Projecting Sensor Noise to the Cartesian Basis - T2100336

Brian Lantz

August 2021

1 Introduction

This document contains the scaling factors to convert the noise of individual sensors used by the ISI (see SEI_sensor_noise.m) into the sensor noise for the various cartesian DOFs. This is to document calculations done long ago.

2 Math

This uses the sensor2CART matricies to calculate the noise. The matrices are documented in T1000388, and we use the projections:
{SeismicSVN}/BSC-ISI/Common/Basis_Change_BSC_ISI/aLIGO_BSC_ISI_ITMX_ETMY.mat &
{SeismicSVN}/HAM-ISI/Common/Basis_Change_Matrices/aLIGO_HAM_ISI_4_5_6.mat.
We use these because X and Y (and rX and rY) are symmetric, and because the matrix for the feed-forward L-4Cs is only saved for HAM4 and HAM5. The columns of each matrix are for the 6 sensors, and the rows are the cartesian DOFs. To calculate the noise for a particular DOF, e.g. x, consider the row for that DOF in the projections matrix. The x row is the first row. The cartesian signal C_x is the sum of the 6 sensors (s_{h1}, s_{h2}, etc) and their associated matrix elements $m_{x,h1}$, $m_{x,h2}$, etc.

$$C_x = (m_{x,h1} * s_{h1}) + (m_{x,h2} * s_{h2}) + ... + (m_{x,v3} * s_{v3}) \tag{1}$$

To calculate the noise, we make 2 assumptions. First, we assume that the noise of the 6 sensors is independent, so the noise terms will add in quadrature. Second, we assume that each of the 6 sensors has the same noise, s_n. We see that the noise N_x will therefore be

$$N_x = \sqrt{(m_{x,h1} * s_n)^2 + (m_{x,h2} * s_n)^2 + ... + (m_{x,v3} * s_n)^2} \tag{2}$$

We factor out the sensor noise, and see that there is a simple scaling between the noise of 1 sensor and the noise for the particular DOF.

$$N_x = s_n * \sqrt{m_{x,h1}^2 + m_{x,h2}^2 + ... + m_{x,v3}^2} \tag{3}$$

ie the noise for a cartesian DOF is the square root of the sum of the squares of the row of matrix elements for that DOF. For example, for the stage 1 CPS in the BSC-ISI, The ST1_CPS2CART matrix is

$$\begin{bmatrix} -0.6667 & 0.3333 & 0.3333 & 0.0144 & 0.0240 & -0.0384 \\ 0 & -0.5773 & 0.5773 & -0.0361 & 0.0305 & 0.0056 \\ -0.3467 & -0.3467 & -0.3467 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.3333 & 0.3333 & 0.3333 \\ 0 & 0 & 0 & -0.5980 & 0.5059 & 0.0921 \\ 0 & 0 & 0 & -0.2389 & -0.3984 & 0.6373 \end{bmatrix} \tag{4}$$

so the scaling S_x for the X direction would be

$$S_x = \sqrt{-0.6667^2 + 0.3333^2 + 0.3333^2 + 0.0144^2 + 0.0240^2 + -0.0384^2} = 0.8179 \tag{5}$$

Thus, the CPS noise for the X direction is 0.8179 * the noise of the stage 1 CPS sensor. Because of the symmetry of the platform, the X and Y scalings are the same, as are rX and rY. By inspection we can see that the Z scaling is $\sqrt{1/3}$ as expected.

3 Scaling Values

The scalings for the sensors on the BSC-ISI are

DOF	Stg1 CPS	Stg1 T240	Stg1 L-4C	Stg2 CPS	Stg2 GS-13
X	0.8179	0.6007	0.8189	0.8210	0.8181
Y	0.8179	0.6007	0.8189	0.8210	0.8181
rZ	0.6005	1.0104	0.7332	0.8350	0.7577
Z	0.5773	0.5773	0.5773	0.5773	0.5773
rX	0.7887	1.4289	0.9937	1.2034	1.1906
rY	0.7887	1.4289	0.9937	1.2034	1.1906

The scalings for the sensors on the HAM-ISI are

DOF	Stg0 L-4C	Stg1 CPS	Stg1 GS-13
X	0.8759	0.8172	0.8172
Y	0.9418	0.8172	0.8172
rZ	1.1365	0.8497	0.8497
Z	0.5978	0.5773	0.5773
rX	1.1722	1.0370	1.0370
rY	0.7921	1.0370	1.0370

Note that since the feed-forward L-4Cs on stage 0 of HAM4 and HAM5 are not uniformly distributed about the center of stage 1, the noise projections do not have the same simple structure of the sensors on the suspended stages.

4 Saved files

All the files for this are in the Seismic SVN. The script to do the calculations is:
SeismicSVN/Common/MatlabTools/make_basis_change_noise_scalings.m

The scalings are all saved as a single data structure in the file
SeismicSVN/Common/MatlabTools/SEI_sensor_noise_projections.mat
This data structure looks like
sensor_noise_projections.{bsc/ham}.stg{0/1/2}.*sensor.dof*

sensors are {cps, t240, gs13, l4c}, and the dofs are {x, y, rz, z, rx, ry}
For example, sensor_noise_projections.ham.stg1.gs13.x = 0.8172.

I also added this to the SEI_sensor_noise function. If you call SEI_sensor_noise with the argument 'projections' (or 'proj') it will return the data structure. For Example.

```
>> freq = logspace(-1,2, 1000);
>> cps_noise_1mm = SEI_sensor_noise('ADE_1mm', freq);
>> proj = SEI_sensor_noise('projections');
>> stg1_cps_x_noise = proj.bsc.stg1.cps.x * cps_noise_1mm;
```