Table 1: Summary of Iteration count, displacement and Residual values at each iteration

| Table of Iterations | | |
|---|---|---|
| **Iteration count** | **$d_{step}^{\,i}$ (cm)** | **$\|\|R_{step}^{i}\|\|$(N)** |
| 0 | $6.67x10^{-3}$ | 20000 |
| 1 | $1.333x10^{-2}$ | 6600 |
| 2 | $1.553x10^{-2}$ | 4356 |
| 3 | $1.6985x10^{-2}$ | 2874.96 |
| 4 | $1.7944x10^{-2}$ | 1897.13 |
| 5 | $1.857637x10^{-2}$ | 1252.1026 |
| 6 | $1.89937x10^{-2}$ | 826.426 |
| 7 | $1.9269x10^{-2}$ | 545.62 |

**Solution 3:**

**Modified Newton Raphson Method:**

**Step#1:**

Number of Iterations required for a converged displacement value=**1**

Displacement and Residual after each iteration:

| Iteration | Displacement (cm) | Residual (N) |
|---|---|---|
| 1 | 0.006667 | 0 |

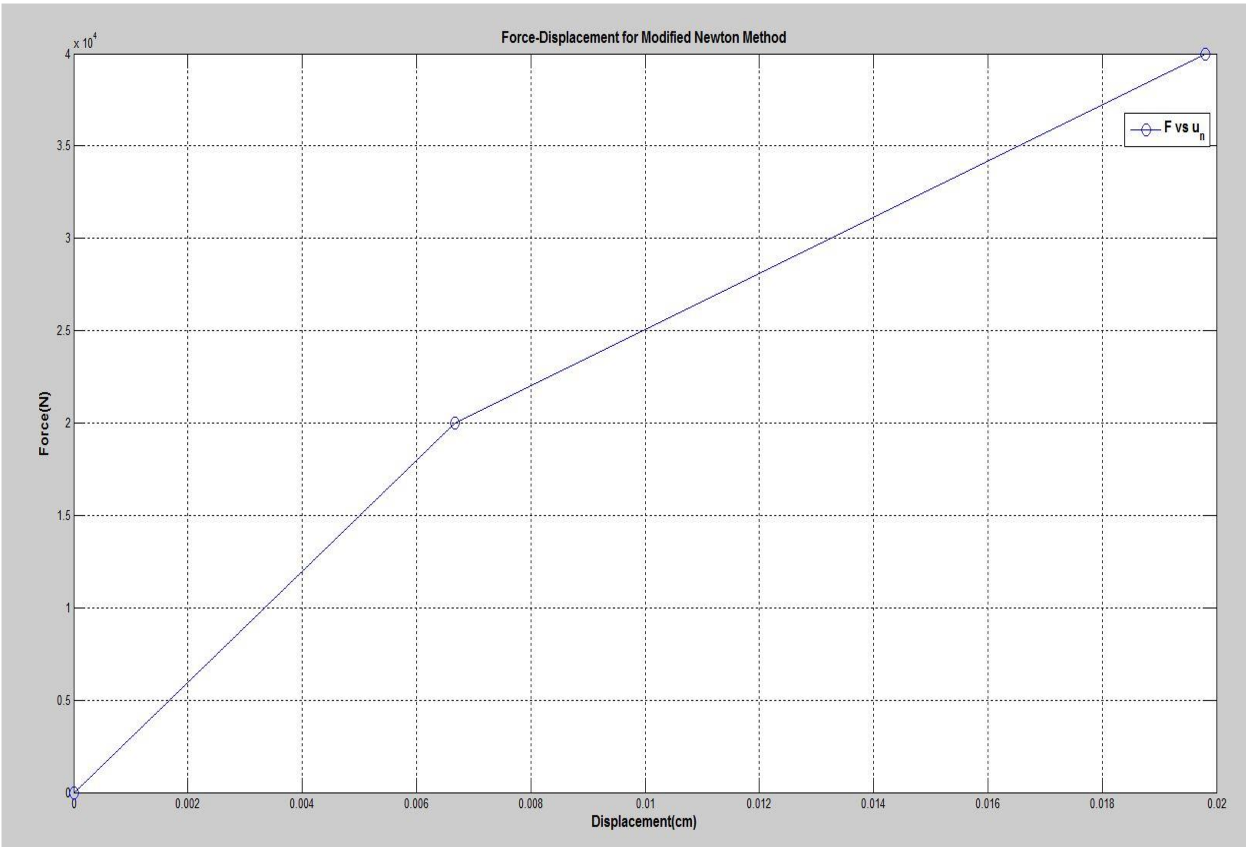
**Step #2**

Table of Displacement and Residual values at each iteration count.

| Iteration # | $d^i$(cm) | Residual (N) |
|---|---|---|
| 1 | 0.013333333 | 6600 |
| 2 | 0.015533333 | 4356 |
| 3 | 0.016985333 | 2874.96 |
| 4 | 0.017943653 | 1897.474 |
| 5 | 0.018576145 | 1252.333 |
| 6 | 0.018993589 | 826.5395 |
| 7 | 0.019269102 | 545.5161 |
| 8 | 0.019450941 | 360.0406 |
| 9 | 0.019570954 | 237.6268 |
| 10 | 0.019650163 | 156.8337 |
| 11 | 0.019702441 | 103.5102 |
| 12 | 0.019736944 | 68.31675 |
| 13 | 0.019759717 | 45.08906 |

| | | |
|---|---|---|
| 14 | 0.019774746 | 29.75878 |
| 15 | 0.019784666 | 19.64079 |
| 16 | 0.019791213 | 12.96292 |
| 17 | 0.019795534 | 8.55553 |
| 18 | 0.019798386 | 5.64665 |
| 19 | 0.019800268 | 3.726789 |
| 20 | 0.01980151 | 2.459681 |
| 21 | 0.01980233 | 1.623389 |
| 22 | 0.019802871 | 1.071437 |
| 23 | 0.019803228 | 0.707148 |
| 24 | 0.019803464 | 0.466718 |
| 25 | 0.01980362 | 0.308034 |
| 26 | 0.019803722 | 0.203302 |
| 27 | 0.01980379 | 0.13418 |
| 28 | 0.019803835 | 0.088558 |
| 29 | 0.019803864 | 0.058449 |
| 30 | 0.019803884 | 0.038576 |
| 31 | 0.019803897 | 0.02546 |
| 32 | 0.019803905 | 0.016804 |
| 33 | 0.019803911 | 0.01109 |
| 34 | 0.019803914 | 0.00732 |
| 35 | 0.019803917 | 0.004831 |
| 36 | 0.019803918 | 0.003188 |
| 37 | 0.01980392 | 0.002104 |
| 38 | 0.01980392 | 0.001389 |
| 39 | 0.019803921 | 0.000917 |
| 40 | 0.019803921 | 0.000605 |
| 41 | 0.019803921 | 0.000399 |
| 42 | 0.019803921 | 0.000264 |
| 43 | 0.019803921 | 0.000174 |
| 44 | 0.019803921 | 0.000115 |
| 45 | 0.019803921 | 7.58E-05 |
| 46 | 0.019803922 | 5.00E-05 |
| 47 | 0.019803922 | 3.30E-05 |
| 48 | 0.019803922 | 2.18E-05 |
| 49 | 0.019803922 | 1.44E-05 |
| 50 | 0.019803922 | 9.49E-06 |
| 51 | 0.019803922 | 6.26E-06 |
| 52 | 0.019803922 | 4.13E-06 |
| 53 | 0.019803922 | 2.73E-06 |
| 54 | 0.019803922 | 1.80E-06 |

| | | |
|---|---|---|
| 55 | 0.019803922 | 1.19E-06 |
| 56 | 0.019803922 | 7.84E-07 |
| 57 | 0.019803922 | 5.18E-07 |
| 58 | 0.019803922 | 3.42E-07 |
| 59 | 0.019803922 | 2.25E-07 |
| 60 | 0.019803922 | 1.49E-07 |
| 61 | 0.019803922 | 9.82E-08 |
| 62 | 0.019803922 | 6.48E-08 |
| 63 | 0.019803922 | 4.28E-08 |
| 64 | 0.019803922 | 2.82E-08 |
| 65 | 0.019803922 | 1.86E-08 |
| 66 | 0.019803922 | 1.23E-08 |
| 67 | 0.019803922 | 8.12E-09 |
| 68 | 0.019803922 | 5.36E-09 |




**Newton-Raphson Method**

**Step#1:**

Number of Iterations required for a converged displacement value=**1**

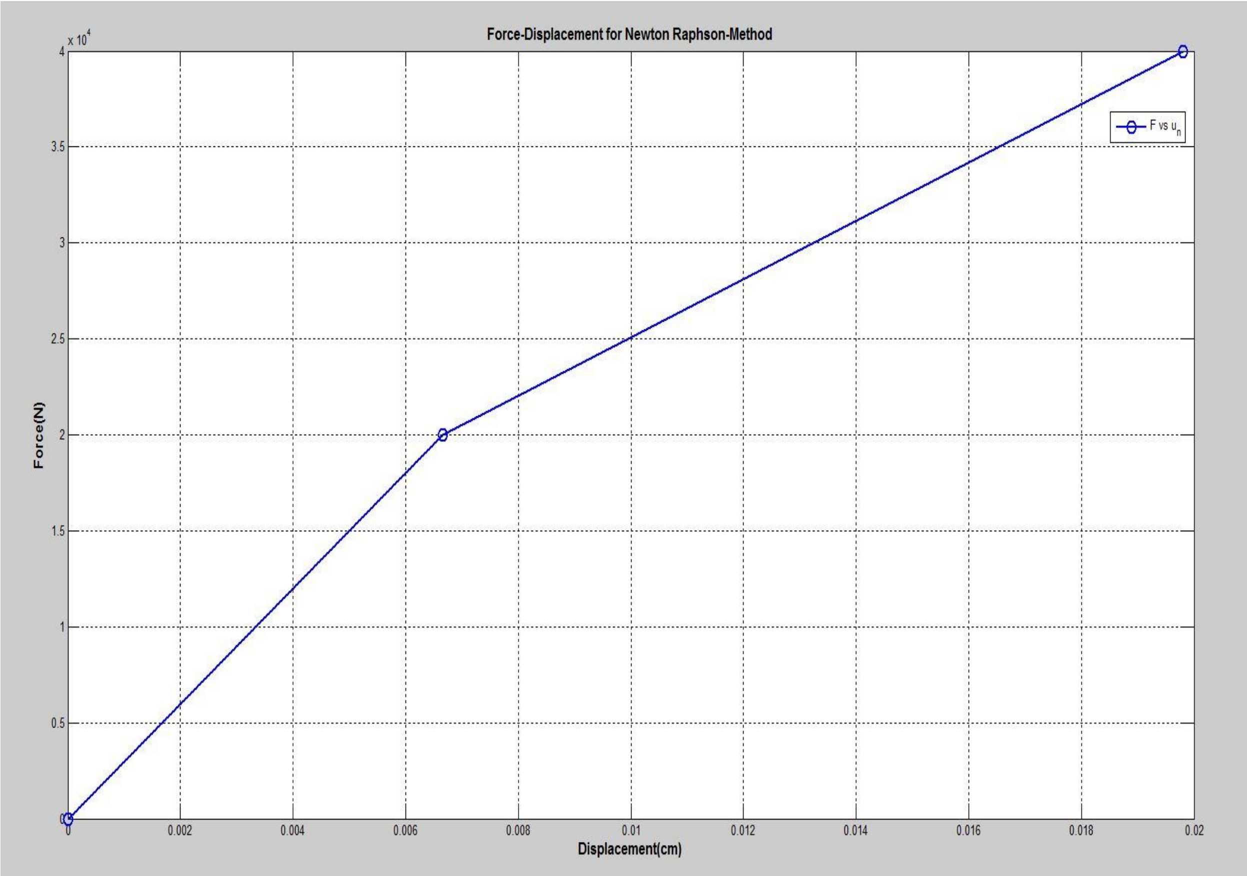
Displacement and Residual after each iteration:

| Iteration | Displacement (cm) | Residual (N) |
|---|---|---|
| 1 | 0.006667 | 0 |

**Step #2**

Table of Displacement and Residual values at each iteration count.

| Iteration | Displacement (cm) | Residual (N) |
|---|---|---|
| 1 | 0.013333 | 6600 |
| 2 | 0.019804 | 0 |



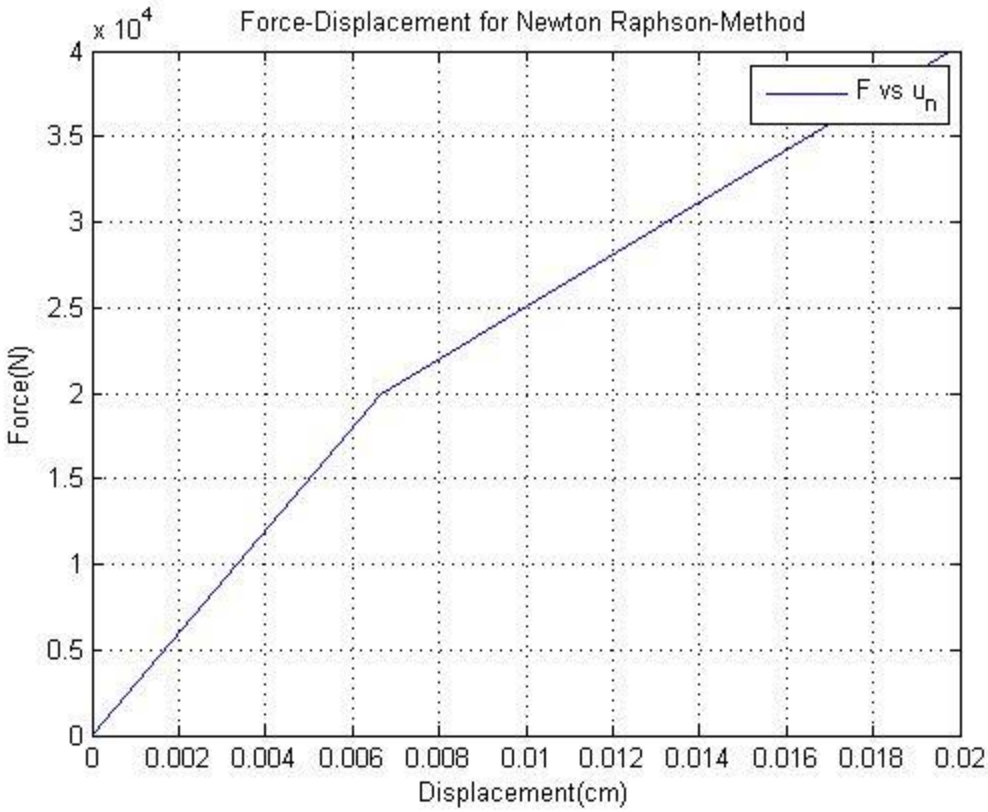

**MATLAB CODES:**

**Newton-Raphson**

```
clear all; close all; clc;

%Newton-Raphson Method
%In the present problem we do not consider unloading part of the force

%Given Parameters:
%state_a: Binary flag to represent the state of the left segment of the rod. It
%assumes a value of 0 for elastic state and 1 for inelastic state.

%state_b: Binary flag to represent the state of the left segment of the rod. It
%assumes a value of 0 for elastic state and 1 for inelastic state.

%num_step: Number of steps in the Newton Raphson

%Number of steps in which the load needs to be applied: 2

%varepsilon: Tolerance parameter for error

%F_ext: External Load as applicable for the step

%F_int: Internal Load as applicable for the step (and iteration)

disp ('%-----------------------------------------------------------------------%')
disp ('%                    CEE570 - Finite Element Method                     %')
disp ('%                              Spring 2016                              %')
disp ('%                  Submitted By: BHAVESH SHRIMALI%                       ')
disp ('%-----------------------------------------------------------------------%')


La=10; Lb=5; L=La+Lb;
F_ext=[2e4;4e4];
epsilon_y=0.002;
E=1e7;A=1;d=0;
E_t=1e5;sigma_y=2e4;
varepsilon=1e-12;
state_a=0;state_b=0;
F_int=0;
K=E*A*(La^-1+Lb^-1);
for i=1:length(F_ext)
    Fe=F_ext(i);
    iter_count=0;
    res_o=Fe-F_int;
    res=Fe-F_int;

    while(abs(res)>varepsilon*abs(res_o))
        iter_count=iter_count+1;
        del_d=res/K;
        d=d+del_d;
        d_arr(iter_count,i)=d;

        epsilon_a=d/La;
        epsilon_b=d/Lb;
```

```
        if epsilon_a<=epsilon_y
            Fa_int=E*epsilon_a*A;
            Ka=E*A/La;
        else
            Fa_int=(E_t*(epsilon_a-epsilon_y)+sigma_y)*A;
            Ka=E_t*A/La;
        end

        if epsilon_b<=epsilon_y
            Fb_int=E*epsilon_b*A;
            Kb=E*A/Lb;
        else
            Fb_int=(E_t*(epsilon_b-epsilon_y)+sigma_y)*A;
            Kb=E_t*A/Lb;
        end
        K=Ka+Kb;
        F_int=Fa_int+Fb_int;
        res=Fe-F_int;
        Residual(iter_count,i)=res;
    end
    iterations(i)=iter_count;
    displacement(i)=d;
end

u_n=vertcat(0,[max(d_arr(:,1));max(d_arr(:,2))]);
Fext=vertcat(0,F_ext);
Res_store=Residual(:,2);
d_final=horzcat(d_arr(:,2),Res_store);
plot(u_n,Fext);
grid on;
xlabel('Displacement(cm)');ylabel('Force(N)');
title('Force-Displacement for Newton Raphson-Method');
legend('F vs u_n');
```

```
%-----------------------------------------------------------------------%
%                    CEE570 - Finite Element Method                     %
%                             Spring 2016                               %
%                   Submitted By: BHAVESH SHRIMALI%
%-----------------------------------------------------------------------%
```



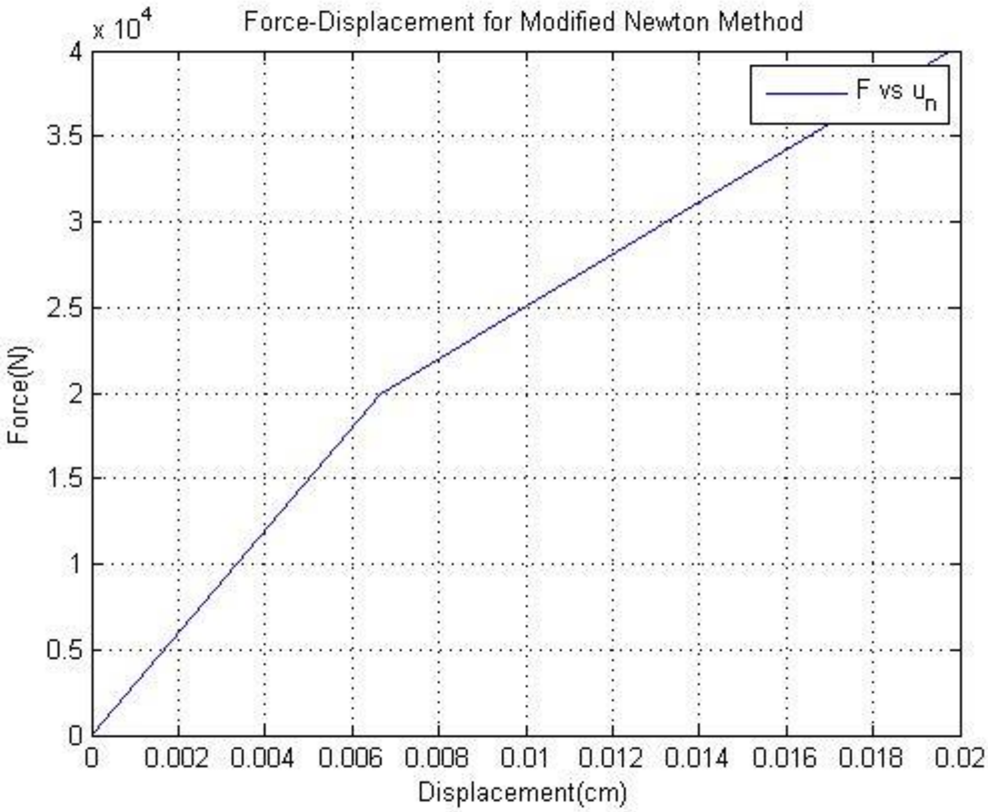

Published with MATLAB® R2013a

**Modified Newton-Raphson Method:**

```
clear all; close all; clc;

%Modified Newton-Raphson Method
%In the present problem we do not consider unloading part of the force

%Given Parameters:
%state_a: Binary flag to represent the state of the left segment of the rod. It
%assumes a value of 0 for elastic state and 1 for inelastic state.

%state_b: Binary flag to represent the state of the left segment of the rod. It
%assumes a value of 0 for elastic state and 1 for inelastic state.

%num_step: Number of steps in the Newton Raphson

%Number of steps in which the load needs to be applied: 2

%varepsilon: Tolerance parameter for error

%F_ext: External Load as applicable for the step

%F_int: Internal Load as applicable for the step (and iteration)

disp ('%-----------------------------------------------------------------------%')
disp ('%                      CEE570 - Finite Element Method                  %')
disp ('%                               Spring 2016                            %')
disp ('%                     Submitted By: BHAVESH SHRIMALI%                   ')
disp ('%-----------------------------------------------------------------------%')


La=10; Lb=5; L=La+Lb;
F_ext=[2e4;4e4];
epsilon_y=0.002;
E=1e7;A=1;d=0;
E_t=1e5;sigma_y=2e4;
varepsilon=1e-12;
state_a=0;state_b=0;
F_int=0;

for i=1:length(F_ext)
    Fe=F_ext(i);
    iter_count=0;
    res_o=Fe-F_int;
    res=Fe-F_int;

    %Check if the left portion of the bar is elastic
    if state_a==0
        Ka=E*A/La;
    else
        Ka=E_t*A/La;
    end
```

```
    %Check if the right portion of the bar is elastic
    if state_b==0
        Kb=E*A/Lb;
    else
        Kb=E_t*A/Lb;
    end
    K=Ka+Kb;
    while(abs(res)>varepsilon*abs(res_o))
        iter_count=iter_count+1;
        del_d=res/K;
        d=d+del_d;
        d_arr(iter_count,i)=d;
        epsilon_a=d/La;
        epsilon_b=d/Lb;
        if epsilon_a<=epsilon_y
            Fa_int=E*epsilon_a*A;
            state_a=0;
        else
            Fa_int=(E_t*(epsilon_a-epsilon_y)+sigma_y)*A;
            state_a=1;
        end

        if epsilon_b<=epsilon_y
            Fb_int=E*epsilon_b*A;
            state_b=0;
        else
            Fb_int=(E_t*(epsilon_b-epsilon_y)+sigma_y)*A;
            state_b=1;
        end
        if iter_count==1
            res_o=Fe-(Fa_int+Fb_int);
        end
        F_int=Fa_int+Fb_int;
        res=Fe-F_int;
        Residual(iter_count,i)=res;
    end
    iterations(i)=iter_count;
    displacement(i)=d;
end

u_n=vertcat(0,[max(d_arr(:,1));max(d_arr(:,2))]);
Fext=vertcat(0,F_ext);
Res_store=Residual(:,2);
d_final=horzcat(d_arr(:,2),Res_store);
plot(u_n,Fext);
grid on;
xlabel('Displacement(cm)');ylabel('Force(N)');
title('Force-Displacement for Modified Newton Method');
legend('F vs u_n');
```

```
%------------------------------------------------------------------------%
%                  CEE570 - Finite Element Method                        %
%                          Spring 2016                                   %
%                Submitted By: BHAVESH SHRIMALI%
%------------------------------------------------------------------------%
```




*Published with MATLAB® R2013a*

**Newton Raphson Method**

| |
|---|
| Load Step (n) and initialize the Iteration Counter to a value of 1. We know $\mathbf{R_n}$ at this point |

| |
|---|
| Calculate **Δd** corresponding to the iteration (i) using the relation |

| |
|---|
| Update the value of $\mathbf{d^{i+1}}$ **as follows:** $\mathbf{d^{i+1}=d^{i}+\Delta d^{i}}$<br>Using this value we evaluate the Strains in the bar: $\boldsymbol{\varepsilon_a} =$ |

| |
|---|
| The strains are checked for the elastic-plastic limit. If the strains exceed the yield strain limit then the stresses, and correspondingly forces, are calculated using the inelastic stress-strain curve. Accordingly $\mathbf{K_a}$ and $\mathbf{K_b}$ are also updated. |

| |
|---|
| $\mathbf{\|\|R(i+1)\|\|=Fext-[F_{aint}-F_{bint}]}$ and check for the residual. |

**Modified Newton-Raphson Method:**

Load Step (n) and initialize the Iteration Counter to a value of 1. We know $\mathbf{R_n}$ at this point

Calculate **Δd** corresponding to the iteration (i) using the relation

Update the value of $\mathbf{d^{i+1}}$ **as follows:** $\mathbf{d^{i+1}=d^{i}+\Delta d^{i}}$

Using this value we evaluate the Strains in the bar: $\boldsymbol{\varepsilon_a} =$

The strains are checked for the elastic-plastic limit. If the strains exceed the yield strain limit then the stresses, and correspondingly forces, are calculated using the inelastic stress-strain curve. $\mathbf{K_a}$ and $\mathbf{K_b}$ are never updated.

$\mathbf{||R(i+1)||=Fext-[F_{aint}-F_{bint}]}$ and check for the residual.