SUGAR: Simulating MEMS

David Bindel

UC Berkeley, CS Division

Outline

- SUGAR group and research goals
- Microsystems overview
- Simulation methodology
- Software architecture
- Research directions
- Concluding thoughts

SUGAR: System simulation

Goal: “Be SPICE to the MEMS world”

- Fast enough for early design stages
- Simple enough to attract users
- Capable of simulating interesting coupled physics

SUGAR group

Faculty	Grad students
A. Agogino	D. Bindel
Z. Bai	J.V. Clark
J. Demmel	D. Garmire
S. Govindjee	R. Kamalian
K.S.J. Pister	S. Lakshmin
	J. Nie

Microsystem applications

Multi-domain micromachines made with IC fab technology.

- Inertial sensors: accelerometers, gyros
- Fluidics: ink jet printers, biolab chips
- Optics: optical switches, projectors
- Pressure sensors: industrial, medical, auto
- RF devices: cell phone, radar components
- Other: microrelays, sensors, disk heads

(List from “Microsystem Design” by S. Senturia)

Microscale physics

- Surface area / volume ratio is large (compared to most macroscopic structures)
- Gravitational forces negligible
- Electrostatic forces substantial
- Frictional forces (particularly “stiction”) important
- Air behaves like a very viscous fluid ($Re << 1$)
- Still classical (vs. nanosystems)

Simulation approaches

- What? Systems vs devices vs TCAD.
- Why? Brainstorming, tuning, design verification, ...
- How?
  - Solve continuum equations (finite elements, finite differences, boundary elements)
  - Solve simplified equations of beam and plate theory
  - Solve network equations
  - These approaches are not mutually exclusive!

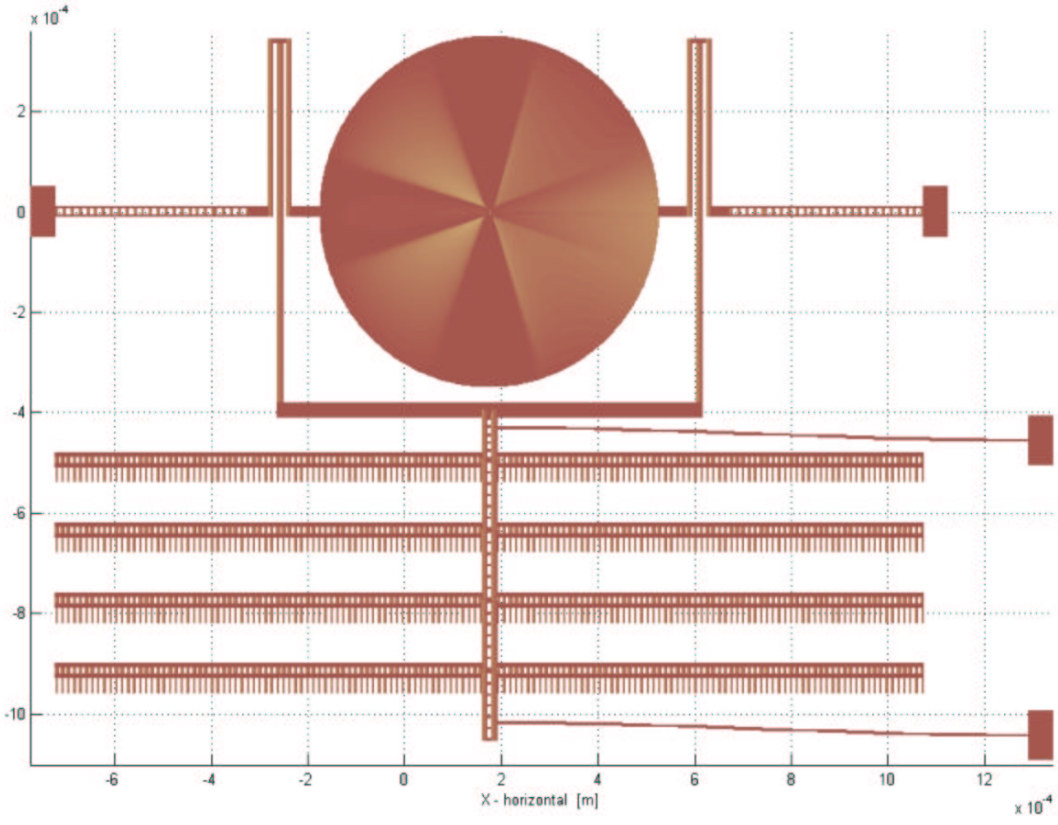
Modeling approach

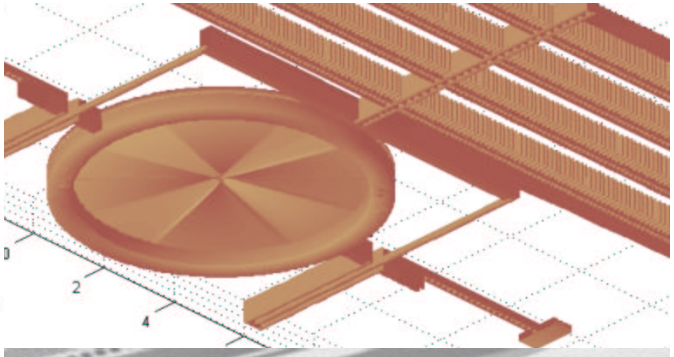
- Similar to circuit and structure simulators; systems modelers
- Add multiphysics: electro-thermo-mechanical coupling, damping interactions

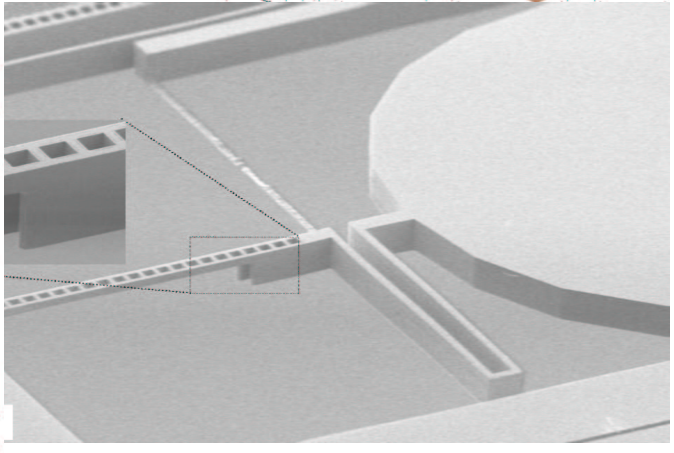
Use balance laws + element models:

- LRC elements, sources
- Beams, rigid connectors, plates, point masses, forces
- Gaps and combs

What can we model?







Of PDEs and ODEs

Get ODEs from spatial discretization of PDEs. Simple example is a Hookean rod:

$$\rho A u_{tt} + EAu_{xx} = F(x,t) \text{ and } u(0,t) = 0, u(x,0) = 0$$

Rewrite PDE in "weak form": for any weight w,

$$\int_0^L w(x,t) \cdot EAu_{xx}(x,t)\, dx = \int_0^L w \cdot F(x,t)\, dx$$

Assume $u(x,t) = \hat{u}(t) \cdot x/L$ and $w(x,t) = \alpha x/L$ to get

$$M\hat{u}_{tt}(t) + K\hat{u}(t) = \hat{F}(t)$$

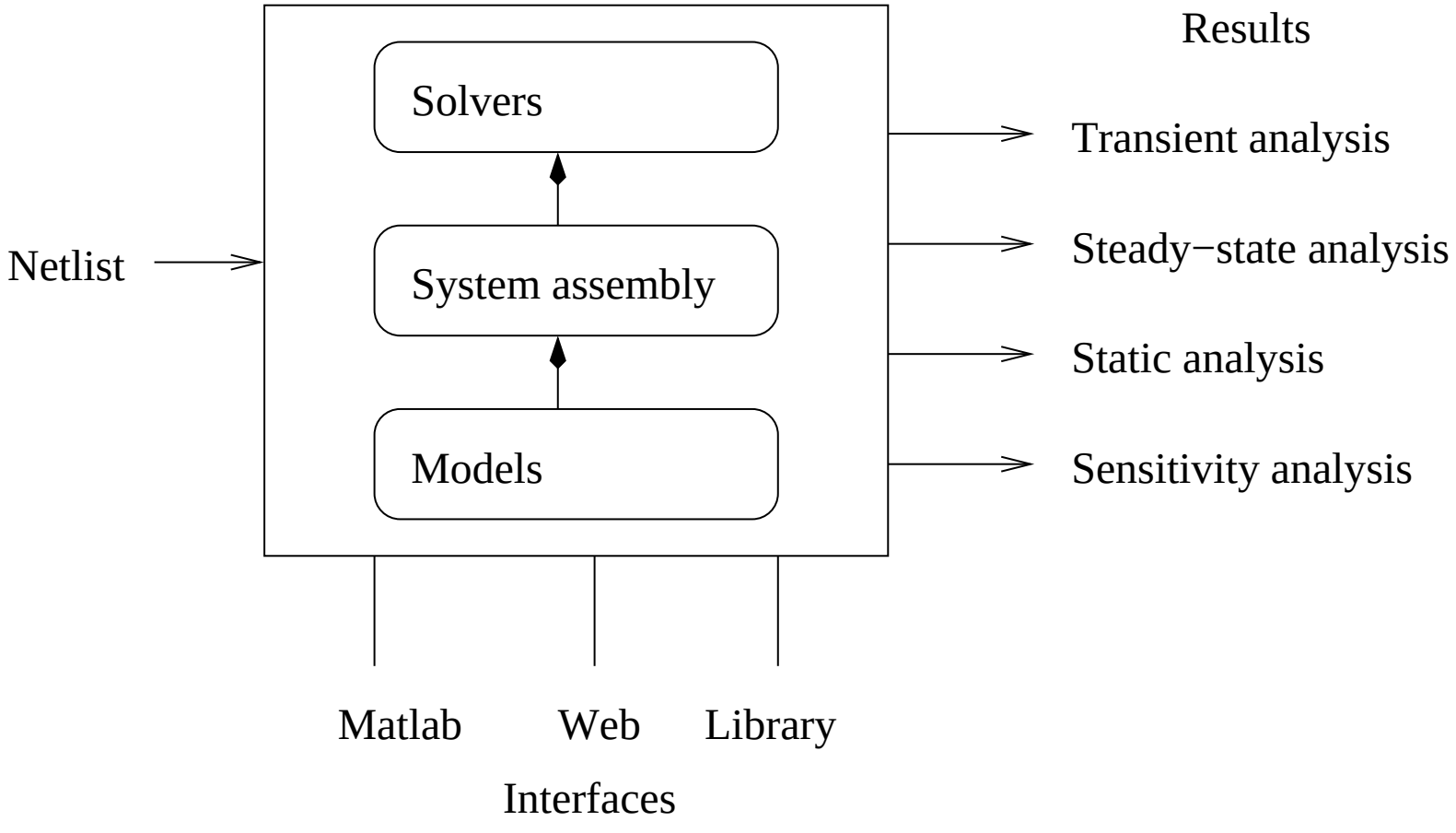
Balance laws

Local element relations are *assembled* based on global balance principles:

- Circuits: KCL and KVL
- Mechanics: force balance and continuity
- Similar principles in other domains

Software structure above the element level is very generic.

SUGAR architecture



Device description language

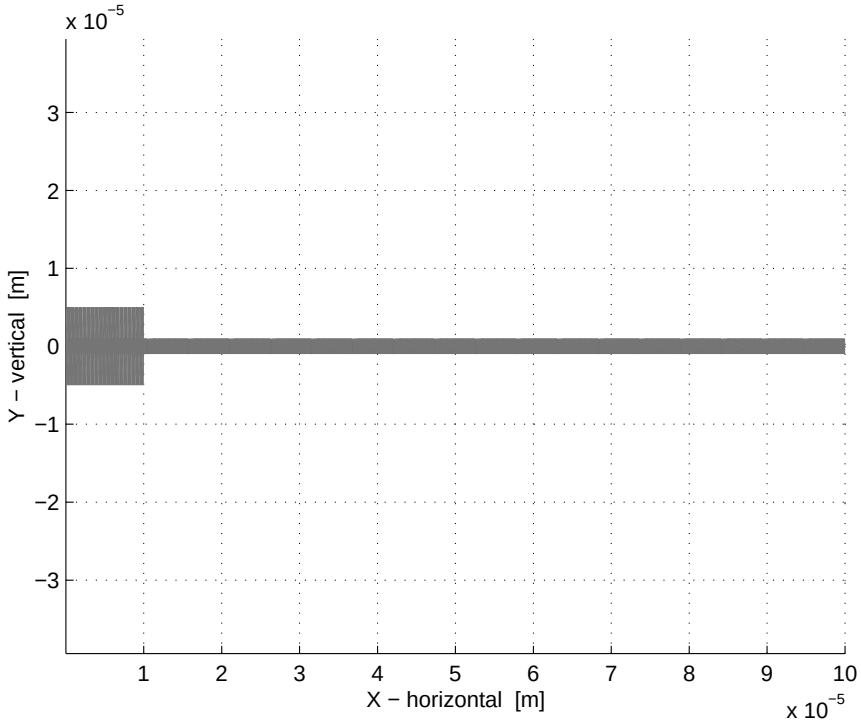
- Based on the Lua scripting language
- Basic ingredients: nodes, materials, elements
- Node positions can be explicit or implicitly determined
- User-defined components (subnets) with named parameters for heirarchical design
- Accept externally defined parameters

Hello world: a cantilever

```
use 'mumps.net'
use 'stdlib.net'

anchor {node 'substrate', p1;
        l=10u,  w=10u}
beam3d {node 'substrate', node 'tip', p1;
        l=100u, w=2u}
f3d    {node 'tip'; F=2u, oz = 90}
```

Hello world: a cantilever



Element routines

- Standard element interfaces for both Matlab and C/C++
- Unused element cases have reasonable defaults
- Basic tasks:
  - Initialize element data
  - Output element data
  - Display element
  - Declare node and branch variables of interest
  - Compute element forces/currents and tangents
  - Declare boundary conditions

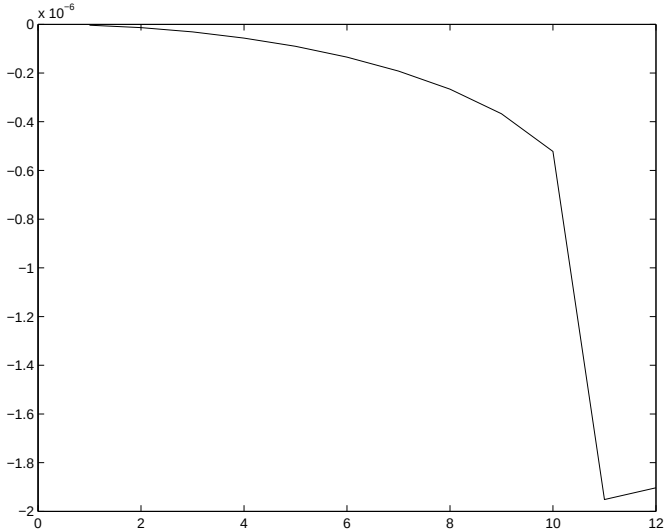
Global analyses

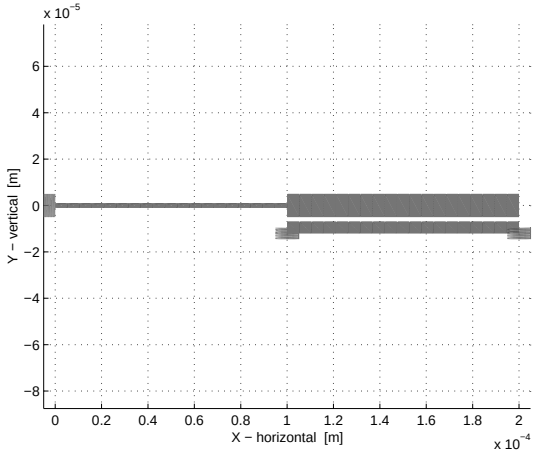
Standard analyses on assembled system:

- Transient response
- Static equilibrium (DC analysis)
- Steady-state small-signal response
- Mode shape determination

Treat assembled systems with standard solver technology (e.g. SuperLU, LAPACK, and ARPACK). Can also assemble system in Matlab and use Matlab analysis routines.

Using the Matlab interface

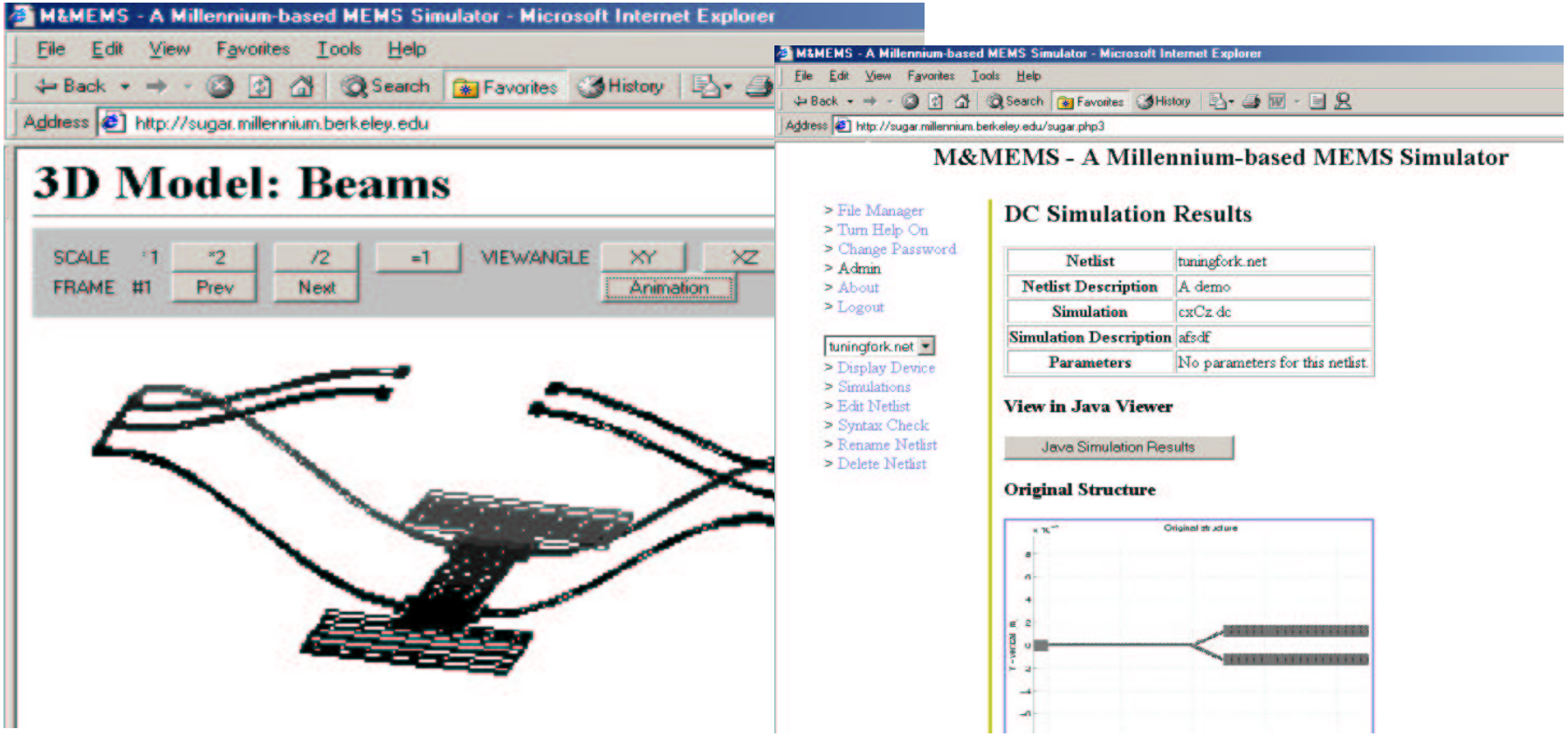





```
dq = [];
for k=1:12
  param.V = k;
  net = cho_load('beamgap2b.net', param);
  dq = cho_dc(net, dq);
  tip(k) = cho_dq_view(dq, net, 'c', 'y');
end
```

M&MEMS: SUGAR on the Web

http://sugar.millennium.berkeley.edu/



Notes on overall architecture

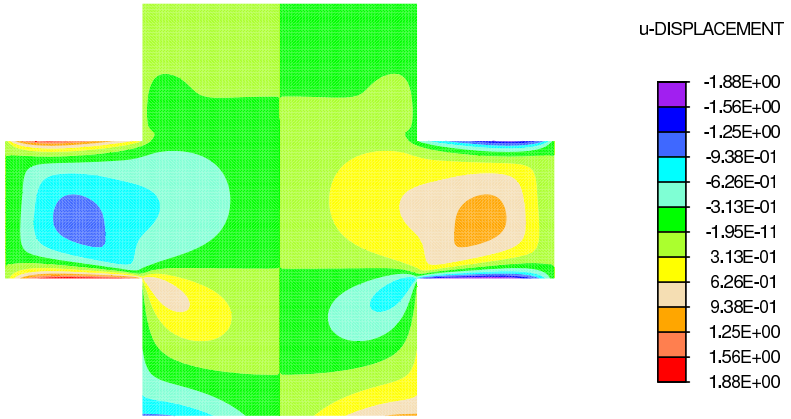
"Lesser artists borrow; great artists steal."
– Stravinsky

- Trying to integrate state of the art from several fields
- Integration a challenge, even with modular design
- Continues to evolve: 3.5 is C++ based with automatically generated interfaces for mixed language management

Research directions

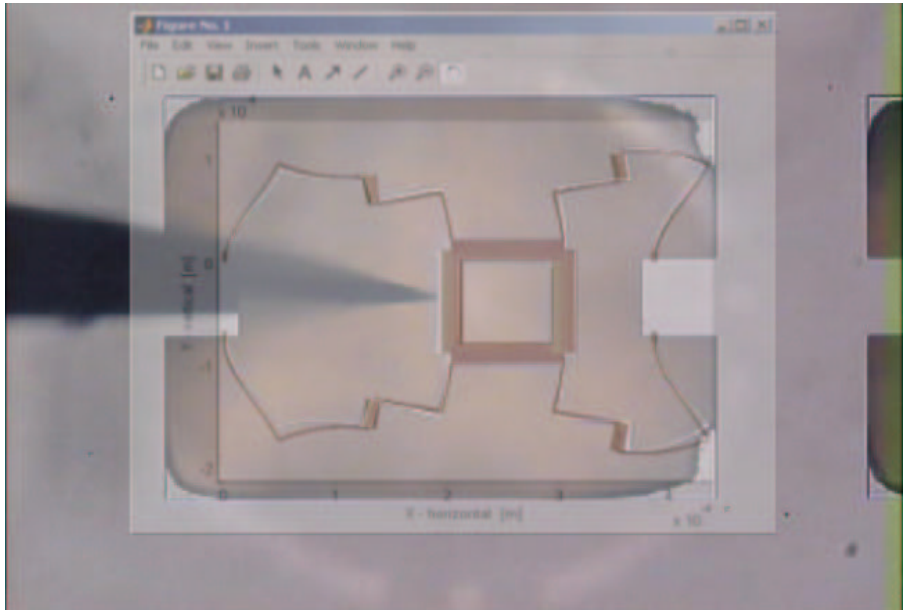
- Software design, integration, and test
- Model development
- Measurement feedback
- Design synthesis, optimization, and sensitivity
- Model reduction
- Parameter-dependent eigencomputations + bifurcation analysis
- What designers need!

Model development



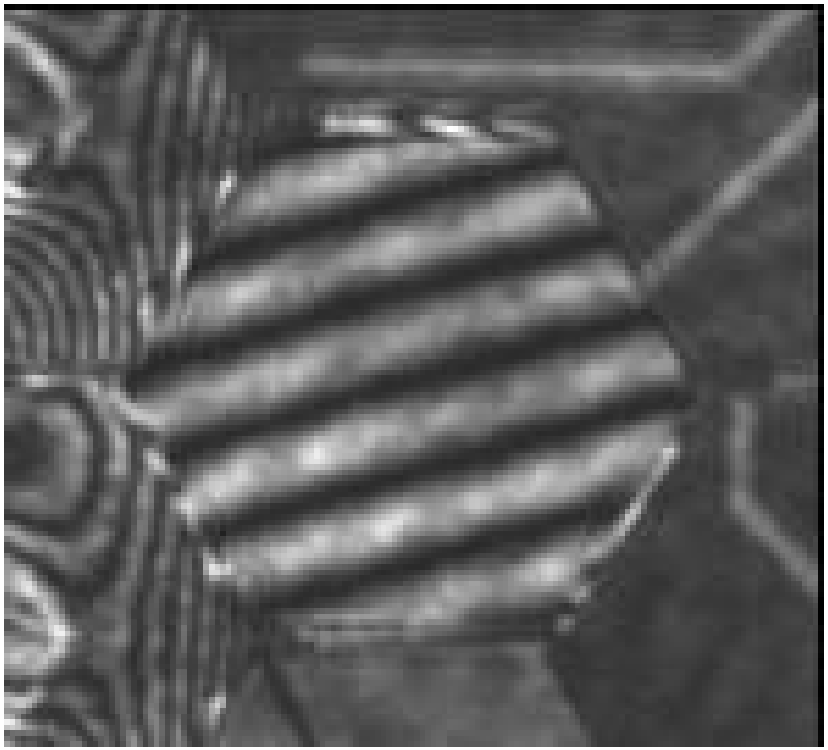
- Incorporation of anisotropic effects
- Improved damping models
- Continued development of multiphysics approximations
- Microfluidic modeling
- Extraction from more detailed simulation

Measurement feedback



- Integrate measurements from probe stations, interferometric systems, Doppler vibrometers
- Compare to simulation in real time to guide experiments
- Use results to extract material properties or tune models

Measurement feedback



Movie of interferometric recording

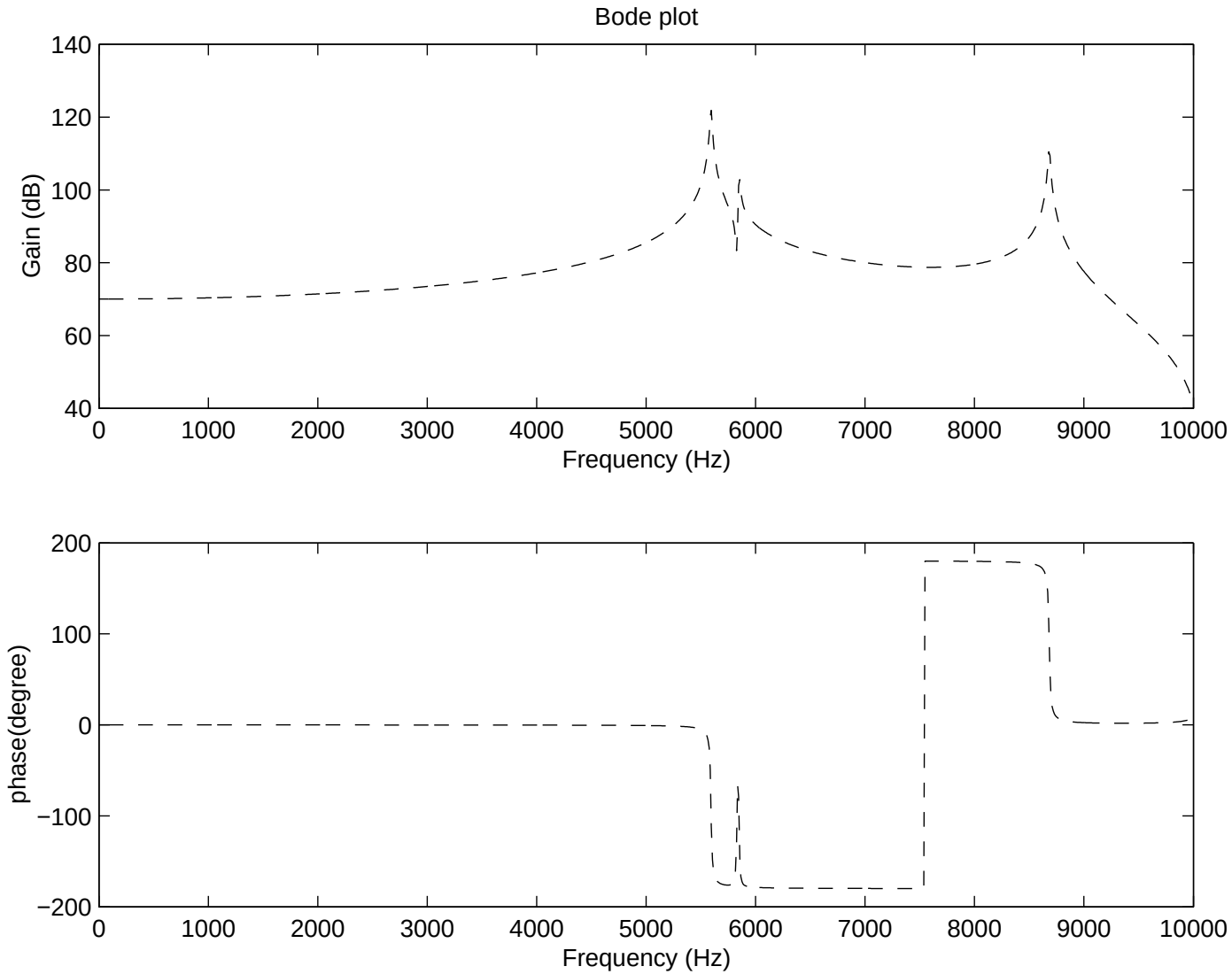
Design optimization and synthesis

- Genetic algorithms and simulated annealing to optimize topology of designs
- Uses SUGAR as a black box to evaluate objective
- General issues:
  - Design rule checking and model feasibility
  - Identifying model limits
  - Optimization of continuous parameters
  - Design sensitivity

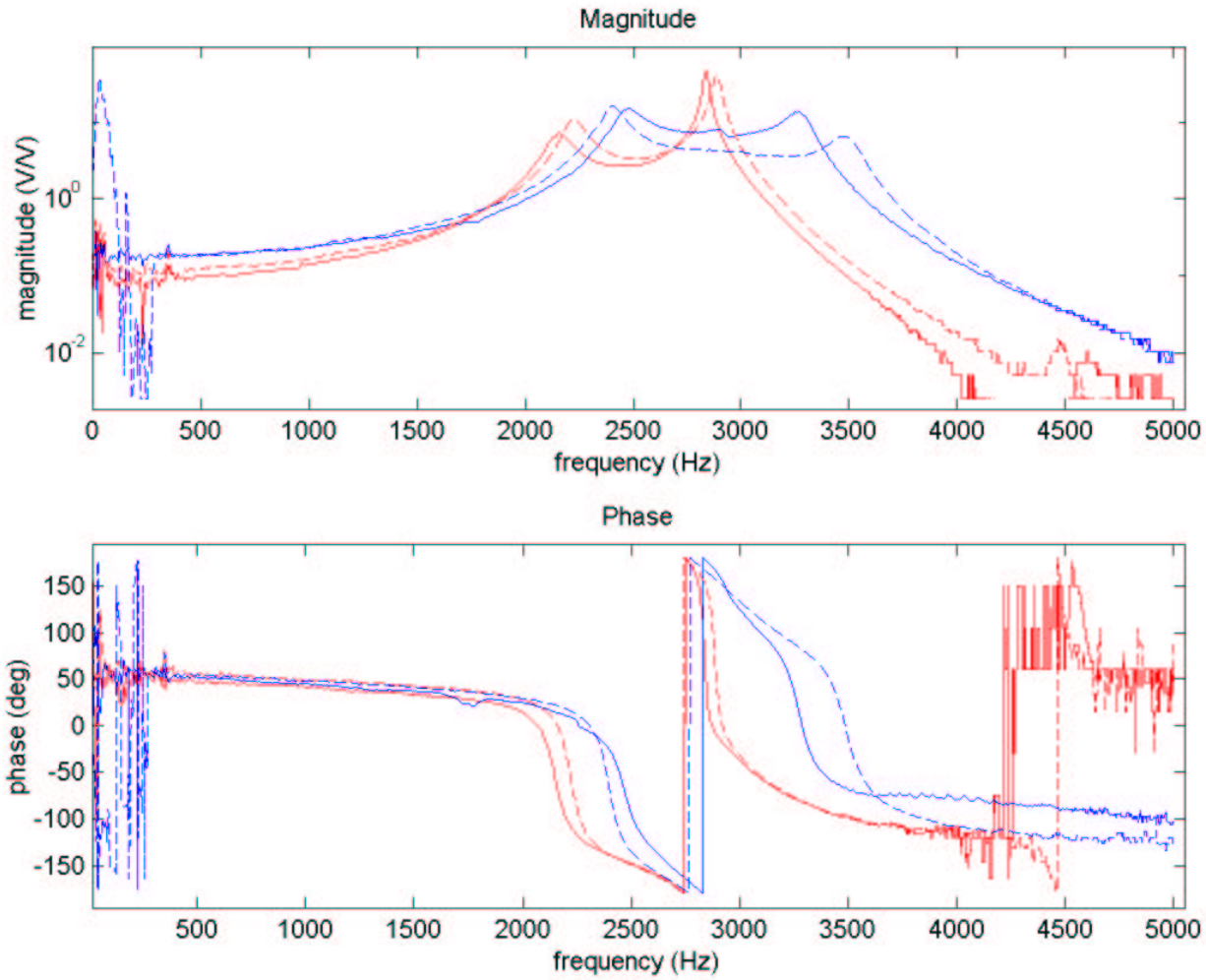
Model reduction

- Used to create macromodels from detailed device simulations
- Also used to reduce large system models
- Need to handle nonlinear effects

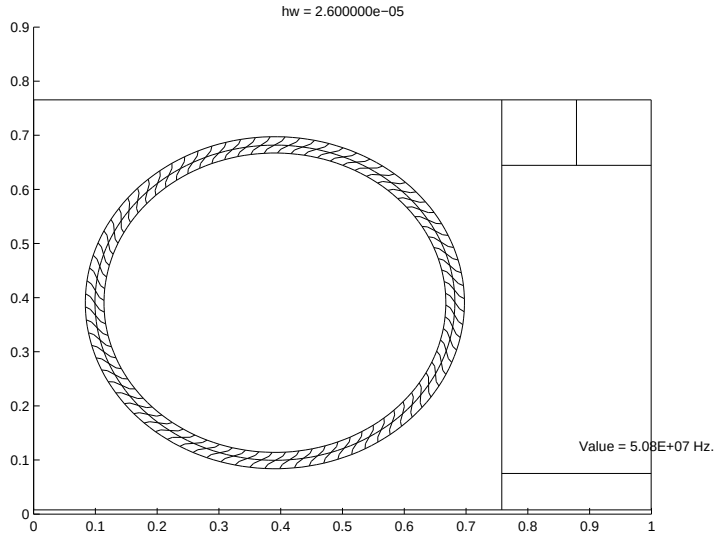
Model reduction

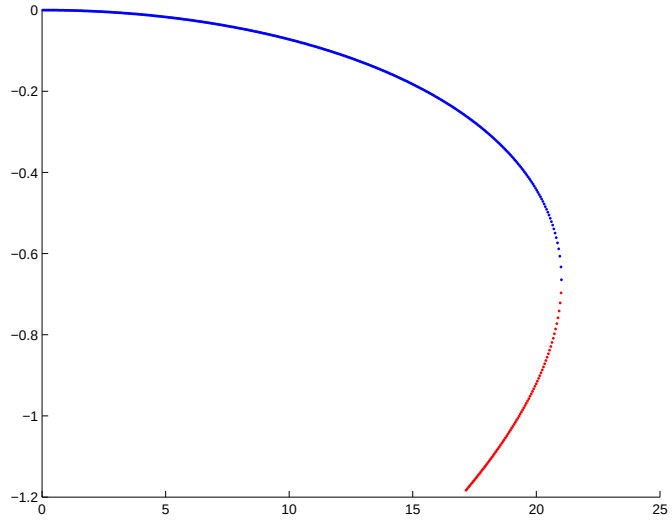


Model reduction



Parametric eigencomputations





- Can track resonant modes as geometry changes
- Track eigenvalues of linearized system to analyze stability of equilibria and bifurcation behavior
- Close relation to model reduction

Concluding notes

- Lots of pieces we don't yet touch (control laws, contact models, fluids, ...)
- Trying to remain relevant to current BSAC research
- Expect some intersection in aims (and SW engineering challenges!) with programs like Ptolemy