

# A decade of gr-specest

Free Spectral Estimation!

September 2019, GNU Radio Conference  
Martin Braun

# What is Estimation (Theory)?

- An unambiguous, mathematically derived algorithm for determining a desired value from a set of noisy inputs
- No guesswork involved! “Guesstimates” are something else, not grounded in scientific rigour.
- Example 1: Determine constant the voltage on a line, but the measurement equipment is injecting white Gaussian noise, with an unknown, but constant and bounded variance

=> [do some math after establishing assumptions]

=> Best solution: Measure N times, average the results:  $\hat{u} = \frac{1}{N} \sum_{i=0}^{N-1} u_i$

# What is Estimation (Theory)?

- An unambiguous, mathematically derived algorithm for determining a desired value from a set of noisy inputs
- No guesswork involved! “Guesstimates” are something else, not grounded in scientific rigour.
- Example 2: How long will it take to complete a project, if there is exactly three tasks, and they can only be completed in order. Task 1 takes 3-6 days, Task 2 takes 2-5 days, Task 3 takes 1-3 days.

=> [Apply some common model]

=> 90% confidence of completion within 12.4 days

# What is Estimation (Theory)?

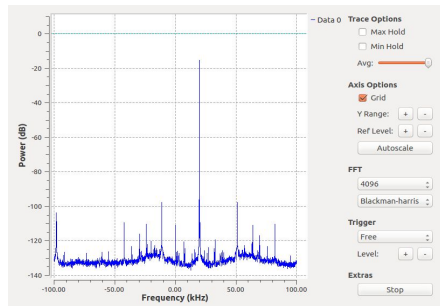
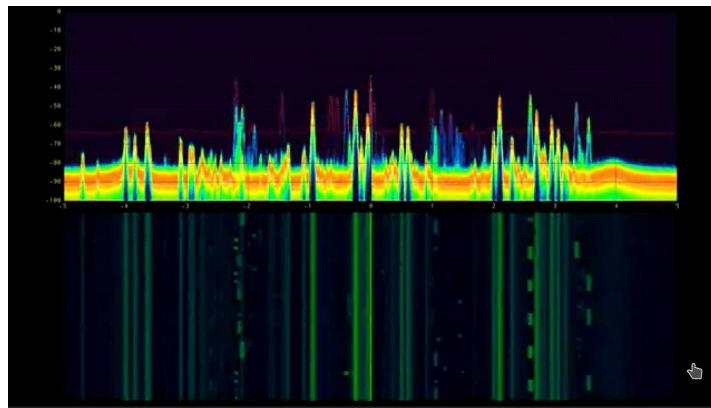
- An unambiguous, mathematically derived algorithm for determining a desired value from a set of noisy inputs
- No guesswork involved! “Guesstimates” are something else, not grounded in scientific rigour.

=> So, spectral estimation is the mathematical derivation of an estimate for “the spectrum” based on noisy measurements.

# What is a spectrum?

- Come to think of it, what is a spectrum? What's the most spectrum-esque picture here?

=> Understanding spectral analysis is useless without first defining “spectrum”



# Non-Parametric Spectral Estimation

# Estimating the Power Spectrum Density

- PSD as expected value of the truncated Fourier transform:

$$S_{xx}(\omega) = \lim_{T \rightarrow \infty} \mathbf{E} \left[ |\hat{x}(\omega)|^2 \right]$$

Where

$$\hat{x}(\omega) = \frac{1}{\sqrt{T}} \int_0^T x(t) e^{-i\omega t} dt$$

- “Average time-limited Fourier transform”

**We know how to do time-limited Fourier transforms of discrete-time signals really fast, don't we!**

- PSD as Fourier transform of the autocorrelation function of the underlying stochastic process:

$$S_{xx}(\omega) = \int_{-\infty}^{\infty} R_{xx}(\tau) e^{-i\omega\tau} d\tau$$

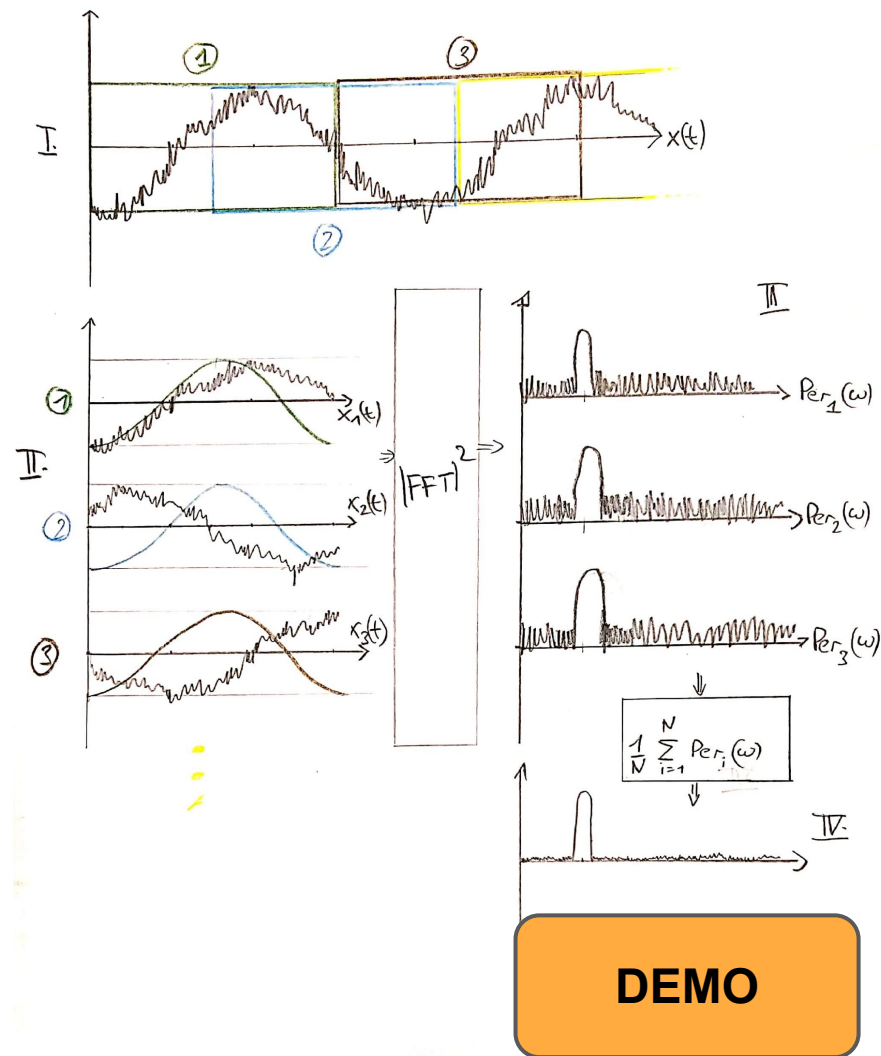
where:

$$R_{xx}(\tau) = \mathbf{E}[X(t)^* X(t + \tau)]$$

**This path leads to correlogram-based methods (not further discussed here)**

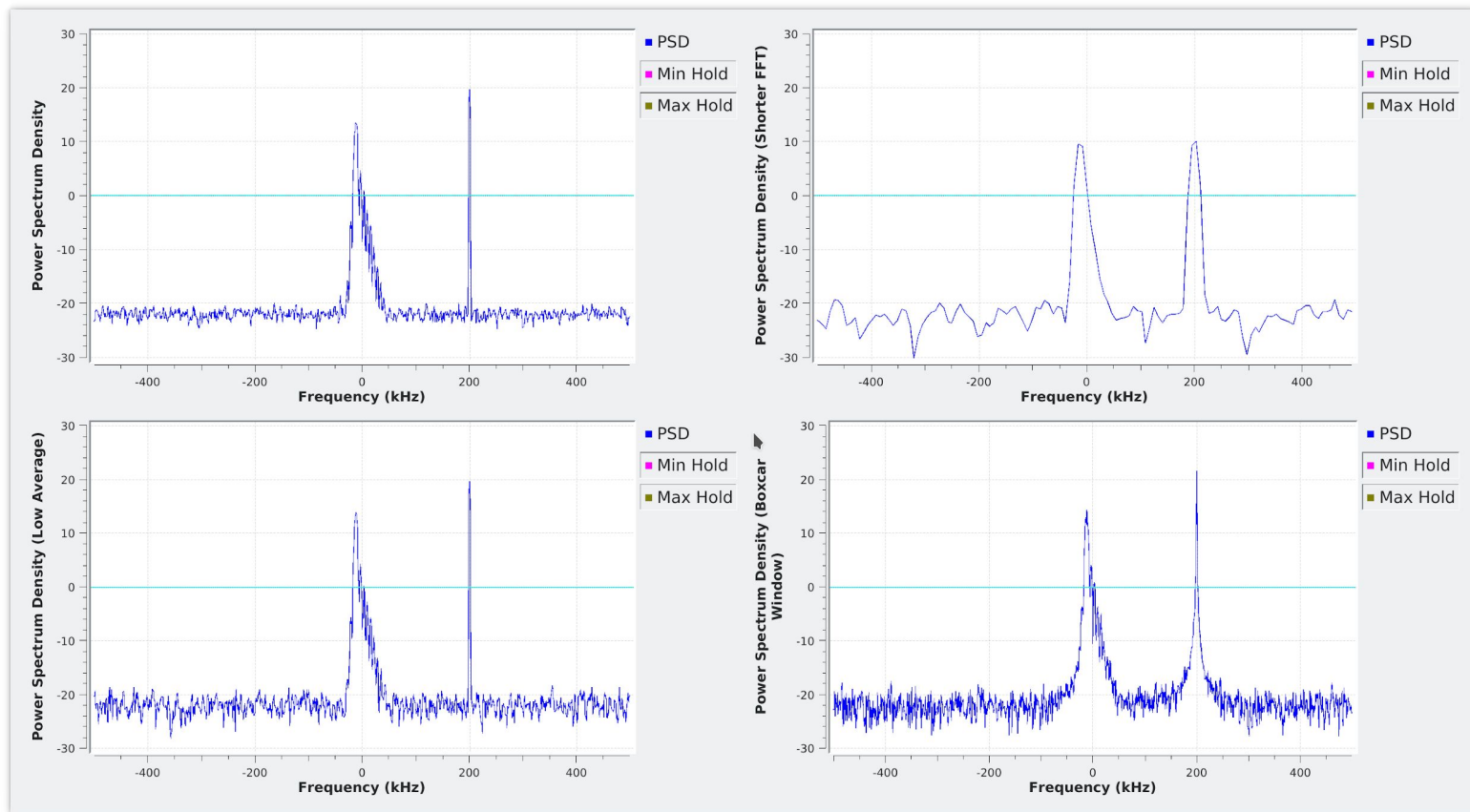
# Welch's Method

- Split signal into  $L$  segments of length  $M$  samples, overlapping by  $D$  samples
- Multiply every segment by a window function ("modify the segment")
- Calculate the FFT of every modified segment
- Calculate the magnitude-squared of every FFT output ("modified periodogram")
- Calculate bin-wise average of periodograms
- For continuous updates, drop last periodogram and acquire new periodogram from new samples



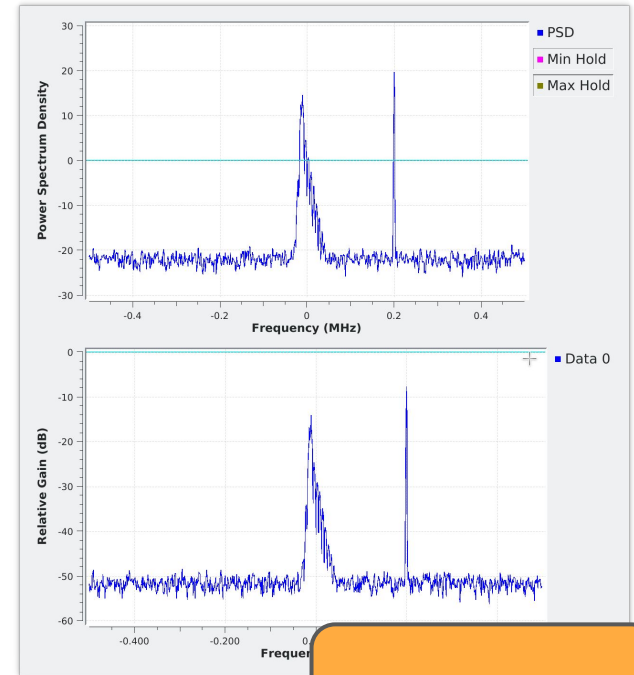


# welch\_spectrum.grc



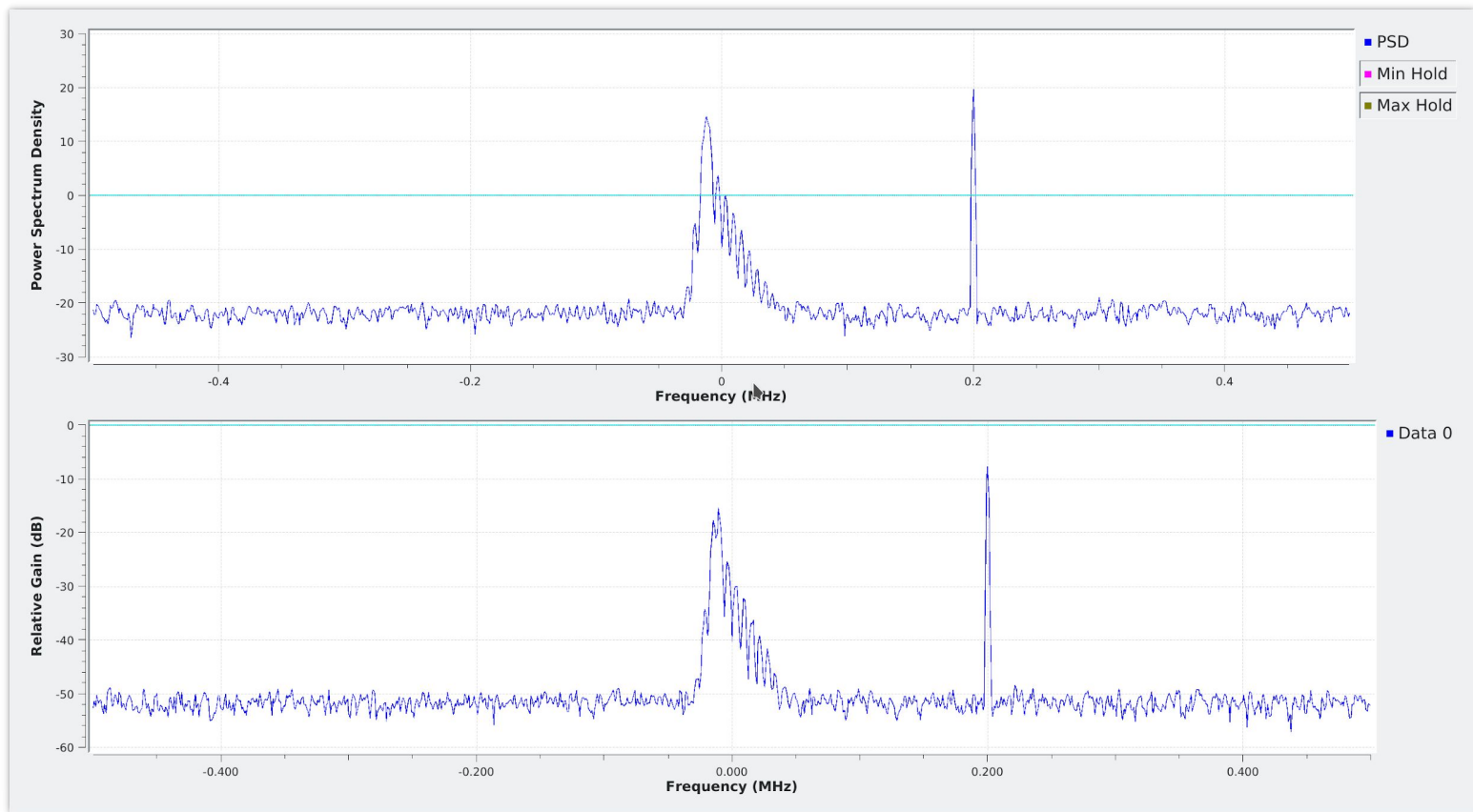
# What is the “QT Frequency Sink”?

- Less sophisticated little brother of the Welch estimator:
- No overlap
- Samples get dropped to accommodate QT Update Interval
- Averaging is done using single-pole IIR
- No power level normalization
- “Dropped some rigour for performance & convenience”



**DEMO**

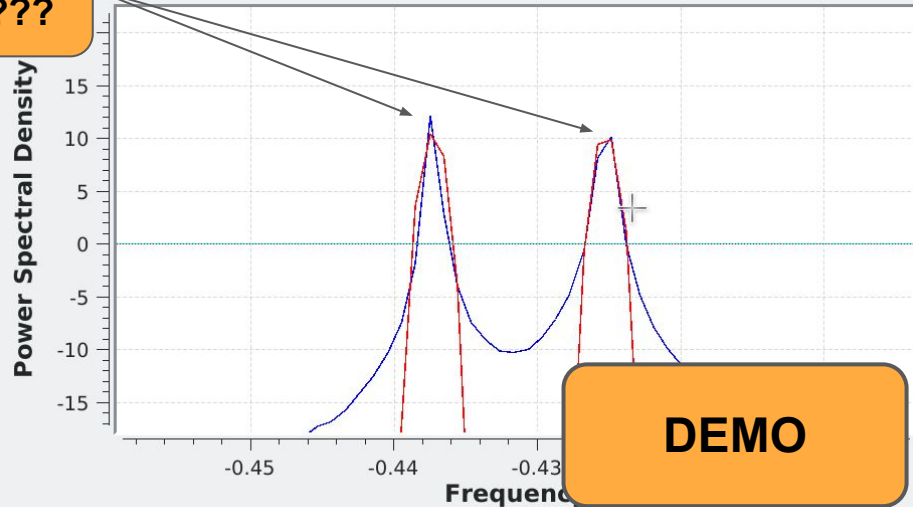
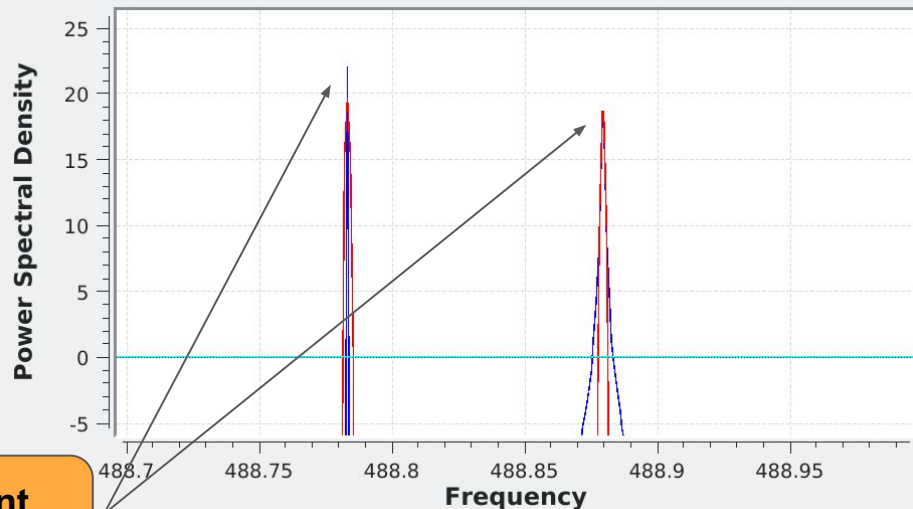
# welch\_qt.grc



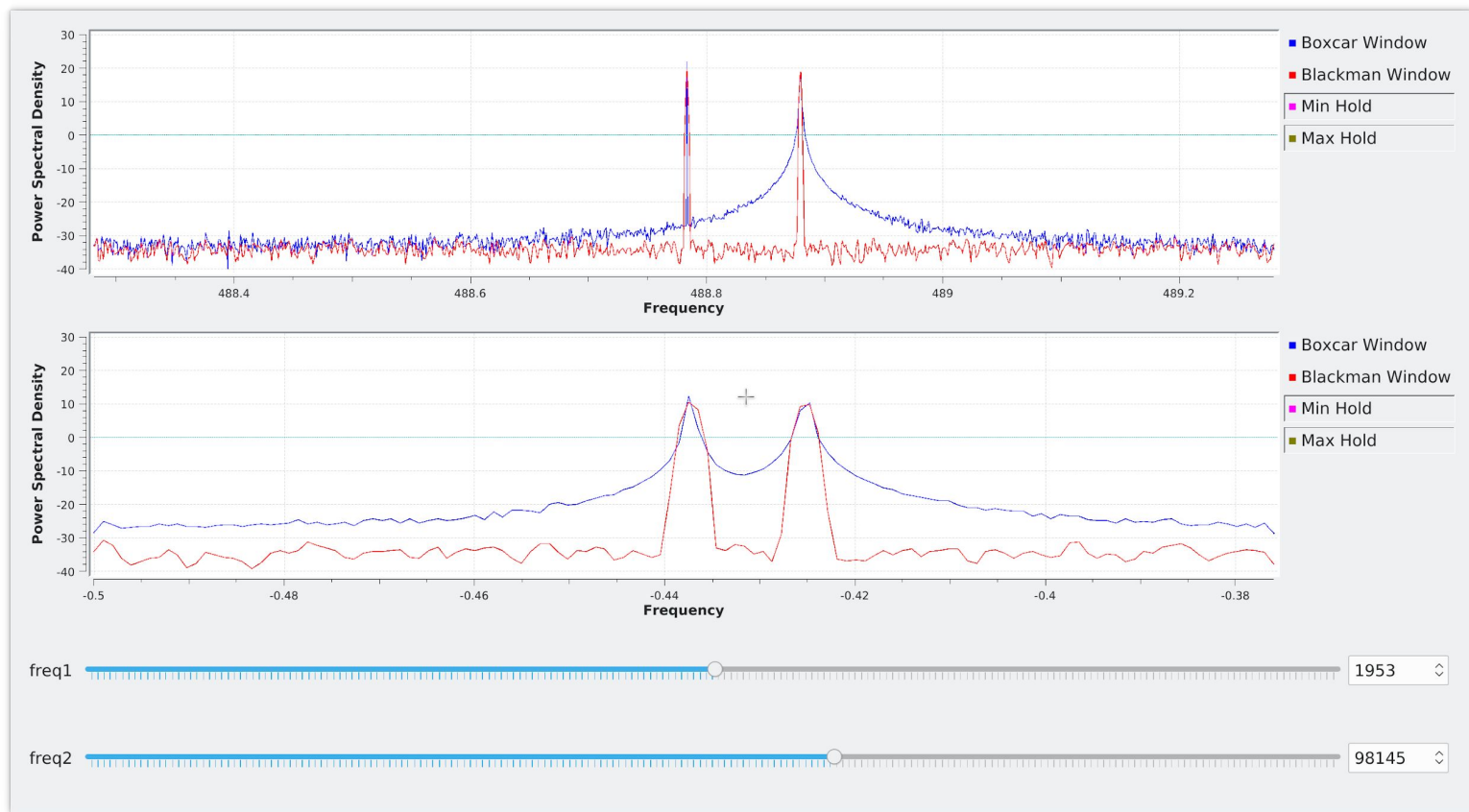
# Practical Considerations

- Window Function: Affects sidelobes and scalloping loss
- FFT length: Affects FFT gain
- => Power peak values are a function of both frequency, and the Welch configuration parameters!

**Four different power levels???**



# welch\_sinusoids.grc



# Parametric Spectral Estimation

# Sinusoidal Detection

- What if I'm only finding sinusoids in noise?

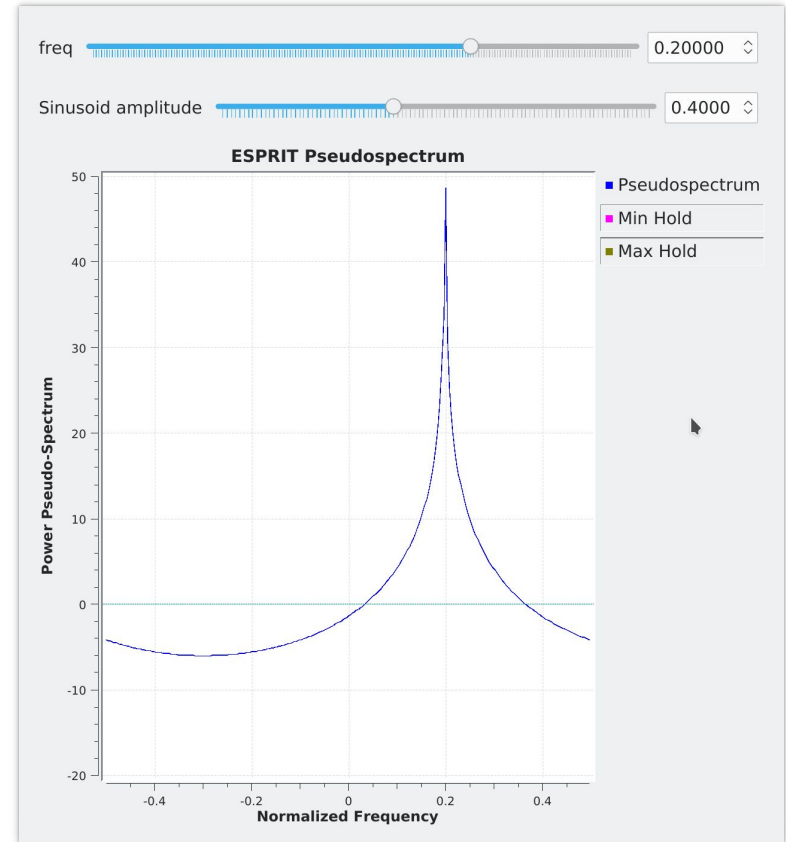
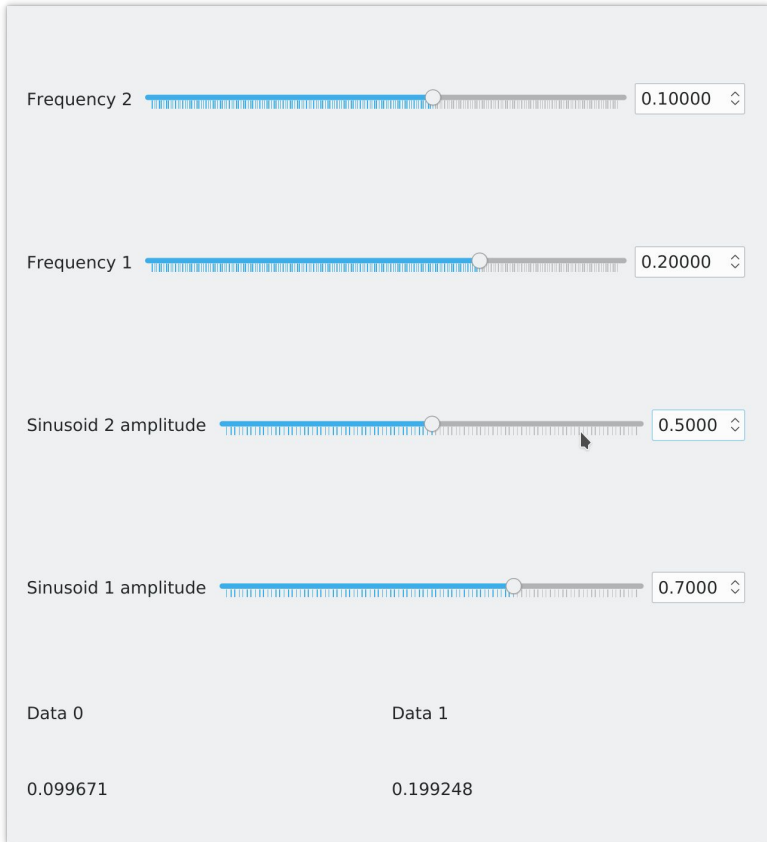
Signal model:

$$y(k) = \sum_{i=0}^n \alpha_i e^{j(\omega_i k + \phi_i)} + z(k)$$

- Then all I'm looking for are parameters of my model
  - “What are the omegas?”
- Solution: Subspace-based, algebraic methods
- MUSIC, ESPRIT are the most commonly cited variations of these methods
- These algorithms can be used as efficient direction-finding algorithms

**DEMO**

# esprit\_freqest.grc and esprit\_pspectrum.grc





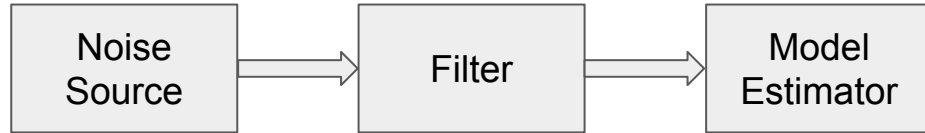
# ESPRIT and MUSIC

- We'll skip the math to avoid incurring the wrath of the MC
- Pseudospectrum vs. Frequency Estimation:
  - We either immediately identify the parameters, and that's it
  - Or we derive a spectral plot out of our method
- The model needs to be well defined: We need to know the number of sinusoids for a good estimate
- The input for MUSIC and ESPRIT is an estimate of the covariance matrix, which improves with more averaging
- The dimensionality of the covariance matrix is an algorithm parameter, it needs to be greater than the number of sinusoids

**DEMO**

# AR/MA Signal Models

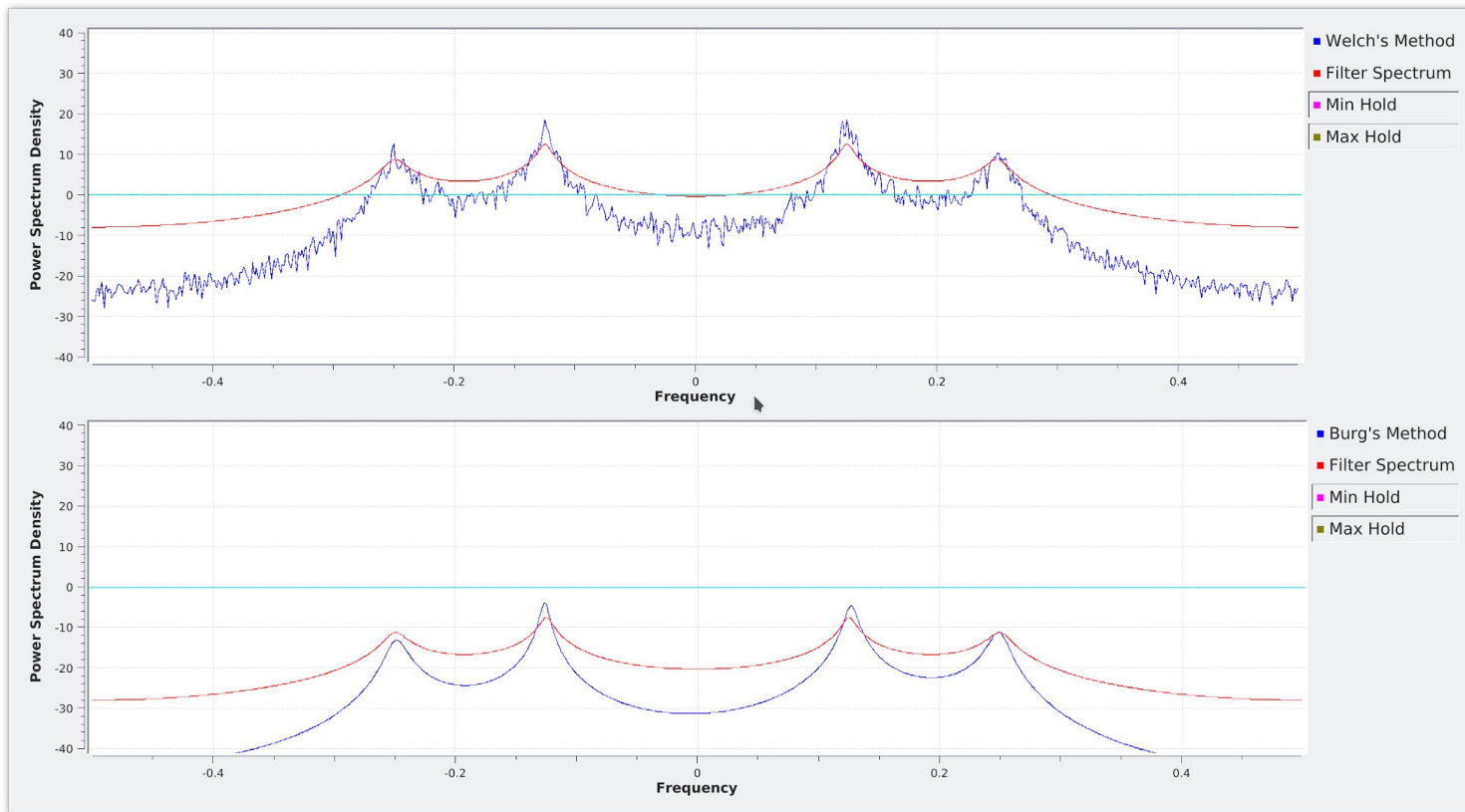
- What if my signal is modeled by an LTI system? Maybe it's a filter?
- How about I directly try and estimate the filter, instead of the entire signal?



- Common algorithm: Burg's Method
  - Other algorithms: Yule-Walker Method, Fast (modified) Covariance Method
  - Other underlying principles: Wiener-Hopf Equation, Levinson-Durbin Recursion  
PARCOR-Coefficients

**DEMO**

# burg.grc



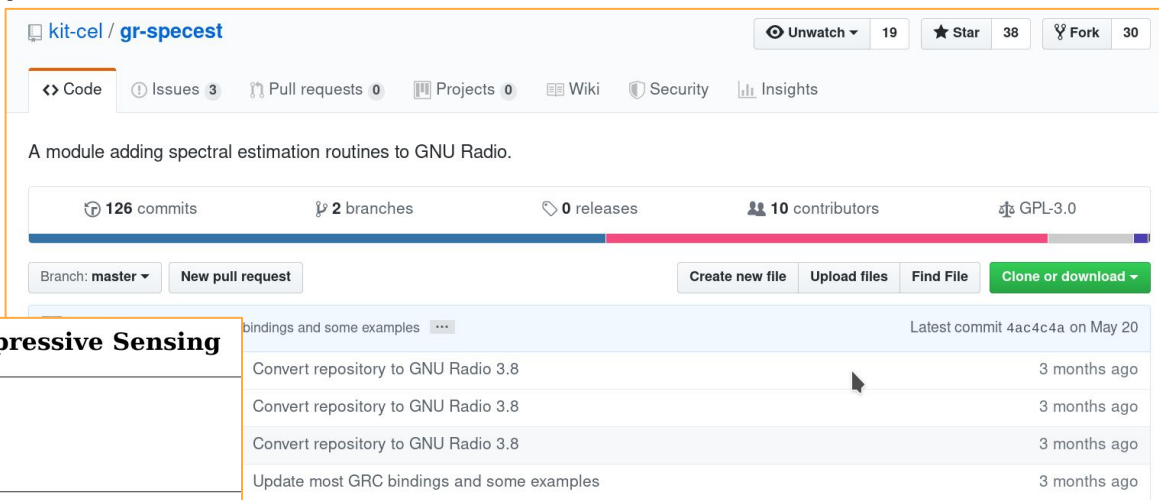
gr-specest's nice little history

# gr-specest's origins

- New challenge: Germany switched to a Bachelor's / Master's system
- In our case, students need to do Bachelor's-Level graduation projects
- What can they do?
  - Need solid practical experience (might go look for a job)
  - Need solid theoretical foundations (might get a Master's or Dr.-Ing.)
  - Should be fun
- Solution: Throw the students at GNU Radio, but let them figure out something with a good theoretical background
  - Easy way to get people in the community
  - Students had relevant open source code out there for employers who care

# gr-specest on the webs

- We quickly disseminated and praised gr-specest everywhere, put it on CGRAN
- Students could point show their moms they were famous on the internet



kit-cel / gr-specest

Unwatch 19 Star 38 Fork 30

Code Issues 3 Pull requests 0 Projects 0 Wiki Security Insights

A module adding spectral estimation routines to GNU Radio.

126 commits 2 branches 0 releases 10 contributors GPL-3.0

Branch: master New pull request Create new file Upload files Find File Clone or download

bindings and some examples Latest commit 4ac4c4a on May 20

Convert repository to GNU Radio 3.8	3 months ago
Convert repository to GNU Radio 3.8	3 months ago
Convert repository to GNU Radio 3.8	3 months ago
Update most GRC bindings and some examples	3 months ago

## [Discuss-gnuradio] Spectral Estimation and Compressive Sensing

**From:** Martin Braun

**Subject:** [Discuss-gnuradio] Spectral Estimation and Compressive Sensing

**Date:** Thu, 5 Mar 2009 11:47:17 +0100

**User-agent:** Mutt/1.5.17 (2007-11-01)

Hi List,

I am happy to say that we from the INT have managed to merge some of our research with GNU Radio development and have released some code on CGRAN. There are two new projects:

1) Spectral Estimation Toolbox

This project aims to enhance GNU Radio with 'proper' spectral estimation routines; so far it only includes Welch's method as a hierarchical block.

# List of all gr-specest algorithms

- Welch's Method (MA and Single-Pole)
- MUSIC (Root-MUSIC, MUSIC Pseudospectrum)
- ESPRIT (Frequency Estimator, Pseudospectrum)
- Thomson's Multitaper Method (MTM)
- Burg's Method
- Fast (modified) covariance method
- FFT Accumulation Method (Cyclostationary Processing, GUI version not yet ported to 3.8)
  
- Utilities: Vector Moving Average, Vector Reciprocal ( $1/x$ ), Stream to Vector Overlap, Pad Vector

# What else came out of gr-specest?

- A group of students benchmarked various math libraries (in 2010) and found that Fortran90 beat the competition -> So of course we now have Fortran in GNU Radio
- It started making us frustrated with creating OOTs. A few OOTs later, modtool was born.
- We learned there were more ways to capitalize on students' work

```
SUBROUTINE ZESPRIT(SAMPLES, LSAMPLES, N, M, OMEGAS)
  IMPLICIT NONE
  INTEGER :: LSAMPLES, N, M, I
  COMPLEX*16 :: SAMPLES(LSAMPLES), R(M,M), EV(N)
  DOUBLE PRECISION :: OMEGAS(N)

  CALL ZCORREST(SAMPLES, LSAMPLES, M, R)
  CALL ZESPRIT_COMMON(R, N, M, EV)

  DO I = 1, N
    OMEGAS(I) = ATAN2(AIMAG(EV(I)), REAL(EV(I)))
  END DO

END SUBROUTINE ZESPRIT
```