Optickle simulation tool for Alignment simulations in high power interferometers

M. Mantovani

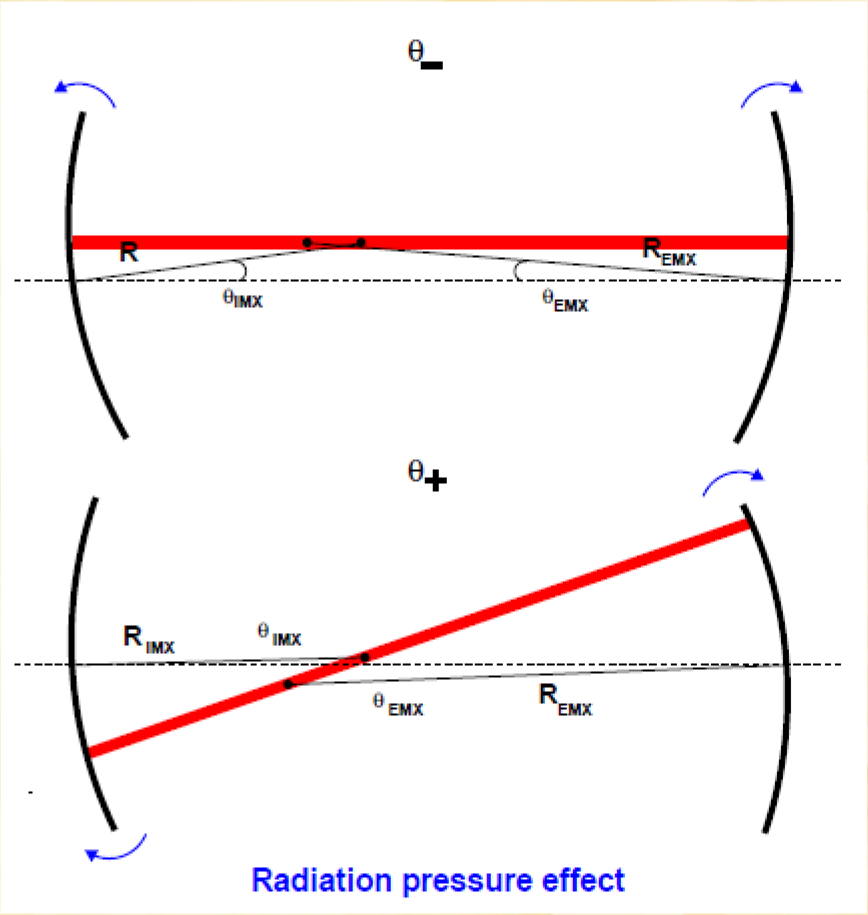
Introduction

- Comparison between `Optickle` and `Finesse`
- Optickle functions description
- Modeling a single Fabry-Perot
  - build the configuration file
  - evaluate the resonance conditions
  - simulate the Mechanical TF
  - evaluate the AA error signals

Introduction: Optickle vs Finesse

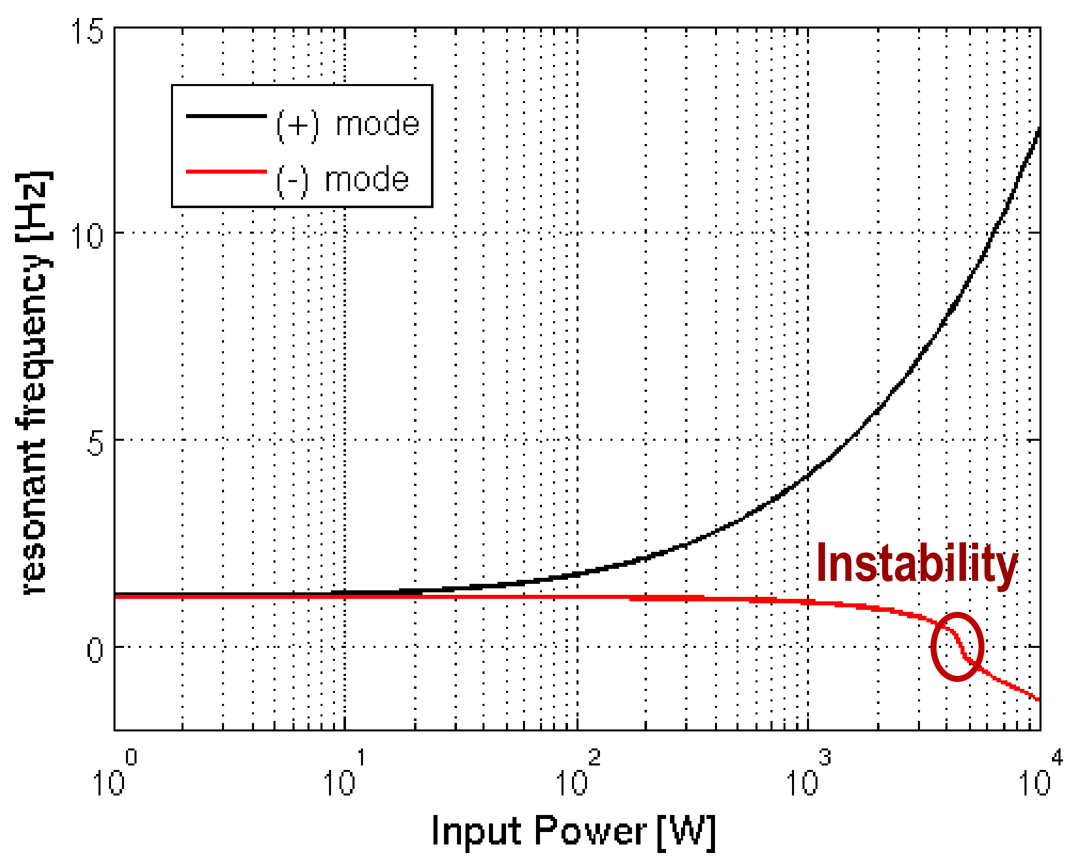
- **High order modes:** fundamental for alignment simulation (at least TEM01)
  - **both in Finesse and Optickle (Optickle up to TEM01)**
- **Pitch and Yaw**
  - **In Finesse while Optickle has only the pitch direction**
- **Mirror static misalingment:** usefull to evaluate the error signal quality (double zeros, etc...)
  - **In Finesse but not in Optickle**
- **Radiation pressure evaluation:** fundamental in high power interferometers
  - **in Optickle but not in Finesse**

Why RP is important?



In an high power cavity the laser beam connects the mirror as an optical spring.

Why RP is important?



Advanced Virgo F-P cavity

$$\omega_+ = \sqrt{\omega_0^2 + \frac{GP_{\mathrm{in}}L}{Ic}\frac{\left(-(g_1+g_2)+\sqrt{4+(g_1-g_2)^2}\right)}{(1-g_1g_2)}}$$
$$\omega_- = \sqrt{\omega_0^2 + \frac{GP_{\mathrm{in}}L}{Ic}\frac{\left(-(g_1+g_2)-\sqrt{4+(g_1-g_2)^2}\right)}{(1-g_1g_2)}}$$

$$\theta_+ = \begin{pmatrix} \frac{2}{1-g_1g_2} \\ \frac{g_1-g_2-\sqrt{4+(g_1-g_2)^2}}{1-g_1g_2} \end{pmatrix}$$
$$\theta_- = \begin{pmatrix} \frac{2}{1-g_1g_2} \\ \frac{g_1-g_2+\sqrt{4+(g_1-g_2)^2}}{1-g_1g_2} \end{pmatrix}$$

The radiation pressure changes the opto-mechanical responses of the cavity.

The (-)-mode becomes softer and the (+)-mode becomes harder

Thus it is fundamental in the design of control schemes for high power interferometers

Optickle

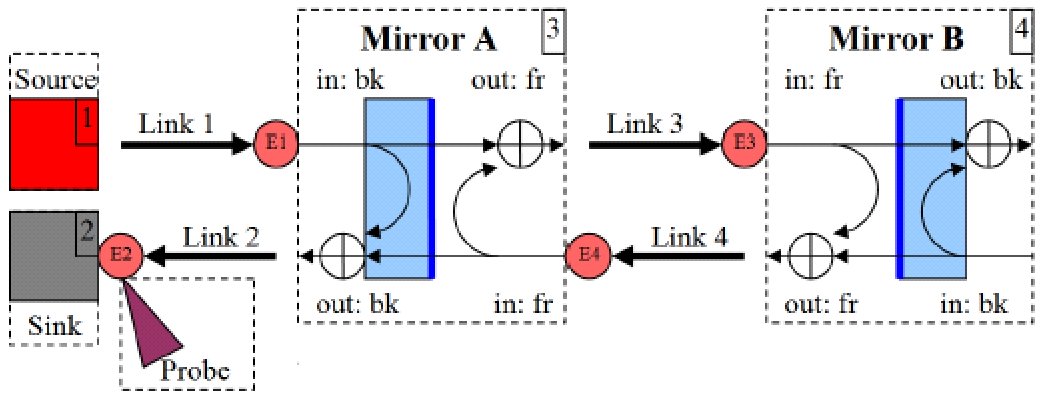
Based on matlab code

http://ilog.ligo-a.caltech.edu:7285/advligo/ISC_Modeling_Software

Optics (similar to finesse)

- BeamSplitter - a beam-splitter
- GouyPhase - an abstract telescope
- Mirror - a general curved mirror
- Modulator - audio frequency phase and amplitude modulation
- RFModulator - radio frequency phase and amplitude modulation
- Telescope - a lense or set of lenses
- Sink - a field sink, used for detectors
- Source - a field source

Connected by links as the Spaces in Finesse



How to model an opto-mechanical system (1-4)

Inizialize the model

$opt = Optickle(vFrf, lambda)$

vFrf	vector of RF component frequencies
lambda	carrier wave length (default 1064 nm)
opt	empty Optickle model

Add the optics

$[opt, sn] = addMirror(opt, name, aio, Chr, Thr, Lhr, Rar, Lmd, Nmd)$

aio	angle of incidence (in degrees)
Chr	curvature of HR surface (Chr = 1 / radius of curvature)
Thr	power transmission of HR surface
Lhr	power loss on reflection from HR surface
Rar	power reflection of AR surface
Nmd	refractive index of medium (1.45 for fused silica, SiO2)
Lmd	power loss in medium (one pass)

$[opt, sn] = addBeamSplitter(opt, name, aio, Chr, Thr, Lhr, Rar, Lmd, Nmd)$

aio	angle of incidence (in degrees)
Chr	curvature of HR surface (Chr = 1 / radius of curvature)
Thr	power transmission of HR surface
Lhr	power loss on reflection from HR surface
Rar	power reflection of AR surface
Nmd	refractive index of medium (1.45 for fused silica, SiO2)
Lmd	power loss in medium (one pass)

Laser, eom, output ports, telescopes...

How to model an opto-mechanical system (2-4)

Linking the optics toghether

$[opt, snLink] = addLink(opt, from, out, to, in, len)$

from	serial number or name of the source optic (field origin)
out	number or name of the output port (e.g., 1, 'fr', etc.)
to	serial number or name of the sink optic (field destination)
in	number or name of the input port (e.g., 2, 'bk', etc.)
len	length of the link

And probes

$[opt, snProbe] = addProbeIn(opt, name, to, in, freq, phase)$

name	name of the new probe
to	serial number or name of the sink optic
in	number or name of the input port
freq	demodulation frequency
phase	demodulation phase (in degrees)

Setting Mechanical TF

$opt = setMechTF(opt, name, mechTF, nDOF)$

name	name of the optic
mechTF	mechanical transfer function (see LTIMODELS)
nDOF	1 for position (default), 2 for pitch

How to model an opto-mechanical system (3-4)

Run the model

$[fDC, sigDC, sigAC, mMech, noiseAC, noiseMech] = tickle(opt, pos, f)$

$[fDC, sigDC, sOpt, noiseOut] = tickle(opt, pos, sCon)$

pos	optic positions (Ndrive x 1 or empty for zeros)
f	vector of evaluation frequencies (empty for DC only)
sCon	control structure from *convertSimulink*
fDC	DC fields (Nlink x Nrf)
sigDC	DC signals for each probe (Nprobe x 1)
sigAC	transfer matrix (Nprobe x Ndrive x Naf)
mMech	modified drive transfer functions (Ndrv x Ndrv x Naf)
noiseAC	quantum noise at each probe (Nprobe x Naf)
noiseMech	quantum noise at each drive (Ndrv x Naf)
sOpt	response structure
noiseOut	quantum noise at each Simulink output

TEM00

$[sigAC, mMech] = tickle01(opt, pos, f)$

$sOpt = tickle01(opt, pos, sCon)$

pos	optic positions (Ndrive x 1 or empty for zeros)
f	vector of evaluation frequencies (empty for DC only)
sCon	control structure from *convertSimulink*
sigAC	TEM01 transfer matrix (Nprobe x Ndrive x Naf)
mMech	modified pitch drive transfer functions (Ndrv x Ndrv x Naf)
sOpt	response structure

TEM01

How to model an opto-mechanical system (4-4)

Getting the optics indexes and the fields

$n = getDriveIndex(opt,\ name,\ driveType)$

name	name of the optic
driveType	name of the drive (e.g., 'pos' or 'amp') (optional)
n	index of this drive (e.g., in sigAC returned from tickle)

$n = getProbeNum(opt,\ name)$

name	name of the probe
n	index of probed (e.g., in sigDC returned from tickle)

} drive and probes indexes

$n = getFieldIn(opt,\ name,\ inName)$

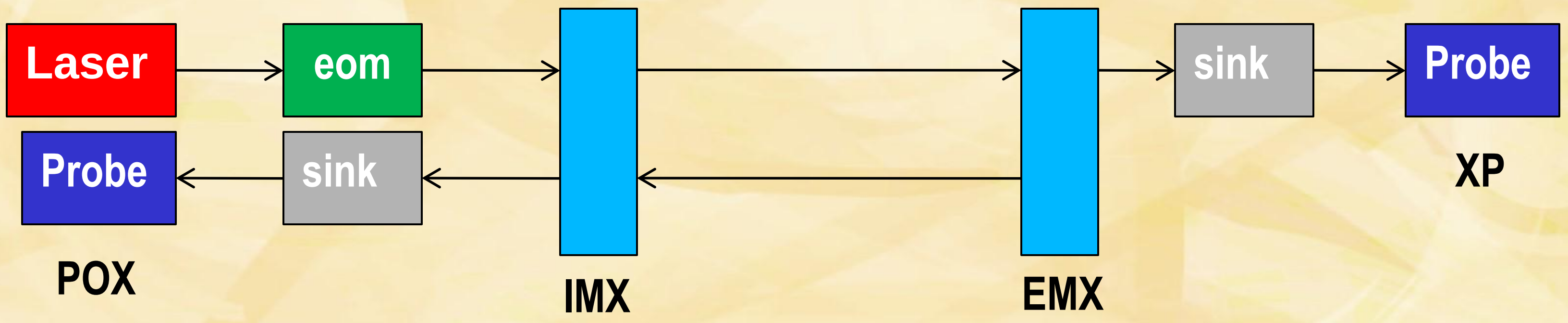
name	name of the optic
inName	name of an input to the optic
n	index of input field (e.g., in fDC returned from tickle)

$n = getFieldProbed(opt,\ name)$

name	name of the probe
n	index of probed field

} getting the fields

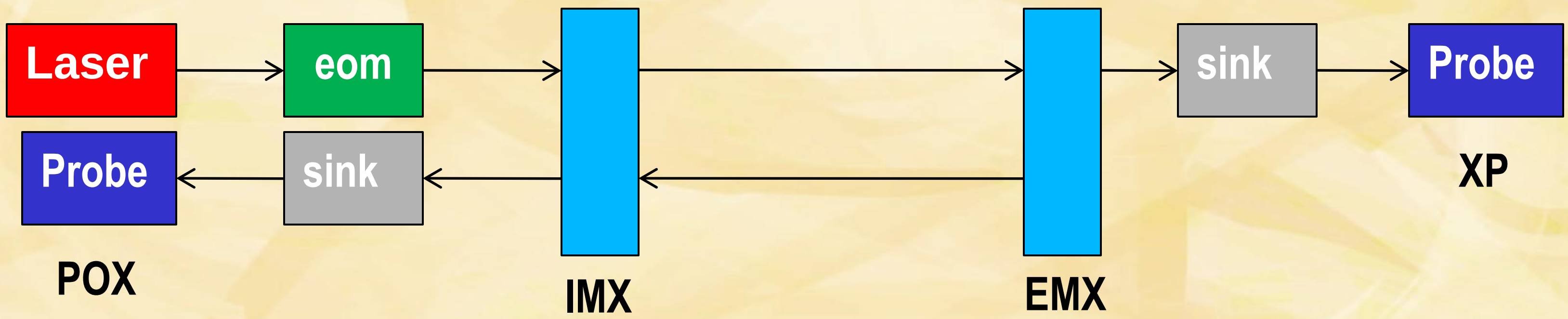
Example – single AdVirgo F-P



Inizialize and build the model

- Define the cavity opto-mechanical parameters (lengths, ROC etc)
- Define the probes parameter (demodulation and Gouy phases)
- Set the optics and link them toghether
- Put the probes

Example – single F-P



Inizialize and build the model

```
% create and run the model
    lambda = 1064e-9;  % laser wavelength
    par.Pin=1;  % input power
    par = param(par);  % optical and mechanical parameters
    par = param_probes(par);  % demodulation and Gouy phase for long and ang diodes
    opt = opt(par);  % optical configuration
    opt = probes(opt, par);  % probes
```

Parameters – `param(par)` (1-2)

Some constants

Laser source

```
function par = param(par);

% basic constants
lambda = 1064e-9;
par.lambda=lambda;
par.c = 299792458;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Input Beam Parameters
f1 = 6270659;  % modulation frequency
Nmod = 1;
% construct modulation vectors
n = (-Nmod:Nmod)';
vFrf = unique([n * f1]);

% input amplitude is just carrier
nCarrier = find(vFrf == 0,1);

vArf = zeros(size(vFrf));
vArf(nCarrier) = sqrt(par.Pin);

par.Laser.vFrf = vFrf;
par.Laser.vArf = vArf;
par.Laser.Power = par.Pin;
par.Laser.Wavelength = lambda;

par.Mod.f1 = f1;
par.Mod.g1 = 0.1;
```

Parameters – `param(par)` (2-2)

Lenghts and ROC

T, L, R

Mechanical parameters

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Detector Geometry (distances in meters)

par.lArmCav =3000;
% Radius of Curvature [m]
par.IMX.ROC = 1416;
par.EMX.ROC = 1646;
% Microscopic length offsets
par.IMX.pos = 0;
par.EMX.pos = 0;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Mirror Parameters

  par.IMX.T = 0.0069;
  par.EMX.T = 5e-6;
  par.IMX.Rar = 100e-6;
  par.IMX.L=0;
  par.EMX.L=100e-6;

% mechanical parameters
par.w = 2 * pi * 0.6; % resonance frequency mirror (rad/s)
par.IMX.mass  = 40;                  % mass mirror (kg)
par.EMX.mass  = 40;

par.pitch = 2 * pi * 1.2;   % pitch mode resonance frequency
rTM = 0.35/2;              % test-mass radius
tTM = 0.2;                 % test-mass thickness
iTM = (3 * rTM^2 + tTM^2) / 12;  % TM moment / mass

par.IMX.I = par.IMX.mass * iTM;
par.EMX.I = par.EMX.mass * iTM;
```

Parameters for the probes – `param_probes(par)`

Dem phase for longitudinal diodes

Dem phase for quadrant diodes

Gouy phases

```
function par = param_probes(par)

% Demodulation phases for longitudinal diodes

par.phi.POX1 = 0;

par.phi.XP1 = 0;

% Demodulation phases for quadrant diodes
% A quadrant

par.phi.POX_A1 = 0;

par.phi.XP_A1 = 0;

% B quadrant

par.phi.POX_B1 = 0;

par.phi.XP_B1 = 0;

%Guoy Phases - in rad

par.gouy.POX =0*pi/180;
par.gouy.XP =0*pi/180;
```

Optical configuration – opt(par) (1-2)

```
function opt = opt(par);

% create an empty model, with frequencies specified
opt = Optickle(par.Laser.vFrf);

% %%%%%%%%%%%%%%%%% source %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% add a source, with RF amplitudes specified
opt = addSource(opt, 'Laser', par.Laser.vArf);

% Modulators
opt = addRFmodulator(opt, 'eom1', par.Mod.f1, i * par.Mod.g1); % 1 * par.Mod.g1 for amp mod

% %%%%%%%%%%%%%%%%% Add Core Optics %%%%%%%%%%%%%%%%%%%%%%%
  [opt, nIMX] = addMirror(opt, 'IMX', 0, 1 / par.IMX.ROC, par.IMX.T, par.IMX.L, par.IMX.Rar);
  [opt, nEMX] = addMirror(opt, 'EMX', 0, 1 / par.EMX.ROC, par.EMX.T, par.EMX.L);

% %%%%%%%%%%%%%%%%% Mech TF %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
dampResA = [1e-3 + 1i, 1e-3 - 1i];
dampResL = [5.4e-8 + 1i, 5.4e-8 - 1i];

opt = setMechTF(opt, 'IMX', zpk([], -par.w * dampResL, 1 / par.IMX.mass)); % longitudinal TF
opt = setMechTF(opt, 'EMX', zpk([], -par.w * dampResL, 1 / par.EMX.mass));

opt = setMechTF(opt, 'IMX', zpk([], -par.pitch * dampResA, 1 / par.IMX.I), 2); % angular TF
opt = setMechTF(opt, 'EMX', zpk([], -par.pitch * dampResA, 1 / par.EMX.I), 2);

opt=setPosOffset(opt,'EMX',par.EMX.pos); % longitudinal position
```

Optical configuration – `opt(par)` (2-2)

```
%%%%%%%%%%%% link the mirror toghether

opt = addLink(opt, 'Laser', 'out', 'eom1', 'in', 0.1);  % I can put it even without space

%%%%%%%%%%%%%%%% X Arm
% link IB to IMX
  opt = addLink(opt, 'eom1', 'out', 'IMX', 'bk', 5);

  opt = addLink(opt, 'IMX', 'fr', 'EMX', 'fr', par.lArmCav );
  opt = addLink(opt, 'EMX', 'fr', 'IMX', 'fr', par.lArmCav);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% tell Optickle to use this cavity basis
opt = setCavityBasis(opt, 'IMX', 'EMX');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Probes

opt = addSink(opt, 'POX1',0.5);
opt = addSink(opt, 'XP1', 0.9839);

% POX
opt = addLink(opt, 'IMX', 'po', 'POX1', 'in', 5);
% XP
opt = addLink(opt, 'EMX', 'bk', 'XP1', 'in', 5);
```

Probes TEM00 – probes(opt,par)

```
function opt = probes(opt, par)

% demodulation frequency
f1 = par.Mod.f1;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

opt = addProbeIn(opt, 'POX DC', 'POX1', 'in',  0, 0);              % DC
opt = addProbeIn(opt, 'POX P1', 'POX1', 'in', f1, par.phi.POX1);   % f1 demod I


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

opt = addProbeIn(opt, 'XP DC', 'XP1', 'in',  0, 0);              % DC
opt = addProbeIn(opt, 'XP P1', 'XP1', 'in', f1, par.phi.XP1);    % f1 demod I

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Longitudinal diodes

Probes TEM01 – probes_01(opt,par)

Angular diodes

```
function opt = probes_01(opt, par)

f1 = par.Mod.f1; % demodulation freq
% POX
[opt, nPOX_A, nPOX_B] = addReadoutGouy(opt, 'POX', par.gouy.POX, 'POX1');
% XP
[opt, nXP_A, nXP_B] = addReadoutGouy(opt, 'XP', par.gouy.XP, 'XP1');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% POX_A & POX_B @ 0, fm1
% Phase Gouy A
opt = addProbeIn(opt, 'POX_A DC', nPOX_A, 'in',  0, 0);                 % DC
opt = addProbeIn(opt, 'POX_A P1', nPOX_A, 'in', f1, par.phi.POX_A1);        % f1 demod I
opt = addProbeIn(opt, 'POX_A Q1', nPOX_A, 'in', f1, par.phi.POX_A1 + 90);  % f1 demod Q
% Phase Gouy B
opt = addProbeIn(opt, 'POX_B DC', nPOX_B, 'in',  0, 0);                 % DC
opt = addProbeIn(opt, 'POX_B P1', nPOX_B, 'in', f1, par.phi.POX_B1);        % f1 demod I
opt = addProbeIn(opt, 'POX_B Q1', nPOX_B, 'in', f1, par.phi.POX_B1 + 90);  % f1 demod Q
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% XP_A & XP_B @ 0, fm1
% Phase Gouy A
opt = addProbeIn(opt, 'XP_A DC', nXP_A, 'in',  0, 0);                 % DC
opt = addProbeIn(opt, 'XP_A P1', nXP_A, 'in', f1, par.phi.XP_A1);        % f1 demod I
opt = addProbeIn(opt, 'XP_A Q1', nXP_A, 'in', f1, par.phi.XP_A1 + 90);  % f1 demod Q
% Phase Gouy B
opt = addProbeIn(opt, 'XP_B DC', nXP_B, 'in',  0, 0);                 % DC
opt = addProbeIn(opt, 'XP_B P1', nXP_B, 'in', f1, par.phi.XP_B1);        % f1 demod I
opt = addProbeIn(opt, 'XP_B Q1', nXP_B, 'in', f1, par.phi.XP_B1 + 90);  % f1 demod Q
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Run the model – few examples

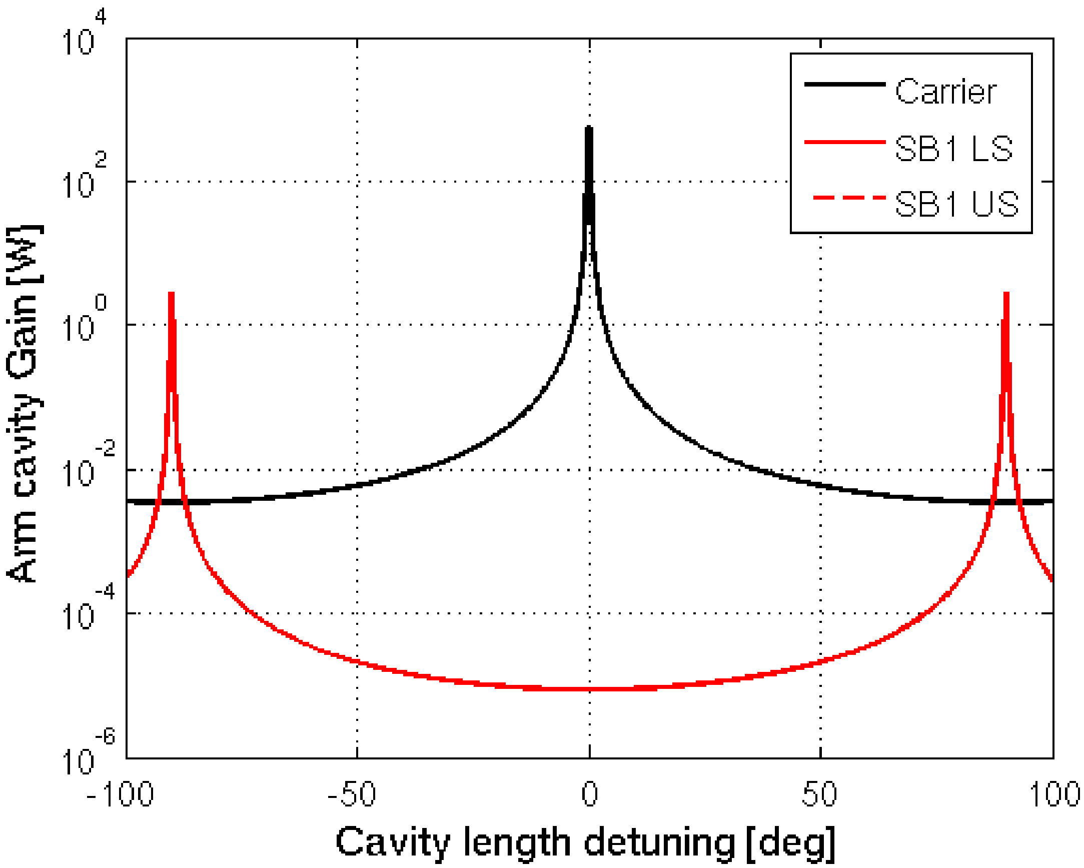
- **Evaluate the resonance conditions**
- **Model the mechanical TF**
  - in low and high power regime
- **Alignment error signals**

Resonance condition

```
% create and run the model
    lambda = 1064e-9;  % laser wavelength
    par.Pin=1;  % input power
    par = param(par);  % optical and mechanical parameters
    par = param_probes(par);  % demodulation and Gouy phase for long and ang diodes
    opt = opt(par);  % optical configuration
    opt = probes(opt, par);  % probes

    % Mirror indexes
nEMX = getDriveIndex(opt, 'EMX');
%%%%%%%%%%%%%%%%%%%%%%%%%%% EMX sweep
pos_start = zeros(opt.Ndrive, 1);
pos_end = zeros(opt.Ndrive, 1);
pos_start(nEMX) = -100/360*lambda;
pos_end(nEMX)   =  100/360*lambda;
N=1000;
[pos, sigDC, fDC] = sweepLinear(opt, pos_start, pos_end, N);

nInside = getFieldOut(opt, 'EMX','fr');
figure()
semilogy((pos(nEMX,:) - mean(pos(nEMX,:)))*360/lambda, abs(squeeze(fDC(nInside, 2, :))).^2/par.Pin*2, 'k','linewidth', 2)
hold on
semilogy((pos(nEMX,:) - mean(pos(nEMX,:)))*360/lambda, abs(squeeze(fDC(nInside, 1, :))).^2/par.Pin*2, 'r','linewidth', 2)
semilogy((pos(nEMX,:) - mean(pos(nEMX,:)))*360/lambda, abs(squeeze(fDC(nInside, 3, :))).^2/par.Pin*2, 'r--','linewidth', 2)
grid on
legend('Carrier','SB1 LS','SB1 US');
xlabel('Cavity length detuning [deg]', 'FontSize', 16);
ylabel('Arm cavity Gain [W]', 'FontSize', 16);
set(get(gcf,'CurrentAxes'),'FontSize',14)
```

Resonance condition

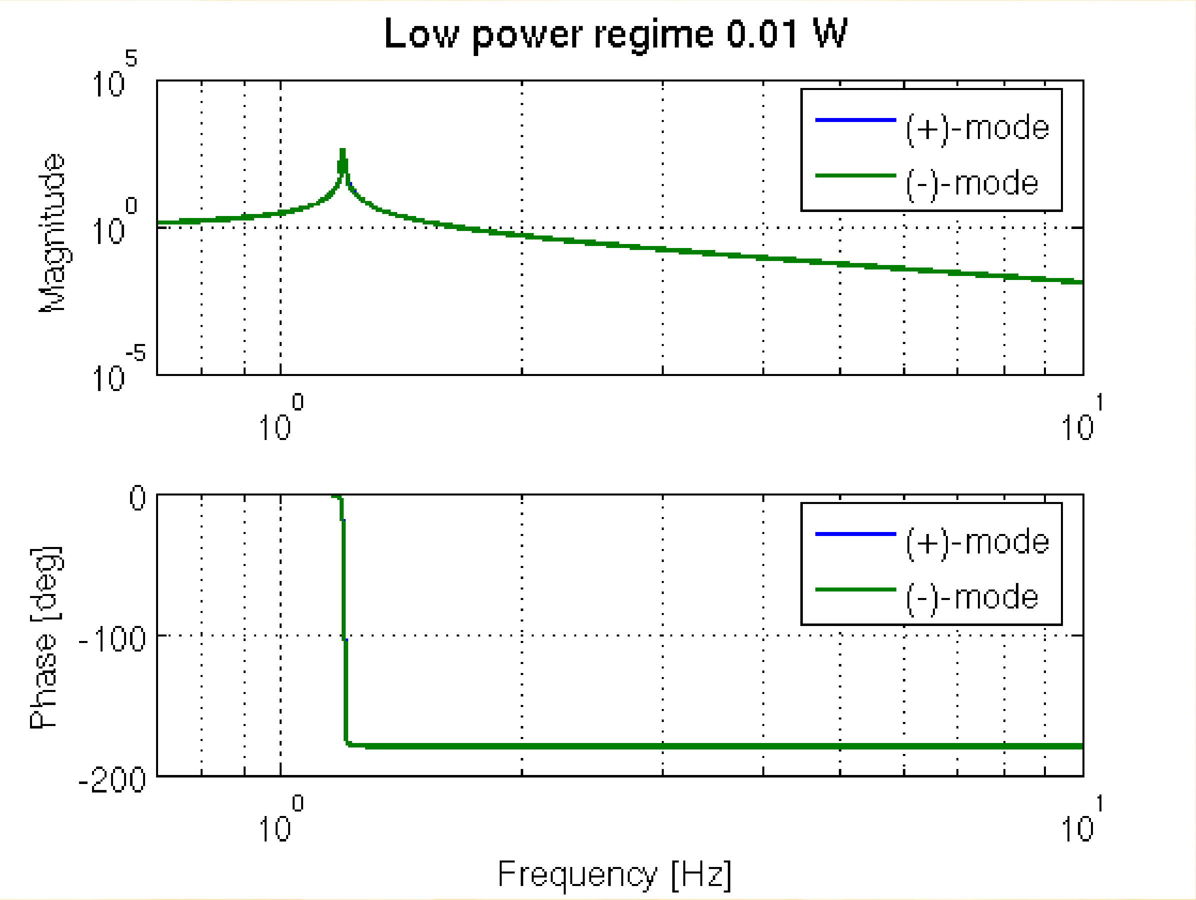
```
% create and run the model
    lambda = 1064e-9;  % laser wavelength
    par.Pin=1;  % input power
    par = param(par);  % optical and mechanical
    par = param_probes(par);  % demodulation an
    opt = opt(par);  % optical configuration
    opt = probes(opt, par);  % probes

    % Mirror indexes
nEMX = getDriveIndex(opt, 'EMX');
%%%%%%%%%%%%%%%%%%%%%%%%%%% EMX sweep
pos_start = zeros(opt.Ndrive, 1);
pos_end = zeros(opt.Ndrive, 1);
pos_start(nEMX) = -100/360*lambda;
pos_end(nEMX)   =  100/360*lambda;
N=1000;
[pos, sigDC, fDC] = sweepLinear(opt, pos_start,

nInside = getFieldOut(opt, 'EMX','fr');
figure()
semilogy((pos(nEMX,:) - mean(pos(nEMX,:)))*360/
hold on
semilogy((pos(nEMX,:) - mean(pos(nEMX,:)))*360/
semilogy((pos(nEMX,:) - mean(pos(nEMX,:)))*360/
grid on
legend('Carrier','SB1 LS','SB1 US');
xlabel('Cavity length detuning [deg]', 'FontSiz
ylabel('Arm cavity Gain [W]', 'FontSize', 16);
set(get(gcf,'CurrentAxes'),'FontSize',14)
```



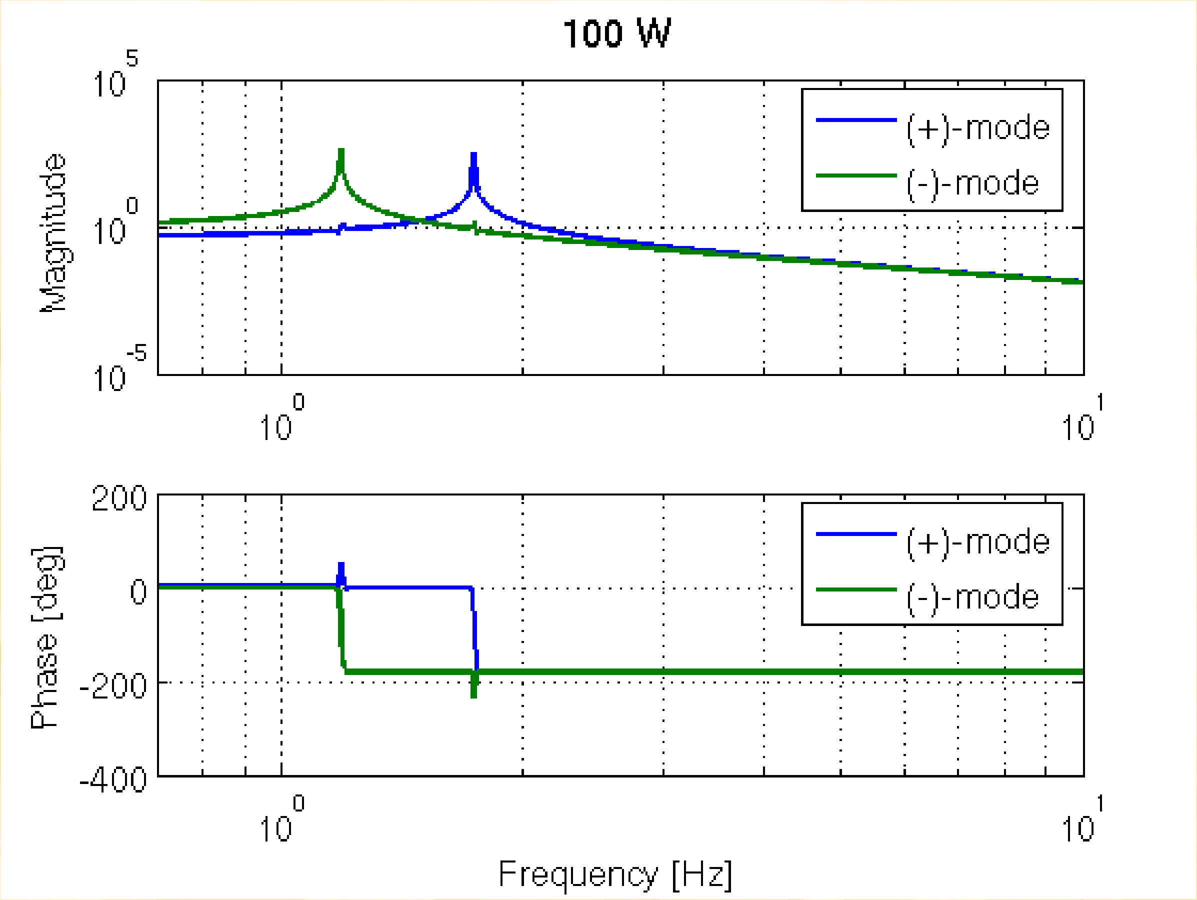
Mechanical TF

```
power=100; % input power [W]
f = logspace(log10(0.01),log10(100),500)'; % frequency vector
f_res=1.2; % pitch res frequency
zg = 1; ng = 1;
[zg,ng] = filtline(zg,ng,1,0,0,f_res,1e3);
[m,p]=bode(zg,ng,2*pi*f); % unperturbed Mech TF

% run the model
    par.Pin=power;
    par = param(par);
    par = param_probes(par);
    opt = opt(par);
    opt = probes(opt, par);

[sigAC, mMech] = tickle01(opt, [], f); % to run the angular TF

nm(1) = getDriveIndex(opt, 'IMX');
nm(2) = getDriveIndex(opt, 'EMX');

% drive matrix
r =0.8557;
drive =       [r      -1; 1     r  ];

% magnitude
MTF(:,1)=( drive(1,1)*getTF(mMech,nm(1),nm(1))+drive(1,2)*getTF(mMech,nm(1), nm(2)) ).*m;   % (+)mode
MTF(:,2)=( drive(2,1)*getTF(mMech,nm(2),nm(1))+drive(2,2)*getTF(mMech,nm(2), nm(2)) ).*m;   % (-)mode
% phase
for k=1:2,
    PTF(:,k)=angle(MTF(:,k))*180/pi+p;
end
```

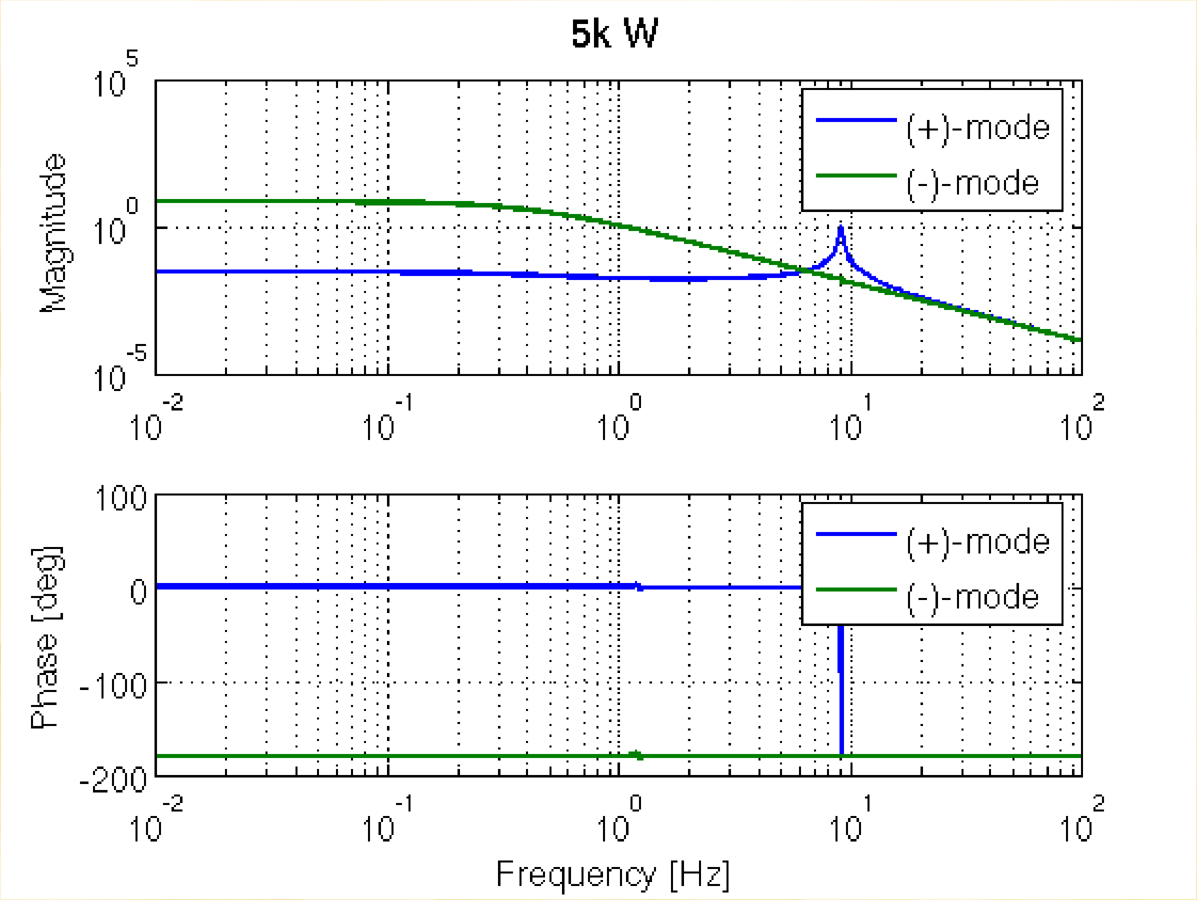
Mechanical TF



Mechanical TF



Mechanical TF

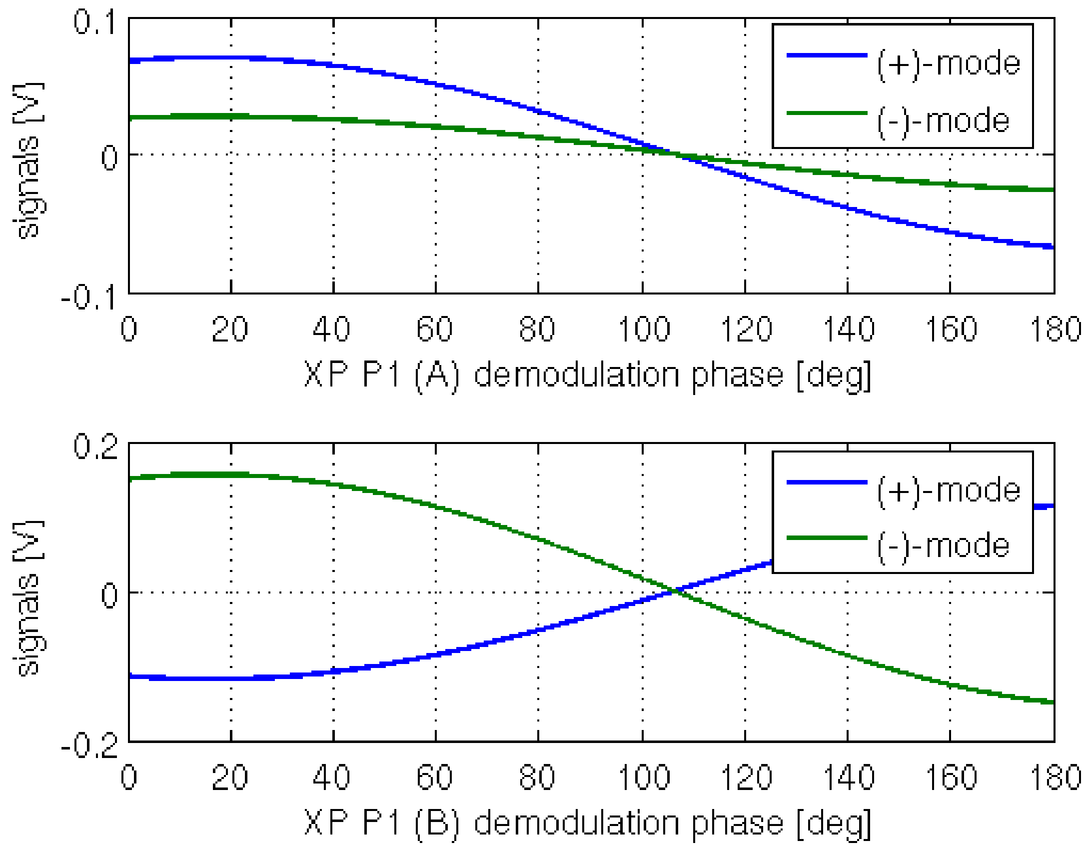


AA error signals

```
dem=linspace(0,180,5000); % demodulation phase
power=135; % input power
r =0.8557;       % define the (+) and (-) d.o.f.
drive =       [r      -1 ;  1     r  ];
f = 1e-10; % frequency

par.Pin=power;
par = param(par);
par = param_probes(par);
opt = opt(par);
opt = probes_01(opt, par);

[sigAC, mMech] = tickle01(opt, [], f); % angular tf
[fDC_l, sigDC_l, sigAC_l, mMech_l, noiseAC_l, noiseMech_l] = tickle(opt, [], f); % longitudinal tf

n(1) = getDriveIndex(opt, 'IMX');
n(2) = getDriveIndex(opt, 'EMX');
dpA = getProbeNum(opt, 'XP_A_P1');
dqA = getProbeNum(opt, 'XP_A_Q1');
dpB = getProbeNum(opt, 'XP_B_P1');
dqB = getProbeNum(opt, 'XP_B_Q1');

for j = 1 : length(dem)
  r = exp(-i * pi * dem(j) / 180);
  for u=1:length(n)
    sa = real(sigAC(dpA, n(u))) + i * real(sigAC(dqA, n(u)));
    MA(j,u)=real(sa * r);
    sb = real(sigAC(dpB, n(u))) + i * real(sigAC(dqB, n(u)));
    MB(j,u)=real(sb * r);
  end

Ma(j,1)=real(drive(1,1)*MA(j,1)+drive(1,2)*MA(j,2));   % (+)
Ma(j,2)=real(drive(2,1)*MA(j,1)+drive(2,2)*MA(j,2));   % (-)
Mb(j,1)=real(drive(1,1)*MB(j,1)+drive(1,2)*MB(j,2));   % (+)
Mb(j,2)=real(drive(2,1)*MB(j,1)+drive(2,2)*MB(j,2));   % (-)
end
```

AA error signals

```
dem=lins
power=13
r =0.855
drive =
f = 1e-1

par.Pin=
par = pa
par = pa
opt = op
opt = pr

[sigAC,
[fDC_l,

n(1) = g
n(2) = g
dpA = ge
dqA = ge
dpB = ge
dqB = ge

for j =
   r = ex
   for u=
      sa =
      MA(j
      sb =
      MB(j
   end
```



nal tf

```
Ma(j,1)=real(drive(1,1)*MA(j,1)+drive(1,2)*MA(j,2));    % (+)
Ma(j,2)=real(drive(2,1)*MA(j,1)+drive(2,2)*MA(j,2));    % (-)
Mb(j,1)=real(drive(1,1)*MB(j,1)+drive(1,2)*MB(j,2));    % (+)
Mb(j,2)=real(drive(2,1)*MB(j,1)+drive(2,2)*MB(j,2));    % (-)
end
```

Conclusions

- **Optickle can be used to model and design the control scheme**
- **For the control noise simulation `Pickle` will be available soon**