How to Shape Noise Spectra for Continuous System Simulation

Andreas Klöckner and Andreas Knoblach and Andreas Heckmann

Noise for continuous-time system simulation is relevant for many applications, where time-domain results are required. Simulating such noise raises the need to consistently shape the frequency content of the signal. However, the methods for this task are not obvious and often state space implementations of form filters are approximated. In this paper, we address the problem with a new method relying on directly using the specified power spectral density for a convolution filter. For the example of railway track irregularities, we explain how to derive the required filters, implement them in the open-source Noise library, and verify the results. The new method produces correct results, is very simple to use, and enables new features for time simulation of physical systems.

Keywords: noise;power spectral density;track irregularity

Citation Notice

Andreas Klöckner, Andreas Knoblach, and Andreas Heckmann. How to shape noise spectra for continuous system simulation. In *Proceedings of the 11th International Modelica Conference*, number 118 in Linköping Electronic Conference Proceedings, pages 411–418, Versailles, France, September 21-23 2015. Linköping University Electronic Press, Linköpings universitet. doi:10.3384/ecp15118411.

```
@INPROCEEDINGS{kloeckner2015how,
  author = {Andreas Kl{\"o}ckner and Andreas Knoblach and Andreas Heckmann},
  title = {How to Shape Noise Spectra for Continuous System Simulation},
  booktitle = {Proceedings of the 11th International Modelica Conference},
  year = {2015},
  number = {118},
  series = {Link{\"o}ping Electronic Conference Proceedings},
  pages = {411-418},
  address = {Versailles, France},
  month = {September 21-23},
  publisher = {Link{\"o}ping University Electronic Press, Link{\"o}pings universitet},
  abstract = {Noise for continuous-time system simulation is relevant for many applications,
    where time-domain results are required. Simulating such noise raises
    the need to consistently shape the frequency content of the signal.
    However, the methods for this task are not obvious and often state
    space implementations of form filters are approximated. In this paper,
    we address the problem with a new method relying on directly using
    the specified power spectral density for a convolution filter. For
    the example of railway track irregularities, we explain how to derive
    the required filters, implement them in the open-source Noise library,
    and verify the results. The new method produces correct results,
    is very simple to use, and enables new features for time simulation
    of physical systems.},
  comment = {ISBN 978-91-7685-955-1, ISSN (print) 1650-3686, ISSN (online) 1650-3740},
  doi = {10.3384/ecp15118411},
  keywords = {noise;power spectral density;track irregularity}
}
```

How to Shape Noise Spectra for Continuous System Simulation

Andreas Klöckner[1] Andreas Knoblach[1] Andreas Heckmann[1]

[1]DLR German Aerospace Center, Institute of System Dynamics and Control, 82234 Weßling, Germany,
andreas.{kloeckner,knoblach,heckmann}@dlr.de

Abstract

Noise for continuous-time system simulation is relevant for many applications, where time-domain results are required. Simulating such noise raises the need to consistently shape the frequency content of the signal. However, the methods for this task are not obvious and often form filters are approximated by state space implementations. In this paper, we address the problem with a new method relying on directly using the specified power spectral density for a convolution filter. For the example of railway track irregularities, we explain how to derive the required filters, implement them in the open-source `Noise` library, and verify the results. The new method produces correct results, is very simple to use, and enables new features for time simulation of physical systems.

Keywords: Noise, power spectral density, track irregularity

1 Introduction

Modeling stochastic signals is of interest in a wide range of applications, such as sensor modeling, aerodynamic turbulence, and rail irregularities. Previous Modelica libraries, such as the `Statistics` library (Haase et al., 2008), allow to precisely define statistical properties of such signals. However, other properties of the noise signals such as the underlying random number generator or the signal's frequency content could not be modeled as conveniently. A Modelica `Noise` library has thus recently been released in order to enable the engineer to conveniently and consistently define noise signals (Klöckner et al., 2014). It is intended to include a subset of sampled noise generators and standard distributions in the Modelica standard library. The remaining functionality will still be available in the `AdvancedNoise` library.

The `Noise` library also introduces a new class of random number generators: DIRCS Immediate Random with Continuous Seed allows to generate random numbers from an input signal without internal states. It thus eliminates the need for time-events, but can be used to generate a random signal directly from the `time` variable. This has been shown to positively affect the simulation performance (van der Linden et al., 2015). Additionally, it allows to define noise signals in dimensions other than the time. This is advantageous in several applications. Rail irregularities e.g. are typically defined with respect to the location on the track. Turbulence models used in aviation also assume a static wind field flown through by the aircraft.

Additionally, the frequency content of noise input to a system must be carefully modeled. It is usually specified by a power spectral density (PSD). If a linear time invariant (LTI) system model is considered, the PSD can be applied in the frequency domain by multiplying the PSD with the squared transfer function of the model. See Frederich (1984) for a railway application and EASA CS-25 (2013) for a typical aircraft application. If a nonlinear model has to be simulated in the time domain, a suitable filter transfer function must be derived from the PSD. In the case, that the PSD is a rational function w.r.t. to the *squared frequency*, a spectral factorization of the PSD can be derived analytically. See Liepmann (1952) for an aeronautical example. Otherwise, the PSD must be approximated by a suitable function. An alternative approach is the recently developed Fractional-Order Modeling Toolbox for Modelica (Pollok et al., 2015), which allows to simulate also non rational transfer functions.

In summary, it is not at all obvious how to parametrize these frequency properties. We thus present a systematic method to shape the frequency content of noise signals. The contributions of this paper are as follows:

1. Using the example of rail irregularities, we summarize how noise is typically specified.
2. We then shortly define the probability distribution of the noise signals generated in this paper.
3. Starting from a given PSD we rigorously derive a way to shape this frequency content onto a noise signal. This method will turn out to be perfectly simple to use and to be applicable to almost any kind of noise spectrum.
4. We finally implement the approach and verify that it yields the same results as conventional methods.

2 Railway track irregularities

Besides safety and operating efficiency, it is an essential goal of railway vehicle design to provide an accepted level of vibration comfort. In order to take human perception into account, different methods and standards exist for passenger comfort assessment. However all of these rely on the accelerations experienced by the passengers as input information.

From a vehicle dynamical point of view these accelerations are the result of forced vibrations of the vehicle/track-system that are excited by track irregularities. Frederich (1984) analyzed a large number of track measurements and introduced representative PSDs for good, average and bad tracks, see Fig. 1. Note, these numbers quantify the irregularity per meter track length or with respect to the spatial frequency (unit: $1/\mathrm{m}$), respectively, and have to be transfered into the time domain taking the vehicle speed into account, see e.g. (Popp and Schiehlen, 2010).

Regarding the vehicle/track system that is excited by the track unevenness we confine ourselves to vertical dynamics and use the simplified quarter car model shown in Fig. 2. The excitation input is introduced as a variable track height z defined as a stochastic function of the longitudinal track position. The wheel/rail contact is represented by a stiff but linear spring/damper system. Rail and its support constitute a dynamical subsystem on the track side of the model, suspension and car body form the vehicle subsystem. The acceleration of the car body a is the output quantity of the model. The resulting Bode diagram is depicted in Fig. 3.

Here, the model is defined linear by intention. Presuming a constant running speed of the vehicle, the acceleration response of the car body can be evaluated in the frequency domain (Knothe and Stichel, 2003, Ch. 6), which provides the opportunity of comparison and validation with results from time domain simulations in Modelica. Fig. 4 presents the pure frequency domain results, that are based on the excitation by a track of all three qualities.

The results presented in this paper are confined to linear systems in order to validate the time-domain simulation approach with a well know frequency domain solution. However, this limitation can be dropped, once the results from time domain simulations with appropriately shaped noise spectra is validated. Time domain simulations are then available for non-linear systems, are capable of running with variable speed and may consider singular disturbances such as running over railway switches as well.

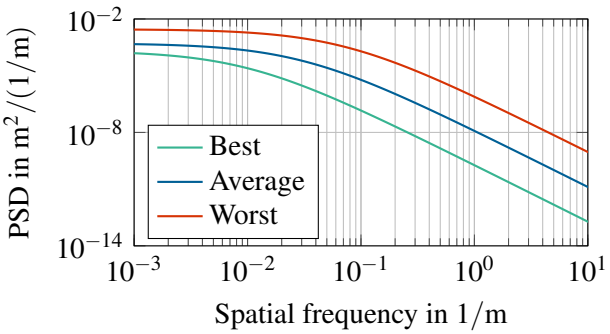


Figure 1. Representative track irregularity PSDs (Frederich, 1984).

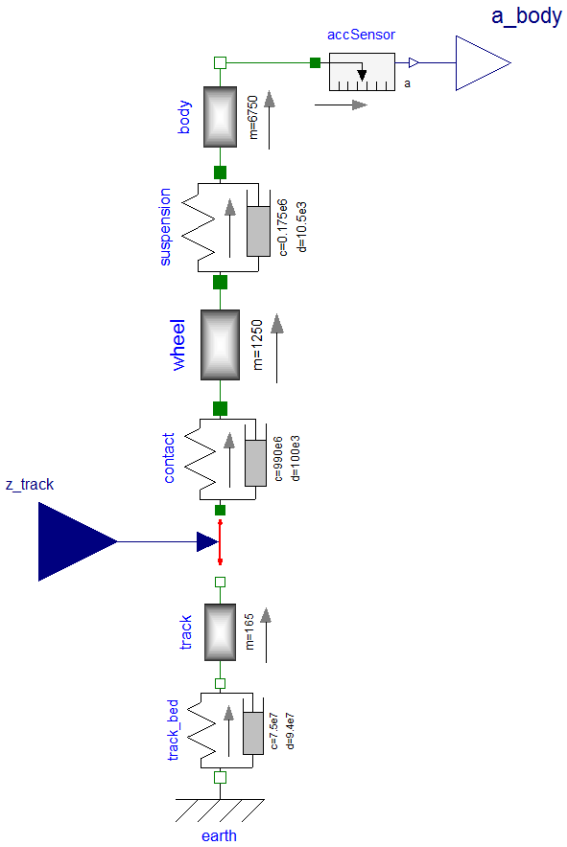


Figure 2. Simplified quarter car model of a railway vehicle in Modelica.

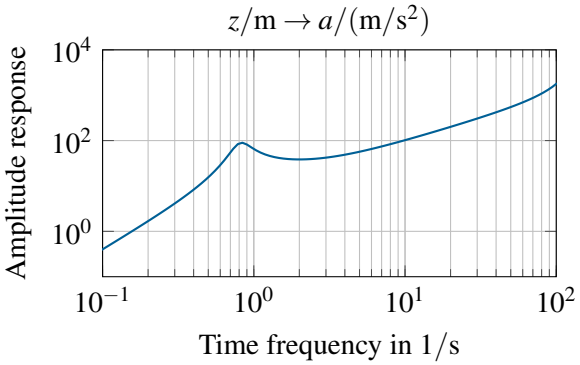


Figure 3. Amplitude response from track irregularity (in m) to body acceleration (in $\mathrm{m/s^2}$).

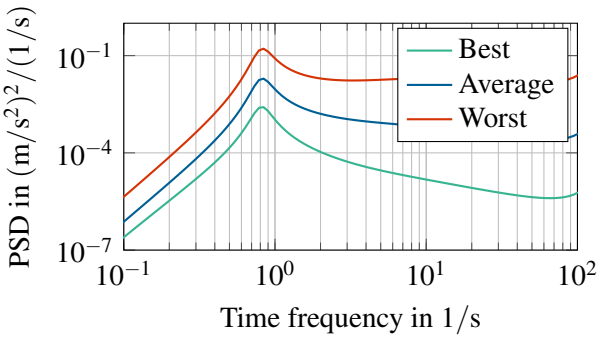


Figure 4. PSD of the body acceleration a for different track irregularities at a velocity of $v = 100\,\mathrm{m/s}$

3 Review of `Noise` parameters

The general degrees of freedom in parameterizing noise have been described in detail earlier (Klöckner et al., 2014). A noise signal can be specified in three steps:

1. Select a random number generator, which generates uniformly distributed random numbers with certain statistical properties, such as subsequent numbers being independent from each other.

2. Transform the uniformly distributed random numbers in order to match a given probability density function, such as for a normal distribution.

3. Interpolate the resulting stream of correctly distributed random numbers.

The random number generators of the `xorshift` family (Vigna, 2014) have been included in the `Noise` library since its last release.[1] These generators have very strong statistical and computational properties and are thus used in the library without exception.

In all cases, where unsampled random numbers are required, the DIRCS generator is used. This random number generator does not require a state but generates a random number directly from a `double` input signal. To this end, the `xorshift64*` algorithm is first initialized with the `double` input signal casted to two `integer` values. After ten iterations, the output of `xorshift64*` is used to seed an `xorshift128+` generator for a final iteration. In this way, high quality random numbers are produced by the efficient `xorshift` generators with a low computational effort for arbitrary input values.

The standard normal distribution is chosen for all random numbers generated in this work. This does not allow to reproduce effects commonly found in measurement noise, such as discretization. However, the choice is reasonable when complex filters are used to shape the actual noise signal to be used in the simulation. Typical filter parameterizations for rail irregularities e.g. assume standard normal distributions of their input signals (SIM, 2003). Additionally, the subsequent interpolation relies on computing the weighted sum of consequent random numbers. Following the central limit theorem, the result will inevitably be shaped towards the normal distribution.

In previous work, we have described three distinct interpolation functions for noise signals. These include piece-wise constant and linear interpolations as well a smooth interpolation using the sinc function. The interpolations yield a continuous-time random signal $r(t)$ by computing the sum of consequent random numbers w_i, weighted with an admissible kernel function $k(t)$:

$$r(t) = \sum_{-n}^{+n} w_i \cdot k(t - i\Delta t), \tag{1}$$

with

$$k(i\Delta t) \stackrel{!}{=} \begin{cases} 1 & \text{if } i = 0 \\ 0 & \text{if } i \neq 0. \end{cases} \tag{2}$$

In this equation, Δt is the sample period of the random numbers and the interpolation base n has to be chosen according to the selected kernel $k(t)$.

However, for the more general case of a given PSD, the final interpolation step has to be replaced by a more powerful approach as described in the following sections.

4 Application of a given PSD

As already explained in Section 2, instead of (band-limited) white noise, colored noise is required for most practical applications. The required frequency content of the noise signal is usually specified by a given PSD $\Phi(f)$. In order to apply this PSD to a raw white noise signal with piece-wise constant interpolation a linear form filter is typically applied (SIM, 2003, VIII-TE:8). This filter $H(f)$ defines a mapping in the frequency domain between the white noise input vector $w(t)$ and the colored noise output vector $r(t)$:

$$R(f) = H(f)W(f) \tag{3}$$

where $R(f)$ and $W(f)$ are the Fourier transforms (FTs) of $r(t)$ and $w(t)$. White noise is defined by its flat PSD of $W(f) \equiv 1$. If the filter $H(f)$ is applied to white noise, the PSD of the colored noise is thus simply

$$\Phi_r(f) = |R(f)|^2 \equiv |H(f)|^2. \tag{4}$$

In order to shape colored noise to a given PSD, the required filter is hence constrained by

$$|H(f)|^2 \stackrel{!}{=} \Phi(f). \tag{5}$$

In practice, the filter is typically applied by fitting a rational transfer function on $\Phi(f)$ which is then simulated as an additional linear block in the model (see Section 4.1). An alternative exploiting the interpolation kernel from Eq. (1) is proposed in Section 4.2.

[1] `https://github.com/DLR-SR/Noise`

4.1 Using a transfer function

Before the approximation of a given PSD with a rational transfer function is explained, important properties are briefly repeated. In principle, the filter $H(f)$ is restricted to rational functions which are usually expressed w.r.t. the Laplace variable $s = d + 2\pi \mathrm{j} f$, i.e.

$$H(s) = \frac{N(s)}{D(s)} = \frac{\sum_{k=0}^{n_z} a_k s^k}{\sum_{l=0}^{n_p} b_l s^l}. \tag{6}$$

The coefficients a_k of the numerator $N(s)$ and the coefficients b_l of the denominator $D(s)$ are real numbers. Both, the numerator and denominator can be factorized which leads to

$$H(s) = \frac{\prod_{k=1}^{n_z} (s - z_k)}{\prod_{l=1}^{n_p} (s - p_l)}. \tag{7}$$

Every zero z_k and every pole p_l is either a real number or two zeros (or poles) are each a complex conjugate pair. A necessary condition to express $H(s)$ in a state space representation is that $H(s)$ is proper, i.e. the number of poles n_p is greater than or equal to the number of zeros n_z. If the real part of all poles and zeros is negative, a transfer function is called minimum phase.[2]

In addition to constraint (5), $H(s)$ must be proper and minimum phase, in order to be realizable by a state space system. Because a suitable transfer function $H(s)$ cannot be analytically computed from a given PSD $\Phi(f)$ in general, a least squares fit is performed:

$$\min_{a_k, b_l} \sum_{i}^{n_f} \left(|H(2\pi \mathrm{j} f)| - \sqrt{\Phi(f)} \right)^2. \tag{8}$$

The coefficients a_k and b_l are chosen as decision variables because they are real numbers and independent from each other. In order to ensure that the filter is proper, $n_z = n_p - 1$ is chosen. The optimization is pursued with MOPS (see Joos et al., 2002) and a Levenberg-Marquardt algorithm is used.

Finally, the minimum phase requirement is fulfilled by a subordinate step. To that end, the zeros z_k and poles p_l of the optimal solution are computed. Afterwards, the real part of every pole/zero is mirrored into the left half plane, e.g.

$$\bar{p}_i = -|\Re(p_i)| + \Im(p_i). \tag{9}$$

Note that the latter operation alters only the phase but not the amplitude of $H(s)$.

[2] Minimum phase means that both the filter and its inverse are stable and causal.

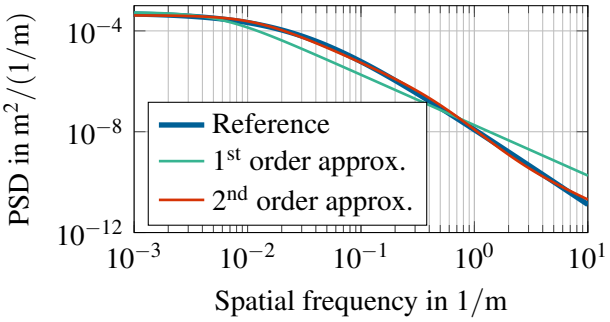


Figure 5. Approximation of the average track irregularity PSD with a first order and a second order filter. The second order filter shows a good fit to the reference.

Figure 5 compares the reference PSD to the PSDs of a first and second order filter. It can be seen that the second order filter is a very good approximation of the average track irregularity. This filter can thus be used to implement the form filter.

4.2 Using the interpolation kernel

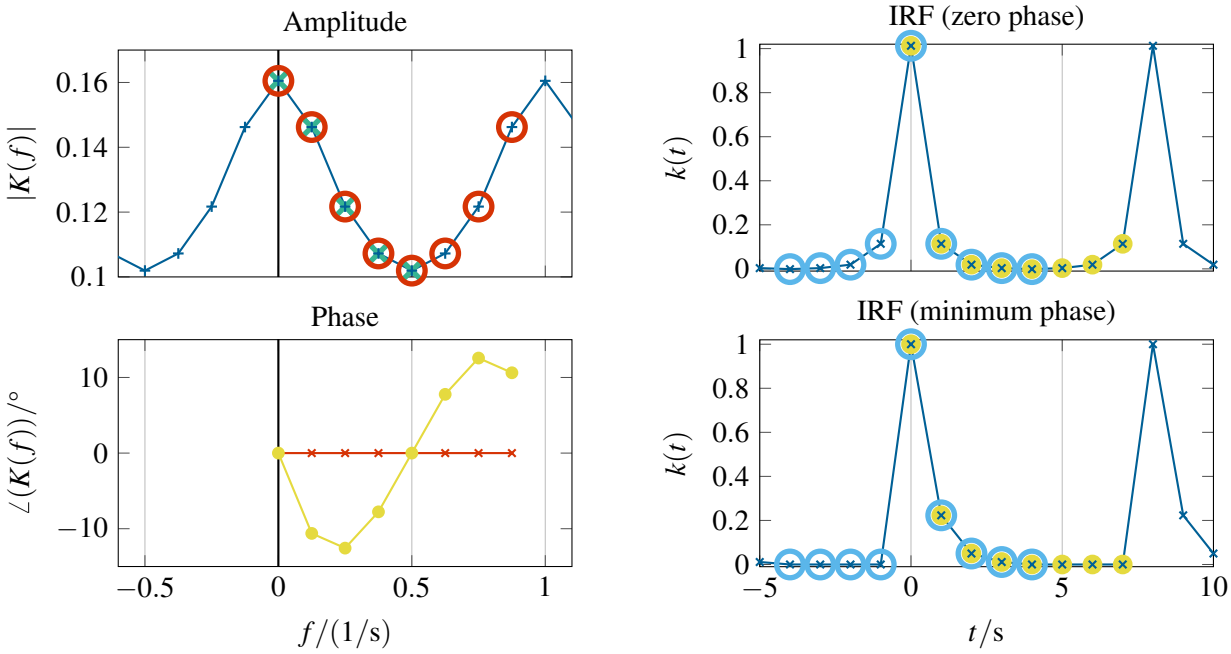
The filter or transfer function is typically implemented using continuous-time states in Modelica. This approach has two major drawbacks: First, expressing the filter in the time domain limits the simulation to a fixed velocity in order to map the location to a time-domain filter. Second, the additional states of the filter require the raw noise signal to be generated accurately using events, which considerably slows down simulation, even if only low accuracy is required. In this paper, we introduce a different approach to shaping the frequency content of the noise signal using the interpolation kernel $k(t)$ from Eq. (1).

4.2.1 Theoretical background

The idea is based on the convolution theorem. It relates the continuous-time integration of the filter states to a convolution integral of the raw noise signal $w_i(t)$ with the filter's IRF $h(t)$. Exploiting the piece-wise constant noise signal, this approach can be further reduced to a sum of weighted random numbers w_i:

$$\begin{aligned} R(f) &= W(f)H(f) \\ &\bullet\!\!-\!\!\circ \\ r(t) &= w_i(t) * h(t) \\ &= \int_{-\infty}^{+\infty} w_i(t) \cdot h(t-\tau) d\tau \\ &= \sum_{-\infty}^{+\infty} \left(w_i \cdot \int_{i\Delta t}^{(i+1)\Delta t} h(t-\tau) d\tau \right). \end{aligned} \tag{10}$$

The weights in (10) are specified by the integral of the IRF $h(t)$. This integral can also be expressed as the step response $\varsigma(t)$ of the filter. Assuming a stable filter,



(a) Frequency domain: In the upper plot, the selection of the correct samples for the iFT and Hilbert transform is illustrated. The given amplitude data (×) is mirrored at $f = 0$ and periodically repeated (—+—). Afterwards, the highlighted samples (○) are chosen. In the lower plot, the zero phase (—×—) and the minimum phase (—●—) are depicted.

(b) Time domain: The resulting IRF for the zero phase (upper plot) and minimum phase (lower plot) are depicted. The results from the iFT (●) are periodically repeated (—×—) in order to chose the correct samples (○).

Figure 6. The selection of the correct samples is illustrated in the time domain and in the frequency domain.

the convolution can finally be approximated using a truncated sum:

$$\begin{aligned} r(t) &= \sum_{i=-\infty}^{+\infty} w_i \cdot \big(\varsigma(t-(i+1)\Delta t) - \varsigma(t-i\Delta t)\big) \\ &\approx \sum_{i=\lfloor t/\Delta t \rfloor - n}^{\lfloor t/\Delta t \rfloor + n} w_i \cdot \underbrace{\big(\varsigma(t-(i+1)\Delta t) - \varsigma(t-i\Delta t)\big)}_{:=k(t)}. \end{aligned} \quad (11)$$

Using this approach, all continuous and discrete states can be eliminated from the noise generation. This was shown before to be advantageous for the simulation performance (van der Linden et al., 2015). Additionally, the interpolation kernel $k(t)$ can be shaped using an arbitrary filter, if its step response is known.

4.2.2 Computation of the IRF

As we have seen, only a suitable step response is required to shape the desired frequency content. This IRF can easily be obtained from the transfer function derived in Section 4.1. However, in order to avoid the approximation with a rational function, the IRF is directly computed from the PSD using the framework of Fourier transform (FT) and inverse Fourier transform (iFT). The resulting IRF is then numerically integrated in order to yield the step response.

Because the PSD describes only the amplitude of the filter and because the filter must be not realized in state space representation it is possible to use the phase as an additional degree of freedom. Here, two different phases are considered: zero phase and minimum phase.

Zero phase: First the zero phase case is considered. For this case all phase are set to zero. The FT of the interpolation kernel is hence simply

$$K(f) = \sqrt{\Phi(f)}. \quad (12)$$

However, for a correct application of available FT algorithms, the frequency samples must be chosen carefully: In order to yield a real valued $k(t)$, $K(f) = \mathrm{conj}(S(-f))$ must hold. It is further helpful to remember that – because time and frequency are both discretized – $K(f)$ and $k(t)$ are periodically repeated. This is illustrated in Fig. 6a for a simple example and the correct samples are marked.

After the iFT, the resulting $k(t)$ is periodically repeated, too. This allows to chose the correct samples as depicted in Fig. 6b. As it can be further seen, the zero phase yields a non-causal IRF which is symmetric to $t = 0$.

Minimum phase: Second, the minimum phase case is considered. In this case, only the amplitude of the FT of the interpolation kernel is given by the PSD:

$$|K(f)| = \sqrt{\Phi(f)}. \quad (13)$$

The minimum phase $\angle(K(f))$ can be computed using the Hilbert transform. For the Hilbert transform, the same samples must be chosen as for the iFT. The resulting minimum phase is depicted in Fig. 6a. Afterwards, the full FT of the interpolation kernel is given by

$$K(f) = |K(f)| \cdot \exp(\mathrm{j} \cdot \angle(K(f))). \tag{14}$$

The transformation into the time domain is subsequently performed in the same way as for the zero phase case. As it can be seen in Fig. 6b, the minimum phase filter is causal, i.e. its IRF is non-zero only for non-negative times $t \geq 0$.

Figure 7 compares IRFs for the average track irregularities as obtained from the procedure outlined above. First, the IRF of the fitted second order filter is evaluated by simulation and by iFT of the PSD shown in Fig. 5. Both results are essentially the same, showing the correct iFT application. The IRFs obtained directly from the given PSD are also shown. The minimum phase IRF is very similar to the fitted filter's IRF, underlining the good fit of the filter. The zero phase IRF is non-causal, as it is non-zero for negative times.

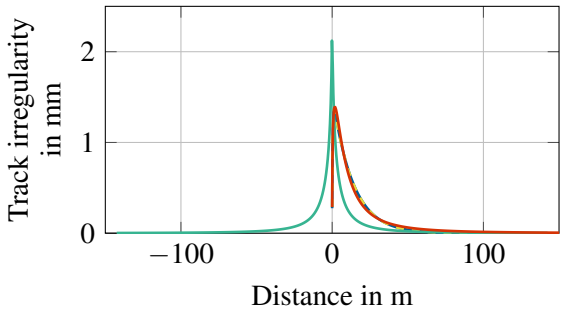


Figure 7. Comparison of different IRF: The IRF of the 2nd order filter is obtained by simulation (——) and by iFT (- - - -). An excellent agreement can be recognized. Additionally, the minimum phase (——) and zero phase (——) IRF are depicted. These correspond directly to the specified PSD.

5 Results

The form filters for average track irregularities are implemented using the `Noise` library according to the procedures outlined above. Using the Dymola 2016 RC2 simulation tool with its DASSL solver, the different steps of the implementation are then verified. To this end, the following simulation experiments are presented:

1. The minimum phase convolution is verified against a state space filter implementation with identical white noise input.
2. The white noise input of the state space implementation is exchanged by an independent noise source not using the DIRCS algorithm.
3. The minimum phase and zero phase IRFs are compared to each other.
4. The quarter car model of the railway vehicle is fed with the zero phase noise convolution filter and compared against the standard frequency domain solution.

5.1 Convolution verification

Figure 8 shows a comparison of the fitted second order filter's state space implementation with its minimum phase convolution implementation. Both filters are driven by identical white noise generated with the DIRCS generator directly from the position on the track. The raw random numbers are generated with a sample period of $\Delta x = 0.4\,\mathrm{m}$ and a standard deviation of

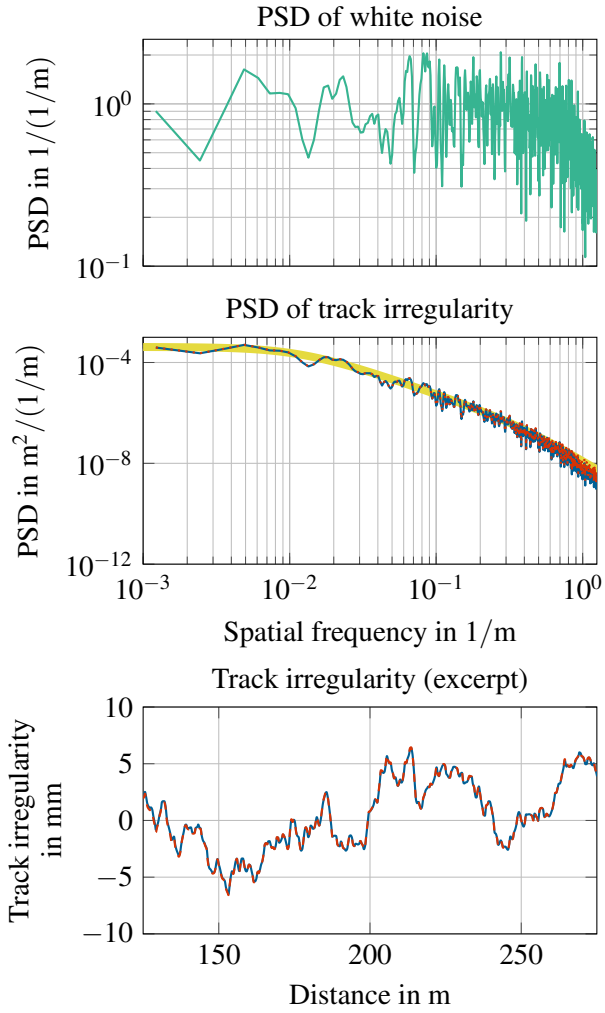


Figure 8. The identical white noise (upper plot) is applied to the second order filter by simulation (——) and by convolution (·····). The PSD of both implementations (middle plot) are identical and correspond well with the reference PSD (——). As it can be seen in the lower plot, the signals are also identical.

$\sigma = \sqrt{0.5/\Delta t}$. It can be seen that the white noise spectrum has indeed a spectral density of 1; except for its random nature. The shape of the white noise spectrum directly corresponds to the shape of the track irregularity PSDs. Both irregularity PSDs can be seen to be identical. This is also the case for the actual simulation output z of the track irregularity.

5.2 Space domain noise verification

In the second step, the minimum phase convolution implementation is compared to a traditional state space implementation fed by a time domain sampled noise. In order to yield comparable results, the time domain noise is sampled with $\Delta t = \Delta x / v$. The standard deviation of the normal distribution is kept the same. Figure 9 compares both results to the reference. Good agreements can be observed in both time frequency domain and spatial frequency domain PSDs. Timing comparisons are not presented, as the current convolution implementation is not yet optimized.

5.3 Minimum and zero phase filters

The comparison of the minimum phase and zero phase convolution implementations can be seen in Fig. 10. The white noise generator is DIRCS in both cases and the sample periods are both $\Delta x = 0.4\,\mathrm{m}$. Both IRFs have a resolution of 0.1 m. Both variants agree very well with the reference in the low frequencies. At very high frequencies, the influence of the two sampling periods are marked with vertical lines. Characteristic marks on the PSDs can be seen at these lines. The respective sampling periods must thus always be chosen high enough to resolve all relevant effects.

5.4 Quarter car railway vehicle

Finally, the zero phase convolution implementation is integrated with the quarter car model of a railway vehicle running at $v = 100\,\mathrm{m/s}$. Figure 11 compares the acceleration of the car body from this simulation to the well trusted frequency domain solution. A very good agreement can be seen.

6 Conclusions

In this paper, we show how to consistently shape a noise signal to match a given frequency content specified by a PSD. The method is introduced at hand of the quarter car model of a railway vehicle, for which well trusted reference solutions are available.

Our method uses the non-recursive random number generator DIRCS. It generates normally distributed random numbers directly from the current location on the track. We then shape the frequency content of the random numbers using a convolution with the impulse response of a form filter. It is shown that the IRF of the form filter can be directly generated from the given PSD using the inverse Fourier transform.

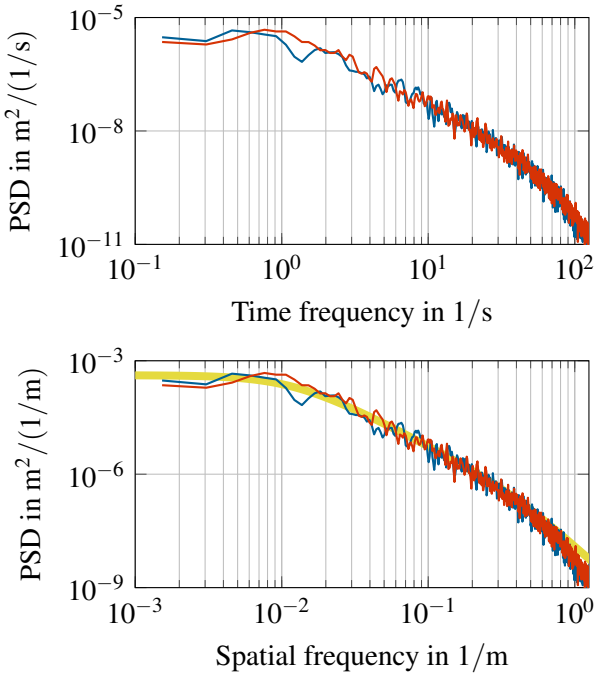


Figure 9. The PSD of the track irregularity from simulation (——) and from convolution (——) are compared in the time frequency domain (upper plot) and in the spatial frequency domain (lower plot). Both agree very well. In the spatial frequency domain, an excellent agreement with the reference (——) can be also seen.

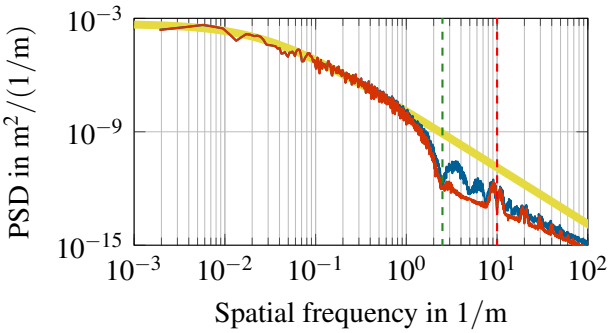


Figure 10. The PSD resulting from the minimum phase (——) and from the zero phase (——) iFT agree well with the reference (——).

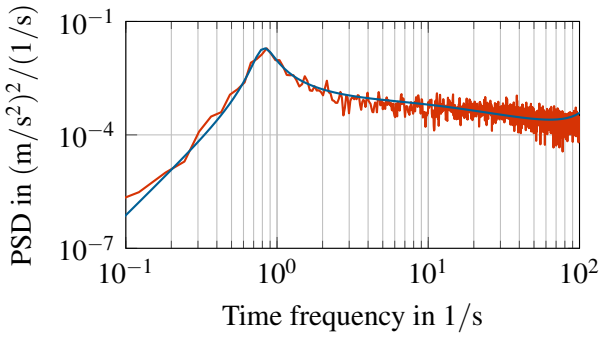


Figure 11. The frequency domain solution is compared (——) to simulation results (——).

The method is implemented in the Modelica `Noise` library and validated against the given spectrum. It is then used to excite a linear quarter car model running at a constant speed. Results obtained with our method agree very well with the standard solutions based on frequency domain computations.

Our method is thus shown to produce correct results. Additionally, it can be employed not only for linear models, but also for non-linear models, vehicles running at varying speed, or in more complex scenarios involving e.g. singular disturbances. Moreover, the method proposed in this paper is perfectly straightforward to use and can also be applied to a variety of problems, such as turbulence or street roughness.

The current implementation is based on the open-source `Noise` library. This makes the method available to a wide audience and also gives room for further improvements. Since our implementation of the convolution has not yet been optimized, further investigations should take into account timing measures. Extensions of the method could possibly be found in higher dimensional noise spectra, correlated noise, or additional effects such as discretization.

Reproducible research

The results of this paper can be reproduced using the code which will be made available on `http://dlr-sr.github.io`.

Acknowledgments

We thank our parents for giving us a great first name.

References

EASA CS-25. Certification specifications and acceptable means of compliance for large aeroplanes, 2013.

Fritz Frederich. Die Gleislage aus fahrzeugtechnischer Sicht. *ZEV–Glasers Annalen*, pages 108–1984, 1984.

Joachim Haase, S Wolf, and C Clauß. Monte carlo simulation with modelica. In *6th International Modelica Conference*, pages 601–604, Bielefeld, Germany, 2008.

H.D. Joos, J. Bals, G. Looye, K. Schnepper, and A. Varga. A multi-objective optimisation-based software environment for control systems design. In *Conference on Computer-Aided Control Systems Design*. IEEE, 2002.

Andreas Klöckner, Franciscus L. J. van der Linden, and Dirk Zimmer. Noise generation for continuous system simulation. In Hubertus Tummescheit and Karl-Erik Årzén, editors, *Proceedings of the 10th International Modelica Conference*, number 96 in Linköping Electronic Conference Proceedings, pages 837–846, Lund, Sweden, March 10-12 2014. Modelica Association and Linköping University Electronic Press. doi:10.3384/ECP14096837. ISBN: 978-91-7519-380-9. ISSN: 1650-3686. eISSN: 1650-3740.

K. Knothe and S. Stichel. *Schienenfahrzeugdynamik*. Springer, Berlin, 2003.

Hans Wolfgang Liepmann. On the application of statistical concepts to the buffeting problem. *Journal of the Aeronautical Sciences (Institute of the Aeronautical Sciences)*, 19(12), 1952. doi:10.2514/8.2491.

Alexander Pollok, Dirk Zimmer, and Francesco Casella. Fractional-order modelling in modelica. accepted and to be presented at the 11th International Modelica Conference, 2015.

K. Popp and W. Schiehlen. *Ground Vehicle Dynamics*. Springer, 2010.

SIMPACK Time Excitations Catalogue. SIMPACK, 24th September 2003. SIMDOC v8.607.

Franciscus L. J. van der Linden, Andreas Klöckner, and Dirk Zimmer. Effects of event-free noise signals on continuous-time simulation perfomance. In Felix Breitenecker, Andreas Kugi, and Inge Troch, editors, *8th Vienna International Conference on Mathematical Modelling*, volume 8 of *Mathematical Modelling*, pages 280–285, Vienna, Austria, 2015. IFAC-PapersOnLine. doi:10.3182/20150218-3-AU-30250.

Sebastiano Vigna. Further scramblings of marsaglia's xorshift generators. *CoRR*, abs/1404.0390, 2014. URL `http://arxiv.org/abs/1404.0390`. Code available at http://xorshift.di.unimi.it/.