Advanced LIGO Suspension Model in Mathematica

Gravitational Wave Advanced Detector Workshop

Elba - May 2002

Mark Barton

Why a new model?

- The GEO design used as a baseline for Advanced LIGO comes with a Matlab model by Torrie, Strain, Fritschel et al.
- The Matlab model is useful (and the low run time means it will continue to have a role) but it has a number of limitations:
  1. Assumes symmetry
  2. Only directly models wires - allows for blades by approximate methods
  3. Only directly models longitudinal elasticity of wires
  4. Provision for thermal noise model separate
  5. Quad version much less thoroughly validated than triple

Why Mathematica?

- A framework for modelling pendulums had been created for earlier work on the X-pendulum (for TAMA300) which could be easily recycled
- Symbolic computation capability allows configuration of pendulum to be specified in high-level terms, with mass and stiffness matrices generated automatically
  - -> more flexible
  - -> more ambitious systems can be modelled
  - -> more reliable (debugging can be done largely by inspection of the specification)

Framework Features (i)

- The framework allows for the following elements to be freely combined:
  - » 6 degree-of-freedom rigid bodies
  - » massless wires with
    - – longitudinal elasticity based on geometric distance between endpoints
    - – bending elasticity (from solution of beam equation)
    - – additional longitudinal elasticity due to bending
  - » springs with
    - – 6x6 matrix of elastic constants between x, y, z, yaw, pitch, roll
    - – vector of 6 pre-stress forces and torques
  - » arbitrary frequency dependence of damping on each source of elasticity

Framework Features (ii)

- The framework computes the following automatically
  - equilibrium position
  - mass matrix M (such that $T = 1/2\ \underline{v}^T\ \mathbf{M}\ \underline{v}$)
  - separate stiffness matrices $\mathbf{K}_i$ (such that $E_i = 1/2\ \underline{x}^T\ \mathbf{K}_i\ \underline{x}$) for each group of terms in the potential with common damping
  - a total stiffness matrix (with damping neglected)
  - eigenfrequencies and eigenmodes
  - coupling matrices (analogous to stiffness matrices) from the support or other sources of displacement input to the pendulum
  - matrix valued "equation of motion" and "coupling" functions of frequency incorporating the frequency-dependent damping with dissipation dilution

Framework Features (iii)

- The framework has model-independent functions for
  - » transfer function plots for arbitrary inputs to arbitrary outputs (or linear combinations thereof)
  - » thermal noise plots for arbitrary degrees of freedom (or linear combinations thereof)
- The framework has utilities and templates that can be customized to create model-dependent functions for
  - » tabular display of normal modes
  - » 3D plots of normal mode shapes

Triple Model Features

- 3 objects representing upper masses and optic
- 6 objects representing tips of blade springs
- 9*6=54 total degrees of freedom
- 6 springs representing elasticity of blade springs
- 10 wires (or fibres depending on parameters)
- parameter names as for Matlab triple model
- parameter sets representing
  - » Glasgow all-metal protoype
  - » current mode cleaner reference design

Quad Model Features

- 4 objects representing upper masses and optic
- 6 objects representing tips of blade springs
- 10*6=60 total degrees of freedom
- 6 springs representing elasticity of blade springs
- 14 wires
- parameter names as for Matlab quad model
- default parameters simulating MIT quad protoype (but easily overridable)

Sample Input (i)

```
allvars = {
      xul,yul,zul,yawul,pitchul,rollul,
      xur,yur,zur,yawur,pitchur,rollur,
      x0,y0,z0,yaw0,pitch0,roll0,
      xil,yil,zil,yawil,pitchil,rollil,
      xir,yir,zir,yawir,pitchir,rollir,
      x1,y1,z1,yaw1,pitch1,roll1,
      xll,yll,zll,yawll,pitchll,rollll,
      xlr,ylr,zlr,yawlr,pitchlr,rolllr,
      x2,y2,z2,yaw2,pitch2,roll2,
      x3,y3,z3,yaw3,pitch3,roll3
};
```

- The specification of the model starts with a list of 'variables" describing the state of the system and a similar list of "parameters" describing the support structure

Sample Input (ii)

```
optic = {x3, y3, z3, yaw3, pitch3, roll3};

opticlf={-sl,-n5,d4};

gravlist = {};
AppendTo[gravlist, mbeu g zul];
AppendTo[gravlist, mbeu g zur];
…
AppendTo[gravlist, m3 g z3];
```

- Specify body coordinate systems for objects
- Specify wire attachment points etc in body coordinates
- Specify gravitational terms in potential

Sample Input (iii)

```
AppendTo[wirelist,{
    massI,
    massIlf,
    massIlfvec,
    optic,
    opticlf,
    opticlfvec,
    Y3,
    ul3,
    kw3,
    {1,0,0},
    M31,
    M32,
    fibretype,
    fibreatype
}];
```

- Specify wires:
  - Masses
  - Attachment points
  - Attachment angles
  - Young's modulus
  - unstretched length
  - net longitudinal elasticity
  - principal axis
  - moment of area parallel and perpendicular to principal axis
  - tags representing damping for longitudinal and bending

Sample Input (iv)

- Specify the kinetic energy as an expression

```
kinetic = (
 (1/2) mbeu Plus@@( Dt[b2s[bladeUL,COM],t]^2)
 +(1/2) omegaB[yawul, pitchul, rollul].IBU.omegaB[yawul, pitchul, rollul]
…
);
```

Sample Input (v)

- Specify values of constants and damping functions in a mix of symbolic and numeric forms

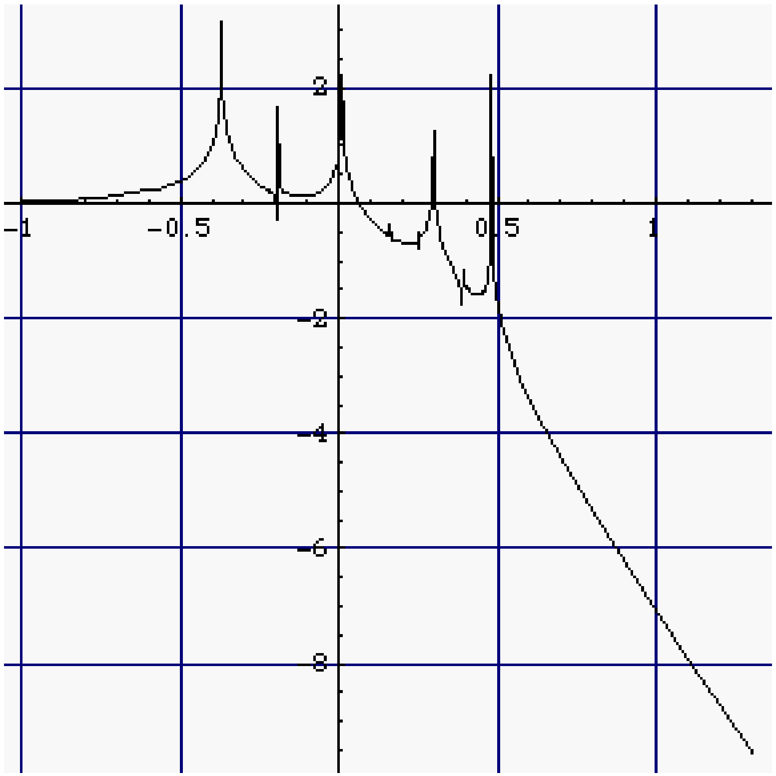
```
defaultvalues = {
    g -> 9.81,
...
    tx -> 0.12,
    tr -> 0.14,
    den3 -> 4000.,
    m3 -> den3*N[Pi]*tr^2*tx,
...
    damping[imag,fibretype] -> ((phisilica + phissilica/r3/2)&),
    damping[imag,fibreatype] -> ((phisilica + phissilica/r3 +
    deltafibre*(2*N[Pi]*#1*taufibre)/(1+(2*N[Pi]*#1*taufibre)^2))&)
};
```

Sample Output (i)

- Eigenmode listing

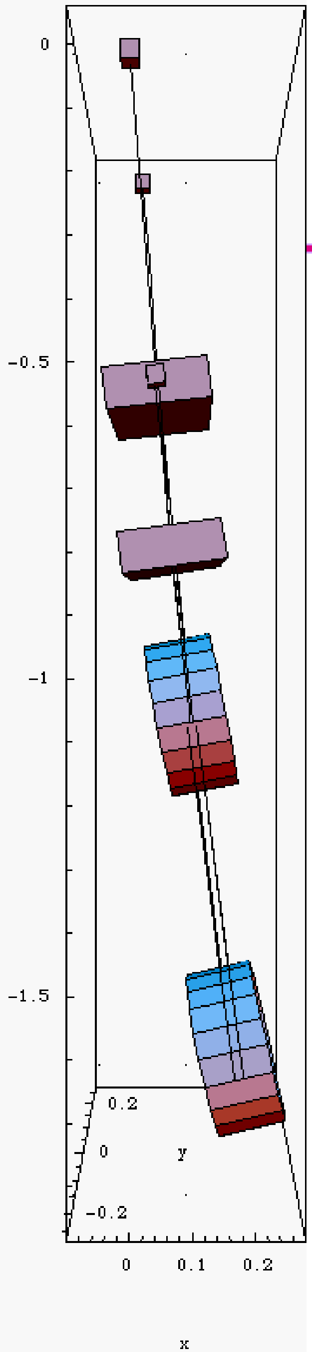
	x	y	z	yaw	pitch	roll
UL blade	0.00058671	0	0	0	-0.000160617	0
UR blade	0.00058671	0	0	0	-0.000160617	0
Mass N	0.0798412	0	0	0	-0.231023	0
IL blade	0.0799878	0	0	0	-0.231074	0
IR blade	0.0799878	0	0	0	-0.231074	0
Mass U	0.131946	0	0	0	-0.287388	-0.000184555
LL blade	0.132145	0	0	0	-0.287503	-0.000184785
LR blade	0.132145	0	0	0	-0.287503	-0.000184785
Mass 2	0.198733	0	0	0	-0.379219	-0.000173231
optic	0.360874	0	0	0	-0.454855	-0.000173412

Sample Output (ii)



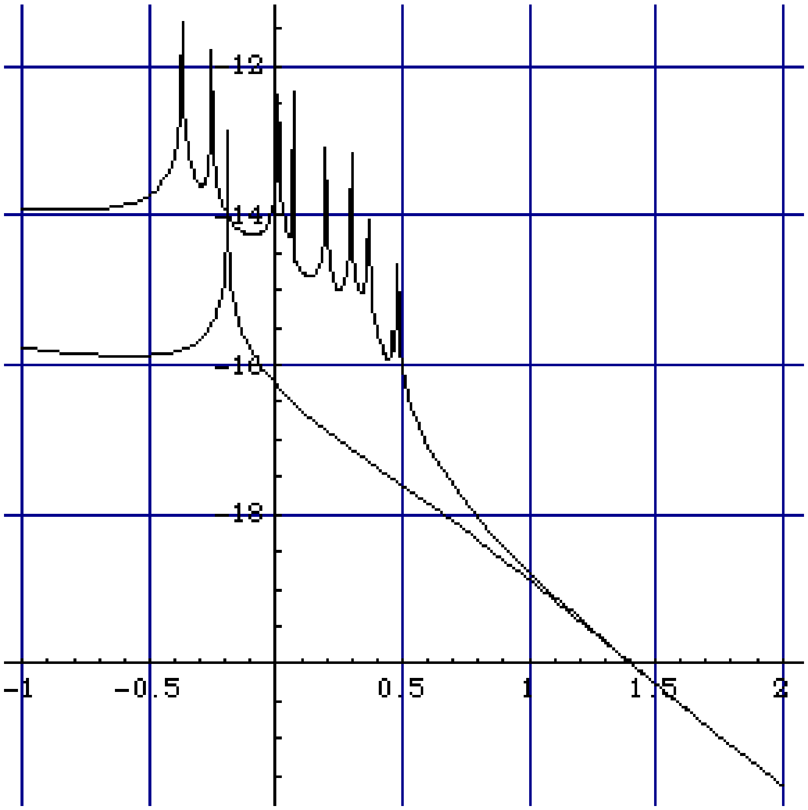
Eigenmode plot ->

<- Transfer function
(x-x)



Sample Output (iii)

- Thermal noise plot
  - » Data from version of MIT quad prototype with fused silica fibres instead of wires in the last stage
  - » Comparison curve (lower) from Fluctuation Dissipation Theorem applied to last stage only



Status

- Triple version validated against Torrie et al’s Matlab triple - > four figure agreement in mode frequencies for key test cases
- Quad version validated against (corrected) Matlab quad model
- Thermal noise at higher frequencies validated against analytical predictions using FDT on last stage only
- Validation for asymmetrical systems in progress
- Blade spring model adequate for low frequency purposes but crude - a better one will be developed over the next few months
- Current versions online:

  http://www.ligo.caltech.edu/~mbarton/SUSmodels/