Using the Finesse-Virgo package

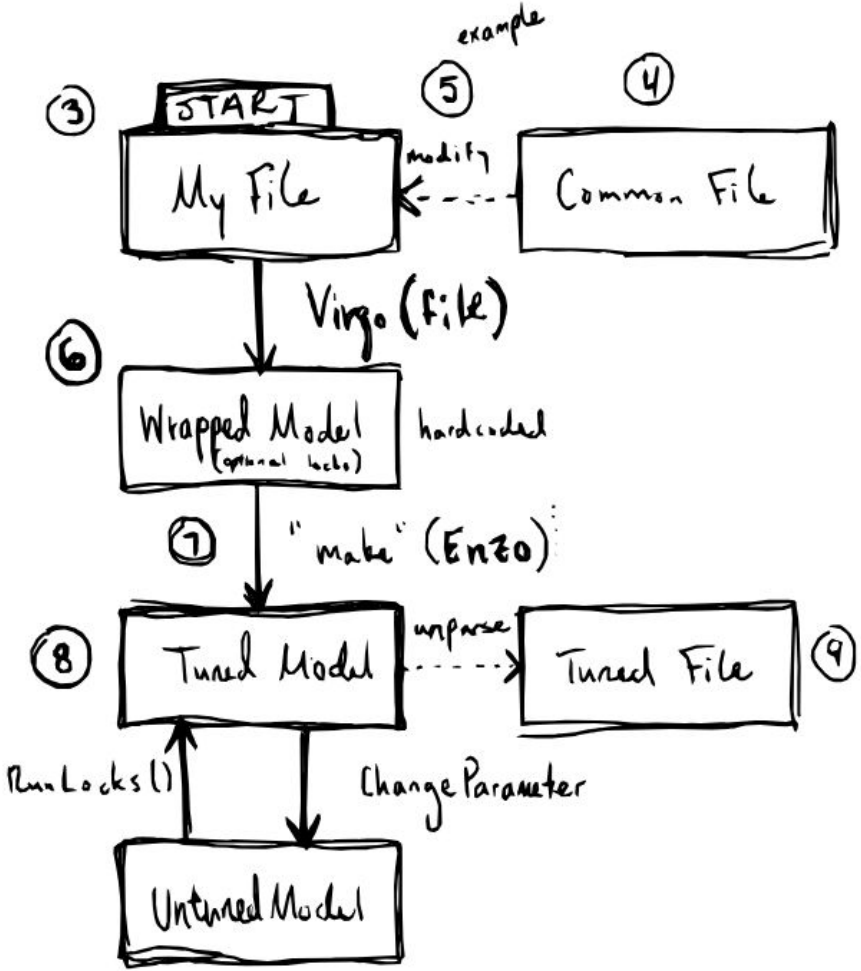
Jonathan Perry, On behalf of the Finesse-Nikhef team

Introduction

The `finesse-virgo` package includes top-level tools and models for simulating Virgo in Finesse 3.

- Common file with current configuration
- Pre-tuning scripts (including locks)
- Diagnostic and informative utility functions

Introduction

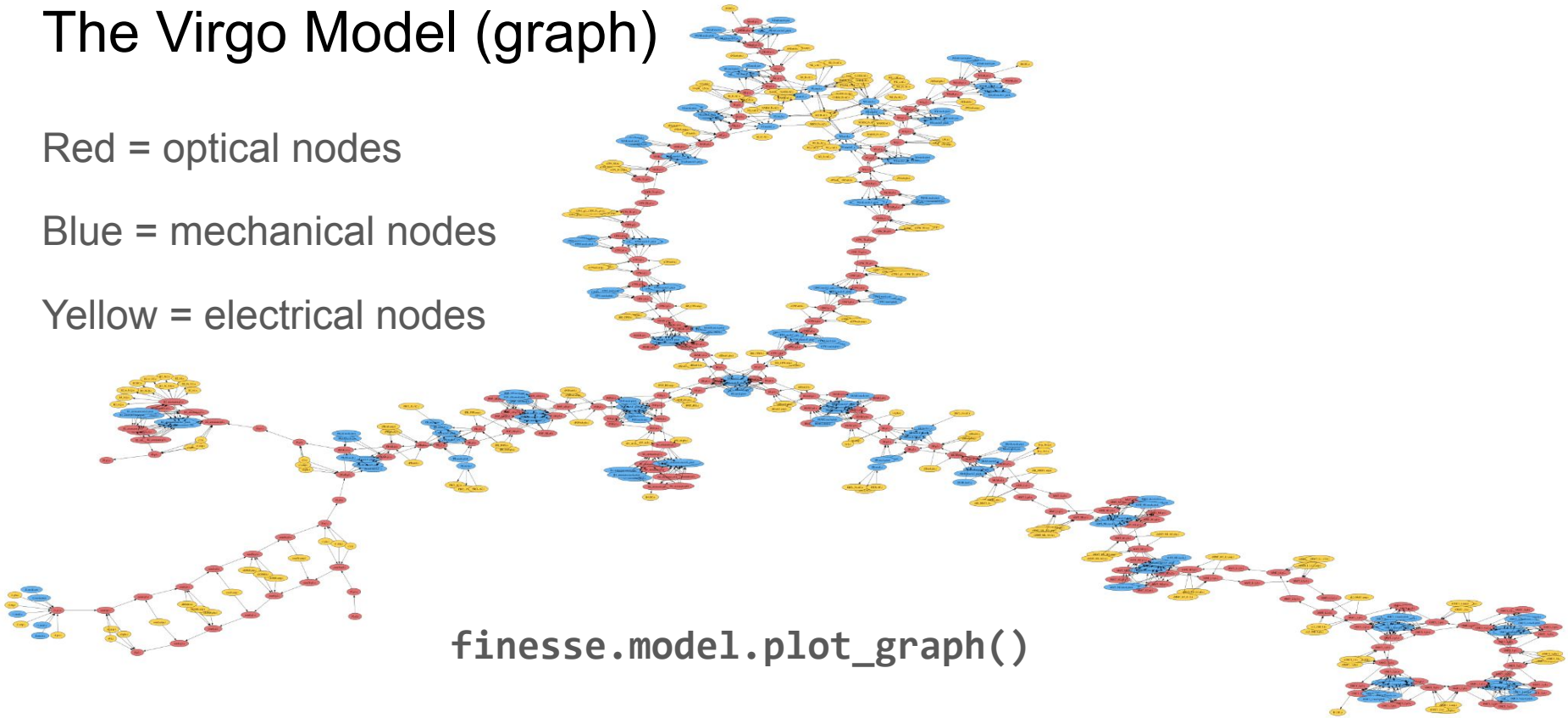


The Virgo Model (graph)

Red = optical nodes

Blue = mechanical nodes

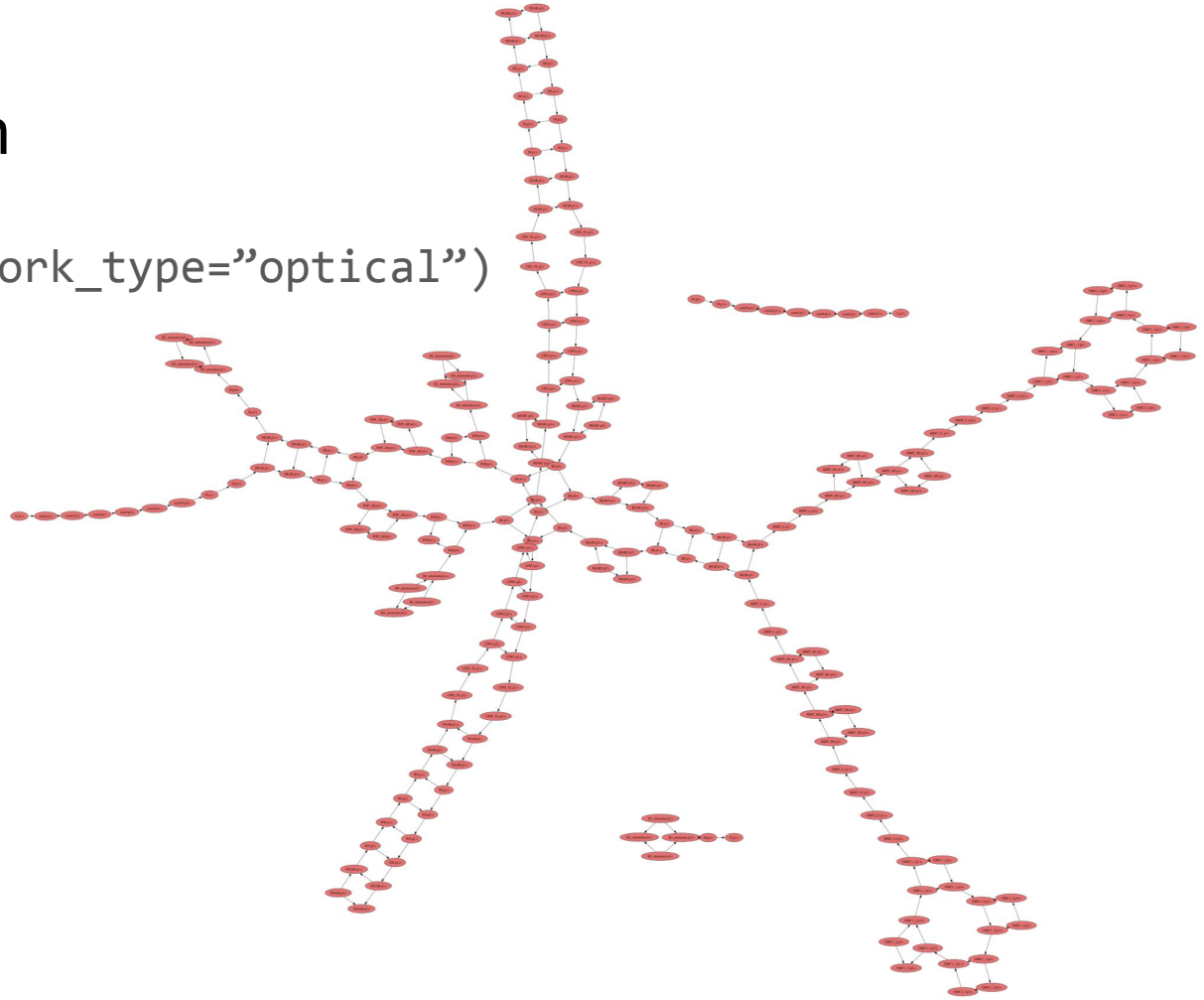
Yellow = electrical nodes



`finesse.model.plot_graph()`

The Optical Graph

```
model.plot_graph(network_type=”optical”)
```



Install / Update

Installation can be done with `pip`:

```
pip install finesse_virgo
```

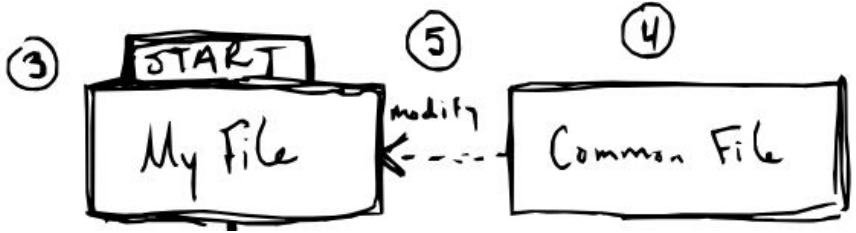
Update to latest version:

```
pip install –upgrade finesse_virgo
```

Import via `finesse`:

```
import finesse.virgo as fv
```

My file (the starting point)



The starting point of every study is a katfile.

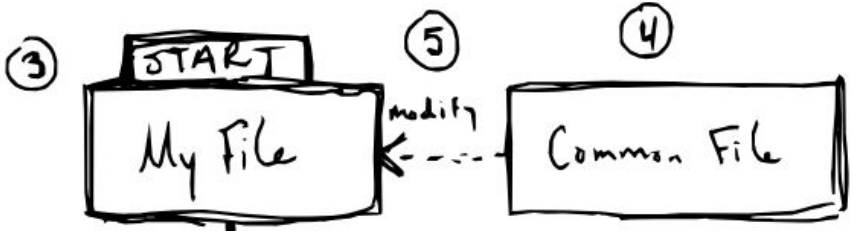
`fv.Virgo("my_file.kat")`

`fv.Virgo("my_file_dir")`

Not currently good for design tasks due to rigid constraints on naming conventions. (optics, DOFs, detectors, etc.) **To be relaxed.**

```
#-----------------------------------------------------------------------
# An Advanced Virgo Plus input file for Finesse 3

# modulation frequencies [TDR, tab 2.3, pp 24]
var f6 6270777   # 6 MHz,  fmod1 in TDR
var f8 4/3*f6   # 8 MHz,  4 / 3 * f6, fmod3 in TDR
var f56 9*f6     # 56 MHz, 9 * f6, fmod2 in TDR

var nsilica 1.44963
var Mloss 30u

var etalonNI 0  # NI etalon tuning
var etalonWI 0  # WI etalon tuning

# Laser and modulators
#-----------------------------------------------------------------------
# EOM parameters from https://logbook.virgo-gw.eu/virgo/?r=34898
# and https://logbook.virgo-gw.eu/virgo/?r=38123
# and https://logbook.virgo-gw.eu/virgo/?r=41551
laser i1 P=40.0
s s0 i1.p1 eom6.p1 L=1m
mod eom6 f=f6 midx=0.22
s sEOM6 eom6.p2 eom8.p1 L=0.1
mod eom8 f=f8 midx=0.15
s sEOM8 eom8.p2 eom56.p1 L=0.1
mod eom56 f=f56 midx=0.25

# REFL, B2 readout
#-----------------------------------------------------------------------
s s1 eom56.p2 FI.p1 L=0.2
# Lossless faraday isolator to pick off B2-beam
dbs FI
s s2 FI.p3 PRAR.p1 L=0

# Beam splitter to obtain 18 mW of power on B2 (with 40W input)
s s3 FI.p4 B2_attenuator.p1 L=0
bs B2_attenuator R=0.99816 T=0.00184
# B2 readout is at B2_attenuator.p3
```

The common file



Start with the “common file” if you don’t have your own:

```
fv.copy_input_files(“local_dir”)

virgo = fv.Virgo(“local_dir”)
```

Includes common optical layout file and **additional katscript** for tuning and controls.

```
# DOFs
##################################################################
# position
dof DARM NE.dofs.z -1 WE.dofs.z +1
dof CARM NE.dofs.z +1 WE.dofs.z +1
dof MICH NI.dofs.z -1 NE.dofs.z -1 WI.dofs.z +1 WE.dofs.z +1
dof PRCL PR.dofs.z +1
dof SRCL SR.dofs.z -1

dof NARM NI.dofs.z +1 NE.dofs.z +1
dof WARM WI.dofs.z +1 WE.dofs.z +1

dof NI_z NI.dofs.z +1
dof NE_z NE.dofs.z +1
dof WI_z WI.dofs.z +1
dof WE_z WE.dofs.z +1

# applied force
dof DARM_Fz NE.dofs.F_z -1 WE.dofs.F_z +1
dof CARM_Fz NE.dofs.F_z +1 WE.dofs.F_z +1
dof MICH_Fz NI.dofs.F_z -1 NE.dofs.F_z -1 WI.dofs.F_z +1 WE.dofs.F
dof PRCL_Fz PR.dofs.F_z +1
dof SRCL_Fz SR.dofs.F_z -1

dof NARM_Fz NI.dofs.F_z +1 NE.dofs.F_z +1
dof WARM_Fz WI.dofs.F_z +1 WE.dofs.F_z +1

dof NI_Fz NI.dofs.F_z +1
dof NE_Fz NE.dofs.F_z +1
dof WI_Fz WI.dofs.F_z +1
dof WE_Fz WE.dofs.F_z +1
```

Example: Modifications to common file

Before parsing:

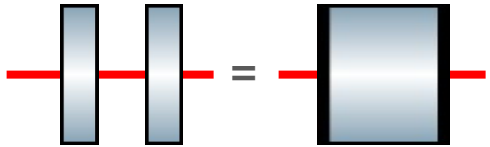
```
mod eomX f=1 midx=0
```

1. Additional modulator
2. Beamsplitter after SRM

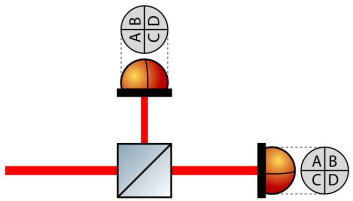
```
# SR pick-off
bs B1p_BS R=0.5 T=0.5
s B1p_BS_s SRAR.p2 B1p_BS.p1
```

After parsing:

1. Define degrees of freedom (for composite mirrors)
2. Create QPD RF readouts (demodulators)



```
dof SR_x SR.dofs.yaw 1 SRAR.dofs.yaw 1
```




```
s B1p_to_QPD_BS B1p_BS.p2 B1p_QPD_BS.p1
nothing B1p_QPD_nf_nada
nothing B1p_QPD_ff_nada
s B1p_QPD_nf B1p_QPD_BS.p2 B1p_QPD_nf_nada.p1 user_gouy_x=0 user_gouy_y=0
s B1p_QPD_ff B1p_QPD_BS.p3 B1p_QPD_ff_nada.p1 user_gouy_x=90 user_gouy_y=90
readout_rf B1p_f50_nf B1p_QPD_nf_nada.p1.i f=f50 output_detectors=true pdtype=xsplit
readout_rf B1p_f50_ff B1p_QPD_ff_nada.p1.i f=f50 output_detectors=true pdtype=xsplit
```

Pre-tuning / “Making” Virgo

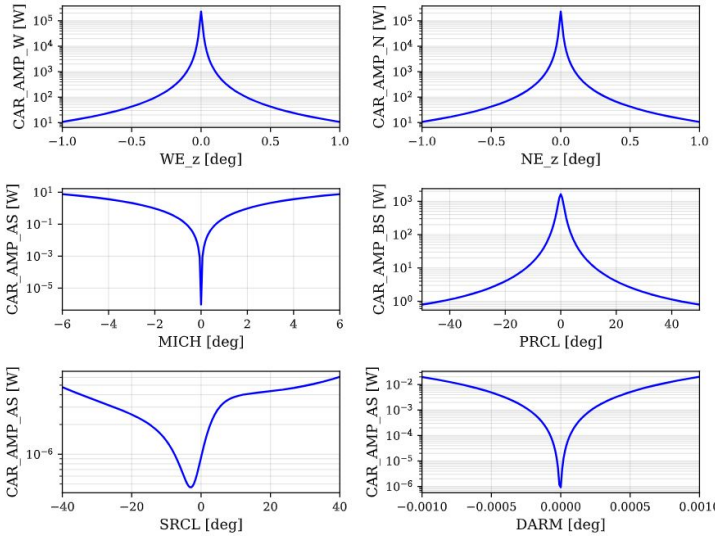
Note: Do not use `make()`, brief overview (in detail with Enzo)

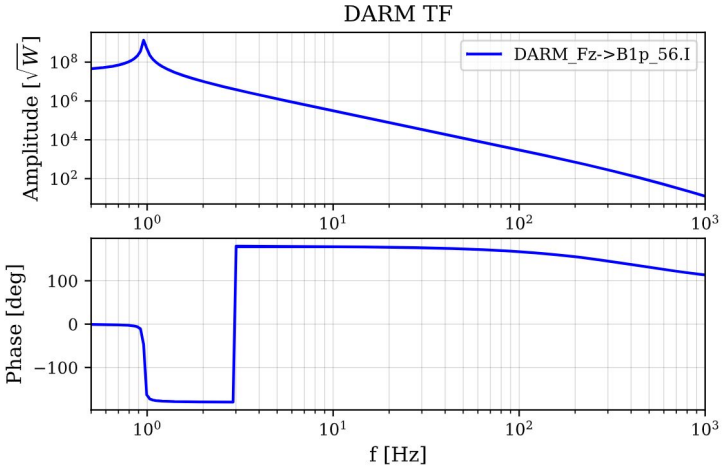
- Defines the “operating point”
- Optimize lengths
- Optimize powers (maximize arms, minimize dark port)
- Optimize demodulation phase
- Optimize lock gains
- Run the locks (zero the error signals)

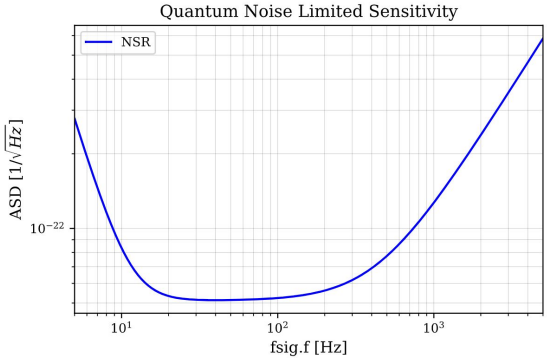
```
lock PRCL_lock B2_8_I PRCL.DC -63.80813077245811 5.3e-06
lock MICH_lock B2_56_Q MICH.DC 1995.7759287359338 9.1e-07
lock CARM_lock B2_6_I CARM.DC -0.1614654491019635 1.6e-05
lock DARM_rf_lock B1p_56_I DARM.DC -0.0065708585507211455 0.0017
lock DARM_dc_lock B1_DC DARM.DC -0.0065708585507211455 0.0017  enabled=false offset=4m
lock SRCL_lock B2_56_I SRCL.DC -3797.910605193875 2.2e-05     # West
```

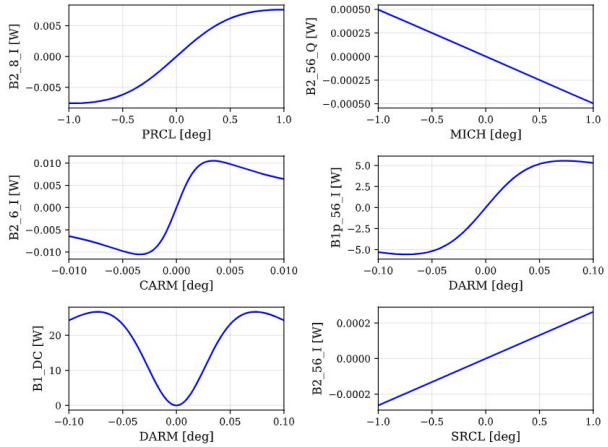
Tuned Model

- Plot powers **`virgo.plot_powers()`**
- Plot error signals **`virgo.plot_error_signal`**
- DARM transfer function **`virgo.plot_DARM()`**
- QNLS **`virgo.plot_QNLS()`**





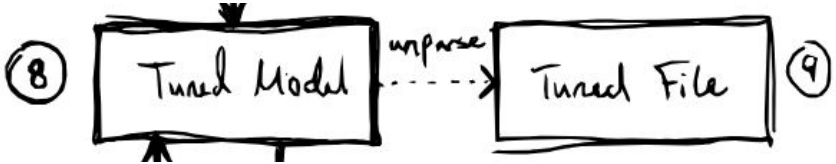




Save tuned model

Save a tuned model for use later using the unparser:

`virgo.model.unparse_file("my_tuned_file.kat")`




```
#------------------------------------------------------------------------
# An Advanced Virgo Plus input file for Finesse 3

# modulation frequencies [TDR, tab 2.3, pp 24]
var f6 6270777   # 6 MHz,  fmod1 in TDR
var f8 4/3*f6   # 8 MHz,  4 / 3 * f6, fmod3 in TDR
var f56 9*f6     # 56 MHz, 9 * f6, fmod2 in TDR

var nsilica 1.44963
var Mloss 30u

var etalonNI 0  # NI etalon tuning
var etalonWI 0  # WI etalon tuning

# Laser and modulators
#######################################################################
# EOM parameters from https://logbook.virgo-gw.eu/virgo/?r=34898
# and https://logbook.virgo-gw.eu/virgo/?r=38123
# and https://logbook.virgo-gw.eu/virgo/?r=41551
laser i1 P=40.0
s s0 i1.p1 eom6.p1 L=1m
mod eom6 f=f6 midx=0.22
s sEOM6 eom6.p2 eom8.p1 L=0.1
mod eom8 f=f8 midx=0.15
s sEOM8 eom8.p2 eom56.p1 L=0.1
mod eom56 f=f56 midx=0.25
```

Example: Running an Experiment

For each modulation frequency:

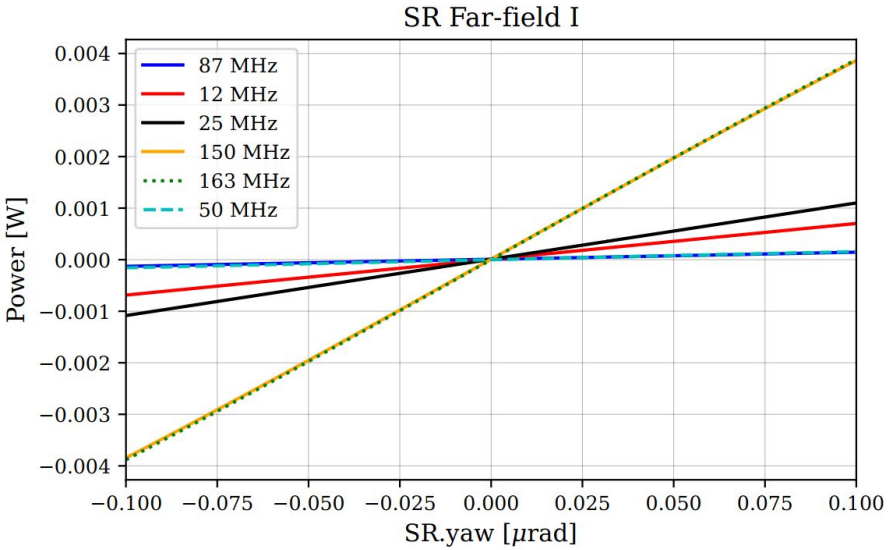
1. Set modulation frequency
2. Add sideband-of-sideband (SoS) frequency
3. Set demodulation frequency
4. Optimize demodulation phase
5. Misalign mirror

```
var f50 (eom56.f-f6)
var f31 (5*eom6.f)
var f87 (eom56.f+(5*f6))
var f68 (11*eom6.f)
var f12 ((11*f6)-eom56.f)
var f81 (13*eom6.f)
var f25 ((13*f6)-eom56.f)
```

```
var f131 (21*eom6.f)
var f75 ((21*f6)-eom56.f)
var f206 (33*eom6.f)
var f150 ((33*f6)-eom56.f)
var f219 (35*eom6.f)
var f163 ((35*f6)-eom56.f)
```

```
model.add_frequency(-virgo.model.get(sos).value)
model.add_frequency(virgo.model.get(sos).value)
```

```
def misalign_BS(virgo, start=-0.05e-6, stop=0.05e-6, steps=200):
    return virgo.model.run(
        fa.Xaxis(
            'BS_x.DC',
            'lin',
            start,
            stop,
            steps,
            relative=True,
            pre_step=fa.RunLocks(method="newton"),
        )
    )
```



Summary

The `finesse-virgo` package includes top-level tools and models for simulating Virgo in Finesse 3.

Install with `pip install finesse_virgo`

1. Start with your modified katfile (or copy the common file to a local directory)
2. Use your modified katfile with the Virgo object
3. Pre-tune the model (check the figures of merit)
4. Save a tuned version (with locks)
5. Run the experiment while maintaining operating point.