Features in the Quad State Space Model

Contents

- Feature summary
- Feature description
- Where in the SVN does the model live?
- 3 steps to edit the model features
- How to call the model

Feature Summary

- Built in features – always included in the model
  - Relative SUS/cage sensors and actuators – new as of 1 Jan 2016
  - Suspension point reaction forces – new as of 1 Jan 2016

- Optional features – your choice to include or not
  - Two chain model – includes both a main and reaction chain
  - Violin modes
  - Damping
    - Import live damping from sites
    - Import damping from a prior GPS time from sites
    - Include optical lever damping (PUM actuation)
    - Load damping from a saved filter file or command space variable

- Possible future features
  - Global control
  - Radiation pressure

Note: Graphical Simulink representations of the model layouts are cited in step 1 of the '3 steps to edit the model features' section. These will provide more detail for some of the descriptions below.

Built in features always included in the model

Feature	Description
Relative SUS/cage sensors and actuators Added 1 Jan 2016	- The model incorporates the relative nature of the sensors and actuators. E.g. the damping OSEMSs see both the SUS and cage and drive both the SUS and cage. - Outputs exist to examine both true SUS displacement and the relative displacement seen by the sensors. Relative outputs at just the top for a single chain, all stages for two-chain models. - In the case of two-chain models, independent drive inputs exist for each chain at all stages, as well as differential inputs for the global control actuators.
Suspension point reaction forces Added 1 Jan 2016	Outputs exist for the reaction forces exerted by the top suspension wires at the suspension point. The damping reaction forces are also projected here. This permits the SUS model to be dynamically combined with the ISI model.

Optional features
your choice to include or not

Feature	Description
Two chains	A model with both a reaction chain and main chain. The chains share seismic inputs and global control sensor/actuator signals. They also have independent force inputs and displacement outputs at all stages. Each chain has its own suspension point reaction force outputs.
Violin modes	You can compile the model with violin modes at any, or all, wires of your choosing. You specify the number of modes you want at each stage. The model simulates left and right wires so that the modes couple to length, yaw, and pitch. Violin modes are only available for the main chain. In the case of two-chain models, only the main chain receives the modes. You can specify measured mode frequencies and Qs by declaring them in the parameter file, e.g. h2etmy.m. The format for this is to add these variables, with frequency in Hz: fibers: fibermode_freq, fibermode_Q uim-pum wires: uimpummode_freq, uimpummode_Q top-uim wires: topuimmode_freq, topuimmode_Q top wires: topmode_freq, topmode_Q The way the model handles these is that any measured values in the parameter file simply replace the modeled values. Thus, if there are gaps in the data, e.g. measurements for modes 1 and 3, but not 2; modeled values fill in these gaps.

Optional features your choice to include or not

Feature	Description
Import live damping filters from sites	The damping loops currently running on a given suspension at a given site can be included in the model. A 5 min delay is built in to allow for latency. The can include top mass damping alone, or top mass with oplev damping. Some setup is required to make this work on your local computer.
Import damping filters from sites from a prior GPS time	Just like the live damping option, except the damping is loaded from a selected prior GPS time. The same setup is required.
Include custom filters from the command line or from a saved file	Make your own damping filters for the model. These are read in via a .mat file or from a command space variable. This must be a struct with fields L.c,T.c,V.c,Y.c,P.c,R.c, L2OL_P.c, L2OL_Y.c. For a saved file, the struct must be named calibFilter for historical reasons. From the command line the struct can be any name.

Note: Any combination of the 3 optional features categories can be included in the model. E.g. you can make a model with two chains, violin modes, and damping; or a single chain with violin modes but no damping; etc.

Possible Future Features

No options yet exist for:

- global control - length and angular loops
- the inclusion of radiation pressure effects

These can be added as needed.

Where to find the model in the SVN

The model files are found in

```
    …/SusSVN/sus/trunk/QUAD/Common/MatlabTools/QuadModel_Production/
```

The model is compiled with the function

```
    generate_QUAD_Model_Production.m
```

```
The supporting files called by this script are:

    define_quadModel_insandouts.m -> defines the input and output indices
    makequad_with_modal_fibers.m  -> adds violin modes
    M0LiveDampingFilters.mdl -> imports main chain damping from sites
    R0LiveDampingFilters.mdl -> imports reaction chain damping from sites
    L2OplevLiveDampingFilters.mdl -> imports oplev damping from sites
    generate_QUAD_SingleChainUndamped_Simulink.slx -> layout description
    generate_QUAD_SingleChainDamped_Simulink.slx -> layout description
    generate_QUAD_BothChainsUndamped_Simulink.slx -> layout description
    generate_QUAD_BothChainsDamped_Simulink.slx -> layout description
```

3 steps to edit the model features

1. If you are changing the signal flow, edit the appropriate Simulink .slx layout file

There are 4 possible layout files for the model depending on which features are called:

1) A single undamped chain: `generate_QUAD_SingleChainUndamped_Simulink.slx`
2) A single damped chain: `generate_QUAD_SingleChainDamped_Simulink.slx`
3) Two undamped chains: `generate_QUAD_BothChainsUndamped_Simulink.slx`
4) Two damped chains: `generate_QUAD_BothChainsDamped_Simulink.slx`

1b. If you are adding new filters that might be imported from the sites then you must also add these to the import list in the appropriate .mdl file, in addition to the layout file. These are separate from the layout because the user might supply their own filter designs. Import files currently exist only for top mass damping. Make new files as needed.

- main chain damping filters for import: `M0LiveDampingFilters.mdl`
- reaction chain damping filters for import: `R0LiveDampingFilters.mdl`
- Oplev damping filters for import: `L2OplevLiveDampingFilters.mdl`

2. If new filters or variables are added to the layout .slx file, they must be passed to the layout by `generate_QUAD_Model_Production.m`

3 steps to edit the model features

3. If the inputs and outputs are modified, make the appropriate changes in `define_quadModel_insandouts.m`.

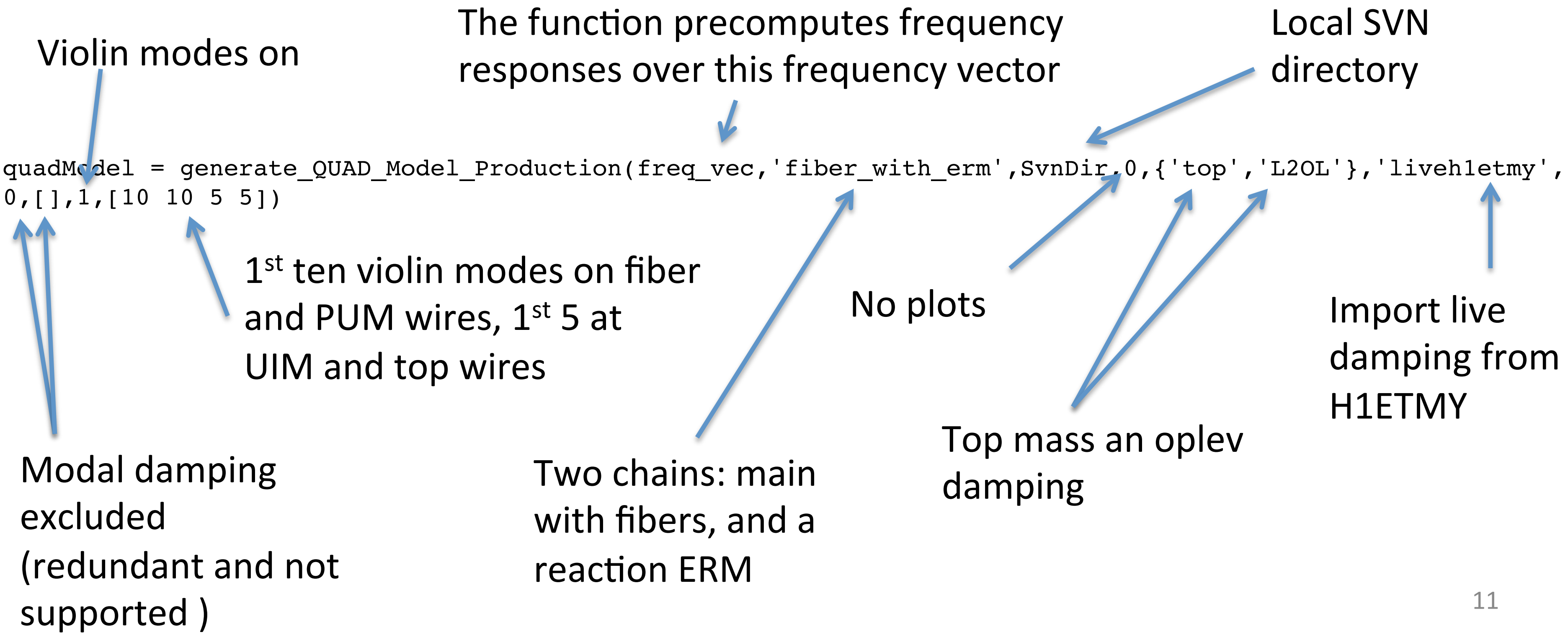
While the Simulink layout file handles all the inputs and outputs for the state space model, and `define_quadModel_insandouts.m` in principle does nothing, it does create a human readable structure array for accessing the dozens of inputs and outputs. Take care in making sure the index numbers here match the layout files.

Note, the violin modes do not appear in the Simulink layout files because they contribute no variables, inputs, or outputs. They only effect the number of states in the state space.

How to call the model

Detailed instructions on how to use this function are commented into the header of `generate_QUAD_Model_Production.m`.

Here is an example of how to call the model with all the features simultaneously.



```
quadModel = generate_QUAD_Model_Production(freq_vec,'fiber_with_erm',SvnDir,0,{'top','L2OL'},'liveh1etmy',
0,[],1,[10 10 5 5])
```

Violin modes on

The function precomputes frequency responses over this frequency vector

Local SVN directory

1st ten violin modes on fiber and PUM wires, 1st 5 at UIM and top wires

No plots

Import live damping from H1ETMY

Top mass an oplev damping

Modal damping excluded (redundant and not supported)

Two chains: main with fibers, and a reaction ERM

Function output from above model call

```
Building a model for a fiber_with_erm QUAD suspension ...
     Using QUAD model:ssmake4pv2eMB5f_fiber
     Using Params file: quadopt_fiber
     Using QUAD model:ssmake4pv2eMB4f_erm
     Using Params file: quadopt_erm
Adding the first 10 fiber violin modes with viscous damping to the model...
Adding the first 10 UIM-PUM wire violin modes with viscous damping to the model...
Adding the first 5 Top-UIM wire violin modes with viscous damping to the model...
Adding the first 5 Top wire violin modes with viscous damping to the model...
Combining main and reaction chain models...
Calculating the frequency response of the open loop model ...
   Finished open loop in 0.07709 seconds.
Closing the loop ...
reading filters for H1ETMY from GPS time 1140933834

loading damping electronics gain of 1.17. This is harcoded in the function with the variable
Damping_electronics_gain!

6 LiveParts found
    M0LiveDampingFilters/M0_DAMP_L :: 4 channels
    M0LiveDampingFilters/M0_DAMP_P :: 4 channels
    M0LiveDampingFilters/M0_DAMP_R :: 4 channels
    M0LiveDampingFilters/M0_DAMP_T :: 4 channels
    M0LiveDampingFilters/M0_DAMP_V :: 4 channels
    M0LiveDampingFilters/M0_DAMP_Y :: 4 channels
Connecting to NDS server nds2.ligo-wa.caltech.edu
Fetching 24 channels, start GPS 1140933834, duration 1 sec
Downloading filter file H1SUSETMY.txt for GPS 1140933835
2 LiveParts found
    L2OplevLiveDampingFilters/L2_OLDAMP_P :: 4 channels
    L2OplevLiveDampingFilters/L2_OLDAMP_Y :: 4 channels
Connecting to NDS server nds2.ligo-wa.caltech.edu
Fetching 8 channels, start GPS 1140933834, duration 1 sec
Reusing results of downloadFilterFile from a previous run (see "help cacheFunction" for details)
6 LiveParts found
    R0LiveDampingFilters/R0_DAMP_L :: 4 channels
    R0LiveDampingFilters/R0_DAMP_P :: 4 channels
    R0LiveDampingFilters/R0_DAMP_R :: 4 channels
    R0LiveDampingFilters/R0_DAMP_T :: 4 channels
    R0LiveDampingFilters/R0_DAMP_V :: 4 channels
    R0LiveDampingFilters/R0_DAMP_Y :: 4 channels
Connecting to NDS server nds2.ligo-wa.caltech.edu
Fetching 24 channels, start GPS 1140933834, duration 1 sec
Reusing results of downloadFilterFile from a previous run (see "help cacheFunction" for details)

Calculating the frequency response of the closed loop model ...
   Finished closed loop in 0.1231 seconds.

quadModel = 

             dampFilters: [1x1 struct]
     mainchain_modelName: 'ssmake4pv2eMB5f_fiber'
    mainchain_paramsName: 'quadopt_fiber'
     reacchain_modelName: 'ssmake4pv2eMB4f_erm'
    reacchain_paramsName: 'quadopt_erm'
           hasFiberModes: '1 through 10 with viscous damping'
    hasUIMPUM_Wire_Modes: '1 through 10 with viscous damping'
    hasTopUIM_Wire_Modes: '1 through 5 with viscous damping'
       hasTop_Wire_Modes: '1 through 5 with viscous damping'
                      ss: [81x75 ss]
    mainchain_pendParams: [1x1 struct]
    reacchain_pendParams: [1x1 struct]
                      in: [1x1 struct]
                     out: [1x1 struct]
                       f: [81x75x2 double]
                dampedin: [1x1 struct]
               dampedout: [1x1 struct]
         maindampFilters: [1x1 struct]
           OLdampFilters: [1x1 struct]
         reacdampFilters: [1x1 struct]
                dampedss: [95x89 ss]
                 dampedf: [95x89x2 double]
```