Solution for Cantilever Beam Deflection – Float Configuration

1. Abstract

In this project, the journal "Cantilever beam electrostatic MEMS actuators beyond pull-in" by Subrahmanyam Gorthi *et al* 2006 *J. Micromech. Microeng.* 16 1800 is analysed. The operational range of electrostatic MEMS parallel plate actuators can be extended beyond pull-in in the presence of an intermediate dielectric layer, which has a significant effect on the behaviour of such actuators. Three possible static configurations namely, floating, pinned and flat of the beam are identified over the operational voltage range. The focus is on solving the equation of static analysis of the floating configuration.

2. Introduction

A cantilever beam is configured parallel to the dielectric layer. The dielectric layer and the cantilever beam is connected to a source with voltage(V). The beam with free end not touching the dielectric surface is called floating.

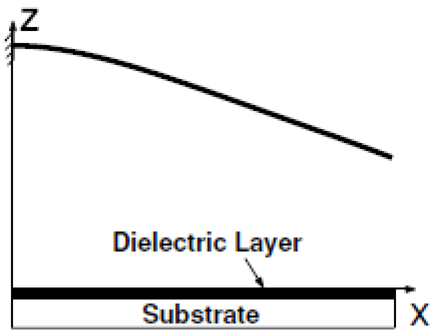

Fig 1. Floating Configuration
(Retrieved from Gorthi etal 2006)

The governing equation for the floating configuration and the boundary values are presented. Newton Raphson's numerical method is made use in solving this non-linear system of equations.

3. Numerical Method

Newton Raphson's approximation is considered to solve this non-linear system. This method solves the governing equations by applying Jacobian matrix, computing the corresponding displacement and iterating until the error is below the threshold. A method of finding the root x_0 of the equation f(x) = 0. The advantage in this scheme are its accuracy and rapid convergence. The method requires initial guess vector with values for function.

$$\vec{Z}_{k+1} = \vec{Z}_k - J^{-1} * \vec{G}(\vec{Z}_k)$$

In the first iteration, $\vec{Z}_k$ is passed as initial guess and the result is swapped to $\vec{Z}_k$ in the consecutive iterations. Jacobian is computed as follows,

$$\text{Jacobian} = \begin{bmatrix} \frac{\partial G_{-1}}{\partial Z_{-1}} & \cdots & \frac{\partial G_{12}}{\delta Z_{12}} \\ \vdots & \ddots & \vdots \\ \frac{\partial G_{12}}{\partial Z_{-1}} & \cdots & \frac{\partial G_{12}}{\partial Z_{12}} \end{bmatrix}$$

This Jacobian is inversed and multiplied with the vector $\vec{G}(\vec{Z}_k)$ and subtracted from the initial guess. This result is used as $\vec{Z}_k$ in the further iterations.

4. Problem Description

The cantilever beam and the dielectric layer is of length(l), width(w) and the thickness of the beam is t_b and dielectric layer is t_d. The gap between beam and the dielectric layer is g. Along the dielectric is the horizontal axis x and vertical axis Z.

Parameter	Description	Value	Unit
w	Beam width	10^{-6}	m
t_b	Beam thickness	$3\text{x}10^{-6}$	m
L	Beam length	$250\text{x}10^{-6}$	m
t_d	Dielectric layer thickness	10^{-3}	m
ρ	Mass per unit length of the beam	2329.6	kg/m3
g	Zero bias height of the cantilever beam	$3\text{x}10^{-6}$	m
E	Young's modulus of the beam material	$169\text{x}10^{9}$	Pa
ϵ_0	Permittivity of free space	$8.85\text{x}10^{-12}$	F/m
ϵ_r	dielectric constant	11.68	

The beam's deflection on varying voltage and grid size is approximated using newton Raphson method. Possible approximations for different grid size say 10, 20, 40 and varying voltage from 0 to 300 is derived.

5. Model Considerations

The parameters are normalized before solving the equation for convergence. The non-dimensional quantity, volt becomes,

$$\hat{V} = \sqrt{\frac{\epsilon 0\; w\; l^4}{2\; EI\; g^3}}\; V$$

$$h = \frac{t_d}{g\epsilon_r}$$

Parameters x and z are normalized with l and g respectively. Since, there is no time consideration, the governing equation is,

$$\frac{\partial \hat{z}}{\partial \hat{x}} = \frac{\hat{V}}{(\hat{z} + h)^2}$$

The derivative is solved using a finite difference scheme. The boundary conditions involve derivatives so two fictitious points are introduced on both sides of the cantilever beam. The x – axis is divided into uniform grid of N points with step size Δx. These points are denoted by x_1, x_2, x_3, … x_N and the fictitious points x_{-1}, x_0, x_{N+1}. x_{N+2}. Let z_j denote the beam deflection at $x = x_j$, j = 1, 2, … N.

The five-point finite difference of the governing equation is,

$$Z_{j-2} + 4Z_{j-1} + 6Z_j - 4Z_{j+1} + Z_{j+2} + \frac{\Delta x^4 V^2}{(Z_j + h)^2} = 0$$

Left boundary condition in finite difference scheme,

$$Z_1 - 1 = 0$$
$$Z_{-1} - 8Z_0 + 8Z_2 - Z_3 = 0$$

Right boundary condition in finite,

$$-Z_{N-2} + 16Z_{N-1} - 30Z_N + 16Z_{N+1} - Z_{N+2} = 0$$
$$-Z_{N-2} + 2Z_{N-1} - 2Z_{N+1} + Z_{N+2} = 0$$

Let the initial guess be,

$\vec{Z}_{k} =$ [1,1 …. 1]; or $\vec{Z}_{k} =$ [g, g……g];

And function vector $\vec{G}(\vec{Z}_k)$ is derived from the governing equation, where j = -1,0,1…N+12

$$G_{-1}(Z_{-1}, Z_0, Z_1, \ldots Z_{N+2}) = 0$$
$$G_0(Z_{-1}, Z_0, Z_1, \ldots Z_{N+2}) = 0$$
$$G_1(Z_{-1}, Z_0, Z_1, \ldots Z_{N+2}) = 0$$
$$\vdots$$
$$G_{N+2}(Z_{-1}, Z_0, Z_1, \ldots Z_{N+2}) = 0$$

We get Jacobian Matrix like below,

$$\begin{bmatrix} 0 & 0 & 1 & & 0 & 0 & 0 \\ 1 & -8 & 0 & \cdots & 0 & 0 & 0 \\ 1 & -4 & A(Z_1) & & 0 & 0 & 0 \\ & \vdots & & \ddots & & \vdots & \\ 0 & 0 & 0 & & A(Z_{10}) & -4 & 1 \\ 0 & 0 & 0 & \cdots & -30 & 16 & -1 \\ 0 & 0 & 0 & & 0 & -2 & 1 \end{bmatrix}$$

And $A(Z_i)$ is computed as,

$$A(Z_i) = 6 - 2\frac{\Delta x^4 V^2}{(Z_i + h)^3}$$

6. Results

The model is programmed in MATLAB to derive the solution of floating configuration. Plots are as below,

A. Grid Size = 10, Volt = 100

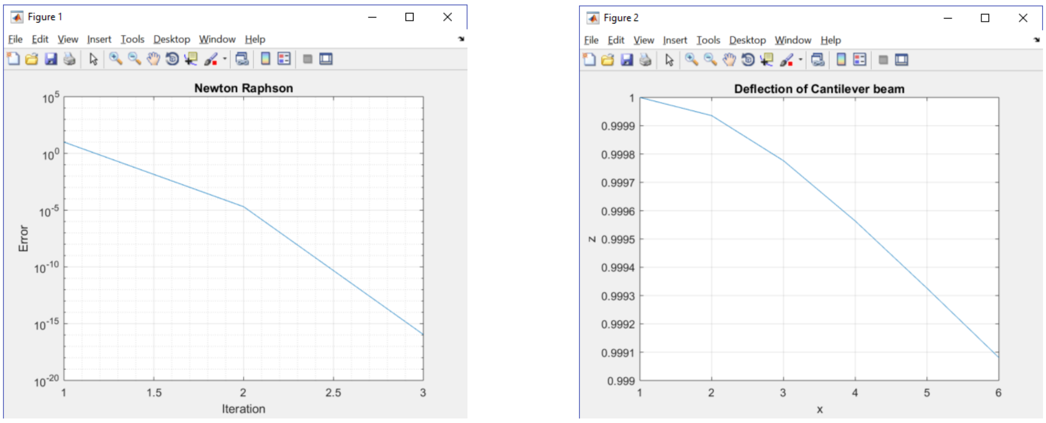

Fig 1. a) Error vs Iterations. b) Normalized z vs x removing fictitous points.

B. Grid Size = 20, Volt = 200

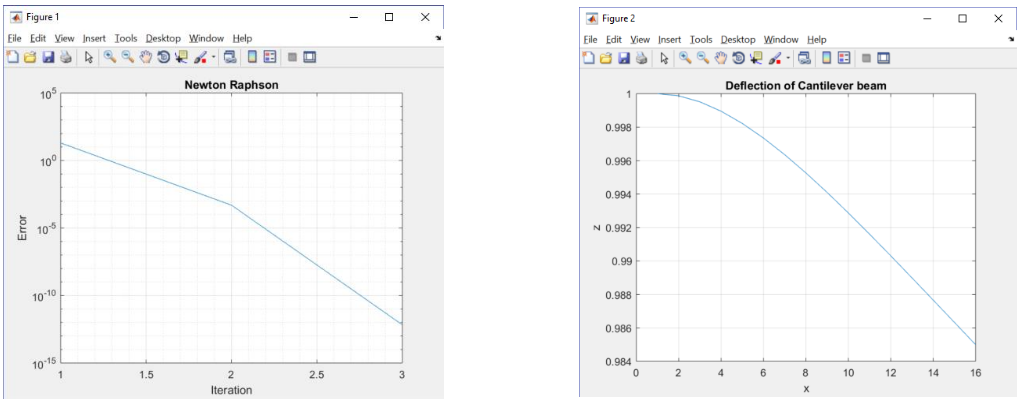

Fig 2. a) Error vs Iterations. b) Normalized z vs x removing fictitous points.

C. Grid Size = 40, Volt = 300

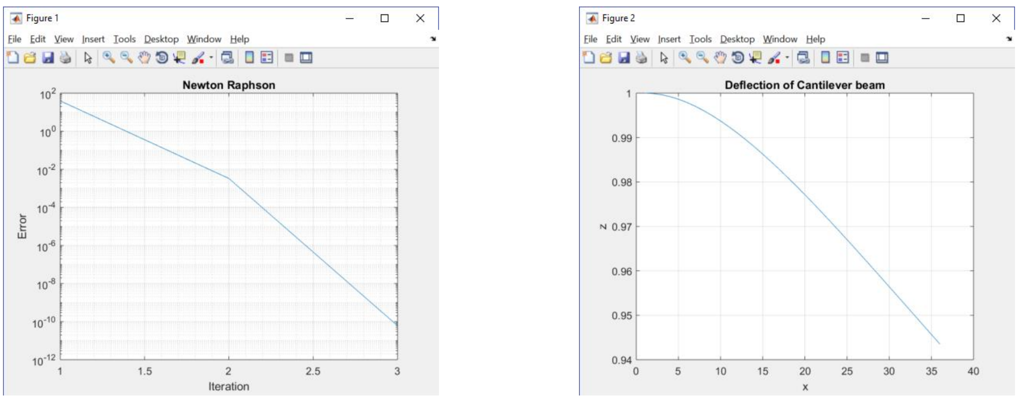

Fig 3. a) Error vs Iterations. b) Normalized z vs x removing fictitous points.

D. Scaled result with Grid Size = 40, Volt = 300.

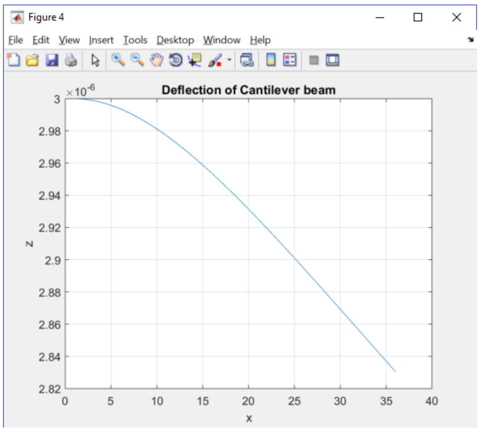

Fig 4. Scaled z vs x removing fictitous points.

7. Discussion

From the results, the deflection increases when there is an increase in the voltage. The difference can be figured out by comparing fig 1,2,3. The approximation is smoother when step size is increased from 10, 20 and 40. Increase the tolerance, increases the level of accuracy in the approximation. A system of $A * x = B$ can be solved faster as A\B in MATLAB. Computing $A(Z_i)$ separately helps in faster computation and convergence. Performing computation having symbolic variables is highly expensive, hence it is recommended to convert them to double type and process it.

8. Conclusion

As stated, the solution for floating configuration of a cantilever beam on various voltages applied on it is derived with the five-point finite difference scheme governing equation. The parameters are normalized for convenience. The governing equations were used in Newton Raphson method of finding the root and the vector of z values is computed with number of iterations until the specified tolerance is reached. The 4 bounding fictitious points are removed from the result and plotted as a graph and the error on each iteration is plotted as semi-log plot. Thus, the floating-point configuration adapted from “Cantilever beam electrostatic MEMS actuators beyond pull-in” - Subrahmanyam Gorthi *et al* 2006 *J. Micromech. Microeng.* 16 1800 is solved.

9. Appendix

MATLAB CODE:

newtonraphson.m

```
clear all;
clc;

% Grid Settings
gd = 40;
```

```
max_itr = 50;
N = gd - 4;
lhat = 1;
steps = lhat/(gd-1);
tolerance = 10^-60;


Z = sym('z', [1 gd]);

%voltage
volt = 300;


% Parameter values
w = 10^-6;
tb = 3*10^-6;
length = 250e-6;
td = 10^-3;
massp = 2329.6;
g = 3*10^-6;
E = 169e9;
e0 = 8.85e-12;
er = 11.68;
I = w*(tb^3)/12;

%Normalised values
Vhat = 2*volt*sqrt((e0*w*length^4)/(2*E*I*g^3));
h = td/(g*er);

val = Vhat^2*steps^4;

% Declate the finite difference function
[fn] = getfunction(gd, val, Z, h);
% Compute the jacobian function
jac = getjacobian(fn, Z, gd);
%initial value
init = zeros(size(Z));
for ii=1:gd
    init(ii) = g;
end

% Perform newton raphson
for i = 1:max_itr
    i
    J = subs(jac, Z, init);
    Fn = subs(fn, Z, init);
    Fn_db = double(Fn);
    J_db = evaljacobian(J, init, val, h)
    X = transpose(init) - (J_db\Fn_db);
    err(:,i) = abs(X-transpose(init));
    init = transpose(X);
```

```
    error(i) = sum(err(:,i));
    if(err(:,i) < tolerance)
        break;
    end
end

% Error vs itertion
figure, semilogy(1:i,error(1:i));
title('Newton Raphson');
xlabel('Iteration');
ylabel('Error');
grid minor;grid;

% plot x vs z
X = X*g;
figure, plot(1:N, X(3:gd-2));
title('Deflection of Cantilever beam');
xlabel('x');
ylabel('z');
grid;
```

getfunction.m

```
function [fn, Z] = getfunction(N,val,Z, h)

%Governing equation
%Z(i-2) -4*Z(i-1) +6*Z(i) -4*Z(i+1) +Z(i+2) +
V.^2.steps^4/(Z(i)+h).^2
fn(1,1) = Z(3) -1;
fn(2,1) = Z(1) -8*Z(2) + 8*Z(4) -Z(5);
index = 2;

for i=3:N-2
    index = index + 1;
    fn(index, 1) = Z(i-2) -4*Z(i-1) +6*Z(i) -4*Z(i+1) +Z(i+2)
+ val/(Z(i)+h)^2;
end

fn(N-1,1) = -Z(N-4) + 16*Z(N-3) - 30*Z(N-2) + 16*Z(N-1) -Z(N)
;
fn(N,1) = -Z(N-4) + 2*Z(N-3) -2*Z(N-1) + Z(N);
```

getjacobian.m

```
function jac = getjacobian(fn, Z, N)

for m=1:N
```

```
    for n=1:N
        jac(m,n) = diff(fn(m,1),Z(n));
    end
end
```

evaljacobian.m

```
function J = evaljacobian(Jac, Z, val, h)

J = double(Jac);
[row, col] = size(J);

for i = 3:row-2
    for j = 3:col-2
        if i==j
            value = 6 - (2*val)/(Z(i)+h)^3;
            J(i,j) = double(value);
        end
    end
end
```

Acknowledgment

I am highly indebted to Dr. Luis Escalona for the guidance throughout the Fall 2018 semester. Also, I would to thank for providing necessary information regarding the project and support in completing the project.

Reference

[1] Subrahmanyam Gorthi *et al* 2006 *J. Micromech. Microeng.* **16** 1800 Cantilever beam electrostatic MEMS actuators beyond pull-in

[2] Thompson, C.M. and Shure, L. For Use with MATLAB; [user's Guide], 1995, MathWorks