Finesse 3

Nikhef

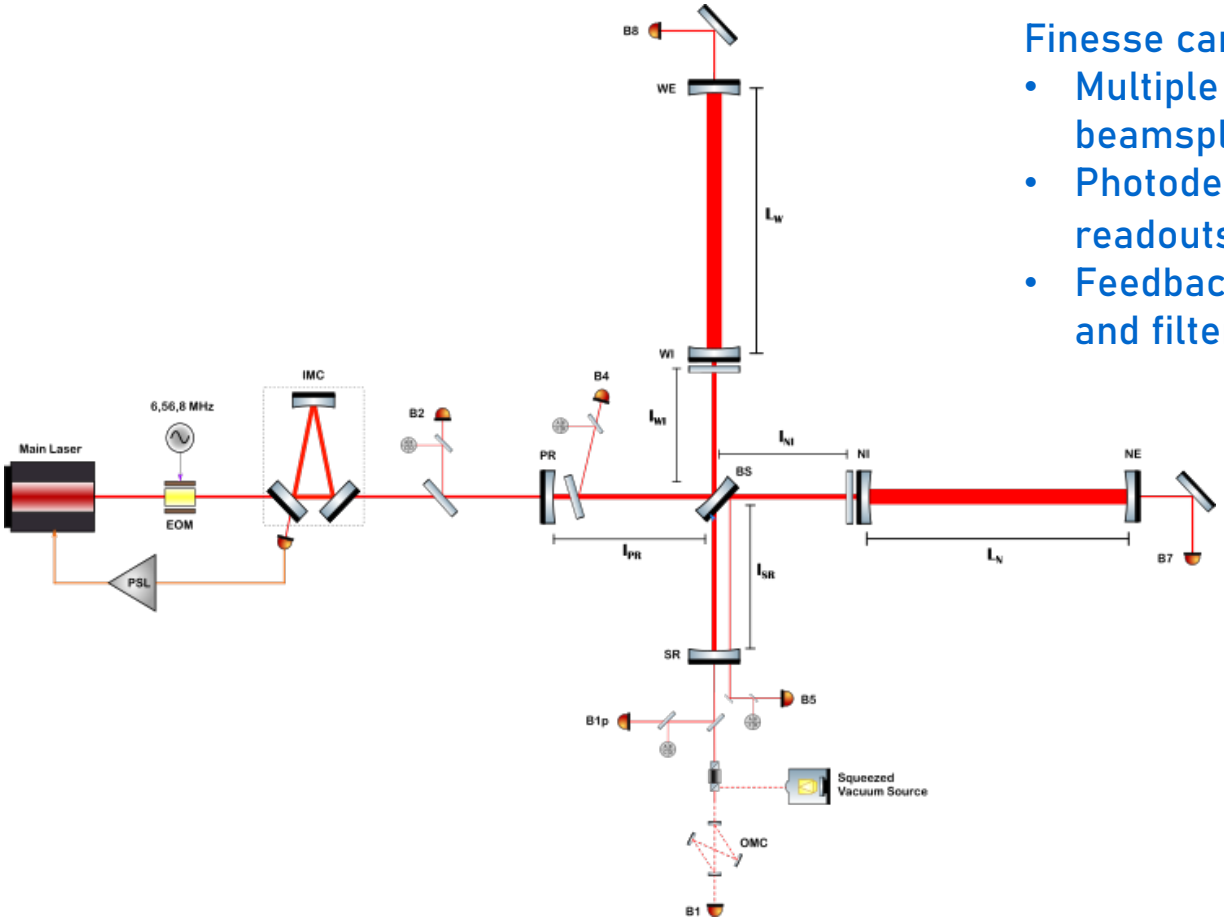
Modeling Current and Next-Generation GW detectors with Finesse.

Enzo Tapia S. for the Finesse team

Well first, what is Finesse?

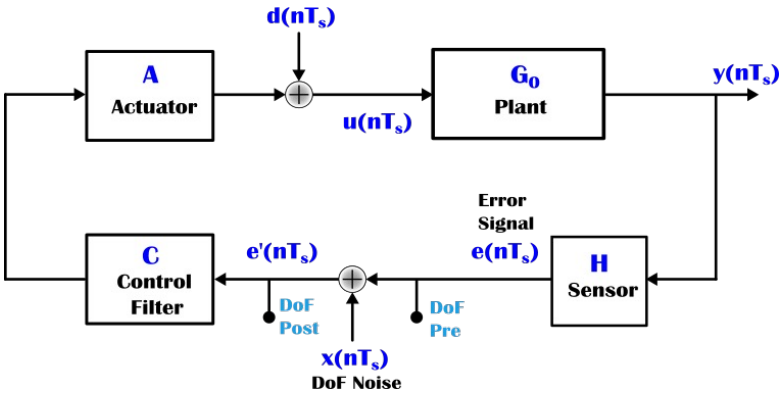
- A simulation tool for modeling optical interferometers.
- Open-source and hosted on Gitlab.
- Initially developed in 1997 by Andreas Freise.
- Open sourced in 2012, led by Daniel Brown.
- Now a Nikhef-based project, led by Andreas Freise and continuously maintained and developed by a small, evolving team (1-5 people, primarily PhD researchers). With support from 2.5 FTE (Full Time Equivalent) software engineers.

Modeling interferometers



Finesse can simulate full interferometer setups:

- Multiple optical components (mirrors, beamsplitters, lenses, modulators).
- Photodetectors, demodulation schemes and optical readouts.
- Feedback control systems via transfer functions and filters.



What makes Finesse special?

- Developed by and specifically for the ground-based GW community.
- Versatile simulation engine for modeling a wide range of complex optical systems.
- Actively supports the needs of LIGO, Virgo and KAGRA, as well as for the design of Einstein Telescope (ET) and Cosmic Explorer (CE).

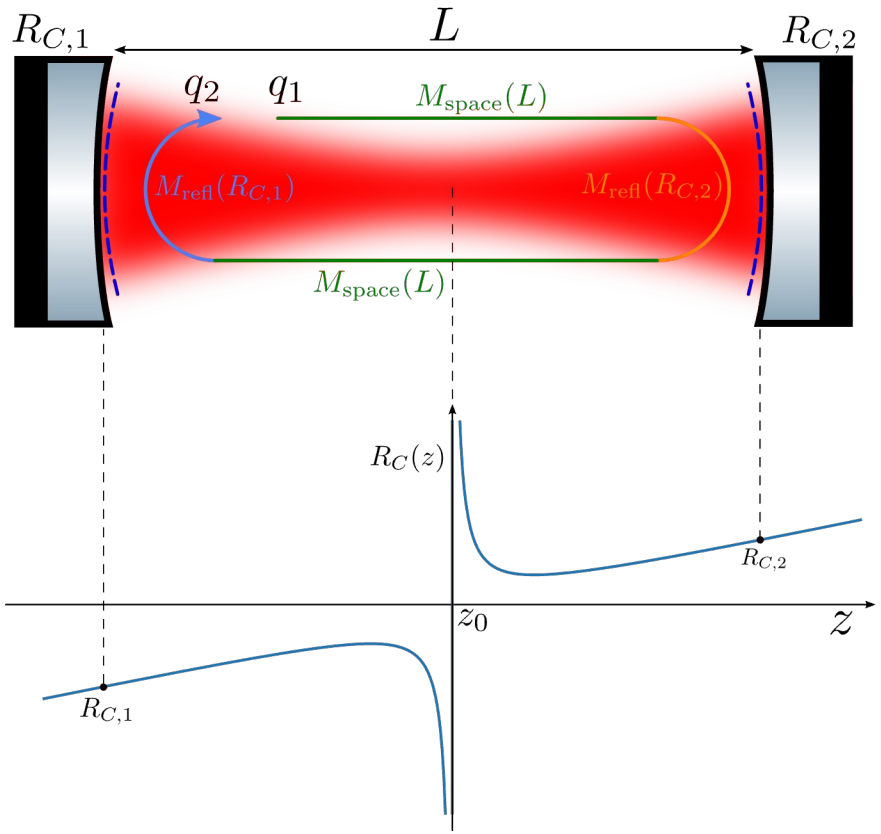
Enter Finesse3

- Full redesign of the tool, in development since 2017.
- Focused on robust interferometer modeling, with an emphasis on speed and modularity.
- Incorporates new features and a variety of optical, electrical and mechanical components.

Finesse 3

What is Finesse Solving?

$$M_{\mathrm{rt}} = M_{\mathrm{space}}(L) \times M_{\mathrm{refl}}(R_{C,2}) \times M_{\mathrm{space}}(L) \times M_{\mathrm{refl}}(R_{C,1})$$



Optical cavity case:

Finesse computes the cavity eigenmode (unchanged after one round trip).
This beam is Gaussian, characterized by the complex beam parameter q.
q encodes beam waist size (w_0), position (z_0) and wavefront curvature (R_C).

The evolution of q through the cavity is computed using ABCD matrices:

$$q' = \frac{Aq + B}{Cq + D}$$

A,B,C,D: elements of the round-trip matrix M_{rt}.

Finesse's core approach

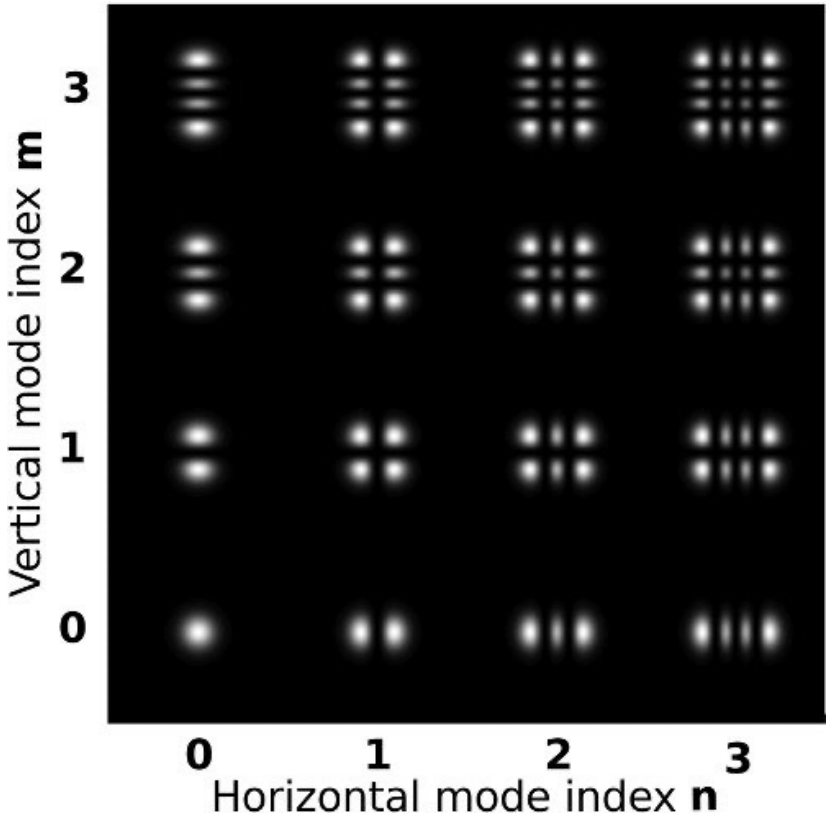
Frequency domain (steady state) analysis.

Paraxial approximation:

- Beam evolves slowly along the propagation axis (z).
- Beam stays close to the optical axis (centered propagation).

Mode expansion:

- Optical fields expressed as a sum of Hermite-Gauss modes.
- Modes are solutions to the paraxial wave equation. Which describe the transverse shape of the beam.



$$E(x,y,z) = e^{-ikz} \sum_{n,m} a_{nm}\, u_{nm}(x,y,z)$$

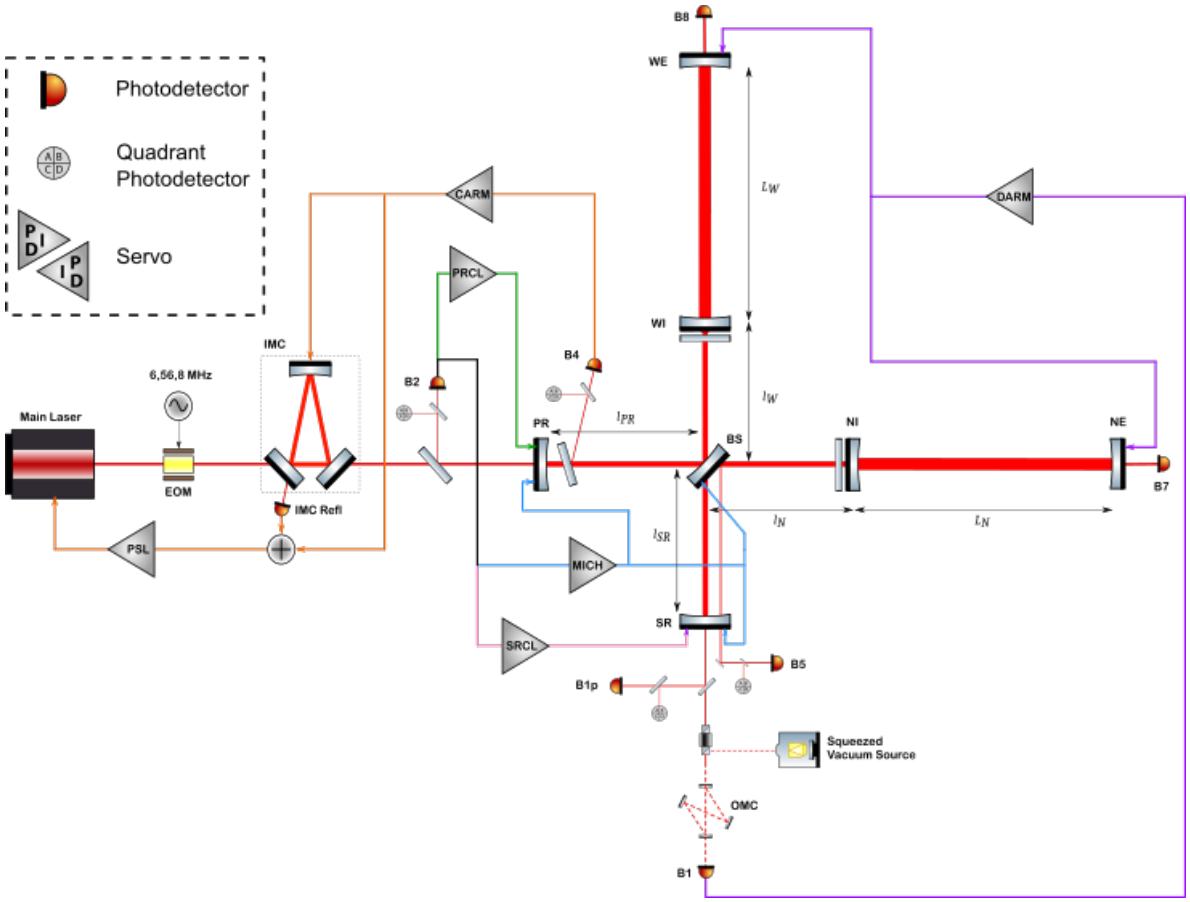
$u_{nm}(x,y,z)$: Hermite-Gauss mode functions.
a_{nm}: complex amplitude coefficient.
k: wave number.


Advanced Virgo Detector
40
7 / 18

A model of the Virgo detector

Dedicated finesse-virgo package developed based on the detector's nominal parameters.

Most of the core optics, degrees of freedom (DoF) and controls are implemented.



A model of the Virgo detector

Main components are organized in two files:

- One defining the optical components.
- One describing detectors and readouts.

```
## Description of the detector:
# Laser with P = 25 W at the default frequency
l laser1 P=25

# Modulations
mod eom1 f=6320742 midx=0.15 order=1
mod eom2 f=56886678 midx=0.15 order=1

# ITF input
link(laser1, 0.1, eom1, 0.1, eom2, PR1.p1)

# Power Recycling Cavity mirrors
m PR1 L=30e-6 T=0.04835 Rc=5.209 phi=45

bs PR2 R=1 T=0 Rc=-1.929
s L1 PR1.p2 PR2.p1 L=8.0619

bs PR3 R=1 T=0 Rc=20.2
s L2 PR2.p2 PR3.p1 L=9.279

# Beam Splitter
bs BS L=30e-6 T=0.5012 alpha=45
s L3 PR3.p2 BS.p1 L=12.33115539

# North Arm
m NITM L=27e-6 T=0.01377 Rc=-1420 phi=45
s L4 BS.p3 NITM.p1 L=6.015178
```

*Simplified version.

Alternatively, python interface:

```
from finesse import Model
from finesse.components import (Laser, Beamsplitter, Mirror)
from finesse.detectors import PowerDetector

virgo_model = Model()
l1 = Laser("l1", P=25)
bs = Beamsplitter("bs", R=0.5, T=0.5)
virgo_model.link(l1, bs)

NE = Mirror("NE", R=0.99, T=0.01)
virgo_model.link(bs.p3, NE)
WE = Mirror("WE", R=0.99, T=0.01)
virgo_model.link(bs.p2, WE)

pd1 = PowerDetector("pd1", virgo_model.bs.p4.o)
```

*Simple Michelson interferometer.

Finding and operating point

Crucial step → bring the model to operating point.
Simply placing the components in the simulation isn't enough.
➢ We need to "commission" the model, just like the real detector.

e.g. Adjustment of Recycling Cavities:

```
model1.get('lPOP_BS').L
```

```
<lPOP_BS.L=5.9399 @ 0x15dcfbf40>
```

```
adjust_recycling_cavity_length(model1, "PRC", "lPRC", "lPOP_BS", verbose = True)
```

```
—  adjusting PRC length
   adjusting lPOP_BS.L by 0.0004736 m
```

```
model1.get('lPOP_BS').L
```

```
<lPOP_BS.L=5.94037359652047 @ 0x15dcfbf40>
```

```
model1.get('lsr').L
```

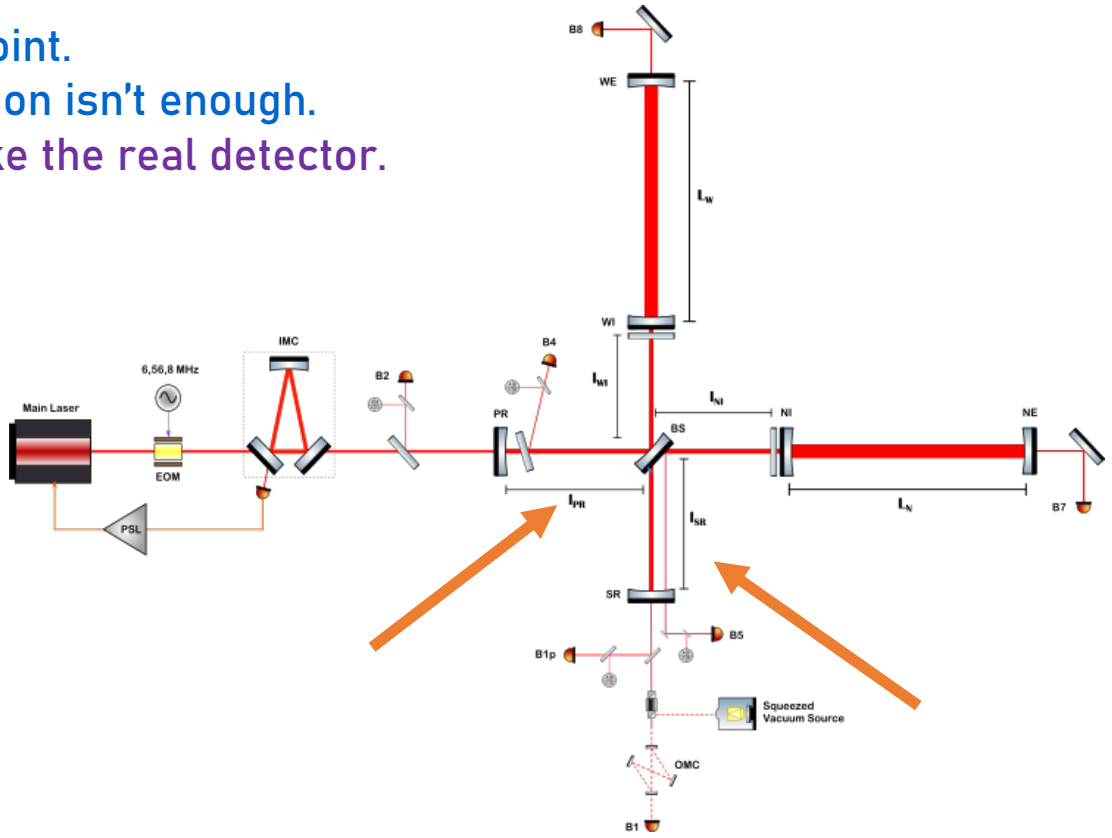
```
<lsr.L=5.943 @ 0x15ddf9a80>
```

```
adjust_recycling_cavity_length(model1, "SRC", "lSRC", "lsr", verbose = True)
```

```
—  adjusting SRC length
   adjusting lsr.L by 0.000883 m
```

```
model1.get('lsr').L
```

```
<lsr.L=5.94388296505047 @ 0x15ddf9a80>
```



Finding and operating point

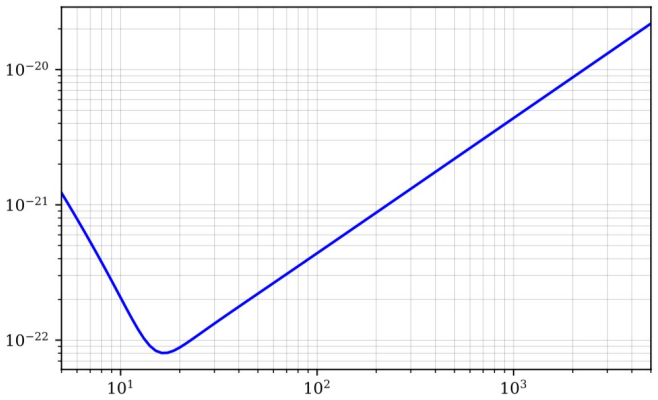
Crucial step → bring the model to operating point.
Simply placing the components in the simulation isn't enough.
➢ We need to "commission" the model, just like the real detector.

Figures of Merit:

- Maximum sensitivity (e.g. quantum noise sensitivity).
- Cavities on resonance and aligned.
- Verification of main DoF transfer functions.
- … and other indicators.

```
out_sensitivity1 = get_QNLS(model1)
plt.loglog(out_sensitivity1.x1, np.abs(out_sensitivity1['NSR_with_RP']))
range_bns1 = inspiral_range.range(out_sensitivity1.x1, np.abs(out_sensitivity1['NSR_with_RP'])**2)
print('BNS range:', np.round(range_bns1, 3), 'Mpc')
```

```
BNS range: 17.262 Mpc
```



* Sensitivity of the model away from the operating point.

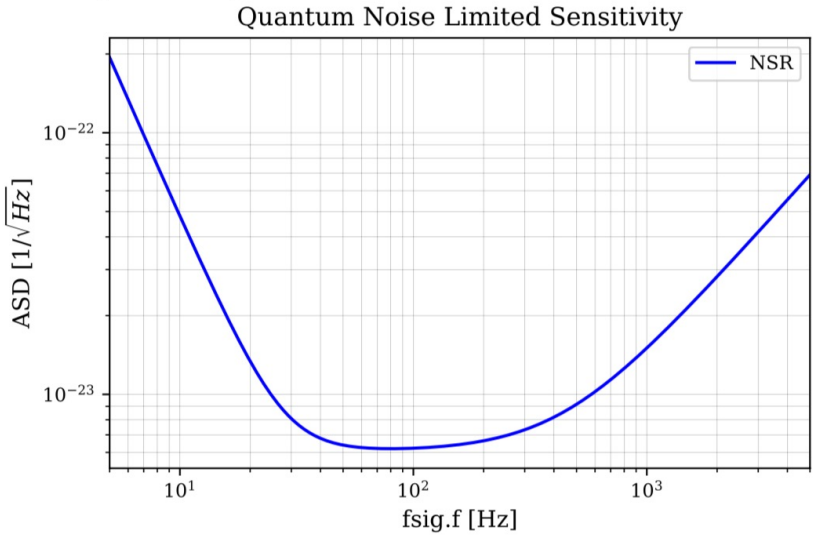
Finding and operating point:

Model must ensure mirrors dynamically adjust, with pm accuracy, to maintain resonance conditions across 5+ degrees of freedom:

- Tune demodulation phases → maximize optical gain, reduce cross-couplings.
- Engage control loops of the main DoFs (Longitudinal and Angular)
- Once locked (operating point) → evaluation of sensitivity.

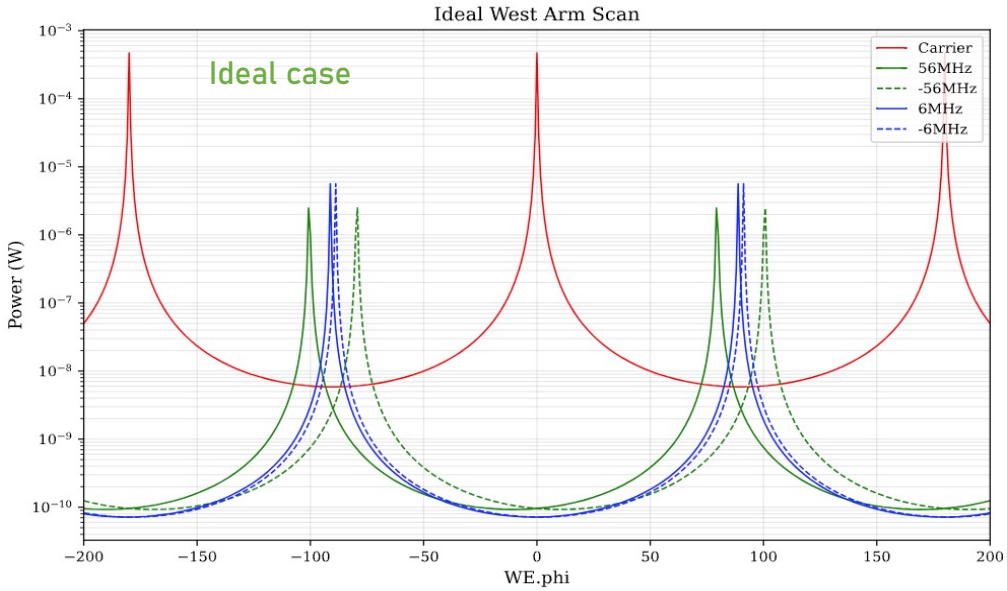
```
virgo.plot_QNLS()
out4 = virgo.get_QNLS()
range_bns4 = inspiral_range.range(out4.x1, np.abs(out4['NSR_with_RP'])**2)
print('BNS range:', np.round(range_bns4, 1), 'Mpc')
```

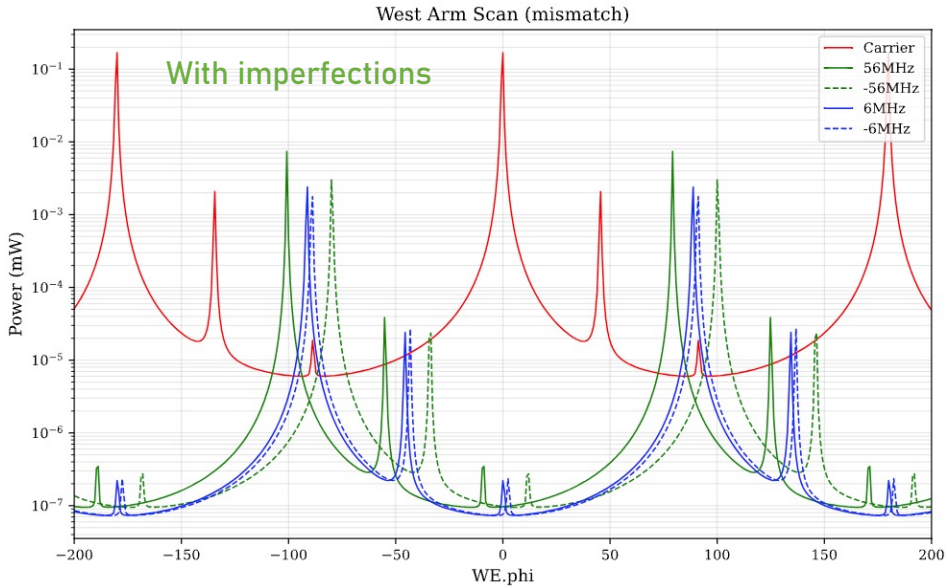
```
BNS range: 162.9 Mpc
```



From Ideal Case to Realistic Interferometer

Model with nominal → incorporate known (measured) imperfection.





Mode-mismatch in arm cavities leads to Higher Order Modes (HOMs) resonating in the interferometer.

Mode-mismatch?
When the beam's shape does not match the cavity or optical system's ideal mode.

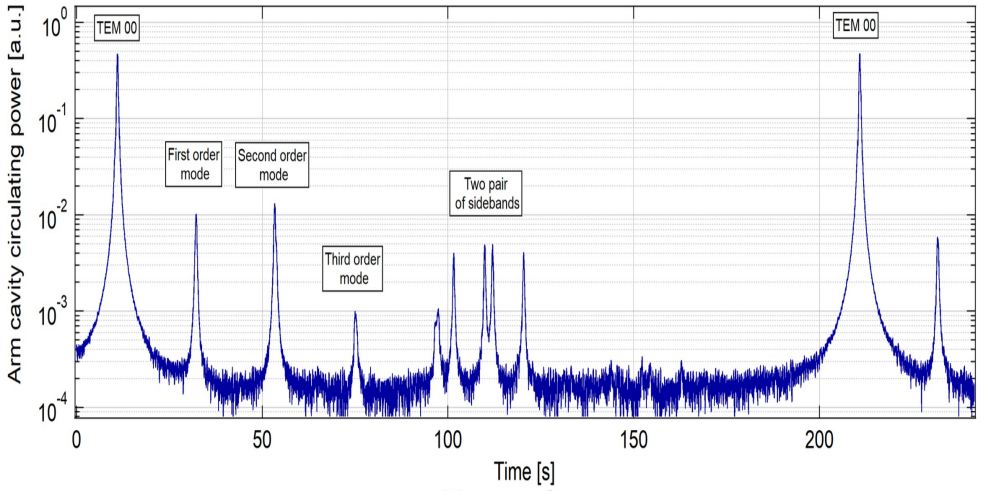
From Ideal Case to Realistic Interferometer

Goal: Model refinement to reproduce Advanced Virgo behavior.

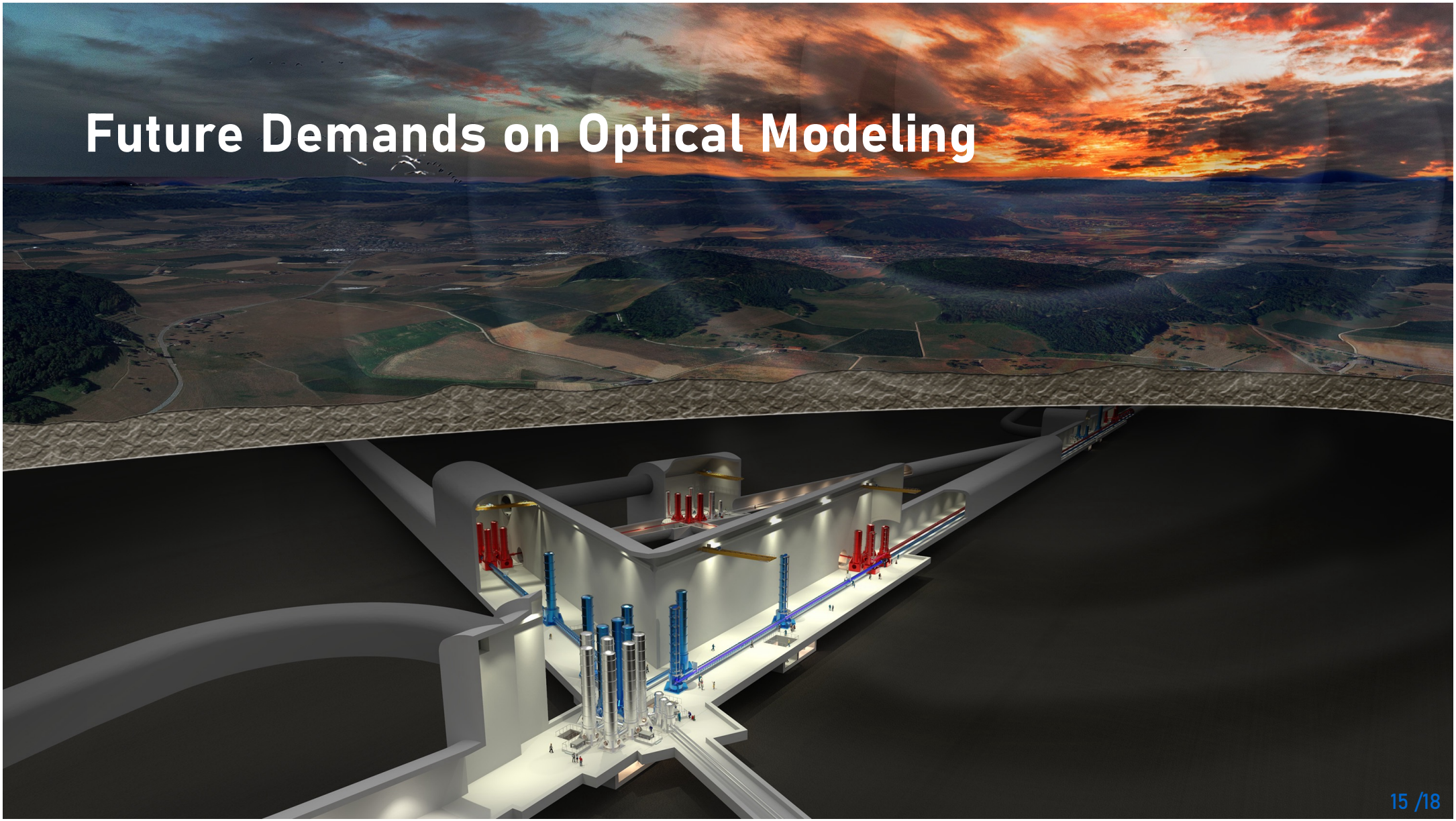
How? Compare simulations with Optical Characterization measurements.

Ongoing:

- Fine-tuning Radius of Curvature (RoC).
- Accounting for astigmatism.
- Incorporating losses.
- Many more …



Future Demands on Optical Modeling

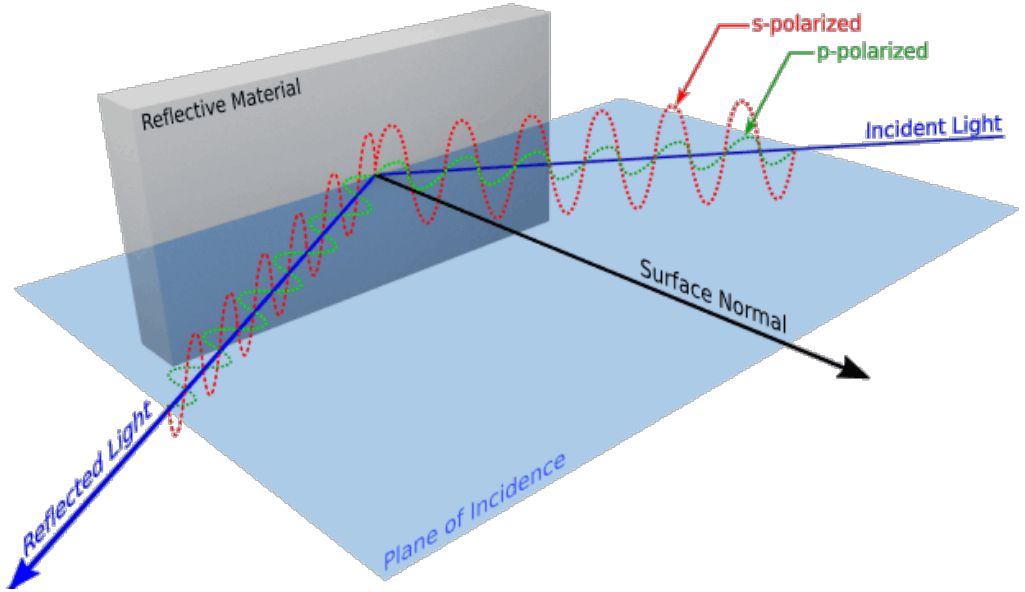


Preparations for the Einstein Telescope (ET)

Currently, Finesse models the electric fields as a one-dimensional complex fields.

$$E(\vec{r},t) = E_0 e^{i(\vec{k}\cdot\vec{r}-\omega t)}$$

A vectorial representation of the electric fields is being implemented in Finesse. Following the evolution of E_s and E_p through any given optical system.



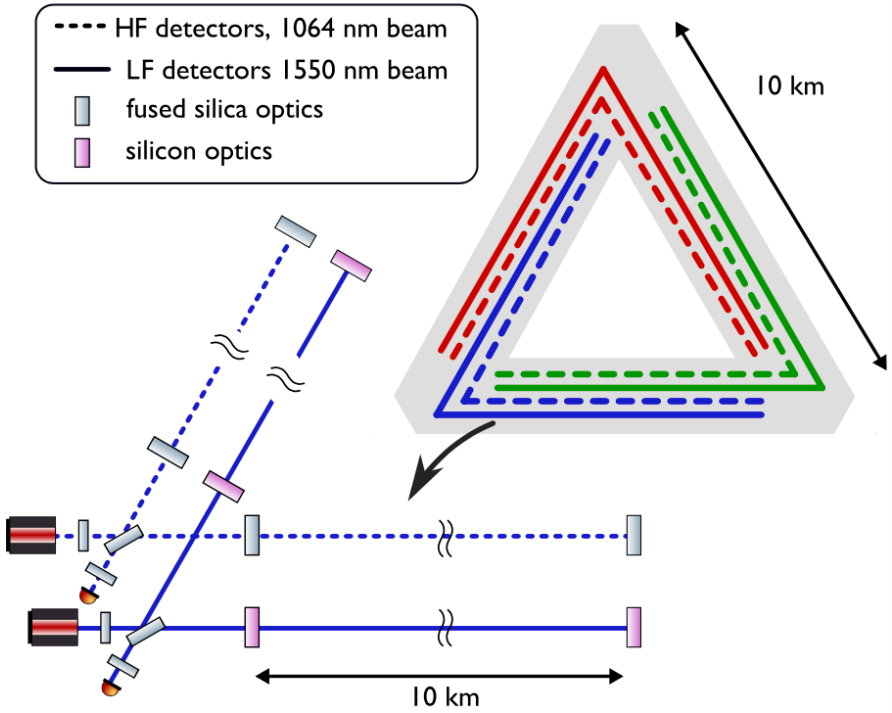
- p-polarization: parallel to the plane of incidence.
- s-polarization: perpendicular to the plane of incidence.

Preparations for ET: Why Polarization?

Polarization effects such as birefringence and anisotropic coatings can significantly affect the phase, contrast, and stability of the interferometer.

Materials like silicon (planned for ET) or sapphire (used in KAGRA) at cryogenic temperatures may introduce birefringence → vectorial modeling is essential.

Lessons from KAGRA:
KAGRA experienced fringe contrast degradation and alignment instability due to unmodeled polarization effects.



Silicon mirrors could alter beam polarization in ET cavities, possibly impacting performance.

Summary and Conclusions

- Widely used for modeling complex optical setups in current detectors like LIGO, Virgo and KAGRA, and in designing future detectors such as ET and CE.
- It comes with extensive documentation and available online: https://finesse.ifosim.org/
- Backed by an active user and developer community, including experts from Nikhef and beyond.

Finesse 3