LASER INTERFEROMETER GRAVITATIONAL WAVE OBSERVATORY
- LIGO -
CALIFORNIA INSTITUTE OF TECHNOLOGY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Technical Note **LIGO-T2100238–v3** 2021/08/18

Red Pitaya Digital Laser Controller

O.Elgabori, F.Salces, A.Gupta, R.Adhikari

California Institute of Technology
LIGO Project, MS 18-34
Pasadena, CA 91125
Phone (626) 395-2129
Fax (626) 304-9834
E-mail: info@ligo.caltech.edu

Massachusetts Institute of Technology
LIGO Project, Room NW22-295
Cambridge, MA 02139
Phone (617) 253-4824
Fax (617) 253-7014
E-mail: info@ligo.mit.edu

LIGO Hanford Observatory
Route 10, Mile Marker 2
Richland, WA 99352
Phone (509) 372-8106
Fax (509) 372-8137
E-mail: info@ligo.caltech.edu

LIGO Livingston Observatory
19100 LIGO Lane
Livingston, LA 70754
Phone (225) 686-3100
Fax (225) 686-7189
E-mail: info@ligo.caltech.edu

http://www.ligo.caltech.edu/

Contents

1 Abstract

This report details using the Red Pitaya (125 MHz 14 bit) electronic board as a digital feedback controller for laser frequency stabilization. There is an exploration of various digital signal processing functionalities of a python interface to the onboard FPGA. Through a plant model approach, we attempt to validate the performance of Red Pitaya for feedback control. This is achieved by configuring the board to perform system identification and fitting frequency response data to a pole-residue model. The aim is to develop an automated device capable of determining the frequency response of some unknown plant and cancelling undesirable features (e.g. resonances) to produce a flat response.

2 Introduction

2.1 LIGO

The Laser Interferometer Gravitational-Wave Observatory (LIGO) detects and studies gravitational waves using laser interferometry. The detectors used by LIGO are advanced Michelson interferometers that are 4 km long with Fabry Perot cavities and power recycling mirrors that make them highly sensitive and capable of detecting length changes at 1/10,000th the width of a proton. One significant factor that impacts the sensitivity of these interferometers is a frequency stabilized laser. As such, it is highly desirable to have a feedback controller that makes adjustments to maintain a stable frequency response.

2.2 Problem

The type of laser utilized by LIGO is a non-planar ring oscillator (NPRO), which are known to have a high intrinsic stability (i.e. no external stabilization) on the order of $10^4\mathrm{Hz}/\sqrt{\mathrm{Hz}}$ at 1 Hz [?]. This quantity denoted the free-running frequency noise of the laser. The laser frequency is stabilized by means of a piezoelectric transducer (PZT) . However, the stabilization is impacted by the mechanical resonances of the PZT as they limit the control bandwidth. These mechanical resonances can be suppressed through the implementation of a digital filter.

There are two types of digital filters: infinite impulse response (IIR) and finite impulse response (FIR). Mathematically, these filters are defined by their impulse response (i.e. infinite or finite) and are represented by

$$y(n) = \sum_{k=0}^{\infty} h(k)x(n-k), \tag{1}$$

$$y(n) = \sum_{k=0}^{N-1} h(k)x(n-k), \tag{2}$$

where y(n) is the output and x(n-k) are the inputs with associated coefficients h(k). Each filter has its own advantages and disadvantages in practice. For instance, IIR filters consume less memory than FIR filters and have lower latency making them faster as well. However, FIR has linear phase characteristics and are more stable as their output values do not have feedback, and thus will not become unstable for any input signal unlike IIR filters. [?]

2.3 FPGA

A field-programmable gate array (FPGA) is an integrated circuit with programmable interconnects that can be customized for a particular application. As these interconnects can be reprogrammed to be used for other purposes other than the originally desired application, it makes this device quite versatile when used in the field. The Red Pitaya (Figure **??**) is an electronic board that possesses a FPGA along with other useful components, such as digital to analog (DAC), analog to digital (ADC) converters, and digital filters that make it well suited for control purposes. The Red Pitaya is preferable to use over other electronic boards with an FPGA as it is low cost, has a user friendly interface, and possesses extensive documentation. In addition, other FPGA-based systems are not as readily usable as the Red Pitaya as this particular board has an open source python package called PyRPL.

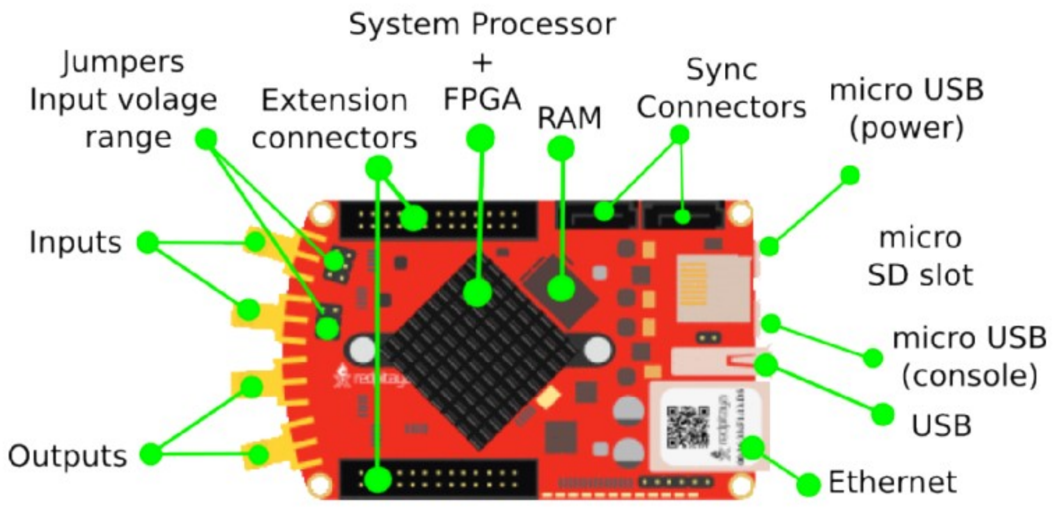

Figure 1: Schematic of the Red Pitaya board[?]

3 Purpose

We aim to improve the stabilization of the laser by suppressing the mechanical resonances of the piezoelectric transducer by the Red Pitaya FPGA [?]. We will accomplish this by programming the FPGA to determine the optimal digital filter — IIR or FIR — to use for suppressing the resonant modes of the piezo. The feedback system is represented by a simple block diagram in figure **??**.

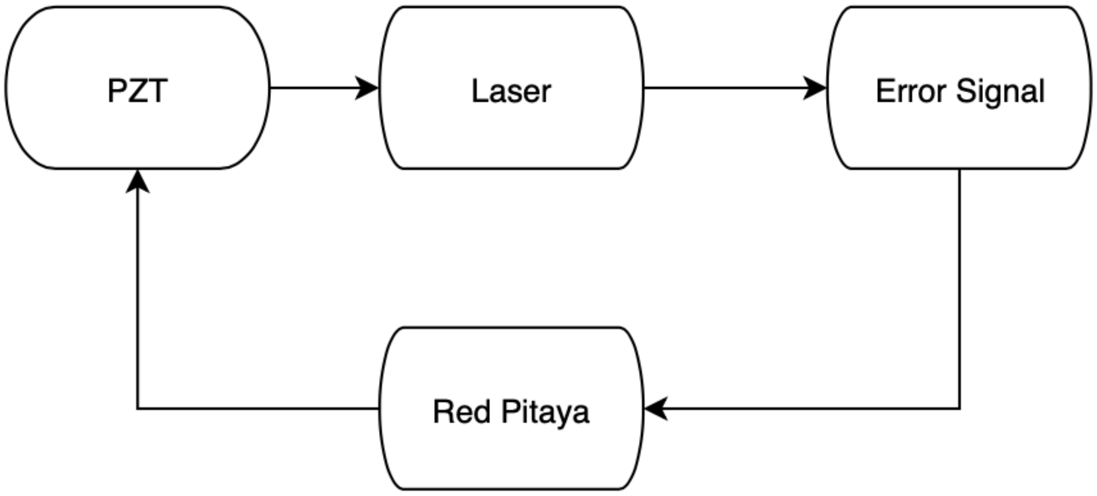

Figure 2: A closed-loop system of the NPRO laser PZT and Red Pitaya controller. The frequency of the laser is compared to a stable reference frequency to produce an error signal that is fed to the controller. The controller then corrects for this error by feeding into the PZT actuator that is used to control the laser frequency.

4 Approach

As FPGA programming can be time consuming, we will use the customizable DSP modules provided by PyRPL. This software package not only provides many instruments, including high order digital filters, it comes with a graphical user interface and has a python API. Following the API manual on PyPRL, we will program the interface for the necessary modules. However, for our purposes it is necessary to expand the modules provided as PyPRL lacks an FIR filter and only has a module for the IIR filter.[?]

5 Timeline

5.1 Weeks 1/2

First, we will familiarize ourselves with the documentation on the Red Pitaya (125 MHz 14 bit) and PyRPL. Then, begin exploring the various functions and capabilities of the FPGA using a test setup before working on the actual NPRO laser.

5.2 Weeks 3/4

After exploring the Red Pitaya, we will set it up near the optical table and connect it to the NPRO laser PZT input. We will then validate the performance of the Red Pitaya using a plant model approach.

5.3 Weeks 5/6

We will begin developing the digital filters that will be programmed on the Red Pitaya and continue validating its performance.

5.4 Weeks 7/8

After implementing the digital filters, the Red Pitaya will be programmed to determine which is the most optimal filter to use for suppressing the resonances of the PZT based on relevant parameters.

5.5 Weeks 9/10

The final two weeks we will perform a final assessment of the Red Pitaya to determine if it could be utilized in other subsystems of LIGO. Then, the remainder of the time left will be spent writing the final paper and preparing for the symposium.

6 Interim Report 1

6.1 PyRPL

Despite its outdated documentation, the software PyRPL provides convenient modules and a user friendly interface to the Red Pitaya. The network analyzer module is of particular interest as it allows us to probe an unknown system's transfer function (the ratio of the output signal of a system to a given input signal). The network analyzer accomplishes this probe through IQ (In-phase/Quadrature) modulation and demodulation. This involves exciting the device under test with a frequency sweep of sine functions, demodulating the output with the same sine and a corresponding cosine, then low-pass filtering and extracting the phase and magnitude.[?] A bode plot of the transfer function for a 1.9 MHz low pass filter (Mini-Circuits Model BLP-1.9) provided by the network analyzer module is shown in figure **??** and **??**. There does appear to be issues with this module, however, as the phase plot does not agree with the one produced by a bode analyzer (a separate application not implemented by PyRPL) using the same Red Pitaya (Figure **??**).

6.2 System Identification and Modeling

Extracting the transfer function of an unknown system using the network analyzer module enables us to perform system identification. Based on this information, we can develop a model (analytical or numerical) and construct digital filters by taking the model's inverse. This can be achieved using the zpk (zero-pole-gain) representation of transfer functions.

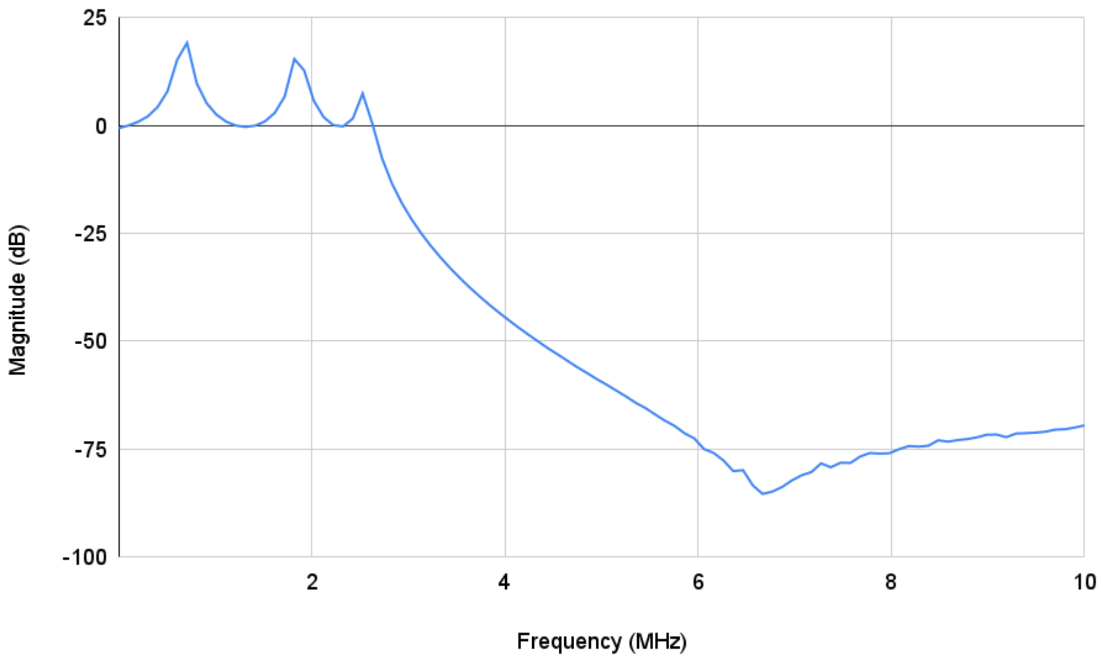

Figure 3: The magnitude of the transfer function for a 1.9 MHz low pass filter produced by the network analyzer.

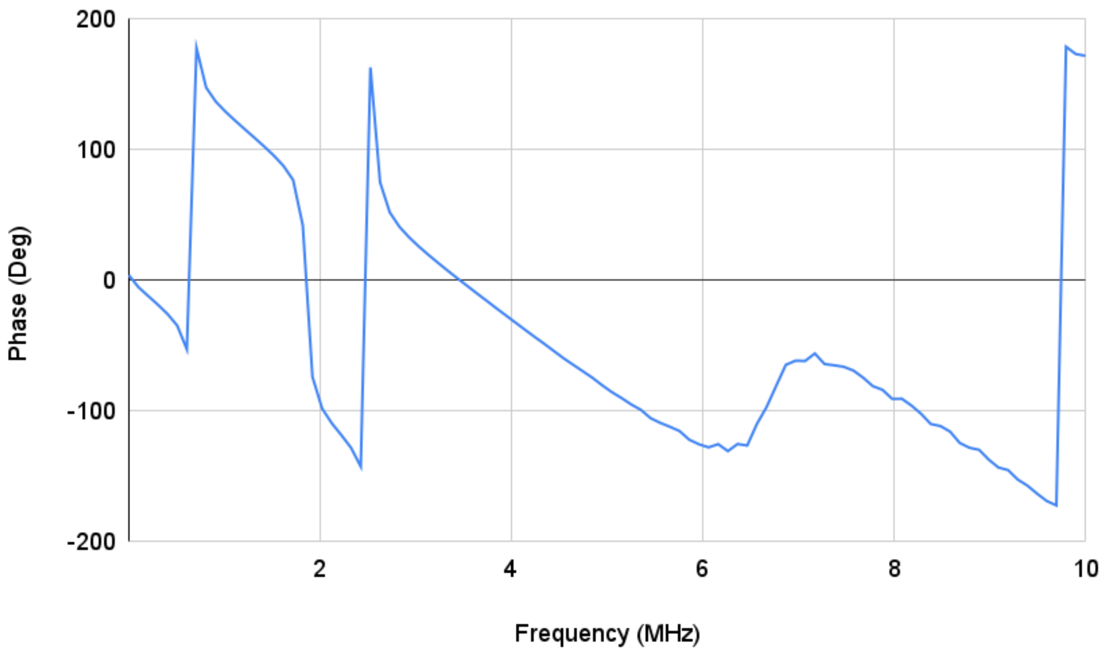

Figure 4: The phase of the transfer function for a 1.9 MHz low pass filter produced by the network analyzer.

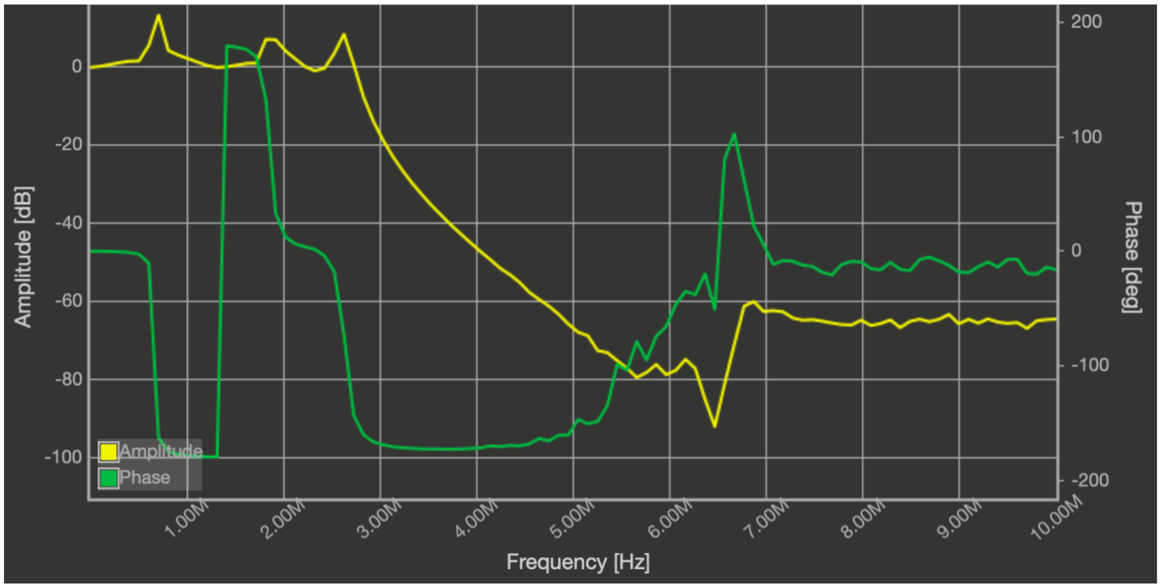

Figure 5: The transfer function of a 1.9 MHz low pass filter produced by the bode analyzer of the Red Pitaya web application.

6.3 Next Steps and Expected Challenges

I will conduct a closer examination of the network analyzer module to determine what is the source of this issue with the phase plot by testing other simple filters and parsing through the python code for PyRPL. Then, I expect to use these filters as test systems for implementing digital filters on the Red Pitaya.

7 Interim Report 2

7.1 Transfer Functions

In the Laplace s-domain, a transfer function is the ratio of system's output to the given input

$$H(s) = \frac{B(s)}{A(s)} \tag{3}$$

$$H(s) = \frac{\sum_{m=0}^{M} b_m s^m}{\sum_{n=0}^{N} a_n s^n} \tag{4}$$

where N must be greater than M for physically stable systems.

Typically, the most useful and preferred representation for a transfer function is the ZPK form,

$$H(s) = K\frac{\prod_{m=0}^{M}(s - z_m)}{\prod_{n=0}^{N}(s - p_n)} \tag{5}$$

where z_m are the zeroes of the transfer function(roots of the numerator polynomial), p_m are the poles of the transfer function (roots of the denominator polynomial), and K is an associated gain term that comes from factoring out the coefficients of the highest power term in each polynomial.

7.2 Capabilities of PyRPL

Using PyRPL, it is possible to digitally simulate transfer functions, such as a bandpass filter, on one Red Pitaya and extract this transfer function through the network analyzer on a separate Red Pitaya board. This simulation can be achieved through one of the available IQ modules that is provided or, for a more complicated transfer function, using the IIR filter.Furthermore, in our examination of the network analyzer module we probe the transfer functions of several different low pass filters. These filters serve as test systems in our attempt to validate the performance of PyRPL and the Red Pitaya to achieve system identification and feedback control. In order to obtain an estimation of the transfer function for these analog devices, the vector fitting algorithm is used to provide a fit to the measured data.

7.3 Vector Fitting

Vector fitting is a method developed by Bjørn Gustavsen to fit measured frequency response data given some set of an initial poles.[?] This method attempts to solve for the parameters of the partial fractions model

$$H(s) = \sum_{m=1}^{N} \frac{r_m}{s - p_m} + d + sh \tag{6}$$

where r_m are the residues, p_m are the poles, and d and h are just some coefficients of the polynomial term of the model. Using some initial set of poles, we write

$$\sigma(s)H(s) = x(s) \tag{7}$$

$$(\sum_{m=1}^{N} \frac{\tilde{r}_m}{s - q_m} + 1)H(s) = \sum_{m=1}^{N} \frac{r_m}{s - q_m} + d + sh \tag{8}$$

where $\sigma(s)$ and $x(s)$ share the same initial poles q_m. Then, we solve an eigenvalue problem to obtain a new set of poles, and iterate the process until $\sigma(s)$ converges to 1 and $q_m = p_m$.

Using an implementation of the vector fitting method in python, I obtain fits for the transfer functions of the various low pass filters shown in figures 7-9. However, the code requires the user to provide an initial guess to the number of poles. In order to automate this process and obtain the best fit, I developed a python function that loops through 1-24 poles as the initial guess for the method and uses the sum of the residuals squared as the criterion for determining the best fit. The upper limit of 24 poles is a practical constraint as this highest order that PyRPL can currently provide.

7.4 Subsequent Goals

For the remainder of this project, I plan setting up the Red Pitaya to simultaneously perform system identification using the network analyzer and serve as a filter using the IIR module. Then, if time permits, I will optimize these processes to make the Red Pitaya efficient in system control.

References

[1] B. Willke , S. Brozek, K. Danzmann, V. Quetschke, and S. Gossler *Frequency stabilization of a monolithic Nd:YAG ring laser by controlling the power of the laser-diode pump source.* Optics Letters 25 (2000)

[2] *Difference between IIR and FIR filters: a practical design guide* (2020) https://www.advsolned.com/difference-between-iir-and-fir-filters-a-practical-design-

[3] M. Okada, T. Serikawa, J. Dannatt, M. Kobayashi, A. Sakaguchi, I. Petersen, and A. Furusawa *Extending the piezoelectric transducer bandwidth of an optical interferometer by suppressing resonance using a high dimensional IIR filter implemented on an FPGA.* Review of Scientific Instruments 91, 055102 (2020).

[4] *Red Pitaya Developer's Guide: Red Pitaya boards comparison* https://redpitaya.readthedocs.io/en/latest/developerGuide/125-10/vs.html

[5] L. Neuhaus, S. Deléglise *Basics of the PyRPL Architecture* https://pyrpl.readthedocs.io/en/latest/basics.html

[6] L. Neuhaus, R. Metzdorff, S. Chua, T. Jacqmin, T. Briant, A. Heidmann, P.-F. Cohadon, and S. Deléglise *PyRPL (Python Red Pitaya Lockbox) — An open-source software package for FPGA-controlled quantum optics experiments.* European Quantum Electronics Conference (2017)

[7] L. Neuhaus, S. Deléglise. *Network analyzer* https://pyrpl.readthedocs.io/en/latest/api.html#network-analyzer

[8] B. Gustavsen and A. Semlyen *Rational approximation of frequency domain responses by vector fitting* IEEE Trans. Power Delivery, vol. 14, no. 3, pp. 1052-1061, July 1999.

8 Appendix: Graphs

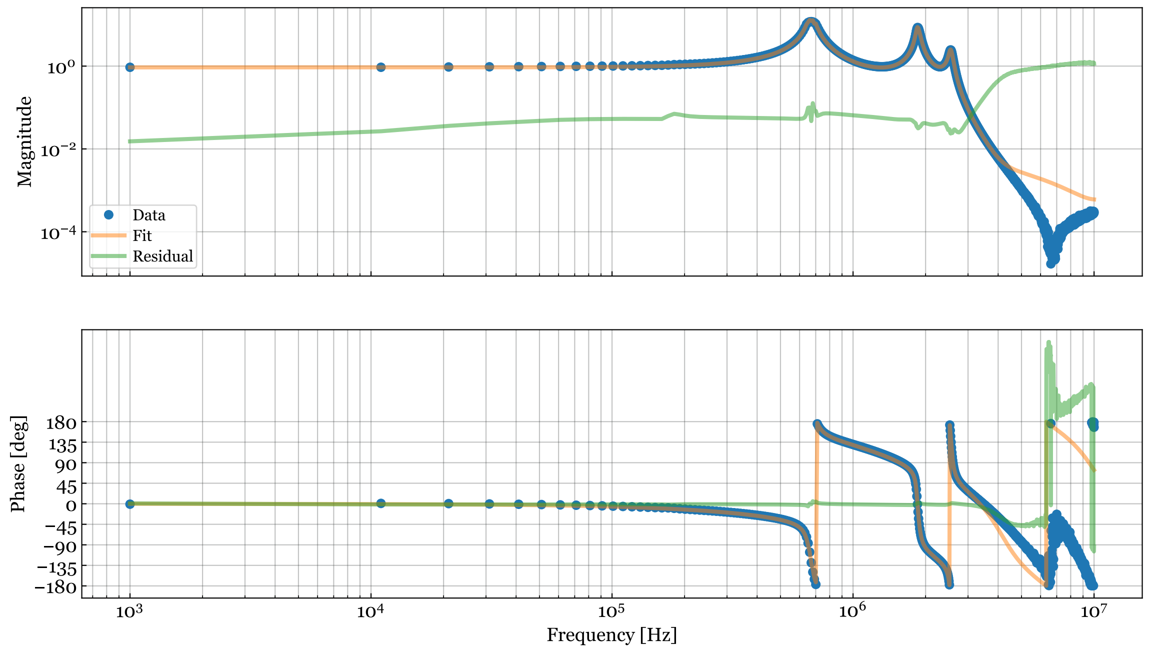

Figure 6: The best fit of the frequency response for a Minicircuits 1.9 MHz low pass filter. The fit is consistent with the data up until approximately 4 MHz and the number of poles used to produce it is 12. The time it took to determine the best fit is 1.26 s.

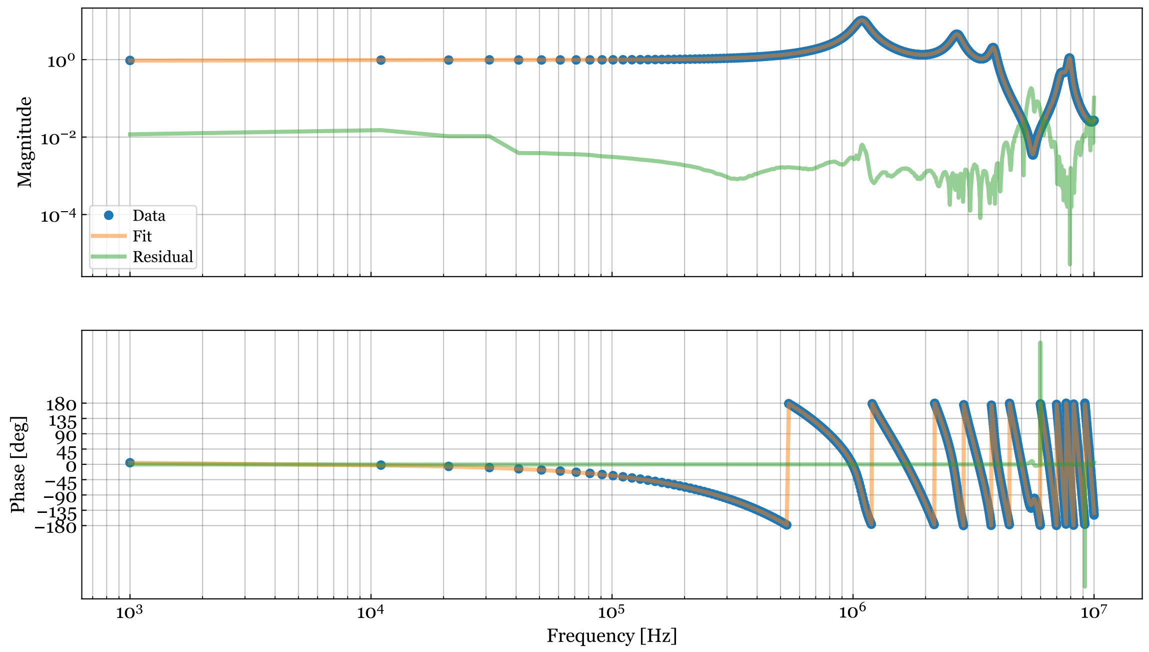

Figure 7: The best fit of the frequency response for a 10 MHz low pass filter. The number of poles used to produce it is 24 and the time it took to determine the best fit is 1.44 s.

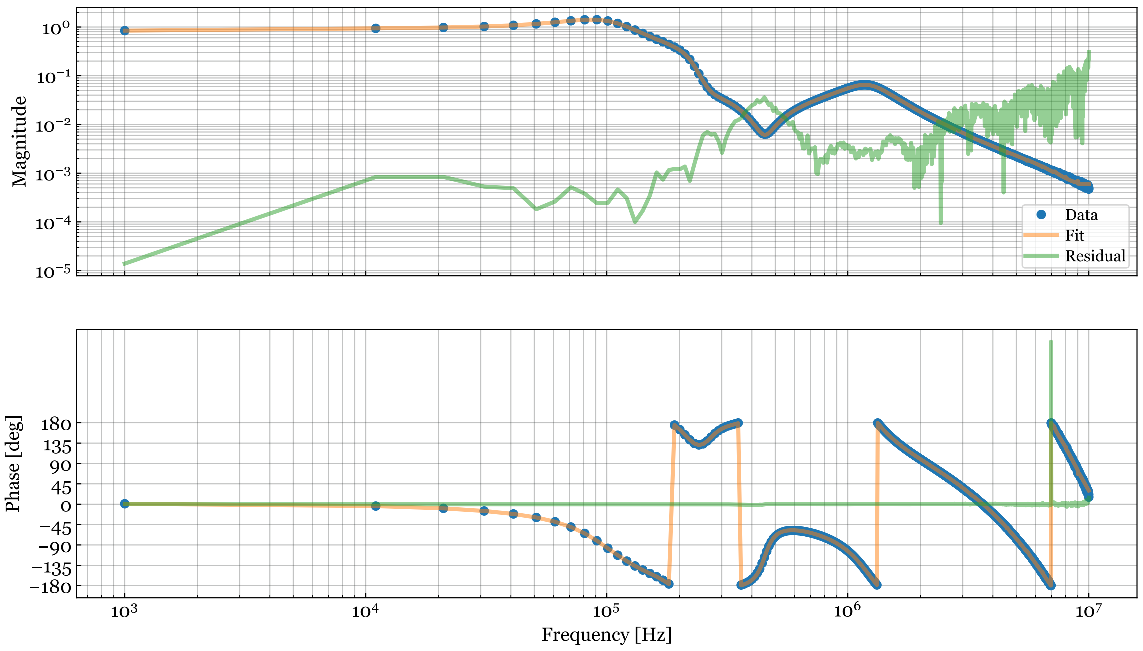

Figure 8: The best fit of the frequency response for a 5^{th} order elliptic low pass filter. The number of poles used to produce it is 16 and the time it took to determine the best fit is 1.32 s.