Electrică

Suport didactic pentru disciplina *Algoritmi Numerici*, 2017-2018

# Cuprins

- 1 Formularea problemei
  - Enunț
  - Dificultăți
- 2 Iterații simple
  - Problemă de punct fix
  - Convergență
  - Jacobian
- 3 Newton
  - Algoritm
  - Convergența
  - Efort de calcul
- 4 Metode care aproximează Jacobianul
- 5 Metode robuste de tip Newton
  - *Damped Newton*
  - *Trust region*
- 6 Metode de descompunere

## Formularea problemei

**Se dau**  $f_k : \mathbb{R}^n \rightarrow \mathbb{R}$ , continue,  $k = 1, \dots, n$ .

**Se cer**  $x_k$  pentru care

$$f_1(x_1, x_2, \dots, x_n) = 0$$

$$f_2(x_1, x_2, \dots, x_n) = 0$$

$$\vdots$$

$$f_n(x_1, x_2, \dots, x_n) = 0$$

## Formularea problemei

Se dă  $\mathbf{F} : \mathbb{R}^n \rightarrow \mathbb{R}^n$ , continuă.

Se cere  $\mathbf{x} \in \mathbb{R}^n$  pentru care

$$\mathbf{F}(\mathbf{x}) = \mathbf{0} \quad (1)$$

unde

$$\mathbf{F} = [f_1, f_2, \dots, f_n]^T \in \mathbb{R}^n$$

$$\mathbf{x} = [x_1, x_2, \dots, x_n]^T \in \mathbb{R}^n$$

Exemplu care conduce la un astfel de sistem: analiza circuitelor rezistive neliniare.

# Dificultăți

- Nu există o metodă simplă de a încadra soluția astfel încât să obținem o metodă garantat convergentă ca în cazul 1D.  
**Metodele de încadrare nu se pot generaliza.**
- Efortul de calcul crește rapid odată cu dimensiunea sistemului.

# Iterații simple

Formularea problemei ca o problemă de punct fix

$$\mathbf{F}(\mathbf{x}) = \mathbf{0} \quad \Leftrightarrow \quad \mathbf{x} = \mathbf{G}(\mathbf{x}) \quad (2)$$

unde

$$\mathbf{G}(\mathbf{x}) = \mathbf{x} + \mathbf{C}\mathbf{F}(\mathbf{x}) \quad (3)$$

și  $\mathbf{C} \in \mathbb{R}^{n \times n}$

Iterații de punct fix

$$\mathbf{x}^{(k+1)} = \mathbf{G}(\mathbf{x}^{(k)}) \quad (4)$$

## Iterații simple

O condiție suficientă de convergență este:

- În 1D  $|g'(x^*)| < 1$  unde  $x^*$  este soluția.
- În nD

$$\rho(\mathbf{G}'(\mathbf{x}^*)) < 1$$

unde  $\mathbf{G}'$  este matricea Jacobian.

și inițializarea este suficient de apropiată de soluție

Deoarece  $\rho(\mathbf{A}) \leq \|\mathbf{A}\| \Rightarrow$  condiție suficientă de convergență:

$$\|\mathbf{G}'(\mathbf{x}^*)\| < 1 \Leftrightarrow \|\mathbf{I} + \mathbf{CF}'(\mathbf{x})\| < 1$$

unde  $\mathbf{F}'$  este matricea Jacobian.

# Iterații simple

## Matricea Jacobian

$$\mathbf{F}'(\mathbf{x}) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \dots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \dots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \dots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} \stackrel{\text{not}}{=} \mathbf{J}_F(\mathbf{x}) \quad (5)$$



## Newton - ideea

Convergența este cu atât mai rapid convergentă cu cât  $\|I + \mathbf{C}\mathbf{F}'(\mathbf{x})\|$  este mai mică.

$\Rightarrow$

Viteza maximă de convergență pentru

$$\|I + \mathbf{C}\mathbf{F}'(\mathbf{x})\| = 0$$

$$\mathbf{C} = -(\mathbf{F}'(\mathbf{x}))^{-1} \quad (6)$$

Iterații Newton:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - (\mathbf{F}'(\mathbf{x}^{(k)}))^{-1}\mathbf{F}(\mathbf{x}^{(k)}) \quad (7)$$

Eșec dacă se întâlnește o matrice Jacobian singulară. 

# Algoritm

Nu se implementează formula

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - (\mathbf{F}'(\mathbf{x}^{(k)}))^{-1} \mathbf{F}(\mathbf{x}^{(k)}) \quad (8)$$

Dacă notăm  $\mathbf{z}$  corecția:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \mathbf{z} \quad (9)$$

atunci

$$\mathbf{z} = -(\mathbf{F}'(\mathbf{x}^{(k)}))^{-1} \mathbf{F}(\mathbf{x}^{(k)})$$

$$(\mathbf{F}'(\mathbf{x}^{(k)}))\mathbf{z} = -\mathbf{F}(\mathbf{x}^{(k)}) \quad (10)$$

La fiecare iterație neliniară

- 1 se calculează corecția prin rezolvarea sist. algebric linear (10);
- 2 se actualizează soluția cu (9).

# Convergența

## Pătratică

Funcția de iterație  $\mathbf{G}(\mathbf{x}) = \mathbf{x} - \mathbf{J}_F^{-1}(\mathbf{x})\mathbf{F}(\mathbf{x})$

$$\mathbf{J}_G(\mathbf{x}^*) = \mathbf{0}$$

Taylor:

$$\mathbf{G}(\mathbf{x}) = \mathbf{G}(\mathbf{x}^*) + (\mathbf{x} - \mathbf{x}^*)^T \mathbf{J}_G(\mathbf{x}^*) + (\mathbf{x} - \mathbf{x}^*)^T \mathbf{H}_G(\mathbf{x}^*)(\mathbf{x} - \mathbf{x}^*)/2$$

O demonstrație riguroasă găsiți [aici](#)

$$\|\mathbf{x}^{(k)} - \mathbf{x}^*\| \leq M \|(\mathbf{x}^{(k-1)} - \mathbf{x}^*)\|^2 \quad (11)$$

# Convergența

## Matricea Hessian

$$\mathbf{F}''(\mathbf{x}) = \begin{bmatrix} \frac{\partial^2 f_1}{\partial x_1^2} & \frac{\partial^2 f_1}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f_1}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f_2}{\partial x_2 \partial x_1} & \frac{\partial^2 f_2}{\partial x_2^2} & \cdots & \frac{\partial^2 f_2}{\partial x_2 \partial x_n} \\ \vdots & & & \\ \frac{\partial^2 f_n}{\partial x_n \partial x_1} & \frac{\partial^2 f_n}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f_n}{\partial x_n^2} \end{bmatrix} \stackrel{\text{not}}{=} \mathbf{H}_F(\mathbf{x}) \quad (12)$$

# Efort de calcul

La fiecare iterație:

- Evaluarea Jacobianului -  $n^2$  evaluări de funcții scalare dacă problema este densă (fiecare componentă a funcției depinde de toate componentele variabilei);
- Rezolvarea unui sistem de ecuații algebrice liniare -  $n^3$  dacă matricea este plină.

**Foarte costisitor!**

## Variante inspirate din metodele 1D

Scop: **reducerea efortului de calcul pe iterație.**

**Dar:** convergență nu va mai fi pătratică  $\Rightarrow$  compromis.

- Newton-Kantorovich (tangente paralele)

$$(\mathbf{F}'(\mathbf{x}^{(0)}))\mathbf{z} = -\mathbf{F}(\mathbf{x}^{(k)}) \quad (13)$$

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \mathbf{z} \quad (14)$$

Sistemul de rezolvat are întotdeauna aceeași matrice a coeficienților  $\Rightarrow$  este eficientă folosirea factorizării.

- Secante - derivatele parțiale din formula Jacobianului se calculează numeric.

$$\frac{\partial f_j}{\partial x_j} = \frac{f_j(x_1^{(k-1)}, x_2^{(k-1)}, \dots, x_j^{(k)}, \dots, x_n^{(k-1)}) - f_j(x_1^{(k-1)}, x_2^{(k-1)}, \dots, x_j^{(k-1)}, \dots, x_n^{(k-1)})}{x_j^{(k)} - x_j^{(k-1)}}$$

## Convergență locală/globală

### Metodă locală

O metoda iterativă este **locală** - dacă ea converge doar dacă inițializarea este suficient de aproape de soluție (*în interiorul razei de convergență.*)

- Metoda Newton și variantele ei sunt locale.
- Dacă inițializarea este foarte proastă, atunci metodele locale pot fi modificate pentru a îmbunătăți convergența lor globală [Martinez]. Exemple de astfel de metode:
  - Newton cu amortizare;
  - quasi-Newton (Broyden);
  - regiuni de încredere;
  - descompunere (separare/decuplare/segregare).

## Metoda Newton cu amortizare

Se calculează corecția  $\mathbf{z}$  la fiecare iterație, dar aproximația nouă se calculează ca

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \mathbf{z}$$

$\alpha_k$  este un parametru scalar care se alege a.î. aproximația  $\mathbf{x}^{(k+1)}$  să fie mai bună decât aproximația  $\mathbf{x}^{(k)}$ .



## Metoda Newton cu amortizare

Exemplu:

$\alpha_k$  se alege a.î.  $\|\mathbf{f}(\mathbf{x}^{(k)})\|_2$  să scadă suficient de mult la fiecare iterație.

sau

$\alpha_k$  se alege a.î.  $h(\alpha) = \|\mathbf{f}(\mathbf{x}^{(k)}) + \alpha\mathbf{z}\|_2$  să fie minimă

- Există o legătură între rezolvarea sistemelor neliniare și tehnicile de minimizare (optimizare).
- Când aproximările au devenit suficient de aproape de soluție, coeficientul de amortizare trebuie să devină 1.

## Metoda regiunii de încredere

Este o metodă mai complicată dar care face metoda de tip Newton să fie mai robustă.

Ideea:

- Se estimează raza unei "regiuni de încredere" în care aproximarea seriei Taylor pe care se bazează metoda Newton să fie suficient de precisă.
- Corecția soluției se ajustează în funcție de raza acestei regiuni de încredere.
- În apropierea soluției, regiunile de încredere sunt suficient de mari a.î sunt permise corecții totale Newton.

Și această metodă folosește **tehnici de optimizare**<sup>1</sup>

<sup>1</sup>Detalii ale acestor algoritmi vor fi prezentate în capitolul de optimizare  18/32

## Metoda regiunii de încredere

Exemple de algoritmi din această categorie:

- Levenberg - Marquardt;
- *double dogleg*;
- etc.

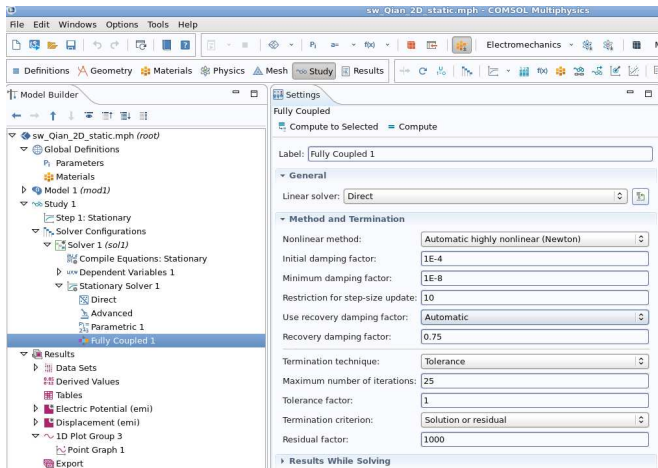
Astfel de algoritmi sunt folosiți de exemplu în:

- funcția matlab `fsolve`;
- COMSOL
- etc

Formularea problemei  
Iterații simple  
Newton  
Metode care aproximează Jacobianul  
Metode robuste de tip Newton  
Metode de descompunere

*Damped Newton*  
*Trust region*

# Fiți atenți la astfel de informații (capturi din COMSOL)



Formularea problemei  
Iterații simple  
Newton  
Metode care aproximează Jacobianul  
Metode robuste de tip Newton  
Metode de descompunere

*Damped Newton*  
*Trust region*

## Fiți atenți la astfel de informații (capturi din COMSOL)

**Method and Termination**

Nonlinear method: Automatic highly nonlinear (Newton)

Initial damping factor: Automatic (Newton)

Minimum damping factor: Constant (Newton)

Restriction for step-size update: Automatic highly nonlinear (Newton)

Use recovery damping factor: Double dogleg

Use recovery damping factor: Automatic

Termination technique: Tolerance

Maximum number of iterations: Tolerance

Tolerance factor: Iterations or tolerance

Termination criterion: 1

Termination criterion: Solution or residual

Residual factor: Solution

Residual factor: Residual

Results While Solving: Solution or residual

Formularea problemei  
 Iterații simple  
 Newton  
 Metode care aproximează Jacobianul  
 Metode robuste de tip Newton  
 Metode de descompunere

*Damped Newton*  
*Trust region*

# Fiți atenți la astfel de informații (capturi din COMSOL)

The screenshot shows the COMSOL Multiphysics interface for a 2D static problem. The main window displays a convergence plot for the electric potential (V) on a surface. The plot shows a sharp increase in potential near the boundary, with a color scale ranging from 0 to 30 V. The plot title is  $U(33)=30.358$  Surface: Electric potential (V).

The left sidebar shows the model tree with the following structure:

- sw\_Qian\_2D\_static.mph (root)
  - Global Definitions
    - Parameters
    - Materials
  - Model 1 (mod1)
    - Study 1
      - Step 1: Stationary
        - Solver Configurations
          - Solver 1 (sol1)
            - Compile Equations: Stationary
              - Dependent Variables 1
                - Stationary Solver 1
                  - Direct
                  - Advanced
                    - Parametric 1
                      - Fully Coupled 1
                        - Results
                          - Data Sets
                            - Derived Values
                            - Tables
                            - Electric Potential (emi)
                            - Displacement (emi)
                            - 1D Plot Group 3
                              - Point Graph 1
                              - Export
                              - Reports

The right sidebar shows the Settings window for the Fully Coupled 1 solver. The settings are:

- Label: Fully Coupled 1
- Compute to Selected: Compute
- General
  - Linear solver: Direct
- Method and Termination
  - Nonlinear method: Automatic
  - Initial damping factor:  $1E-4$
  - Minimum damping factor:  $1E-8$
  - Restriction for step-size update: 10
  - Use recovery damping factor: Automatic
  - Recovery damping factor: 0.75
  - Termination technique: Tolerance
  - Maximum number of iterations: 25
  - Tolerance factor: 1
  - Termination criterion: Solution o
  - Residual factor: 1000
- Results While Solving

The bottom right window shows the Messages and Progress window, displaying a continuation table for the parameter U = 30.3023. The table has the following columns: Iter, SolEst, ResEst, Damping, StepSize, #Res, #Jac, #Sol, LinErr, LinRes.

| Iter | SolEst  | ResEst | Damping   | StepSize | #Res | #Jac | #Sol | LinErr  | LinRes |
|------|---------|--------|-----------|----------|------|------|------|---------|--------|
| 1    | 0.0071  | 51     | 1.0000000 | 0.067    | 145  | 49   | 130  | 3.1e-05 | 2e-10  |
| 2    | 0.00018 | 0.99   | 1.0000000 | 0.0092   | 146  | 50   | 132  | 0.00029 | 3e-10  |

Continuation parameter U = 30.3023.

| Iter | SolEst | ResEst  | Damping   | StepSize | #Res | #Jac | #Sol | LinErr  | LinRes  |
|------|--------|---------|-----------|----------|------|------|------|---------|---------|
| 1    | 0.16   | 3.1e+02 | 1.0000000 | 0.26     | 150  | 51   | 135  | 3.4e-05 | 1.2e-10 |
| 2    | 0.47   | 4.2e+02 | 0.1000000 | 0.51     | 151  | 52   | 137  | 6e-06   | 4.4e-10 |
| 3    | 0.62   | 4e+02   | 0.1056976 | 0.68     | 152  | 53   | 139  | 3.9e-05 | 5.9e-10 |
| 4    | 1.3    | 3.9e+02 | 0.0314815 | 1.3      | 153  | 54   | 141  | 3.5e-05 | 1.2e-09 |
| 5    | 2.1    | 3.9e+02 | 0.081466  | 2.1      | 154  | 55   | 143  | 0.00059 | 2.4e-09 |
| 6    | 3.9    | 3.9e+02 | 0.0020440 | 3.9      | 155  | 56   | 145  | 0.0004  | 4.9e-09 |
| 7    | 8      | 3.9e+02 | 0.0005094 | 8        | 156  | 57   | 147  | 0.0026  | 9.9e-09 |
| 8    | 16     | 3.9e+02 | 0.0001267 | 16       | 157  | 58   | 149  | 0.0016  | 1.9e-08 |
| 9    | 34     | 3.9e+02 | 0.0001000 | 34       | 158  | 59   | 151  | 0.011   | 3.7e-08 |

Continuation parameter U = 30.907.

# Idea

Până acum componentele Jacobianului au fost evaluate în același punct.

$$f_1(x_1, x_2) = 0,$$

$$f_2(x_1, x_2) = 0$$

Newton - aplicat întregului sistem

$$\begin{bmatrix} x_1^{(k+1)} \\ x_2^{(k+1)} \end{bmatrix} = \begin{bmatrix} x_1^{(k)} \\ x_2^{(k)} \end{bmatrix} - \begin{bmatrix} \frac{\partial f_1}{\partial x_1}(x_1^{(k)}, x_2^{(k)}) & \frac{\partial f_1}{\partial x_2}(x_1^{(k)}, x_2^{(k)}) \\ \frac{\partial f_2}{\partial x_1}(x_1^{(k)}, x_2^{(k)}) & \frac{\partial f_2}{\partial x_2}(x_1^{(k)}, x_2^{(k)}) \end{bmatrix}^{-1} \begin{bmatrix} f_1(x_1^{(k)}, x_2^{(k)}) \\ f_2(x_1^{(k)}, x_2^{(k)}) \end{bmatrix} \quad (15)$$

## Ideea: Jacobi-Newton

Iterații - ca la Jacobi; Newton - aplicat pe grupuri de ecuații.

- se rezolvă  $f_1(x_1, x_2^{(k)}) = 0 \Rightarrow x_1^{(k+1)}$ ;
- se rezolvă  $f_2(x_1^{(k)}, x_2) = 0 \Rightarrow x_2^{(k+1)}$ ;

Deplasări simultane:

$$x_1^{(j+1)} = x_1^{(j)} - f_1^{-1}(x_1^{(j)}, x_2^{(k)}) f_1(x_1^{(j)}, x_2^{(k)}) \rightarrow x_1^{(k+1)}$$

$$x_2^{(j+1)} = x_2^{(j)} - f_2^{-1}(x_1^{(k)}, x_2^{(j)}) f_2(x_1^{(k)}, x_2^{(j)}) \rightarrow x_2^{(k+1)}$$

Condiție: să existe inversele.



# Ideea: Jacobi-Newton

Aproape echivalent cu

$$\begin{bmatrix} x_1^{(j+1)} \\ x_2^{(j+1)} \end{bmatrix} = \begin{bmatrix} x_1^{(j)} \\ x_2^{(j)} \end{bmatrix} - \begin{bmatrix} \frac{\partial f_1}{\partial x_1}(x_1^{(j)}, x_2^{(k)}) & 0 \\ 0 & \frac{\partial f_2}{\partial x_2}(x_1^{(k)}, x_2^{(j)}) \end{bmatrix}^{-1} \begin{bmatrix} f_1(x_1^{(j)}, x_2^{(k)}) \\ f_2(x_1^{(k)}, x_2^{(j)}) \end{bmatrix} \quad (16)$$

Termenii care indică cuplajul  $\partial f_1 / \partial x_2$ ,  $\partial f_2 / \partial x_1$  sunt neglijați.

## Ideea: Gauss-Seidel-Newton

Iterații - ca la Gauss-Seidel; Newton - aplicat pe grupuri de ecuații.

- se rezolvă  $f_1(x_1, x_2^{(k)}) = 0 \Rightarrow x_1^{(k+1)}$ ;
- se rezolvă  $f_2(x_1^{(k+1)}, x_2) = 0 \Rightarrow x_2^{(k+1)}$ ;

Deplasări succesive:

$$x_1^{(j+1)} = x_1^{(j)} - f_1^{-1}(x_1^{(j)}, x_2^{(k)}) f_1(x_1^{(j)}, x_2^{(k)}) \rightarrow x_1^{(k+1)}$$

$$x_2^{(j+1)} = x_2^{(j)} - f_2^{-1}(x_1^{(k+1)}, x_2^{(j)}) f_2(x_1^{(k+1)}, x_2^{(j)}) \rightarrow x_2^{(k+1)}$$

Condiție: să existe inversele.

## Aplicații tipice

- Această abordare este mai ales utilă în probleme multifizice, în care separarea sistemului în grupuri de ecuații se face pe considerente fizice.
- Fiecare grup de ecuații provine din formulările matematice ale unor probleme foarte bine definite și cunoscute, cu operatori matematici foarte bine studiați.
- Rezolvarea pe componente (cuplaj slab) poate fi mai robustă decât rezolvarea simultană (cuplaj tare).



Formularea problemei  
Iterații simple  
Newton  
Metode care aproximează Jacobianul  
Metode robuste de tip Newton  
Metode de descompunere

# Rezolvarea pe componente (cuplaj slab) în COMSOL

The screenshot displays the COMSOL Multiphysics interface for a 3D static electromechanics problem. The main window shows the 'Settings' panel for a 'Segregated Step' within 'Study 1'. The 'General' section is expanded, showing 'Spatial coordinates (Material) (comp1.xyz)' as the variables. The 'Components' are set to 'All', and the 'Linear solver' is 'Direct'. The 'Method and Termination' section is also expanded, showing 'Nonlinear method' as 'Constant (Newton)', 'Damping factor' as '1', 'Jacobain update' as 'On every iteration', 'Termination technique' as 'Iterations', and 'Number of iterations' as '1'. The 'Model Builder' on the left shows the hierarchy of the study, with 'Segregated Step 2' selected. The 'Graphics' window on the right shows a 3D wireframe model of a rectangular box with dimensions in micrometers (μm). The x-axis ranges from 0 to 100, the y-axis from 0 to 80, and the z-axis from 0 to 4. A coordinate system (x, y, z) is shown at the bottom left of the graphics window.

Formularea problemei  
Iterații simple  
Newton  
Metode care aproximează Jacobianul  
Metode robuste de tip Newton  
Metode de descompunere

# Rezolvarea pe componente (cuplaj slab) în COMSOL

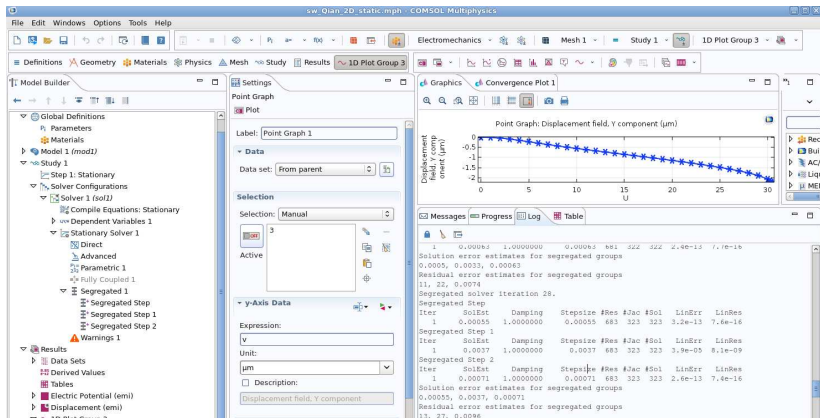
The screenshot displays the COMSOL Multiphysics interface for a static study. The 'Model Builder' tree on the left shows a 'Study 1' containing a 'Stationary' step, which is further divided into 'Stationary Solver 1' and 'Segregated 1'. The 'Segregated 1' step is expanded to show 'Segregated Step 3' selected. The 'Settings' window for 'Segregated Step 3' is open, showing the following configuration:

- Label: Segregated Step 3
- General: Variables: Electric potential (comp1.V)
- Components: All
- Linear solver: Iterative 1
- Method and Termination:
  - Nonlinear method: Automatic (Newton)
  - Initial damping factor: 1
  - Minimum damping factor: 1E-4
  - Restriction for step-size update: 10
  - Use recovery damping factor: Automatic
  - Recovery damping factor: 0.75
  - Termination technique: Tolerance
  - Maximum number of iterations: 20
  - Tolerance factor: 10
  - Termination criterion: Solution or residual
  - Residual factor: 1000

The 'Graphics' window on the right shows a 3D wireframe model of a rectangular box with dimensions 100 μm by 60 μm by 4 μm. The 'Messages' window at the bottom indicates that the file 'sw\_qian\_3d\_static.mph' has been opened.

Formularea problemei  
Iterații simple  
Newton  
Metode care aproximează Jacobianul  
Metode robuste de tip Newton  
Metode de descompunere

# Rezolvarea pe componente (cuplaj slab) în COMSOL



## Referințe

- [Ioan98] D. Ioan et al., *Metode numerice în ingineria electrică*, Ed. Matrix Rom, Bucuresti, 1998. (Capitolul 16)
- [Cheney] Ward Cheney and David Kincaid, *Numerical Mathematics and Computing*, Brooks/Cole publishing Company, 2000.
- [Heath] Michael Heath, *Scientific computing. An Introductory Survey*, McGraw Hill 2002 (capitolul 5 din carte) și alte resurse de la  
disponibilă la <http://heath.cs.illinois.edu/scicomp/>
- [Martinez] Jose Mario Martinez, *Algorithms for Solving Nonlinear Systems of Equations*, 1994, disponibil la  
<http://www.ime.unicamp.br/martinez/nato.pdf>