DIGITAL GENERATION OF NOISE-SIGNALS WITH ARBITRARY CONSTANT OR TIME-VARYING SPECTRA

J. Tückmantel, CERN, Geneva, Switzerland

Abstract

Noise sources in the RF system of an accelerator produce longitudinal emittance increase or particle loss. This noise is inherent, from the beam-control system electronics, external sources or high power components, or can be purposely injected for a specific need such as bunch distribution modification or controlled emittance increase. Simulations to study these effects on the beam require precise reproduction either of the total noise measured on the hardware, or of the noise spectrum to be injected and optimized to produce the desired changes. In the latter case the ‘optimized’ noise source has also to be created in real-time to actually excite the beam via the RF system. This paper describes a new algorithm to create noise spectra of arbitrary spectral density varying with cycle time. It has very good statistical properties and effectively infinite period length, important for long simulation runs. It is spectrally clean and avoids undesired mirror spectra. Coded in C++, it is flexible and fast. Used extensively in simulations it has also successfully created controlled emittance increase in the SPS by the injection of artificial real-time RF noise.

MOTIVATION

In the LHC coast undesired RF noise may blow up bunches and must be avoided. However, added noise is required to blow up bunches during the ramp to produce beam stability in both the SPS and in the LHC. In this case the noise excitation spectrum has to follow the varying synchrotron frequency spectrum. To study these cases, a very flexible digital noise generator was developed for a simulation program [1]. Previously single tailored noise-spectra for blow-up were created by complex hardware means (e.g. [2]). For the blow-up in the SPS with commercial instruments difficulties were encountered due to spectral tails, mirror-spectra and low flexibility. To solve this the digital noise generator was used by uploading its data onto an Arbitrary Wave Generator and then playing back the waveform (for up to 6 s at 10 kHz rate), successfully blowing up the proton bunches in the SPS.

For LHC about 20 min of slow blow-up are needed requiring too much data for direct playback from an AWG. But it could be realized by timesharing with calculation, buffering and output on a single platform; this will even allow spectral corrections in real-time (spectral feed back).

NOISE WITH CONSTANT SPECTRUM

We start with a complex array \( \{g_n\} \) filled with time-domain white noise data for \( 1 \leq n \leq N \), with \( N \) a multiple of 4. The spectrum obtained from \( \{g_n\} \) by Fourier-transform, restricted to positive frequencies exclusively, is weighted with the user supplied spectral (amplitude) weight function. An inverse Fourier-transform results in a set \( \{r_n\} \) of N time-domain data with the desired, still time-invariant, noise spectrum.

The weight function has to be defined in the range 0 to 1; later it will be scaled linearly onto the range \( f_{\text{low}} \) to \( f_{\text{up}} \) for the (possibly time-varying) absolute frequency band. The weight function should be proportional to the square root of the desired local spectral ‘power’ density; absolute amplitudes are irrelevant at this stage.

Practically \( \{g_n\} \) is created by

\[
g_n = \exp(2\pi i \cdot f_n) \cdot \sqrt{-2 \cdot \ln x_{2n+1}}
\]

with \( x_{2n} \) and \( x_{2n+1} \) generated by two independent calls to a top quality pseudo-random generator with equi-distributed output between 0 and 1, the ‘Mersenne Twister’ [3]. The latter has extensive freedom from sequential correlations and a quasi-infinite period length of about \( 10^{6000} \) data samples. The arrays \( \{\text{Re}(g_n)\} \) and \( \{\text{Im}(g_n)\} \) then present (mutually correlated) zero-centred Gaussian distributions with \( \sigma=1 \) (see e.g. [4]).

Fig. 1: FFT of noise data strings designed to target different (amplitude) spectral distribution: rectangular (top-left), trapezoidal (top-right), triangular (bottom-left, as used in the SPS blow-ups) and \( \cos^2 \) (bottom-right). There are sharp spectral ends and no side-lobes nor tails. (Green: linear scale, light blue: log scale; ‘measured’ in ‘sliding average’ mode with rectangular window)

To avoid periodicity and discontinuities for runs with more than \( N \) data, a second set \( \{s_n\} \) is always used in parallel. It is created exactly as \( \{r_n\} \) but with independent random numbers. With a constant parameter \( 0 \leq \psi < 2\pi \) a new variable can be defined: \( u_n = r_n \cos \psi + s_n \sin \psi \). It can be shown [4] that \( \{u_n\} \) has the same spectral properties as its parent arrays. Here \( \psi \) will change as slowly as possible like \( \psi_n=2\pi n/N \), introducing a frequency component corresponding to the lowest one present in the initial
spectra, hence changing the spectral distribution only insignificantly with respect to the user-defined one.

Now at \([n \mod N]=3N/4\) \(\{r_n\}\) and when \(n\) is multiple of \(N\) (i.e. \([n \mod N]=0\) \(\{s_n\}\) can be replaced by a new, statistically independent array without discontinuities nor periodicities in \(\{u_n\}\). Since even for indices only close to the above special ones the concerned weight functions \(\sin \psi\) or \(\cos \psi\) are close to zero, no transients appear at the change of arrays.

This procedure then guarantees, for all practical purposes, an unlimited supply of non-periodic noise data samples of high statistical quality having the desired spectrum on any sequential sub-set.

**NOISE WITH VARIABLE SPECTRUM**

When music, recorded (analogue) on a magnetic tape, is played back faster/slower than recorded, all frequencies appear higher/lower by the ratio of playback speed to recording speed; the same effect is used here in digital realization. Since one has to satisfy all irrational speed ratios, the time domain signal coming from the above Fourier transform has to be interpolated smoothly, or significant phantom side lobes may appear on the spectrum.

In a (discrete) spectral representation the highest represented frequency component \(f_{\text{max}}\) has in time domain only two data samples per oscillation, preventing smooth interpolation up to this frequency. But one can represent the precise equivalent of the present spectrum of \(N\) frequency channels as an \(L \times N\) channel spectrum (with e.g. \(L=8\)) by appending at the high frequency end \((L-1) \times N\) channels with zero amplitude (Fig. 2). Then – after \(L \times N\)-channel Fourier transform to time domain – all initial frequency channels are presented by \(L\) times the number of points per oscillation, allowing good smooth interpolation for the whole frequency range up to \(f_{\text{max}}\). This method is applied here; practically it is incorporated in the creation of the stable spectra of the previous chapter, delivering an \(L\) times denser (complex) data stream ready for smooth interpolation.

Such smooth interpolation with correspondingly chosen step-width allows any ratio of playback to original speed within the required range to be realized (Fig. 3). For a constant time step \(\Delta t\) (i.e. playback rate \(f_{\text{Clock}}=1/\Delta t\)) it is then possible by continuous adjustment of the above interpolation step-width to create a data stream with a spectrum between \(f=0\) and \(f=\Delta f=f_{\text{Up}}(t)-f_{\text{Low}}(t)\) such that its amplitude distribution, \(x\)-scaled from \([0,\Delta f]\) to \([0,1]\), is identical to the user-defined spectral amplitude-function.

![Fig. 3: Left plots: Smoothly interpolated data from the same Fourier transform into time-domain; left-bottom: half the step width of left-top. Right plots: playback with identical time-steps \(\Delta t=1/f_{\text{Clock}}\): ‘bottom procedure’ produces data of half the frequency of ‘top procedure’.

Fig. 2: Left plots: frequency domain. Right plots: time domain, dots→ discrete Fourier transform, lines→ the underlying (smooth) function. Top: original frequency domain representation, bottom: appended frequency domain representation. (displayed \(L=4\), in the code \(L=8\)).

Fig. 4: Screenshot (Polaroid) from a digital signal analyzer: the measured (constant) ‘triangular noise’ (195 Hz – 225 Hz) was created by the digital noise generator and played back from an AWG at 10 kHz rate (applied for bunch blow-up in the SPS in coast).

Before transferring the data, it is scaled by a constant factor such that the rms-value of the total output stream corresponds to a user-defined rms-value \(X_0\); intrinsically this rms-value remains constant even for variations of \(f_{\text{Up}}\) and \(f_{\text{Low}}\). This means also that a change of the spectral bandwidth \(f_{\text{Up}}-f_{\text{Low}}\) causes the local noise ‘power’ to change inversely to the bandwidth scaling-factor. This can
be modified with the time dependent relative amplitude \(a_{rel}(t)\) having the initial (and default) value 1.

The noise data is observed stroboscopically at the rate \(f_{\text{clock}}\), hence any line at frequency \(f < f_{\text{clock}}/2\) has a twin line at \(f = f_{\text{clock}} - f\) \((>f_{\text{clock}}/2)\). To avoid unpredictable results, the desired spectral density function should only be specified for a range between \(0 \leq f_{\text{low}} < f_{\text{up}} < f_{\text{clock}}/2\) (not \(f_{\text{clock}}\)). In any case \(f_{\text{clock}}\) has to be chosen large enough to avoid too much granularity of the output.

### MONOCHROMATIC LINES

In LHC klystrons are used as power amplifiers; these have a strong dependence of the RF phase from the DC voltage. Therefore RF phase noise shows up at multiples of the power grid frequency. The measured phase noise line spectrum has been reconstructed and the beam simulated [5]. Also ‘brushing through a bunch’ with a sliding monochromatic line to produce ‘hollow bunches’ has shown the desired effects on the bunch profile.

To apply this in practice a set of \(M\) monochromatic lines of given (real) amplitude \(a^{(m)}\), frequency \(f^{(m)}\) and starting phase \(\psi^{(m)}\) \((1 \leq m \leq M)\) can be ‘switched on’, alone or superimposed on the other ‘smooth’ noise. Each line starts with the complex status time-domain variable

\[
a^{(m)}_0 = a^{(m)} \cdot \exp(2\pi i \cdot \psi^{(m)})
\]

and advances (positively) at each clock-tick – labelled by the index \(k\) as

\[
a^{(m)}_{k+1} = a^{(m)} \cdot \exp(2\pi i \cdot f^{(m)}/f_{\text{clock}})
\]

All lines can be produced in two modes: either with stable \(f^{(m)}\) – as for power grid multiples – or automatically appearing as an integral part of a varying ‘smooth’ spectrum; in this case \(f^{(m)}\) defines the position with respect to the initial \(f_{\text{low}}\) and \(f_{\text{up}}\). The user may ‘update manually’ any ‘fixed’ \(f^{(m)}\).

Fig. 5: Snapshot of two fixed lines (left) and a variable smooth spectrum with incorporated lines (right). (Green: linear scale, light blue: log scale)

Fig. 6: Graphical User Interface screenshot (digital) as used for the SPS blow-up tests. Top-left: GUI-fields for input specifications of the spectral shape (as trapezium) and three linear interpolation points for \(f_{\text{low}}(t), f_{\text{up}}(t)\) and \(a_{rel}(t)\); bottom-left: [0–1] normalized spectrum, bottom-right: time-domain output representation. The absolute noise-amplitude was adjusted with the AWG output level.

### ACKNOWLEDGMENT

The author wants to thank T. Bohl, T. Linnecar and E. Shaposhnikova for helpful discussions and U. Wehrle for his contributions to the hardware realization and the GUI.

### REFERENCES


[7] G. Papotti et al., this conference TUPP059