Symbolic mode computations for fused silica TIR cavity

Evan D. Hall

Invalid Date

Table of contents

1 Round-trip $ABCD$ matrix and cavity stability **2**

2 Gouy phase **3**

3 Free spectral range **3**

4 Transverse mode spacing **4**
4.1 Variation with RoC 4
4.2 Waist size 6

5 Cross-check: LIGO's linear FP cavities **7**

```
import os
import sys
```

```
import matplotlib as mpl
import numpy as np
import sympy as s

from matplotlib import pyplot as plt
from scipy import constants as const
```

```
s.init_printing()
```

```
def makegrid(ax):
    ax.grid(True, which="major", color="k", alpha=0.05)
    ax.grid(True, which="minor", color="k", alpha=0.025)
```

1 Round-trip $ABCD$ matrix and cavity stability

First we define the variables we'll use for the symbolic computation.

```
ell, R, n, lam = s.var("ell R n lambda", real=True)
L = s.var("L", real=True)
```

We also define the numerical substitutions we'll use later, including the tangential and sagittal radii of curvature.

```
aoi = np.pi / 4 # rad
R_n = 5 # m
lam_n = 1064e-9 # m
ell_n = np.sqrt(2) * 0.0254 # m
L_n = 4 * ell_n # m
Rt_n = R_n * np.cos(aoi) # m
Rs_n = R_n / np.cos(aoi) # m
n_n = 1.45 # --
```

```
subs = [(n, n_n), (ell, ell_n), (lam, lam_n)]
subst = [(R, Rt_n)]
subss = [(R, Rs_n)]
```

We need the matrix to propagate the beam from bounce to bounce, and the reflection matrix at the curved surfaces.

```
Mprop = s.Matrix([[1, ell/n], [0, 1]])
Mrefl = s.Matrix([[1, 0], [-2*n/R, 1]])
```

Now we can find the round-trip propagation matrix, treating the bounces at the flat faces as identity matrices.

```
Mtot = s.simplify(Mprop @ Mrefl @ Mprop @ Mprop @ Mrefl @ Mprop)
Mtot
```

$$\begin{bmatrix} 1 - \frac{8\ell}{R} + \frac{8\ell^2}{R^2} & \frac{4\ell\left(R^2 - 3R\ell + 2\ell^2\right)}{R^2 n} \\ \frac{4n(-R+2\ell)}{R^2} & 1 - \frac{8\ell}{R} + \frac{8\ell^2}{R^2} \end{bmatrix}$$

We define the cavity stability as $(A + D)/2$. A stable cavity has $|(A + D)/2| < 1$.

```
stab = (Mtot[0,0] + Mtot[1,1]) / 2
stab
```

$$1 - \frac{8\ell}{R} + \frac{8\ell^2}{R^2}$$

2 Gouy phase

The Gouy phase of the TEM_{mn} mode is

$$\zeta_{mn} = \left(m + \frac{1}{2}\right)\zeta_\mathrm{T} + \left(n + \frac{1}{2}\right)\zeta_\mathrm{S}$$

with

$$\zeta_{\mathrm{T,S}} = \operatorname{sgn} B \arccos\left(\frac{A_{\mathrm{T,S}} + D_{\mathrm{T,S}}}{2}\right),$$

T referring to the tangential $ABCD$ parameters and S to the sagittal parameters (assuming the signs of B_T and B_S are identical).

```
gouy_t = s.acos(stab).subs(subs + subst).evalf()
gouy_s = s.acos(stab).subs(subs + subss).evalf()
print(f"Gouy phase (T): {gouy_t * 180 / np.pi:.1f} deg.")
print(f"Gouy phase (S): {gouy_s * 180 / np.pi:.1f} deg.")
```

```
Gouy phase (T): 23.1 deg.
Gouy phase (S): 16.3 deg.
```

3 Free spectral range

The FSR is $\nu_\mathrm{FSR} = c/nL$, with L being the *round trip physical length* of the cavity.

```
fsr = const.c / n_n / L_n
print(f"FSR: {fsr/1e9:.3f} GHz")
```

```
FSR: 1.439 GHz
```

4 Transverse mode spacing

The transverse mode spacing for higher-order modes (HOMs) can be found from the Gouy phase as follows. To maintain resonance, the longitudinal mode frequency ν_{mn} for the TEM(m,n) mode must satsify

$$2\pi p = k_{mn}L + \zeta_{mn} = \frac{2\pi\nu_{mn}}{c}L + \zeta_{mn} = \frac{2\pi\nu_{mn}}{\nu_{\mathrm{FSR}}} + \zeta_{mn}$$

for some $p \in \mathbb{Z}$. Hence,

$$\nu_{mn} - \nu_{00} = \nu_{\mathrm{FSR}} \times \frac{\zeta_{mn} - \zeta_{00}}{2\pi} = \nu_{\mathrm{FSR}} \times \frac{m\zeta_{\mathrm{T}} + n\zeta_{\mathrm{S}}}{2\pi}.$$

In particular, the $(1,0)$ and $(0,1)$ modes have spacings

$$\nu_{\mathrm{TMS,T}} = \nu_{10} - \nu_{00} = \nu_{\mathrm{FSR}} \times \frac{\zeta_{\mathrm{T}}}{2\pi} \tag{1}$$

$$\nu_{\mathrm{TMS,S}} = \nu_{01} - \nu_{00} = \nu_{\mathrm{FSR}} \times \frac{\zeta_{\mathrm{S}}}{2\pi}. \tag{2}$$

```
tms_t = fsr / (2 * np.pi) * gouy_t
tms_s = fsr / (2 * np.pi) * gouy_s
print(f"TMS (T): {tms_t / 1e6:.0f} MHz")
print(f"TMS (S): {tms_s / 1e6:.0f} MHz")
```

```
TMS (T): 92 MHz
TMS (S): 65 MHz
```

Since we observe some HOM nearly co-resonant with $(0,0)$, we can compute

```
print(f"Nearest TEM(0,n) mode order n: {fsr // tms_s}")
print(f"Location of TEM(0,n) below 00 mode: {(fsr % tms_s) / 1e6:.2f} MHz")
```

```
Nearest TEM(0,n) mode order n: 22
Location of TEM(0,n) below 00 mode: 1.31 MHz
```

4.1 Variation with RoC

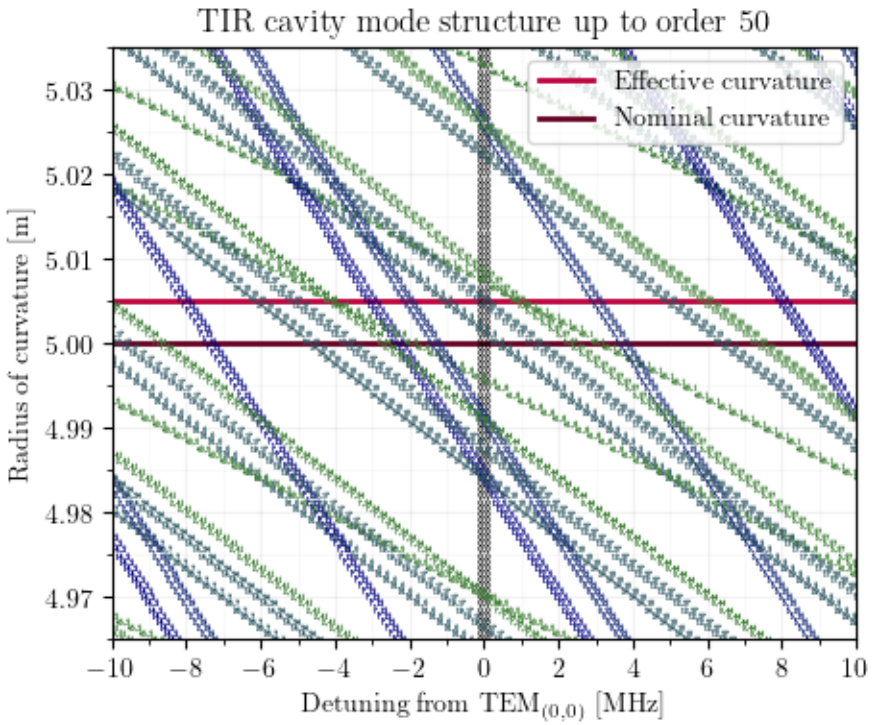
Now we vary R and look at nearby transverse modes.

```
gouy_np = s.lambdify((ell, R), s.acos(stab))
R_arr = np.linspace(4.965, 5.035, 100)
gouy_t_arr = gouy_np(ell_n, R_arr * np.cos(aoi))
gouy_s_arr = gouy_np(ell_n, R_arr / np.cos(aoi))
```

```
temmax = 50
m_arr, n_arr = np.meshgrid(np.arange(50), np.arange(50))
```

```
fig_tms_roc, ax_tms_roc = plt.subplots()
plot_thresh = 10e6 # Hz
plot_colors = np.array([3*[0]] + [[x/3, 2*x/3, (x/2+0.5)*(1-x)]
                                  for x in np.linspace(0, 1, 5)])
mdict = dict(mew=0.2)#, ms=20)
ax_tms_roc.plot(plot_thresh / 1e6 * np.array([-1, 1]), 2*[5.005],
                label="Effective curvature", color="xkcd:rose red")
ax_tms_roc.plot(plot_thresh / 1e6 * np.array([-1, 1]), 2*[5],
                label="Nominal curvature", color="xkcd:maroon")
for ii, (mm, nn) in enumerate(zip(m_arr.ravel(), n_arr.ravel())):
    if mm + nn < temmax:
        nu_tms_mn_arr = fsr / (2 * np.pi) * (mm * gouy_t_arr + nn * gouy_s_arr)
        nu_tms_mn_arr_mod = nu_tms_mn_arr % fsr
        nu_tms_mn_arr_mod[nu_tms_mn_arr_mod > fsr/2] -= fsr
        if np.any(np.abs(nu_tms_mn_arr_mod) < plot_thresh):
            mt = f"${mm},{nn}$"
            ll = ax_tms_roc.plot(
                nu_tms_mn_arr_mod / 1e6,
                R_arr,
                color=plot_colors[(mm + nn) // 10],
                marker=mt,
                lw=0,
                ms = 1*(len(mt)-1),
                #markevery=(nn, 1),
                **mdict,
            )
        #if ii == 0:
        #    plt.setp(ll, color="k")
ax_tms_roc.set_xlim(plot_thresh / 1e6 * np.array([-1, 1]))
ax_tms_roc.set_ylim([R_arr[0], R_arr[-1]])
ax_tms_roc.xaxis.set_major_locator(mpl.ticker.MultipleLocator(2))
ax_tms_roc.xaxis.set_minor_locator(mpl.ticker.MultipleLocator(1))
ax_tms_roc.yaxis.set_major_locator(mpl.ticker.MultipleLocator(0.01))
ax_tms_roc.yaxis.set_minor_locator(mpl.ticker.MultipleLocator(0.005))
```

```
ax_tms_roc.set_xlabel("Detuning from TEM$_{(0,0)}$ [MHz]")
ax_tms_roc.set_ylabel("Radius of curvature [m]")
ax_tms_roc.set_title(f"TIR cavity mode structure up to order {temmax}")
ax_tms_roc.legend(loc="upper right")
makegrid(ax_tms_roc)
fig_tms_roc.savefig("tir_tms_roc.pdf")
```



4.2 Waist size

From Kogelnik and Li, given a set of ABCD parameters describing a round-trip propagation through a stable cavity, the cavity's q paramter is given by

$$\frac{1}{q} = \frac{D-A}{2B} \pm \frac{4\mathrm{i}}{2B}\sqrt{1-\left(\frac{A+D}{2}\right)^2}.$$

Hence,

$$w^2 = \frac{2\lambda B/\pi}{\sqrt{1-\left(\frac{A+D}{2}\right)^2}}.$$

Note that in our case, $A = D$, implying $\operatorname{Re} q = 0$, so the beam is planar. We have defined our round-trip matrix to start and end at the planar faces, which (in terms of optical path length) are equidistant from the curved faces, so we indeed expect a planar phase front for the beam here.

```
w0 = s.sqrt(2 * lam * Mtot[0,1] / s.pi / s.sqrt(1 - stab**2))
w0_t = w0.subs(subs + subst).evalf()
w0_s = w0.subs(subs + subss).evalf()
print(f"Waist (T): {w0_t*1e3:.2f} mm")
print(f"Waist (S): {w0_s*1e3:.2f} mm")
```

```
Waist (T): 0.41 mm
Waist (S): 0.48 mm
```

5 Cross-check: LIGO's linear FP cavities

```
Re, Ri = s.var(r"R_\text{E} R_\text{I}")
```

```
L_LIGO = 3995 # m; the ONE-WAY length now
fsr_LIGO = const.c / (2 * L_LIGO)
```

```
Mfp = s.simplify(Mprop @ Mrefl.subs(R, Re) @ Mprop @ Mrefl.subs(R, Ri)).subs([(ell, L), (n, 1)])
```

```
gouy_LIGO = s.acos((Mfp[0,0] + Mfp[1,1]) / 2).evalf(subs={Re: 2245, Ri: 1934, L: 4e3})
print(f"Round-trip Gouy phase (LIGO): {gouy_LIGO*180/np.pi:.1f} deg.")
```

```
Round-trip Gouy phase: 47.9 deg.
```

```
tms_LIGO = fsr_LIGO / (2 * np.pi) * gouy_LIGO
print(f"TMS (LIGO): {tms_LIGO:.0f} Hz")
```

```
TMS (LIGO): 4994 Hz
```