Phase noise & digital noise: What is it and Why it is important for groundbreaking RF-applications

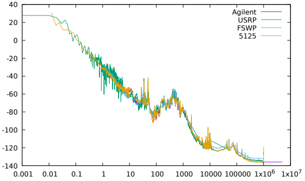
P. -Y. Bourgeois

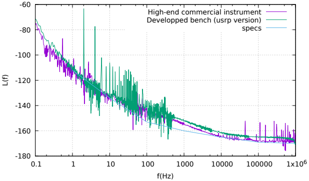
CNRS - FEMTO-ST - T&F Department,
Univ. Bourgogne Franche-Comté

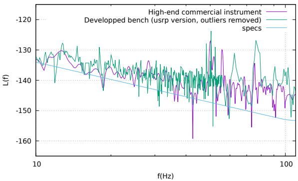
European GNU Radio Days, June 17th, 2019

https://github.com/oscimp

keep in touch with us on IRC: /**join #oscimp on chat.freenode.net:6697**
(and/or #photonics)







Would you ever trust your measurement system ?

Motivations

Sampled-data systems T&F metrology

- High-precision digital methods and instrumentation.
- Mastering realtime RF signal processing
- Fully digital signal analysis, quantization, noise
- Analyzers (counters / DPNMS)
- State-space controls, L/A, DDS, PLL ...
- GNSS / VLBI / Space applications
- Frequency synthesis/transfer / Millisecond Pulsar Timing
- SDR / Telecom / Networks
- Pulse programming, Quantum computing
- Fundamental physics (gravitational red-shift, anisotropy of the speed of light, variation of constants...)

Motivations

High-Precision TF systems under development

- "Compact" single ion-trapped Yb^+ optical clock
- Cavity-based optical oscillators
- Cryogenic Microwave oscillators / atomic oscillators
- State-of-the-Art Fully Digital Phase Noise Analyzer

LabCom FASTLAB: FEMTO-ST/OSU-THETA/Gorgy Timing

- GPS proof[1])
- Certified and Secure Time Dissemination
- Frequency Dissemination

R&T CNES (National French Space Agency) / DGA

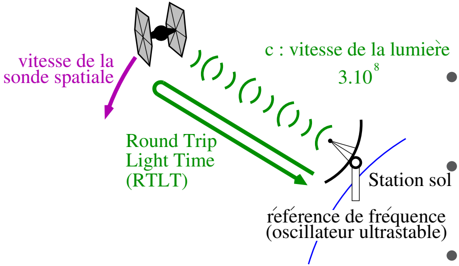
- Digital Oscillator and high-stability Frequency dissemination for space systems

[1] Spoofing GPS is it really the time we think it is, and are we really where we think we are ?, FOSDEM 2019, Goavec-Merou/Friedt/Meyer

But Why Stable Frequencies can be critical ?

Straight answer : to get a valuable mobile phone capture / to be at the right place.

The VLBI example



- Earth-ISS (350 km) : RTLT=2 ms
- Earth-Moon (384 400 km) : RTLT=2.6 s
- Earth-Venus ($43.2 \cdot 10^6$ km : RTLT=144s
- ...

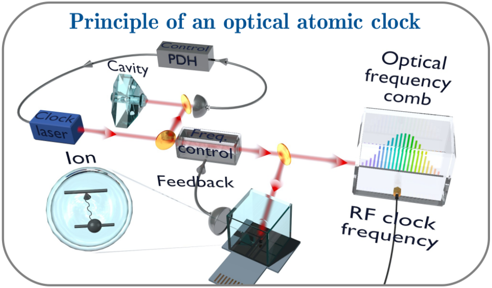
Fractional frequency deviation (i.e. stability) $\frac{\delta f}{f} \sim 10^{-15}$

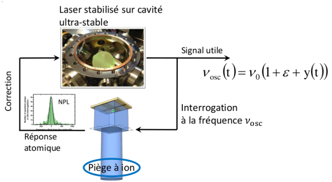
$\rightarrow$ Earth-Moon distance measurement at 20μm (1/4th of a hair)

$\rightarrow$ A 'watch' that gains or loses 1 s / 300 000 centuries

Optical Clock & Oscillators

The Yb^+ clock context

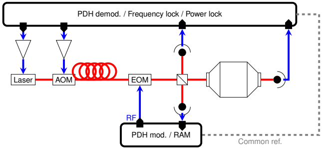


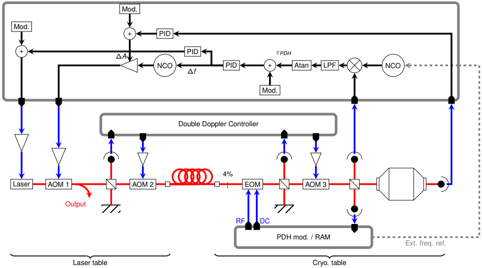


+ Laser femto-seconde pour transférer
le signal dans le domaine RF

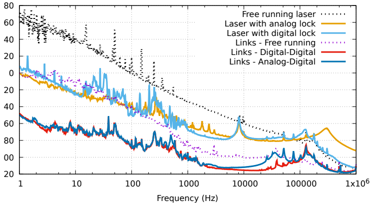
Cavity Stabilized Lasers

Today, nearly every control is already implemented with digital electronics, and we have demonstrated they race favorably against their analog counterparts :)

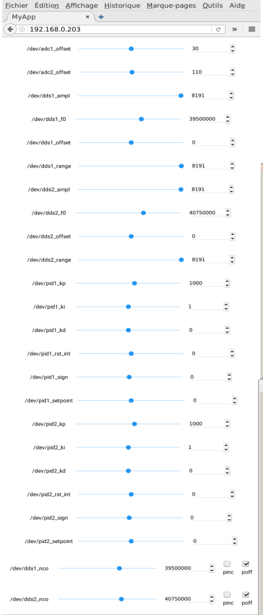




- expected limitation $\sim 2 \cdot 10^{-18}$@1s
- up to date, ~ 95 % of digital stuff in lab's microwave photonics experiments
- uses our ecosystem **https://github.com/oscimp**
- and handy's REMI webserver / ZMQ / GNU Radio backends ;)



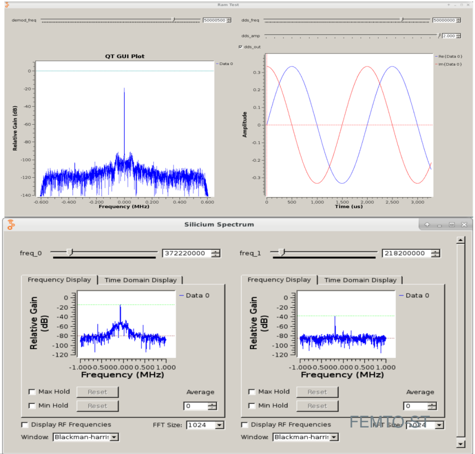
Cavity Stabilized Lasers



expected limitation ~ 2e-18@1s

use our ecosystem https://github.com/oscimp

and handies REMI webserver / ZMQ / GNU Radio backend ;)



Harware & dev. solutions : Ecosytems

- **ARTIQ**[2] (m-labs/NIST) : bitstream generation from python + python on processor and RPC communivation
- **RFNoC**[3] (EttusResearch) : HDL algos within the GNURadio firmware (USRP), routing blocks trough GNURadio $\rightarrow$ SDR
- **Pyrpl**[4] : software abstraction layer on platforms based on D/A FPGA frontends
- **FEMTO-ST DTF ecosystem** [5] : interfaces normalizations, drivers, libraries for software abstraction, python wrapper, QT GUI, toolkit

---

[2] https://m-labs.hk/artiq/slides timing.pdf
[3] https://www.ettus.com/sdr-software/detail/rf-network-on-chip
[4] http://pyrpl.readthedocs.io/en/latest
[5] https://github.com/oscimp/oscimpDigital

FEMTO-ST DTF ecosystem has been released

web-search "**github oscimp**" or "github oscimpdigital"

Keys

- Originally developped by GGM & PYB, part of the platform **Oscillator IMP** with Armadeus boards (then came the Zynq).
- First public release: end 2018
- Supported: Redpitaya (full) and ADALM-PLUTO boards (partial !no-> full !!)
- Hardware compatibility : ZC706 (full), USRP x310 (WIP), B210 (Pluto-like), Altera SoC platform (demo), ZCU (tbd)...
- Tutorials to help newcomers, (training at UBFC) and special events[6]

Oscillator IMP **https://github.com/oscimp/oscimpDigital**

6 https://gnuradio-fr-19.sciencesconf.org/

Digital : pros / cons (brief)

- reconfigurability ☺
- stability (aging), no drift of constants or offsets☺
- error handling, commands, observation ☺
- remote control ☺
- cross-talk / EMC (==)
- quantization ☹(careful design to preserve dynamics)

- learning curve ☹☹
- long time constants ☺
- bandwidth $> 1MHz$ ☹
- exotic algorithms ☺
- MIMO / SDR ☺
- non-linearities compensation ☺
- evolution / portability ☺

OK, what's noise ?

my favorite slide to students

Rule #1 : A "perfect" signal does not exists (or at 0 K :), it is just a mathematical point of view.
Actually that means that transporting information is possible ;)

Rule #2 : Each time you manipulate a signal, it'll likely getting worse In other words, **filtering generally adds noise somewhere.**[7]
$\rightarrow$ Keep the highest SNR through the signal path is the one and only one solution.
A good sound engineer would teach you (repeat that 10 times every morning): **garbage in = garbage out.**

[7] "But why you'll tell us later that PLL helps filtering noise ???"

The noise concept

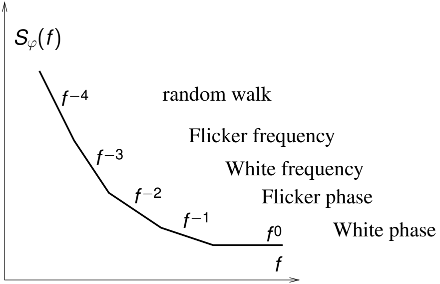
#1 A pure monochromatic wave does not exists so far. As soon as a single atom moves, thermal noise is generated. Multiple atoms add complications, environment as well.

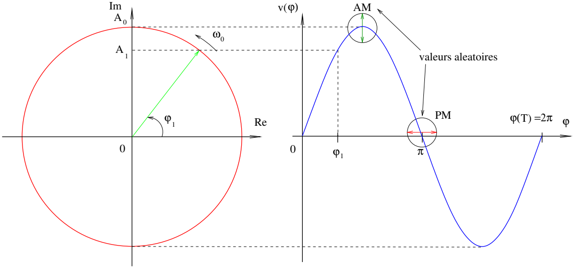
Fundamental limits to the measurement principle

#2 Although 'white noise' is widely known, it's certainly not the only kind of noise. Any noise may fall into 3 main areas :

- Brownian motion
- Johnson noise or Thermal noise (identical to the previous one actually)
- Shot noise (Poisson Law phenomena)
- Flicker noise (1/f noise)

#3 We verify experimentally that noises phenomena are "parametric". They follow and can be represented by a "power law" model ..





$$v(t) = A_0 e^{j\varphi(t)} = A_0 e^{j(\omega_0 t + \varphi)}$$

'Real' signal:

$$v(t) = A(t) \cdot \cos(\omega(t) \cdot t + \varphi(t))$$

$$\begin{aligned} \text{where } A(t) &= A_0 \cdot (1 + \alpha(t)) \text{ - amplitude modulation (AM)} \\ \omega(t) &= \omega_0 + \frac{\mathrm{d}\varphi}{\mathrm{d}t} \text{ - frequency modulation (FM)} \\ \varphi(t) &= \varphi_0 + \Delta\varphi(t) \text{ - phase modulation (PM)} \end{aligned}$$

$$v(t) = A(t) \cdot \cos(\omega(t) \cdot t + \varphi(t))$$

$$\begin{aligned}
\text{where } A(t) &= A_0 \cdot (1 + \alpha(t)) \text{ - amplitude modulation (AM)} \\
\omega(t) &= \omega_0 + \frac{\mathrm{d}\varphi}{\mathrm{d}t} \text{ - frequency modulation (FM)} \\
\varphi(t) &= \varphi_0 + \Delta\varphi(t) \text{ - phase modulation (PM)}
\end{aligned}$$

Having

$$\omega_0 = 2\pi\nu_0$$

$$\alpha(t) \ll 1$$

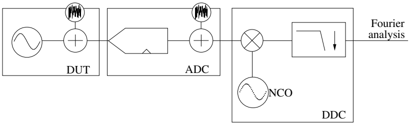
$$\frac{1}{2\pi} \cdot \frac{\mathrm{d}\varphi}{\mathrm{d}t} \cdot \frac{1}{\nu_0} \ll 1$$

The signal is phase/amplitude modulated

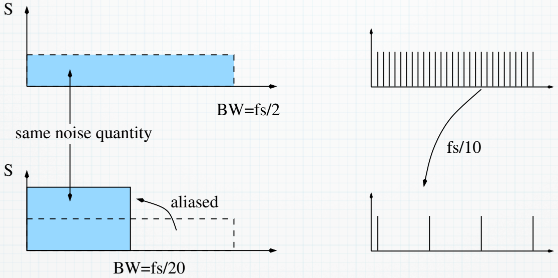
$$v(t) = A_0 \cdot (1 + \alpha(t)) \cdot \cos(\omega_0 t + \varphi_0 + \Delta\varphi(t))$$

The SDR-DDC kernel

- SDR applications : extract information (message) from mixed rf signals
- Metrology : no information to extract but noise
- Anyhow, same technique!



The effect of brutal decimation

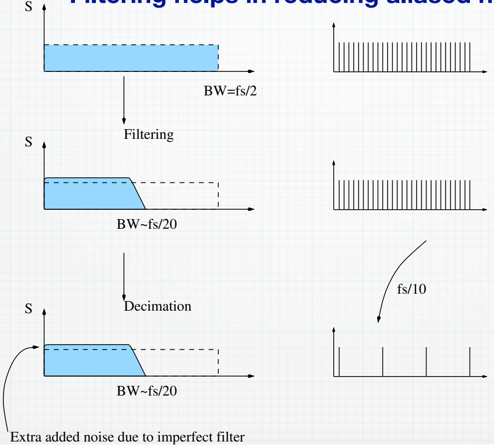


As a thumb rule

A decimation by 10 rises the noise spectrum by 10 dB

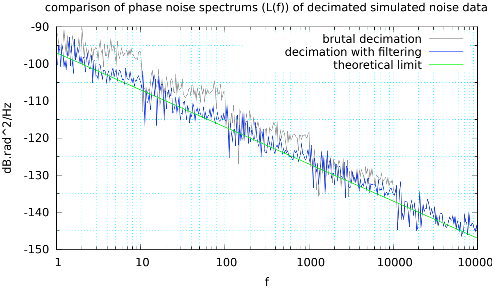
The Bw is reduced but aliased in lower decades

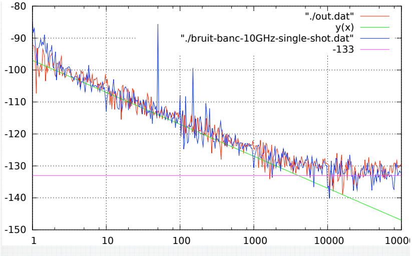
Filtering helps in reducing aliased noise



Successive Filtering/Decimation stages

helps reducing the measurement bandwidth!

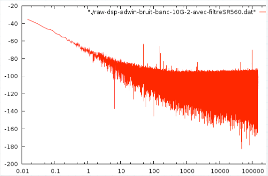


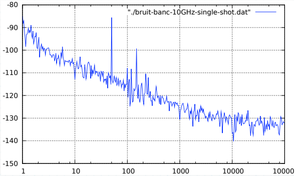


noise simulator part of SigmaTheta software (https://sourcesup.renater.fr/projects/sigmatheta/)

FFT is linear

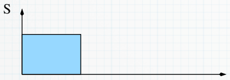
A FFT algorithm gives a linear spectrum, i.e. returns the power from linearly spaced freq. vector

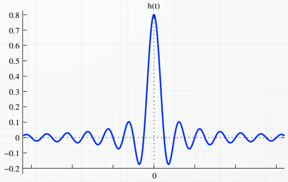




We must artificially redistribute the energy logarithmically to interpolate the spectrum

Perfect averager



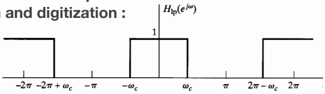


$$TF^{-1}\left[\mathrm{rect}_{2f_c}\left(f/fs\right)\right] = 2f_c\mathrm{sinc}(2\pi f_c t) = h_t$$

simple averaging not suitable (successive aliasings of aliases)

filtering is a time domain convolution process

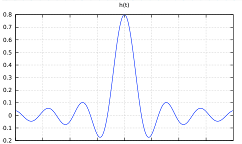
because of finite length and digitization :



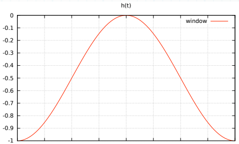
(from Oppenheim)

Side effects of apodisation

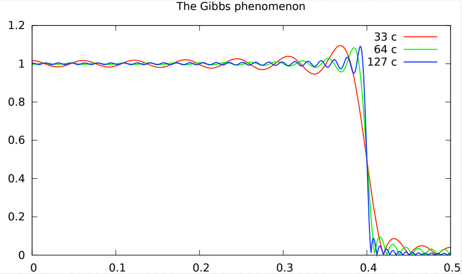
The sinc function must be truncated and windowed to preserve causality



x

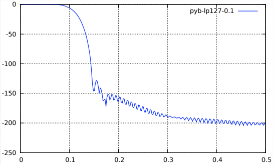


which in turn introduces sides effects



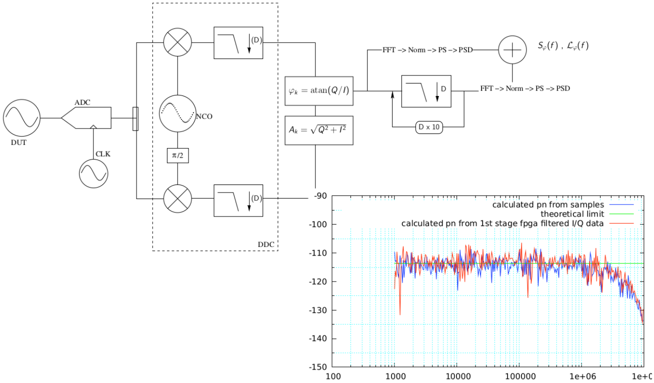
Filtering is like a bottomless pit

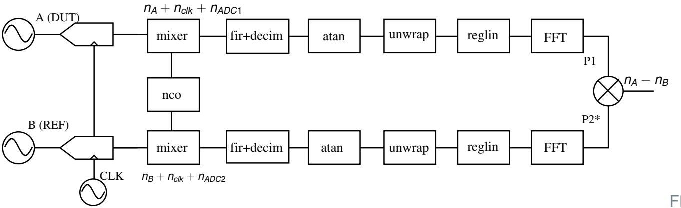
reducing the Gibbs phenomenon in turn lead to side effects
ripples has been lowered (e.g. using remez-like algorithm) but slope gets softened



a few trade-offs :
filter length
sharp cutoff
equiripples minimized
cascaded filters
overlapping
aliasing management
...

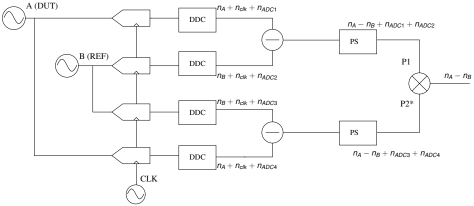
Digital Phase Noise Measurement Principle



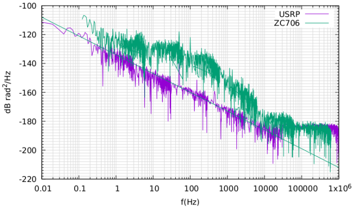


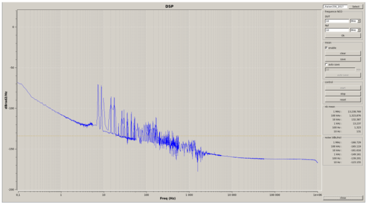
Digital Phase Noise Measurement Principle

State-of-the-art 4-ch Fully Digital Phase Noise Measurement System



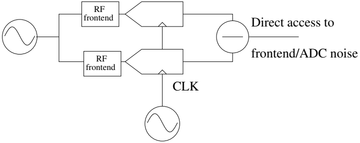
- Uncorrelated terms vanish at a rate of $\dfrac{1}{\sqrt{n_{corr}}}$
- One version uses OscimpDigital from A-Z
- One version with USRP (using their embedded DDC)
- 250 000 lines of code in 4 years / 2 people (again, thanks GGM)

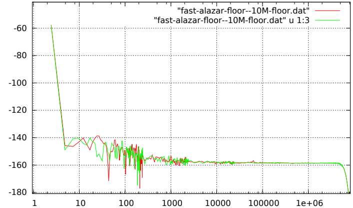




Noise Frontend Charaterization

ADC / frontend noise





$\mathcal{L}(f) = -158.6\ dbV^2/Hz$ is the single channel ADC noise limit (3 dB deduction from 2ch taken into account).

- $q = \frac{v_{fsr}}{2^M} \sim 6 \cdot 10^{-4}$
- $\sigma^2 = \frac{q^2}{12} = 3.1 \cdot 10^{-8} V^2$
- total noise= $\mathcal{N}_t = \frac{\sigma^2}{FPBW}$ if $FPBW > f_N = fs/2$ else $\mathcal{N}_t = \frac{\sigma^2}{f_N}$. Here, $FPBW = 150 \cdot 10^6$ so $\mathcal{N}_t = 2 \cdot 10^{-16} V^2/Hz$.
- $S_{floor} = -156.8 dbV^2/Hz$
- Measurement indicates -158.6+3=-155.6.
- From this we deduce $ENOB = \log_2\left(1 + \frac{v_{fsr}}{\sqrt{12 \cdot f_N \cdot 10^{\frac{S_{floor}}{10}}}}\right) =$ 11.93bits matching the datasheet $ENOB$...

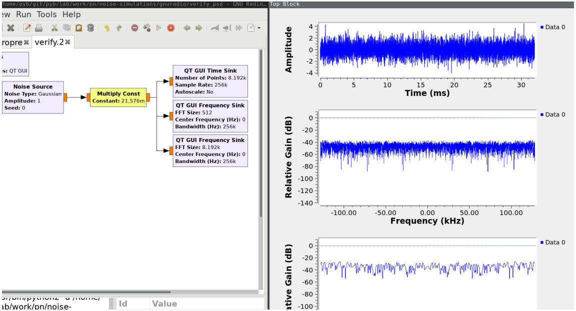
GNU Radio as a measurement analyzer ?

At first sight GNU Radio is not meant to be a measurement system, but a data manipulation / visualization system:
It works with arbitraty units or relative dB whatever :)
No need to know v_{fsr}, carrier power, etc... which also depends on what's going on in front of GNU Radio.
Let's take an example:

- consider a random noise signal generator using a limit-centered gaussian shape (i.e. use `normrnd` in Octave or `Noise source/ gaussian` in GNU Radio.
- add a bit of physical measurement
- perform your spectral measure

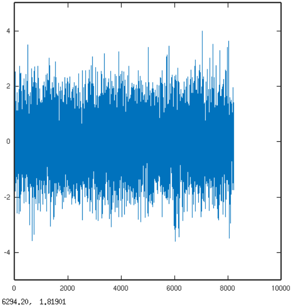
Such a noise should have a mean $\mu = 0$ and a variance $\sigma = 1$.

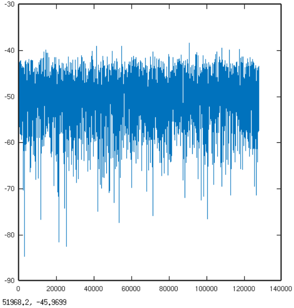
256 kHz samp rate like the venerable HP3562 '(and want to have a look at the DSP



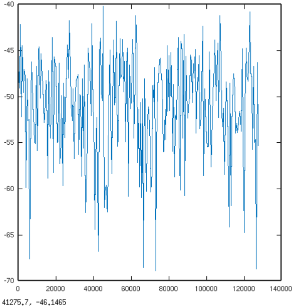
35 45 hum...

The measurement bandwidth remains the same, so the level should be ?
It's likely there was a normalization on N only, **which contradicts the definition of the PSD.**

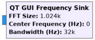


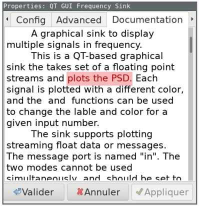


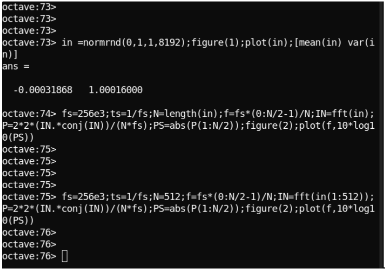
-47

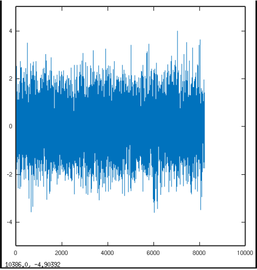


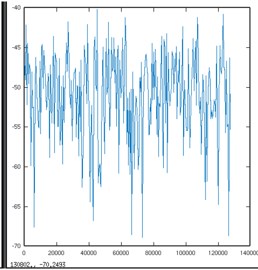
-48











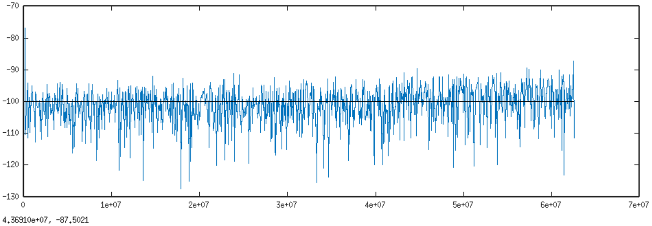
or just use N=8192;in=normrnd(0,1,1,N); fs=256e3; periodogram(in,rectwin(N),N,fs)
in=normrnd(0,1,1,512); periodogram(in,rectwin(length(in)),length(in),fs)

(Of course, one can verify that changing sampling rate has no effect on the displayed level.) Also, usually PSD is mainly plotted 1-sided (at least in T&F metrology).

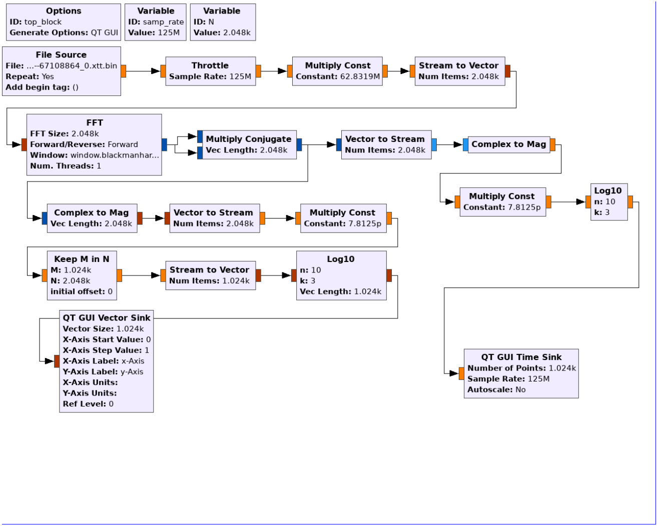
GNU Radio as a measurement analyzer ?

Anyway the show must go on, Queen said ;)

The objective here is to simulate a white phase noise of $-100\,\mathrm{dBrad}^2/\mathrm{Hz}$ of a 10 MHz carrier (0 dBm / 50 Ω) sampled at 125 MHz (alike redpitaya) and measure it.

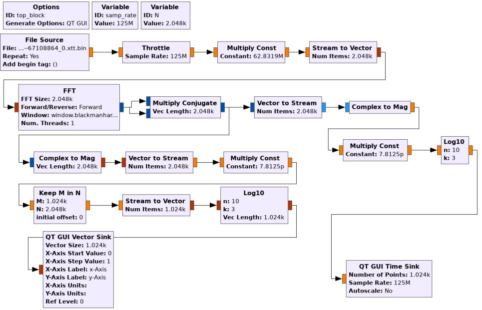


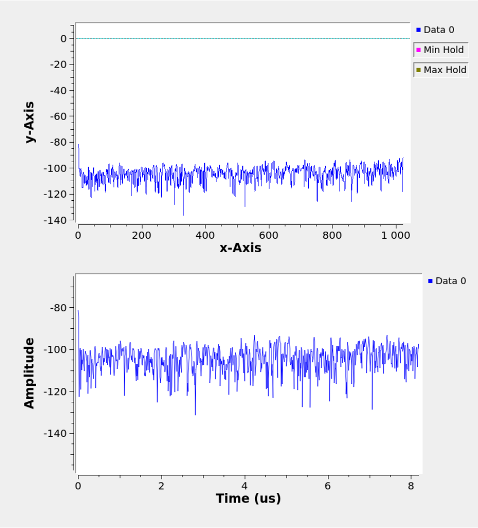
GNU Radio as a measurement analyzer ?



The 1st GNU Radio phase noise analyzer

It works!





-> The 1st GNU Radio phase noise analyzer :)
(from simulated data) (+ ref bruiteur that will be incorporated within oscimp framework)
To embedd in a block (-> a block of blocks / straight C++ programming preferred ? / still need for variables/constants definition block let to the experimentalist. ->
+ DSP noise simulator (A. Hugeat/PYB), online soon ; what if compatible with GNU Radio ?(open question)