

Task 2 – Sampling-Theory Studio

Introduction: Sampling an analog signal is a crucial step for any digital signal processing system. The Nyquist–Shannon sampling theorem guarantees a full recovery of the signal when sampling with a frequency larger than or equal to the bandwidth of the signal (or double the maximum frequency in case of real signals).

Description: Develop a desktop application that illustrates the signal sampling and recovery showing the importance and validation of the Nyquist rate. Your application should have the following features:

- **Sample & Recover:** Allow the user to load a mid-length signal (around 1000 points length), visualize and sample it via different frequencies, then use the sampled points to recover the original signal using Whittaker–Shannon interpolation formula (please, refer to [this wiki page](#) for more information). Sampling frequency should be shown in either actual frequency or normalized one (with respect to the maximum frequency, i.e. ranges from $0 \times f_{\max}$ to $4 \times f_{\max}$ for example). You should use four graphs, one for displaying the original signal along with the markers for the sampled points, another one to display the reconstructed signal, a third one to display the difference between the original signal and the reconstructed one, and lastly the fourth one to display the frequency domain to allow the user to inspect if there is any aliasing happening or not. Watch out how you will arrange these four graphs in your software in a convenient way for the user.
- **Load & Compose:** The loaded signal can come from a file or a signal mixer/composer in your application. In the signal mixer, the user can add multiple sinusoidal signals of different frequencies and magnitudes. The user can also remove any of the components while preparing the mixed signal. Always remember default values and not to display something empty for the user waiting for his/her input.
- **Additive noise:** The user can add noise to the loaded signal with custom/controllable SNR level. The program should be able to show the dependency of the noise effect on the signal frequency.
- **Real-time:** Sampling and recovery should be done in real time upon user changes (i.e. no button called “update” or “Refresh” is allowed).
- **Different reconstruction methods:** The Whittaker–Shannon is the most popular reconstruction methods. However, it has its own problems. Also, there are several other methods. Allow the user to explore a couple of these methods to check the difference among them in reconstruction performance. The user should be able to choose the reconstruction method via a combobox. Be ready to explain and discuss these methods during your submission showing the pros and cons each one has. Also, be ready with some signal examples that exploit such pros and cons of each method.
- **Resize:** The application can be resized easily without messing up your UI.
- **Different sampling scenarios:** In your submission, you need to prepare at least 3 testing synthetic signals (generated and saved through your Composer) that address different testing scenarios. One example test case is the one mentioned in the lecture:
 - o A mix (i.e. summation) of 2Hz and 6Hz sinusoidals. If sampled with 12Hz or above, then it should be recovered nicely. But if sampled with 4Hz, then the two frequencies will show up as just one frequency. What if the signal is sampled with 8Hz?

You need to provide 2 more examples; each would exploit a problem or illustrate a tricky feature.

Code practice:

- Same practices from Task 1 (i.e. proper variable and functions names) will continue with task 2.