EU contract number RII3-CT-2003-506395 CARE-Note-2007-016-ELAN

Description of PLACET compatible ground motion generator

Y. Renier and P. Bambade
LAL, Univ Paris-Sud, IN2P3/CNRS, Orsay, France
B. Bolzon
LAPP-CNRS-IN2P3-Universite de Savoie, Annecy-le-vieux, France

14th August 2007

Abstract

Goals of ATF2 will be to provide nanometer size beams and sub-nanometer stability. To achieve it, simulations of feedback systems should be done based on realistic ground motion generators. A generator which produces output compatible with the PLACET simulations was developed with Matlab to reproduce the vibration spectra measured on the ATF floor at KEK. Spatial coherence between elements was also introduced in an approximate way. This generator is described here.

1 Introduction

The goal is to provide an accurate ground motion generator which can accept as input the positions of the beam-line elements and which produces output compatible with the tracking code PLACET [1]. Measurements made on the ATF ring floor at KEK are available [2] and were used to obtain parameterizations of the spectra in frequency domain and of the spatial coherence for different beam-line element separations. The described program was made with Matlab [3] since it allows easy and efficient data treatment and visualization.

2 Generating noise according to a spectrum

An easy way to obtain n random values in temporal domain according to a given spectrum is to generate n values from a white noise generator, compute the discrete Fourier transform of theses values, multiply the obtained spectrum by the wanted one and compute the inverse discrete Fourier transform of the result. Let's review these steps in more detail.

2.1 Generate white noise

White noise is defined by a constant spectrum in frequency domain. The corresponding generation is not obvious. However, most languages contain by default functions doing it easily (e.g *rand* function in C or Matlab). The need to reproduce sequences is handled by making the initializing seed a parameter of the main program. This can be achieved in Matlab with *rand('seed',seed)*. The generator is run for the wanted number of steps n (cf. Figure 1), independently for the *nb_elmt* elements of the beam line. An array of n by *nb_elmt* random values is hence made with the Matlab function *rand(n,nb_elmt)*.

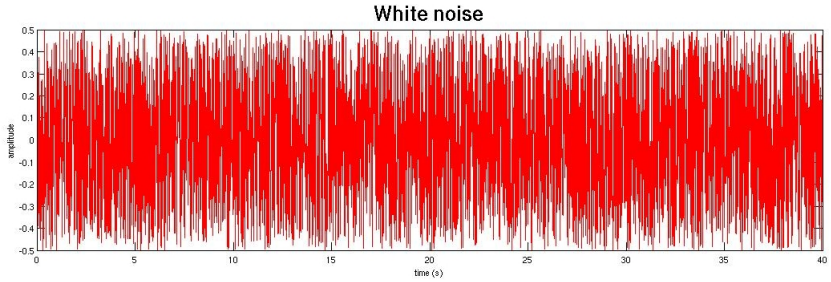

Figure 1: Example of white noise as function of time.

2.2 Transform values according to the wanted spectrum

To modify these white noise spectra, their discrete Fourier transform is first computed,

$$\tilde{X}(k) = \sum_{n=0}^{N-1} x(n) e^{-\frac{i2\pi kn}{N}} \tag{1}$$

using the FFT (Fast Fourier Transform) available in Matlab (cf. Figure 2).

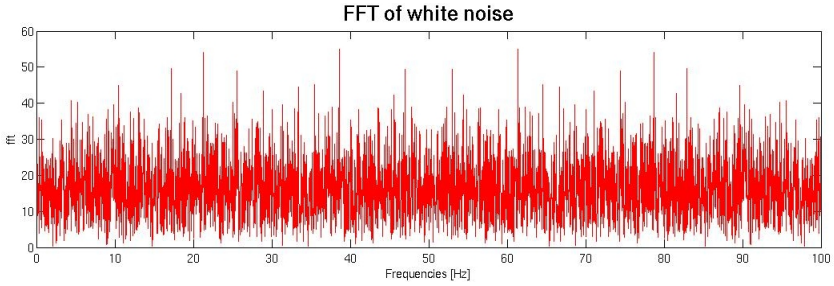

Figure 2: Example of white noise in frequency domain, obtained by taking the FFT of the time sequence in Figure 1.

The calculation yields $\frac{n}{2}$ frequencies with $\Delta f = \frac{1}{n\Delta t}$, so the maximum frequency will be $f_{max} = \frac{1}{2\Delta t}$. As a f_{max} which is too low will distort the simulation, one can divide Δt by $factor$. To maintain the total time span of the simulation, $n \times factor$ values are then needed internally, even if only $\frac{1}{factor}$ of the values in the generated sequence are finally used. During the rest of this

paper, this transformation will be implicit.

Once the FFT is computed, since *rand()* produces values between 0 and 1, the constant term must be set to 0 ($\tilde{X}(k=0)=0$), for it to represent 0-centered variations. The FFT is moreover also normalized ($\tilde{X}(k)$ coefficients) and then multiplied by the FFT of the absolute measured displacements along the y axis ($\tilde{H}(k)$ coefficients) to obtain the final FFT representing the expected vertical vibrations ($\tilde{Y}_{abs}(k)$ coefficients):

$$\tilde{Y}_{abs}(k) = \tilde{H}(k) \cdot \tilde{X}(k) \tag{2}$$

The result of this procedure is shown in Figure 3.

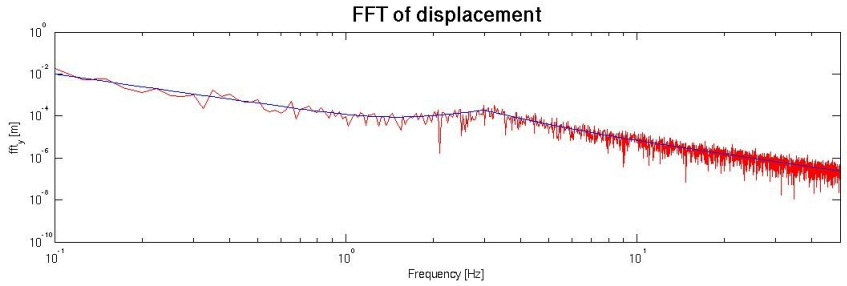

Figure 3: Generated Fourier transform of vertical displacements. The fitted shape of the FFT from the measurements is shown as a continuous line.

To obtain the absolute displacements in temporal domain, the inverse discrete Fourier transform is then computed :

$$y_{abs}(n) = \frac{1}{N} \sum_{k=0}^{N-1} \tilde{Y}_{abs}(k) e^{\frac{i2\pi kn}{N}} \tag{3}$$

using the IFFT (Inverse Fast Fourier Transform) available in Matlab.

Having generated spatially uncorrelated absolute displacements for each element of the beam line with the wanted frequency spectra (cf. Figure 4), the next step is to add coherence between them.

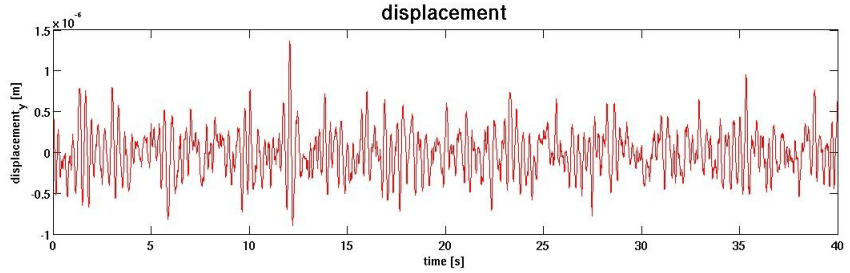

Figure 4: Vertical displacement in temporal domain for one of the beam line elements.

3 Introducing spatial coherence based on measurements

3.1 Coherence between two points

The coherence $C_{y_1 y_2}(k)$ is a real function between 0 and 1 which gives a measure of the correlation between y_1 and y_2 at each frequency ω :

$$C_{y_1,y_2}(k) = \frac{|\tilde{y}_1(k)\tilde{y}_2^*(k)|}{|\tilde{y}_1(k)| \cdot |\tilde{y}_2(k)|} \tag{4}$$

As the functions compared involve the same observable (here, displacements) measured at different places, it is called spatial coherence.
To generate two signals y_1 and y_2 with coherence $C(k)$ and similar Fourier transforms, a way is to start from two uncorrelated ones, y_1' and y_2', with wanted Fourier transforms, and to take :

$$\left\{ \begin{array}{lcl} \tilde{y}_1(k) & = & \tilde{y}_1'(k) \\ \tilde{y}_2(k) & = & C(k)\tilde{y}_1'(k) + (1 - C(k))\tilde{y}_2'(k) \end{array} \right. \tag{5}$$

This linear combination conserves the Fourier transform, and produces the desired coherence (the k-dependence is forgotten for readability):

$$\begin{array}{rcl} C_{y_1,y_2} & = & \dfrac{|\tilde{y}_1\tilde{y}_2^*|}{|\tilde{y}_1| \cdot |\tilde{y}_2|} = \dfrac{|\tilde{y}_1'(C\tilde{y}_1' + (1-C)\tilde{y}_2')^*|}{|\tilde{y}_1'| \cdot |\tilde{y}_2|} \\ & = & \dfrac{\left|C\,|\tilde{y}_1'|^2 + (1-C)\tilde{y}_1'\tilde{y}_2'^*\right|}{|\tilde{y}_1'| \cdot |\tilde{y}_2|} \\ & = & C\dfrac{|\tilde{y}_1'|^2}{|\tilde{y}_1'| \cdot |\tilde{y}_2|} = C\left|\dfrac{\tilde{y}_1}{\tilde{y}_2}\right| \ : \ y_1' \ y_2' \text{ are uncorrelated} \\ & = & C \ : \ y_1 \text{ and } y_2 \text{ have similar Fourier transforms} \end{array}$$

3.2 Coherence between n points

To create n signals $y_1, ..., y_n$ with given coherence between each other ($C_{y_i y_j}$ (i,j) in $[1, n]^2$), the way studied is the extension to n points of the 2-point method presented in section 3.1. One starts from n uncorrelated signals with desired Fourier transforms $(y_1', ..., y_n')$ and take :

$$\tilde{y}_m(k) = \sum_{i=1}^{m} \left[\tilde{y}_i'(k) \cdot \left(C_{y_m,y_i}(k) - C_{y_m,y_{i-1}}(k)\right)\right] \text{ with } C_{y_m,y_0} = 0 \tag{6}$$

where $C_{y_m y_i}(k)$ is the desired coherence between points separated by $y_m - y_i$, obtained by fitting the experimental data. This linear combination can also be shown to conserve the Fourier transform and can be expected to provide some coherence properties. Data generated by this method were analyzed in the same way as the measurements. The coherence found was however not the expected one : it was underestimated for nearby points and overestimated for points far from each other. One can however try to fit the $C_{y_m,y_i}(k)$ functions to at least avoid the cases of overestimation of the coherence. An example of results from applying such a procedure is shown in Figure 5. It can be used to provide simulated vibration data suitable to obtain conservative results.

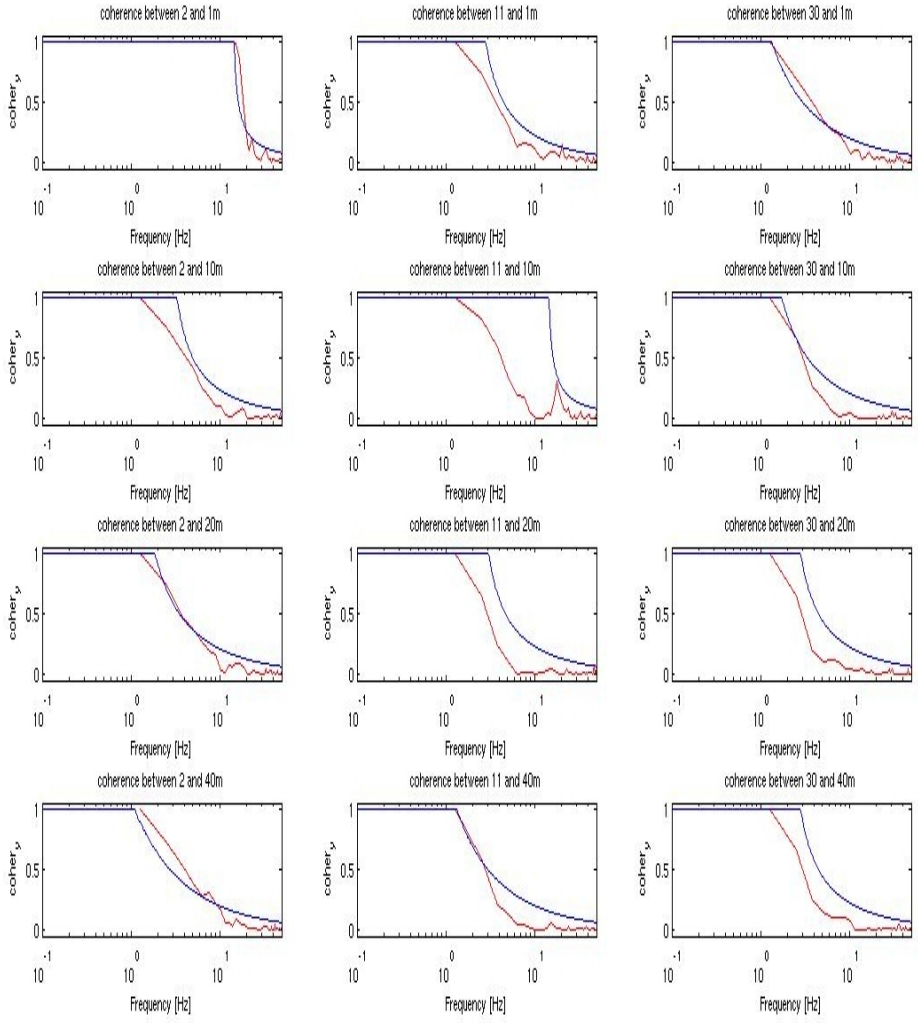

Figure 5: Simulated coherence between various points in red, fitted shape based on measurements in continuous blue line.

4 Conclusion

The developed algorithm reproduces well the frequency dependance, in a simple way. It also provides an approximate method to introduce spatial coherence. This method can be tuned to give reliable results for points far from each other, or near the IP, while avoiding overestimation for all other cases. Hence, simulation results obtained with it can be considered conservative.
Theoretically more rigorous methods, for instance based on 2D FFT, will nevertheless be needed for a better description of the coherence and will be considered next.

5 Acknowledgments

One of us (YR) would like to thank J. Brossard for his help using Matlab. We also would like to thank R. Sugahara for kindly providing the measurements of the ATF floor motion. We acknowledge the support of the European Community-Research Infrastructure Activity under the FP6 Structuring the European Research Area programme (CARE, contract number RII3-CT-2003-506395).

6 References

[1] D. Schulte, A. Latina, N. Leros, P.Eliasson and D'Amico. The Tracking code PLACET, https://savannah.cern.ch/project/placet
[2] R. Suguhara, private communication.
[3] Matlab 7 Programming,
http://www.mathworks.com/access/helpdesk/help/pdf_doc/matlab/matlab_prog.pdf