Cavity Model

Jorge Alberto Diaz Cruz

March 18, 2020

Contents

1 From math to code

This section describes the mathematical manipulation of the cavity's differential equation so it can be implement in python code. Let's start with a set of two first order diferential equations derived in [1]:

$$\frac{d\vec{S}_\mu}{dt} = -\omega_{f_\mu}\vec{S}_\mu + \omega_{f_\mu}e^{-j\theta_\mu}\left(2\vec{K}_g\sqrt{R_{g_\mu}} - R_{b_\mu}\vec{I}_{\text{beam}}\right) \tag{1.1}$$

$$\frac{d\theta_\mu}{dt} = \omega_{d_\mu} \tag{1.2}$$

Where the subcript μ represents each eigenmode of the cavity and:

$$\vec{V}_\mu = \vec{S}_\mu e^{j\theta_\mu} \tag{1.3}$$

ω_{f_μ}:	Cavity bandwidth
$\vec{K}_{\text{g}}$:	Incident wave amplitude in $\sqrt{\text{Watts}}$
$R_{\text{g}_\mu} = Q_{\text{g}_\mu}(R/Q)_\mu$:	Coupling impedance of the drive port
$\vec{I}_{\text{beam}}$:	Beam current
$R_{\text{b}_\mu} = Q_{\text{L}_\mu}(R/Q)_\mu$:	Coupling impedance to the beam
$\omega_{d_\mu} = 2\pi\Delta f_\mu$:	Detune frequency

Let's now define:

$a = 2\sqrt{R_{g_\mu}}\omega_{f_\mu}$
$b = R_{b_\mu}\omega_{f_\mu}$
$c = \omega_{f_\mu}$

So we can write equation 1.1 as:

$$\frac{d\vec{S}_\mu}{dt} = -c\vec{S}_\mu + e^{-j\theta_\mu}\left(a\vec{K}_g - b\vec{I}_{\text{beam}}\right) \tag{1.4}$$

For a complex $\vec{S}_\mu = S_r + jS_i$ we can split equation 1.4 in the real and imaginary parts:

$$\frac{d\vec{S}_\mu}{dt} = -c(S_r + jS_i) + (cos\theta - jsin\theta)(a\vec{K}_g - b\vec{I}_{\text{beam}}) \tag{1.5}$$

$$\frac{dS_r}{dt} = (a\vec{K}_g - b\vec{I}_{\text{beam}})cos\theta - cS_r \tag{1.6}$$

$$\frac{dS_i}{dt} = -(a\vec{K}_g - b\vec{I}_{\text{beam}})sin\theta - cS_i \tag{1.7}$$

If we also assume a complex incident wave $\vec{K}_g = K_r + jK_i$ we have:

$$\frac{dS_r}{dt} = (aK_r - b\vec{I}_{\text{beam}})cos\theta - cS_r + aK_i sin\theta \tag{1.8}$$

$$\frac{dS_i}{dt} = -(aK_r - b\vec{I}_{\text{beam}})sin\theta - cS_i + aK_i cos\theta \tag{1.9}$$

Equations 1.2, 1.8 and 1.9 can be implemendted in python using the following code:

```
# Define Cavity model
def cavity(z, t, RoverQ, Qg, Q0, Qprobe, bw, Kg_r, Kg_i, Ib, foffset):

Rg = RoverQ * Qg
Kdrive = 2 * np.sqrt(Rg)
Ql = 1.0 / (1.0/Qg + 1.0/Q0 + 1.0/Qprobe)
K_beam = RoverQ * Ql
w_d = 2 * np.pi * foffset

a = Kdrive * bw
b = K_beam * bw
c = bw

yr, yi, theta = z

dthetadt = w_d
dydt_r = (a * Kg_r - b * Ib)*np.cos(theta) - (c * yr) + a * Kg_i * np.sin(theta)
dydt_i = -(a * Kg_r - b * Ib)*np.sin(theta) - (c * yi) + a * Kg_i * np.cos(theta)
return dydt_r, dydt_i, dthetadt
```

References

[1] LBNL LLRF team, "LCLS-II System Simulations: Physics," October 7, 2015.