Vibration analysis with data acquisition in the real-time

Description

In the following example we show how vibration data are obtained in the real-time using three different sensors. Three different sensors are simulated in the document "vibrationsender.mdd", therefore "vibrationsender.mdd" has to be opened and evaluated in order to successfully accomplish this example. The first sensor collects uncorrupted data, while the second sensor is corrupted by additive white Gaussian noise (AWGN) and finally the third sensor data is corrupted with tone interference. The aim of this example is to show how the data is received in real-time using TCP/IP channels and later processed to perform vibration analysis using spectral methods. The spectral methods used in this document are periodogram and spectrogram.

Collecting vibration data

The vibration data is collected using three vibration sensors through three different TCP/IP channels. The data is recorder using three variables s1, s2 and s3. The appropriate function receive() is coded for that purpose.

```
s1:=0     s2:=0     s3:=0     GPs1:=0     GPs2:=0     GPs3:=0

Sp:=matrix create(257 , 1 , 0)              GSP3D:=graph3d(0 , 0)

receive( )
{
   g1:= channel connect("127.0.0.1" , 1805)
   g2:= channel connect("127.0.0.1" , 1806)
   g3:= channel connect("127.0.0.1" , 1807)
   c:= 0
   while(true)
   {
      buffer1:= channel read(g1 , true)
      buffer2:= channel read(g2 , false)
      buffer3:= channel read(g3 , false)
      n:= size(buffer1 )
      t:= 1/n
      c+= n · t
      d1:= col2vec(curve2d(x , c , c+(n-1)·t , n) , 0)
      s1= join mat cols(d1 , buffer1)
      s2= join mat cols(d1 , buffer2)
      s3= join mat cols(d1 , buffer3)
      periodcalc(buffer1 , buffer2 , buffer3 )
   }
}
```

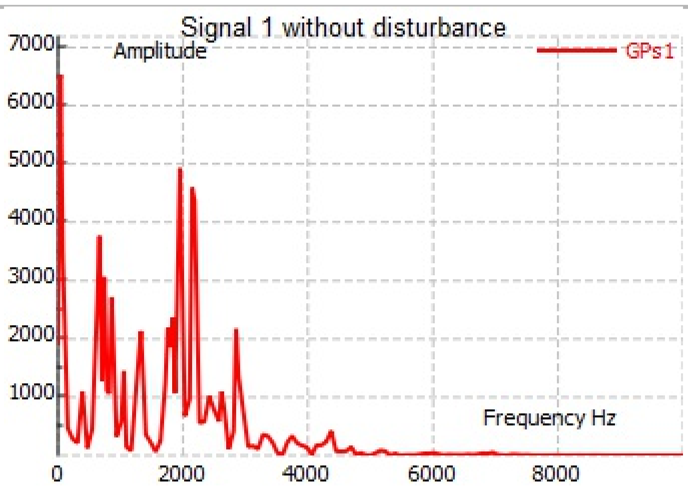
Vibration data analysis

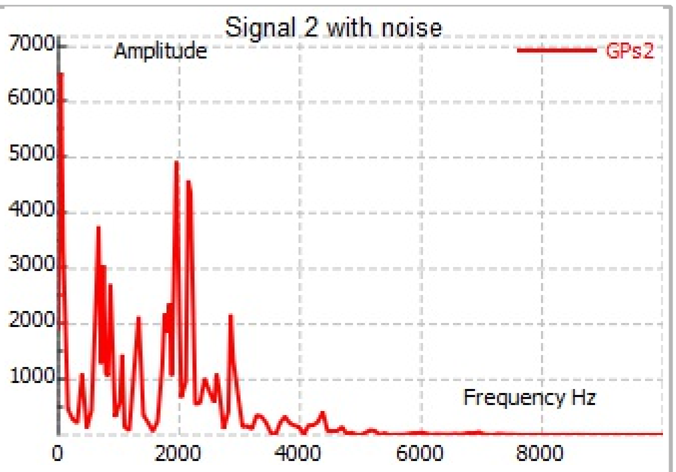
Periodogram, power spectral density and spectrogram are common tools for vibration analysis. In the real time operation, it is very straightforward to use periodogram for each data segment which is received through the channel. In this manner, we can analyze spectral content in real time as it changes. We use rectangular window functions to achieve better frequency resolution. At the same time we can pack all calculated periodograms in a special matrix that represents the spectrogram. The collection process is continuous, and data shown as spectrogram contains the last ten seconds of the collected data. In the next segment we calculate periodograms for all channels and show the spectrogram function.

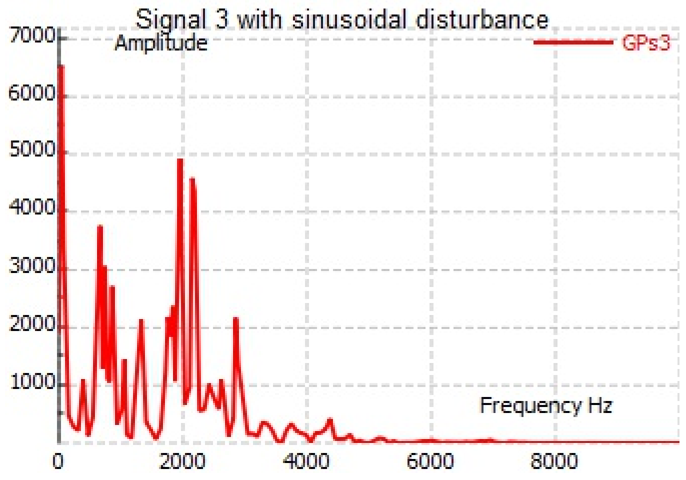
```
periodcalc(vec1 , vec2 , vec3)
{
    Fs := 20000
    Ps1 := periodogram(vec1 , "hanning" , 512)
    Ps2 := periodogram(vec2 , "hanning" , 512)
    Ps3 := periodogram(vec3 , "hanning" , 512)
    ff := ynodes(f , 0 , Fs / 2 , 257)
    GPs1 = join mat cols(ff , Ps1)
    GPs2 = join mat cols(ff , Ps2)
    GPs3 = join mat cols(ff , Ps3)
    Sp = join mat cols(Sp , Ps1)
    if(size(Sp) / 257 > 10)
    {
        r1 := size(Sp) / 257 - 10
        r2 := r1 + 9
        Sp = subset(Sp , 0 , r1 , 256 , r2)
    }
    GSp := data3d(Sp , xx , 0 , 10000 , y , 0 , 10)
    set widget value(GSP3D , GSp )
}
```

Results

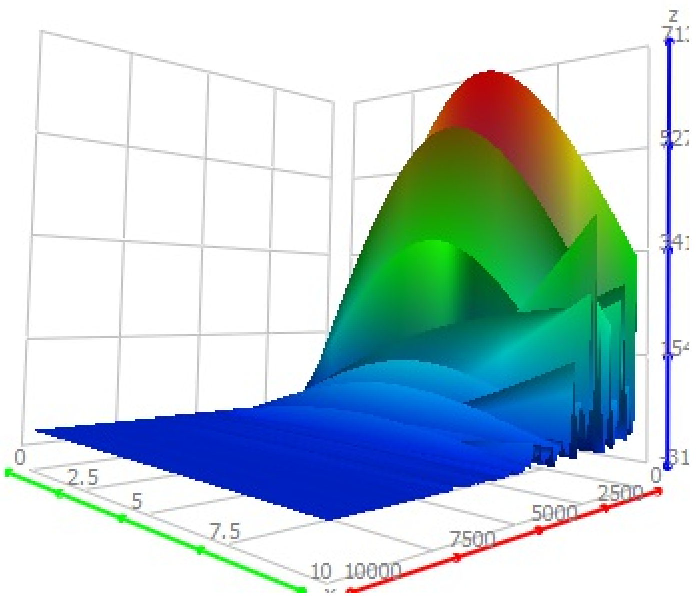
Following graphs show results. The first three graph display instantaneous frequencies in each sensor.







The following graph displays the spectrogram of the last ten seconds in the best channel without noise.



receive( )