

# 3. Seminar

## 3 NICHTLINEARE FINITE-ELEMENTE-ANALYSE

3.1 LAGRANGESCHE BEWEGUNGSGLEICHUNG

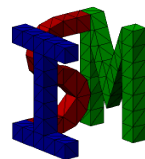
3.2 NEWTON-RAPHSON-ITERATION

3.3 DIE ELEMENTE

3.4 BEISPIEL

3.4.1 Struktur

3.4.2 Nichtlineare statische Analyse



# 3 Nichtlineare Finite-Elemente-Analyse

## 3.1 Lagrangesche Bewegungsgleichung

Newton-Raphson-Iteration:

- Geometrisch und physikalisch nichtlinear

$$\left( {}^t_{0}\mathbf{K}_L^{(i-1)} + {}^t_{0}\mathbf{K}_{NL}^{(i-1)} \right) \Delta \mathbf{U}^{(i)} = {}^{t+\Delta t}\mathbf{R} - {}^{t+\Delta t}_{0}\mathbf{F}^{(i-1)}$$

Modifizierte Newton-Raphson-Iteration:

- Geometrisch und physikalisch nichtlinear

$$\left( {}^t_{0}\mathbf{K}_L + {}^t_{0}\mathbf{K}_{NL} \right) \Delta \mathbf{U}^{(i)} = {}^{t+\Delta t}\mathbf{R} - {}^{t+\Delta t}_{0}\mathbf{F}^{(i-1)}$$

- Physikalisch nichtlinear

$${}^t\mathbf{K} \Delta \mathbf{U}^{(i)} = {}^{t+\Delta t}\mathbf{R} - {}^{t+\Delta t}\mathbf{F}^{(i-1)}$$

${}^t_{0}\mathbf{K}_L$  = inkrementelle lineare Steifigkeitsmatrix, die den Anfangsverschiebungseffekt enthält

${}^t_{0}\mathbf{K}_{NL}$  = inkrementelle nichtlineare Steifigkeitsmatrix, die die Geometrieänderung und die Anfangsspannungen berücksichtigt

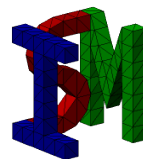
${}^{t+\Delta t}\mathbf{R}$  = Vektor der äußeren Knotenpunktlasten zur Zeit  $t + \Delta t$

${}^t_{0}\mathbf{F}, {}^t\mathbf{F}$  = Vektor der zu den Elementspannungen äquivalenten Knotenpunktkräfte

$\Delta \mathbf{U}^{(i)}$  = Vektor der inkrementellen Knotenpunktverschiebungen

${}^t\mathbf{K}$  = lineare Steifigkeitsmatrix, ohne Anfangsverschiebungseffekt

Institut für Strukturmechanik



$${}^t_0 \mathbf{K}_L = \int_{{}^0V} {}^t_0 \mathbf{B}_L^T {}_0 \mathbf{C} {}^t_0 \mathbf{B}_L d^0V$$

$${}^t_0 \mathbf{K}_{NL} = \int_{{}^0V} {}^t_0 \mathbf{B}_{NL}^T {}^t_0 \mathbf{S} {}^t_0 \mathbf{B}_{NL} d^0V$$

$${}^t_0 \mathbf{F} = \int_{{}^0V} {}^t_0 \mathbf{B}_L^T {}^t_0 \bar{\mathbf{S}} d^0V$$

$${}^t \mathbf{K} = \int_V \mathbf{B}_L^T \mathbf{C} \mathbf{B}_L dV$$

$${}^t \mathbf{F} = \int_V \mathbf{B}_L^T {}^t \Sigma dV$$

- lineare Verzerrungs-Verschiebungsmatrix

$${}^t_0 \mathbf{B}_L = {}^t_0 \mathbf{B}_{L0} + {}^t_0 \mathbf{B}_{L1}$$

- nichtlineare Verzerrungs-Verschiebungsmatrix

$${}^t_0 \mathbf{B}_{NL}$$

- inkrementelle und gesamte Materialeigenschaftsmatrix

$${}_0 \mathbf{C}, \mathbf{C}$$

- Vektor der zweiten Piola-Kirchhoffschen Spannungen

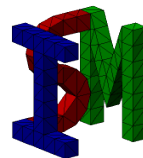
$${}^t_0 \bar{\mathbf{S}}^T = \begin{bmatrix} {}^t_0 S_{xx} & {}^t_0 S_{yy} & {}^t_0 S_{zz} & {}^t_0 S_{xy} & {}^t_0 S_{yz} & {}^t_0 S_{xz} \end{bmatrix}$$

- Matrix der zweiten Piola-Kirchhoffschen Spannungen

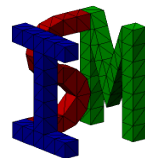
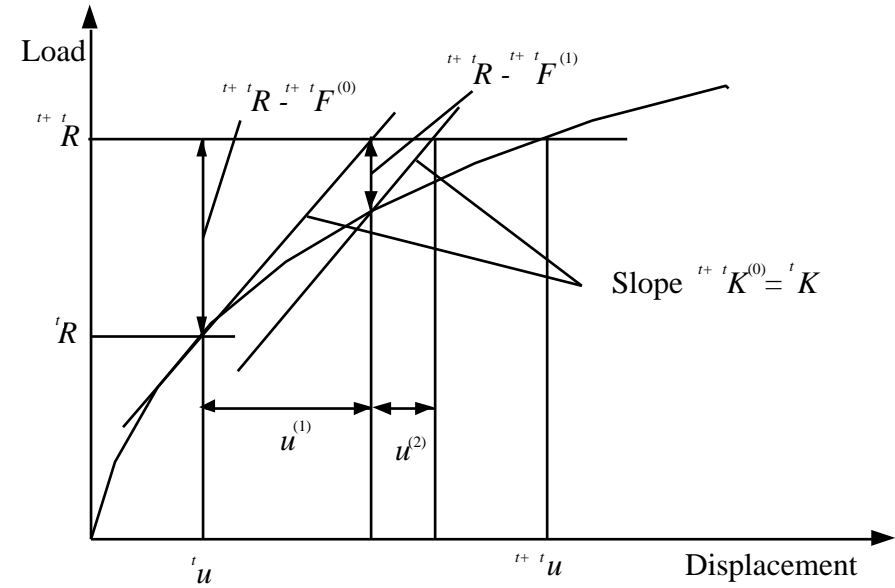
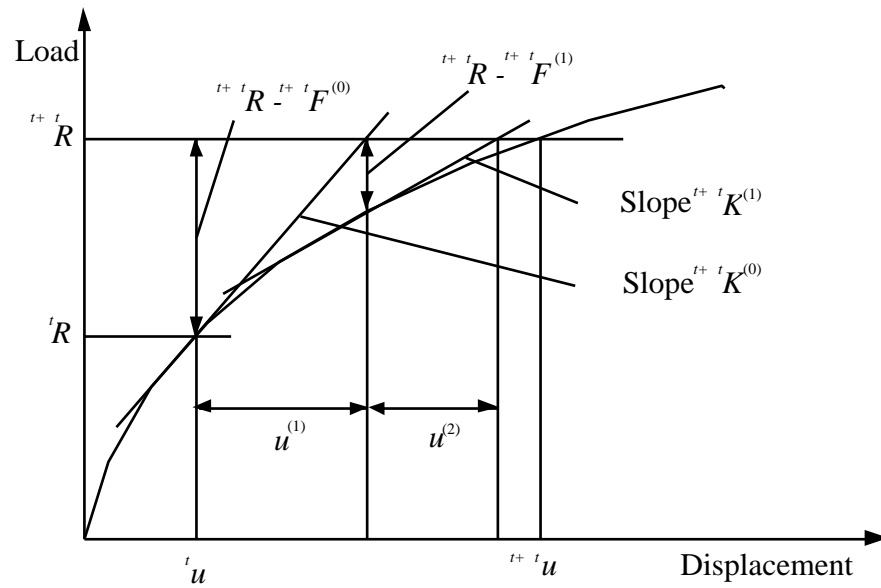
$${}^t_0 \mathbf{S} = \begin{bmatrix} {}^t_0 S_{xx} \mathbf{I}_3 & {}^t_0 S_{xy} \mathbf{I}_3 & {}^t_0 S_{xz} \mathbf{I}_3 \\ {}^t_0 S_{yx} \mathbf{I}_3 & {}^t_0 S_{yy} \mathbf{I}_3 & {}^t_0 S_{yz} \mathbf{I}_3 \\ {}^t_0 S_{zx} \mathbf{I}_3 & {}^t_0 S_{zy} \mathbf{I}_3 & {}^t_0 S_{zz} \mathbf{I}_3 \end{bmatrix}$$

- Vektor der Cauchyschen Spannungen

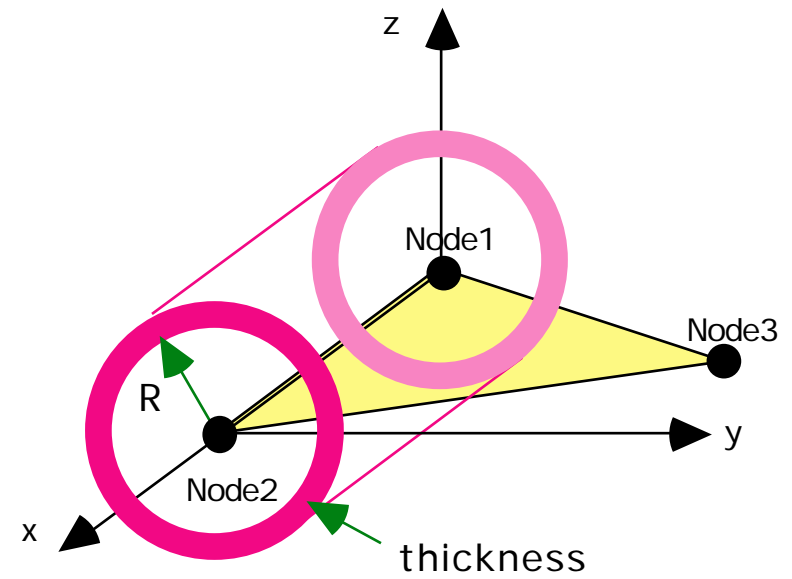
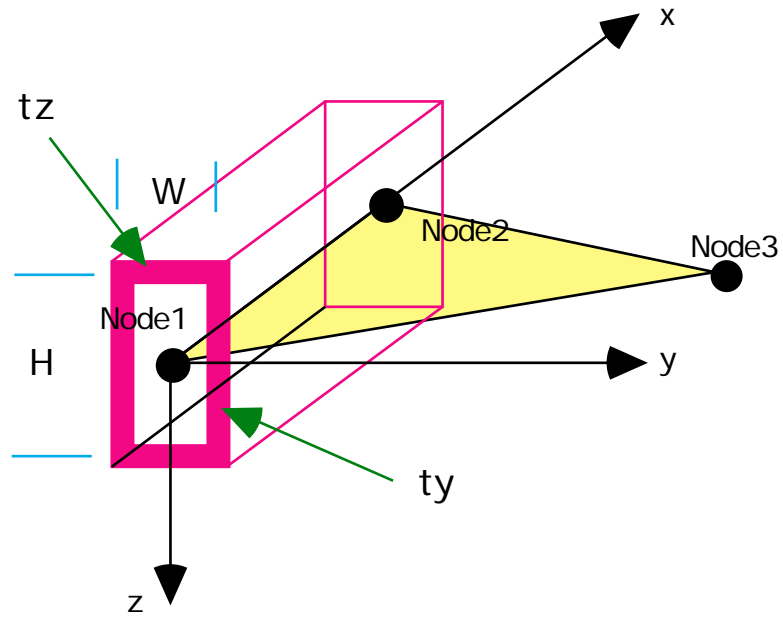
$${}^t \Sigma^T = \begin{bmatrix} {}^t \sigma_{xx} & {}^t \sigma_{yy} & {}^t \sigma_{zz} & {}^t \sigma_{xy} & {}^t \sigma_{yz} & {}^t \sigma_{xz} \end{bmatrix}$$



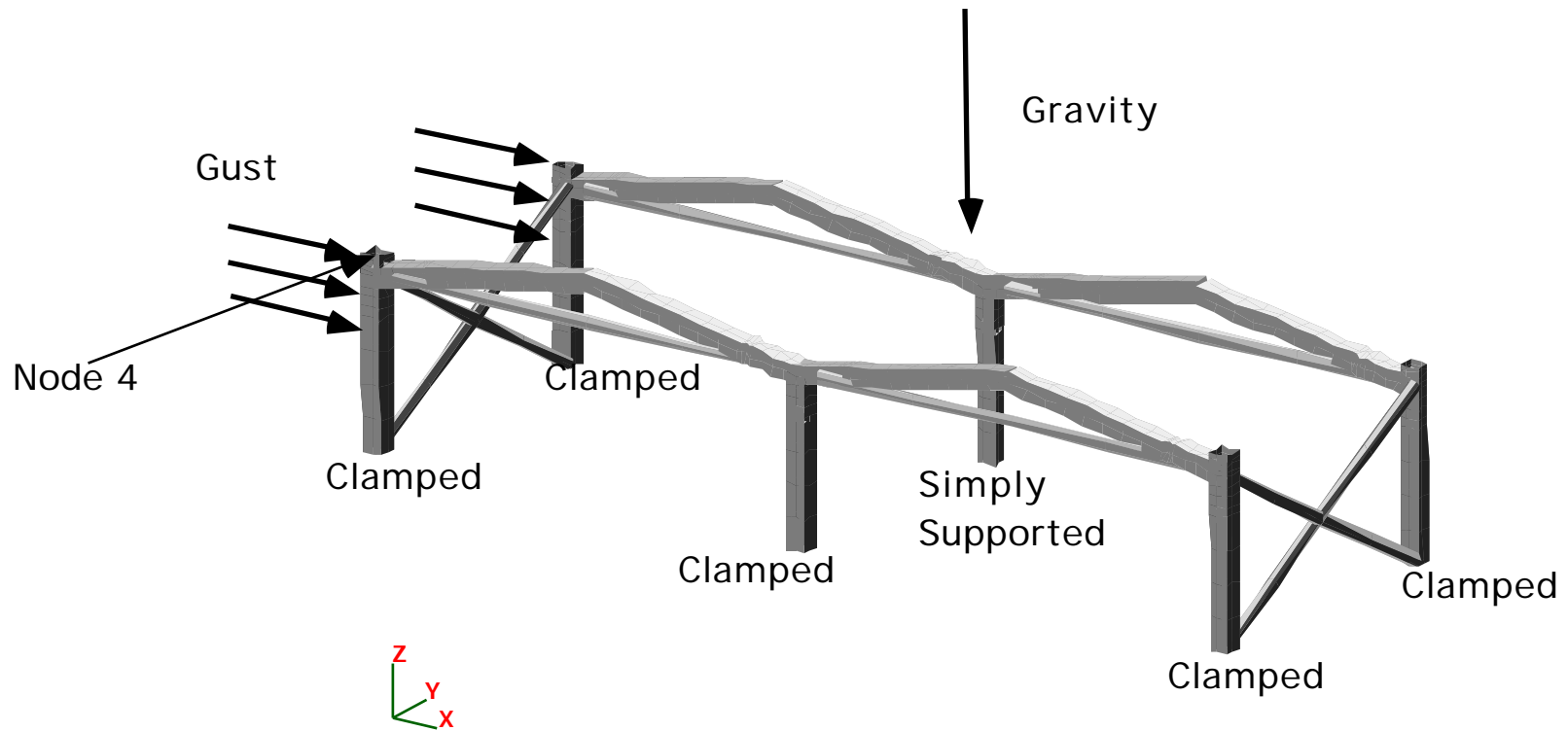
### 3.2 Newton-Raphson-Iteration



### 3.3 Die Elemente BOX und PIPE



### 3.4 Beispiel



### 3.4.1 Struktur

PreSLang file

*Halle.s*

```
* NODE DATA/
NODE ALLOCATE, REPLACE, 66,/
NODE CREATE, THREE_D, 1 0 0 0,/

* NODE RESTRAINTS/
NODE MODIFY, RESTRAINTS, 1 3
    0 1 0 0
    0 0 1 0
    0 0 0 1
, /

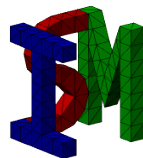
* NODE CONSTRAINTS/
NODE MODIFY, CONSTRAINTS, 26 3
    0 1 3 1 0 0
    0 1 3 0 1 0
```

```
0 1 3 0 0 1
, /

* ELEMENT INCIDENCES/
ELEMENT ALLOCATE, REPLACE, 64,/
ELEMENT CREATE, BOX, 1
    1 2 6,/

* GROUPS/
GROUP ALLOCATE, REPLACE, 3, /
GROUP CREATE, ELEMENTS, 3 64 1 2 3 4
    ...
    61 62 63 64, AllElements/
GROUP CREATE, NODES, 1 34 4 5 8 12 13 14
    ...
    57 58, roof/
GROUP CREATE, NODES, 2 6 2 3 4 35 37 39,
    front/
```

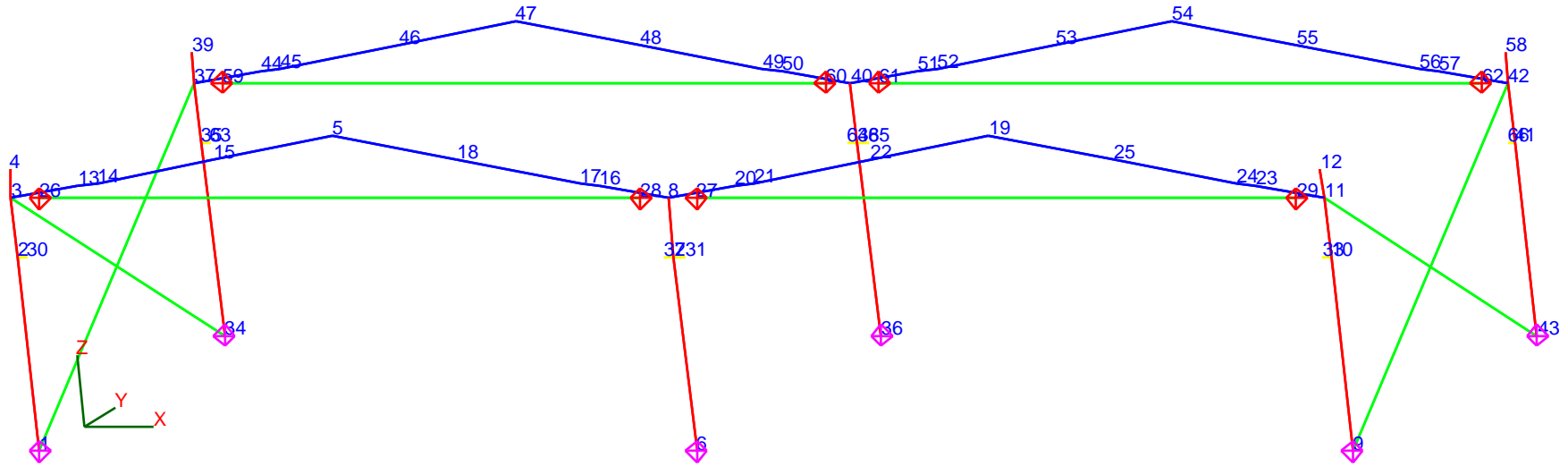
Institut für Strukturmechanik



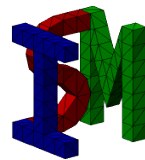
# Anwenderkurs

# SLang

the Structural Language



Institut für Strukturmechanik



## Anwenderkurs

# SLang

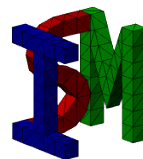
the Structural Language

```
* ELEMENT DATA/  
OBJECT CREATE, REAL VECTOR REPLACE, 3,  
                MATERIAL_DATA/  
OBJECT READ, , MATERIAL_DATA  
                7800 2.1e+11 0.3 ,/  
OBJECT CREATE, REAL VECTOR REPLACE, 4,  
                PHYSICAL_DATA/  
OBJECT READ, , PHYSICAL_DATA  
                0.5 0.5 0.007 0.007 ,/
```

```
ELEMENT MODIFY, MATERIAL_LAW, 1  
                LINEAR_1D, /  
ELEMENT MODIFY, MATERIAL_DATA, 1  
                MATERIAL_DATA, /  
ELEMENT MODIFY, PHYSICAL_DATA, 1  
                PHYSICAL_DATA, /
```

```
* LOAD CASES/
```

Institut für Strukturmechanik



### 3.4.2 Nichtlineare statische Analyse

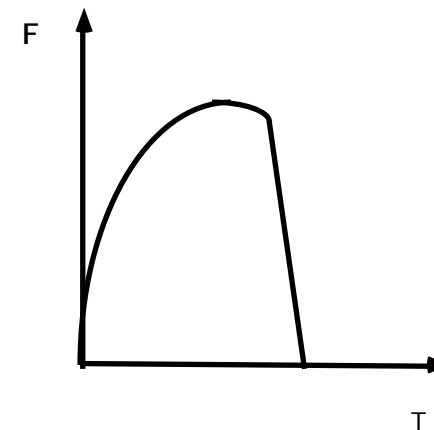
Slang file

*FE\_nonlinear.s*

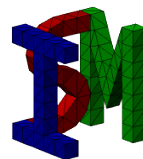
```
#include Halle.s
```

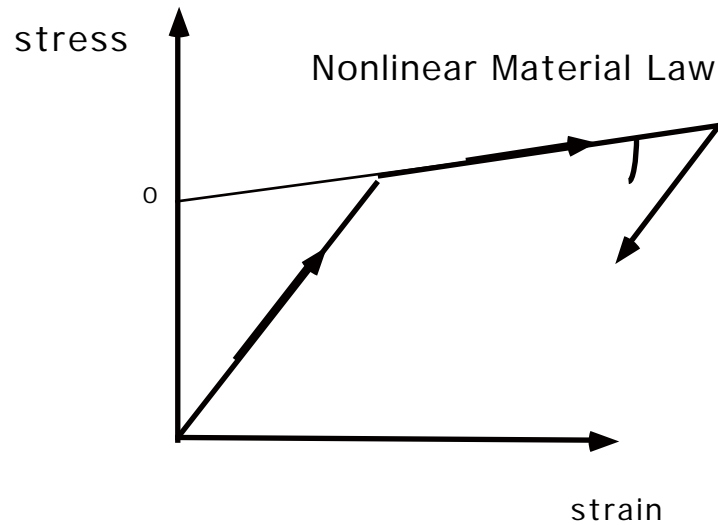
```
*-----/
Build the global load vector
(t+delta t)
  R (loading is deformation-indepent)
-----/
object create, real vector, 15, any_gust/
object read, , any_gust 1 3 5 9 10 11
          12 12.4 12.5 11 10 5 2 1 0,/
object operate, \mul, any_gust 9e3,/
object create, real matrix, 15 2,
          dispload/
object assemble, column, any_gust
          dispload (2),/
object create, real vector, 3, direction/
object read, , direction 0 0 1, /
```

```
node acceleration, translate inertial
          total init, -9.81 direction,/
load allocate, replace, 3,/
element build, total mass, ,/
load build, elements total accelerations,
          1, deadload/
load build, group, 2 front
          1.3 0 0 0 0
          0,windload/
```



Institut für Strukturmechanik

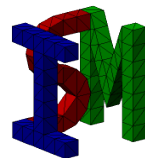


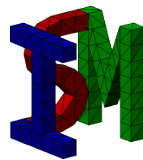
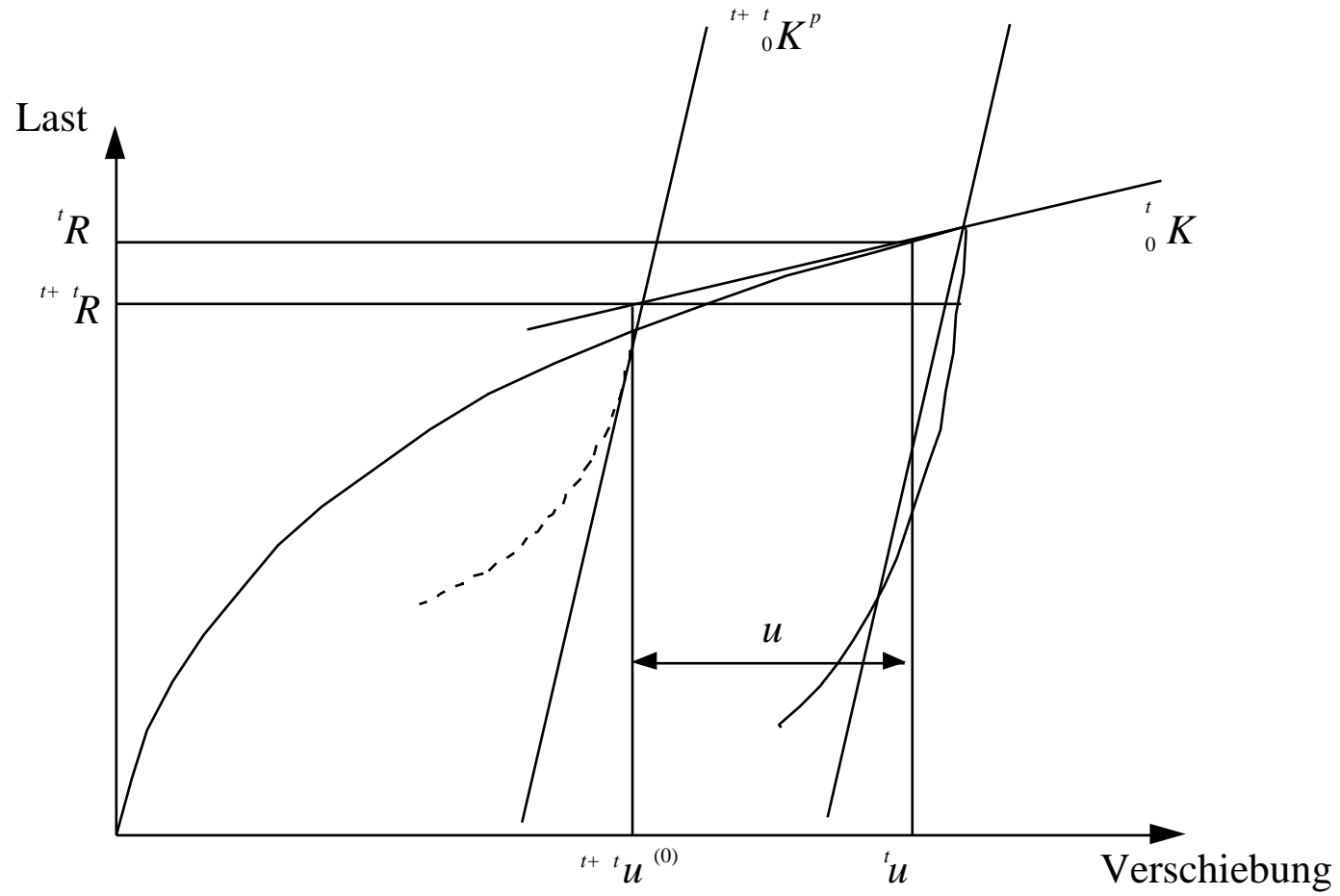


```
*-----
Material properties
-----/
object create, real vector replace, 5,
                    material/
object read,, material 7600 2.1e11 0.3
                    2.4e8 0.0,/
```

```
*-----
Load variation
-----/
#label gust
    object modify, add, i 1,/
    global vector, replace load, 2
    deadload 1 windload any_gust(i),
                    glob_l/
    control gosub,, iterate,/
control if,less, i 15 gust,/
```

```
#label iterate
*-----
    1 (0)    2 (0)
    F,      F,....,
    0        0
-----/
element build, res_force total,,/
global vector, res_force, ,
                    old_resforce/
```





```

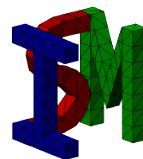
*-----
1      1 (0)  2      2 (0)
R - F      , R - F      ,....
0      0      0      0
-----/
linalg lincomb, ,2 glob_1 1
      old_resforce -1, force_diff/
*-----
0      0      1      1
K + K      , K      K      ,....
0 L 0 NL  0 L 0 NL
-----/
element build, total stiffness, , /
global matrix, stiffness compact, ,
      com_k/
compact factorize, , com_k, glob_sky/
compact solve, replace,glob_sky
      force_diff, disp_diff/
*-----
1 (0)  0  2 (0)  1
U = U, U = U ,....
-----/
node extract, doflist displacements
      replace, ,old_disp/

```

```

node extract, total displacements
      replace, , save_disp/
linalg lincomb, ,2 disp_diff 1
      old_disp 1, new_disp/
node merge, doflist displacements,
      new_disp,/
*-----
0 p 0 p      1 p 1 p
K + K      , K      K      ,....
0 L 0 NL  0 L 0 NL
-----/
element build, stiffness total,,/
global matrix,stiffness compact,,
      com_k/
compact factorize, , com_k, glob_sky/
node merge, total displacements,
      save_disp,/

```



```

*-----/
Start the Newton-Raphson iteration
-----/
#label newton

*-----/
i=1,i=2,....
-----/
object modify, add, iteration_count
                                1,/
control if, integer equal,
    iteration_count 30 failure ,/
*-----/

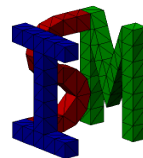
                -1T -1 -1
Static analysis: U = L   D L R,
calculated: increment displacement
vectors
-----/
compact solve, replace, glob_sky
                force_diff, disp_diff/
*-----/
Build the old displacements vector
1 (0) 0   1 (1) 1 (2)  2 (0) 1
  U   = U, U   ,   U   ,..., U   = U

```

```

-----/
node extract, doflist displacements
                replace, ,old_disp/
control if, integer less,
    iteration_count 10 go_ahead,/
object modify, set, disp_factor 0.7,/
#label go_ahead
*-----/
                1 (0)          (1) 1 (1)
                U   + Delta U   = U   ,
                1 (1)          (2) 1 (2)
                U   + Delta U   = U
-----/
linalg lincomb, ,2 disp_diff
disp_factor old_disp 1, new_disp/
node merge, doflist displacements,
                new_disp,/

```



```

*-----
Build the increment restoring force
vektor
Build the old restoring force vektor
Build the out-of-balance load vektor
Calculated the second vector norm
|| 1      1 (1) || || 1      1 (2) ||
|| R - F || || R - F ||
||      0 || ||      0 ||
-----/
element build, res_force total,,/
global vector, res_force, , old_resforce/
linalg lincomb, , 2 glob_l 1 old_resforce
-1 , force_diff/
linalg norm, replace, force_diff,
diff_norm/
*-----
Iteration break off
-----/
linalg norm, replace, glob_l,
glob_l_norm/
object operate,\div, diff_norm
glob_l_norm,/

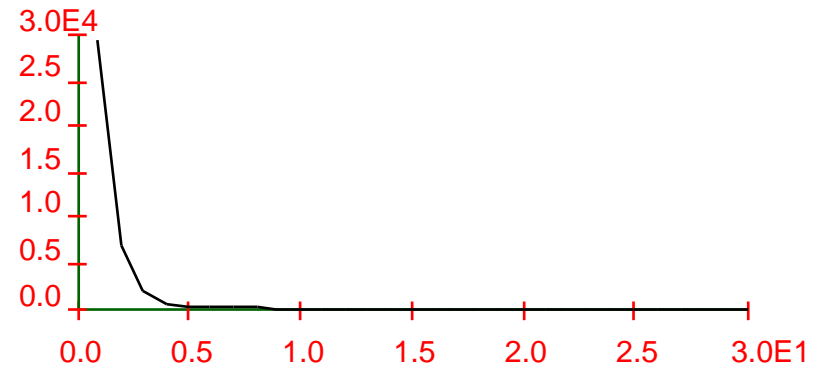
```

```

object modify,, convergence
(iteration_count) diff_norm,/
output view, axes all_points
title, convergence convergence
"||R-F|| / ||R||",/

```

convergence of iteration procedure



```

control if, real less, 1e-2 diff_norm
newton,/

```

