Introduction to LLRFLibsPy Library

Zheqiao Geng

RF Forum
14.09.2023

Outline

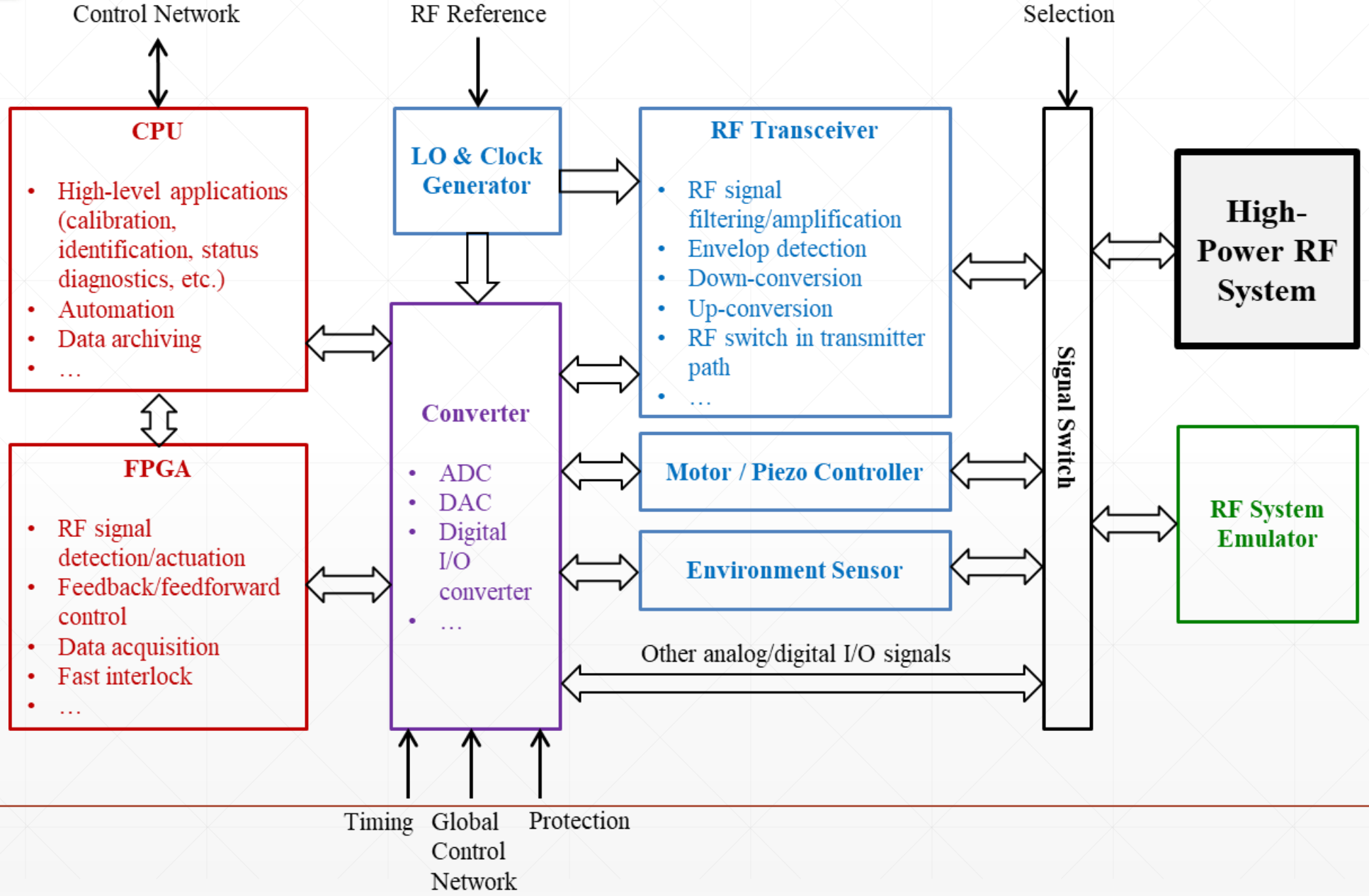
- Motivation
- LLRFLibsPy Overview
- Library Functions
  - RF Simulation
  - RF Control
  - RF Detection and Actuation
  - RF Noise Analysis
  - RF Calibration
  - RF System Identification
  - Other functions
- Status and Outlook

Motivation

State-of-the-art LLRF System R&D Topics

- **Push RF field control performance to extreme** (e.g., very-low-noise RF signal detector, ultra-large control bandwidth with minimum loop delay, high-frequency synthesizer with fast DACs, enhanced RF-mechanical control strategies, etc.)
- **Apply advanced control algorithms** (e.g., optimal control, robust control, disturbance-observer-based control, adaptive feedback/feedforward, etc.)
- **Improve the automation level and intelligence** (e.g., large-scale state machine, optimization of RF system operation with machine learning surrogate models, smart exception detection and handling, etc.)
- **RF control strategy for new scenarios** (e.g., multiple-harmonic beam-loading control in wide-band cavities, etc.)
- **Standardization** (e.g., standard hardware like MicroTCA or cPCI-serial, firmware libraries, software libraries, etc.)

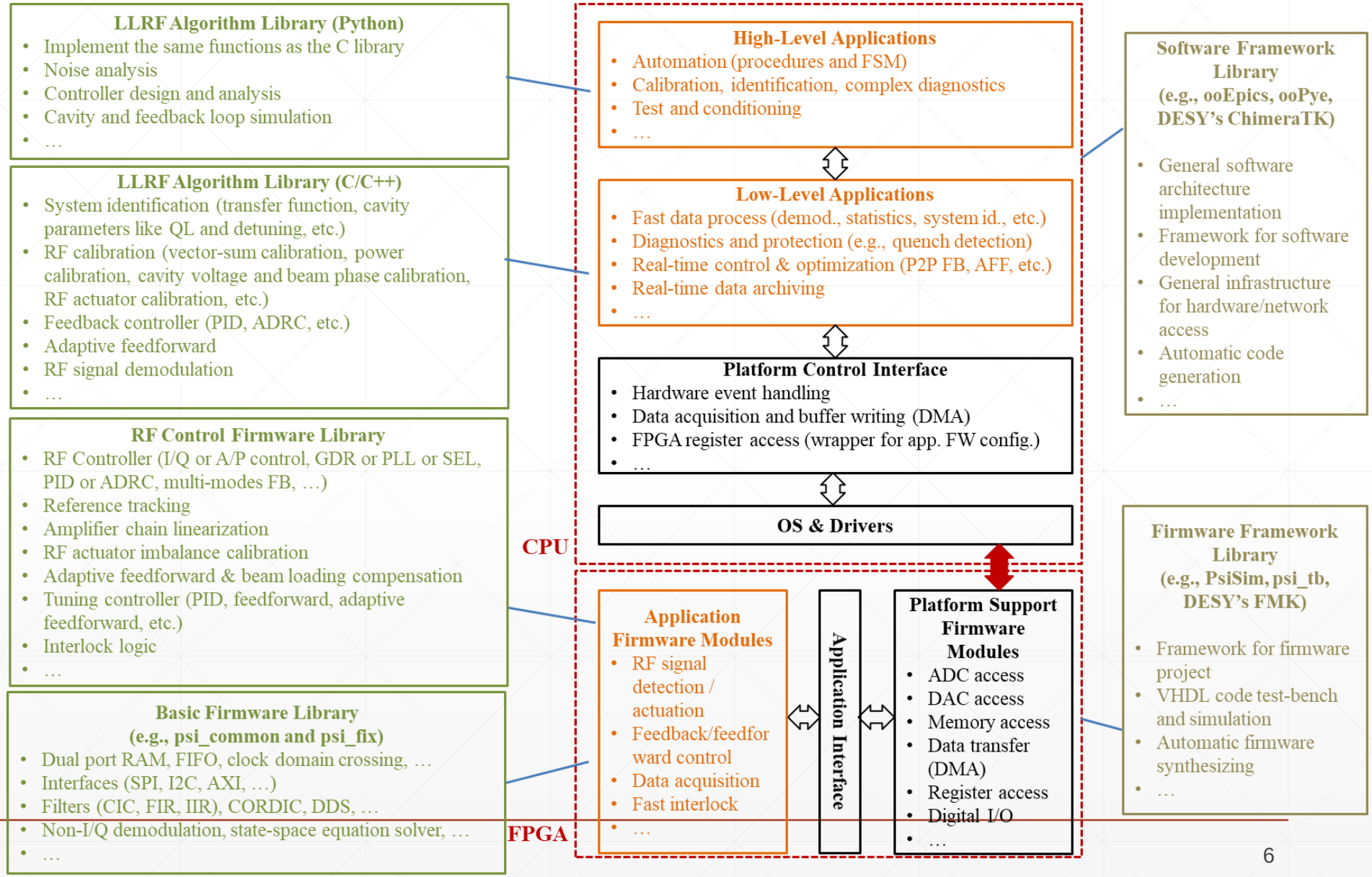
Functional Architecture of a LLRF System



Functional Architecture of LLRF Firmware & Software

Firmware Libraries

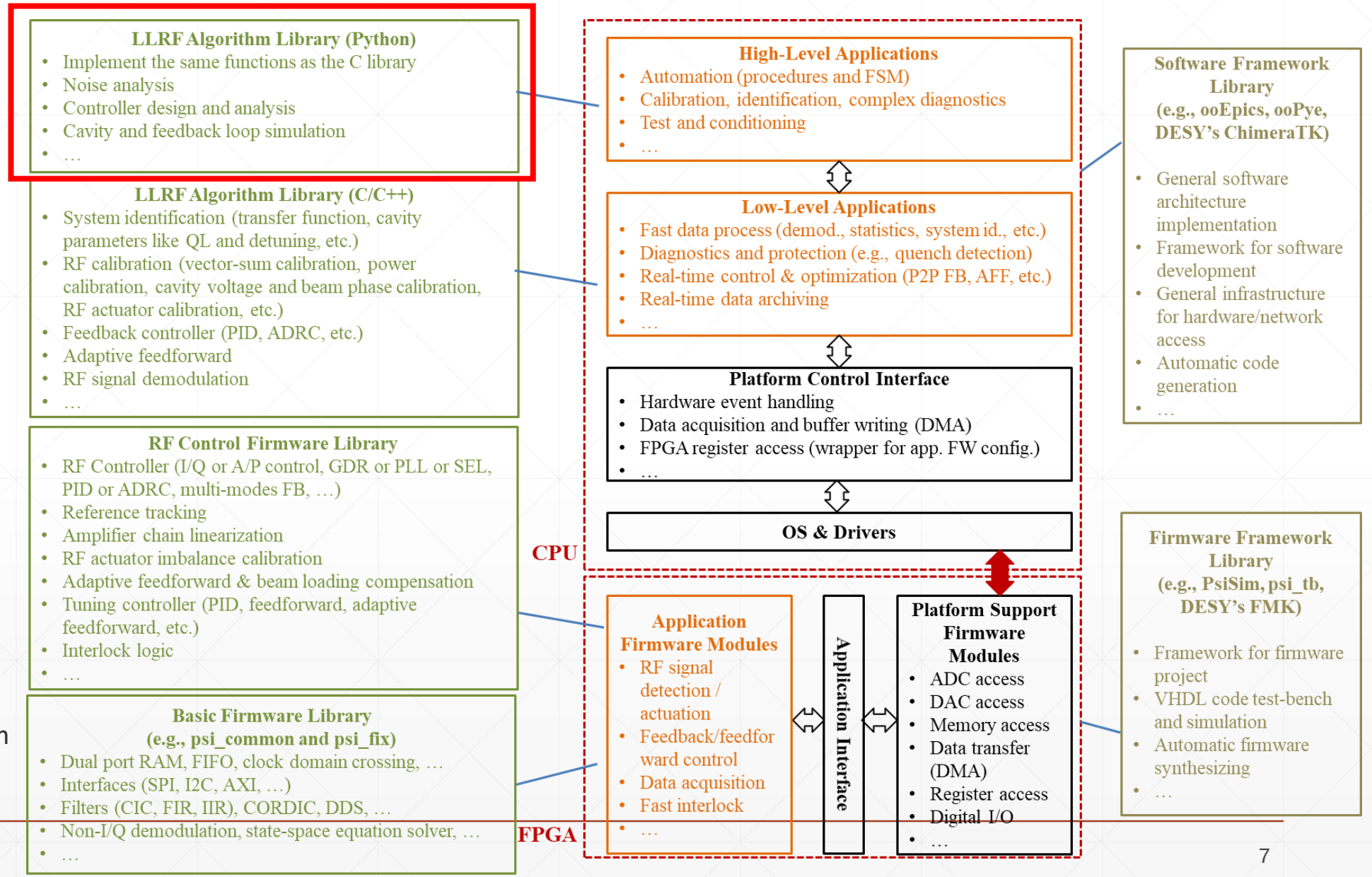
- **Firmware Framework Library**
  - Construct the firmware project structure
  - Simulate the firmware code
  - Automate the firmware synthesize
  - Access common components (e.g., ADC, DAC, etc.)
- **Basic Firmware Library**
  - Collect firmware build components common for different applications
- **RF Control Firmware Library**
  - Implement high-level LLRF control modules
  - Can be assembled and configured for fast prototyping of LLRF controllers



Functional Architecture of LLRF Firmware & Software (cont.)

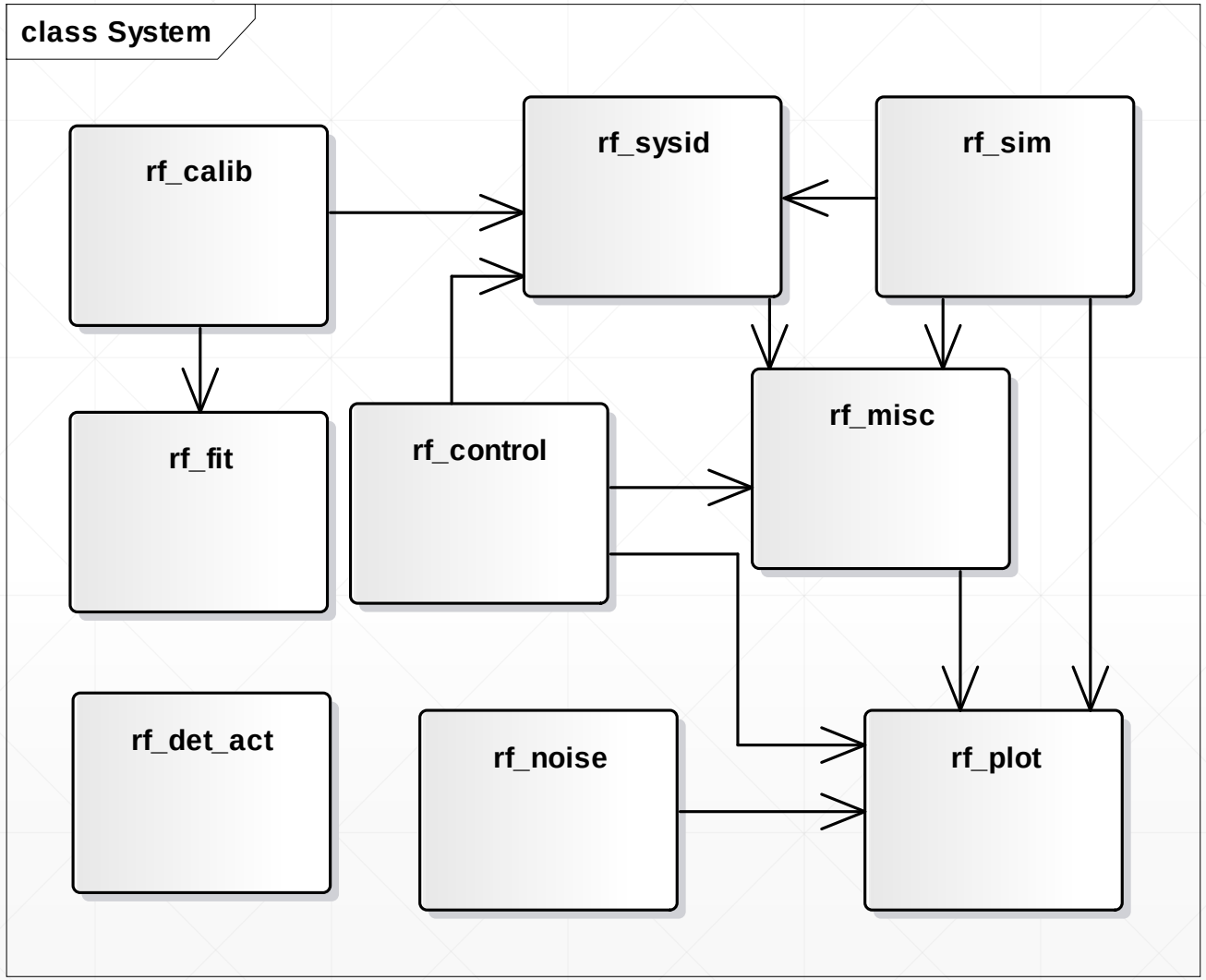
Software Libraries

- **Software Framework Library**
  - Define a common software architecture
  - Implement infrastructures for multi-threading, hardware access, and network access
- **LLRF Algorithm Library (C/C++)**
  - Implement RF control domain algorithms
  - Support soft-real-time applications
- **LLRF Algorithm Library (Python)**
  - Implement same algorithms as the C/C++ library
  - Also implement features like noise analysis, feedback controller design, and system simulation
  - For high-level applications and automation

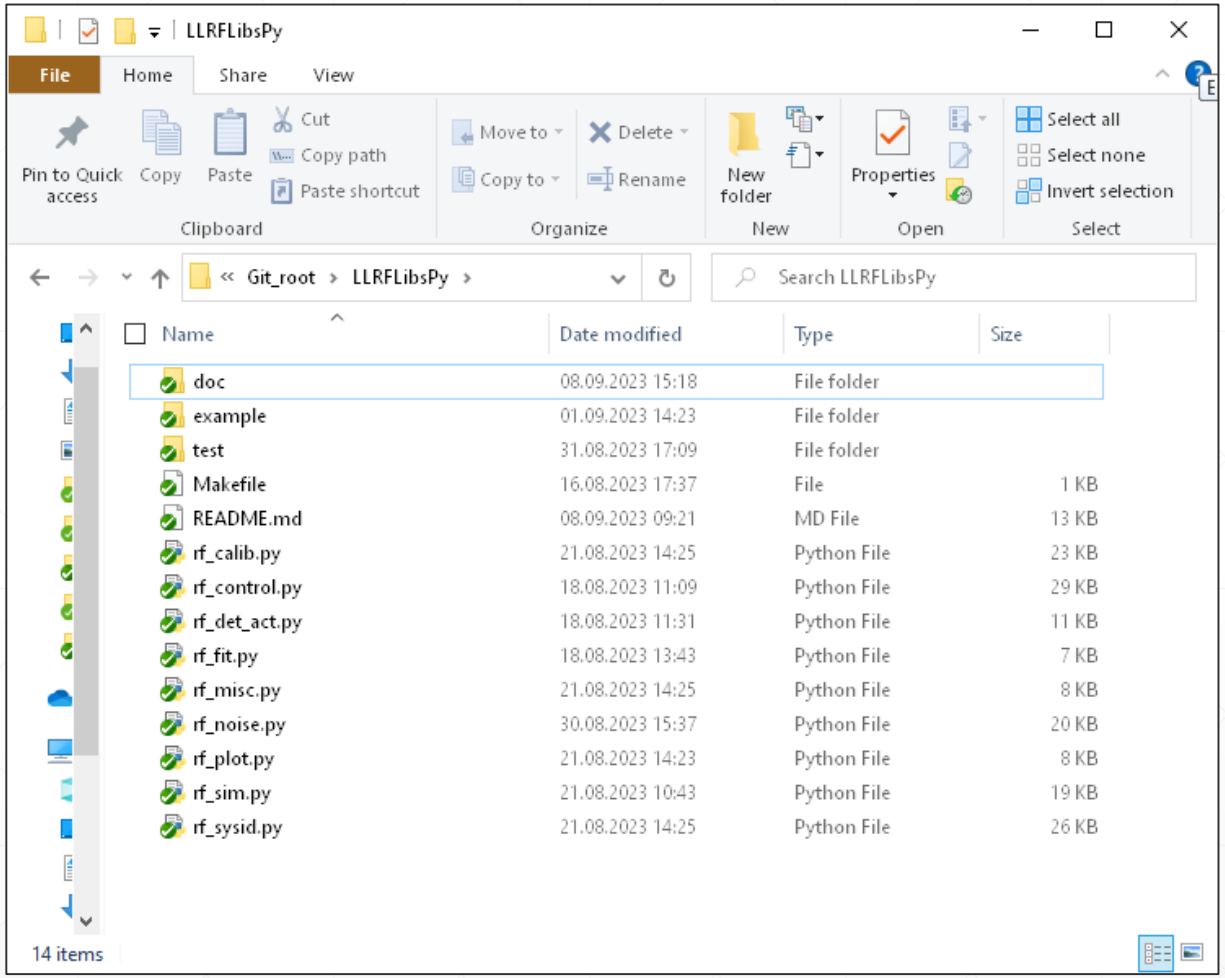


LLRFLibsPy Overview

Contents of LLRFLibsPy



Python modules in LLRFLibsPy



Files in the Git repository

Contents of LLRFLibsPy (cont.)

Description of Modules

`rf_sim`	Simulate the cavity response; determine the operation parameters
`rf_control`	Design and analyze RF feedback/feedforward controllers
`rf_det_act`	Measure RF amplitude and phase from ADC samples
`rf_noise`	Noise analysis, generation, and filtering
`rf_calib`	RF calibrations like virtual probe, RF actuator offset/imbalance, forward and reflected, and power calibrations
`rf_sysid`	Identify the system transfer function and characteristic parameters (e.g., cavity QL and detuning)
`rf_fit`	Fit data to sine/cosine, circle, ellipse or Gaussian functions
`rf_misc`	Misc. functions like saving data to files and reading data from files
`rf_plot`	Plot functions for internal usage

Note: the library relies on "`numpy`" and "`scipy`", "`matplotlib`" is also loaded if any plot function is enabled.

Library Functions

RF Simulation

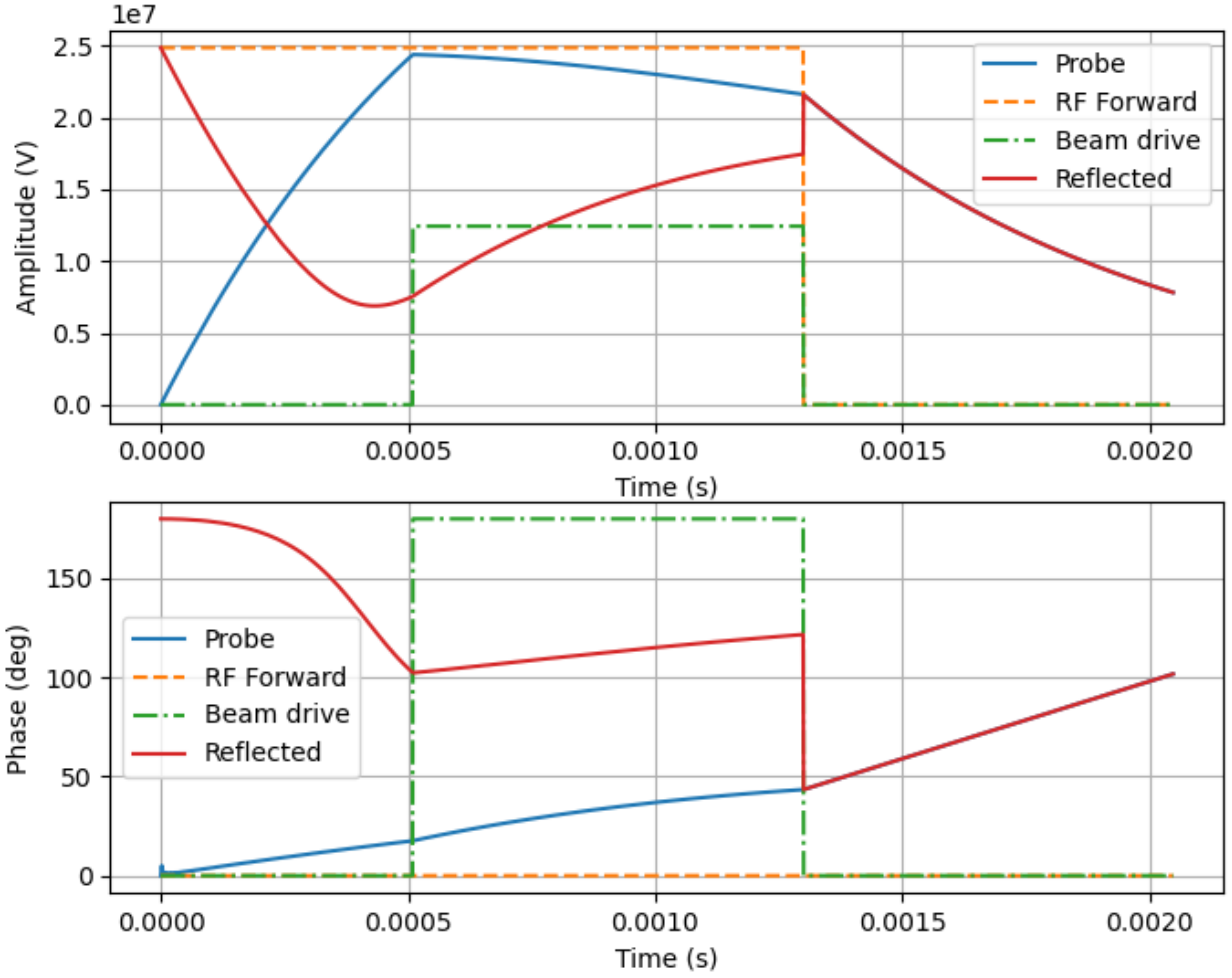
Function	Example	Comments
cav_ss	example_sim_cavity_basic.py	Derive a continuous state-space equation of a cavity
cav_ss_mech		Cavity state-space equation with mechanical modes (to be implemented)
cav_impulse	example_aff_ilc.py	Derive the cavity impulse response from the cavity parameters
sim_ncav_pulse	example_sim_cavity_basic.py	Simulate cavity (with constant QL and detuning) response to a pulsed input
sim_ncav_step	example_sim_cavity_basic.py	Simulate cavity (with constant QL and detuning) response for a time step
sim_ncav_step_simple	example_sim_cavity_basic.py	Simulate cavity (with constant QL and detuning) response for a time step (simplified cavity equation only with the fundamental passband mode)
rf_power_req	example_power_req.py example_power_req2.py	Calculate the required RF power for desired cavity voltage and beam current
opt_QL_detuning	example_power_req.py example_power_req2.py	Calculate the optimal QL and detuning for minimizing the reflection power

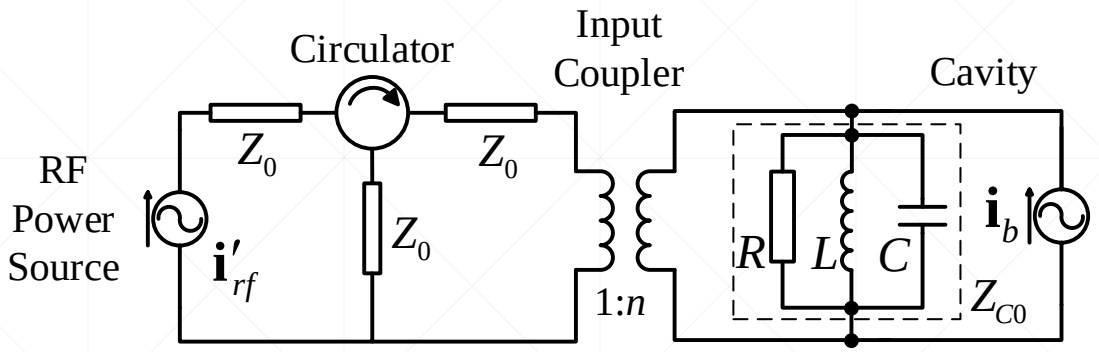
RF Simulation (cont.)

example_sim_cavity_basic.py

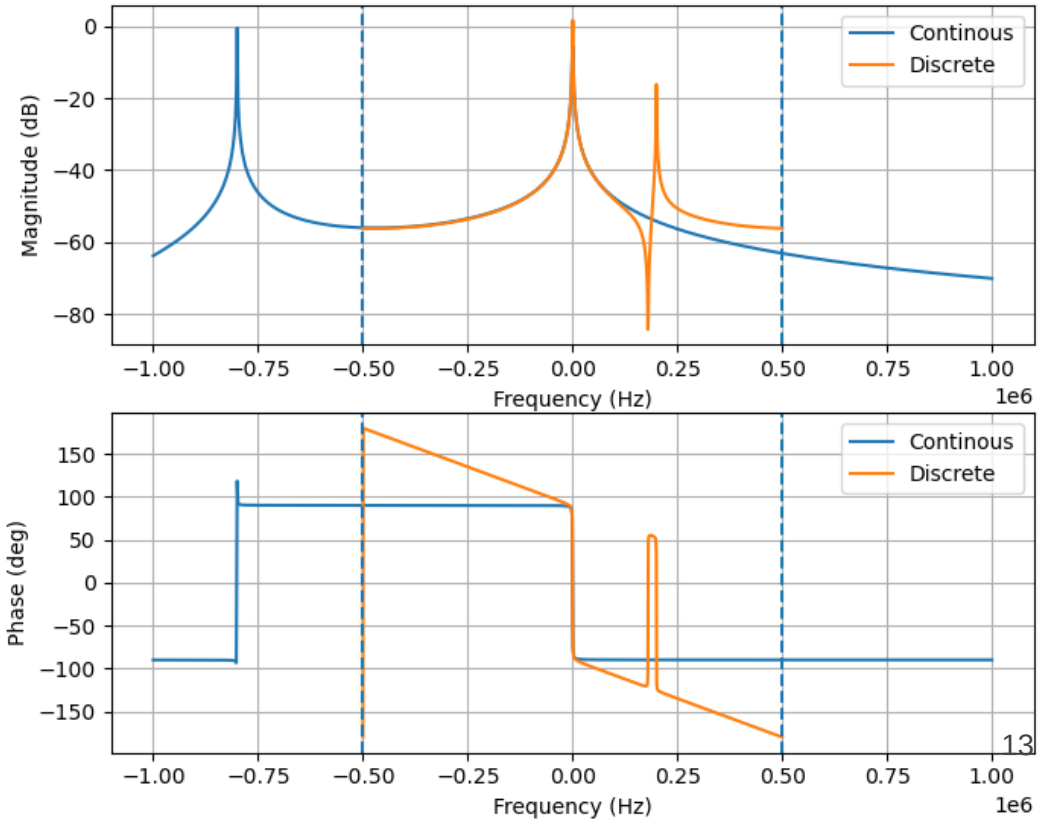
$$\begin{bmatrix} \dot{v}_{CI} \\ \dot{v}_{CQ} \end{bmatrix} = \begin{bmatrix} -\omega_{1/2} & -\Delta\omega \\ \Delta\omega & -\omega_{1/2} \end{bmatrix} \begin{bmatrix} v_{CI} \\ v_{CQ} \end{bmatrix} + \omega_{1/2} R_L \begin{bmatrix} i_{CI} \\ i_{CQ} \end{bmatrix}$$

Cavity Response Simulated with Continuous State-space Model





State-space System Discretization

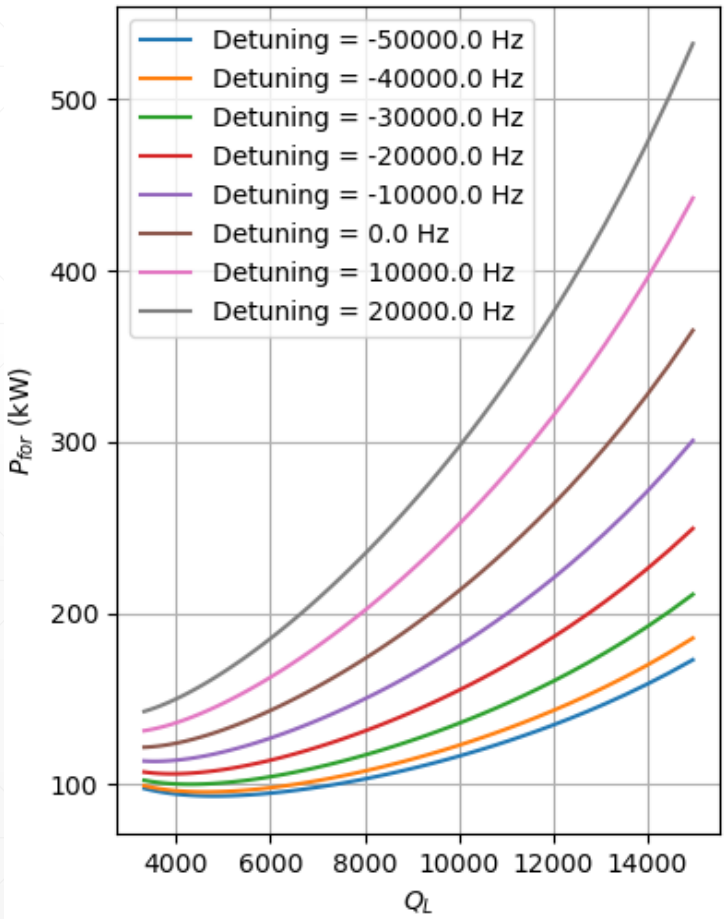


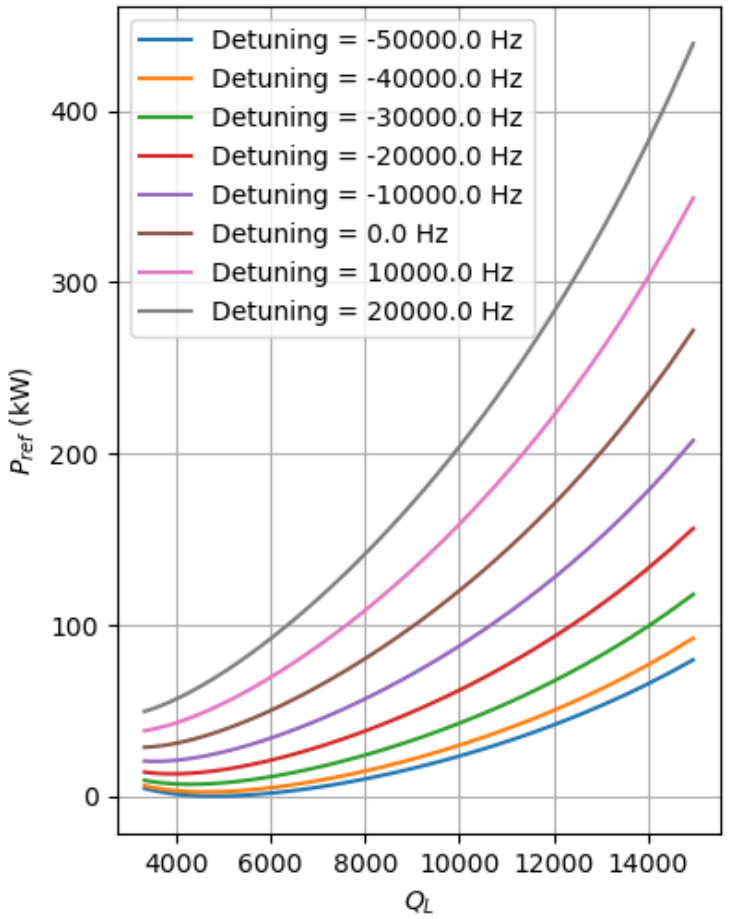
RF Simulation (cont.)

example_power_req2.py

$$P_{for} = \frac{\beta+1}{\beta}\frac{v_{C0}^2}{8R_L}\left[\left(1+\frac{2R_L i_{b0}\cos\varphi_b}{v_{C0}}\right)^2+\left(\tan\psi+\frac{2R_L i_{b0}\sin\varphi_b}{v_{C0}}\right)^2\right]$$

$$P_{ref} = \frac{\beta+1}{\beta}\frac{v_{C0}^2}{8R_L}\left[\left(\frac{\beta-1}{\beta+1}-\frac{2R_L i_{b0}\cos\varphi_b}{v_{C0}}\right)^2+\left(\tan\psi+\frac{2R_L i_{b0}\sin\varphi_b}{v_{C0}}\right)^2\right]$$





Required RF power for a cavity with following parameters:

- Cavity resonance frequency: 499.5935 MHz
- Cavity voltage: 0.35 MV
- Beam average current: 0.4 A
- Beam sync. phase (ring convention): 147.6 deg
- Cavity Q0: 29909
- r/Q = 3.4/Q0 MΩ

Optimal beta = 5.17, QL = 4848, detuning = -54775 Hz

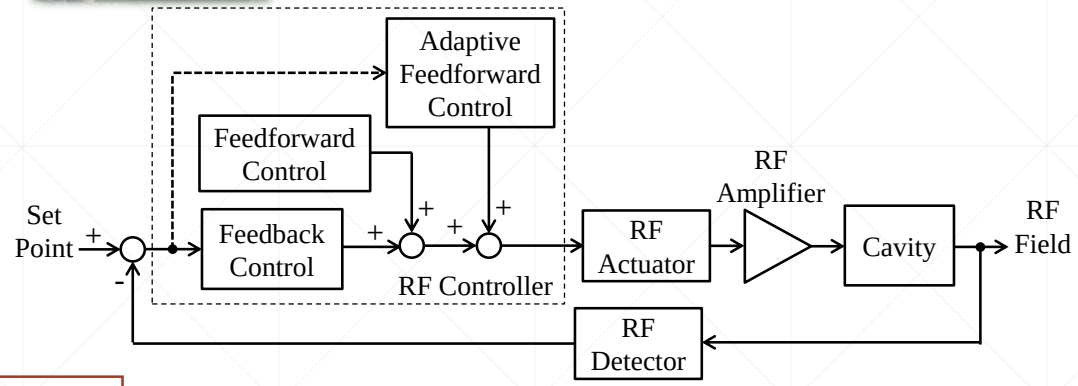
RF Control

Function	Example	Comments
ss_discrete	example_sim_cavity_basic.py	Discretize a continuous state-space system and compare the frequency responses
ss_cascade	example_feedback_analysis.py	Cascade two state-space systems (either continuous or discrete - C/D)
ss_freqresp	example_feedback_analysis.py	Calculate and plot the frequency response of a state-space system (C/D)
basic_rf_controller	example_feedback_basic.py	Derive a basic continuous RF I/Q controller: P + I + frequency notches
control_step	example_feedback_basic.py	Perform one time-step execution of the discretized controller
loop_analysis	example_feedback_analysis.py	Analyze the sensitivity/complementary sensitivity of an RF control loop (C/D)
cav_sp_ff	example_sp_ff.py	Derive the setpoint and feedforward waveforms for desired cavity voltage and beam loading
ADRC_controller	example_feedback_adrc.py	Derive a basic ADRC controller (the observer and gain)
ADRC_control_step	example_feedback_adrc.py	Perform one time-step execution of the discretized controller including the ADRC observer
AFF_timerev_lpf	example_aff_timerev_lpf.py	Time-reversed low pass filter-based adaptive feedforward
AFF_ilc_design	example_aff_ilc.py	Derive the ILC gain matrix from the impulse response and weighting matrices
AFF_ilc	example_aff_ilc.py	Apply the ILC algorithm to calculate the feedforward correction signal
resp_inv_svd	example_resp_matrix_inv.py	Response matrix inversion with SVD (with singular value filtering)
resp_inv_lsm	example_resp_matrix_inv.py	Response matrix inversion with lease-square method (with regularization)

RF Control (cont.)

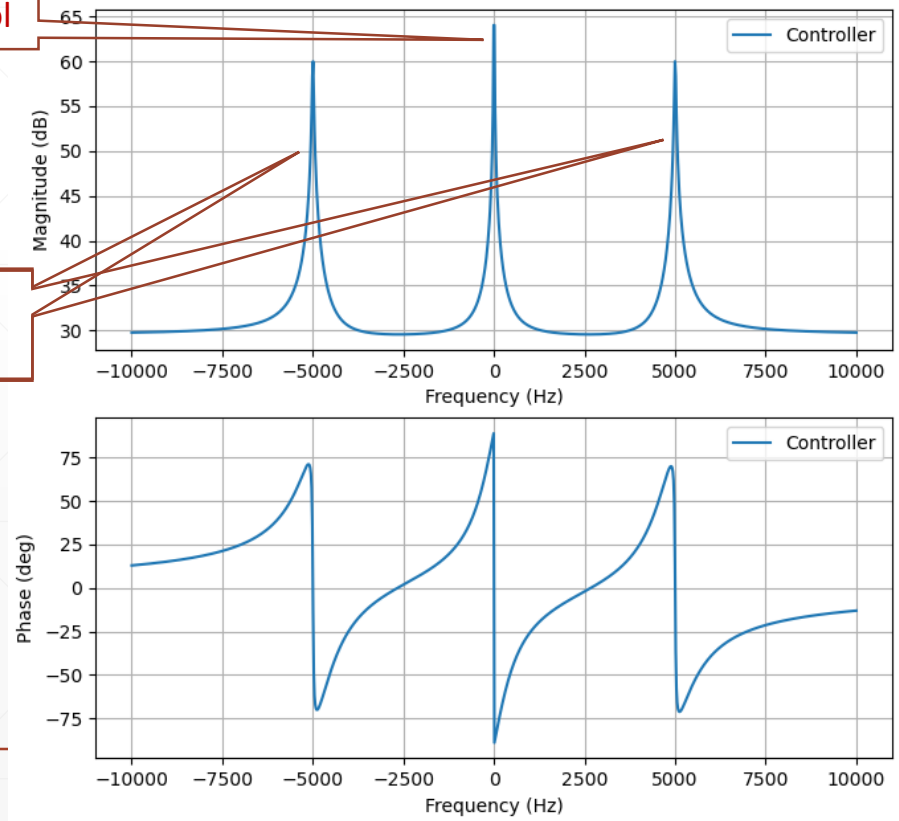
example_feedback_basic.py

$$K(\hat{s}) = K_P + \frac{K_I}{\hat{s}} + \sum_{i=1}^{n} \frac{g_i \omega_{1/2,i}}{\hat{s} + \omega_{1/2,i} - j\omega_{notch,i}} + \sum_{i=1}^{n} \frac{g_i \omega_{1/2,i}}{\hat{s} + \omega_{1/2,i} + j\omega_{notch,i}}$$

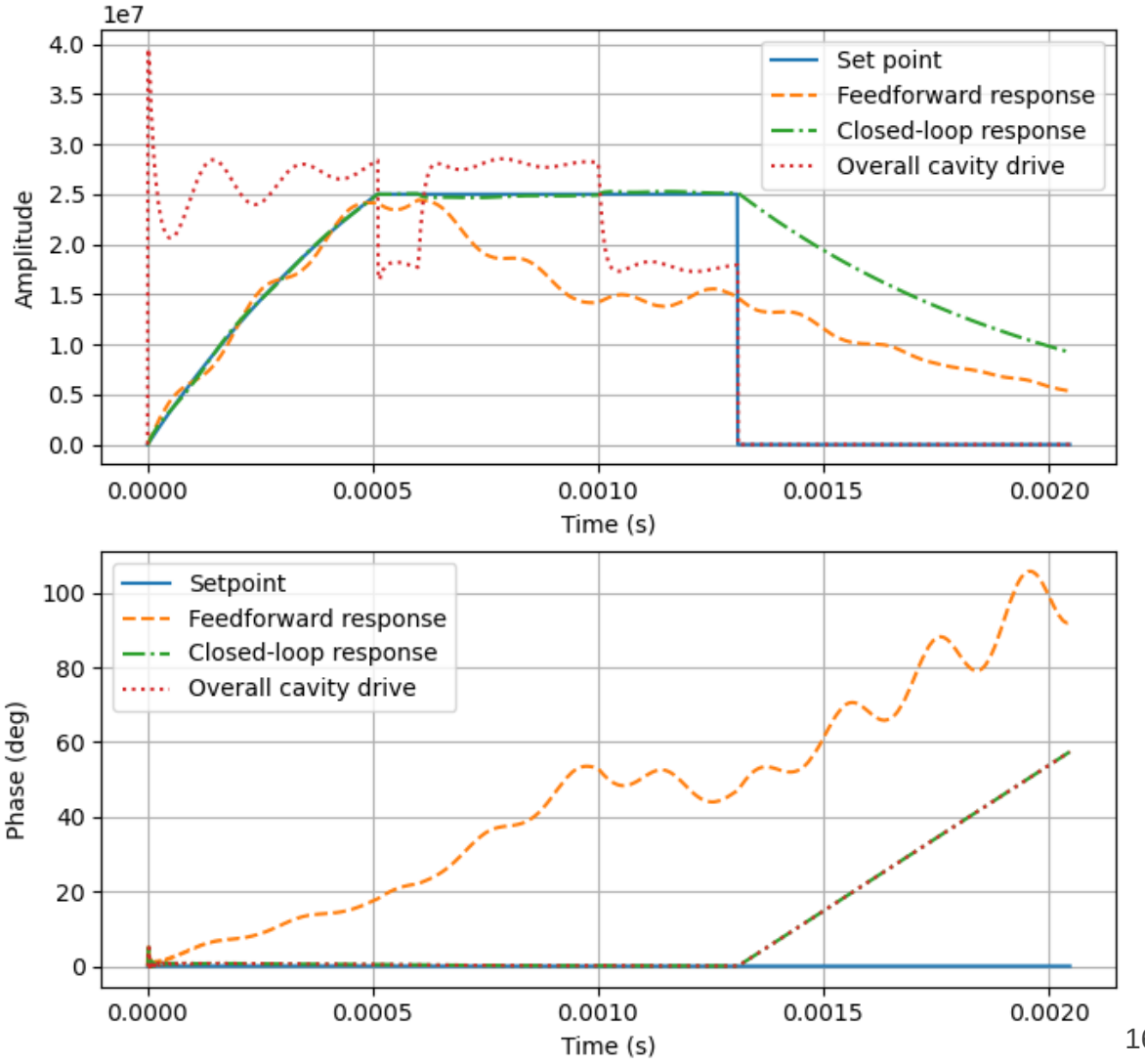


PI Control

Notch filter



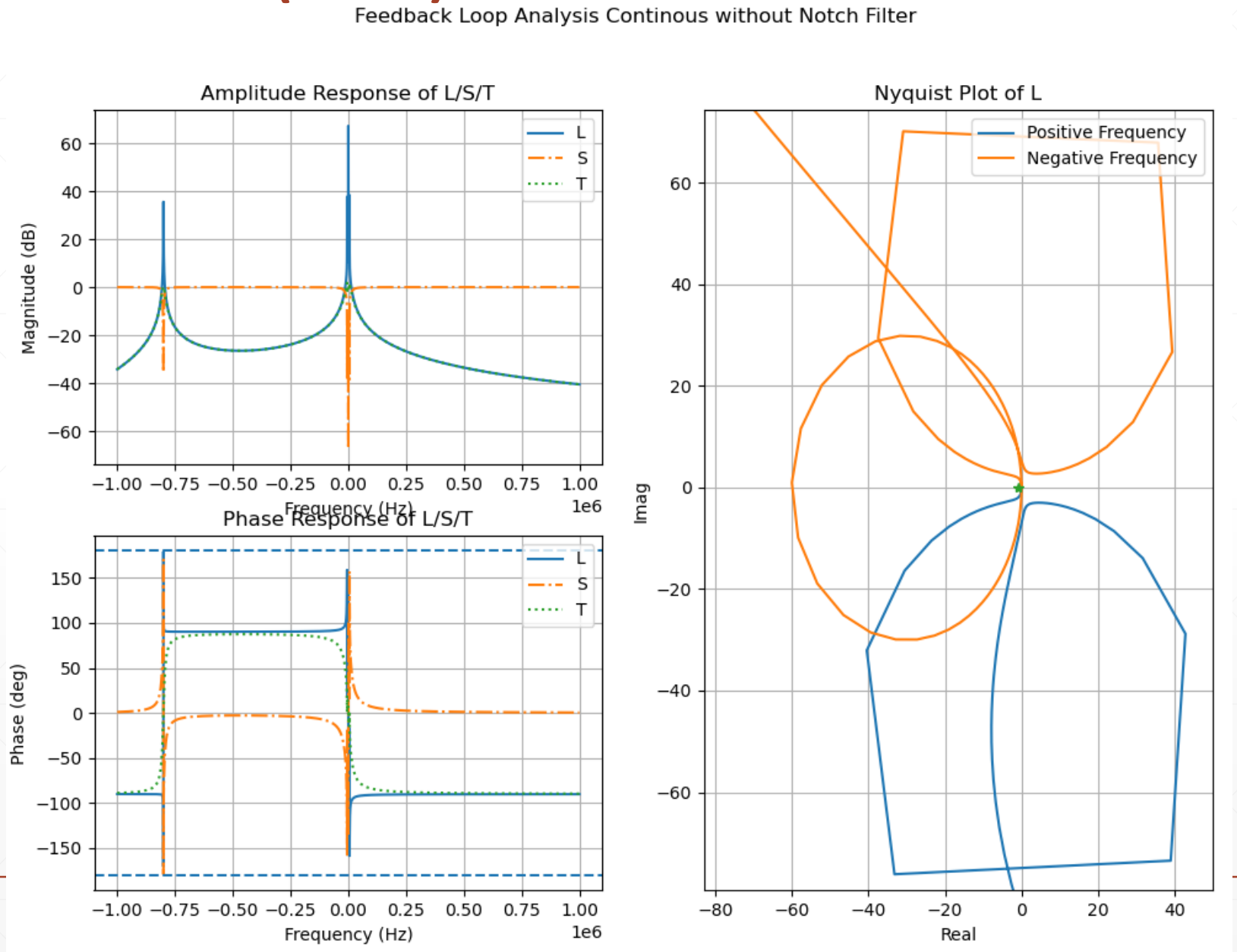
Basic RF Controller frequency response



Cavity response with feedforward or/and feedback

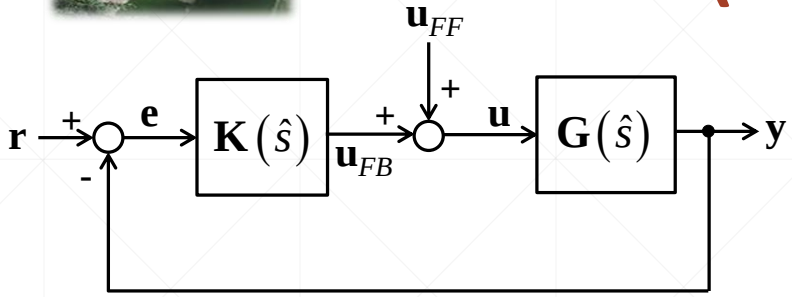
RF Control (cont.)

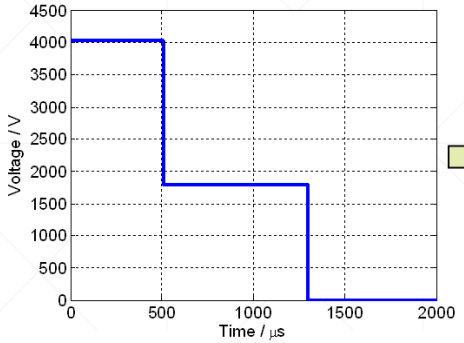
example_feedback_analysis.py



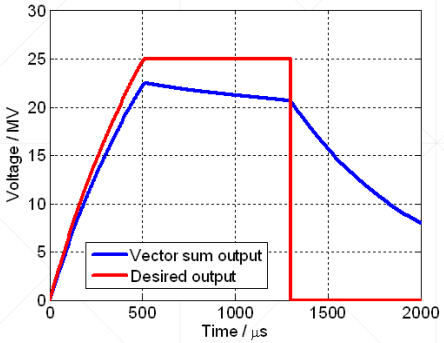
RF Control (cont.)

example_aff_timerev_lpf.py



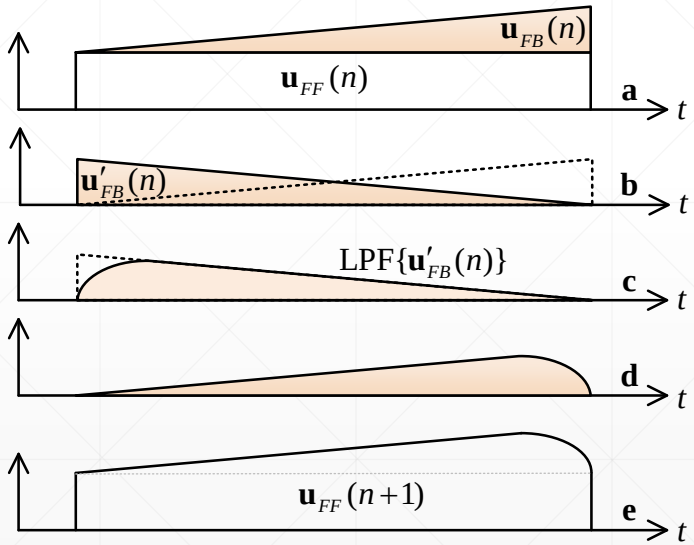


RF System

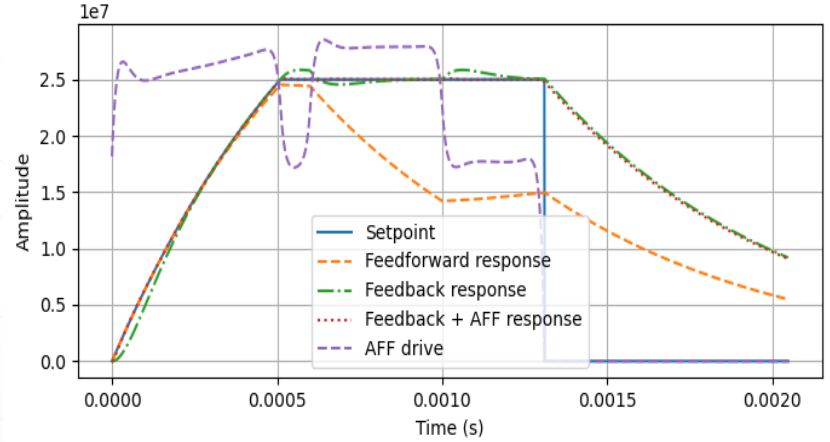


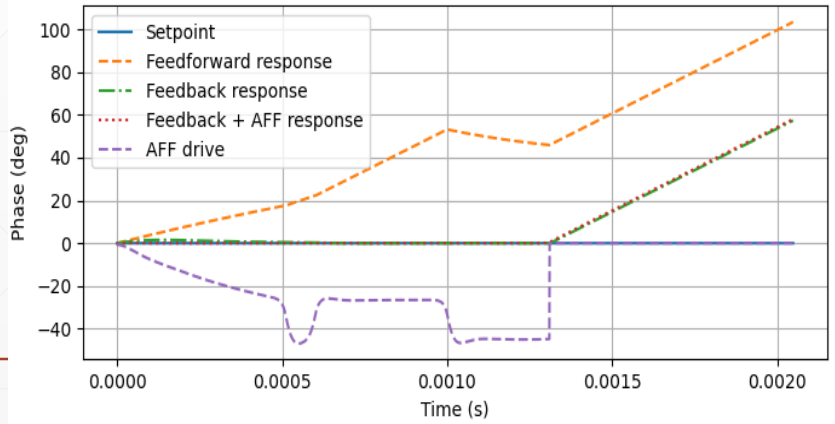
Feedforward Control

Adapt feedforward with feedback on

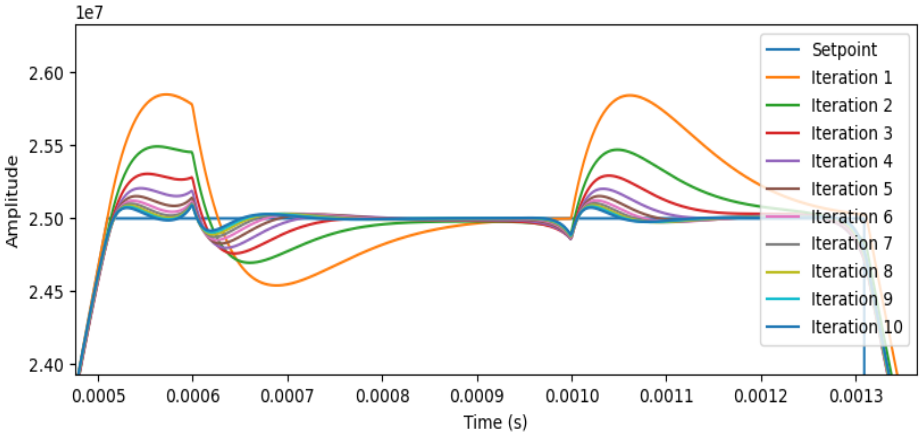


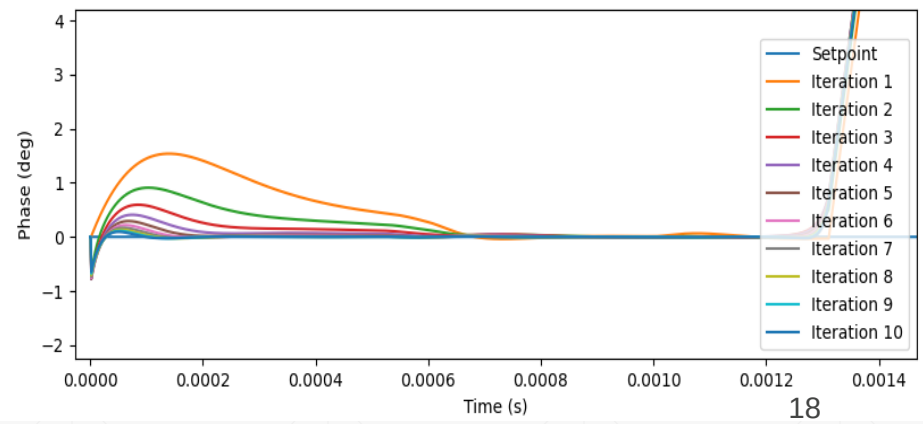
Adaptive feedforward with time-reversed low-pass filter





Cavity response after 10 iterations





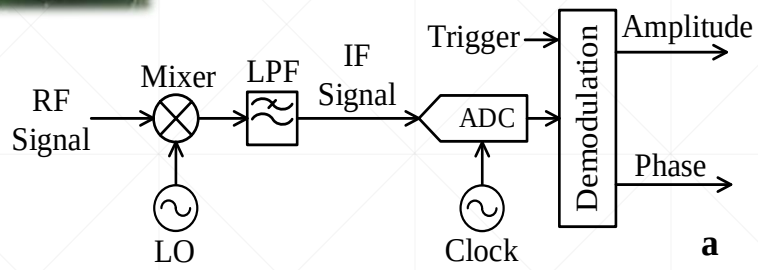
Flattop amplitude/phase convergence

RF Detection and Actuation

Function	Example	Comments
noniq_demod	example_demod.py	Perform non-I/Q demodulation of a given raw sampling waveform
twop_demod	example_demod.py	Demodulate raw waveform with every two samples
asyn_demod	example_demod.py	Demodulate raw waveform sampled by asynchronous clock (a reference signal waveform is needed)
self_demod_ap	example_demod.py	Demodulate raw waveform with Hilbert transform, returning the amplitude and phase waveforms
iq2ap_wf	example_demod.py	Convert I/Q waveforms to amplitude/phase waveforms
ap2iq_wf		Convert amplitude/phase waveforms to I/Q waveforms
norm_phase		Normalize phase (scalar or waveform) to a specific range (default ±180°)
pulse_info		Derive the pulse info like pulse width, pulse offset, etc.

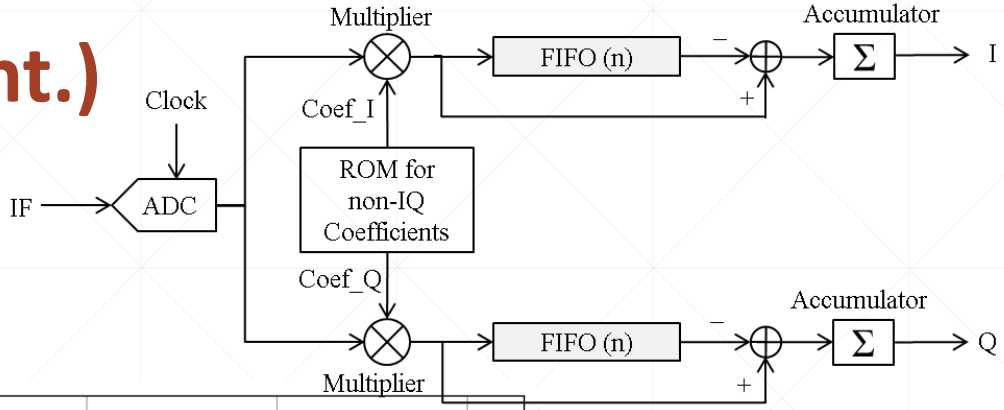
RF Detection and Actuation (cont.)

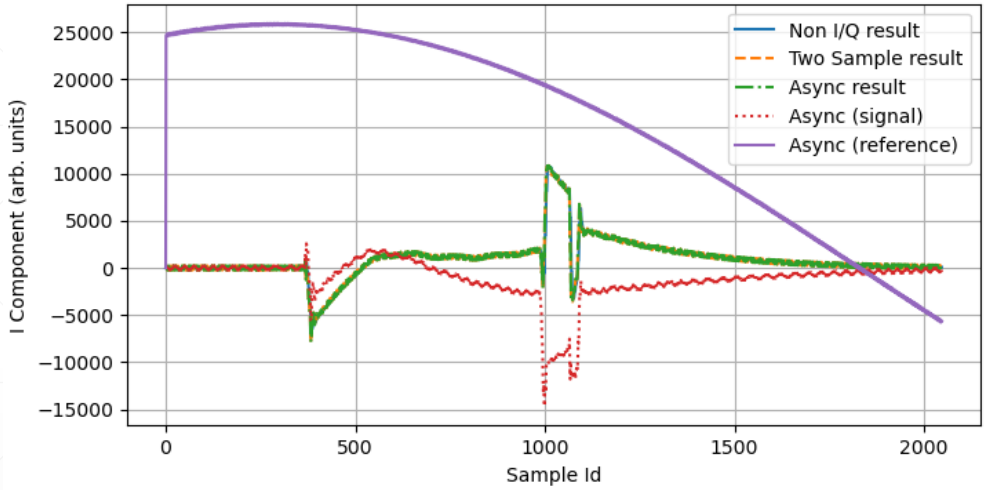
RF signal detection scheme

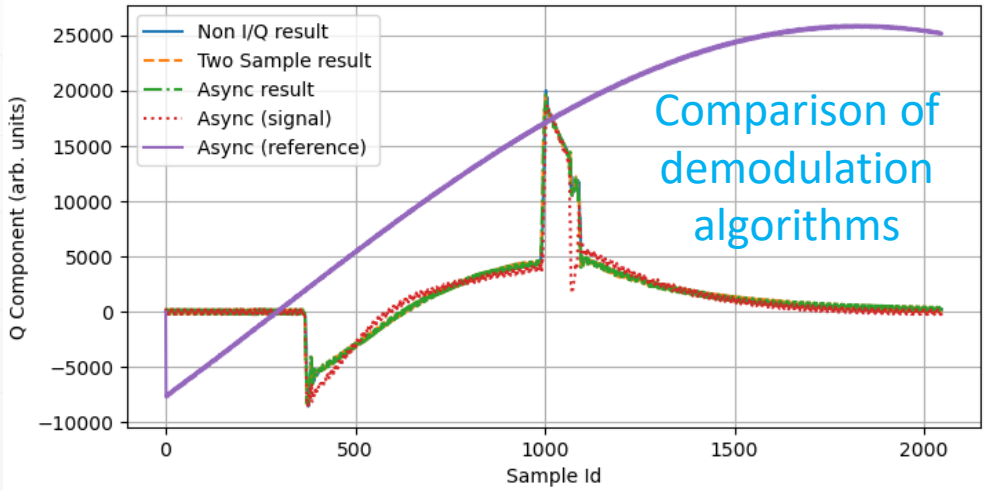


$$I_k = \frac{2}{n}\sum_{l=k-n+1}^{k} x_l \sin(l\Delta\varphi)$$

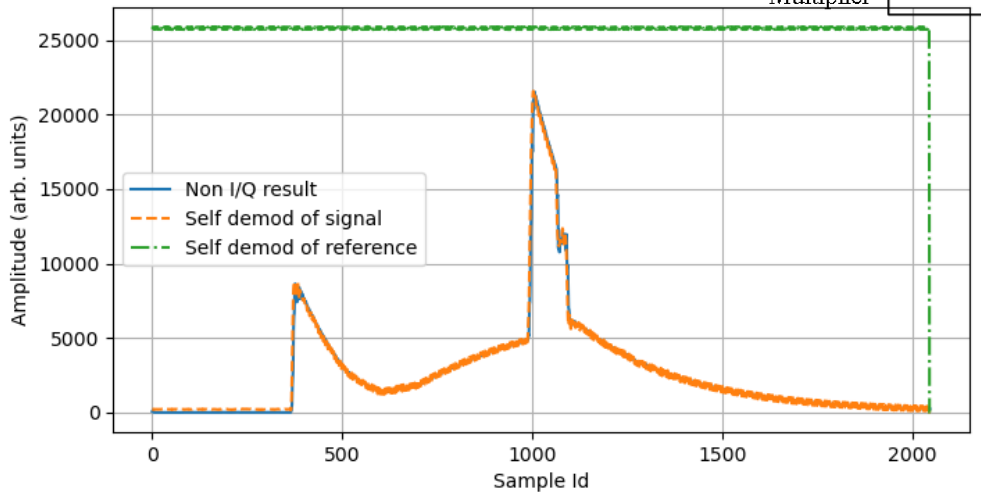
$$Q_k = \frac{2}{n}\sum_{l=k-n+1}^{k} x_l \cos(l\Delta\varphi)$$



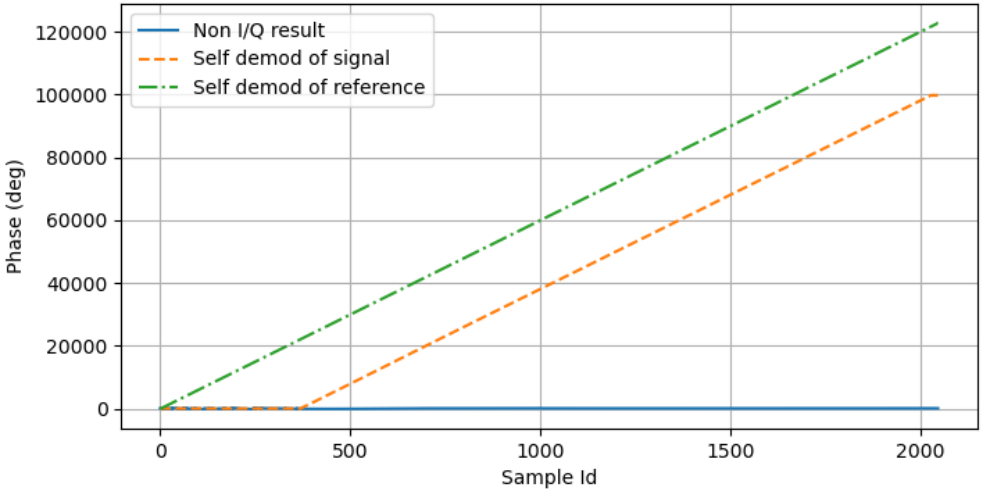




Comparison of demodulation algorithms



Implementation of non-I/Q demodulation algorithm



Self-demodulation with Hilbert Transform

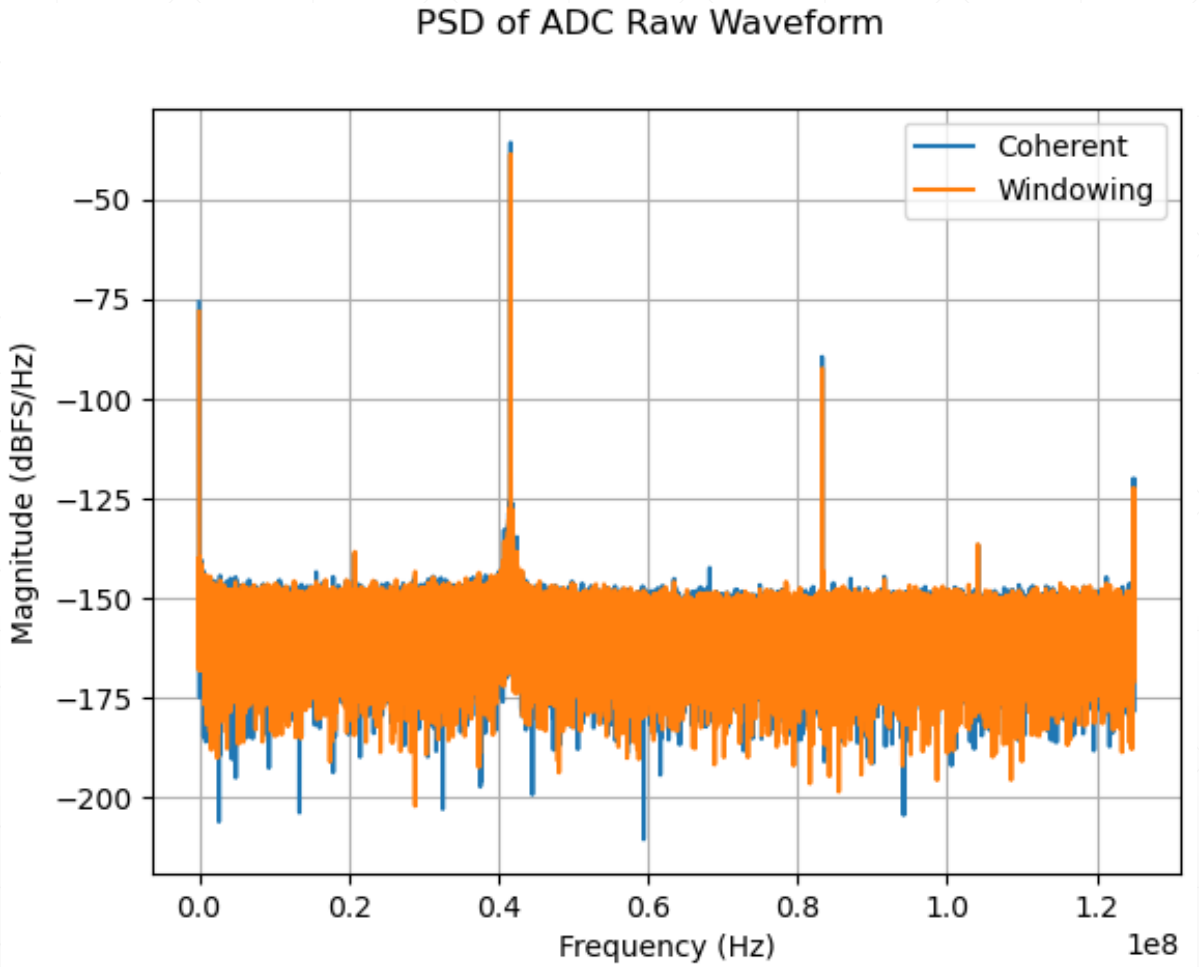
example_demod.py

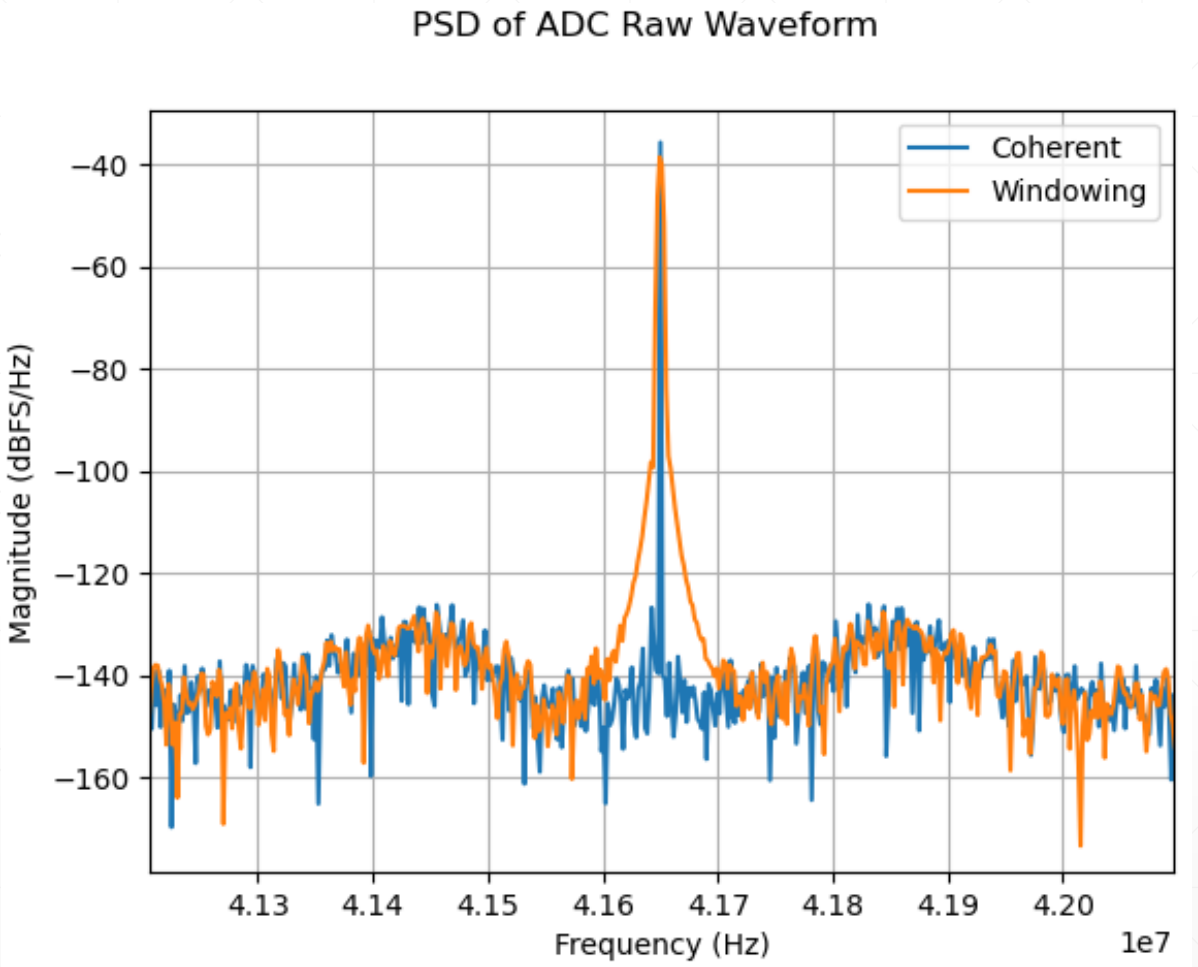
RF Noise Analysis

Function	Example	Comments
calc_psd_coherent	example_noise_psd.py	Calculate the power spectral density (PSD) of a coherent sampled data series
calc_psd	example_noise_psd.py	Calculate the PSD of a general data series
rand_unif		Generate uniform distributed random numbers
gen_noise_from_psd	example_noise_time_series.py	Generate noise series from a given PSD spectrum
calc_rms_from_psd	example_noise_time_series.py	Calculate the RMS jitter from a given PSD spectrum
notch_filt	example_calib_for_ref_sc_cavity.py	Apply notch filter to a data series
design_notch_filter	example_feedback_analysis.py	Design a notch filter in state-space format
filt_step	example_feedback_basic.py	Apply a single time step of state-space filter
moving_avg		Moving average without compensating for the group delay

RF Noise Analysis (cont.)

`example_noise_psd.py`

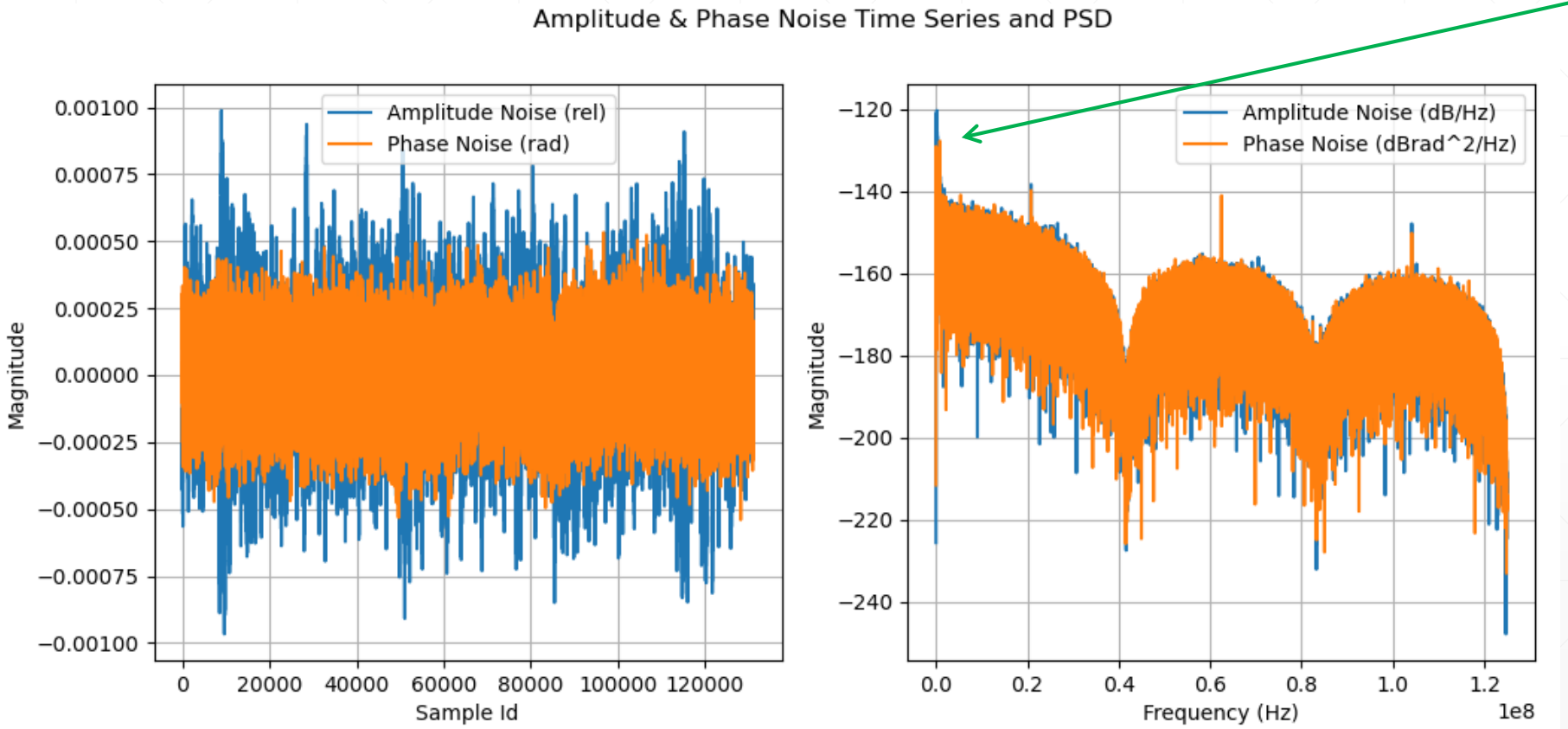


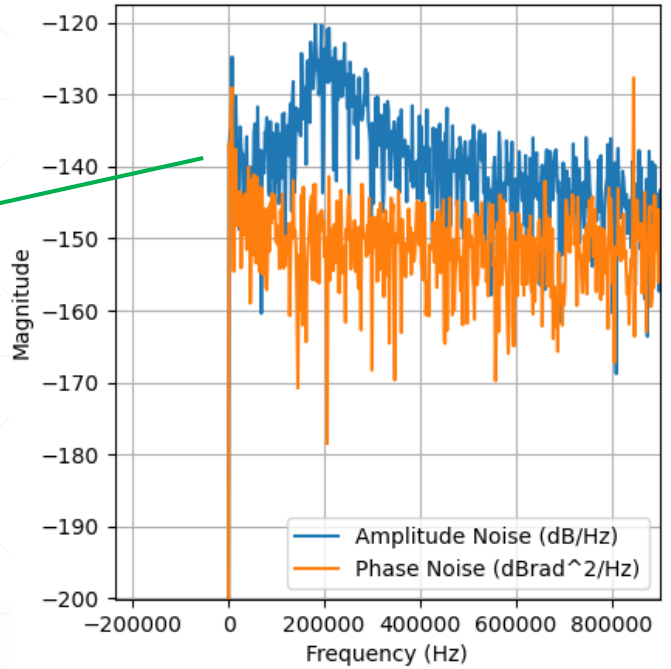


Calculate the Power Spectral Density (PSD) spectrum of ADC raw data from SwissFEL S-band. Compare the spectrum precision of coherent sampling (data covers full IF periods) with that with Blackman windowing

RF Noise Analysis (cont.)

`example_noise_psd.py`



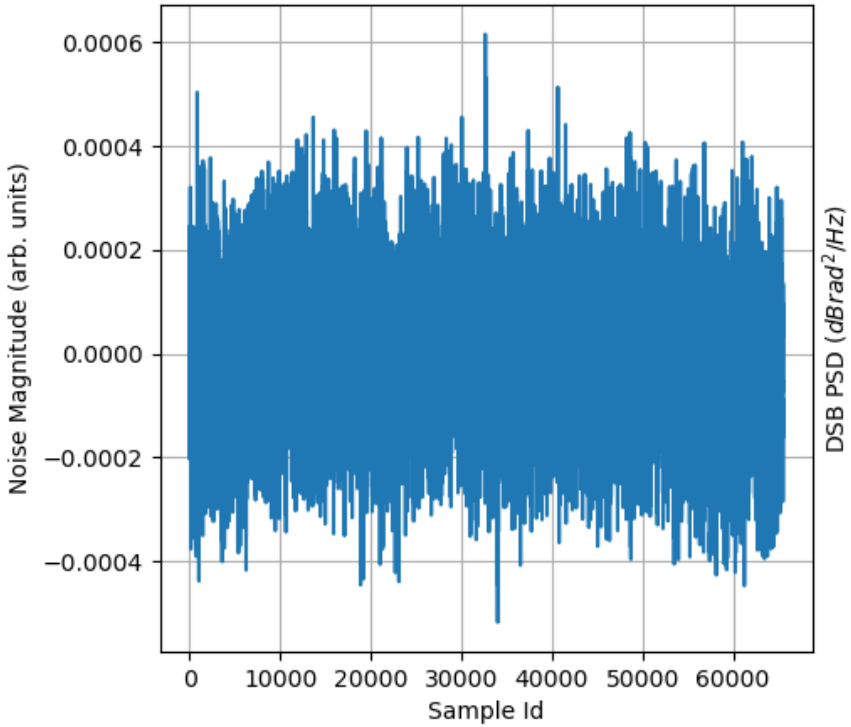


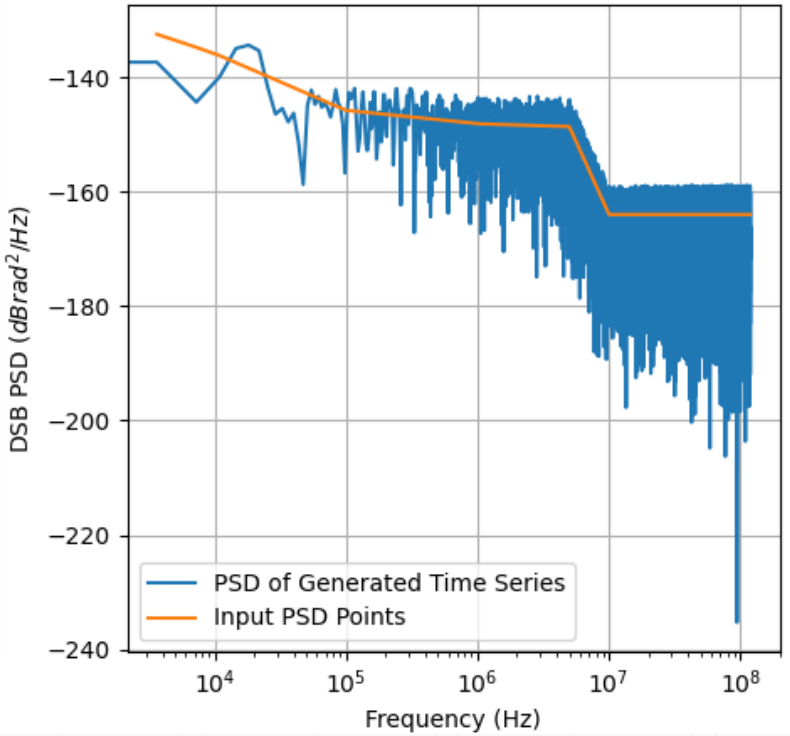
RF Noise Analysis (cont.)

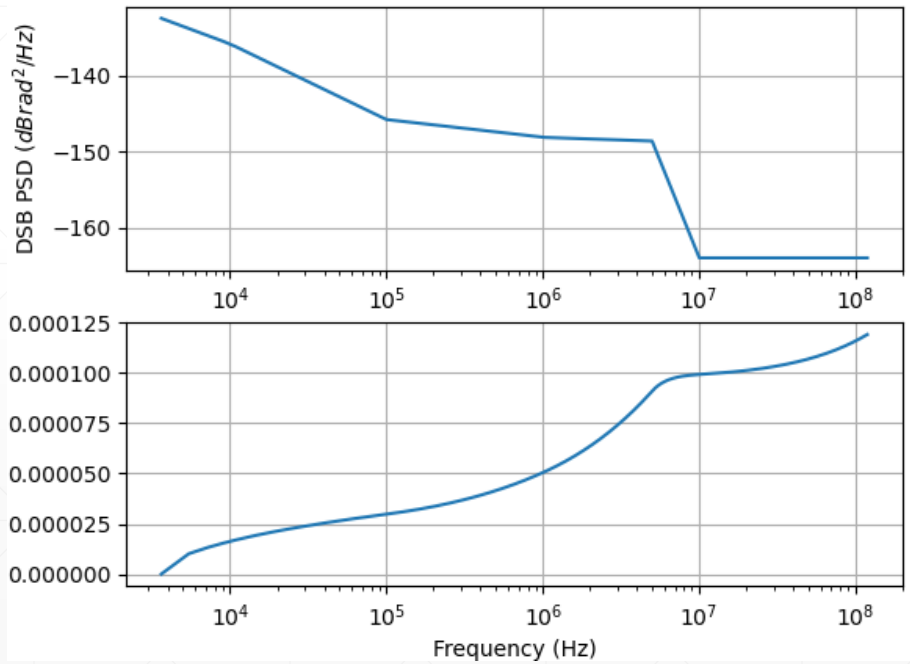
example_noise_time_series.py

Input PSD values:

Offset Frequency (Hz)	10	100	1e3	10e3	100e3	1e6	5e6	10e6
Noise PSD (dBrad^2/Hz)	-82.03	-107.07	-128.22	-135.86	-145.81	-148.15	-148.62	-164







Time series generation and PSD spectrum verification

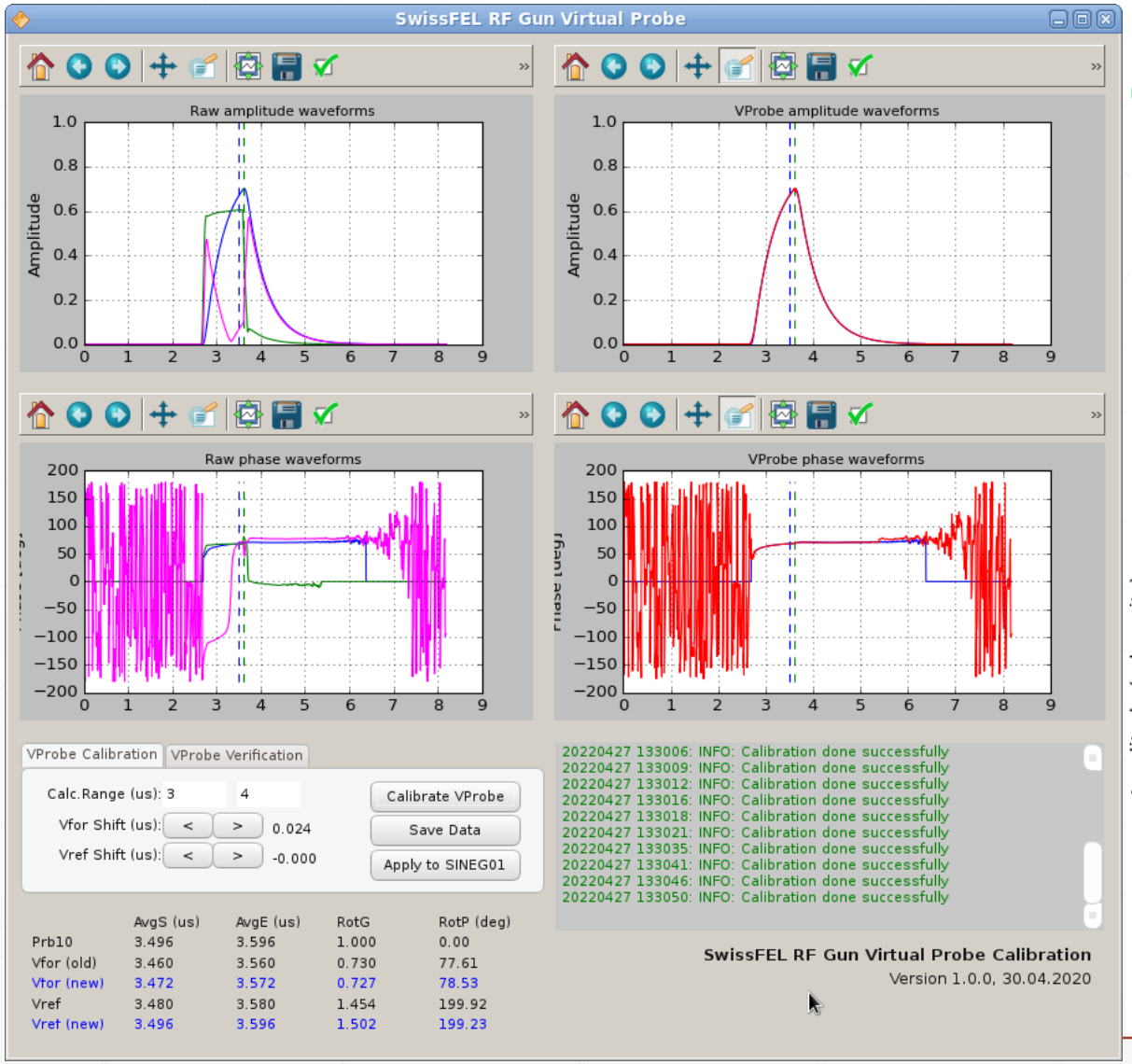
Noise PSD spectrum -> jitter estimation

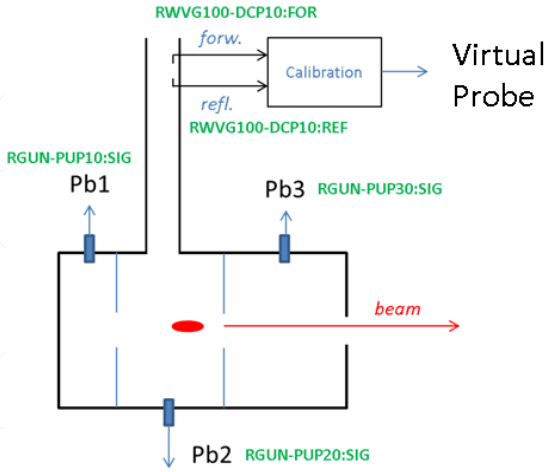
RF Calibration

Function	Example	Comments
calib_vprobe	example_calib_virtual_probe.py	Calibrate cavity virtual probe with the forward & reflected signals
calib_dac_offs	example_calib_dac_offs.py	Calibrate DAC offset for I/Q modulator used in direct upconversion
calib_iqmod	example_calib_iqm.py	Calibrate I/Q modulator imbalance for direct upconversion
calib_cav_for	example_calib_for_ref_nc_cavity.py	Calibrate cavity forward signal using forward and probe
calib_ncav_for_ref	example_calib_for_ref_nc_cavity.py	Calibrate cavity forward & reflected signals with constant QL and detuning
calib_scav_for_ref	example_calib_for_ref_sc_cavity.py	Calibrate cavity forward & reflected signals with time-varying QL and detuning
for_ref_volt2power	example_sim_cavity_basic.py	Calibrate forward & reflected power from forward & reflected voltage
phasing_energy	example_fit_funcs.py (demo fitting)	Calibrate energy gain and beam phase with phase scan and energy measurement
egain_cav_power	example_power_to_vacc.py	Estimate steady-state standing-wave cavity voltage from drive power
egain_cgstr_power	example_power_to_vacc.py	Estimate constant-gradient traveling-wave structure ACC voltage from drive power
calib_vsum_poor		Calibrate vector sum by rotating and scaling all signals referring to a selected channel
calib_sys_gain_pha	example_calib_sys_gain_phase.py	Calibrate system gain and system phase

RF Calibration (cont.)

example_calib_virtual_probe.py

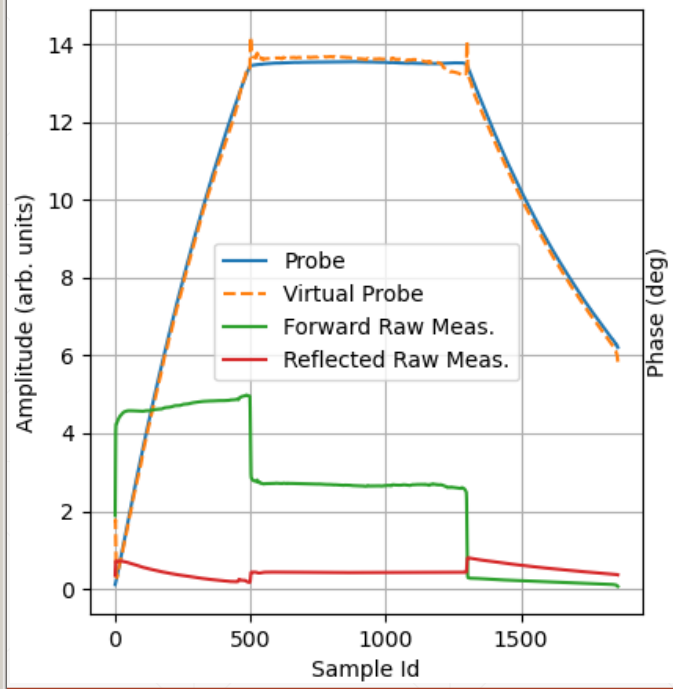


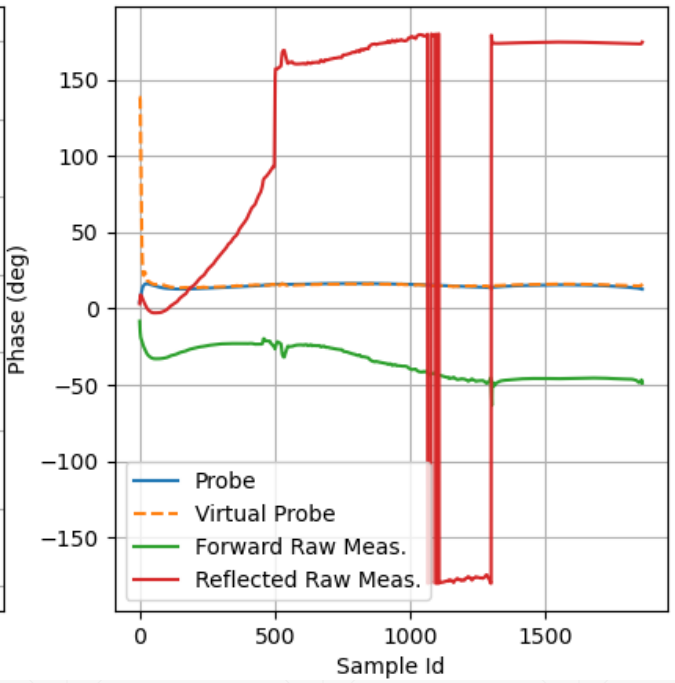


$$\left.\begin{aligned} \mathbf{v}_{for} &= \mathbf{a}\mathbf{v}_{for,mea} + \mathbf{b}\mathbf{v}_{ref,mea} \\ \mathbf{v}_{ref} &= \mathbf{c}\mathbf{v}_{for,mea} + \mathbf{d}\mathbf{v}_{ref,mea} \end{aligned}\right\} \Rightarrow$$

$$\mathbf{v}_{probe} = \mathbf{v}_{for} + \mathbf{v}_{ref} = \mathbf{m}\mathbf{v}_{for,mea} + \mathbf{n}\mathbf{v}_{ref,mea}$$

$$(\mathbf{m} = \mathbf{a} + \mathbf{c},\ \mathbf{n} = \mathbf{b} + \mathbf{d})$$





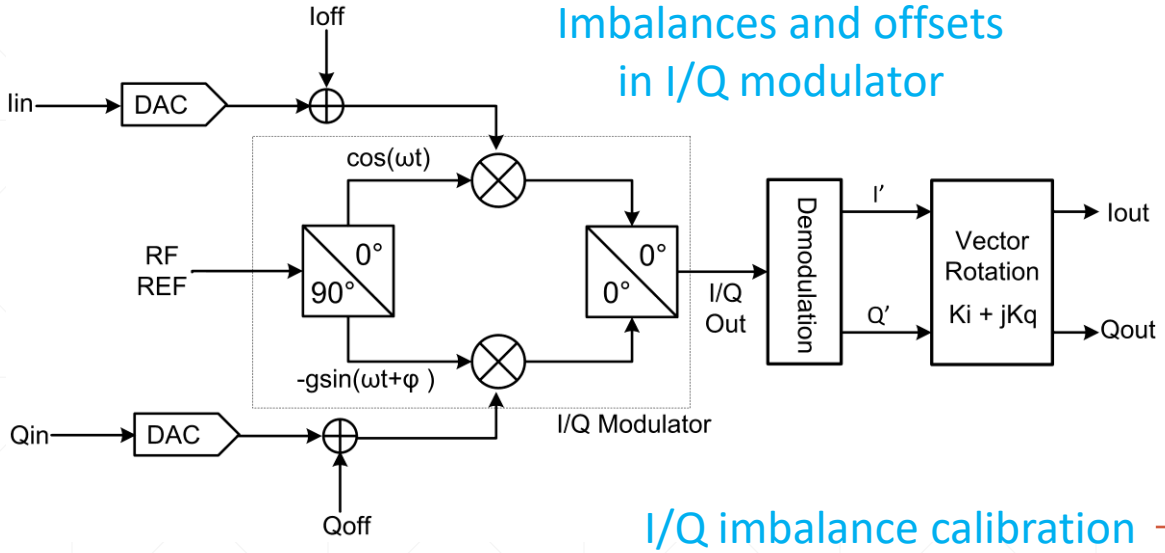
Virtual probe calibration of SwissFEL Gun

Virtual probe calibration of a TESLA cavity

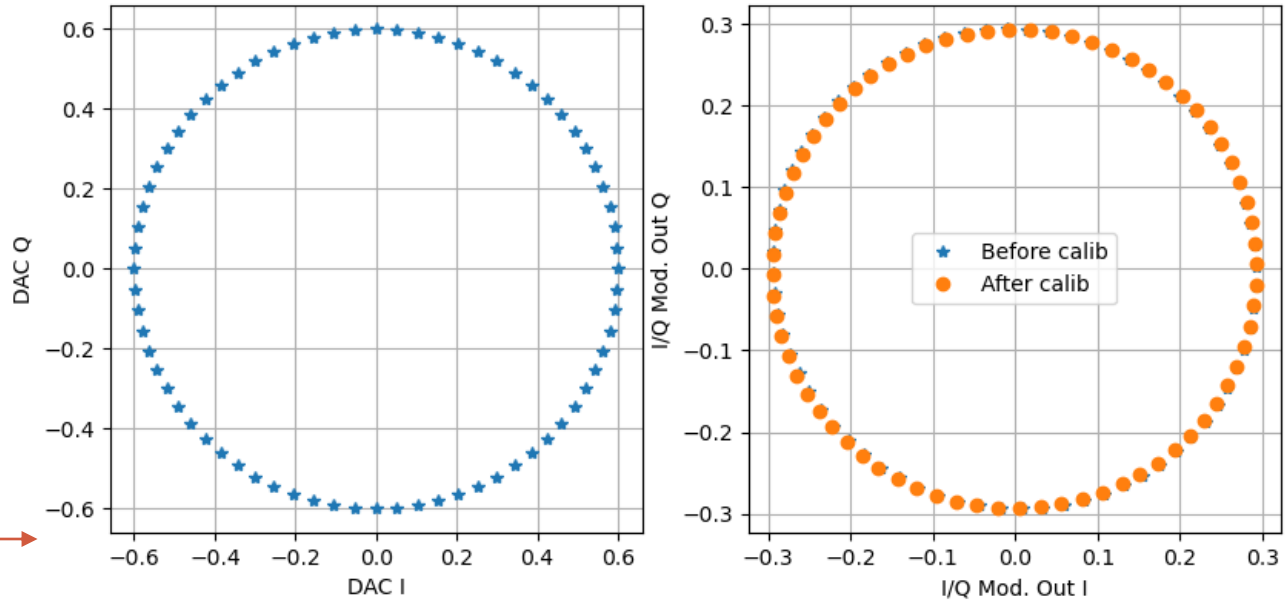
RF Calibration (cont.)

example_calib_dac_offs.py
example_calib_iqm.py

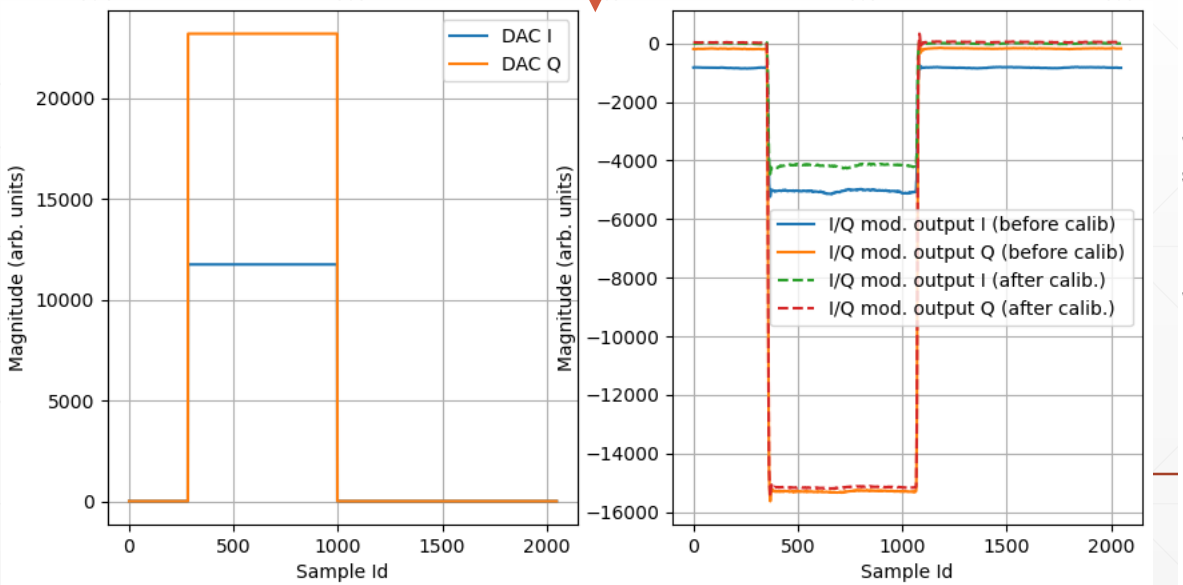
Imbalances and offsets in I/Q modulator

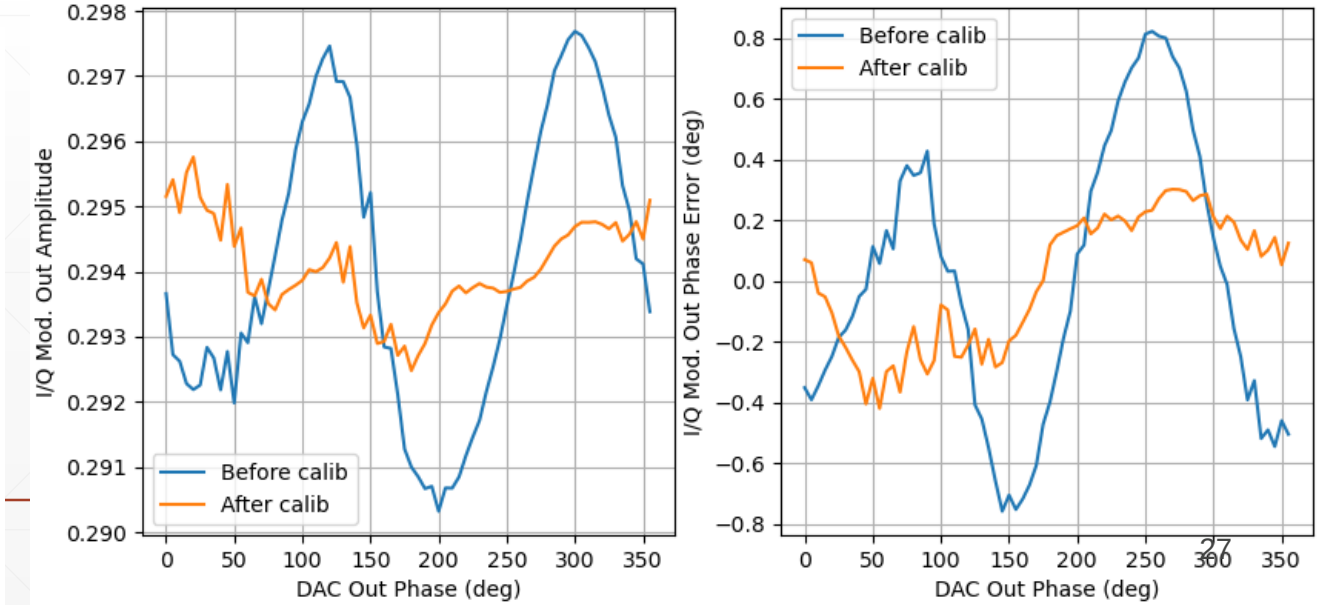


I/Q imbalance calibration



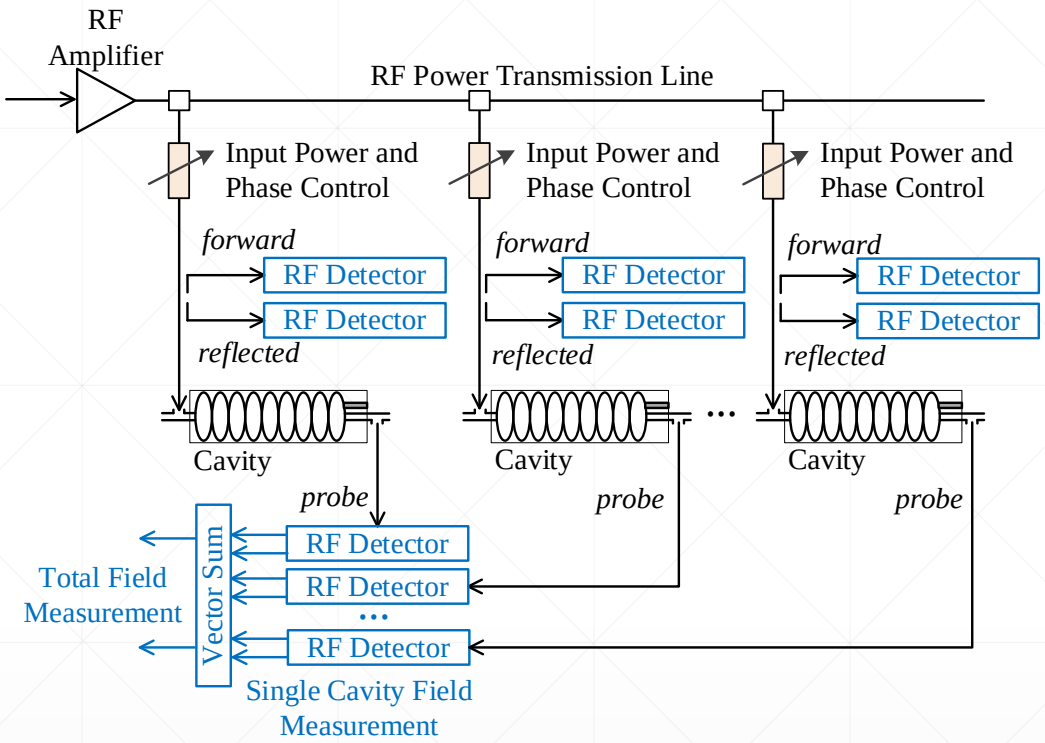
DAC offset calibration





RF Calibration (cont.)

example_calib_for_ref_nc_cavity.py

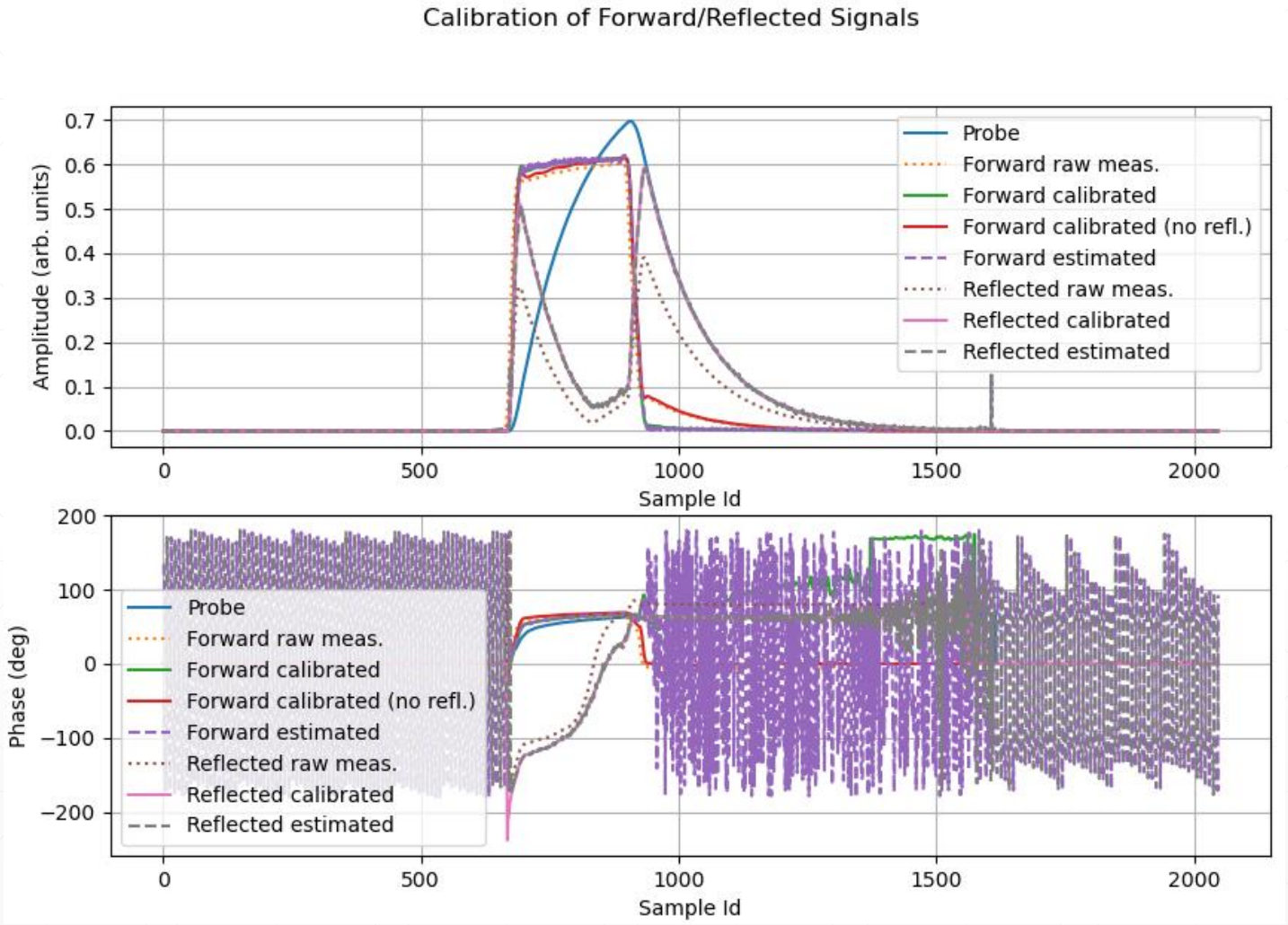


$$\mathbf{v}_{C,cal} = \mathbf{v}_{for,cal} + \mathbf{v}_{ref,cal},$$

$$\mathbf{v}_{for,cal} = \mathbf{a}\mathbf{v}_{for,mea} + \mathbf{b}\mathbf{v}_{ref,mea},$$

$$\mathbf{v}_{ref,cal} = \mathbf{c}\mathbf{v}_{for,mea} + \mathbf{d}\mathbf{v}_{ref,mea}.$$

Calibrate the directional coupler directivity and arbitrary measurement path

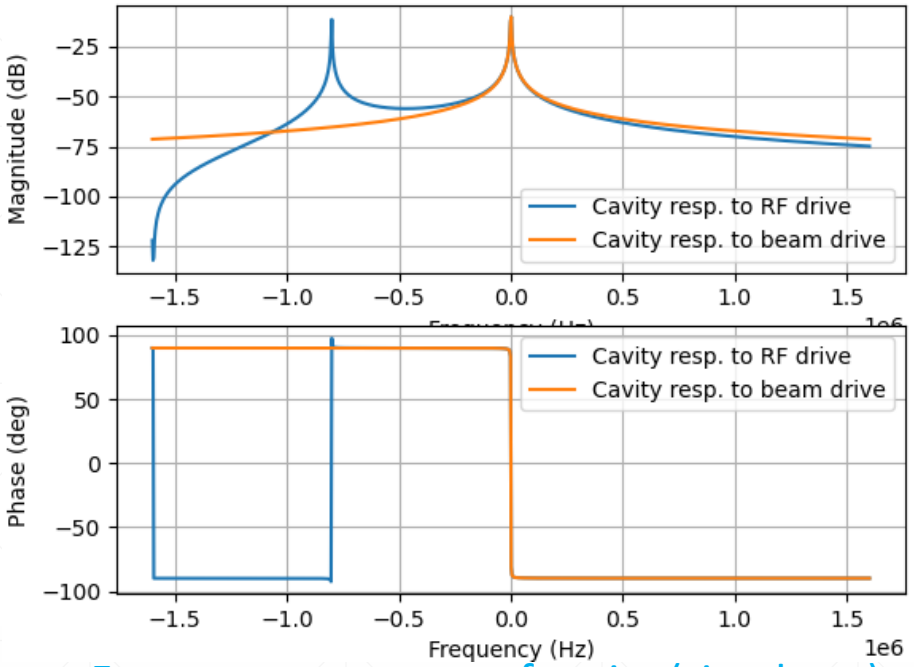


RF System Identification

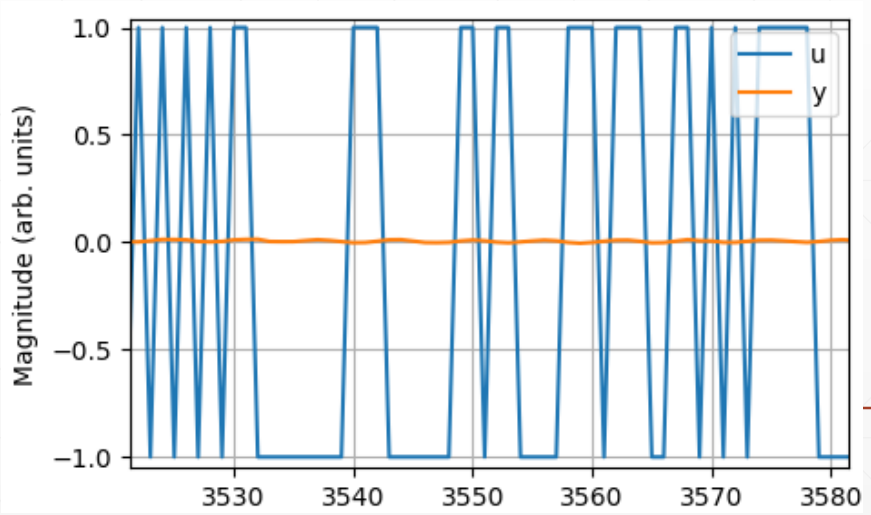
Function	Example	Comments
prbs	example_sysid_prbs_etfe.py	Produce the PRBS signal for system identification
etfe	example_sysid_prbs_etfe.py	Empirical Transfer Function Estimation (ETFE)
half_bw_decay	example_calib_for_ref_sc_cavity.py	Calculate the cavity half-bandwidth from the RF pulse decay stage
detuning_decay	example_calib_for_ref_sc_cavity.py	Calculate the cavity detuning from the RF pulse decay stage
cav_drv_est	example_calib_for_ref_nc_cavity.py	Estimate the cavity drive and reflected signals from probe signal
cav_par_pulse	example_calib_for_ref_sc_cavity.py example_sysid_caveq.py	Calculate the cavity parameters (half-bandwidth and detuning) within the pulse by directly solving the cavity equation
cav_par_pulse_obs	example_calib_for_ref_sc_cavity.py example_sysid_caveq.py	Calculate the cavity parameters (half-bandwidth and detuning) within the pulse using the ADRC observer
cav_beam_pulse_obs	example_sysid_caveq.py	Calculate the beam drive voltage within the pulse using the ADRC observer
cav_observer		Construct the ADRC observer and estimate the denoised cavity voltage and the general disturbances
iden_impulse	example_aff_ilc.py	Identify the impulse response of a real/complex SISO system from data
beta_powers		Identify the cavity input coupling factor using steady-state forward & reflected powers

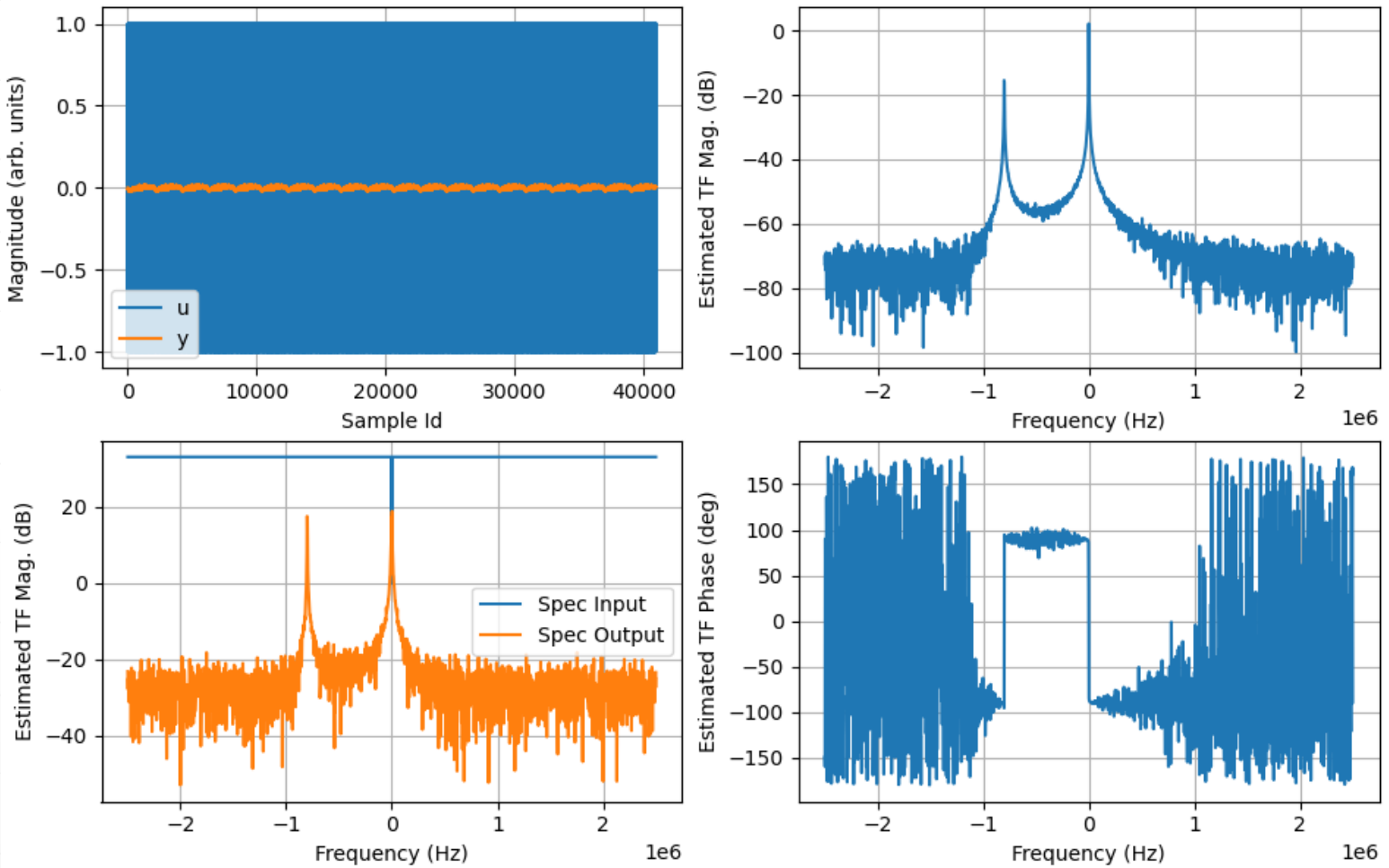
RF System Identification (cont.)

`example_sysid_prbs_etfe.py`



Frequency response of cavity (simulator)



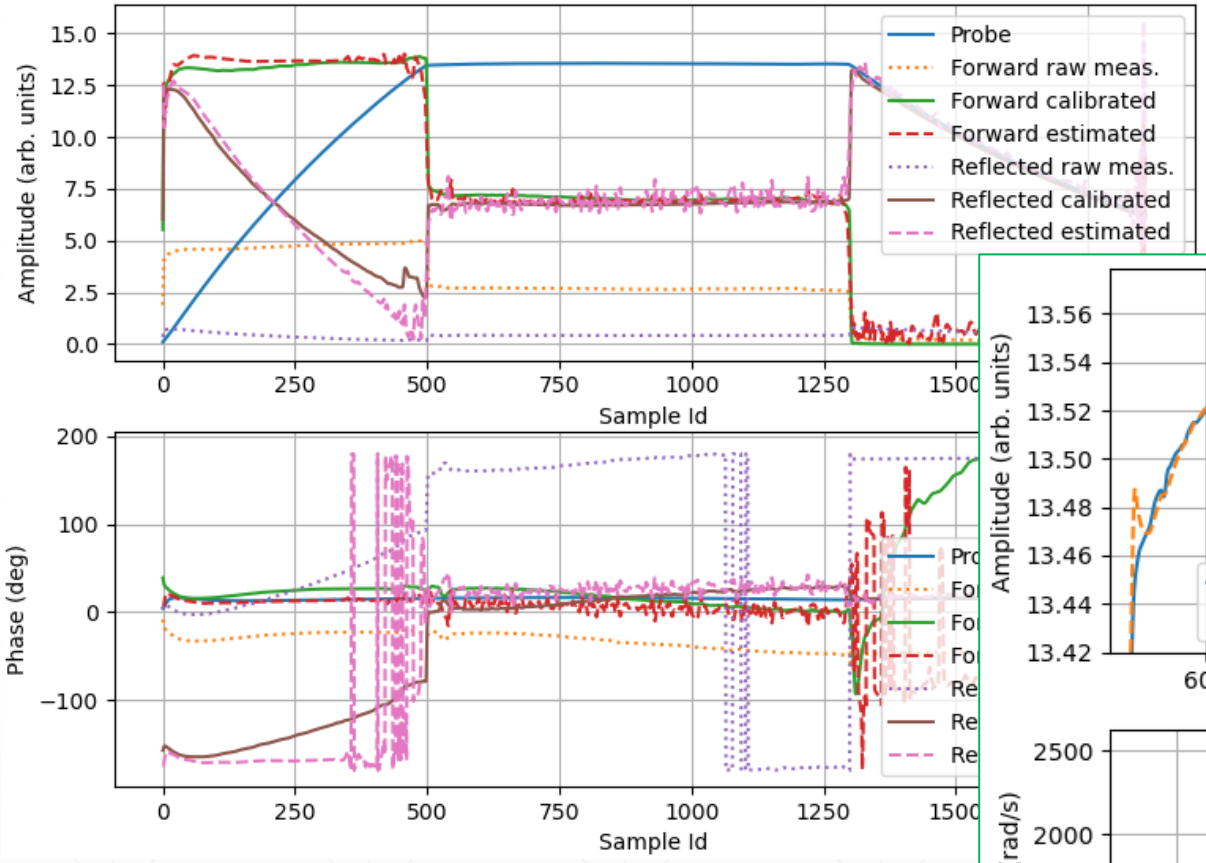


System identification with Empirical Transfer Function Estimation (ETFE) method

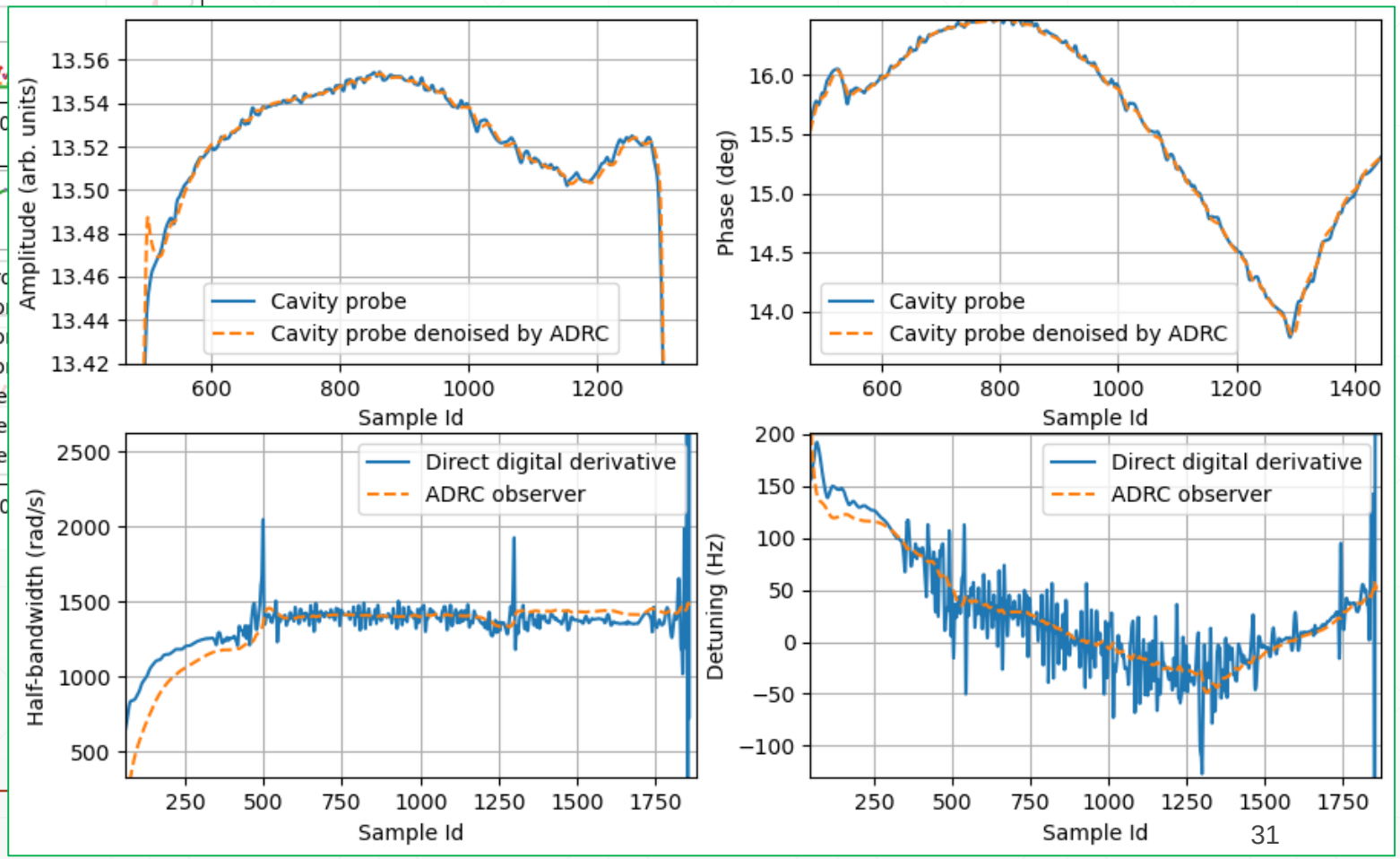
Pseudorandom Binary Sequence (PRBS) input and output

RF System Identification (cont.)

`example_calib_for_ref_sc_cavity.py`



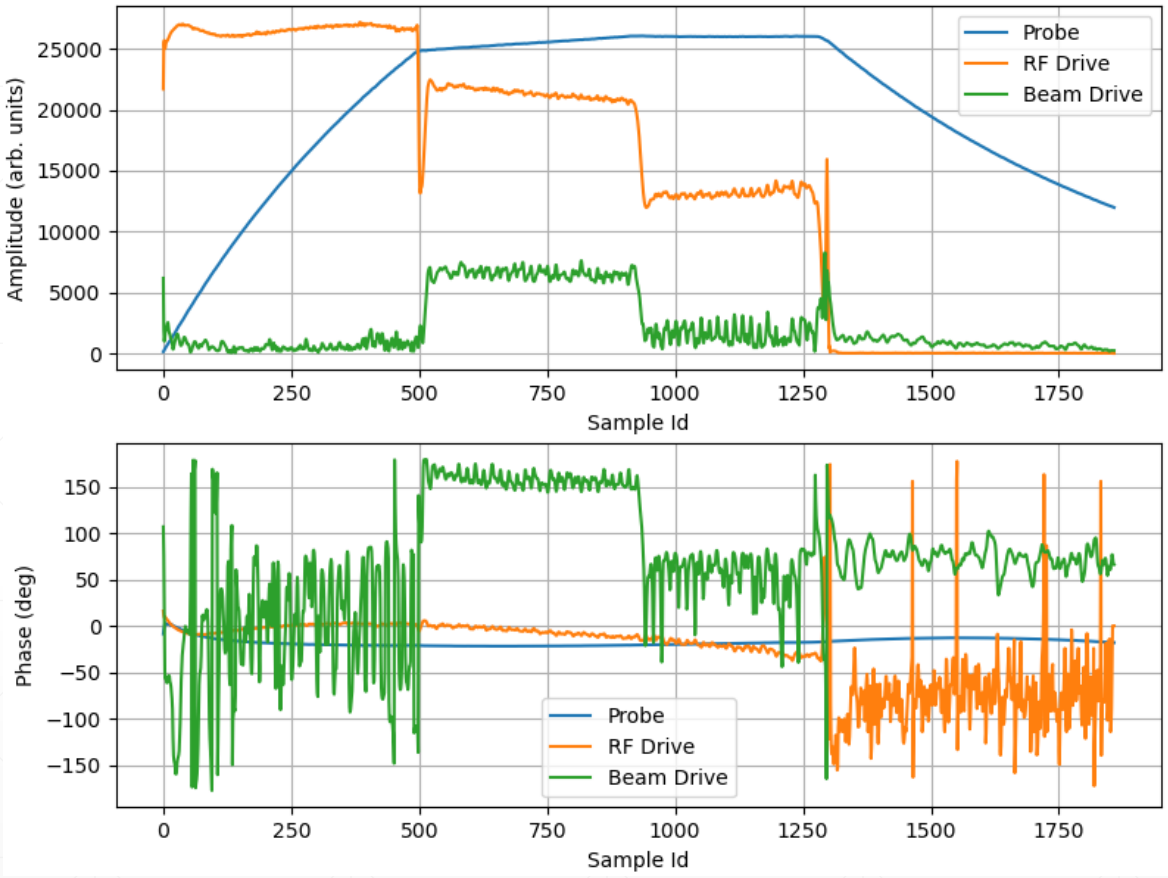
Calibration of forward and reflected signals to the same reference plane of probe signal

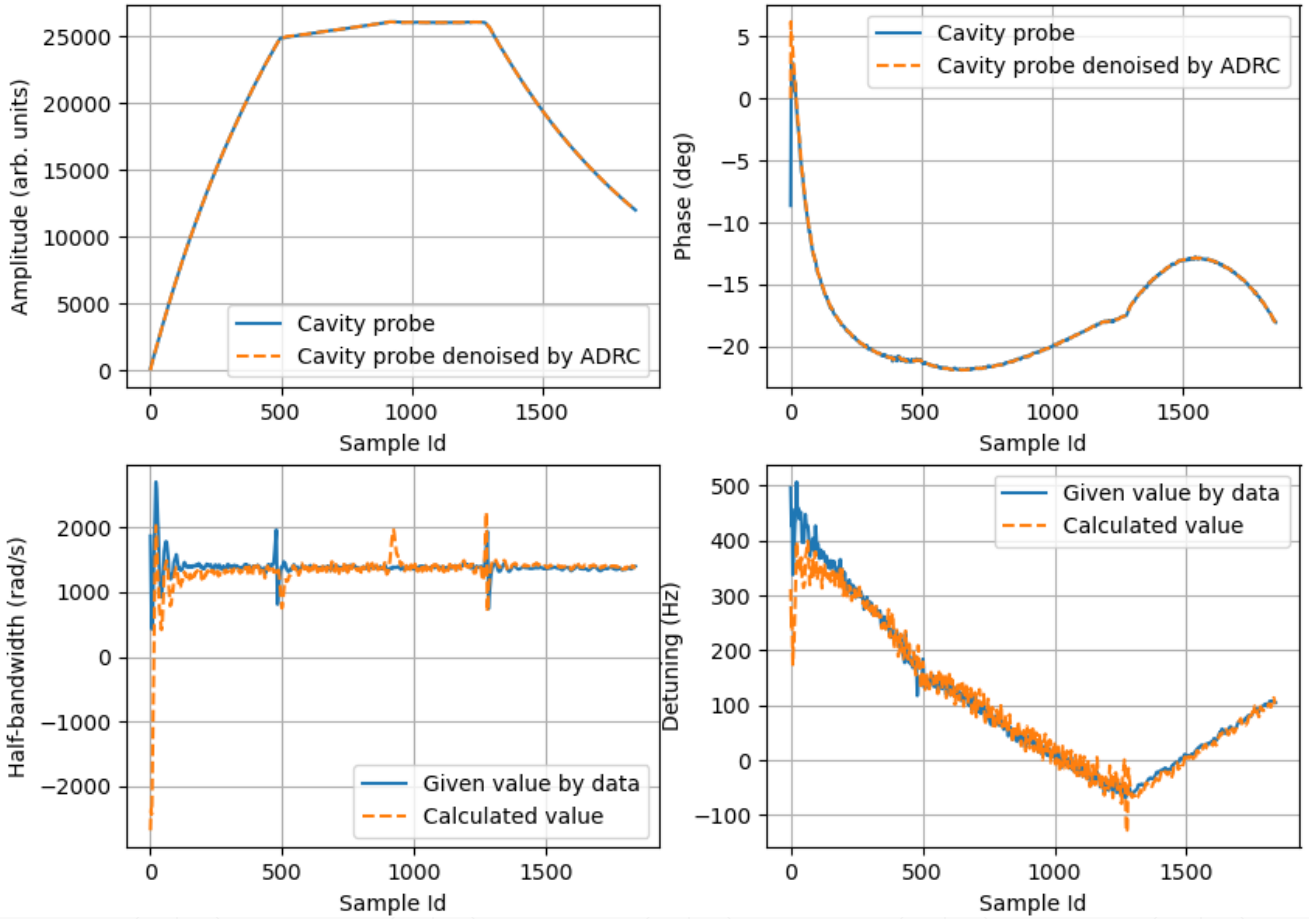


Smooth the cavity probe signal, and calculate the half-bandwidth and detuning of the cavity during the RF pulse

RF System Identification (cont.)

`example_sysid_caveq.py`





Cavity probe signal, RF drive signal and beam drive signal (already calibrated to the same reference plane as the probe)

Calculate the half-bandwidth and detuning during the pulse with beam on

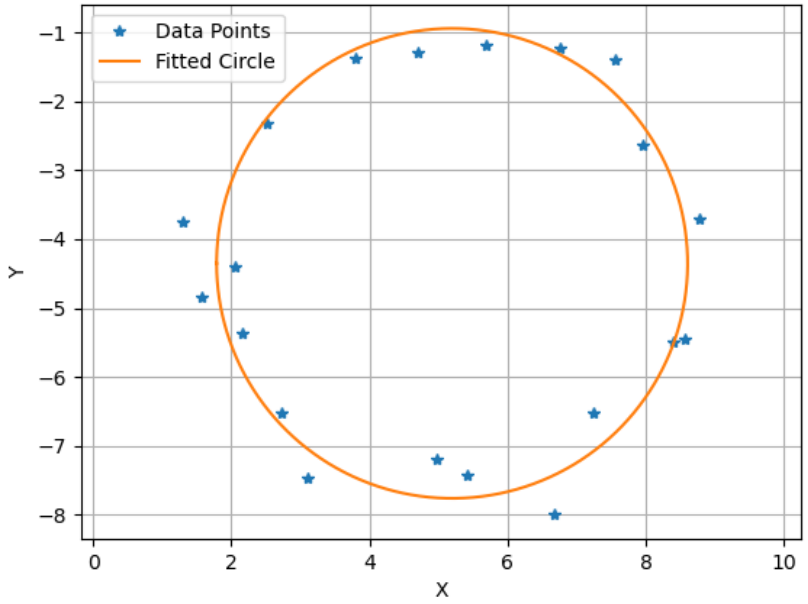
Other functions

Function	Example	Comments
fit_sincos	example_fit_funcs.py	Fit the sine or cosine function
fit_circle	example_fit_funcs.py	Fit the 2D points to a circle function
fit_ellipse	example_fit_funcs.py	Fit the 2D points to an ellipse function
fit_Gaussian	example_fit_funcs.py	Fit a 1D Gaussian function

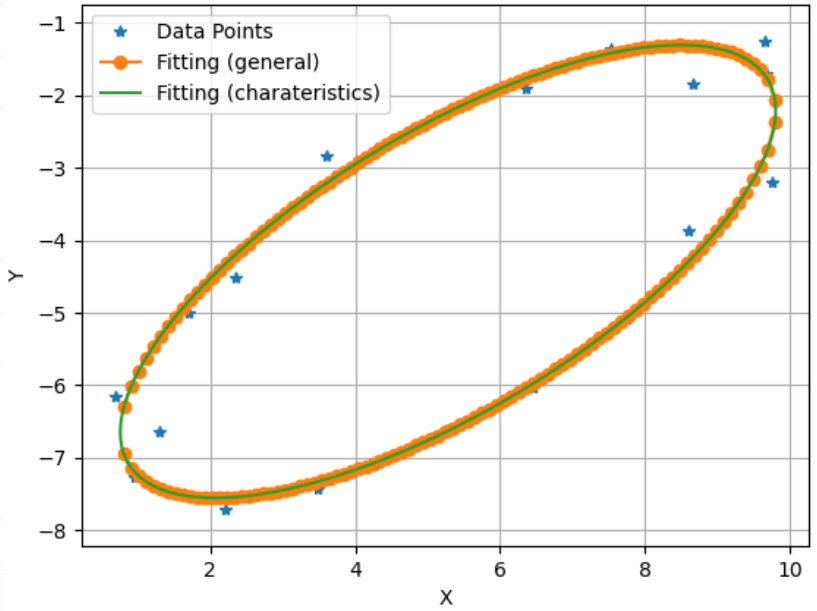
Function	Example	Comments
save_mat		Save a dictionary into a Matlab .mat file
load_mat		Load a Matlab .mat file into a dictionary
get_curtime_str		Get a string of the current date/time
get_bit		Get a bit of an integer
add_tf		Adding two transfer functions in numerator/denominator
plot_ellipse	example_fit_funcs.py	Plot an ellipse using its characteristics
plot_Guassian	example_fit_funcs.py	Plot a 1D Gaussian distribution

Other functions (cont.)

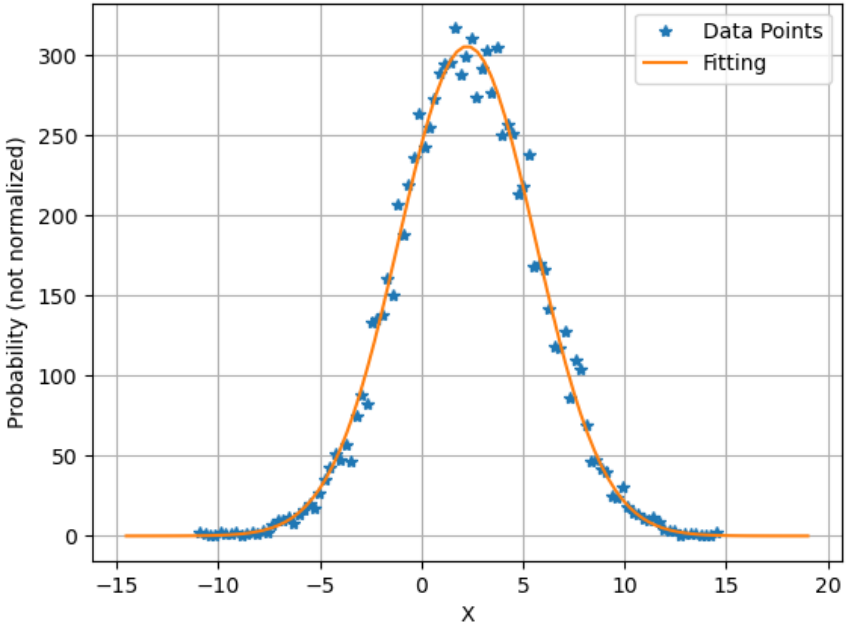
`example_fit_funcs.py`



Fit a circle



Fit an ellipse



Fit a Gaussian distribution

Status and Outlook

Status and Outlook

- Basic functions implemented (https://git.psi.ch/GFA/RF/Libraries/llrflibspy)
- Next steps:
  - Continue the development of library functions
  - Documentation
  - Open-source

Thank you for attention!