HiQLab: Simulation of Resonant MEMS

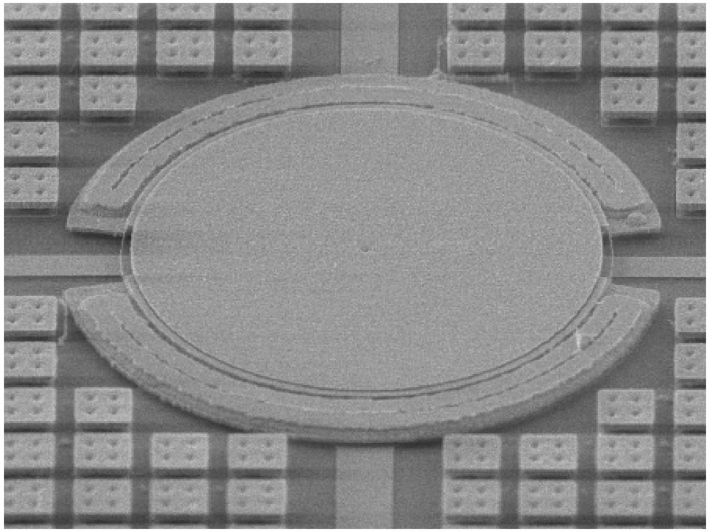
Tsuyoshi Koyama
Civil Engineering, UC Berkeley

Overview

- Introduction
- Basic analysis
- Anchor loss
  - Disk resonator
- Electromechanical systems
  - Dielectric filled gap resonator

Motivation

- Understanding the disk resonator.
  - What causes changes in Q ?
  - What damping mechanisms affect Q ?



SEM of 41.5 um radius poly-SiGe disk resonator

Goal for HiQLab

- •Introduction
- •Basic analysis
- •Anchor loss
- •EM systems

Understand resonant MEMS behavior

- Accurately model damping mechanisms (fluid damping, material losses, etc.)
- Develop fast solvers
- Verify with experiments
- Help designers

Difficulties

- Contribution of each damping mechanism is unclear
- Physically realistic damping models are required
- Mathematical issues (Damping makes the problem naturally non-symmetric, ω must be resolved for accurate Q)

Advantages of developing HiQLab

- Incorporate physically realistic models
  - Need to work with low level details.
  - Easier if source code is accessible.
- Take advantage of problem structure
  - Less restriction on accessible problems
  - Develop solvers for fast computation
- Free software to collaborate with the community

Current capabilities

- Finite Element tool
- Dimensions:
  - 1D, 2D, 3D, Axisymmetric
- Analysis:
  - Steady-state/Static
- Elasticity, scalar wave
- Coupled problems:
  - Electromechanical
  - Thermomechanical

Special features

- Damping mechanisms:
  - Anchor loss
  - Thermoelastic damping
- Efficient Algorithms:
  - Eigenfrequency computation
  - Arnoldi based Reduced Order Model (ROM) for transfer function computation
- MATLAB/Lua user interface

User interface

•**Introduction**
•Basic analysis
•Anchor loss
•EM systems

MATLAB	Lua
1. Write Lua input file - mesh description - define functions	1. Write Lua input file - mesh description - define functions
2. Process in MATLAB - solve problem - visualize results	2. Process in Lua - solve problem

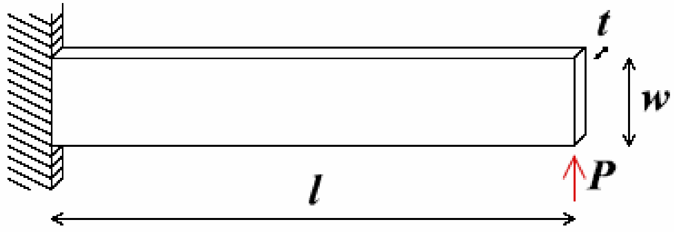
	MATLAB	Lua
Pros	rich solvers visualization tools	fast, free
Cons	license computational overhead	visualization not ready yet

User interface

- Introduction
- Basic analysis
- Anchor loss
- EM systems

	MATLAB	Lua
	1. Write Lua input file - mesh description - define functions 2. Process in MATLAB - solve problem - visualize results	1. Write Lua input file - mesh description - define functions 2. Process in Lua - solve problem
Pros	rich solvers visualization tools	fast, free
Cons	license computational overhead	visualization not ready yet

Step 1. Lua input file



Cantilever beam

•Include file
Make mesh object

```
require 'common.lua'
mesh = Mesh:new(2)
```

•Make element

```
etype = make_material_e('silicon2','planestress')
```

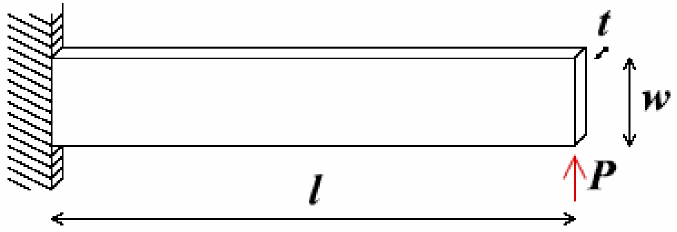
•Define geometry and block

```
order    = 3          -- Order of element
dense    = 2.0e-6     -- Approximate element size
l        = 10e-6      -- Beam length
w        =  2e-6      -- Beam width
t        =  1e-6      -- Beam thickness
P        =  3e-6      -- Tip force
mesh:blocks2d({0,l},{-w/2,w/2},etype,order,dense)
```

•Define and set boundary conditions

```
function bc_function(x,y)
  if x==0 then return 'uu', 0,  0; end
  if x==l and y==-w/2
          then return ' f',    P/t; end
end
mesh:set_bc(bc_function)
```

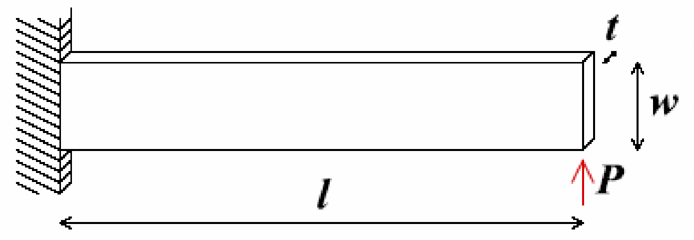
Make mesh




```
require 'common.lua'
mesh = Mesh:new(2)
```

- Include files with require
  - 'common.lua' defines materials, block generator
- Define a new mesh (which *must* be called mesh)
  - Number of dimensions = 2

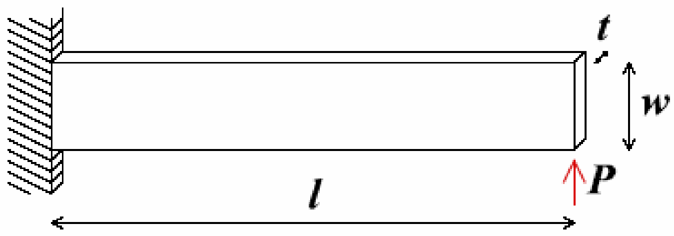
Make element




```
etype = make_material_e('silicon2','planestress')
```

- Define a polysilicon material in plane stress
  - Properties of several materials such as PolySi, SCS, SiGe, SiC are included in the database 'materials.lua'

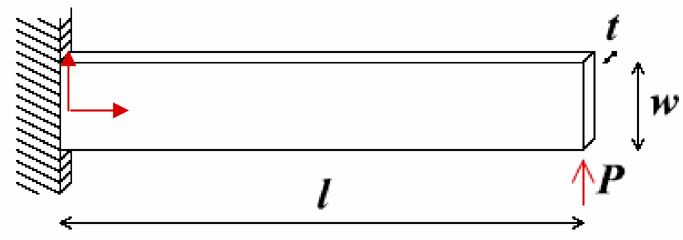
Define geometry




```
l        = 10e-6     -- Beam length
w        =  2e-6     -- Beam width
t        =  1e-6     -- Beam thickness
P        =  3e-6     -- Tip force
order    = 3         -- Order of element
dense    = 2.0e-6    -- Approximate element size
mesh:blocks2d({0,l},{-w/2,w/2},etype,order,dense)
```

- Define symbols for geometry and meshing parameters
- Mesh the region [0, l] X [-w/2, w/2]
  - Element size is determined by dense
  - Order of interpolation is determined by order

Boundary conditions



- Introduction
- **Basic analysis**
- Anchor loss
- EM systems

```
function bc_function(x,y)
  if x==0 then return 'uu', 0,  0; end
  if x==l and y==-w/2
          then return ' f',   P/t; end
end
mesh:set_bc(bc_function)
```

- Define nodal boundary conditions with a function evaluated at each node
- Function returns
  - A string to specify displacements or force BCs
  - Value of boundary displacements or forces
- Examples:
  - Returns nothing – no boundary conditions
  - ‘uu’, 0, 0 – zero displacements in x and y
  - ‘ u’, 0 – zero displacements in y only
  - ‘f’, 1 – unit force in x direction

Step 2. MATLAB script file

- •Introduction
- •Basic analysis
- •Anchor loss
- •EM systems

•[Initiate HiQLab]

```
init.m
```

•Load Lua input file

```
[mesh,L] = Mesh_load(filename);
```

•Solve problem

```
static_state(mesh);
```

•Visualize results

```
plotmesh(mesh);
plotfield2d(mesh);
```

- In MATLAB, init.m must be run first

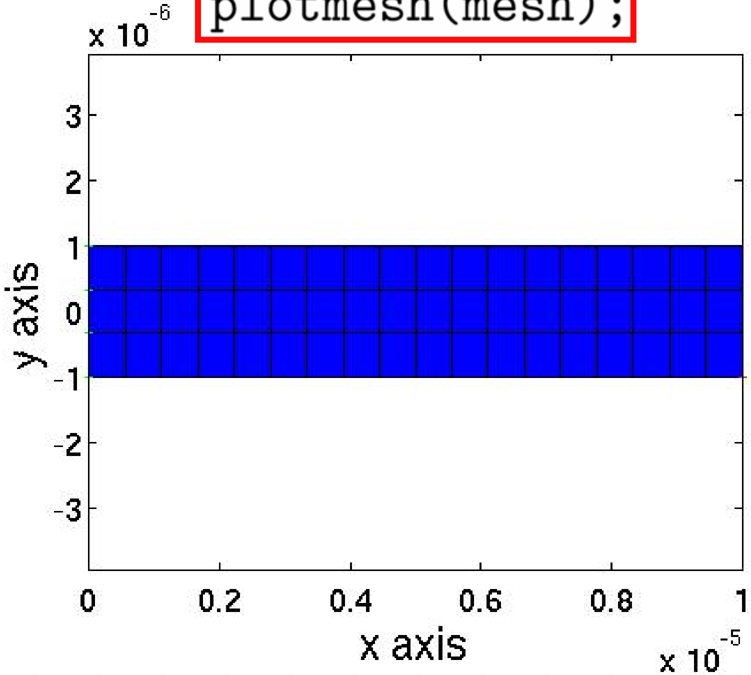
Solve and visualize (static state)

- Solve for static state `static_state(mesh);`

$$\mathbf{U} = \mathbf{K}^{-1}\mathbf{F}$$

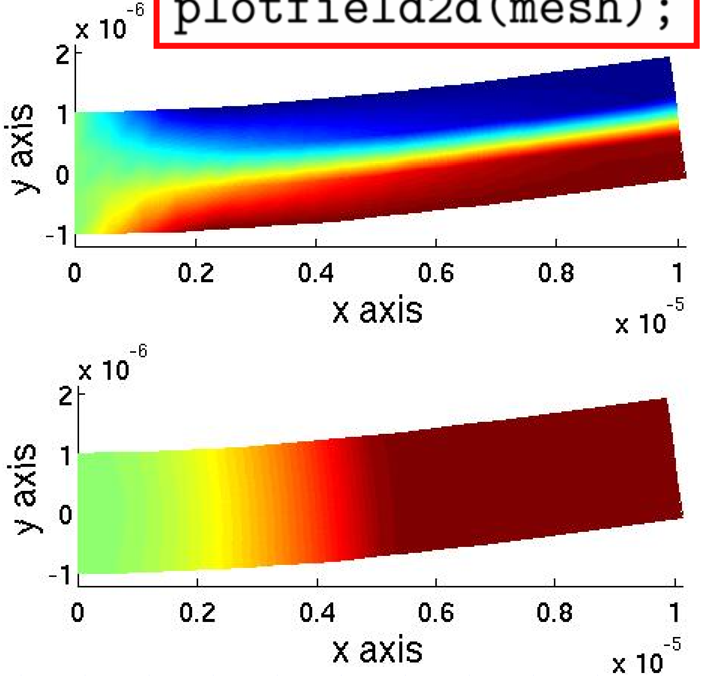
- Visualize result

`plotmesh(mesh);`



FE mesh

`plotfield2d(mesh);`



Top: x disp, Bot: y disp

Solve and visualize (modal analysis)

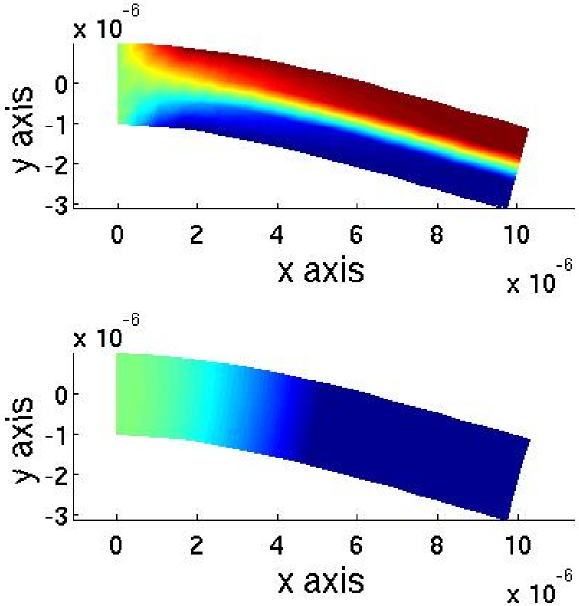
- Solve for modes

```
[V,w] = mechmode(mesh,w0,nev);
```

$$\left(\mathbf{K}-\omega^2\mathbf{M}\right)\mathbf{V}=0$$

- Visualize result

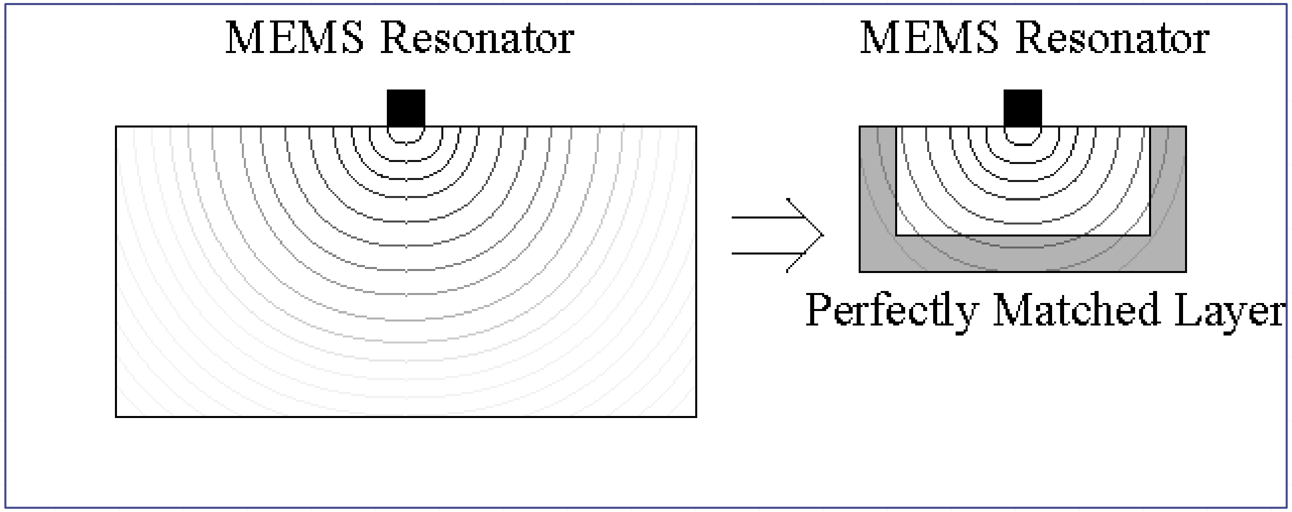
```
plot_mode(mesh,V,w);
```



Mode shape
(Top: x disp, Bot: y disp)

Anchor loss modeling

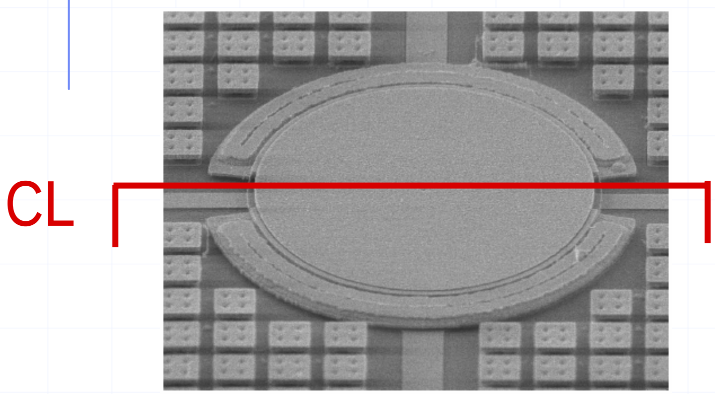
- Model infinite domain with finite domain using perfectly matched layers (PML)

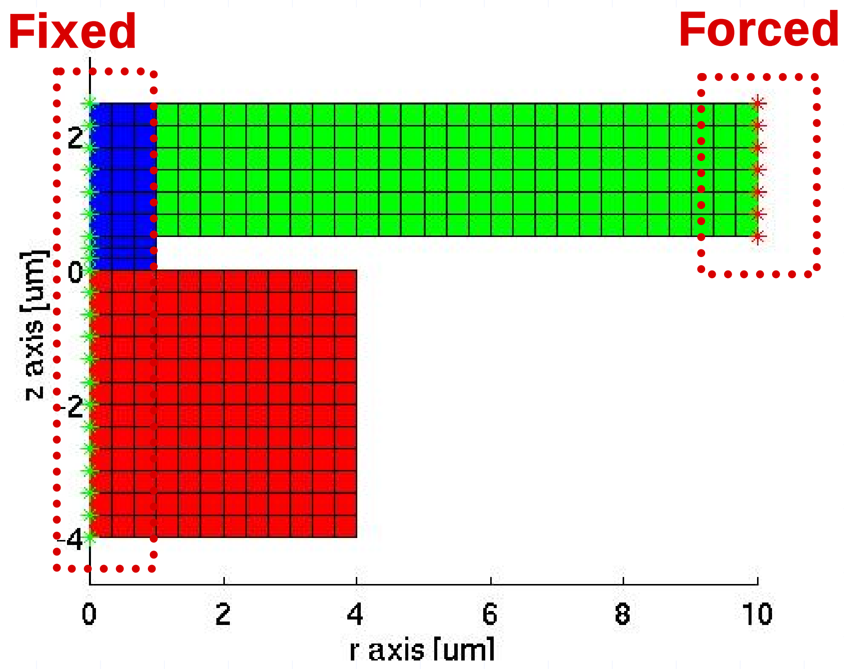


•Outgoing waves are absorbed with zero impedance mismatch at the boundaries.

Disk resonator

- Axis-symmetric analysis
- Based on the beam example construct file

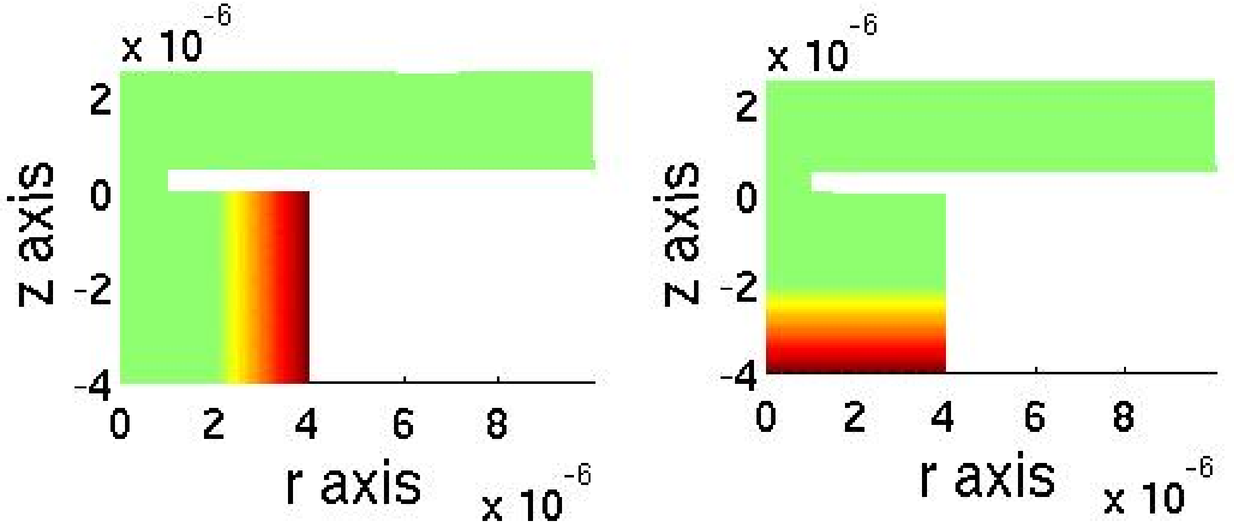




diskmesh.lua

Lua stretch function


```
function stretch_function(x,y)
    -- Compute and return stretch values sx,sy
end
elt:set_stretch(stretch_function)
```



Stretch function (Left: r, Right: z)

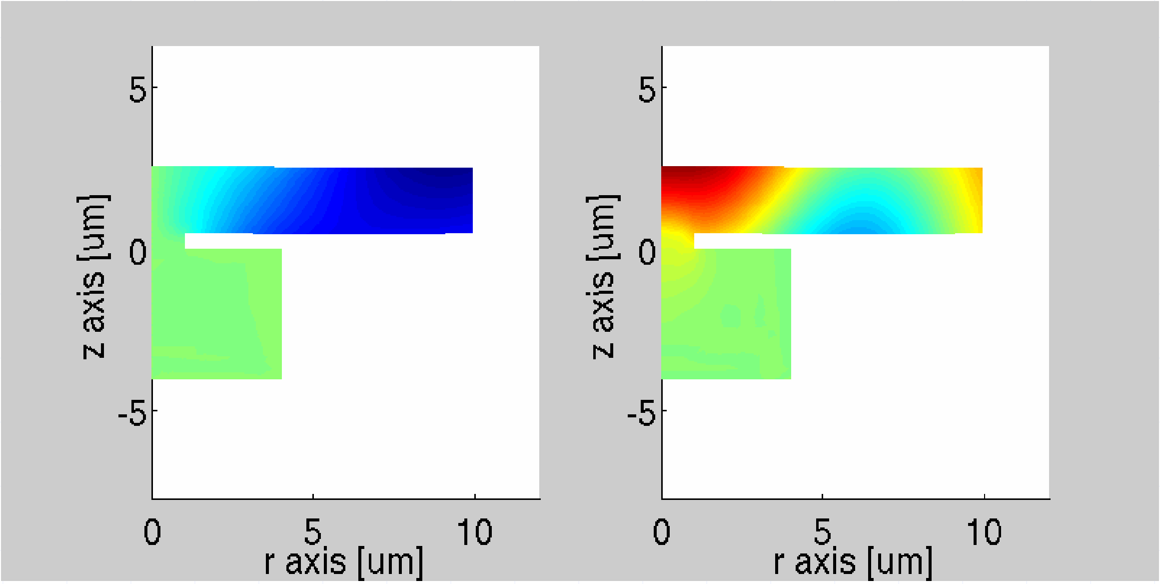
Modal analysis

•Introduction
•Basic analysis
•**Anchor loss**
•EM systems

```
[mesh,L] = Mesh_load('diskmesh.lua');
```

```
w0       = 2.889953e+08*2*pi;
nev      = 1;
[V,w,Q]  = mechmode(mesh,w0,nev);
```

```
plot_mode(mesh,V,w);
```



Left: x disp, Right: y disp

Forced analysis

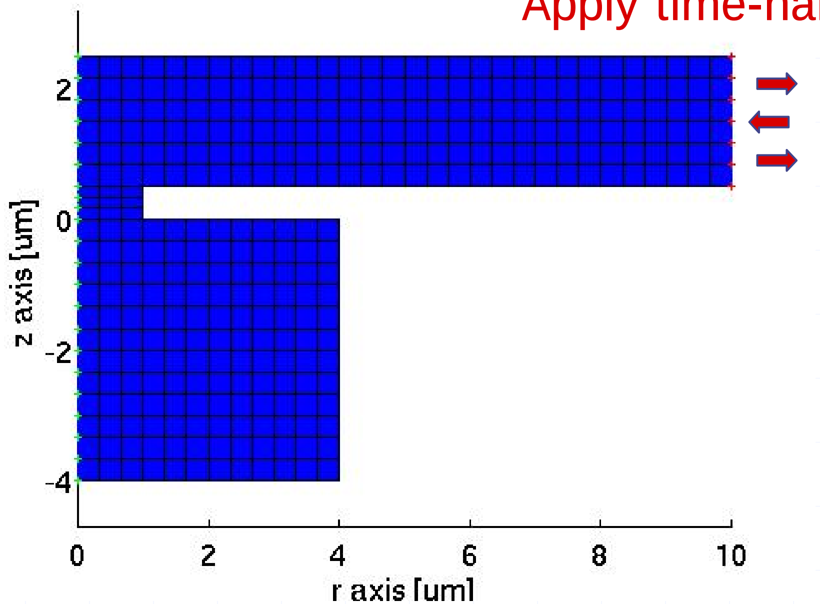
```
[mesh,L] = Mesh_load('diskmesh.lua');
```

```
w         = 2.889953e+08*2*pi;
drive_pat = Mesh_get_drive_f(mesh,'bode_drive_function');
forced_state(mesh,drive_pat,w);
```

```
plotcycle2d(mesh,scale,opt);
```

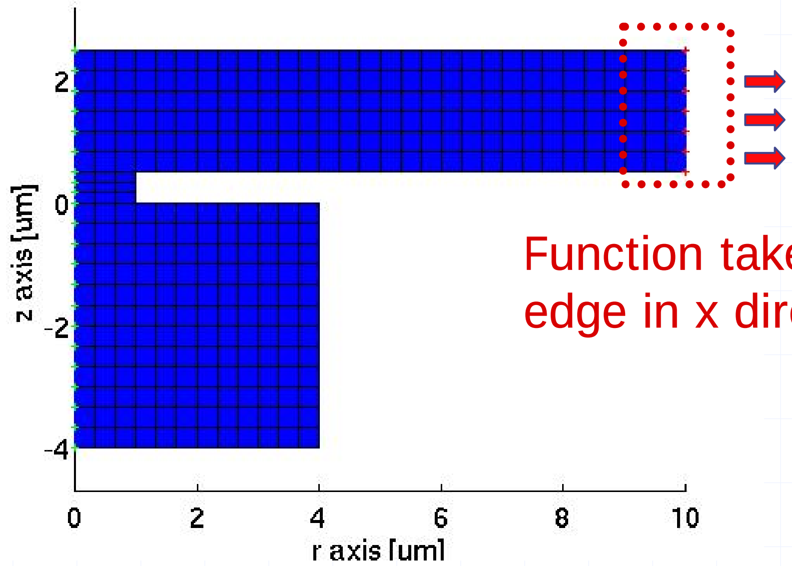
- Introduction
- Basic analysis
- **Anchor loss**
- EM systems

Apply time-harmonic force



Lua drive and sense functions

```
function bode_drive_function(x,y)
  if mesheq(x,rdisk) then return 'f', 1; end
end
```

- Functions evaluated at each node take same form as bc functions to construct vector
- Sense function evaluated in the same form

Solve and visualize (forced analysis)

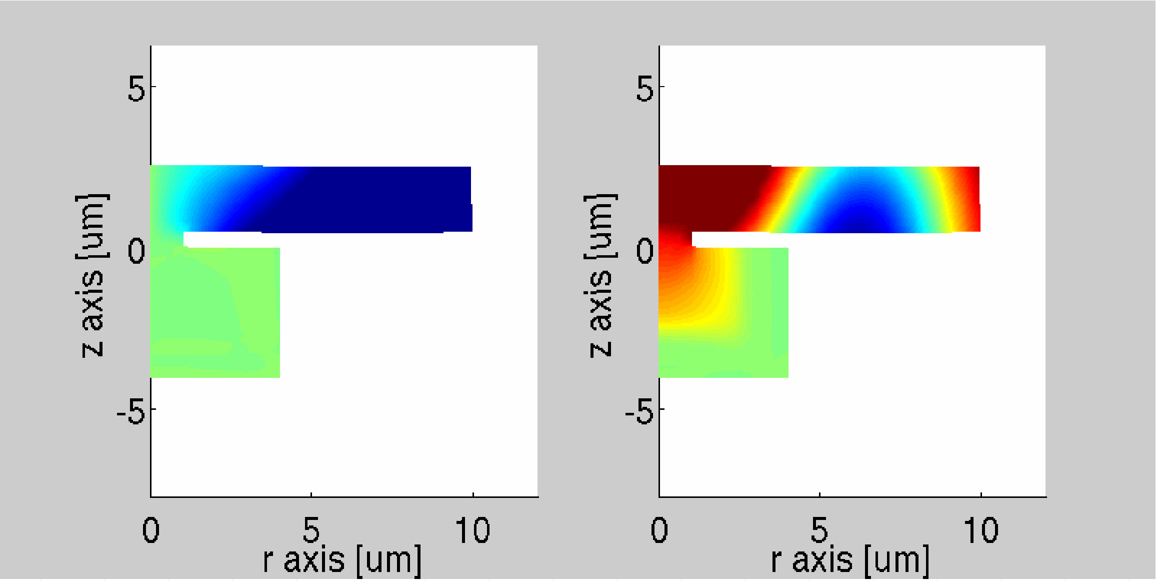
- Solve for forced response

```
forced_state(mesh,drive_pat,w);
```

$$\mathbf{U} = \left(\mathbf{K} - \omega^2\mathbf{M}\right)^{-1}\mathbf{F}$$

- Visualize results

```
plotcycle2d(mesh,scale,opt);
```



Left: x disp, Right: y disp

Transfer function

```
[mesh, L] = Mesh_load('diskmesh.lua',param);
```

```
wc          = 2.889953e+08*2*pi;
drive_pat = Mesh_get_drive_f(mesh,'bode_drive_function');
sense_pat = Mesh_get_sense_u(mesh,'bode_sense_function');
[H,freq] = second_order_bode(mesh,wc,drive_pat,sense_pat);
```

```
plot_bode(freq,H);
```


- Computes transfer `second_order_bode`

$$H(\omega) = \mathbf{L}^T \left(\mathbf{K} - \omega^2 \mathbf{M}\right)^{-1} \mathbf{B}$$

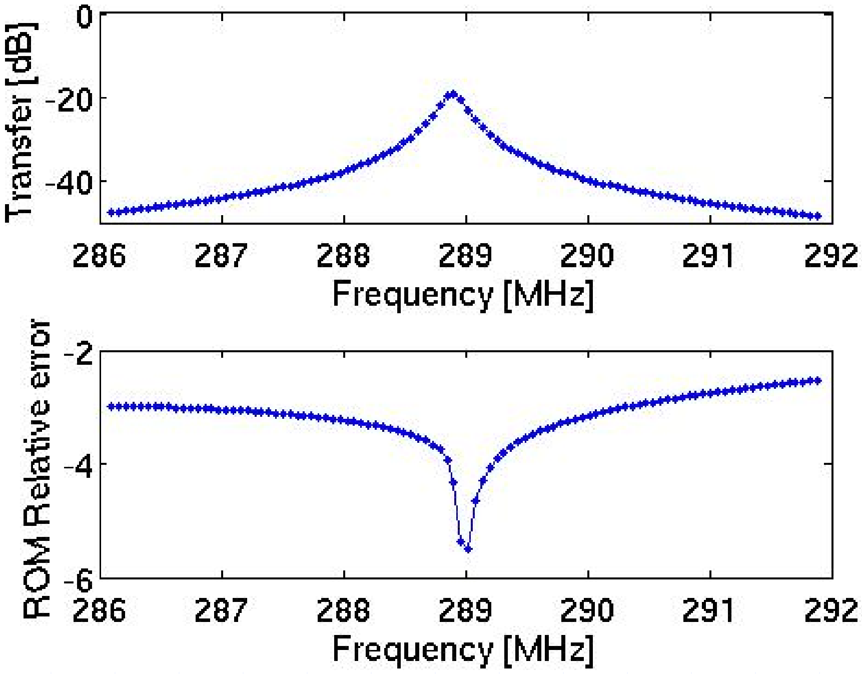
Reduced order modeling

- Introduction
- Basic analysis
- **Anchor loss**
- EM systems

```
opt.kmax = 2;
[H,freq] = second_order_bode(...,opt);
```

- Specify size of reduced order model

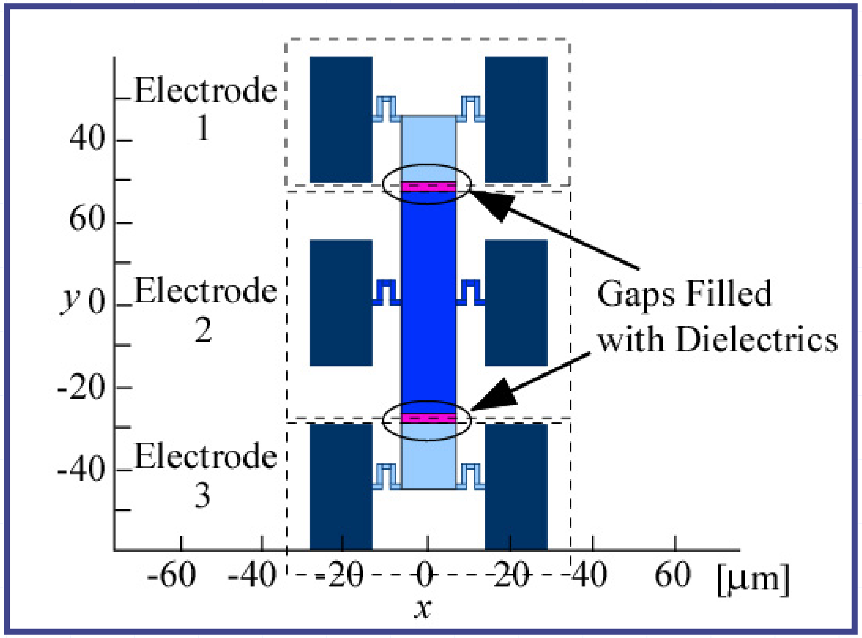
$$H(\omega) \approx \mathbf{L}_k^T \left(\mathbf{K}_k - \omega^2 \mathbf{M}_k\right)^{-1} \mathbf{B}_k$$



Dielectric filled gap resonators

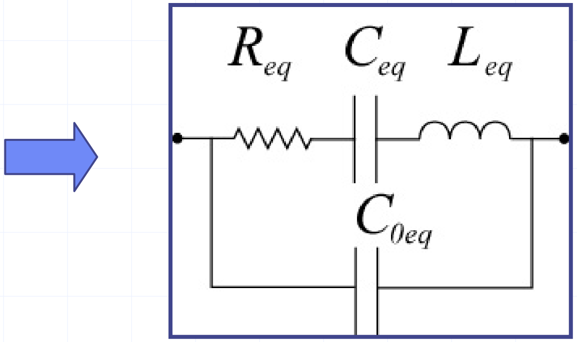
- Increase transduction efficiency by filling electrostatic air gap with dielectrics

•Introduction
•Basic analysis
•Anchor loss
•EM systems



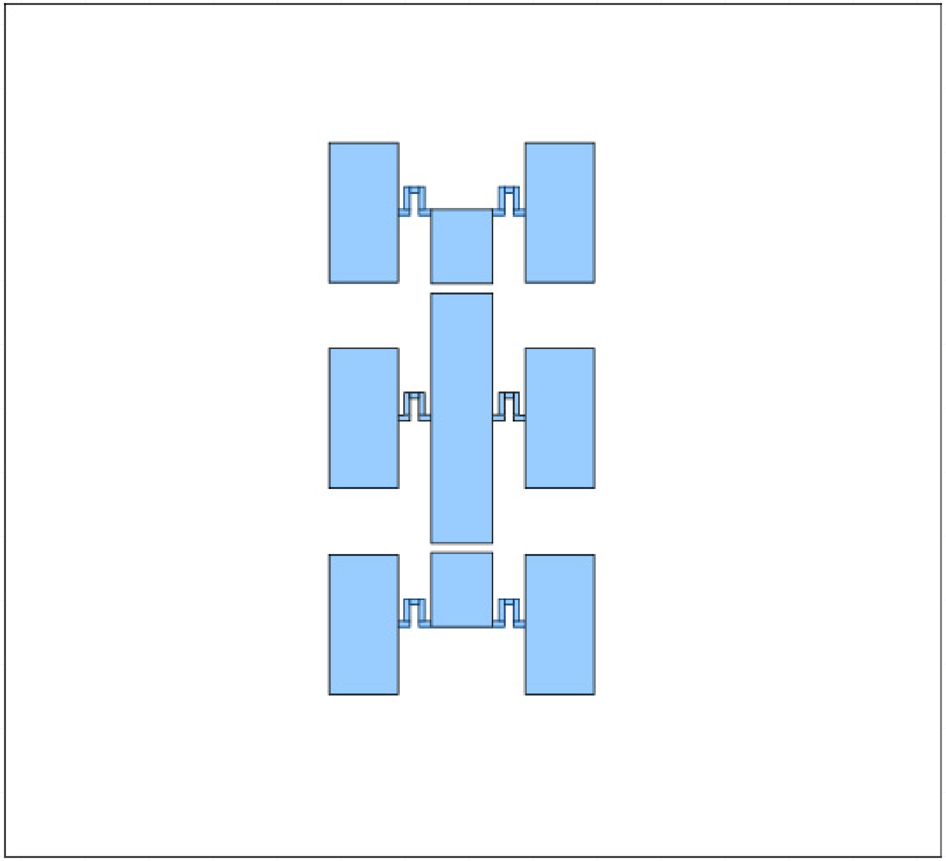
Schematic of resonator

•Extract equivalent LRCC parameters!



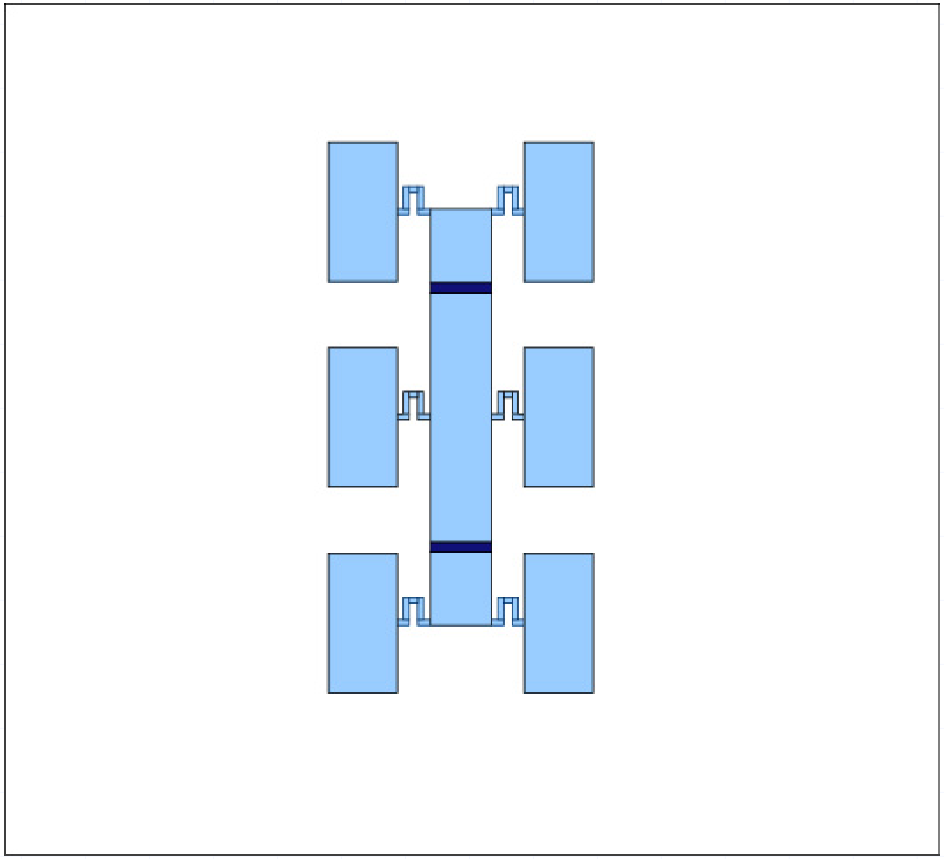
Lua input file

```
etype_c  = make_material_e(gap_mat,...)
mesh:blocks2d
```



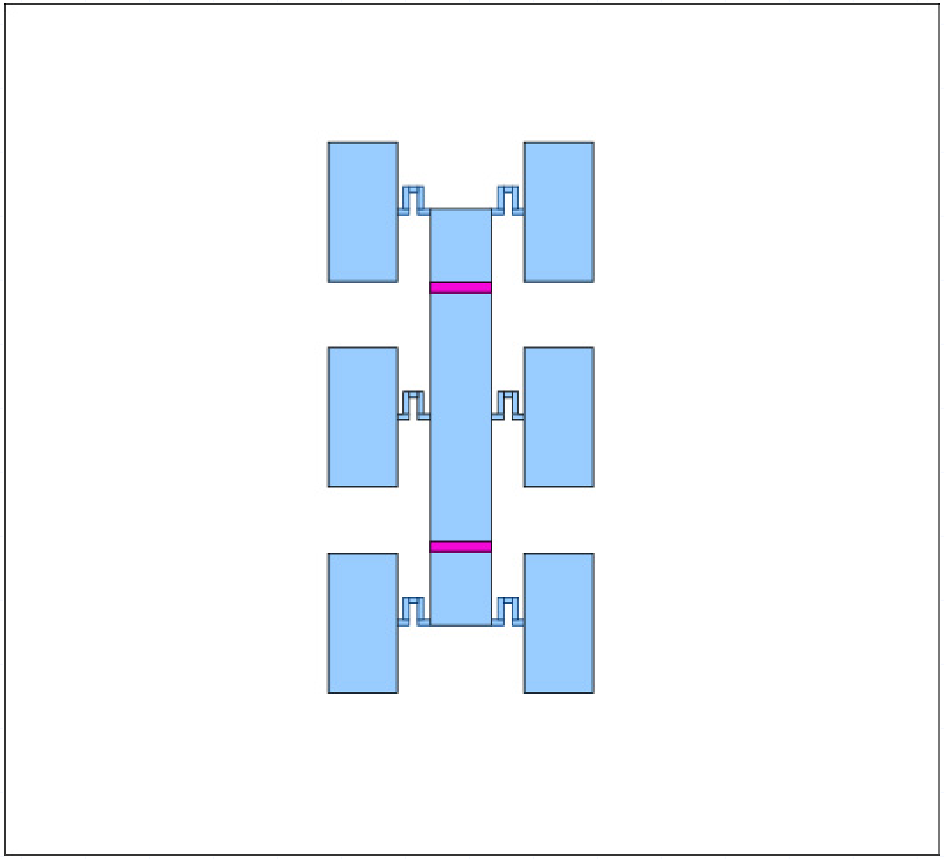
Lua input file

```
etype_g  = make_material_e(gap_mat,...)
mesh:blocks2d
```



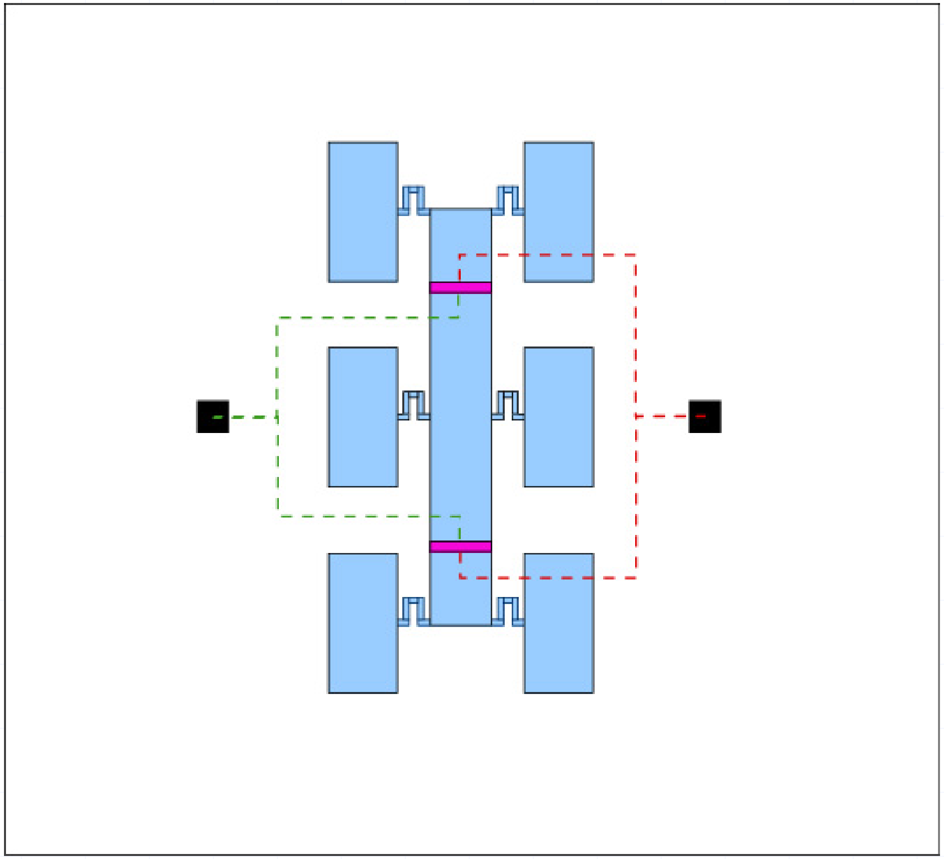
Lua input file

```
etype_em = make_material_couple_em(eps,...)
mesh:blocks2d
```



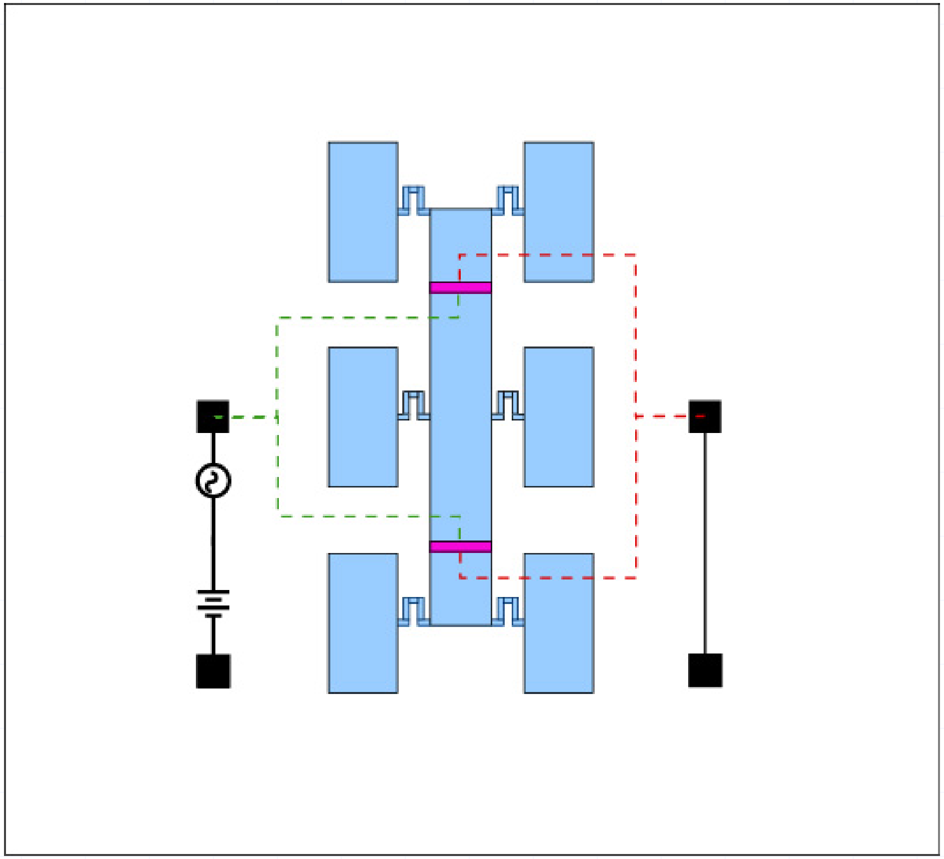
Lua input file

`add_electrode`



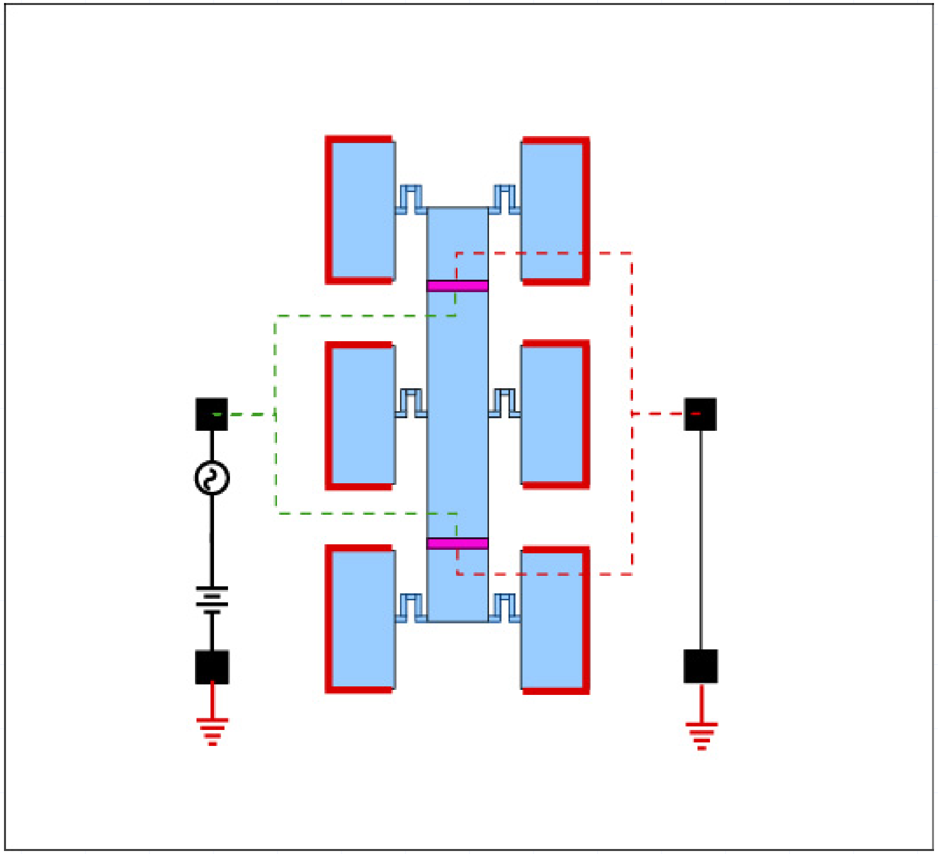
Lua input file

```
wire    = make_material_circuit_wire()
mesh:add_element
```



Lua input file

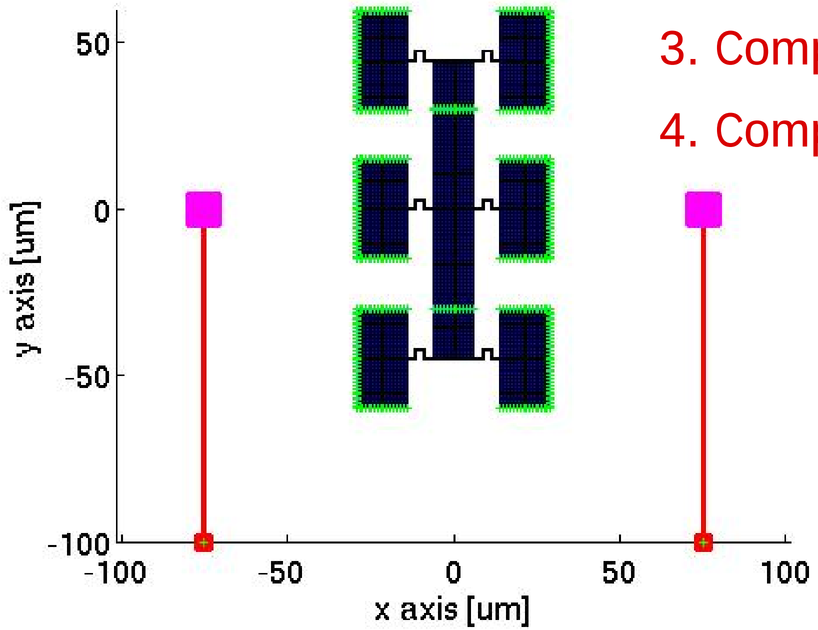
```
function mech_bc(x,y)
function ground_voltage(x,y)
mesh:set_bc{mech_bc,ground_voltage}
```



Analysis steps

- Steps to extraction of parameters

1. DC analysis (Find operating point)
2. Modal analysis (Find desired mode)
3. Compute equivalent LRCC parameters
4. Compute admittance (Check results)



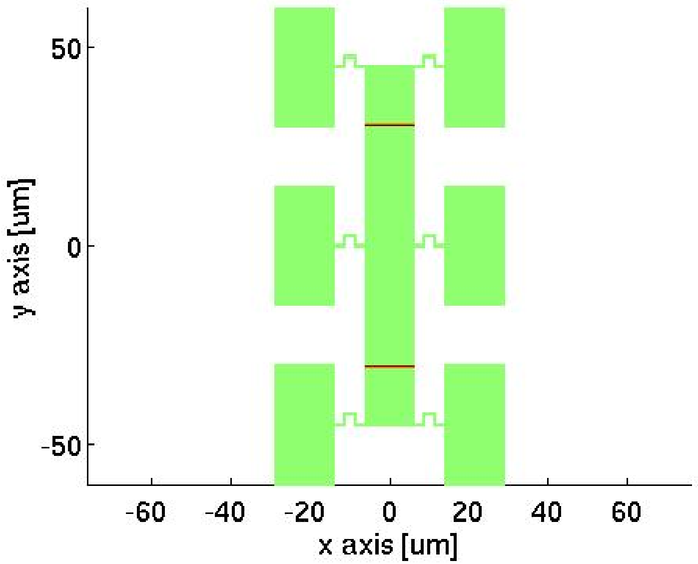
- Plane stress analysis
- Electromechanical coupling is nonlinear

DC analysis (static state)


```
[mesh, L] = Mesh_load('dielectric_drive.lua');
```

```
sopt.nonlinear = 'NR';
static_state(mesh,sopt);
```

```
plotfield2d(mesh);
```



Color represents potential field

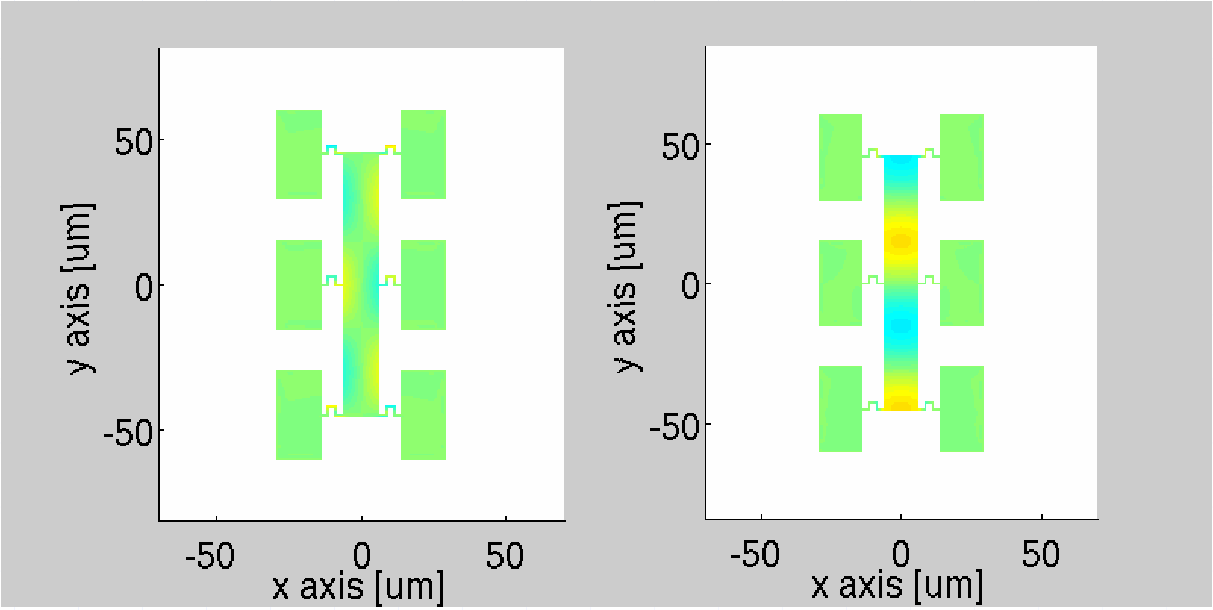
Modal analysis

- Introduction
- Basic analysis
- Anchor loss
- **EM systems**

```
[mesh, L] = Mesh_load('dielectric_drive.lua');
```

```
sopt.nonlinear = 'NR';
static_state(mesh,sopt);
[V,w,Q] = emcmode(mesh, w0, nev, param);
```

```
plot_mode(mesh,V,w);
```



Resonant mode at 140.02[MHz] **(Left: x disp, Right: y disp)**

Equivalent LRCC parameters

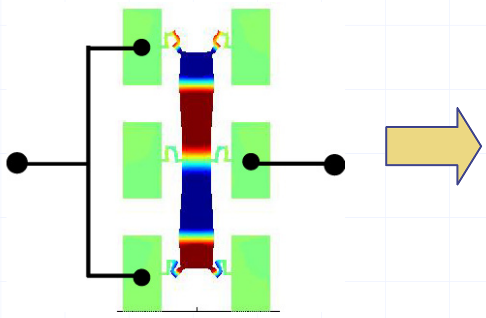
Enables simple calculations.

```
[mesh, L] = Mesh_load('dielectric_drive.lua');
```

```
sopt.nonlinear = 'NR';
static_state(mesh,sopt);
equiv_LRCC(mesh,w0,param);
```

- Introduction
- Basic analysis
- Anchor loss
- **EM systems**

Resonator at specific mode



Device

$L_{eq}=$ 0.179[H] $V_{dc}=$ 100 [V]

$R_{eq}=$ 536[Ω] k = 160 ε_0

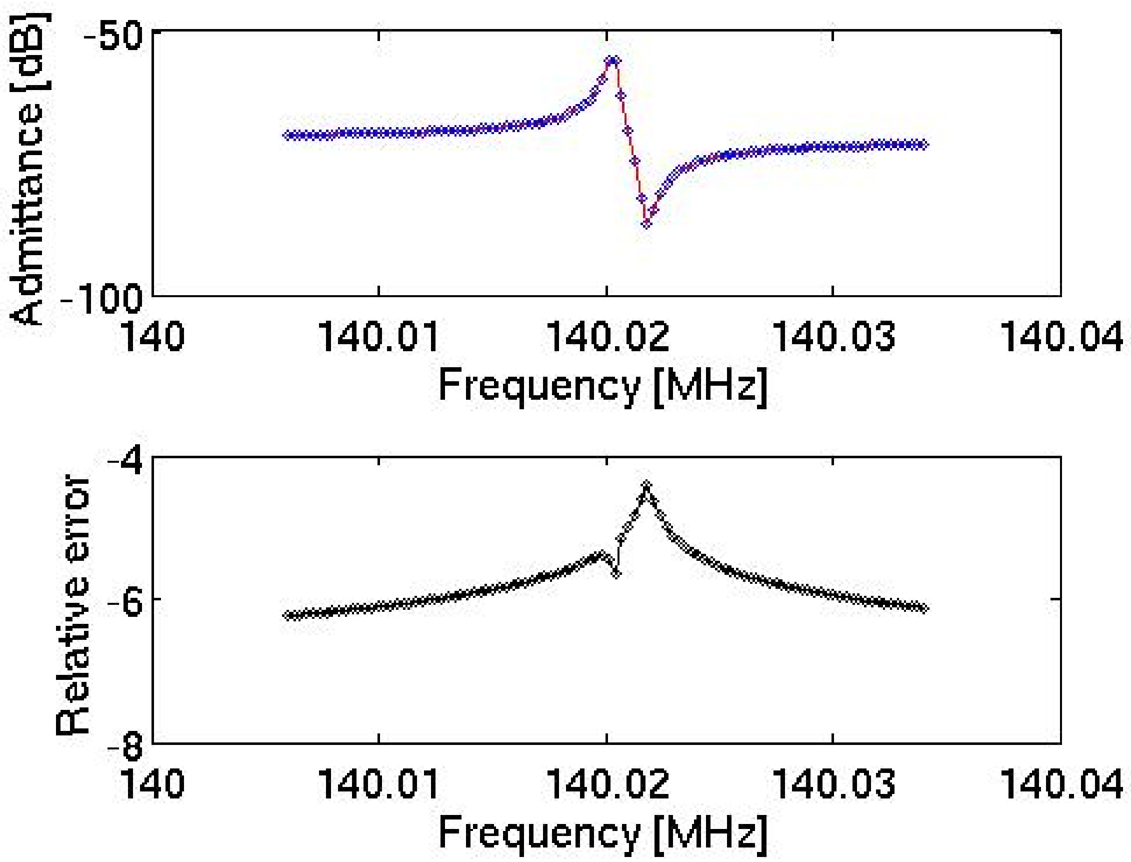
$C_{eq}=$ 7.20[aF] $f_c=$ 140.20[MHz]

$C_0=$ 0.341[pF]

Compute admittance to check results

- Admittance plot and relative accuracy

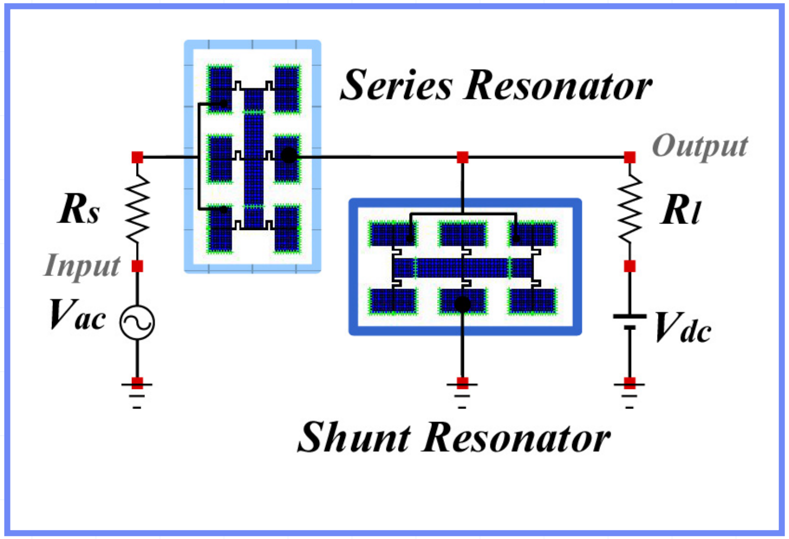
•Introduction
•Basic analysis
•Anchor loss
•**EM systems**

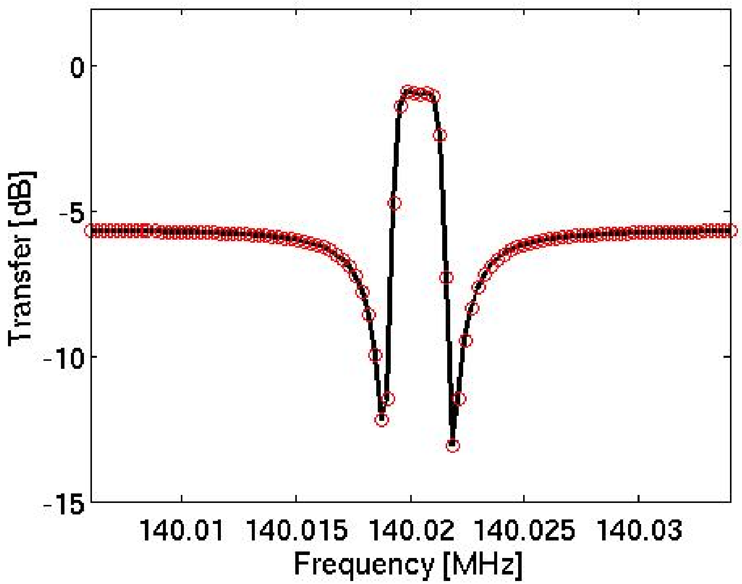


Ladder filter

◆ Electrically coupled resonators

- •Introduction
- •Basic analysis
- •Anchor loss
- •EM systems





Schematic of ladder configuration (10 kΩ Termination) Transmission plot

Summary

- HiQLab: Resonant MEMS Simulator
  - Simulates Anchor loss / Thermoelastic damping (Simulations have been verified with experiments. "Anchor Loss Simulation in Resonators". MEMS'05).
  - Efficient algorithms for *Q* evaluation which take advantage of underlying structure: Anchor loss / Thermoelastic damping
  - First dielectric drive simulation
- http://www.cs.berkeley.edu/~dbindel/hiqlab
  - Software is freely available
  - Tutorial slides and relevant papers
- Ongoing and future work
  - Resonator optimization for target bandpass filter design
  - Parallel algorithms for solution of larger problems
  - Modeling of other sources of damping

Acknowledgements

- PIs: Prof. Sanjay Govindjee (Civil Engineering)
  Prof. James Demmel (CS and Math)
  Prof. Roger Howe (Stanford)

- Post doctoral students:
  Dr. Emmanuel Quevy(Electrical Engineering)

- Graduate students:
  David Bindel (CS)
  Wei He (Civil Engineering)
  Members of the SUGAR group