



MALİK ENES
ŞAFAK

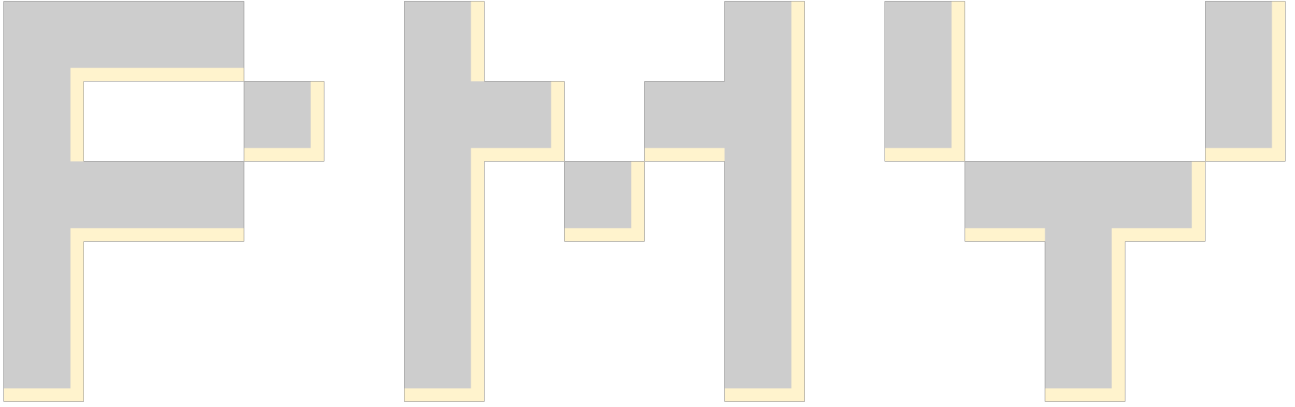
FMY-WEB.GITHUB.IO

Mesleki Matematik

Dijital Sinyal 0.1

İçindekiler

Ayrık Zamanlı Sinyaller (Discrete Time Signals).....	3
Sinyal.....	3
Örnekleme	4
Ayrık Zamanlı Sistemler (Discrete Time Systems)	12



MALİK ENEŞ
ŞAFAK

PMY-WEB.GITHUB.IO

Ayrık Zamanlı Sinyaller (Discrete Time Signals)

Sinyal

Sinyal, en basit haliyle, bir bilgiyi taşıyan herhangi bir şeydir. Sinyaller bir sistemin o anki durumunu veya davranış şeklini anlatabilir. Örneğin, konuşma sinyali zamanın bir fonksiyonu olarak, resim sinyali ise iki boyutlu bir parlaklık fonksiyonu olarak ifade edilebilir.

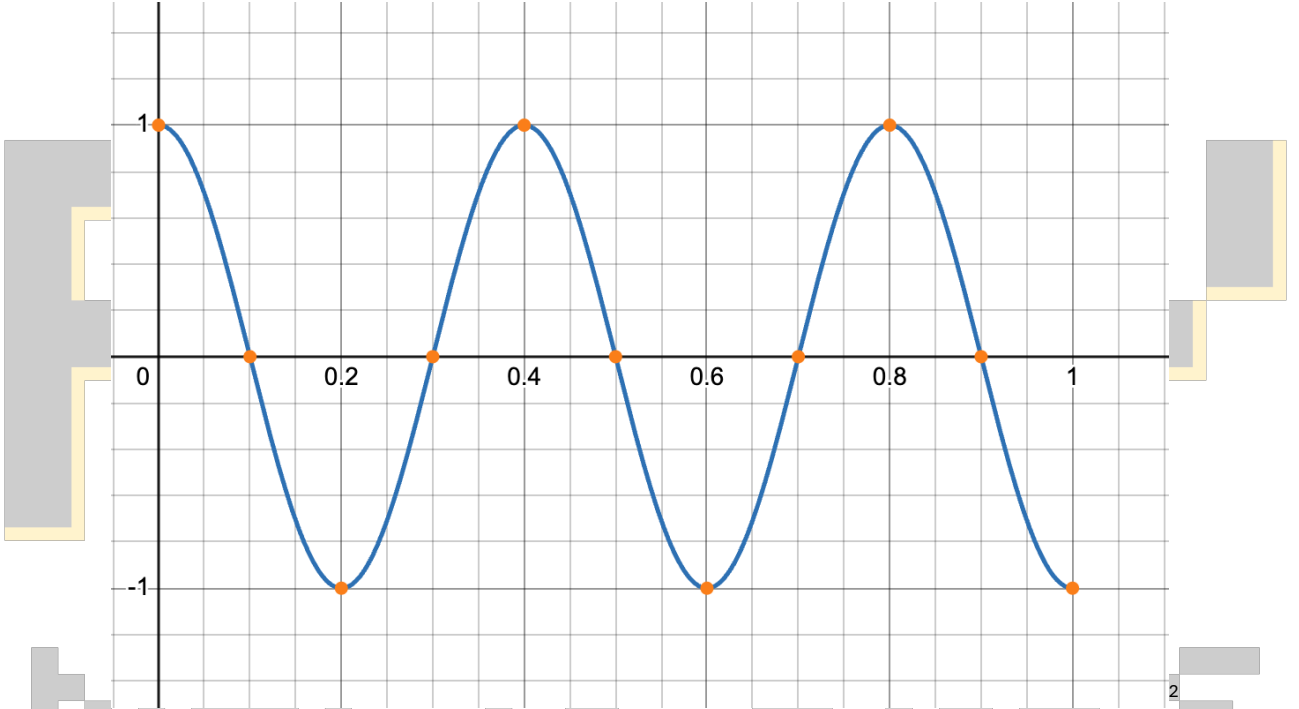


Normalde sürekli zamanlı (continuous time, yaygın kullandığımız şekliyle analog) olan bu fonksiyonların dijital olarak iletilebilmesi için örneklenmesi gereklidir.

Örnekleme

Örnekleme, sürekli zamanlı olan bir fonksiyonun ayrık zamanlı bir diziye dönüştürülmesi işlemi olarak tanımlanabilir. Dolayısıyla ayrık zamanlı sinyaller matematiksel olarak sayılar dizisi ile temsil edilir.

$$y[n] = \sin(2\pi nT)^1$$



Yukarıdaki ifadede **n** sayısı dizi içerisinde kaçınıcı eleman olduğunu belirtirken **T** sayısı ise örnekleme periyodunu belirler. **Örnekleme periyodu** sürekli zamanlı sinyal içerisinde kaç saniyede bir örnek alınacağını gösterir. Örnekleme periyodu

$$f_s = \frac{1}{T}$$

formülü ile örnekleme frekansına çevrilebilir. **Örnekleme frekansı ya da örnekleme oranı** ise 1 saniyede kaç adet örnek alınacağını gösterir. Dolayısıyla yukarıdaki formül

$$y[n] = \sin \left(2\pi f \frac{n}{f_s} \right)$$

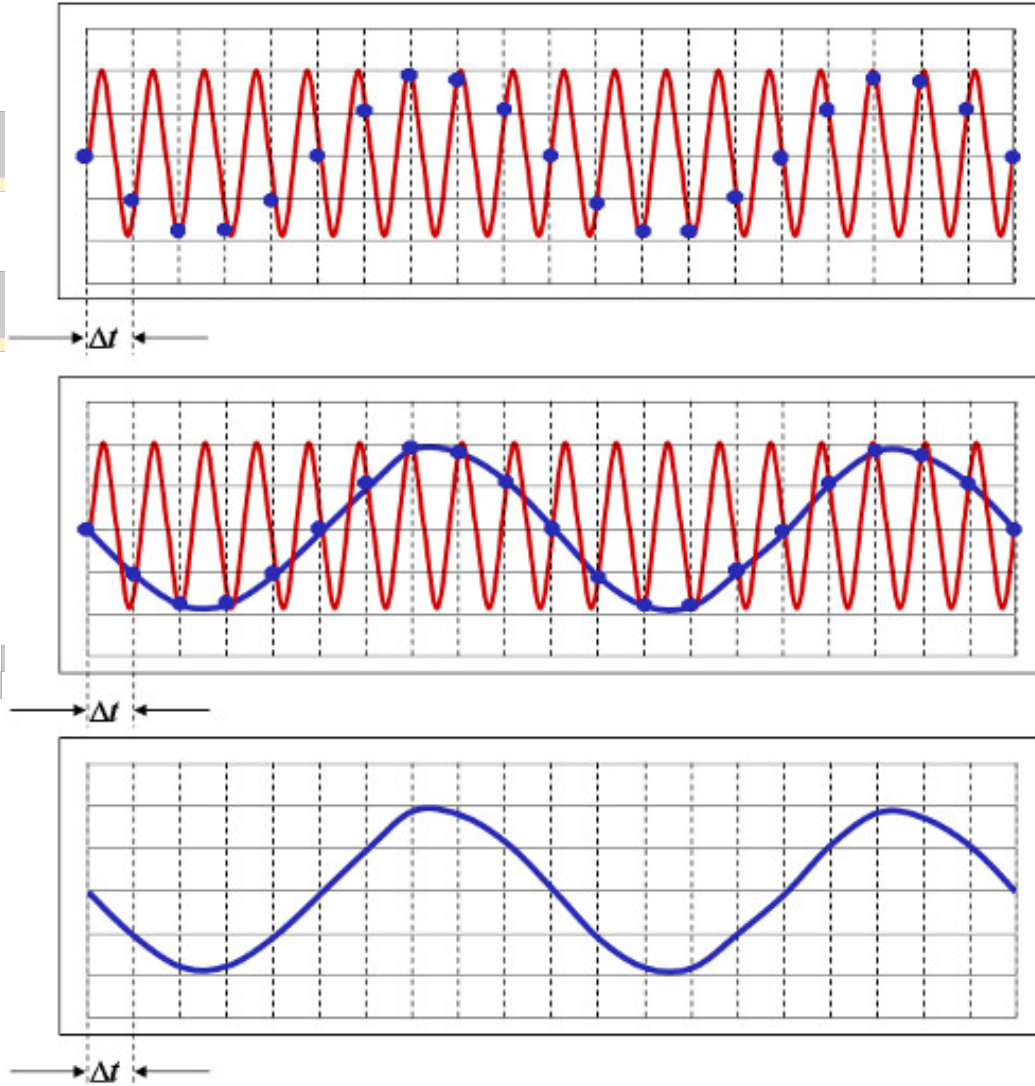
olarak da yazılabilir.

¹ <https://www.desmos.com/calculator/jjrpgz4eeo>

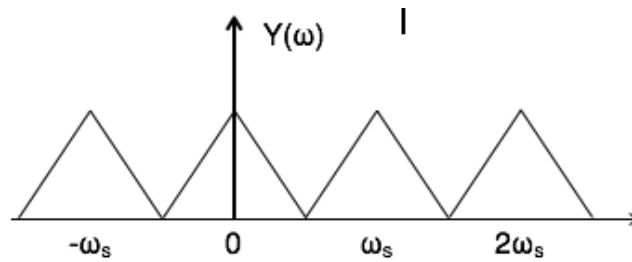
² Buradaki görsele bakarak y dizisinin $[1, 0, -1, 0, 1, 0, -1, 0, 1, 0, -1]$ olduğunu söyleyebiliriz (diziler genellikle köşeli parantez içerisinde gösterilir). Her bir örneğin arasındaki mesafeye bakarak ise örnekleme periyodunun 0.1 saniye, dolayısıyla 10 Hz olduğu sonucuna varabiliriz. Peki saniyede 10 adet örnek alıyorsak görselde neden 11 adet örnek var?

Bir sinyalin herhangi bir bozulmaya uğramadan örneklenebilmesi için içerisinde bulunan en yüksek frekansın en az iki katı kadar örnek alınması gerekir³. Bu sebeple örnekleme frekansının yarısı olan frekansa (yani örneklenebilecek maksimum frekansa) **Nyquist frekansı** adı da verilmektedir.

Eğer Nyquist frekansından daha yüksek frekanslı bir sinyal örneklenirse bu durumda **katlanma (aliasing)** ortaya çıkar.



FM4



E.10

³ Bu teorem Nyquist-Shannon teoremi olarak geçer. Nyquist ve Shannon bu fenomeni birbirlerinden haberdar olmadan keşfetmişlerdir. Shannon'un teorem hakkında açıklaması için <https://www.essrl.wustl.edu/~jao/itrg/shannon.pdf> (19. Bölüm, Band Limited Ensembles of Functions)

Yukarıdaki grafikte x eksenine sisteme verilen frekansı, y eksenine ise sistemin örnekleyeceği frekansı göstermektedir. ω_s ise örnekleme frekansıdır⁴.

Sisteme giren f frekansına karşılık sistemin örnekleyeceği frekans

$$y(f) = \begin{cases} \text{mod}(f, f_s), & \text{mod}(f, f_s) < \frac{f_s}{2} \\ \frac{f_s}{2} - \text{mod}(f, \frac{f_s}{2}), & \text{mod}(f, f_s) \geq \frac{f_s}{2} \end{cases}$$

formülü ile hesaplanabilir⁵.

Matematiksel olarak aliasing'i şöyle kanıtlayabiliriz, örneğin örnekleme frekansının 1.25 katında bir sinüs dalgasını örneklemek istediğinizi düşünün. Bu frekansı

$$1.25f_s = 1f_s + 0.25f_s$$

olarak yazabiliriz. Bu durumda örnekleyeceğimiz sinyali

$$y[n] = \sin\left(2\pi * 0.25f * \frac{n}{f_s} + 2\pi * f_s * \frac{n}{f_s}\right)$$

olarak yazabiliriz. Sinüs fonksiyonu içerisinde toplamının sağındaki kısım, n her zaman tam sayı olması gerektiği için, birim çember etrafında 2π dönmeye sebep olacaktır. Eğer birim çember etrafında 2π kadar dönerseniz, birim çemberin çevresi kadar döneceğiniz için, aynı yere geri dönmüş olursunuz. Dolayısıyla toplamının sağındaki kısım örneklenecek sinyal üzerinde etkisini kaybedecek, sadece toplamının solundaki kısım örneklenmiş sinyale dahil olacaktır.

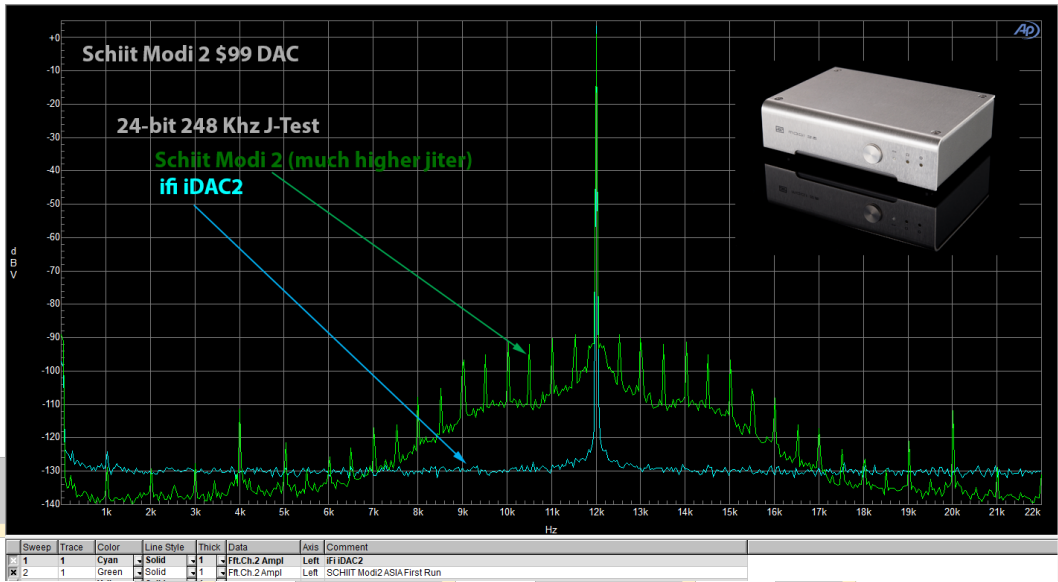
Örnekleme esnasında alınan örneğin tam örnekleme periyodunda alınması önemlidir. Yani 0.1 saniye örnekleme periyoduna sahip bir sistemde 0.1 yerine 0.11'de örnek alırsanız elde edeceğiniz sonuç, siz daha sonradan bu örneği 0.1 saniyede alınmış gibi değerlendireceğiniz için, bozulmuş olur⁶. Bu probleme **faz bozulması, zaman sapması ya da jitter** adı verilir.

PMY-WEB.GITHUB.IO

⁴ <https://www.desmos.com/calculator/onk7uwcfb8>

⁵ Bunun yerine yukarıdaki grafiği aklınıza getirerek hesaplamak çok daha kolay olacaktır.

⁶ Burada aklınıza "Öyle değerlendirmesek olmaz mı?" sorusu gelebilir. Maalesef olmaz. Bunun birkaç farklı sebebi vardır. İlk olarak örneklemenin sabit bir periyotta yapılıyor olması örneğin alındığı zamanı kaydetme ihtiyacını ortadan kaldırır. İkinci olarak sistem örnek almak için aynı miktarda gecikiyorsa bu zaten çok büyük bir sorun değildir (ses olarak düşünürsek geri oynattığınızda sesi orijinalinden biraz daha tiz ya da pes duyarsınız). Fakat böyle bir gecikme genellikle farklı miktarlarda gerçekleşir (çoğunlukla örnekleme donanımından kaynaklanan bir sorundur, yani ses kartınızdan). Bu da örneklediğiniz sinyalin içerisinde var olmayan frekansların oluşmasına sebep olur. Son olarak insan algısı, özellikle çok düşük frekanslı hareketlerde (yani görüntüde ya da ışıktaki) periyodikliği bozulan şeylere alışkın değildir, dolayısıyla bu bozulmadan rahatsız olur. Sesteki etkisini duymak için <https://www.youtube.com/watch?v=EPH0W0NDbGc>, görüntüdeki etkisini görmek için <https://www.youtube.com/watch?v=5SSU-s0AUH0>



Jitter, ses kartlarında, çoğunlukla J-Test adı verilen yöntem ile ölçülür. Bu yöntem örnekleme frekansının dörtte birinde ya da Nyquist frekansının yarısında bir kare dalganın ses kartının girişine verilerek frekans ekseninde analiz edilmesiyle gerçekleştirilir. Kare dalga Nyquist frekansının yarısında olduğu ve kare dalganın bir sonraki harmoniği Nyquist frekansının 1.5 katında olacağı için frekans spektrumunun tek bir sinüs dalgası göstermesi gerekir⁷ (yukarıdaki resimde mavi çizgi buna bir örnektir). Eğer başka frekanslar da ortaya çıkıyor ise kullanılan donanımda jitter problemi var demektir.

Bir sinyali örneklerken, belirli bir örnekleme periyodunda, sinyalin genliği ile ilgileniriz. Bu genlik verisini bir yerden başka bir yere iletebilir, kaydedebilir veya üzerinde işlem yapabiliriz. Analog bir sinyali kaydederken çoğunlukla bir çeşit yuvarlama yapmamız gerekir çünkü sonsuza kadar giden bir sayıyı⁸, yeteri kadar yerimiz olmadığı için, kaydedemeyiz.

Ses sinyalini örneklerken, ses kartımız, genlik değeri olarak voltajı ölçer. Çünkü biz ses sinyalini, standart olarak, voltaj farkı üzerinden iletiriz⁹. Ve yine standart olarak bu voltajın -1 Volt ile +1 Volt aralığında olacağını kabul ederiz. Örneğin bir mikrofonu ses kartınıza bağladığınızda mikrofonunuz

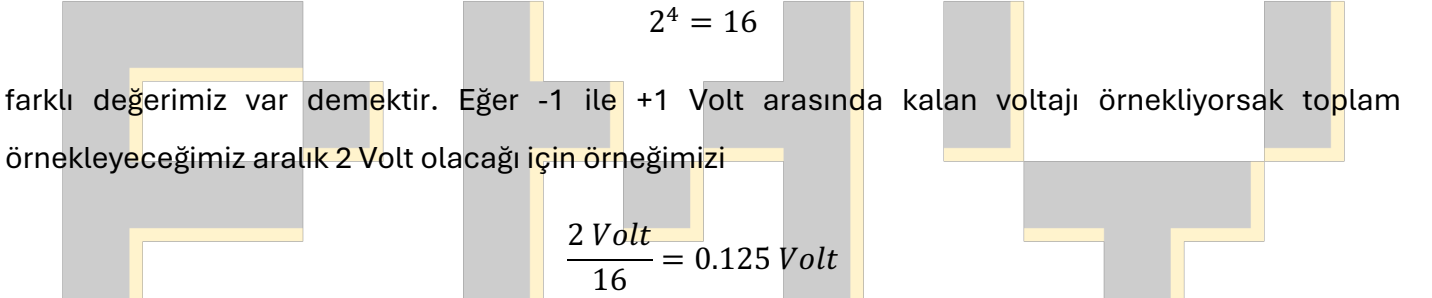
⁷ Çünkü ses kartınız girişine gelen sinyalin, çalıştığı örnekleme oranına göre, Nyquist frekansının üzerinde kalan frekanslarını kesmesi gerekir. Bu durumda kare dalganın bütün harmonikleri filtre tarafından kesileceği için sadece temel harmonik kalır, yani tek bir sinüs dalgası görmeniz gerekir. Eğer ses kartınız bu filtrelemeyi yapmıyor olsaydı sürekli aliasing problemiyle karşılaşmış olurdunuz. Çünkü sadece elektronik ya da dijital olarak üretilen sinyaller değil, mikrofon tarafından alınan sinyaller de 20kHz'in üzerinde frekanslar içerir.

⁸ Burada örnek olarak π sayısını düşünebilirsiniz. π sayısı sonsuza kadar uzar. Bu metni okuduğunuz ekrana baktığınızda bile yuvarlak ya da kenarları yuvarlatılmış bir sürü şey görebilirsiniz. Bunların hepsinin çizilebilmesi için bir şekilde π sayısına ihtiyacımız var. Eğer bunları hesaplamak için π sayısının tamamını bilgisayarımızda (ya da telefonumuzda) tutacak olsaydık, sizce ne kadar alana ihtiyacımız olurdu? Teorik olarak sonsuz, fakat biz π sayısının tamamını henüz bilmiyoruz. 2 Nisan 2025 tarihi itibarıyla “sadece 300,000,000,000,000” basamağını biliyoruz. Her bir basamağın 1 byte yer kapladığını düşünsek yaklaşık 273 terabyte alana ihtiyacımız olurdu. Dolayısıyla bir noktada “bu kadar basamak bize yeter” dememiz gerekir.

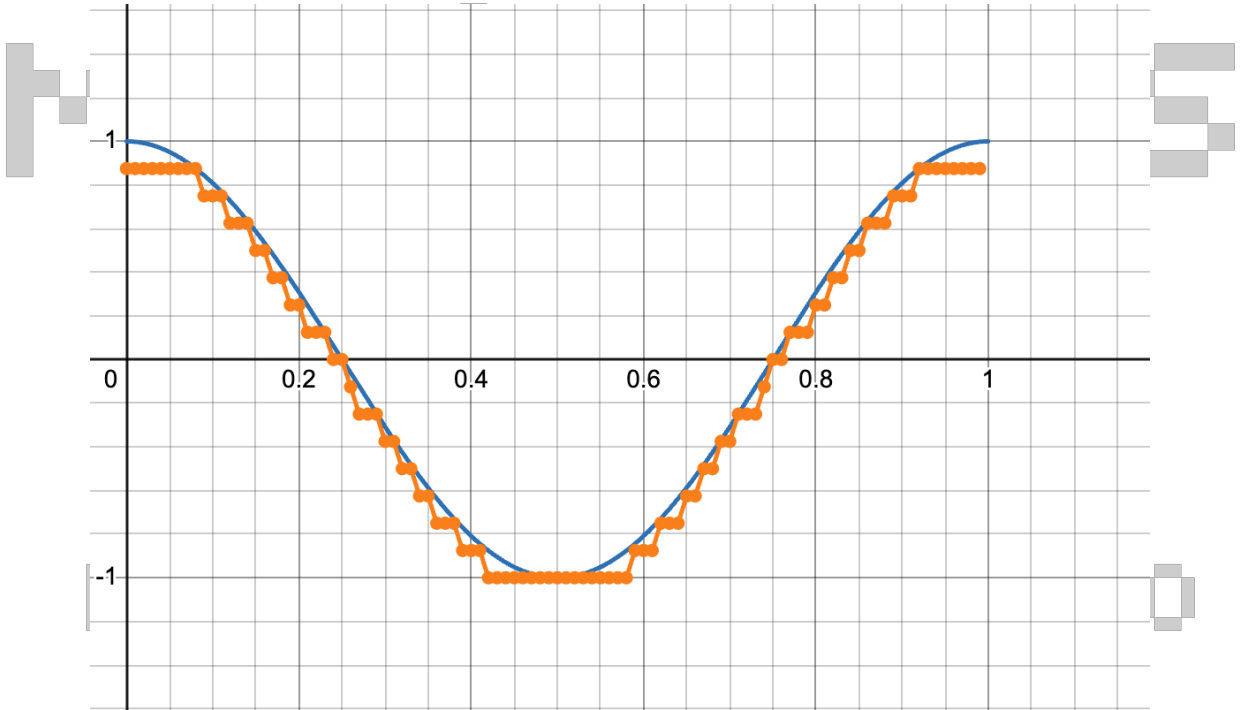
⁹ Başka şekillerde de iletebilirdik fakat kabul ettiğimiz standart bu şekilde. Ayrıca, voltaj farkı ve voltaj aynı anlama gelmektedir. Voltaj her zaman iki farklı değer arasındaki farkı bize söyler. Günümüzde çoğunlukla bu fark toprak (0 Volt) ile sinyal arasındaki fark olduğu için voltaj farkı demek yerine sinyalin voltajı demek de uygun bir kullanım şeklidir.

akustik enerjiye karşılık olarak voltaj farkı üretir. Ses kartınız ise bu voltaj farkını ölçerek sayılara dönüştürür. Bu dönüşüm esnasında ses kartınız voltajı bir şekilde yuvarlamak zorundadır çünkü ne kadar hassas ölçerseniz ölçün voltajın bir noktada bittiğini göremezsiniz¹⁰.

Bilgisayar içerisinde ayrık zamanlı sinyallerin genlik bilgisini saklarken genellikle **bit derinliğinden (bit depth)** bahsederiz. Bit derinliği her bir örneğin kaç bit (ikilik basamak) kullanılarak saklanacağını belirler. Örneğin her bir örnek 4-bit olarak saklanıyorsa analog olarak ölçeceğimiz voltajı temsil edebileceğimiz



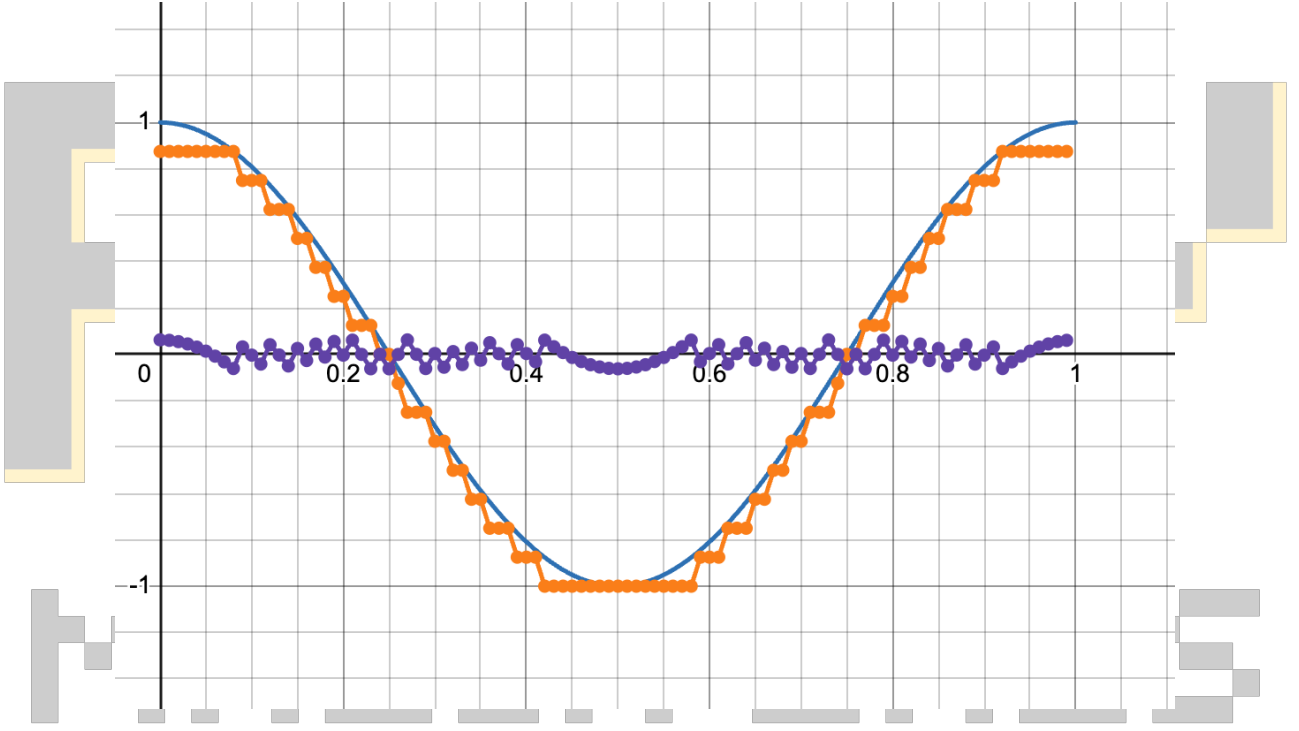
katlarındaki sayılara yuvarlamamız gerekir. Bu yuvarlama işlemine **kuantizasyon (quantization)** ya da **yuvarlama (rounding)** adı verilir¹¹.



¹⁰ Bu demek değildir ki analog sinyaller ya da ekipmanlar sonsuz çözünürlüğe sahiptir. Bir noktadan sonra ölçeceğimiz şey mutlaka gürültü olacaktır. Analog ekipmanların sonsuz çözünürlüğe sahip olduğuyla ilgili genel bir yanlış kanı bulunmaktadır. Bir sinyali, üzerinde değişiklik yapmadan, sonsuz çözünürlükte saklayacak ya da iletecek hiçbir sistem yoktur.

¹¹ <https://www.desmos.com/calculator/peqdyheg8>

Kuantizasyon işleminden sonra ortaya çıkan örnekler örneklenen sinyalin birebir temsilidir. Bit derinliğinin ne kadar az ya da fazla olduğu örneklenmiş sinyalin çözünürlüğüne etkisi yoktur. Yani, 1kHz olan bir sinüs dalgası örnekleniyorsa, bit derinliği ne olursa olsun 1kHz sinüs dalgası elde edilecektir. Bit derinliği sadece gürültüyü etkiler ve kuantizasyon işlemi sonucu ortaya çıkan gürültüye **kuantizasyon gürültüsü (quantization noise)** adı verilir. Bu gürültünün kaynağı orijinal sinyal ile yuvarlanmış versiyonu arasındaki farktır¹².



Bu gürültünün seviyesini

$$FS_{qn} = \frac{1}{2^{d_b}}$$

formülüyle hesaplayabiliriz. FS_{qn} , kuantizasyon gürültüsünün FS değerini, d_b ise bit derinliğini göstermektedir. Örneğin 4-bit ile örneklenmiş bir sinyal içerisindeki gürültünün seviyesinin

$$FS_{qn} = \frac{1}{2^4} = \frac{1}{16} = 0.0625FS \approx -24dBFS$$

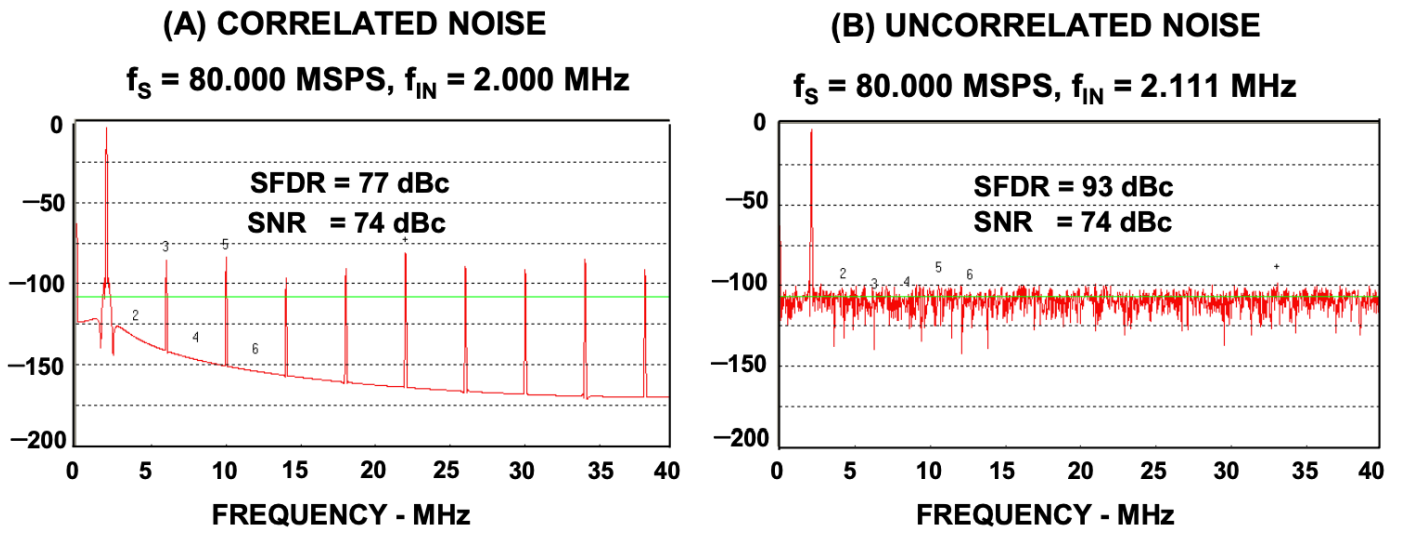
olacağını hesaplayabiliriz¹³.

¹² <https://www.desmos.com/calculator/hif29eloin>

¹³ Bir sinyal içerisindeki kuantizasyon gürültüsünü anlatmak için kullanılan bir diğer kavram SQNR'dır (Signal to Quantization Noise Ratio, Sinyal ile Kuantizasyon Gürültüsünün Oranı). Yukarıdaki örneğe baktığımızda eğer sinyalimiz 1 FS ise 4-bit için SQNR'ın 0.0625 olduğunu söyleyebiliriz ($\frac{1}{0.0625} = 0.0625$). Fakat sinyalimiz 0.0625 FS ise SQNR'ımız 1 olacaktır ($\frac{0.0625}{0.0625} = 1$). Bu durumda diyebiliriz ki, SQNR değerimiz 1 veya daha büyük bir sayı ise sinyalimizin seviyesi kuantizasyon gürültüsünün altında kalacak ve dolayısıyla duyulmayacaktır.

Yukarıda görüleceği üzere bu gürültü örneklenen sinyalin içeriğine çok bağlıdır ve periyodik hareketler oluşturabilir. Bu periyodik hareketler sinyal içerisinde var olmayan frekansların ortaya çıkmasına sebep olabilir. Bu nedenle kuantizasyon esnasında gürültünün belirli bir frekansta toplanmasının önlenmesi ve beyaz gürültü gibi duyulması için **dithering** yapılması gerekir.

Dithering işlemi kuantizasyon gürültüsünün enerjisinin belirli frekanslarda toplanmasını önleyerek bütün frekans eksenine yayar. Gürültünün toplam enerjisi sabit kalsa da bütün frekanslara yayıldığı için çok daha az rahatsız edici duyulur¹⁴. Kısacası, dithering, kuantizasyon işlemi sonucunda oluşan gürültünün örneklenen sinyal ile bir ilişkisinin kalmamasını ve tamamen beyaz gürültü gibi duyulmasını sağlar.



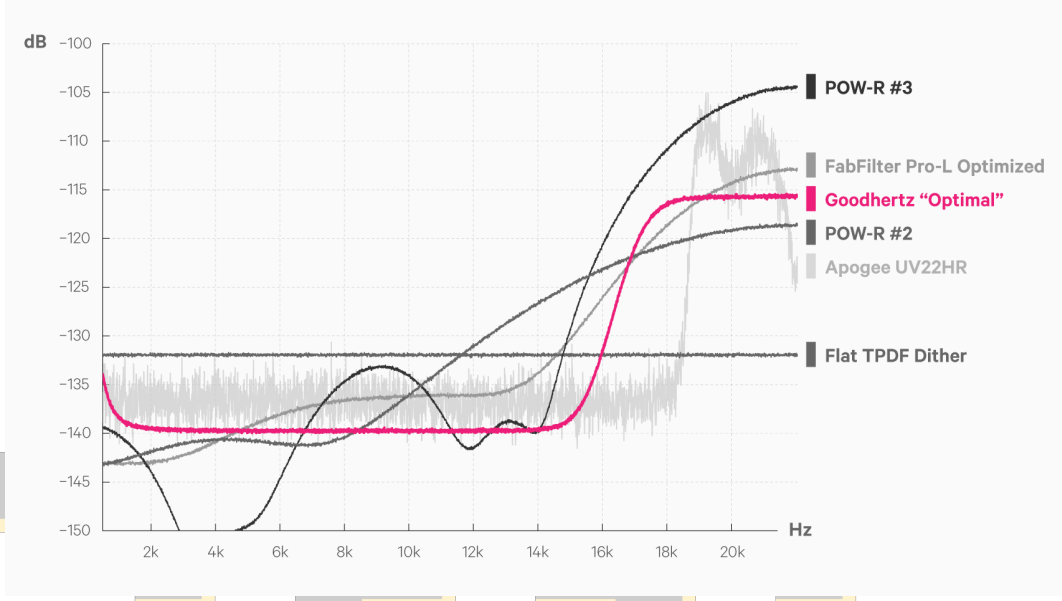
Dithering işlemi sonrasında ortaya çıkan gürültüyü, yine toplam gürültü enerjisini değiştirmeden¹⁵, daha az duyulabilir hale getirme işlemine ise **gürültü şekillendirme (noise shaping)** adı verilir. Bu işlem gürültünün enerjisini insan kulağının daha az hassas olduğu bölgelerde artırıp, daha hassas olduğu bölgelerde azaltmak olarak tarif edilebilir. Gürültüyü eşit gürülük eğrisi¹⁶ kullanarak şekillendirmek gürültü şekillendirme yöntemlerinden biridir.

PMY-WEB.GITHUB.IO

¹⁴ Ve belli bir süre sonra duyulmamaya başlar. İnsan beyni sabit gürültüyü belirli bir süre sonra filtreler. Hakkında düşünürseniz etrafınızda gürültü yayan bir sürü ses kaynağının olduğunu farkedebilirsiniz. Hakkında düşünmezseniz, hiçliğin ortasında düşen bir ağacın çıkarttığı ses gibi, var olmayacaklardır.

¹⁵ Mümkün olsa toplam gürültü enerjisini değiştirmek isterdik tabiki. Ama maalesef mümkün değil. Yapabileceğimiz en iyi şey toplam gürültü enerjisini “arttırmadan” müdahalede bulunmak.

¹⁶ Eşit gürülük eğrisini hatırlamıyorsanız logaritma ders notunun son bölümüne bakabilirsiniz.



Bu sayede algılanan dinamik alanı¹⁷ da gerçek dinamik alandan daha büyükmüş gibi duyurmak mümkündür^{18, 19}

MALİK ENES
ŞAFAK

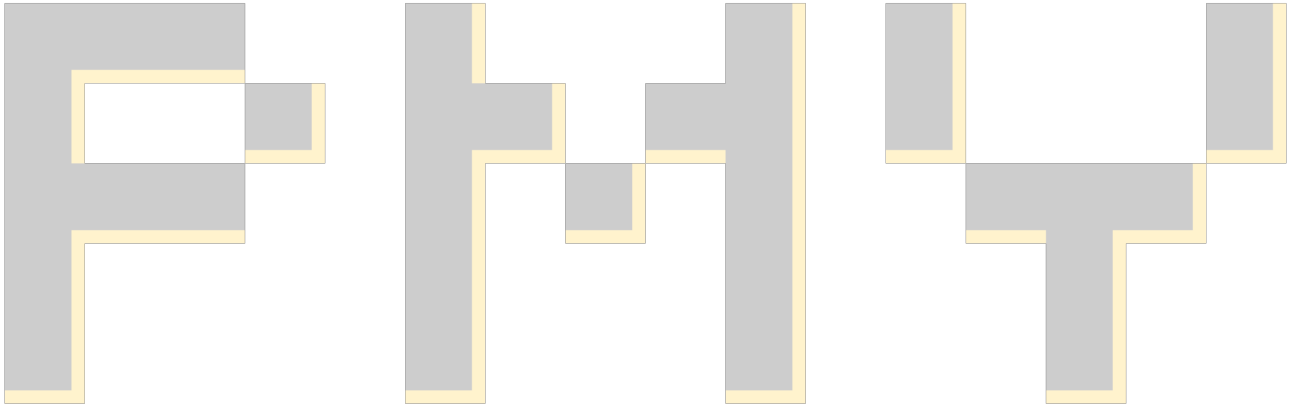
PMY-WEB.GITHUB.IO

¹⁷ Dinamik alan, sinyalin maksimum ulaşabileceği seviye ile sabit gürültünün olduğu seviye arasındaki alanı ifade eder. Bir sinyal dinamik alanın üzerine çıkarsa bozulur, dinamik alanın altında kalırsa gürültünün içerisinde kaybolur.

¹⁸ Psikoakustik olarak.

¹⁹ Bu bölümde anlatılan her şeyi mükemmel bir şekilde görsel olarak anlatan Akash Murthy'nin Digital Audio Fundamentals video serisine ulaşmak için https://www.youtube.com/playlist?list=PLbqhA-NKGP6B6V_AiS-jbvSzdd7nbwwCw

Ayrık Zamanlı Sistemler (Discrete Time Systems)



MALİK ENEŞ
ŞAFAK

PMY-WEB.GITHUB.IO