LIGO Scientific Collaboration meeting
Livingston March 15th, 2001

# Preliminary Results of the E2E/MSE Mechanical Simulations for the Advanced LIGO

LIGO-G010089-00-R

The E2E Team

Presented by: **V. Sannibale**,

# Talk Overview

- The MSE Simulation Engine for E2E.
  - Mechanical objects available
  - Main functionalities
  - integration with other tools
- The Advanced LIGO quadruple suspension simulation.
  - Approximations
  - Results
  - What needs to be done
  - Goals we can achieve with this kind of simulation
- Conclusions.

# MSE Basics: Mechanical Objects

- RigidBody
  - parameters: Mass, Principal Axis Moment of Inertia
- Blade
  - parameters:
- Elastic Line
  - Wire
    - parameters: Diameter, Length, Mass Density, Young Modulus, # of Modes
  - Beam
    - parameters: Cross Section Momenta, Length, Mass density, Young Modulus Poisson Coeff.,# of modes
  - Spring
    - parameters: Spring Constant, Length
- Dashpot
  - parameters: Dissipation Coefficients Matrix
- Sensors/Actuators
  - Position Sensor/Actuators
  - Force Sensor/Actuators
- Clamp (constraint)
- U2F (parameterized mech. object)

# MSE Basics: Functionalities

- Time domain simulation (impulse/step response)
- Frequency domain simulation (transfer functions, noise spectral densities...)
- State Space Representation.
- Integration with E2E (almost complete)
- Thermal noise prediction using modal analysis (under development).

# MSE Basics: Matlab Interfacing

- MSE outputs for Matlab:

  *filename*.ss : file containing the state space description ($A, B, C, D$ matrices)

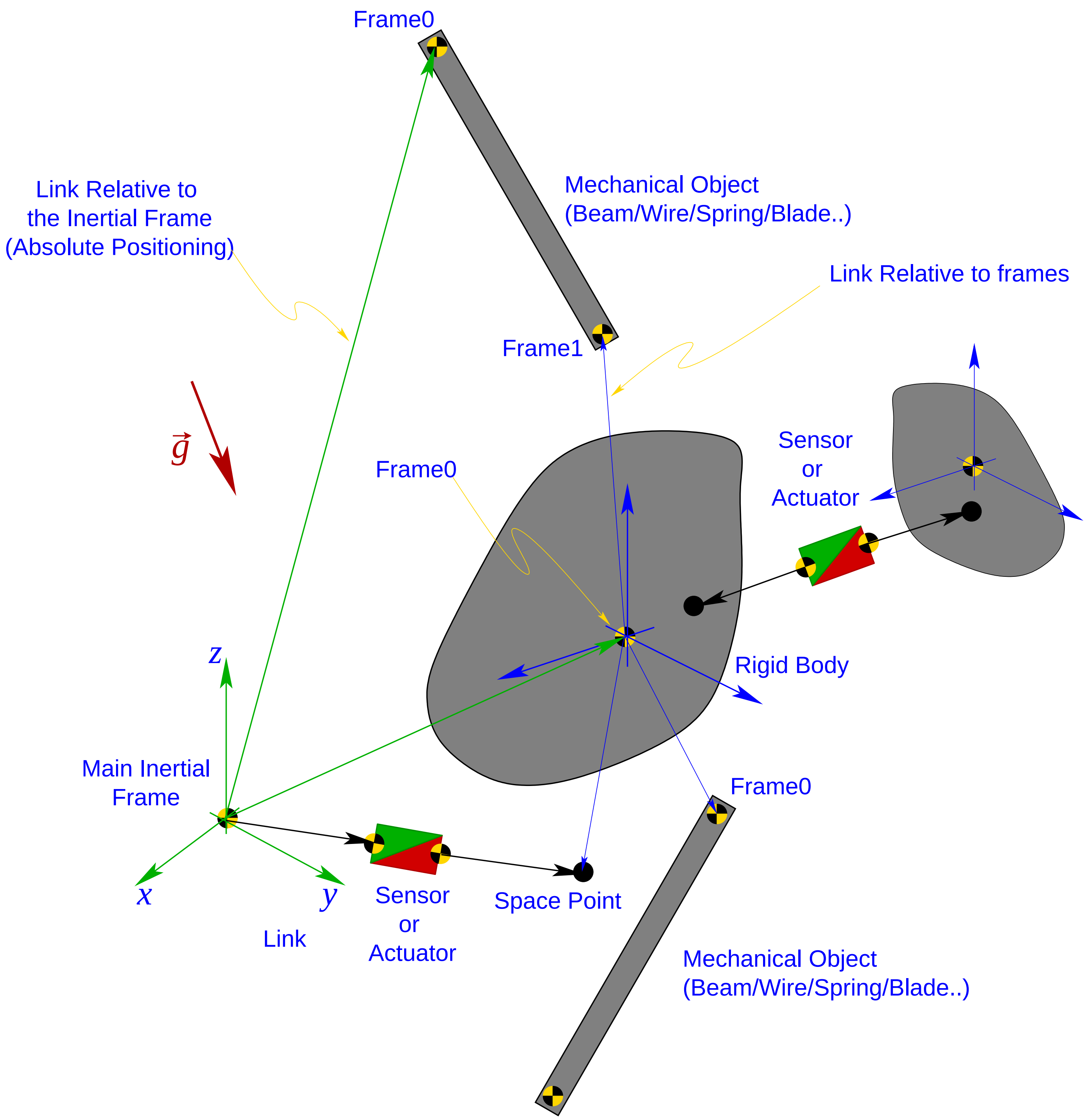
  *filename*.txt : file containing the state input and output vector description in a "human" readable form.

- Matlab function to import data from MSE: MSERead

```
>>help MSERead
Purpose : import state space description from a MSE simulation.

Synopsis: Structure = MSERead(FileName)

Example : Pend = MSERead('pendulum.ss')

it creates a structure containing the following elements :
Pend.In : n of inputs
Pend.Out : n of outputs
Pend.DOF : state space vector dimension
Pend.A : A matrix
Pend.B : B matrix
Pend.C : C matrix
Pend.D : D matrix
Pend.ss : state space structure
```

- Problems: Matlab capability of handling large matrices could be a limiting factor. A More flexible interface is probably needed (input output selection or model simplifications).

# MSE Basics: Mechanical Topology Definition

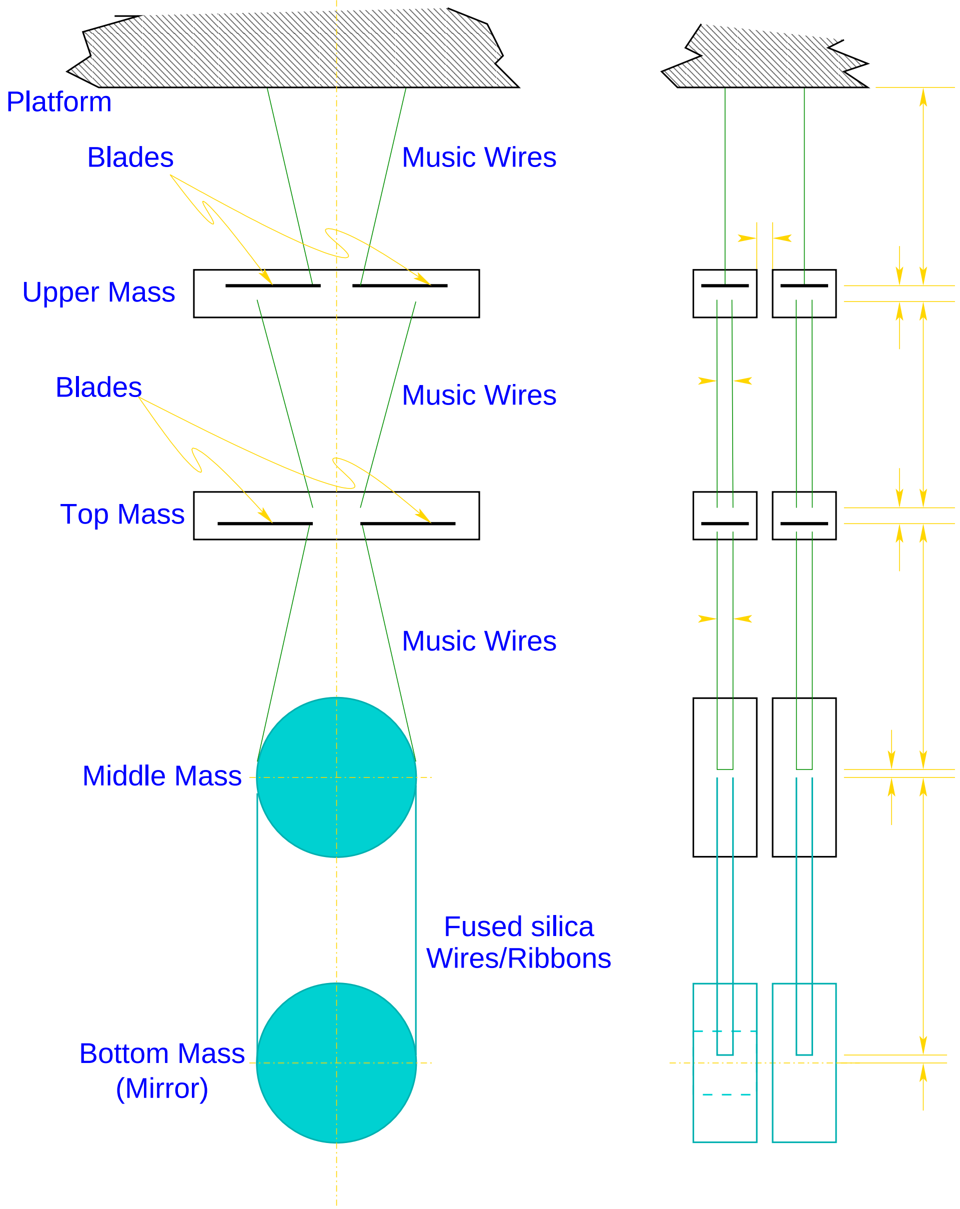


# Simulation: Quadruple Pendulum (QP), Main Approximations

- General Approximation:
  - Complete three-dimensional simulation.
  - Small oscillation regime (simulation in a linear regime =¿ no parametric oscillations, etc ...).
  - Viscous Damping (Structural Damping under construction).

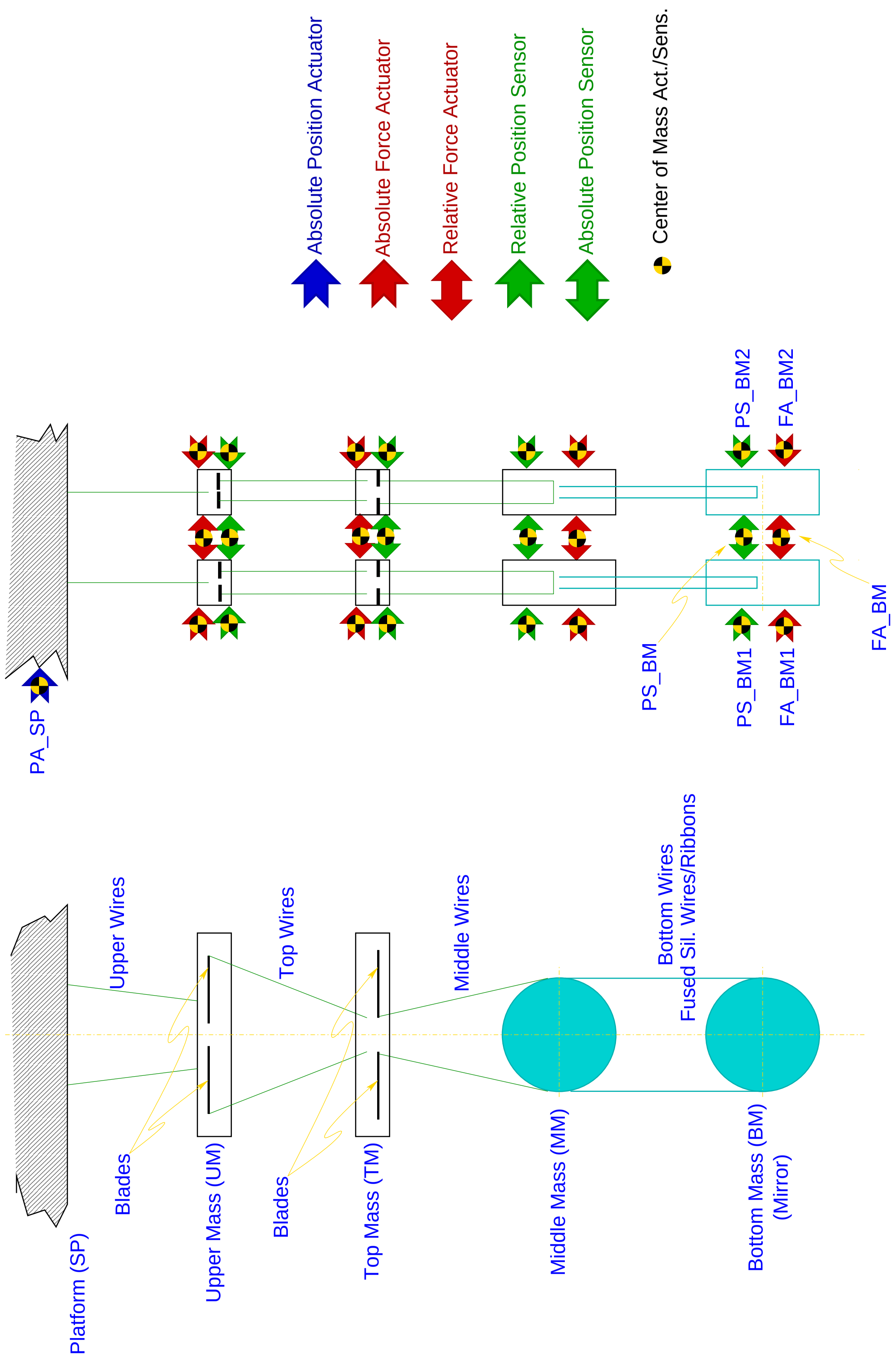
- Both quadruple chains implemented.
- Rigid body D.O.F. completely described.
- Blades simulated using springs.
- Actuators and sensor acting and sensing respect the C.of.M (simplification just for easier understanding)
- Violin Modes not implemented (coming soon ...under debugging).
- Blades Internal Modes not implemented yet (coming soon... under debugging).
- Center of Percussion effect not implemented (Under debugging)

# Simulation: LIGO-II SUS, Sketch

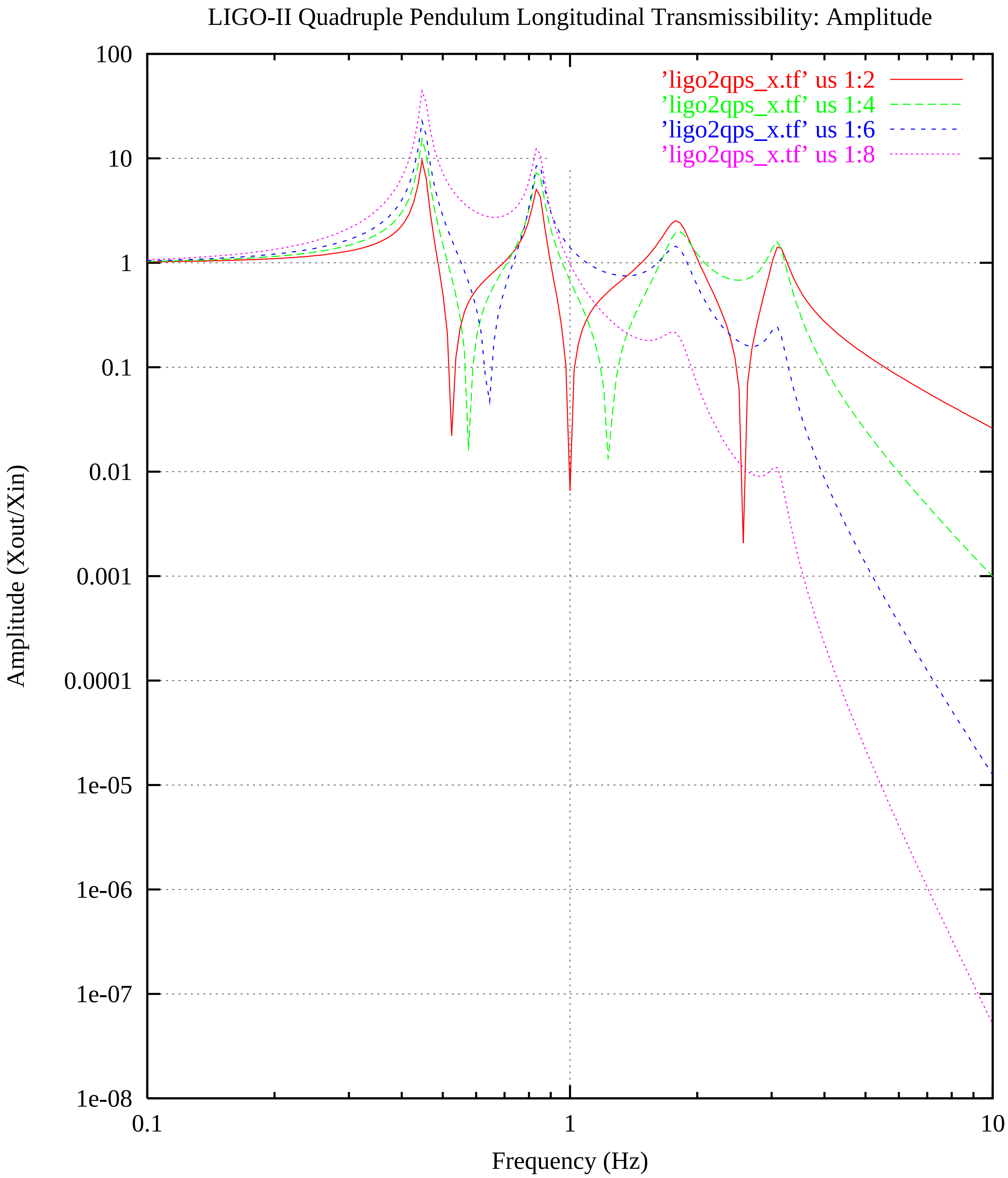



- Total number of geom./phys. parameters : $\sim 60 \times 2$
- Total number of DOF : $\sim 30 \times 2$
- Total number of inputs : $\sim 24 \times 2$
- Total number of outputs : $\sim 30 \times 2$

# Simulation: Suspension Act./Sens Map

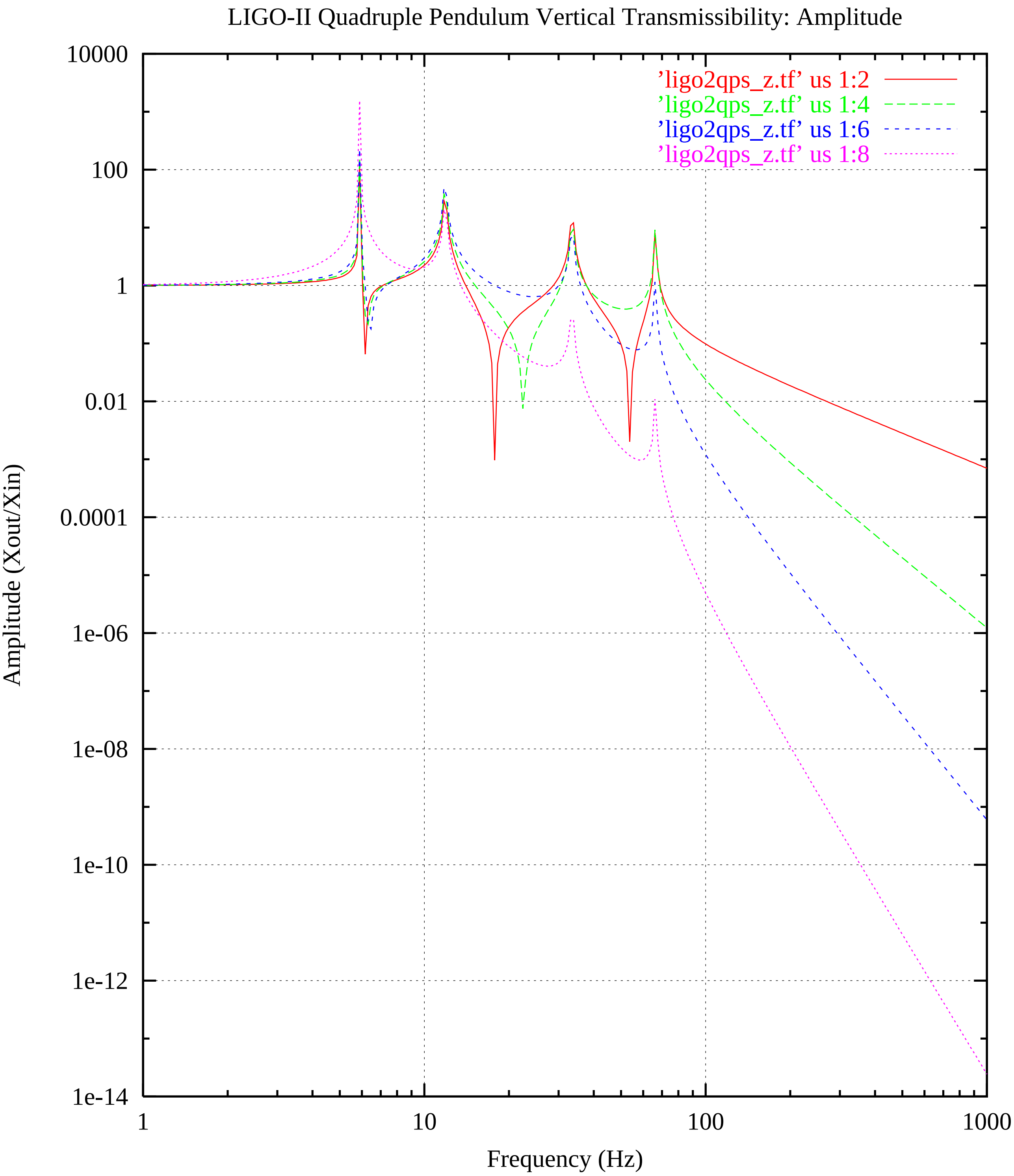


# QP Simulation: Longitudinal Transmissibility



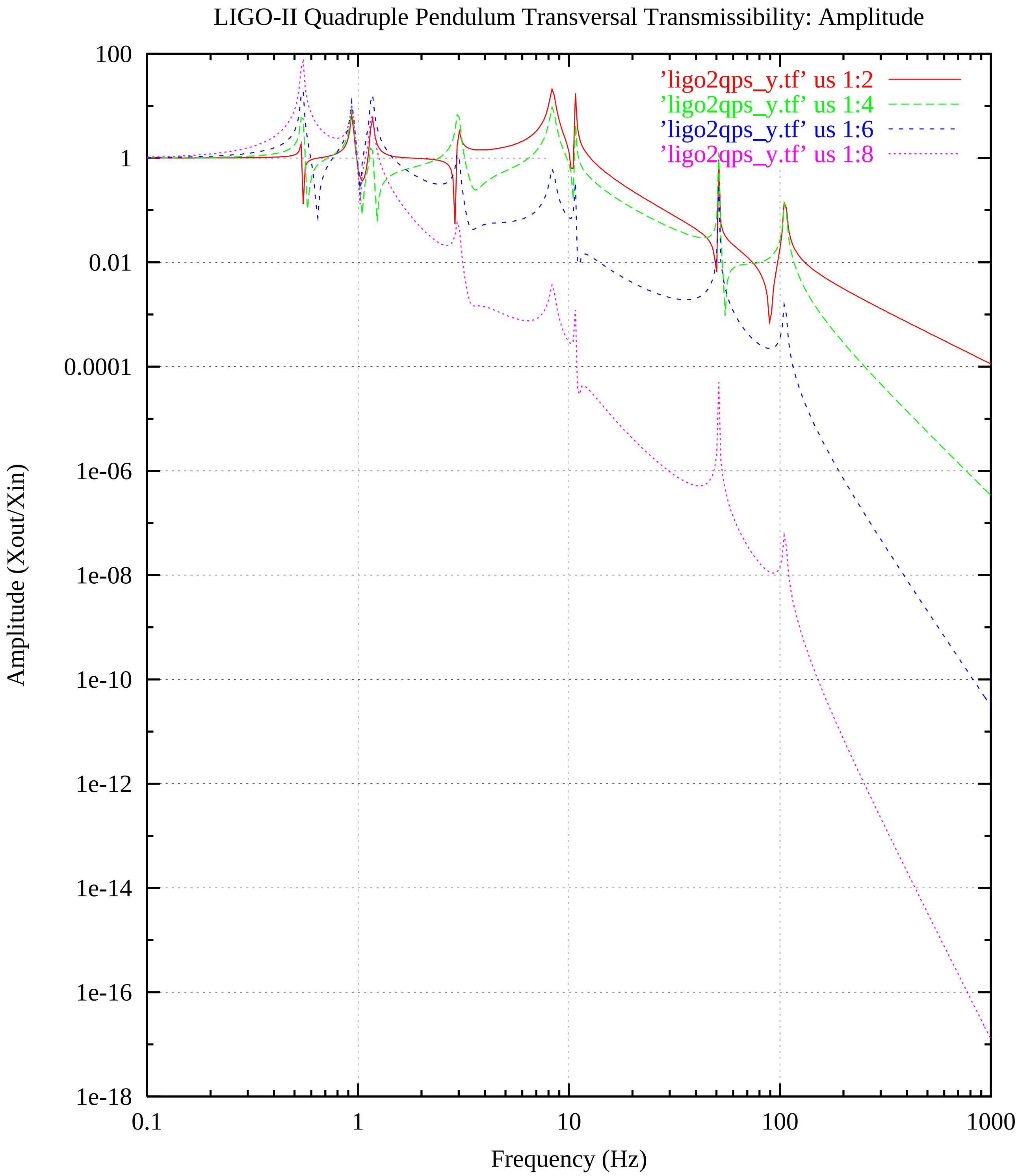

Pitch mode coupling removed for easy cross-check (wires hooking points in the center of mass horizontal plane of the rigid bodies).

# QP Simulation: Vertical Transmissibility



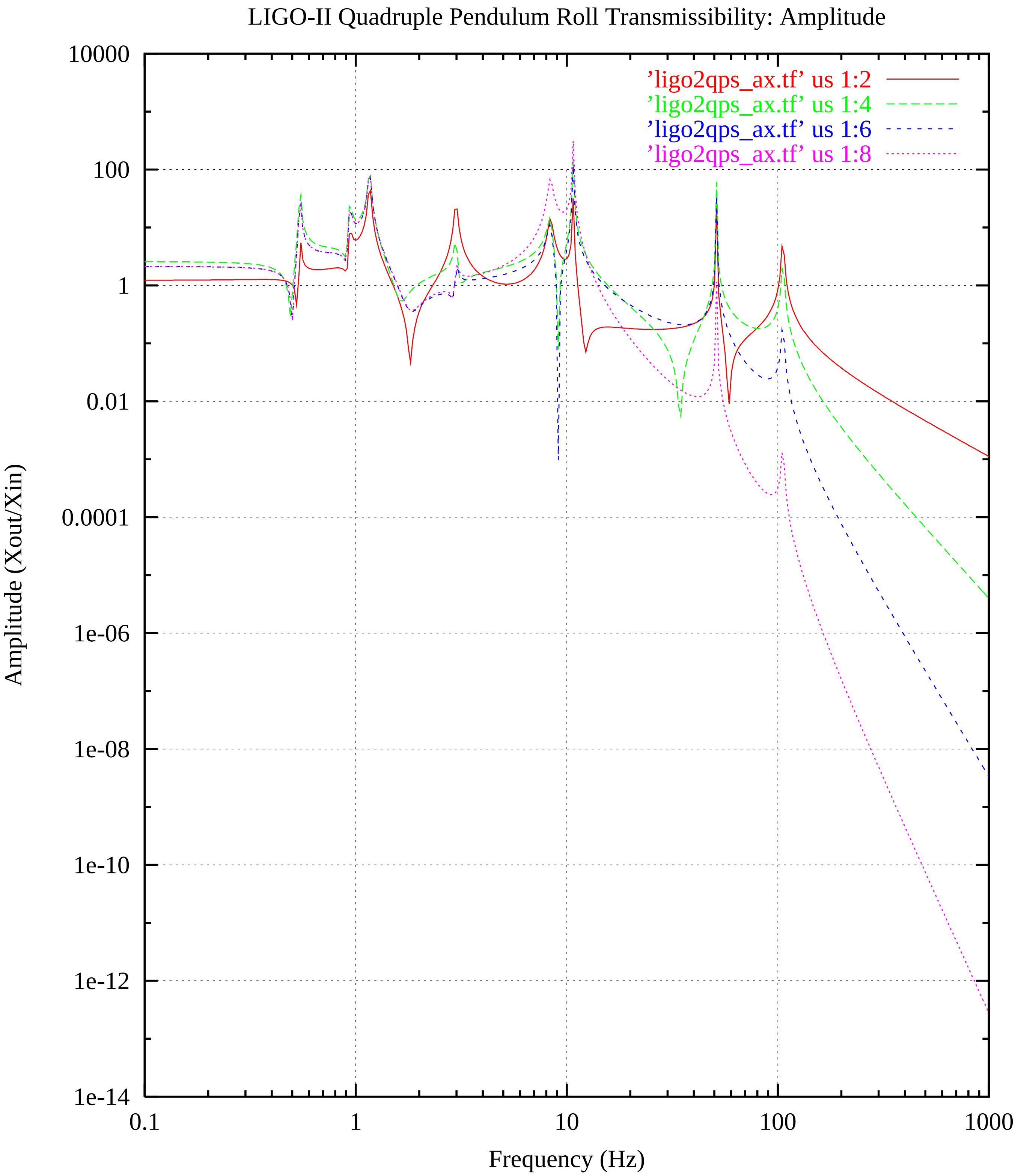

No coupling with other modes for easy cross check (No suspension asymmetries introduced) Vertical bounce modes at "high frequency" because of the absence of the blades.

# QP Simulation: Transversal Transmissibility




Coupling with the vertical bounce modes at "high frequency" because of the absence of the blades.

# QP Simulation: Roll Transmissibility

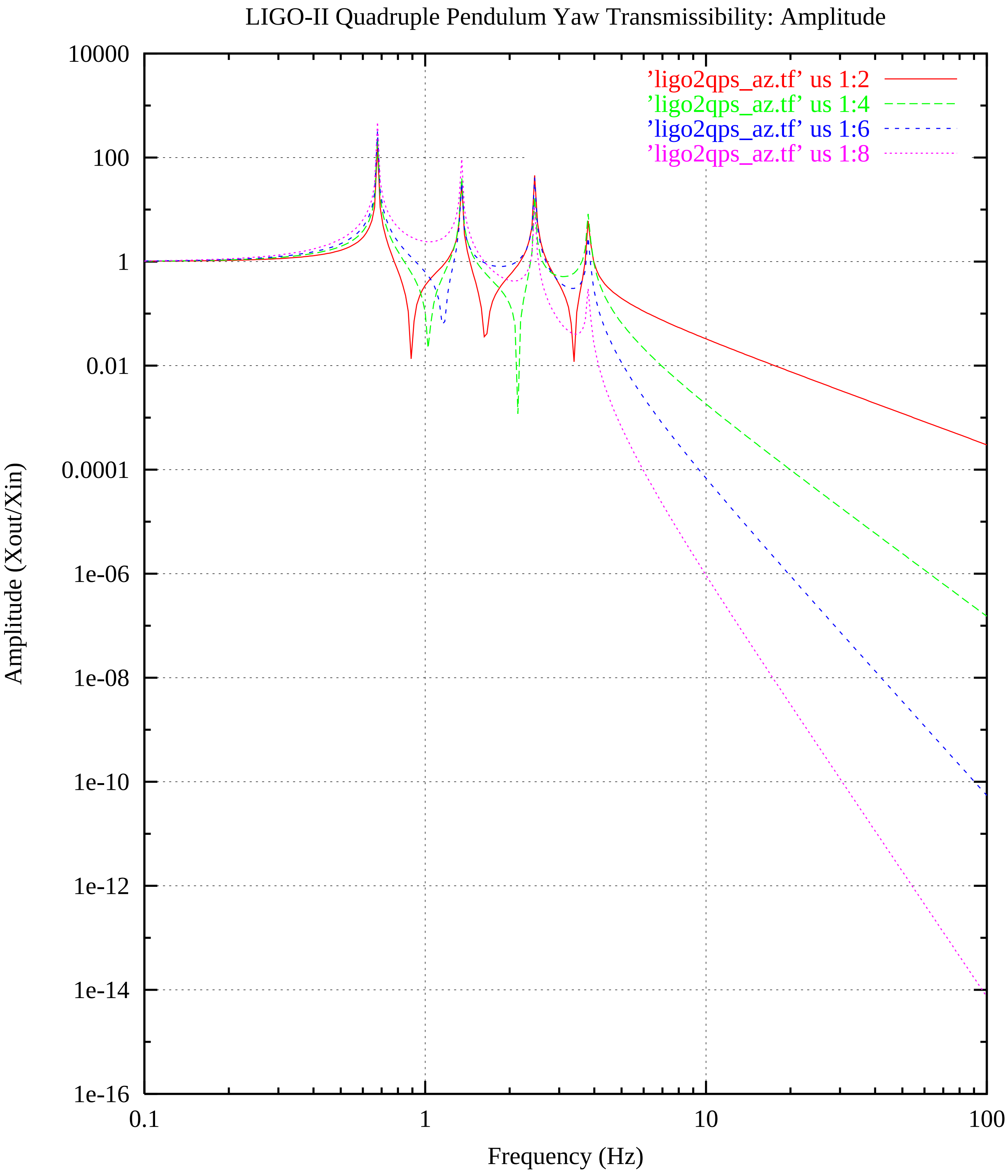



Coupling with the vertical bounce modes at "high frequency" because of the absence of the blades.

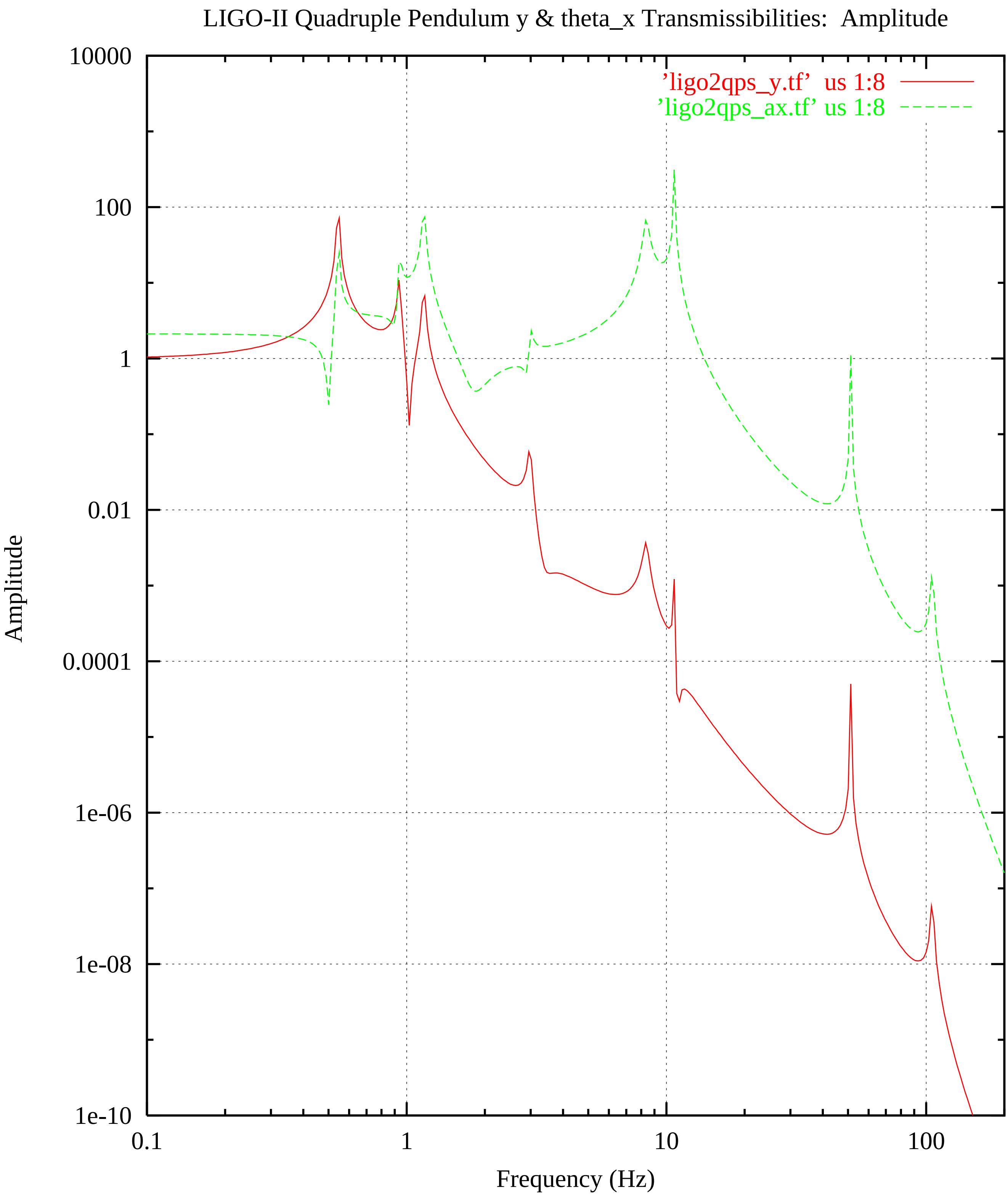
# QP Simulation: Pitch Transmissibility

Flexural mode of the wires not implemented => no propagation of the pitch trough the suspension

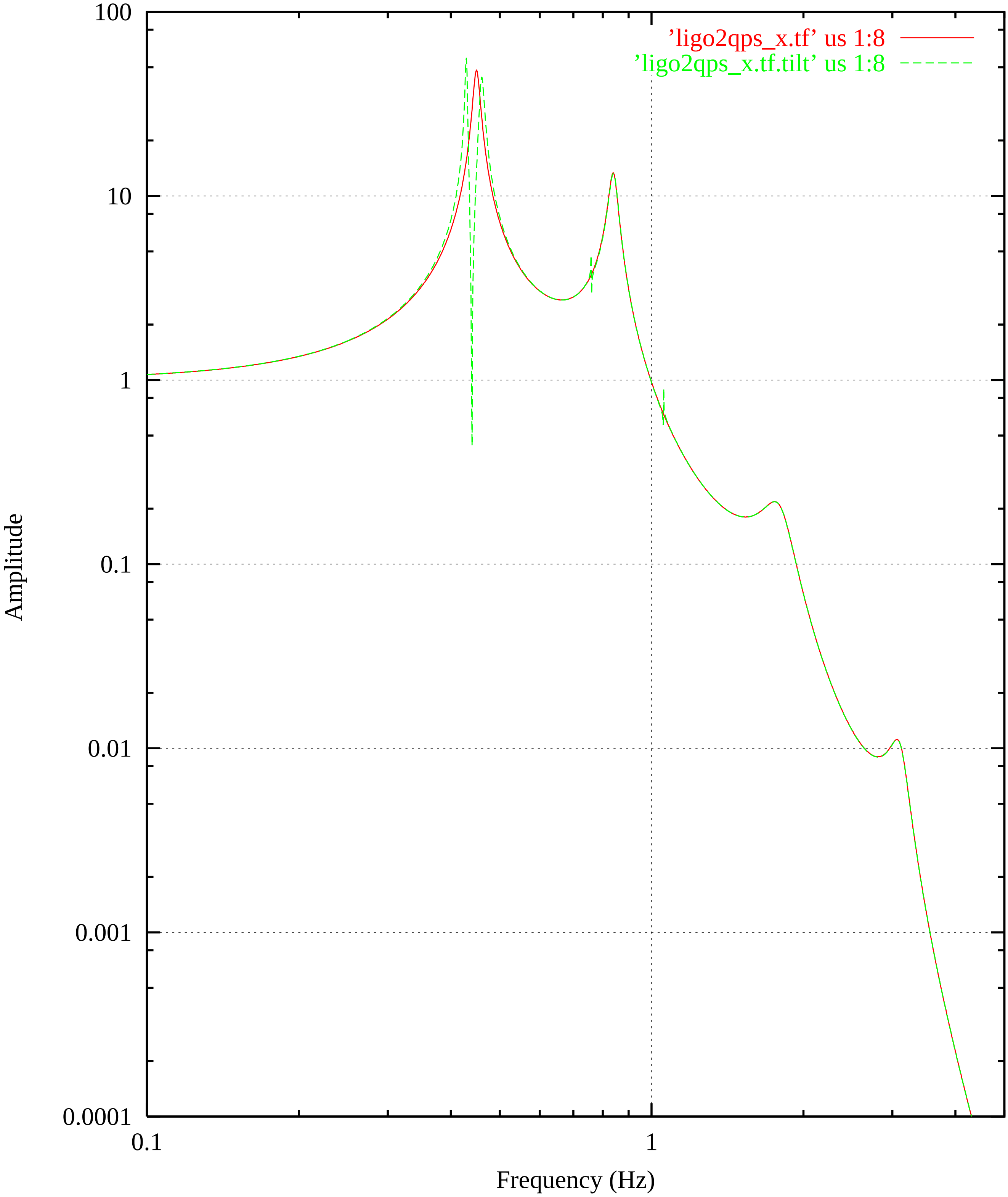
# QP Simulation: Yaw Transmissibility



# QP Simulation: Transv. vs Roll Transmissibility



# QP Simulation: Pitch Coupling with Long. Translation




Wire Hooking Point : vertical displacement from Center of Mass 2 mm

# Simulation: Validation

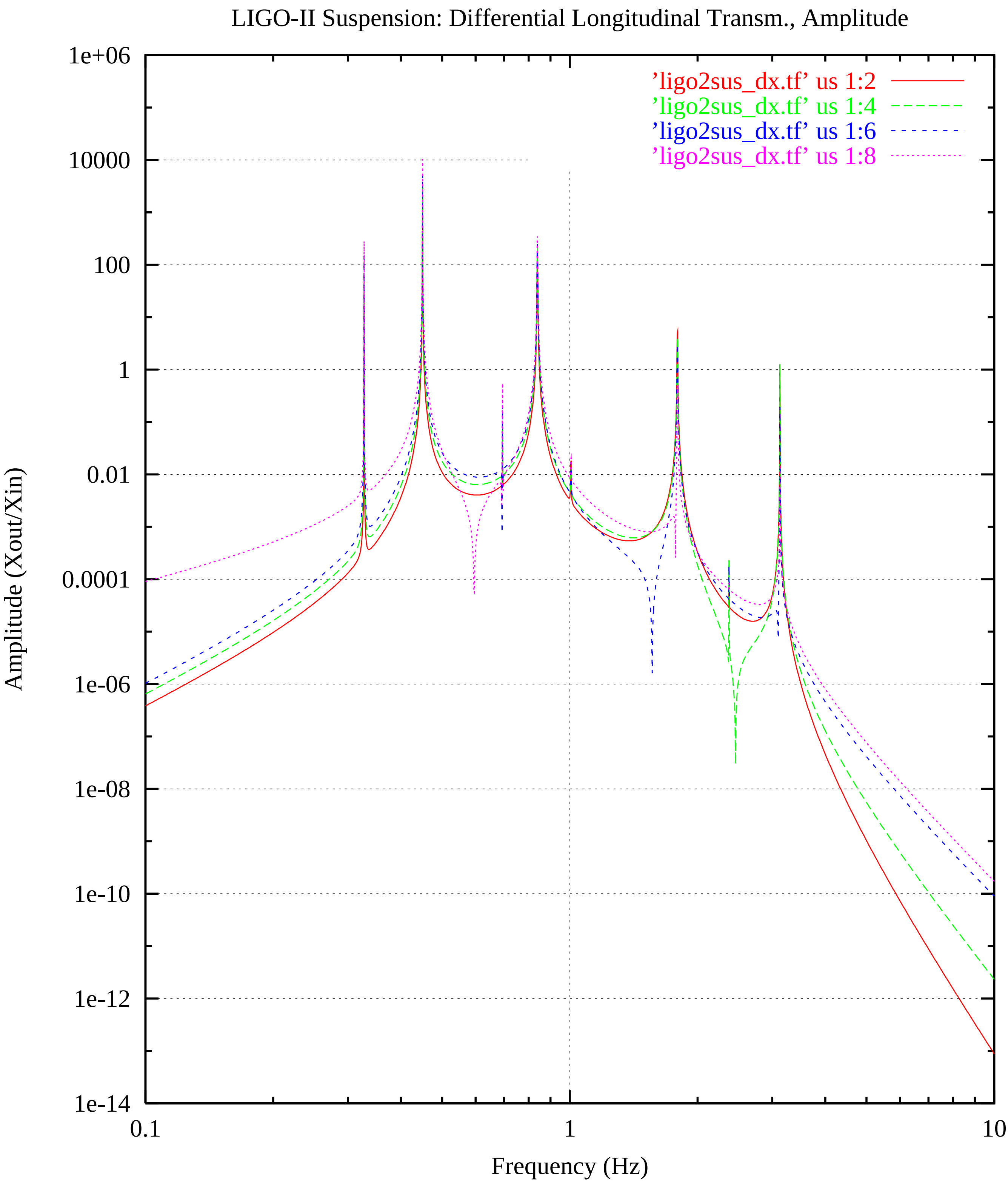
Need of a set of measurement to have a sufficient confidence on the validity of the simulation on a prototype:

- Rough Mode Contents Validation : Spectral Density of sensors with chain properly excited. (reasonably easy task).

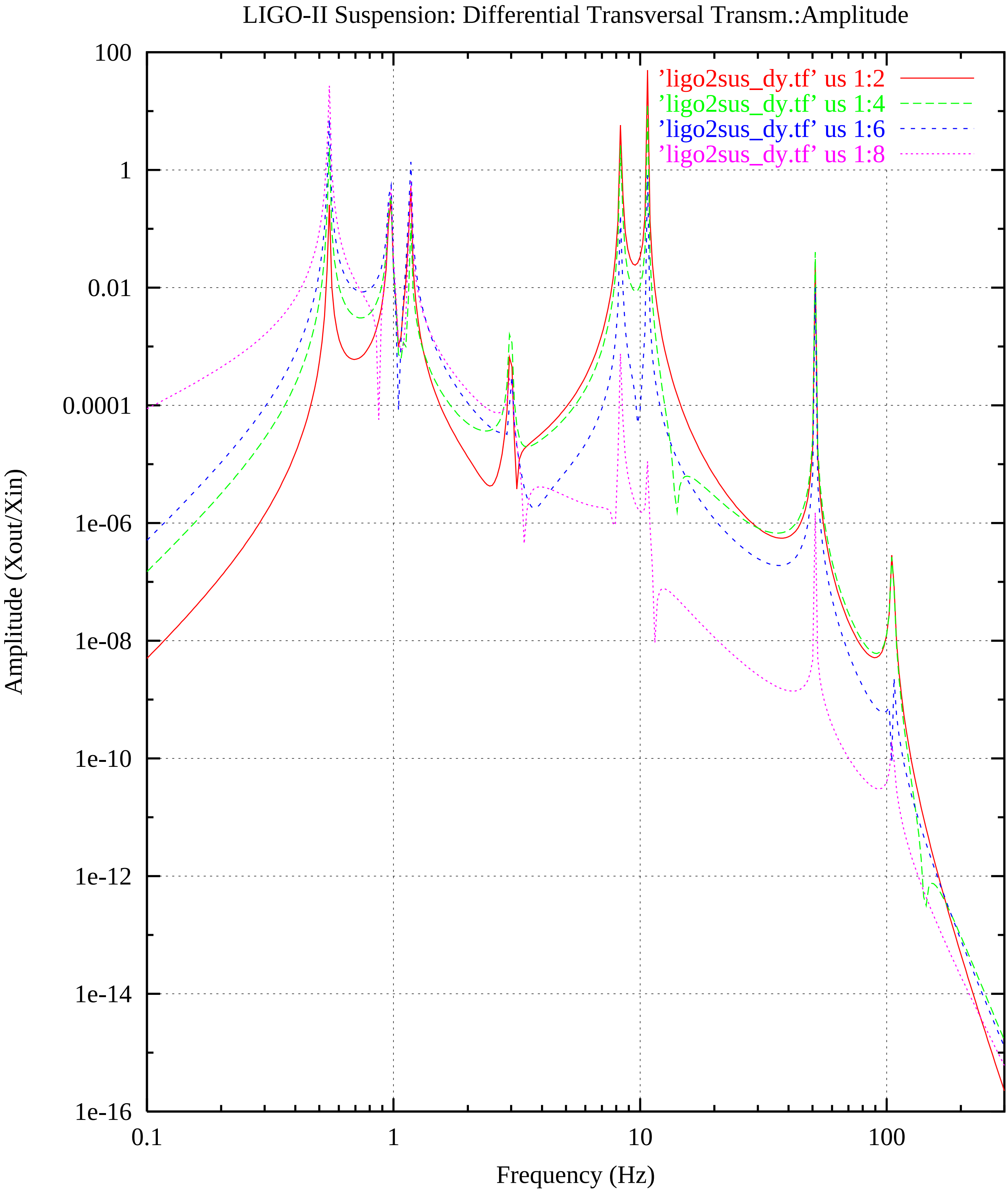
- Transmissibility: Stage by stage transfer function (quite time consuming and hard task => impractical). Probably the upper stages transfer functions can give enough information for the validation of the simulation.

- Ring Down measurements to characterize the modes Quality Factors. (time consuming and hard task).

# LIGOII-SUS Simulation: Longitudinal Transmissibility

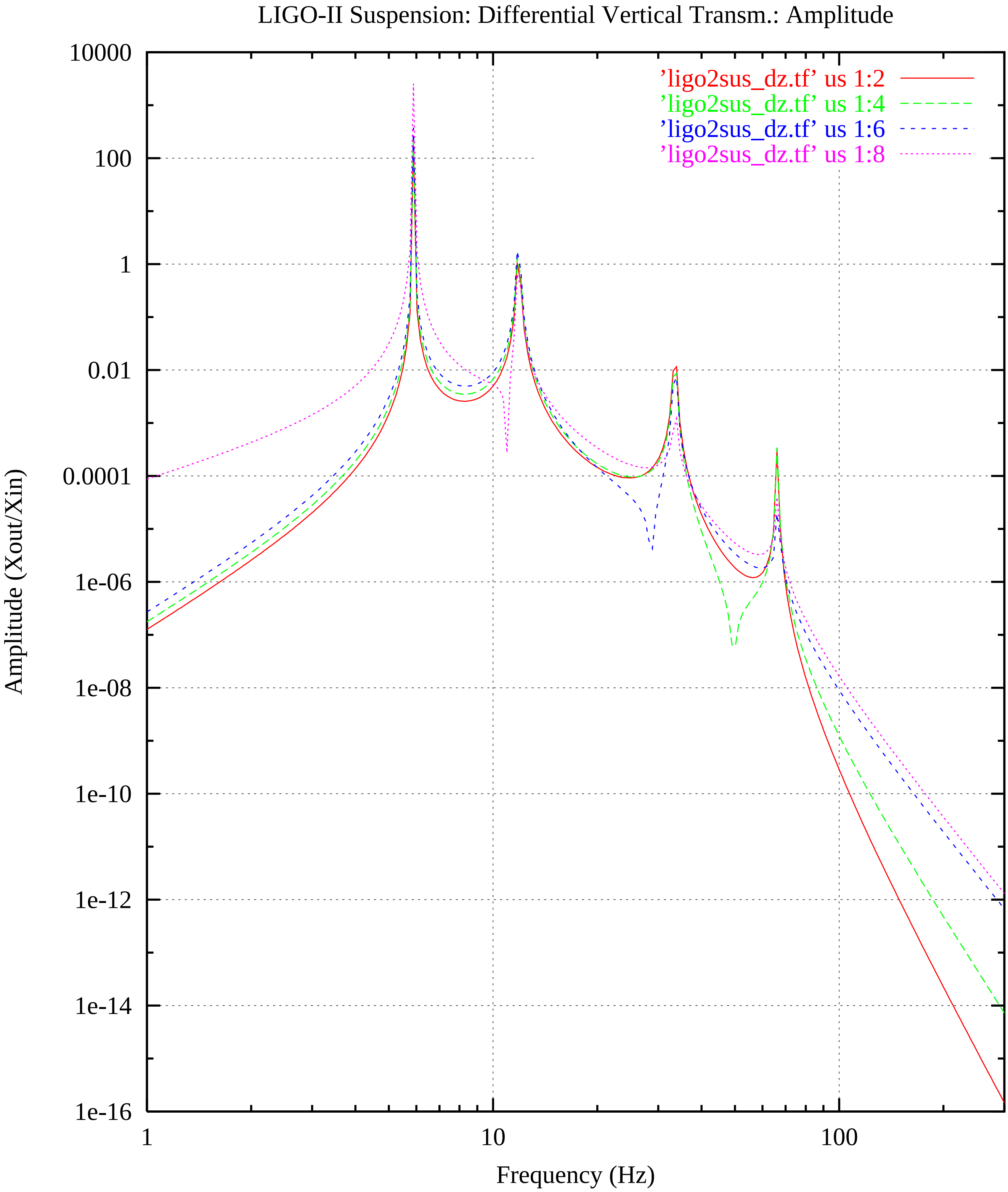


# LIGOII-SUS Simulation: Transversal Transmissibility



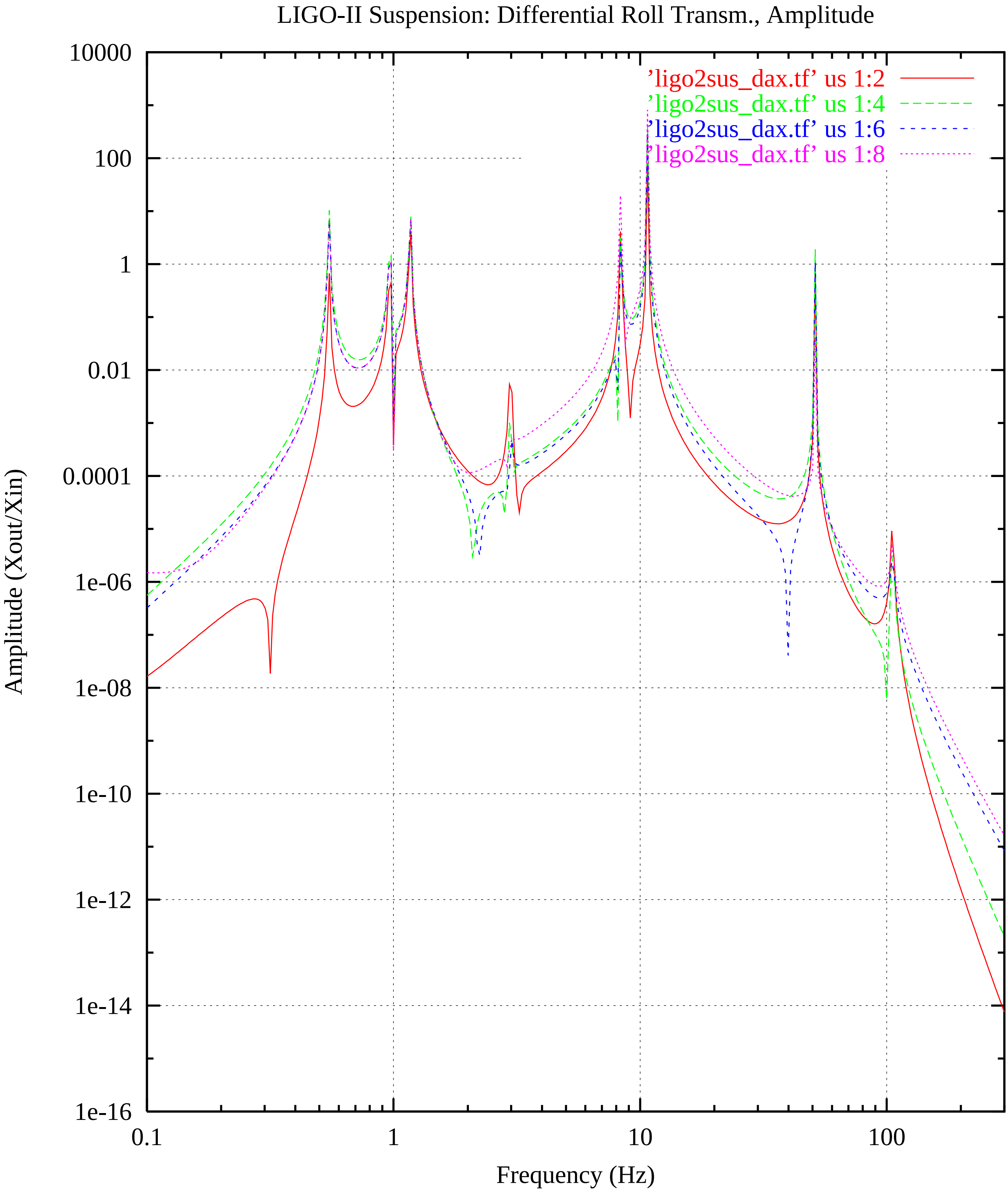

Last stage reacting chain with wires 2 mm longer

# LIGOII-SUS Simulation: Vertical Transmissibility




Last stage reacting chain with wires 2 mm longer

# LIGOII-SUS Simulation: Roll Transmissibility




Last stage reacting chain with wires 2 mm longer

# LIGO-II SUS Simulation: What Needs to be done

- Systematic check of all the transfer function simulation outputs.
- Introduce the blades dynamics (under debugging)
- Introduce the suspension violin modes (under debugging)
- Introduce a more realistic sensor actuator topology
- Improve the mse interfacing to Matlab

# Conclusions

- LIGO II suspension three-dimensional simulation for the rigid body modes completed.
- Results obtained agree with the outputs of previous well debugged simulations (GEO folks).
- Preliminary check shows a reasonable "behavior" of the simulation. (Ready for a preliminary study of global/local control issues).
- Integration with Matlab to study SUS control ready to be used (some optimization probably needed)