LASER INTERFEROMETER GRAVITATIONAL WAVE OBSERVATORY

LIGO Laboratory / LIGO Scientific Collaboration

LIGO-T070039-v6 *LIGO* April 27, 2011

SIS (Stationary Interferometer Simulation) manual

Hiroaki Yamamoto

Distribution of this document:
LIGO Science Collaboration

This is an internal working note
of the LIGO Project.

California Institute of Technology
LIGO Project – MS 18-34
1200 E. California Blvd.
Pasadena, CA 91125
Phone (626) 395-2129
Fax (626) 304-9834
E-mail: info@ligo.caltech.edu

Massachusetts Institute of Technology
LIGO Project – NW17-161
175 Albany St
Cambridge, MA 02139
Phone (617) 253-4824
Fax (617) 253-7014
E-mail: info@ligo.mit.edu

LIGO Hanford Observatory
P.O. Box 1970
Mail Stop S9-02
Richland WA 99352
Phone 509-372-8106
Fax 509-372-8137

LIGO Livingston Observatory
P.O. Box 940
Livingston, LA 70754
Phone 225-686-3100
Fax 225-686-7189

http://www.ligo.caltech.edu/

1 Introduction

This is a document about "*Stationary Interferometer Simulation*", reffered to as SIS hereafter, a simulation package to calculate fields in a stationary interferometer of various configurations.

This software was developed to help the design of the advanced LIGO Interferometer. The optical configuration of AdvLIGO is a dual recycled Michelson cavity with two long FP cavity arms. The Michelson cavity may be a simple short cavity, which is intrinsically unstable, or may contain a mode matching telescope to make the cavity stable.

This simulation package simulates various optical configurations. As of this writing, only those written in italic have been implemented.

- *FP arm*
- *Coupled cavity consisted of a marginally stable Michelson cavity and a FP arm*
- *Coupled cavity consisted of a stable Michelson cavity and a FP arm*
- ~~Cavity formed by two recycling cavities~~
- Power recycled interferometer
- Dual recycled interferometer

The details of the optics can be included in the simulation, including sizes and wedge angles, surface figures and aberrations and thermal deformations. AOS components, like TCS and baffles, can be included. Any measured data in a common format can be used in the simulation without any prior conversion to a specific format.

The interaction of fields with optics is calculated in the spatial domain. The plane of the interaction is split into grid points, and the grid size, typically a few mm, determines the spatial resolution. The propagation of fields is calculated by using the Maxwell's equation for a plane wave with paraxial approximation. For the conversion of fields between the two expressions, the spatial distribution and the superposition of plane waves with discrete transverse frequencies, is done using FFT. This method is slow compared to other methods, like the one based on the modal model, but details of interferometer performance can be simulated with little compromise.

For the locking of a resonator, the Pound-Drever-Hall signal normalized by the cavity power is used. This method is more closer to the real experimental method than the method to maximize the cavity power or the method to force the round trip phase of the cavity field to be 0. This makes simulation results more close to real measurements and the simulation results can be used to analyze the data directly. When there is more than one cavity in the system, each cavity is locked repeatedly one after another to get to the stationary fields in the entire system.

This document is consisted of two major sections. The first section, "SIS Users' manual", explains how to use the program. The second section, "Physics in SIS", explains details how the physics is implemented in the code.

I. SIS Users' manual

2 Quick Start

2.1 Introduction

The program can be started in a terminal window by typing the program name, sis, with possible optional parameters. The next section shows how to interact with the program.

Black texts are prompts asking your input, green texts are results of your request. Red texts are the ones you type, and blue texts are explanations. Lines marked as "…" are lines omitted.

2.2 Running…

```
SIS> Choose configuration to simulation
SIS>       BS         FP         FPBS       MSCC       SCC        cancel
SIS> Select 1 item(s)
SIS> Type "name" to choose item(s)  >> FP
                     % choose FP to simulate
=> No default database available
SIS> next action
                     % main menu to chooseactions
SIS>    lock           calcField       signalGen       timeTrace       delL
SIS>    modeAmp        saveField       mirrorInfo      storeMap        simSpec
SIS>    runSpec        summary         help            exit
SIS> Select 1 item(s)
SIS> Type "name" to choose item(s)  >> sim
                     % change simulation specification
                     % To choose a item from a list, a string which can uniquely
                     identify the desired item
PARAM> ITM
                     % print characteristics of ITM
ITM.aperture = 0.34
ITM.thickness = 0.2
ITM.Wfft = 0
ITM.mech.x = 0
ITM.mech.y = 0
ITM.mech.z = 0
ITM.mech.tX = 0
ITM.mech.tY = 0
ITM.mech.tZ = 0
ITM.opt.T = 0.005
ITM.opt.R = 0.995
ITM.opt.ROC = 2076
ITM.opt.refrIndex = 1.44963
ITM.opt.HR_phase = (null)
ITM.opt.trans_phase = (null)
ITM.opt.trans_loss = (null)
ITM.opt.AR_trans = (null)
ITM.oscillation.amplitude = (null)
ITM.oscillation.phase = 0
PARAM> ITM.mech.tY = 1e-7  % set yaw to be 1e-7. To set pitch, tZ=PI/2, tY=picth
PARAM> exit  % end chaning setup
```

```
SIS> next action
SIS>    lock            calcField       signalGen       timeTrace       delL
SIS>    modeAmp         saveField       mirrorInfo      storeMap        simSpec
SIS>    runSpec         summary         help            exit
SIS> Select 1 item(s)
SIS> Type "name" to choose item(s)  >> lock   % lock FP cavity
......          % locking status information
......
    SISFP + ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
              locked, exiting...
                  power = 793.01435427262, loss = 15.688590934686 ppm
                  diffraction loss = 0.37378789541354 ppm
             ----------------------------------------------------------
lock succeeded
SIS> next action
SIS>    lock            calcField       signalGen       timeTrace       delL
SIS>    modeAmp         saveField       mirrorInfo      storeMap        simSpec
SIS>    runSpec         summary         help            exit
SIS> Select 1 item(s)
SIS> Type "name" to choose item(s)  >> mode  % modal analysis
SIS> Select a field to be saved in a file
SIS>    0.inputField        1.fromITM           2.toETM             3.fromETM
SIS>    4.toITM             5.promptRefl        6.totalRefl         all
SIS>    exit
SIS> Select 1 item(s)
SIS> Type "name" to choose item(s)  >> 1  % choose “1.fromITM”
+++ Field "fromITM" +++
Mode base  : z = -1997.71       z0 = 395.4753
           : w = 0.05959586     R = -2076
% field is fit by TEM00 function with Rx, Ry, wx, wy, x0, y0 are free parameters
% when Rx = Ry or wx = wy, only one value is printed.
% power / 00 are the total power and 00 mode power
Fit result : w = 0.0595937      R = -2075.98       x0 = -0.002538406   y0 = 0
             power / 00 = 793.0144 / 791.3287
SIS> Which function set to use for mode expansion
SIS>       LaguerreGauss       HermiteGauss        nextField           exit
SIS> Select 1 item(s)
SIS> Type "name" to choose item(s)  >> H   % hermite gauss expansion
SIS> Max mode for expansion (def=7,[1:INF])  >> 2  % up to n+m = 2
+++ Field "fromITM" +++

 Amplitude = ( Re, Im ) [ subPower, fraction ]
% HG(m,n) mode amplitude, power and fraction of this mode power out of total
HG( 0 , 0 ) = (28.130565,1.3060768e-05)   [ 791.32870, 0.997874 ]
HG( 1 , 0 ) = (-1.19819158,-0.49803912)   [ 1.683706, 0.00212317 ]
HG( 0 , 1 ) = (-7.4928640e-16,-2.73970578e-16)   [ 6.364900e-31, 8.026210e-34 ]
HG( 2 , 0 ) = (0.0299262,0.03005437)      [ 0.0017988, 2.2683627e-06 ]
HG( 1 , 1 ) = (4.8918034e-16,3.94997e-17) [ 2.408576e-31, 3.0372418e-34 ]
HG( 0 , 2 ) = (-1.07930389e-05,3.5727782e-05)   [ 1.392964e-09, 1.756543e-12 ]

Total power = 793.01435 and power fraction of higher modes is 1.8707537e-07

SIS> next action
```

```
SIS>    lock            calcField       signalGen       timeTrace       delL
SIS>    modeAmp         saveField       mirrorInfo      storeMap        simSpec
SIS>    runSpec         summary         help            exit
SIS> Select 1 item(s)
SIS> Type "name" to choose item(s)  >> save   % save fields for detail abalysis
SIS> Select a field to be saved in a file
SIS>    0.inputField        1.fromITM           2.toETM             3.fromETM
SIS>    4.toITM             5.promptRefl        6.totalRefl         all
SIS>    exit
SIS> Select 1 item(s)
SIS> Type "name" to choose item(s)  >> 1  %1.fromITM chosen
SIS> Output file name (def="fromITM.dat")  >> [CR] % to choose default
SIS> File format. (def=2,[0:INF])  >> ?   % ask for help
SIS> 0: re im in 2NxN, 1: x y re im in 4xN, 2: Binary, 10: freq, 12: FreqBinary
SIS> File format. (def=2,[0:INF])  >> 2  % this is recommended
Data saved in fromITM.dat with N = 256 in window = 0.7
SIS> next action
SIS>    lock            calcField       signalGen       timeTrace       delL
SIS>    modeAmp         saveField       mirrorInfo      storeMap        simSpec
SIS>    runSpec         summary         help            exit
SIS> Select 1 item(s)
SIS> Type "name" to choose item(s)  >> exit  % done, quit
[Macintosh-4:~/Desktop] hiro%
```

2.3 Analysing…

A matlab function, *sisin*, is defined in sisbin.m to load the data stored in a binary format. The usage is

[ITM, xlist] = sisbin(‘fromITM.dat’);

ITM is a matrix of the field amplitude, and xlist is a one dimensional vector of grid positions along one side. ITM(iy, ix) is the complex amplitude at x = xlist(ix), y = xlist(iy).

A few examples of analysing the field will be in order. The beam size, field ROC and HGmn amplitudes are given as the result of the mode analysis given above.

- ITM power vs exp(-2 r^2 / 6cm^2)

```
>> semilogy( xlist(129:end).^2, abs(ITM(129:end,128)).^2, 'x', xlist(129:end).^2,
abs(ITM(129,129))^2*exp(-2*xlist(129:end).^2/0.0595937^2))
```

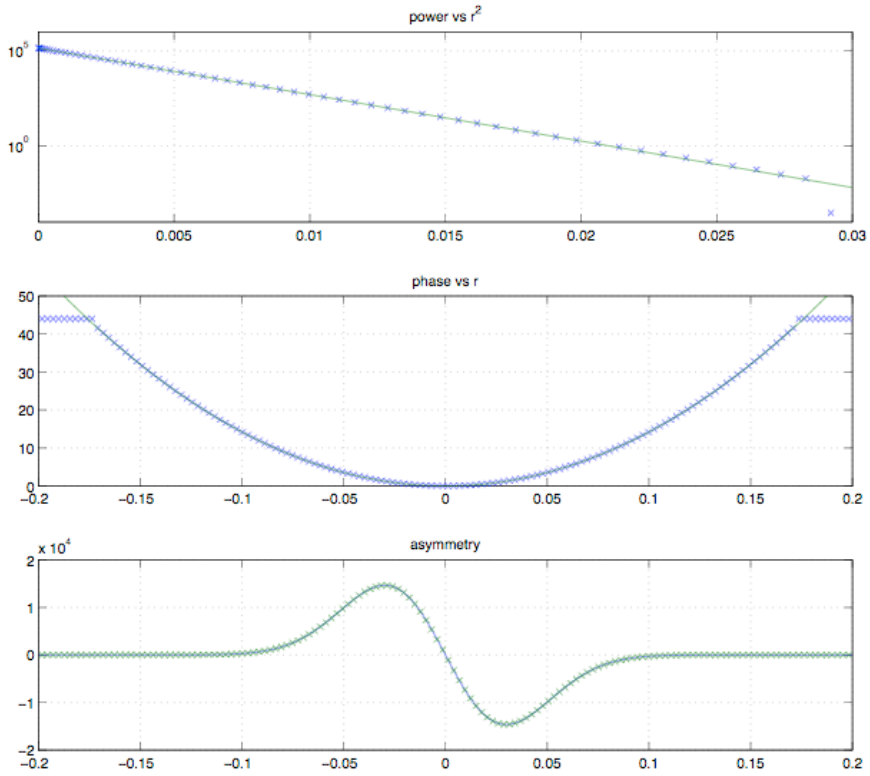
- ITM phase vs k r^2 / 2 R

```
>> plot( xlist, unwrap( angle(ITM(:,128)) )+44, 'x', xlist, xlist.^2*2*pi/1.064e-
6/(2*2075.982) )
```

- ITM asymmetry vs asym of (HG00 + HG01) exp(-2 r^2/w^2)

```
>> ydist = (28.130565245246+i*1.306076480155e-05)+xlist*sqrt(2)/0.0595937*(-
1.1981915798971-i*0.49803914931392)/sqrt(2)*2;

>> plot( xlist, (abs(ydist).^2 - abs(ydist(end:-1:1)).^2).*exp(-
2*xlist.^2/0.0595937^2)/0.0595937^2*(2/pi), xlist, (abs(ITM(129,:)).^2-abs(ITM(129,end:-
1:1)).^2), 'x' )
```



2.4 Real life example – thermal effect and ring heater

In the real life use of SIS, two input files are used. One contains information about the system to simulate, like ITM ROC and transmittance, and another contains a sequence of commands you would manually type in. These two files are explained in the following section.

A FP cavity with thermal deformation is used to explain how to use these files. When you have sisDB_FP.mcr and tcs.in, you can simulate the thermal effect and the use of the ring heater by just typing @tcs.in.

2.4.1 Database file

In the previous example, the IFO configuration was setup interactively. When a complex system is analyzed, it is tedious if the setup should be done each time. All configuration setups, like ITM.opt.ROC = 1971 can be saved in a file and can be loaded when the program starts up.

The default name of the database file is sisDB_FP.mcr for FP, sisDB_SCC.mcr for a stable coupled cavity and sisDB_MCC.mcr for a marginally stable coupled cavity.

You can specify the database filename when you run the program, like

sis –db myDatabase.dat

and you can override definitions using –par runtime directive, like

sis [-db databaseName] –par varName = val

Then, after the database is loaded, the value of the variable named "varName" is changed to val. So, you can run the program with a slightly modified configuration without changing the database each time for minor change.

If the working directory has similar database files, lile, sisDB_FP.mcr and sisDB_FP_advLIGO.mcr, then you are prompted to choose one, if you had not defined the database file name using –db runtime option.

2.4.2 Command sequence input file

When you have to type same commands again and again, you can use this input file to simplify this repetitive task. For example, when you want to analyze a system with different conditions, you will type in same commands after setting specific parameters (lock the cavity, calculate field curvatures, and save data).

"SIS User Interface" section below explains more about this. Usually, you store your command sequence using "@(fileName" and "@)", edit the file content, and playback the command sequence using "@fileName". You can include @fileName token in another sequence input file.

2.4.3 A example simulating thermal effect and ring heater

sisDB_FP.mcr : a file defining FP with thermal effect and ring heater

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

% "%" is a comment tag, the rest of the line is discarded.
% #print can be sued to print a message or values of variables

#print "FP with thermal effect and ring heater"

% variables defined in this default sections can be overriden by the runtime option,
%      -par var = name
% variables can be redefined during the run, var = name and
% string expressions, like ITM.opt.HR_phase, use that new value when needed.

#defaults

% ring heater correction
RINV_ITM_RH = 0
RINV_ETM_RH = 0

#enddef

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

% Basic FFT setup
Nfft = 256        % number of grid points
Wfft = 0.70       % FFT window size, if not otherwise specified

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

```
% ITM and ETM specificaction
% mirror definition = [ aperture, thickness, mech, opt ]
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

aperture = 0.34
ITM.aperture = aperture
ETM.aperture = aperture

% mirror mechanical data = [ x, y, z, tX, tY, tZ ]
% definitions of the coordinate system is defined in the manual.
% [tX,tY,tZ] : pitch rotation = [0, -pitch,PI/2], yaw rotation = [0, yaw, 0]
ITM.mech = [ 0, 0, 0, 0, 0, 0 ]
ETM.mech = [ 0, 0, 0, 0, 0, 0 ]

%-------------------------------------------------------------------------------
% mirror optoical data = [ T, R, ROC, refrIndex, "HR_phasemap", "trans_phasemap" ]
%
% THERMOELASTIC( beamSize, Psubs, Pcoat [, T0] ) for surface deformation
% THERMALPHASE( beamSize, Psubs, Pcoat [, T0] ) for thermal lens in substrate
%    values in ( ) below are typical values for advLIGO arm
%
%    beamSize : gaussian beam width on the mirror,
%               if negative, the resonating field width is used
%    Psubs : power absorbed in substrate (0.084W),
%            if negative, it is used as an absorption rate ( (0.02ppm/cm) *100 m^-1)
%    Pcoat : power absorbed in coating (0.425W),
%            if negative, it is an absorption rate (0.5E-6)
%-------------------------------------------------------------------------------

ROC_ITM = 1971                  % cold state ITM ROC
ROC_ETM = 2191                  % cold state ETM ROC

T_ITM = 0.014             % ITM power transmittance
Pcav = 650e3              % typical advLIGO arm power
Pin = Pcav * T_ITM / 4    % calculate the input power for this FP
wITM = 0.055              % typical beam size on ITM
wETM = 0.062              % typical beam size on ETM

PsubsRate = 0.02e-6 * 100  % substrate abosrption
PcoatRate = 0.5e-6              % coating absorption

armLoss = 75e-6                 % typical arm loss, modstly scattering loss, expected
for advLIGO.

ITM.opt.T = T_ITM
ITM.opt.R = 1 - ITM.opt.T - armLoss/2
ITM.opt.ROC = ROC_ITM

% The thermal effect changes the surface figure
% A ring heater bents the mirror, so the figure changes like r^2 / 2 * (1/R)
% variables x, y, z, rr = x^2 + y^ and r = sqrt(rr) can be used on the right hand side
ITM.opt.HR_phase = THERMOELASTIC( -wITM, -PsubsRate, -PcoatRate ) + rr / 2 * RINV_ITM_RH
% and the transmission phase
```

```
ITM.opt.trans_phase = THERMALPHASE( -wITM, -PsubsRate, -PcoatRate )

ETM.opt.T = 5e-6
ETM.opt.R = 1 - ETM.opt.T - armLoss/2
ETM.opt.ROC = ROC_ETM
% ETM surface figure change by thermal deformation
ETM.opt.HR_phase = THERMOELASTIC( -wETM, 0, -PcoatRate ) + rr / 2 * RINV_ETM_RH

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Definition of the cavity
% cavity definition =
%    [ L0, delL, matchToInput, Nfft, Ncut1, Ncut2, lockFP, adaptiveGrid ]
%
% L0 is macro length, rounted to integer multiple of NdYAG wave length
% delL is a microscopic correct
% total length = lambda(NdYAG) * floor( L0 / lambda ) + delL
% set matchToInput to 1 or 3 to automatically set delL to mode match to the input beam

FPcav = [ 3995.420, 0, 3, 0, 0, 0, 1, 0 ]

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% input beam specification
% inputBeam definition =
%   [ "BeamType", power, index1, index2, waistSize, waistPosition, matchToCavity ]
%
% input type = "LG" for LaguerreGuass( index1, index2 )
%              "HG" for HermiteGauss( index1, index2 )
%
% waist size and position are calculated when you use matchToCavity = 1.
% You can setup the input mode by hand.
% Then you need to set matchToCavity = 0 so that no automatic adjustment is done

inputBeam.beamType = "HG"
inputBeam.power = Pin
inputBeam.index1 = 0
inputBeam.index2 = 0
inputBeam.waistSize = 0
inputBeam.waistPosition = 0
inputBeam.matchToCavity = 1
```

tcs.in : FP simulation main command input

```
% lock and save info
@tr1.in
% save fields and ITM surface data for the cold state
@saveData.in
% calculation FP state 5 times to get stable
% @RBEG N means to run the commands between @RBEG and @REND N times
@RBEG 5
@tr1.in
@REND
% save fields and ITM surface data after lock without ring heater
```

```
@saveData.in
% setup the ring heater to make the combined curvature to be the cold state curvature
sim
RINV_ITM_RH = 1/ROC_ITM - 1/HOTROC_ITM
RINV_ETM_RH = 1/ROC_ETM - 1/HOTROC_ETM
exit
% run 5 times with the ring heater
@RBEG 5
@tr1.in
@REND
% save fields and ITM surface data after lock with ring heater
@saveData.in
% exit the run
exit
```

tr1.in : lock and save info

```
% commands to lock the cavity and save useful information
lock
% calculate parameters
sim
HOTROC_ITM = FLDROC(4)   % ROC of the field going to ITM
HOTROC_ETM = FLDROC(2)   % ROC of the field going to ETM
HOTWZ_ITM = FLDWZ(4)     % beam size of the field going to ITM
HOTWZ_ETM = FLDWZ(2)     % beam size of the field going to ETM
exit
% save values in a file
@PRINT log.dat << HOTROC_ITM HOTWZ_ITM HOTROC_ETM HOTWZ_ETM FPcav._power FPcav._cavLeng
FPcav._totalLoss FPcav._diffLoss
```

saveData.in : save field and mirror data in files

```
% saveField is a command to save field data in a file
saveField
% save all fields
all
% directory name where the data files are stored.
% See the section in the SIS manual "Variable file names with @'s and *'s"
FieldData*
% data format in the file
2
% storeMap stores the mirror data
storeMap
% store ITM map
0.ITM
% store HR_phase
HRsurface
% file name of the data
ITMData*.dat
% data format in the flle
0
```

2.4.4 Running the example

Place these files, sisDB_FP.mcr, tcs.in, tr1.in and saveData.in in a directory, run the sis program, and type "FP". The content of sisDB_FP.mcr is loaded, and you will see output of print commands in the definition file.

```
SIS>
SIS> Choose configuration to simulation
SIS>
SIS>       BS         FP         FPBS       MSCC       SCC        cancel
SIS>
SIS> Select 1 item(s)
SIS> Type "name" to choose item(s)  >> FP
* FP with thermal effect and ring heater
SIS>
SIS> next action
SIS>
SIS>       lock            calcField       signalGen       timeTrace       delL
SIS>       modeAmp         saveField       mirrorInfo      storeMap        simSpec
SIS>       runSpec         summary         help            exit
SIS>
SIS> Select 1 item(s)
SIS> Type "name" to choose item(s)  >> @tcs.in
```

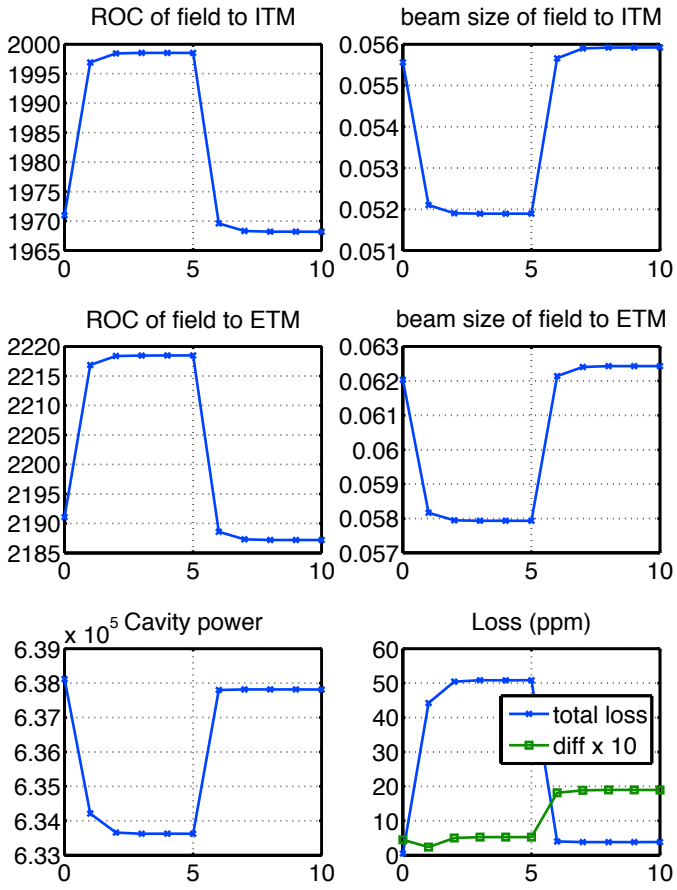
Then you type @tcs.in. Then, the content in tcs.in is executed just as you type them by hand. At the end, you will see several files and folders created.

- log.dat : data saved by tr1.in after each lock
- ITMDataN.dat : ITM HR_phase data, N=0 for cold state, N=1 for hot without ring hater and N=2 for hot with ring heater
- FieldDataN : this folder contains all fields in the simulation. Load data using sisbin as [fld, xy] = sisbin('filename'). Then fld(iy,ix) is the complex amplitude at (x,y) = (xy(ix), xy(iy)).

2.4.5 Some analysis of this run

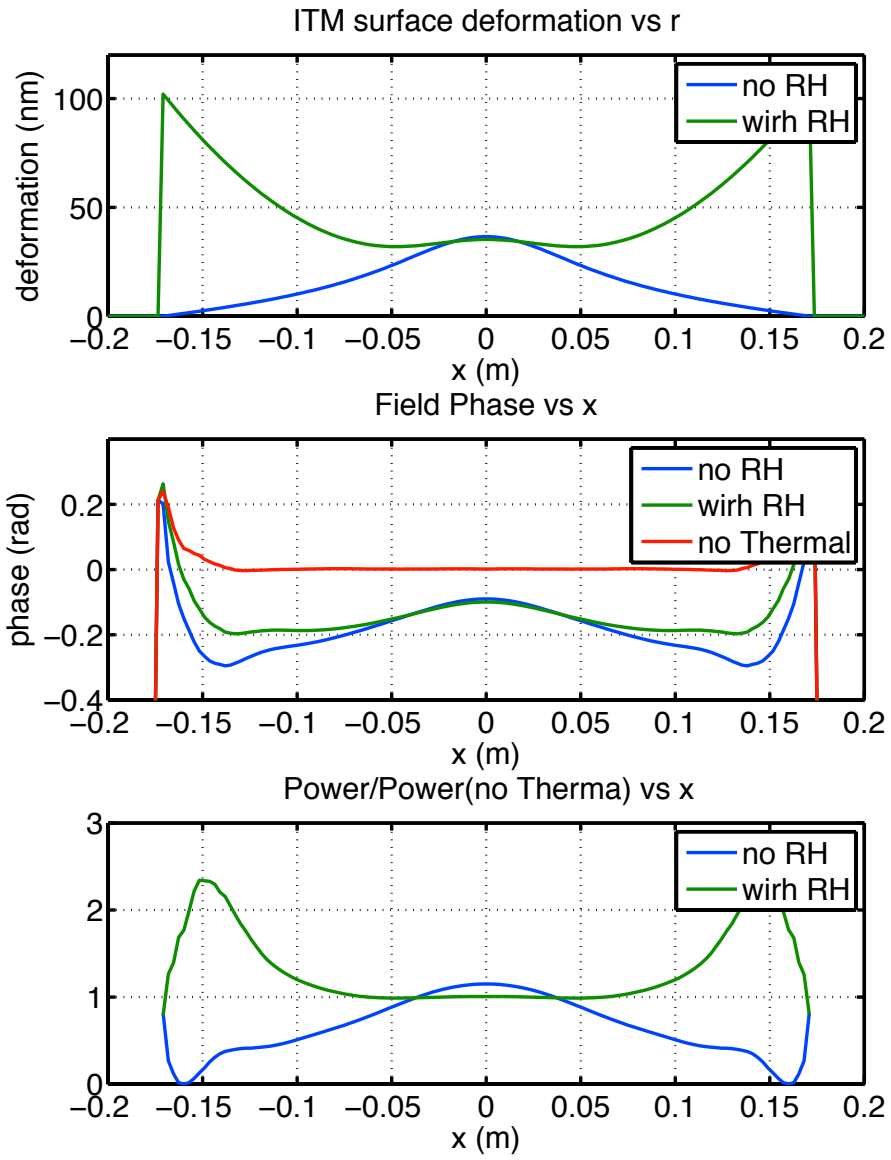
The effect of the thermal deformation goes as follows. At each run, the fields calculated in the previous run is used to calculate the thermal deformation of optics. This process is repeated until the power in the new run is the same as the previous run.

In this example, the first result corresponds to the cold state, following 5 runs are the ones to find the consistent answer without ring heater effect, and the last 5 runs are the one to find the consistent answer with ring heater effects. From the result shown below, it takes only a few repetition is needed to get the consistent answer.



This is a plot based on the data in log.dat. Values at x axis 0 corresponds to a cold state, x = 5 corresponds to a heated state without ring heater and the last point to a heated sate with ring hearter. The ring heater correction is calculated in a simple minded way, but it is observed that the beam profiles on each mirror are restored to those at the cold state. The total loss goes does as well, which shows the restoring of the mode matching of the input beam and the resonating field in the cavity.

The fields with ring heater correction are close to fields at the cold state, but are not quite the same. The data, ITMData and FieldData are analyzed to see the details.



The top plot is the surface deformation of ITM. The blue line is the thermal deformation and the green one is the sum of the thermal effect and the ring heater correct. At the cold state, this is simply flat. So the ring heater cannot nullify the thermal effect.

The second plot is the phase of the field going to ITM, and the bottom is the power of the field going to ITM normalized by the power of the same field in the cold state. Green lines, hot state with ring heater correction, are close to flat, cold state field, than blues ones, hot state without ring heater.

3 Main menu

```
lock        calcField   signalGen   timeTrace   telescope
delL        modeAmp     saveField   mirrorInfo  storeMap
summary     simSpec     runSpec     help        exit
```

This is the main dispatcher to choose the next action.

```
lock       : Lock the cavity
calcField  : Calculate stationary field
signalGen  : Generate audio signal by sinusoidal motion of mirrors
timeTrace  : Move mirror and save field evolution
delL       : Print and set the cavity length
modeAmp    : Decompose a field by LG or HG
saveField  : Save field in a file
mirrorInfo : View mirror information
storeMap   : Store mirror maps
simSpec    : Set simulation parameters
runSpec    : Set run conditions, like convergence criteria
summary    : Print summary status
help       : main help
exit       : Exit this process
```

3.1 Lock

This command initiates the locking process of the cavity by adjusting independent length degree of freedom. After lock is done, fields can be viewed in various ways using modeAmp and saveField.

3.2 calcField

Stationary fields in a cavity is calculated for the given setups of optics and the input field. Afer the fields are calculated, fields can be viewed in various ways using modeAmp and saveField.

3.3 signalGen

This command is used to generate and analyze signal sidebands generated by motions of mirrors. Before this command is issued, either *lock* or *calcField* is be used to calculate the stationary fields. The signal sideband is generated by small motions of mirrors interacting with the stationary fields.

The field motion is specified by mirrorObj.oscillation.amplitude and mirrorObj.oscillation.phase. The mirror motion in the z direction (perpendicular to the mirror surface with HR coating) is calculated by the amplitude as

$$d(x,y) = \sin(\Omega \cdot t) \cdot amp(x,y)$$

where Ω is the signal sideband frequency. oscillation.amplitude can be specified either by a data file using DATAFILE("filename") or by a formula, like "1e-18*laguerre(2, 1, 2*r*r/(w*w))" for a surface vibration or "1e-10*x" for yaw motion or "1e-20" for displacement motion.

oscillation.phase is used to define the relative phase of various mirror motions. E.g., when ITM.oscillation.phase = 0 and ETM.oscillation.phase = 0, then two mirrors move in phase (L_{FP} =

L0 $-d_{ITM}$ - d_{ETM}) and when ETM.oscillation.phase = PI, the two mirrors move 180 degree out of phase (L_{FP} = L0 – d_{ITM} + d_{ETM}).

Signal sidebands are generated by the algorithm discussed in the physics section of this document.

3.3.1 "choose mirrors to move"

When you choose this command, you are prompted for a list of mirrors to shake. Only those mirrors are listed whose oscillation.amplitude are specified.

3.3.2 "File name tag"

Then you specify the name tag for files which are created automatically. "tag_src.dat" is the stationary field, and "tag_frequency.dat" (actual name is like "tag_100.dat" for the signal at 100Hz) is the signal sideband field for the frequency. When more than one mirrors are moving, this is the total result, not a signal due to the motion of one mirror.

For a FP cavity, these are the fields pointing to ITM, and for a coupled cavity, these are the ones pointing to ITM (_FP_*.dat) and to RM (_RC_*.dat).

3.3.3 "Choose how to specify frequencies"

Then, you specify the frequencies of the signal sideband. You can specify either by a series of frequencies (linear or log) or by a table of explicit frequencies. Once specified, the program starts calculating all sidebands together using threads.

When you choose to specify by a table, you are prompted to enter frequencies. Multiple frequencies can be entered in a line by using "," as a separator. The end of the data is marked by a empty line. E.g., the following lines specify 8 frequencies.

```
SIS> frequencies >> freq3-4, freq3-2, freq3, freq3+4
SIS> frequencies >> 0
SIS> frequencies >> 100, 200, 300
SIS> frequencies >>
```

Variables defined elsewhere (most likely in simSpec section) can be used to specify the frequency. In this example, freq3 is calculated in simSpec from cavity parameters.

3.3.4 Results and analysis

After the signal calculation is completed, summary result like the following is printed.

```
**********************************
freq  18201.5, power = 7.227188083419e-11, with full power of 280.67780369422
freq  18205.5, power = 7.2832778848184e-11, with full power of 280.67780369422
freq        0, power = 2.4708917614136e-15, with full power of 280.67780369422
freq -19321.8, power = 7.2271881393102e-11, with full power of 280.67780369422
freq -19317.8, power = 7.2832779055485e-11, with full power of 280.67780369422
**********************************
```

This shows the power of the signal sideband for each frequency, together with the power of the source stationary field, marked as "full power". These are the powers of the fields stored in the "tag_xxx.dat" files.

The details of the signal can be studied by the commands “modeAmp” and “saveFields”. These are the same functions provided in the main menu.

```
SIS> f = 18201.5 : next action
SIS>       modeAmp         saveField       next            exit
```

“next” moves to the next frequency, and “exit” quits the analysis of the signal. When there are more than one source, this detailed study can be done only for the second source.

3.4 telescope

This command prints the current definitions of telescopes and calculates the output fields of telescopes. One telescope is defined in the following way:

outField = TELESCOPE(inField, { TELE_ TYPE, par, … }, { TELE_ TYPE, par, … }, …)

inField is a name of the initial field which goes to the telescope. This can be the name any of the fields interacting with the main cavity or any other fields defined using TELESCOPE function. They are displayed when “saveField” is selected in the main menu. Instead of the field name, the number associated with the field followed by “.” serves the same. E.g., inputField or 0. both specified the input to the telescope is the initialField.

TELE_ TYPE specifies the telescope element type and following parameters specify the element characteristics.

(1) {TELE_SPACE, distance} : propagation of a field for a length specified by the first argument. When “distance” is positive, it is rounded to an integer multiple of the NdYAG wavelength. When “distance” is negative, it is interpreted as a change of the gouy phase.

(2) {TELE_LENS, fNumber, radius, x0, y0} : transmission through a lens with f-number of fNumber and radius of radius, center being placed at (x0,y0)

(3) {TELE_MIRROR, ROC, thetaAOI, phiAOI, radius, x0, y0} : reflection by a mirror with a radius of curvature of ROC and mirror radius of radius, center being placed at (x0,y0). (x0,y0) is measured by the coordinate of the HR side or the reflected field. The incidence direction is defined by thetaAOI (def=0) and phiAOI (def=0). The direction vector is

[sin(thetaAOI) cos(phiAOI), sin(thetaAOI) sin(phiAOI), cos(thetaAOI)]

When phiAOI = 0, the beam comes in the plane of cavity with an incident angle of thetaAOI.

Any number of telescope elements can be placed to define one telescope. The first parameter of each element is mandatory. Default values of radius, x0 and y0 and infinite, 0 and 0, respectively.

3.5 delL

This one sets the value of a cavity in m. The default value is the result of the last action, i.e., it is the locked cavity length if the last action was lock.

When the absolute value is > 10^{-4}, it is interpreted as a change in units of the wavelength of the YAG lazer, i.e.,

the new length = old length + dell * YAG wave length

This is to detune the cavity length off from the locked length by an offset.

Anytime, the macroscopic cavity length is "L0" and the microscopic length is "_cavLeng". E.g., for a coupled cavity, "CCav.L0" is the macroscopic optical length of the Michelson cavity and "CCav.FPcav._cavLeng" is the microscopic adjustment of the FP cavity.

3.6 modeAmp

One can do a simple mode analysis of all fields.

3.7 saveField

One can save any fields calculated.

3.8 storeMap

Various information of mirrors are stored in data files. If you want to save multiple files, e.g., saving files in a loop, see the section explaining "Variable file names with @'s and *'s".

There are three data file formats.

format 0

data are stored in NxN matrix formats. When loaded in matlab as

dat = load('dataFileName');

then dat(ix, iy) is the ix'th point in x axis and iy's point in y axis.

format 1

data are stored in a NNx3 format, where NN is the number of non zero values.

x1 y1 val(x1,y1)

x1 y2 val(x1,y2)

…

format 2

header information (SIS ASCII Format 1) about the mirror is inserted atop of the data of format 1, and this file can be used as an input data file for SIS using the built-in DATAFILE function.

3.8.1 HRsurface

This is the HR side surface data created using the specification by mirror.opt.HR_phase, i.e., the HR side surface phasemap except for the curvature specified by mirror.opt.ROC. The file can be used to view the actual surface data specified by built in functions like NOISESPEC or THERMALPHASE.

3.8.2 TransPhase

Transmission phase in units of radian.

3.8.3 FilteredHR

HR Surface map after low pass.

3.8.4 PSDHR

After calculating 2D PSD(fx, fy) using map data in a square region specified by the user, three kinds of 1D PSD for positive frequency are calculated as follows.

$$PSD1x(f) = \int PSD2(f, fy)\,dfy + \int PSD2(-f, fy)\,dfy$$

$$PSD1y(f) = \int PSD2(fx, f)\,dfx + \int PSD2(fx, -f)\,dfx$$

$$PSD1r(f) = f\int PSD2(f, \phi)\,d\phi$$

All of these 1D PSD satisfies the following power formula.

$$\int_0^\infty PSD1(f)\,df = \iint dx\,dy\ \delta(x,y)^2 / area$$

These 1D PSD's, together with frequencies, are stored in an ASCII text file in the following format.

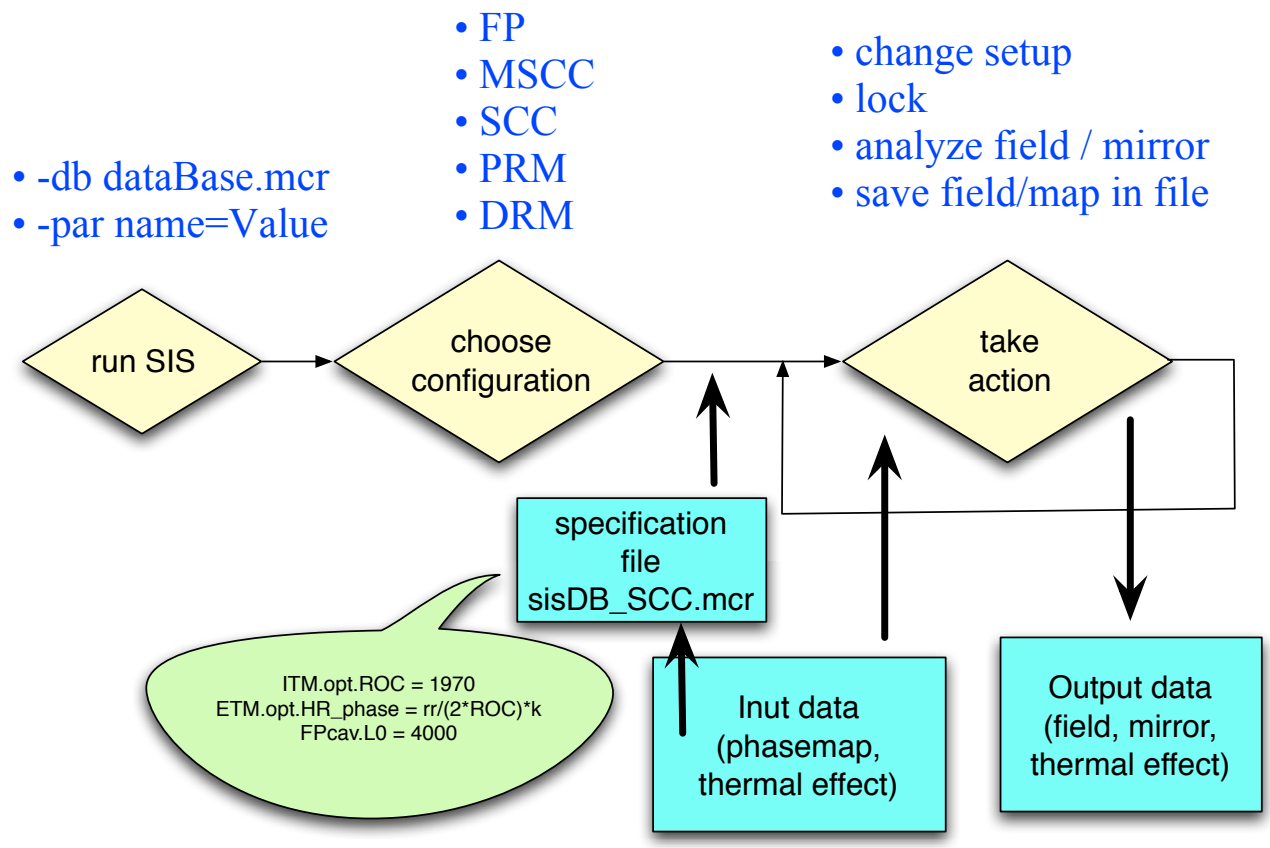
f1	PSD1x(f1)	PSD1y(f1)	PSD1r(f1)
f2	PSD1x(f2)	PSD1y(f2)	PSD1r(f2)

…

The unit of the frequency is 1/mm and that of PSD is nm^2 mm.

4 SIS setups

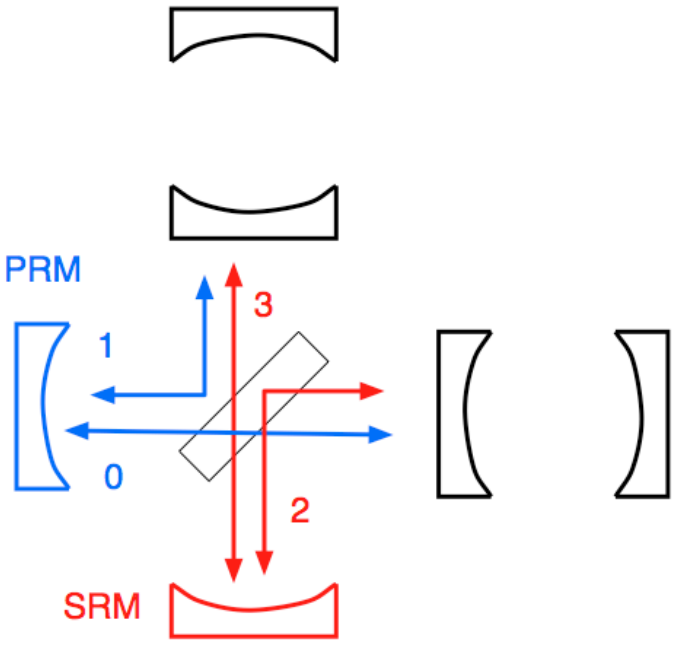
4.1 Work flow



4.2 Optical configurations

4.2.1 FP

4.2.2 Coupled cavity



4.2.2.1 MSCC

4.2.2.2 SCC

4.3 FAQ

4.3.1 Q : Do I have to prepare any input files to run the program?

A: No, you don't. When you run the program and choose which optical configuration to simulate, a default values typical for advanced LIGO are automatically setup as default values. Type "simSpec" in the main work area and then type "definition" at the prompt of PARAM>, and you will see all specifications. When you keep only those settings you want to customize in sisDB.mcr, like ITM.opt.ROC = 1973, then when you run sis next time, you can simulate that customized configuration, i.e., default + your changes.

4.4 Input files to specify the custom optical configuration

When you run the program, sis, you choose which optical configuration to simulate. When you choose one, e.g., FP, the program loads in a file named sisDB_FP.mcr. If there is no file with that name, the program tried to read sisDB.mcr. If you want to specify a name of the file, you give it as a runtime option –db:

sis –db specFileName

4.5 SIS runtime options

4.5.1 –db databaseFileName

Define specification file

4.5.2 –thread Nthread

specify the number of threads used. Default is 4 as of Jan, 2008.

4.5.3 –par name1 = val1 (, name2 = val2 …)

define variable name(s) and associated value(s). These are defined before the database is read in. See "defaults" argument below.

4.6 SIS user interface

4.6.1 Help

Type "help" or "?" at any prompt, and you will see some (hopefully) helpful information.

4.6.2 Storing key strokes

It often happens that you want to repeat similar calculation. When you type "@(command.in" (command.in can be any legal unix file name, but it is recommended to use ".in" as the file name extension), then all your key strokes and prompts are stored in the file "command.in", until you type "@)". When you type "@command.in", then the stored key strokes are repeated. The file is a simple text file and can be modified by using a text editor. Any text string following "%" is a discarded, so any comments can be placed with "%". All the prompts stored in the file is preceded by "%" and the prompts are in the file so that you can find to what prompt you are replying.

4.6.3 Loop

@REPEATBEGN (or @RBEG) Nbegin Nend (or Ntot)

actions …

@REPEATEND (or @REND)

Actions between @REPEATEND and @REPEATEND are repeated Nend – Nbegin – 1 times. If only Ntot is specified, it is the same as Nbegin = 0 and Nend = Ntot. During the loop, iLoop is changed from Nbegin to Nend. "iLoop" and "iLoopEnd" variables can be used to change this loop counter and the loop end. Loop is useful to (1) simulate the response for a range of variables (e.g., cavity length from -1nm to +1nm around the resonance with a step of 0.1nm), (2) generate many random surfaces using different random numbers and (3) repeat field calculation with the thermal deformation (thermal deformation is calculated from the resonating field, then the field is calculated using the thermal deformation, repeat this cycle until stationary).

4.6.4 Variable file names with @'s and *'s

When specifying output file or directory names, if the name contains @ and / or *, the actual name is created in the following way.

- "xxx@@@yyy" : the actual name is "xxx" + iLoop value with N[@] digits + "yyy", where N[@] is the number of consecutive @'s. iLoop is generally 0, but changes in the loop as a loop counter. In the loop defined by @RBEG and @REND, when a file name is specified by "ITM@@@Field.dat", the actual file used is ITM002Field.dat when iLoop is 2.
- "xxx***yyy" : the actual name is "xxx" + consecutive number with N[*] digits + "yyy", where N[*] is the number of consecutive *'s. The consecutive number is found to avoid duplication of file names. E.g., if "ITM**Field.dat" is use to specify a file name, the first one created is ITM00Field.dat, the second one will be named ITM01Fied.dat, etc.

4.6.5 @SHELL

starts an session to interact with the process using popen and pclose. The session ends with "exit" command. This can be used to, e.g., rename files or move files.

4.7 Basic rule of the input file and simSpec

4.7.1 Comment

"%" marks the start of a comment, and all text until the end of the line is neglected.

4.7.2 Printing on screen or saving in a file

#print "text" prints the text, and #print variableName prints the value of the variable together with the variable name.

```
#print "This is FP spec : " ITM_ROC   ETM_ROC   CavLen
```

will print a line like

```
This is FP spec : "ITM_ROC" = 1971  "ETM_ROC" = 2191  "CavLen" = 4000
```

4.7.3 Defining variables

A variable can be defined by writing name = value, where name is a legitimate language C variable name. It should not contain a period and should not be reserved words.

4.7.4 C style syntax

User defined variables and predefined variables can be used to define other variables using C syntax, C functions and SIS build in functions.

4.7.5 Defining vector

A vector can be specified by a series of values in a square bracket, like [0, 1, 2].

4.7.6 Viewing existing objects and variables

When changing interactively in simSpec sowkrspace, type “variables” to see all variables, type “definition” to see predefined objects, type “functions” to see functions added in SIS, and type “name = value” to change the definition. Type “help” when you need to see these commands.

4.7.7 Line continuation

A long line can be split into multiple lines using a continuation char ‘\’ at the end the line. Lines like

```
Line = a \
+ b
```

is the same as

```
Line = a + b
```

4.7.8 Default value assignment :

```
#defaults
name1 = val1
…
#enddef
```

These defines default values, i.e., they are used if variable nameN is not defined so far. If nameN value is assigned using –par runtime argument, that value is used.

With one definition file, one can run different cases by assigning different values to the variables in the default session through runtime option –par.

Example :

In sisDB_FP.mcr

```
#defaults
ITM_ROC_HOT = 1941
#enddef
ITM_ROC_COLD = 1971
ITM.opt.HR_phase = rr /2 * (1/ITM_ROC_HOT – 1/ITM_ROC_COLD)
…
```

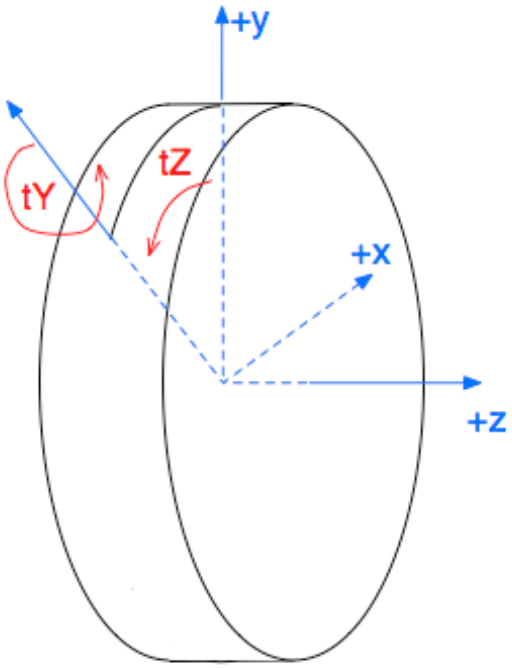
Run with different ITM_ROC_HOT without modifying the mcr file.

```
./StatIFOSim –par  ITM_ROC_HOT = 1951
```

4.8 Basic physics setting

4.8.1 mirror rotation [tX, tY, tZ]

The y axis is rotated around the z-axis (perpendicular to the surface, pointing out from HR surface) by tZ, then rotated by tY around the new y axis. To specify a yaw rotation, use [0, yaw, 0] and to specify a pitch rotation, use [0, -pitch,PI/2]



4.8.2 inputBeam.matchToCavity

This is used to automate the input beam setting. When this is 1 or 2 or 3, the mode of the input beam, the waistSize and waistPosition (negative for converging beam) are automatically calculated to match to the cavity specification. See the definition of the modeBaseScheme below. When the surface deformation, like mirror phase maps or thermoelastic deformation are defined for mirrors, the effective curvature is used to find the best mode base. If you want to use the mode base specified by the waistSize and waistPosition, set matchToCavity to 0.

If you set this to be negative, the mode base is calculated once for the first run, and that base is used for the rest of the run.

In order to define the input mode base by other mode quantities, use built in functions to convert to the waist size and the distance to waist. By using wROC2dist2waist and wROC2waistSize, one can use the beam size and ROC of the field to specify the mode. After the first run, one can get the beam size and ROC which matches to the cavity by :

wref = waistDist2W(inputBeam.waistSize, inputBeam.waistPositon)

ROCref = waistDist2ROC(inputBeam.waistSize, inputBeam.waistPositon)

Then, one can setup the input beam whose beam size and ROC are off by ε_w and ε_{ROC} by

inputBeam.waistSize = wROC2waistSize (wref + ε_w , ROCref + ε_{ROC})

inputBeam.waistPositon = wROC2dist2waist (wref + ε_w , ROCref + ε_{ROC})

4.8.3 inputBeam specification by data

inputBeam.fldAmp and inputBeam.fldPhase can be used to load the input field from data files. The field is calculated by amp * exp(i phase). The unit of the phase in degree by default, and if the file name contains “radian”, radian is used as the unit of the phase.

The modal mode parameters need to be specified in addition to these field data files.

4.8.4 FPcav.matchToInput :

FPcav.L0, the macroscopic cavity length, is rounded to an integer multiple of the laser wavelength. Microscopic adjustment is done by FPcav.delL. To make the cavity resonant, delL needs to be set to correct for the Gouy phase. When matchToInput is 1 or 2 or 3, this delL is calculated automatically to compensate the Gouy phase of the incoming mode. See the definition of modeBaseShceme below. This option is to help to set the initial condition. With this option on, the lock will be quicker and, even without locking, calField will give the field near resonance. If you want to scan the field near resonance by explicitly specifying the cavity length, i.e., FPcav.delL, you need to set this option to be 0, otherwise, some correction is automatically added to delL before the calculation is done.

4.8.5 Curvature of RM :

When 0, it is set to match with the ROC of the beam calculated staring from the arm.

4.8.6 CCav.MMT2_MMT3 :

The distance between the MMT2 and MMT3 optics. When this value is negative, this distance is adjusted using other constraints. If CCav.RM_beamSize is not 0, the distance is adjusted so that the beam size on the RM becomes the value specified by RM_beamSize. If CCav.RM_beamSize is 0, then the curvature of the RM is used to adjust the distance so that the curvature of the field on RM matches with the RM curvature.

4.9 Parameters

4.9.1 Mode base specification

```
enum modeBaseScheme {
noChoice_MBID = 0,
```

mode is not defined automatically

```
coldMirror_MBID = 1,
```

mirror.opt.ROC is used to calculate the HR side curvature and the lens effect of the mirror.

```
hotMirrorROC_MBID = 2,
```

All mirror specifications (ROC,phasemap,transmission map,etc) are used to calculate the surface shape and transmission phases and they are fit parabola to calculate the ROC and the thin lens effect of the mirror.

```
hotMirrorEffectiveROC_MBID = 3
```

Optics quantities are calculated using the same method as above, but the ROC and the thin lens effect are calculated using a Gaussian beam shape resonating in the cavity.

};

4.9.2 inputBeam.matchToCavity

When the value is 0, inputBeam.[waistSize, waistPosition] are used to setup the input beam. When a value of modeBaseScheme element is used, the input beam base is setup to match to the mode of the optical system.

When the value is negative of modeBaseScheme, then the input beam mode is calculated once using the scheme defined by (negative of) the value, and that beam shape is used until the parameter is changed again. For example, if you want to study the effect of the thermal deformation using the beam which matches to the cold state cavity, set the value to be -1. Then the input beam is calculated to match with the cold cavity, and the same input is used for the subsequent calculations with evolving thermal deformations.

4.10 SIS built in functions

4.10.1 Loading data file

4.10.1.1 DATAFILE(“filename” (, vector<int> WykoIndex))

DATAFILELP : Same as DATAFILE except the raw data is low pass filtered using FIR1 filter with order 50.

This function reads the file. Various file formats are supported, including Wyko ascii data. The data need to be specified on grid points in a rectangular area, called data area below. E.g., when the data set is defined in a rectangular area, Wx by Wy, with number of grids Nx and Ny, then data(i,j) (i=0~Nx-1, j=0~Ny-1) is the average of the data in a area

$$x_0 - \frac{Wx}{2} + \frac{Wx}{Nx} i \sim x_0 - \frac{Wx}{2} + \frac{Wx}{Nx}(i+1)$$
$$y_0 - \frac{Wy}{2} + \frac{Wy}{Ny} j \sim y_0 - \frac{Wy}{2} + \frac{Wy}{Ny}(j+1)$$

When an array of values are defined in an vector format, {val1, val2, …, valN}, Zernike polynomials of those indexes are subtracted from the data. See NOISESPEC section below about the correspondence between the value and actual functions.

4.10.1.2 Data file format

Data can be stored in various different ways. The first two formats have the data area information stored in the file. The third format uses the file name to specify the data area.

4.10.1.2.1 Wyko ASCII data

If the first line in the data file is

```
Wyko ASCII Data File Format 2  0     1
```

the header information specified by “X Size”, “Y Size”, “Pixel_size”, “Aspect” and “Wavelength” are used to interpret the rest of the data.

Nx = X Size, Yx = Y Size, Wx = Nx*Pixel_size, Wy= Nx*Pixel_size*Aspect and the center of the rectangle is (0,0). Units of the data values is Wavelength which is specified in units of nanometer. In other words, the actual value in meter is data(i,j) * Wavelength / 10^9.

When the third value is not a number, it is treated as a bad value.

4.10.1.2.2 SIS ASCII Format 1

If the first line in the data file is

```
SIS ASCII Format 1
```

the header information specified by “XSize”, “YSize”, “XPixel”, “YPixel”, “x0” and “y0” are used to interpret the rest of the data.

Nx = XSize, Yx = YSize, Wx = XSize*XPixel, Wy=YSize*YPixel and the center of the rectangle is (x0,y0). The units of data is meter.

If there is only one value in a line, it is treated as the data at i = line/Ny, j = mod(line,Nx), or

data(0,0)
data(0,1)
…
data(0,Ny-1)
data(1,0)
…
data(Nx-1,Ny-1)

If there are there values in a line, they area treated as the x value, y value and data at (x,y).

xi yj data(xi,yj)

If there are two values in a line, they area treated as the x value and y value and the data at (x,y) is bad.

xi yj [bad data point]

An example of this format is

```
SIS ASCII Format 1
XSize = 64
YSize = 64
XPixel = 8e-4
YPixel = 8e-4
x0 = -4e-4
y0 = -4e-4
END_HEADER
  -2.5600000e-002 -2.5600000e-002  0.0000000e+000
  -2.4800000e-002 -2.5600000e-002  0.0000000e+000
  -2.4000000e-002 -2.5600000e-002  0.0000000e+000
  -2.3200000e-002 -2.5600000e-002  0.0000000e+000
  -2.2400000e-002 -2.5600000e-002  0.0000000e+000
…
```

4.10.1.2.3 Data area in data file name

If the first line of the data file is not none of the above formats, the file name is interpreted to specify the data area.

If the file name has either of the following symbols followed by numbers, they represent the rectangle size or number of grids.

_N	number of grids for x and y axis
_Nx(_Ny)	number of grids for x (y) axis
_W	rectangle size for x and y axis in units of mm
_Wx(_Wy)	rectangle size for x (y) axis in units of mm

For example, data_N128_W350.dat means the data in the file is contained in a rectangle of 35cm x 35cm and the number of grid points are 128 x 128. The order of these tags, additional characters in the file name and extensions are of no importance.

Lengths can have names of units, “cm”, “mm” and “um”, like file_W200um. This means the data window size is 200 micron. Defaut length is “mm”, as is mentioned above.

The center of the rectangle is always (0,0).

The data format in the file can be either one row format (1 x Nx*Ny), the same as for the SIS ACSII Format, or Ny x Nx matrix format of data points, with data being separated by space, tab or comma.

val(ix=0,iy=0), val(ix=1,iy=0), …, val(ix=Nx-1,iy=0)

…

val(ix=0,iy=Ny-1), val(ix=1,iy=Ny-1), …, val(ix=Nx-1,iy=Ny-1)

4.10.1.3 Low pass filter

When lowPassOption is passed, low pass filtering is applied.

lowPassOption = 1 uses matlab fir1 filter as

$$filter = fir1(lowPassParam, 2 * FFTBin / dataBin)$$

for each direction. If the normalized cutoff frequency is larger than 1, no filter is applied. If it is larger than 0.99, it is limited by 0.99.

lowPassParam is used to define the order of the filter for now, but it will be change to the required accuracy in the future version of SIS.

4.10.2 Thermal effects based on Hello-Vinet model

The physics content of this function is explained in Ch. 13.

4.10.2.1 THERMOELASTIC(double beamSize, double Psubs, double Pcoat, [,double T0])

This is the deformation of the HR side surface due to thermal heating. This quantity should be used to specify the HR_phase deformation, like

ITM.opt.HR_phase = THERMOELASTIC(-1, -2e-6, -0.5e-6)

When parameters are positive, they represent following quantities.

- beamSize : thermal deformation is calculated assuming that the beam shape is a pure Gaussian mode with the beam size specified by beamSize.
- Psubs : Absolute power absorbed in the substrate. ~~For a nominal material and for the high finesse arm design, it was assumed to be 0.084W at the full power load.~~ Because of the adoption of very low absorption substrate, Heraeus 3001, with expected absorption rate of 0.02ppm/cm, for ITM, BS and CP, this can be set to 0.
- Pcoat : Absolute power absorbed in the coating on the HR surface. ~~For the high finesse design, the nominal value at the full load was 0.425W.~~ For the low finesse design with the full power load of 624kW, this becomes 0.312W.

When these parameters are negative, thermal effects are calculated using the stationary field distribution in the previous result in the following way.

- beamSize : The beam shape of the previous result is used to calculate the actual deformation. The value itself is not used.
- Psubs : The absolute value of the absorbed power in the substrate is calculated using (-Psubs) as the absorption rate / meter. The total absorption in the substrate is calculated by multiplying the stationary field, this absorption rate and the optic thickness. ~~Originally, 200ppm/m was used,~~ but a very small absorption rate (0.4 ppm/m) was observed for the low OH content optic, Heraeus 3001, to be used for BS, ITM and CP. So this can be effectively 0.
- Pcoat : The absolute value of the absorbed power in the coating is calculated using (-Pcoat) as the absorption rate. The nominal value is 0.5ppm.

4.10.2.2 THERMALPHASE(double beamSize, double Psubs, double Pcoat, [,double T0])

This is the thermal lens induced in the substrate. This quantity should be used to specify the transmission phase of an optic, like

ITM.opt.trans_phase = THERMALPHASE(-1, -0.4e-6, -0.5e-6)

4.10.2.3 Addendum

When one of these functions are used in the specification, after each run, a data file for each function call is created so that the thermal deformation can be simulated in the following run using the information of the current run. The file name is like FPcavETMThermoElastic_N256_W700. This means that the file is for the THERMOELASTIC function used for ETM in FPcav, number of grid points is 256 and the window size is 0.7m.

One needs to run the simulation using the same configuration to find the thermally equilibrium solution. In order to let the simulation know that the thermal state of optics is changed, the simplest trick is to touch the optic or optical system, like

touch FPcav % notify that the cavity parameters need to be updated

in simSpec workspace or

@@touch FPcav

in the main workspace for a shortcut without going into simSpec workspace.

4.10.3 Random surface generation

There are four functions to generate mirror phase maps based on a given PSD. RANDOM2D uses an analytic expression of a 2D PSD and generates phasemaps whose PSD is statistically the same as the given PSD. RANDOMMAP uses a existing phasemap to define the PSD and RANDOM1D uses a simple popular form as the 1D PSD specification. NOISESPEC is obsolete, and RANDOM2D is much more flexible and supersedes NOISESPEC.

All of these functions share a set of common variables, which are explained in the section for RANDOM2D.

4.10.3.1 Basic algorithm

For a FFT window setting, W for the window size and N for the number of grids, the Amplitude Spectral Density is calculated at discrete points, fx = -df*(N-1), -df*(N-2), …, df*N, and the same for fy, where df = 1/W.

The magnitude of ASD at (fx,fy) is calculated by using an analysis formula or by using a PSD calculated from a real phasemap data. Then, at each (fx,fy), the complex ASD is calculated by multiplying (rand1 + i x rand2) / √2, where rand1 and rand2 are two separate random numbers with the normal distribution with the width of 1.

This is based on the algorithm developed by F. Bondu.

Constraints is imposed that the spatial map after FFT conversion from this ASD becomes a real (not complex) distribution.

4.10.3.2 RANDOM2D(int rand_seed, double rms, "2d PSD(fx,fy,ff,f)", vector<int> WykoIndex)

This function generates a phasemap using an analytic expression of 2D PSD. For example, the third argument can be specified, using predefined variables A, B and C, as

"A/(1+(B*f)^2)^((C+1)/2)"

The function can use fx, fy, f and ff as predefined variables, where $ff=fx^2+fy^2$, and $f=\sqrt{ff}$.

The first argument, rand_seed, specified the random number seed. Following few cases have specific meanings.

- 0 : random number seed is generated using time. So all sequence of random numbers are random.

- -1 : the new seed is generated using the previous seed.
- -2 : the previous seed is used as the new seed, so the same random number is generated as before.

If the first rand_seed is specified using an explicit number, like 123, and when rand_seed for the following events are set to be -1, a reproducible series of random numbers are generated.

When you want to use the same random number as before, set the rand_seed to be -1. E.g., run once with two mirrors have aberration and, in the following run, just one surface has same aberration while the other with perfect surface.

The second parameter, rms, specifies the rms of the distribution. When a positive number is specified, that number is the rms value of the final distribution after subtracting Zernike terms, discussed below. If a negative value is specified, the absolute value is used as the rms of the distribution before the Zernike term is subtracted, so the final rms can be different from this specified value. When rms = 0, then there will be no renormalization will be applied.

The last argument, WykoIndex is a series of Zernike polynomial Wyko indexes which are subtracted from the surface data. A series of indexes are specified by the following form:

{ index1, index2, …, indexN }

If one negative number between -1 and -36 is given for the last argument, Zernike terms up to that order are subtracted, i.e., -N is the same as specifying {1, …, N}. The index can be the Wyko serial number up to 36 or a number whose absolute number is larger than or equal to 1000.

The serial number scheme assigns a number to a Zernike polynomial with a radial (n) and azimuthal (m) index. A few examples are shown in the following table.

Wyko	Function	Description	Radial	azimuthal	Serial No
0	1	Piston	0	0	0
1	$\rho\cos\theta$	X-tilt	1	1	1001
2	$\rho\sin\theta$	Y-tilt	1	-1	-1001
3	$2\rho^2-1$	Power	2	0	2000
4	$\rho^2\cos2\theta$	Astigmatism, axis at 0°	2	2	2002
5	$\rho^2\sin2\theta$	Astigmatism, axis at 45°	2	-2	-2002
6	$(3\rho^3-2\rho)\cos\theta$	3rd order Coma, y-axis	3	1	3001
7	$(3\rho^3-2\rho)\sin\theta$	3rd order Coma, x-axis	3	-1	-3001
8	$6\rho^4-6\rho^2+1$	3rd order spherical aberration	4	0	4000
9	$\rho^3\cos3\theta$	Tri-Foil, base of y-axis	3	3	3003
10	$\rho^3\sin3\theta$	Tri-Foil, base of x-axis	3	-3	-3003

Table 1 Zernike polynomials

$$\int_0^1 \rho d\rho \int_0^{2\pi} d\theta \;\; Z_{n,m}(\rho,\theta)^2 = \begin{cases} \dfrac{\pi}{n+1} (m=0) \\ \dfrac{\pi}{2(n+1)} (m \neq 0) \end{cases}$$

The rms value for each Zernike term is

$$\sigma^2 = \frac{\sum Z_{n,m}(x_i,y_j)^2}{M} - (\frac{\sum Z_{n,m}(x_i,y_j)}{M})^2 = \frac{\sum \Delta x \Delta y Z_{n,m}^2}{\Delta x \Delta y \cdot M}$$
$$= \frac{\iint dx\, dy Z_{n,m}(x,y)^2}{\Delta^2 \cdot M} = \frac{R^2 \iint \rho\, d\rho Z_{n,m}(x,y)^2}{\Delta^2 \cdot M}$$
$$= \frac{R^2 \iint \rho\, d\rho Z_{n,m}(x,y)^2}{\Delta^2 \cdot \pi R^2 / \Delta^2} = \frac{\iint \rho\, d\rho Z_{n,m}(x,y)^2}{\pi}$$
$$= \begin{cases} \dfrac{1}{n+1} (m=0) \\ \dfrac{1}{2(n+1)} (m \neq 0) \end{cases}$$

In this formula, R is the radius of the circle, Δ (=Δx=Δy) is the size of spacing and M is the number of data points in the circle.

A genetic serial number is defined based on the two indexes n and m as

$$sign(m) \times (1000 \cdot n + abs(m))$$

This index is shown at the last column in the table.

4.10.3.3 RANDOMMAP(int rand_seed, double rms, "DATAFILE", "ASD formula, @ for data", vector<int> WykoIndex)

The random map is generated using a PSD of existing phasemap data. The map data spefieid in the third argument, is used to calculate ASD.

The fourth argument can be used to specify the modification of this ASD. For example, if you want to place a cutoff at fcut, then you set the fourth argument to be

"@ * if(f < fcut, 1, 0)"

c-style formula with SIS extensions can be used to specify the formula, and @ is used to specify the magnitude of the ASD calculated from the data.

4.10.3.4 RANDOM1D(int rand_seed, double rms, double A, double B, double C, vector<int> WykoIndex)

This function uses a simple 1D PSD parameterization to specify the 2D PSD in the following way. One popular parameterization of 1D PSD is

$$PSD1(f) = \frac{A}{\left(1 + (B \cdot f)^2\right)^C}$$

When A, B and C for this expression is specified using the 3rd, 4th and 5th argument, then 2D PSD is calculated as

$$PSD2(f,\phi) = \frac{D \cdot B}{\sqrt{1+(B \cdot f)^2}} PSD1(f)$$

$$D = \frac{\Gamma(\frac{C+1}{2})}{2\sqrt{\pi}\Gamma(\frac{C}{2})} = \begin{cases} \frac{1}{2\pi} & (C=1) \\ \frac{1}{4} & (C=2) \end{cases}$$

This is based on one definition of 1D PSD (LIGO-T070082, Optical Scattering by J.C.Stover). If one prefers another definition that

$$PSD1(f) = f\int PSD2(f,\phi)d\phi$$

then, simply use RANDOM2D with PSD formula = PSD1(f)/(2πf).

4.10.3.5 NOISESPEC(int rand_seed, double rms, int power, vector<int> WykoIndex)

This map is specified the power of PSD. The third parameter, power, determines the original surface psd spectrum power, i.e.,

$$\text{psd} = f^{-2\ \text{power}}.$$

After subtracting Zernike terms (see below), the spectrum can change.

4.10.4 Field property calculation

These functions calculate properties of the field specified by an integer argument fieldIndex. To find the value of fieldIndex of an specific field, go to modeAmp in the main menu.

```
FPOWER( fieldIndex )          : power of the field
FFPROD( fieldIndex1, fieldIndex2, demodPhase )
                      : real( field1 * conj(field2) * exp(-i demodPhase) )
FLDROC( fieldIndex )          : ROC of the field
FLDWZ( fieldIndex )           : beam size w(z) of the field
FHGPWR( fieldIndex, m, n )    : power of HG(m,n) component
FLGPWR( fieldIndex, p, m )    : power of LG(p,m) component
```

These quantities are calculated by fitting the field using the following functional form.

$$F(x,y) = A \cdot \exp(-(x-x_0)^2(\frac{1}{w_x^2} + i\frac{k}{2R_x}) - (y-y_0)^2(\frac{1}{w_y^2} + i\frac{k}{2R_y}))$$

The range of data used is estimated by a simple rule based on the power distribution. The range is set when the power in the next point is less than 1% of the center or if the slope of the power becomes increasing.

4.10.5 Mirror property calculation

These functions calculate properties of the mirror specified by an integer argument mirrorIndex. To find the value of mirrorIndex of an specific mirror, go to mirrorInfo in the main menu.

```
MHRROC( mirrorIndex )         : Effective ROC of the HR surface
MHRDELZ( mirrorIndex )        : Effective piston of the HR surface
MAPPING( inMap, "xnew formula by x and y", "ynew formula by x and y" )
  : out(x,y) = inMap(xnew, ynew). E.g., to rotate the original mirror by phi,
    xnew = cos(phi) x - sin(phi) y, ynew = sin(phi) x + cos(phi) y
```

4.10.6 Cavity property calculation (FP : Fabry-Perot cavity)

```
fp_rayleigh_range(L,R1,R2)    : Rayleigh range of a FP cavity
fp_dist2waist(L,R1,R2)        : Distance between front mirror to waist in FP
fp_gouyphase(L,R1,R2)       : Gouypahse change from front to back in FP
fp_wITM(L,R1,R2)      : beam size on ITM of a FP cavity
fp_wETM(L,R1,R2)      : beam size on ETM of a FP cavity
ext_rayleigh_range(z0,z,n)
          : Rayleigh range outside of FP front mirror with refractive index n
extfp_dist2waist(z0,z,n)
```

```
          : Distance of the outside of FP front mirror to waist
beamSizeYAG(z0,z)            : Beam size for YAG laser
wROC2dist2waist(w,ROC)       : Distance to waist using beam size and ROC
wROC2waistSize(w,ROC)        : Waist size using beam size and ROC
waistDist2W(w0,z)            : Beam size using waist and distance to waist
waistDist2ROC(w0,z)          : ROC using waist and distance to waist
lens_dist2waist(z0,z,f)      : Rayleigh range after going through a lens
lens_rayleigh_range(z0,z,f)  : Distance to waist after going through a lens
```

4.10.7 Math functions

```
hermite(n,x)                 : Hermite polynomial
laguerre(p,m,x)              : Laguerre polynomial
wyko(index,r,theta)          : zernike polynomial in wyko order
zernike(n,m,r,theta)         : zernike polynomial
zernikeFlat(n,m,r,theta)     : Zernike(1,theta) for r > 1
jbessel(N,x)                 : Bessel function
gamma(x)                     : gamma function for x > 0
step(x,x0,y0,x1,y1)          : = y0 for x<x0, = y1 for x>x1, smoothly
                               connected in between using 3rd order polynomial.
                               The derivatives at x0 and x1 are 0.
```

4.10.8 Special functions

The conditional function “if” is implemented as follows:

val = if(double condition, double val_for_true, double val_for_false)

This function returns val_for_true if condition is true or non zero, and returns val_for_false when condition is false or 0.

4.10.9 Standard C functions

In addition to these special functions, SIS accepts most of the standard C functions.

- Unary operators : +, -, !
- Binary operators : ^, *, /, +, -, <, <=, >, >=, ==, !=, &&, ==
  (a^b = pow(a,b), not bitwise xor)
- Unary functions : sqrt, sin, cos, tan, atan, acos, asin, atan, log, log10, exp, sinh, cosh, tanh, fabs, ceil, floor
- Binary functions : pow, atan2, fmod, max, min

5 Field calculation

5.1 Object structure

Cavity objects, like SISFP and SISCC, are derived class of SISResonator.

SISResonator class is defined in SISComponents.h and cc files. This should be fixed.

5.2 Stationary Field

Stationary field is searched until the power does not move more than calcAccuracy.

```
SISAppFrame
-> SISAppFrame(SISAppFP,SISAppCC)::calcStationaryFieldCase
-> SISResonator(SISFP,SISCC)::calcStationaryField
        -> SISResonator::calcStationaryField
              ( simple dispatcher to simplify arguments )
              -> SISResonator::calcStationaryFieldSR
                    Loop to find stationary
                    Print summary every 20th reported with prefix
                    “calcStationaryField[N] =”
                    Print when converged reported with prefix
                    “calcStationaryField[N] =”
                    runSpec option “printCount” controls how often
                    printed. Nsecondary*1000 + Nprimary
                    -> SISResonator(SISFP,SISCC)::oneEvolution
                          two calls per loop
                    -> SISUtils::calcABC
                    -> SISResonator(SISFP,SISCC)::printPeriodicInfo
                    -> SISResonator(SISFP,SISCC)::saveInfo
SISResonator(SISFP)::oneEvolution
        Two way propagation
SISResonator(SISFP)::calcOptOutputs
        -> SISResonator(SISFP)::calcStationaryField
SISResonator(SISCC)::oneEvolution
        Michelson cavity propagation + getReflector->calcOptOutputs
SISResonator(SISCC)::calcStationaryField
```

```
        Handle cases that the source can be from FP arm
```

5.3 Locked field

Lock point is searched until it does not move more than l0/Finesse * lockAccuracy.

```
SISAppFrame
-> SISAppFrame(SISAppFP,SISAppCC)::lockIDCase
        SISAppFrame(SISAppFP)::lockIDCase
              -> lockFP
        SISAppFrame(SISAppCC)::lockIDCase
              -> lockCC
                    -> lockCCHandler
```

6 Function definitions

6.1 Laguerre Gauss

6.1.1 Common

$$\rho = \frac{\sqrt{2}r}{w(z)}$$

$$L_p^m(r) = \frac{e^r r^{-m}}{p!}\frac{d^p}{dr^p}(e^{-r}r^{p+m})]$$

$$r^{-m}L_p^{-m}(r) = (-)^m \frac{(p-m)!}{p!}L_{p-m}^m(r)$$

$$\int_0^\infty e^{-r}r^m L_p^m(r)L_q^m(r)dr = \frac{(p+m)!}{p!}\delta_{pq}$$

$$L_0^m(r) = 1$$

$$L_1^m(r) = (1+m) - r$$

$$L_2^m(r) = \frac{1}{2}\{(2+m)(1+m) - 2(2+m)r + r^2\}$$

$$L_3^m(r) = \frac{1}{6}\{(3+m)(2+m)(1+m) - 3(3+m)(2+m)r + 3(3+m)r^2 - r^3\}$$

$$\iint r\,dr\,d\phi\ \ conj(LG_{pm}(r,\phi))F(r,\phi)$$

$$= C\bullet\int \xi d\xi \exp(i\frac{k}{2R(z)}r^2)\rho^m L_p^m(\rho^2)\int_0^{2\pi} d\phi \exp(-i\cdot m\cdot\phi)F(r,\phi)$$

$$= C\bullet\int \xi d\xi \exp(i\frac{k}{2R(z)}r^2)\rho^m L_p^m(\rho^2)\times$$

$$\int_0^{\pi/2} d\phi[\cos(m\ \phi)\{F(\phi)+F(-\phi)+(-)^m(F(\pi-\phi)+F(\phi-\pi)\}$$

$$-i\ \sin(m\ \phi)\{F(\phi)+F(-\phi)-(-)^m(F(\pi-\phi)+F(\phi-\pi)\}]$$

$$C = w(z)\sqrt{\frac{2}{\pi}}\sqrt{\frac{p!}{(p+m)!}}$$

$$\xi = \exp(-\frac{1}{2}(\frac{r}{w(z)})^2)$$

$$\rho = \frac{\sqrt{2}r}{w(z)} = \sqrt{-4\ln(\xi)}$$

6.1.2 exp(i m ϕ)

$$LG_{pm}(r,\phi,z) = \sqrt{\frac{2}{\pi}}\sqrt{\frac{p!}{(p+m)!}}\frac{1}{w(z)}\exp(i\cdot m\cdot\phi)\rho^m L_p^m(\rho^2)\times$$
$$\exp[-(i\frac{k}{2R(z)}+\frac{1}{w(z)^2})r^2 + i\cdot(1+2p+m)\eta(z)]$$

$$\sqrt{\frac{p!}{(p-m)!}}\rho^{-m}L_p^{-m}(\rho^2)\exp(i\cdot(-m)\cdot\phi)\exp[i\cdot(2p-m+1)\eta]$$

$$= \sqrt{\frac{p!}{(p-m)!}}\rho^{-m}[\rho^{2m}(-)^m\frac{(p-m)!}{p!}L_{p-m}^m(\rho^2)]\exp(i\cdot(-m)\cdot\phi)\exp[i\cdot(2p-m+1)\eta]$$

$$= (-)^m\sqrt{\frac{(p-m)!}{p!}}\rho^m L_{p-m}^m(\rho^2)\exp(i\cdot(-m)\cdot\phi)\exp[i\cdot(2p-m+1)\eta]$$

$$= (-)^m\sqrt{\frac{q!}{(q+m)!}}\rho^m L_q^m(\rho^2)\exp(i\cdot(-m)\cdot\phi)\exp[i\cdot(2q+m+1)\eta]$$

q = p – m (m:1~p ⇒q:0~p-1)

By use the original set of expressions for m ≥0, and by use this conversion of the expression with negative m to positive m for 0 > m ≥ -p, the entire set can be represented using Laguerre functions with only positive m.

$$LG_{pm}(r,\phi,z)=\sqrt{\frac{2}{\pi}}\sqrt{\frac{p!}{(p+m)!}}\frac{1}{w(z)}\exp(i\cdot m\cdot\phi)\rho^{|m|}L_p^{|m|}(\rho^2)\times$$
$$\exp[-(i\frac{k}{2R(z)}+\frac{1}{w(z)^2})r^2+i\cdot(1+2p+|m|)\eta(z)]$$

6.1.3 cos(m ϕ), sin(m ϕ)

$$LG_{pm}(r,\phi,z)=\sqrt{\frac{2}{\pi}}\sqrt{\frac{2}{1+\delta_{0m}}}\sqrt{\frac{p!}{(p+m)!}}\frac{1}{w(z)}\left\{\begin{array}{c}\cos(m\cdot\phi)\\ \sin(m\cdot\phi)\end{array}\right\}\rho^m L_p^m(\rho^2)\times$$
$$\exp[-(i\frac{k}{2R(z)}+\frac{1}{w(z)^2})r^2+i\cdot(1+2p+m)\eta(z)]$$

6.1.4 node number vs LG index

N=2p+|m|	exp(i m ϕ) : |m|≤N	cos(m ϕ), sin(m ϕ) : m ≥ 0
0	(0,0)	(0,c0)
1	(0,1), (0,-1)	(0,c1), (0,s1)
2	(1,0), (0,2), (0,-2)	(1,c0), (0,c2), (0,s2)
3	(1,1), (1,-1), (0,3), (0,-3)	(1,c1), (1,s1), (0,c3), (0,s3)
4	(2,0), (1,2), (1,-2), (0,4), (0,-4)	(2,c0), (1,c2), (1,s2), (0,c4), (0,s4)

cm : cos(m ϕ), sm : sin(m ϕ)

Number of modes with node number = N : N + 1

Number of modes with node number ≤ N : (N + 1)(N + 2) / 2

7 Miscellaneous

7.1 Warnings

7.1.1 Beam size, mirror size and FFT window size

When the mirror aperture is small and the loss assuming a Gaussian distribution (loss = exp(-2 r^2 / w^2)) is larger than 1ppm, a warning is printed.

When the FFT window is smaller than the mirror size and if the power outside of the FFT window (loss = exp(2 $(Wfft/2)^2$ / w^2)) is larger than 0.1ppm, a warning is printed.

8 Example specification files

Examples how to use the code. You don't need to specify most of them. Type "definition" to find the default values and if the default value is good for you, just remove them. All are defined here for documentation purpose.

These are versions as of January 2008. Mostly obsolete and to be replaced

8.1 sisDB_FP.mcr LIGO I FP with data file

```
#print "FP for initial LIGO"

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

% Basic FFT setup
Nfft = 256
Wfft = 0.35

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

% mirror definition = [ aperture, thickness, mech, opt ]

aperture = 0.24
ITM.aperture = aperture
ETM.aperture = aperture

% mirror mechanical data = [ x, y, z, tX, tY, tZ ]
ITM.mech = [ 0, 0, 0, 0, 0, 0 ]
ETM.mech = [ 0, 0, 0, 0, 0, 0 ]

% mirror optoical data = [ T, R, ROC, refrIndex, "HR_phasemap", "trans_phasemap" ]
%
% NOISESPEC( int rand_seed, double rms, int power, vector<int> WykoIndex )
% int rand_seed = 0 to generate seed randomly using time,
% -1 to generate a random number using last seed, -2 to use the last seed.
% psd = f^-(2 power)
% vector<int> WykoIndex : subtract Zernike polynomial specified by the array
% If WykoIndex[0] = -N, all Zernile polynomials with Wyko index <= N subtracted.
%
% THERMOELASTIC( beamSize, Psubs, Pcoat [, T0] ) for surface deformation
% THERMALPHASE( beamSize, Psubs, Pcoat [, T0] ) for thermal lens in substrate
%
%     Psubs : power absorbed in substrate (0.084W), if negative, it is used as an
absorption rate (2E-6*100 m^-1)
%     Pcoat : power absorbed in coating (0.425W), if negative, it is an absorption
rate (0.5E-6)
%

ITM.opt.T = 0.027
ITM.opt.R = 1 - ITM.opt.T
ITM.opt.ROC = 14.76e3
ITM.opt.refrIndex = SIO2_REFRACTIVE_INDEX
ITM.opt.HR_phase = DATAFILE( "4ITM03-180-C92401-TP.asc.txt", 1, 70 );

ETM.opt.T = 7e-6
ETM.opt.R = 1 - ETM.opt.T
ETM.opt.ROC = 8.73e3
ETM.opt.refrIndex = SIO2_REFRACTIVE_INDEX
ETM.oscillation.amplitude = 1e-15

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

```
% L0 is macro length, rounted to integer multiple of NdYAG wave length
% delL is a microscopic correct
% total length = lambda(NdYAG) * floor( L0 / lambda ) + delL
% set matchToInput to 1 to adjust delL to mode match to the input beam

% cavity definition = [ L0, delL, matchToInput, Nfft, Ncut1, Ncut2, lockFP,
adaptiveGrid ]
FPcav = [ 3995.420, 0, 3, 0, 0, 0, 1, 0 ]

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

% input beam specification
% use matchToCavity to set the input beam waist size and position
% input type = "LG" for LaguerreGuass( index1, index2 )
%              "HG" for HermiteGauss( index1, index2 )
% inputBeam definition = [ "BeamType", power, index1, index2, waistSize,
waistPosition, matchToCavity ]

inputBeam.beamType = "HG"
inputBeam.power = 1
inputBeam.index1 = 0
inputBeam.index2 = 0
inputBeam.waistSize = wROC2waistSize( beamSize, ROC )
inputBeam.waistPosition = wROC2dist2waist( beamSize, ROC )
inputBeam.matchToCavity = 1
```

8.2 sisDB_FP.mcr AdvLIGO FP with thermal deformation

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

#print "Advanced LIGO FP arm simulation with thermal effect and ring heater
compensation"

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

RH_NO_CASE = 0   % no ring heater
RH_AT_ETM  = 1   % only at ETM
RH_AT_BOTH = 2   % both at ETM and ITM

#defaults

% ITM / ETM mirror
ASYM_CASE = 1
aperture = 0.34
T_ITM = 0.014
T_ETM = 30e-6

% RH option
RH_CASE = RH_NO_CASE
```

```
% aberration RMS
NOISE_RMS = 0.7e-9
% see for ITM/ETM random surface
ITM_SEED = 12345
ETM_SEED = 54321

#enddef

#if ASYM_CASE
% new asymmetric design
ITM_ROC_COLD = 1971
ETM_ROC_COLD = 2191
#else
% symmetric design
ITM_ROC_COLD = 2076
ETM_ROC_COLD = 2076
#endif

#if RH_CASE == RH_NO_CASE
ITM_ROC_HOT  = ITM_ROC_COLD
ETM_ROC_HOT  = ETM_ROC_COLD
#elseif RH_CASE == RH_AT_ETM
ITM_ROC_HOT  = ITM_ROC_COLD
ETM_ROC_HOT  = ETM_ROC_COLD - 50
#else
ITM_ROC_HOT  = ITM_ROC_COLD - 30
ETM_ROC_HOT  = ETM_ROC_COLD - 30
#endif

#print RH_CASE ITM_ROC_COLD ITM_ROC_HOT ETM_ROC_COLD ETM_ROC_HOT

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

% default FFT setup
Nfft = 256
Wfft = 0.70

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

% mirror definition = [ aperture, thickness, mech, opt ]

ITM.aperture = aperture
ETM.aperture = aperture

% mirror mechanical data = [ x, y, z, tX, tY, tZ ]
ITM.mech = [ 0, 0, 0, 0, 0, 0 ]
ETM.mech = [ 0, 0, 0, 0, 0, 0 ]

% mirror optoical data = [ T, R, ROC, refrIndex, "HR_phasemap", "trans_phasemap" ]
%
% NOISESPEC( int rand_seed, double rms, int power, vector<int> WykoIndex )
% int rand_seed = 0 to generate seed randomly using time,
% -1 to generate a random number using last seed, -2 to use the last seed.
```

```
% psd = f^-(2 power)
% vector<int> WykoIndex : subtract Zernike polynomial specified by the array
% If WykoIndex[0] = -N, all Zernile polynomials with Wyko index <= N subtracted.
%
% THERMOELASTIC( beamSize, Psubs, Pcoat [, T0] ) for surface deformation
% THERMALPHASE( beamSize, Psubs, Pcoat [, T0] ) for thermal lens in substrate
%
%    Psubs : power absorbed in substrate (1.7e-4W / 0.084), if negative, it is used
as an absorption rate ( 0.4e-6 m^-1 / 2E-6*100 m^-1)
%        ( low OH substrate / orignal substrate )
%     Pcoat : power absorbed in coating (0.425W), if negative, it is an absorption
rate (0.5E-6)
%

T_ITM = 0.014
powerIn = 800e3 * T_ITM / 4

%parameters for thermal deformation
beamWidth = 0.06
PsubsPwr = 0.084 * 0.4 / 200
PcoatPwr = 0.425
PsubsRate = 0.4e-6
PcoatRate = 0.5e-6

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

% ITM specifications

ITM.opt.T = T_ITM
ITM.opt.R = 1 - ITM.opt.T
ITM.opt.ROC = ITM_ROC_COLD
ITM.oscillation.amplitude = 1e-15
ITM.oscillation.phase = PI
ITM.opt.refrIndex = SIO2_REFRACTIVE_INDEX
ITM.opt.HR_phase = \
NOISESPEC( ITM_SEED, NOISE_RMS, 1, -5 )
%   THERMOELASTIC( -beamWidth, -PsubsRate, -PcoatRate ) \
% + rr / 2 * (1/ITM_ROC_HOT - 1/ITM_ROC_COLD) + \
% + NOISESPEC( ITM_SEED, NOISE_RMS, 1, -5 )
% end of HR_phase. this line needed as a termination line because above lines end
with "\"
% ITM.opt.trans_phase = THERMALPHASE( -beamWidth, -PsubsRate, -PcoatRate )

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

% ETM specifications

ETM.opt.T = T_ETM
ETM.opt.R = 1 - ETM.opt.T
ETM.opt.ROC = ETM_ROC_COLD
ETM.oscillation.amplitude = 1e-15
ETM.oscillation.phase = 0
ETM.opt.refrIndex = SIO2_REFRACTIVE_INDEX
```

```
ETM.opt.HR_phase = \
NOISESPEC( ETM_SEED, NOISE_RMS, 1, -5 )
%  THERMOELASTIC( -beamWidth, -PsubsRate, -PcoatRate ) \
%+ rr / 2 * (1/ETM_ROC_HOT - 1/ETM_ROC_COLD) \
%+ NOISESPEC( ETM_SEED, NOISE_RMS, 1, -5 ) \
% end of HR_phase. this line needed as a termination line because above lines end
with "\"
% ETM.opt.HR_phase = DATAFILE( “thermalDeform_N256_W700.txt” )
% ETM.opt.trans_phase = THERMALPHASE( beamWidth, PsubsPwr, PcoatPwr )

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

% FP cavity specifications

% L0 is macro length, rounted to integer multiple of NdYAG wave length
% delL is a microscopic correct
% total length = lambda(NdYAG) * floor( L0 / lambda ) + delL
% set matchToInput to 3 to adjust delL to mode match to the input beam
% This value has the same meaning as that for matchToCavity below.
% This is used to make the first guess of the cavity length for ease of locking,
% so use 0 (no adjustment) or 3 (best guess).

% cavity definition = [ L0, delL, matchToInput, Nfft, Ncut1, Ncut2, lockFP ]
FPcav = [ 3995.420, 0, 3, 0, 0, 0, 1 ]

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

% input beam specification
% use matchToCavity to set the input beam waist size and position
% input type = "LG" for LaguerreGuass( index1, index2 )
%              "HG" for HermiteGauss( index1, index2 )
%  inputBeam  definition  =  [  "BeamType",  power,  index1,  index2,  waistSize,
waistPosition, matchToCavity ]
% matchToCavity
%   0 : Use the provided waistSize and waistPosition values
%   1 : use opt.ROC values of mirrors to find the mode
%   2 : use mirror maps, build using HR_phase etc, to find the mode
%   3 : use gaussian beam weighted map to find the mode
%    -1, -2, -3 : Calculate the mode for the first run and do not update unti this
value is changed

inputBeam.beamType = "HG"
inputBeam.power = powerIn
inputBeam.index1 = 0
inputBeam.index2 = 0
inputBeam.waistSize = 0.0080636845404989
inputBeam.waistPosition = -1405.871235801
inputBeam.matchToCavity = -1
```

8.3 sisDB_MSCC.mcr AdvLIGO coupled cavity with marginally stable cavity

```
#print "Coupled Cavity with cold mirrors"

#defaults
```

```
BX = 0.224
BY = 0.259

T_ETM = 34e-6  % 74.55e-6
T_ITM = 0.014
T_SRM = 0.20
T_PRM = 0.036
FPAperture = 0.34
MC_SCALE = 1
Config = 2

% aberration RMS
NOISE_RMS = 0.7e-9
% see for ITM/ETM random surface
ITM_SEED = 12345
ETM_SEED = 54321
#enddef

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Basic FFT setup

Nfft = 512
Wfft = 0.70

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% mirror definition = [ aperture, thickness, mech, opt ]

%%% Coupled Cavity %%%%%%%%
% config = 0 : PRM - BS - ITMX
%        = 1 : RRM - BS - ITMY
%        = 2 : SRM - BS - ITMX
%        = 3 : SRM - BS - ITMY
CCav.RM2BS = 3.318
CCav.BS2ITM = 5.261
CCav.delRM = 5.5000966812588e-7
CCav.matchToInput = 0

ITM.opt.ROC = 1971
ETM.opt.ROC = 2191

LossRM = 0
CCav.config = Config

#if Config == 0 || Config == 1
RM.opt.T = T_PRM
#else
RM.opt.T = T_SRM
#endif

RM.opt.R = 1 - RM.opt.T - LossRM
RM.opt.ROC = 1368

%%% ETM %%%%%%%%
```

```
ETM.aperture = FPAperture
ETM.thickness = 0.2
ETM.mech.y = 0
ETM.mech.z = 0
ETM.mech.tX = 0
ETM.mech.tY = 0
ETM.mech.tZ = 0
ETM.opt.T = T_ETM
ETM.opt.R = 1 - ETM.opt.T
ETM.opt.refrIndex = 1.44963
ETM.opt.HR_phase = NOISESPEC( ITM_SEED, NOISE_RMS, 1, -5 )
ETM.opt.trans_phase =
ETM.opt.trans_loss =
ETM.opt.AR_trans =
ETM.oscillation.amplitude = 1e-15
ETM.oscillation.phase = 0

%%% ITM %%%%%%%%

ITM.aperture = FPAperture
ITM.thickness = 0.2
ITM.mech.x = 0
ITM.mech.y = 0
ITM.mech.z = 0
ITM.mech.tX = 0
ITM.mech.tY = 0
ITM.mech.tZ = 0
ITM.opt.T = T_ITM
ITM.opt.R = 1 - ITM.opt.T
ITM.opt.refrIndex = 1.44963
ITM.opt.HR_phase = NOISESPEC( ETM_SEED, NOISE_RMS, 1, -5 )
ITM.opt.trans_phase =
ITM.opt.trans_loss =
ITM.opt.AR_trans = if ( 4*( x*x / (BX*BX) + y*y / (BY*BY) ) < 1, 1, 0 )
ITM.oscillation.amplitude = 1e-15
ITM.oscillation.phase = PI

%%% BS %%%%%%%%

BS.aperture = 0.37 * MC_SCALE
BS.thickness = 0.06
BS.mech.x = 0
BS.mech.y = 0
BS.mech.z = 0
BS.mech.tX = 0
BS.mech.tY = 0
BS.mech.tZ = 0
BS.opt.T = if(CCav.config == 0 || CCav.config==3, 1, 0 )
BS.opt.R = 1 - BS.opt.T
BS.opt.ROC = 1e+20
BS.opt.refrIndex = 1.44963
BS.opt.HR_phase =
```

```
BS.opt.trans_phase =
BS.opt.trans_loss =
BS.opt.AR_trans =
BS.centerOffset = 1000
BS.flat = 0.00

%%% RM %%%%%%%%

RM.aperture = 0.26 * MC_SCALE
RM.thickness = 0.2
RM.mech.x = 0
RM.mech.y = 0
RM.mech.z = 0
RM.mech.tX = 0
RM.mech.tY = 0
RM.mech.tZ = 0
RM.opt.refrIndex = 1.44963
RM.opt.HR_phase =
RM.opt.trans_phase =
RM.opt.trans_loss =
RM.opt.AR_trans =

%%% FP cavity %%%%%%%%

FPcav.L0 = 3994.75
FPcav.delL = 0
FPcav.matchToInput = 3   % same as inputBeam.matchToCavity
FPcav.Ncutoff1 = 0
FPcav.Ncutoff2 = 0

%%% Input beam %%%%%%%%
% matchToCavity = 0 : use the specified value
%               = 1 : couple to cold FP arm (use curvatures given in the spec)
%                   = 2 : copuld to hot FP arm (use curvatures including surface
deformation)
inputBeam.beamType = LG
inputBeam.power = 1
inputBeam.RF = 0
inputBeam.index1 = 0
inputBeam.index2 = 0
inputBeam.waistSize = 0.01
inputBeam.waistPosition = -2000
inputBeam.matchToCavity = 1
```

8.4 sisDB_SCC.mcr AdvLIGO coupled cavity with stable cavity

```
% Config = -1 : symmatric cavity with ROC = 2076-2076 for PRM - BS - ITMY cavity
%        =  0 : PRM - BS - ITMX
%        =  1 : RRM - BS - ITMY
%        =  2 : SRM - BS - ITMX
%        =  3 : SRM - BS - ITMY
% design = 1 or 2 for PRM

#defaults
```

```
ITM_SEED = 12345
ETM_SEED = 54321
NOISE_RMS = 0.7e-9

BX = 0.224
BY = 0.259

Config = 2
design = 2

CCav.MMT2_MMT3 = -16.6767

% BS and MMT3 aperture scaled by this factor
MirrorScale = 1
#enddef

ETM.opt.T = 74.55e-6
% ETM.opt.T = 34e-6
ITM.opt.T = 0.014
T_SRM = 0.20
T_PRM = 0.036

PsubsRate = 2E-6*100
PcoatRate = 0.5e-6

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Basic FFT setup

Nfft = 1024
Wfft = 0.70
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

FPcav.Nfft = 512
CCav.NfftProp23 = 2048
CCav.RM_beamSize = 0  % use RM curvature to adjust L_MMT2_MMT3

%%% Coupled Cavity %%%%%%%%

% default FP ROCs
ITM.opt.ROC = 1971
ETM.opt.ROC = 2191
FPcav.delL = 0
FPcav.matchToInput = 1

% two PRM designs
#if design == 1
MMT2_ROC = -2.325
RM_ROC = 9.6884
#else
MMT2_ROC = -2.335
RM_ROC = 8.9521
#endif
```

```
CCav.BS2ITM = 4.5
CCav.MMT3_BS = 20.655
CCav.RM_MMT2 = 16.5858
MMT3.opt.ROC = 35.048

CCav.config = Config
CCav.matchToInput = 1
CCav.delRM = 0

% PRM <=> FPX
#if Config == 0

MMT2.opt.ROC = MMT2_ROC
RM.opt.ROC = RM_ROC
RM.opt.T = T_PRM
RM.opt.R = 1 - RM.opt.T
FPcav.matchToInput = 0
CCav.matchToInput = 0
#if design == 1
FPcav.delL = 4.6354277991395e-07
CCav.delRM = -1.7883922618696e-07
#else
FPcav.delL = 4.6354277556758e-07
CCav.delRM = -1.6122892291616e-07
#endif

#elseif Config == 1

MMT2.opt.ROC = MMT2_ROC
RM.opt.ROC = RM_ROC
RM.opt.T = T_PRM
RM.opt.R = 1 - RM.opt.T
FPcav.matchToInput = 0
CCav.matchToInput = 0
#if design == 1
FPcav.delL = 4.6354277991395e-07
CCav.delRM = -1.966158e-07
#else
FPcav.delL = 4.6354277567406e-07
CCav.delRM = -1.7899725204988e-07
#endif

% SRM <=> FPX
#elseif Config == 2

MMT2.opt.ROC = -2.155
RM.opt.ROC = 107.29
RM.opt.T = T_SRM
RM.opt.R = 1 - RM.opt.T
FPcav.matchToInput = 0
CCav.matchToInput = 0
FPcav.delL = 4.6354277570509e-07
CCav.delRM = 1.2150533021935e-07
```

```
% SRM <=> FPY
#elseif Config == 3

MMT2.opt.ROC = -2.155
RM.opt.ROC = 107.29
RM.opt.T = T_SRM
RM.opt.R = 1 - RM.opt.T
FPcav.matchToInput = 0
CCav.matchToInput = 0
FPcav.delL = 4.6354277608691e-07
CCav.delRM = 1.0373709511164e-07

#else

CCav.config = 1
ITM.opt.ROC = 2076
ETM.opt.ROC = 2076
MMT3.opt.ROC = 31.059
MMT2.opt.ROC = 1.856
RM.opt.ROC = -76.46
CCav.delRM = 1.2577e-7

#endif

#print ITM.opt.ROC  ETM.opt.ROC MirrorScale RM.opt.ROC CCav.delRM

% use for ring heater
ETM.opt.HR_phase = ""

% Useful parameters

FPAperture = 0.34

%%% FP cavity %%%%%%%%

FPcav.L0 = 3994.75
FPcav.Ncutoff1 = 0
FPcav.Ncutoff2 = 0

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% mirror definition = [ aperture, thickness, mech, opt ]
% mech = x, y, z, tX, tY, tZ

%%% ETM %%%%%%%%

ETM.aperture = FPAperture
ETM.thickness = 0.2
ETM.mech = [0,0,0,0,0,0]
ETM.opt.R = 1 - ETM.opt.T
ETM.opt.refrIndex = 1.44963
%ETM.opt.HR_phase = NOISESPEC( ETM_SEED, NOISE_RMS, 1, -5 )
ETM.opt.trans_phase =
```

```
ETM.opt.trans_loss =
ETM.opt.AR_trans =
ETM.oscillation.amplitude = 1e-15
ETM.oscillation.phase = 0

%%% ITM %%%%%%%%

ITM.aperture = FPAperture
ITM.thickness = 0.2
ITM.mech = [0,0,0,0,0,0]
% ITM.opt.T = 0.01
ITM.opt.R = 1 - ITM.opt.T
ITM.opt.refrIndex = 1.44963
% ITM.opt.HR_phase = NOISESPEC( ITM_SEED, NOISE_RMS, 1, -5 )
ITM.opt.trans_phase =
ITM.opt.trans_loss =
ITM.opt.AR_trans = if ( 4*( x*x / (BX*BX) + y*y / (BY*BY) ) < 1, 1, 0 )
ITM.oscillation.amplitude = 1e-15
ITM.oscillation.phase = PI

%%% BS %%%%%%%%

BS.aperture = 0.37 * MirrorScale
BS.thickness = 0.06
BS.mech = [0,0,0,0,0,0]
BS.opt.T = if(CCav.config == 0 || CCav.config==3, 1, 0 )
BS.opt.R = 1 - BS.opt.T
BS.opt.ROC = 1e+20
BS.opt.refrIndex = 1.44963
BS.opt.HR_phase =
BS.opt.trans_phase =
BS.opt.trans_loss =
BS.opt.AR_trans =
BS.centerOffset = 1000
BS.flat = 0

%%% MMT3 %%%%%%%%

MMT3.aperture = 0.26 * MirrorScale
MMT3.thickness = 0.2
MMT3.mech = [0,0,0,0,0,0]
MMT3.opt.T = 0
MMT3.opt.R = 1
MMT3.opt.refrIndex = 1.44963
MMT3.opt.HR_phase =
MMT3.opt.trans_phase =
MMT3.opt.trans_loss =
MMT3.opt.AR_trans =

%%% MMT2 %%%%%%%%

MMT2Size = 0.06
MMT2.aperture = 0.26
```

```
MMT2.aperture = MMT2Size * 2 / 3
MMT2.thickness = 0.2
MMT2.Wfft = MMT2Size
MMT2.mech = [0,0,0,0,0,0]
MMT2.opt.T = 0
MMT2.opt.R = 1
MMT2.opt.refrIndex = 1.44963
MMT2.opt.HR_phase =
MMT2.opt.trans_phase =
MMT2.opt.trans_loss =
MMT2.opt.AR_trans =

%%% RM %%%%%%%%

RM.aperture = 0.07
RM.thickness = 0.2
RM.Wfft = 0.10
RM.mech = [0,0,0,0,0,0]
RM.opt.R = 1 - RM.opt.T
RM.opt.refrIndex = 1.44963
% power in Michelson cavity = power in the arm / ( Finesse * 2 / PI )
% PwrRM = 800e3 / (PI/2)
% RM.opt.HR_phase =  800e3 / (450 * 2 / PI ) * THERMOELASTIC( 0.002, 0, PcoatRate )
RM.opt.trans_phase =
RM.opt.trans_loss =
RM.opt.AR_trans =

%%% Input beam %%%%%%%%
% matchToCavity = 0 : use the specified value
%               = 1 : couple to cold FP arm (use curvatures given in the spec)
%                   = 2 : copuld to hot FP arm (use curvatures including surface
deformation)
inputBeam.beamType = LG
inputBeam.power = 1
inputBeam.RF = 0
inputBeam.index1 = 0
inputBeam.index2 = 0
inputBeam.waistSize = 0.01
inputBeam.waistPosition = -2000
inputBeam.matchToCavity = 1
```

9 Objects and variables

9.1 SCC

9.1.1 Field index

0.fromRM	1.toBSfromRMorMMT	2.fromBStoITM
3.toITMfromBS	4.fromITMtoBS	5.toBSfromITM
6.fromBStoRMorMMT	7.toRM	8.inputBeam
9.totalREFLfromRM	10.leakREFLfromRM	11.promptREFLfromRM
12.promptREFLfromITM	13.fromITMtoETM	14.toETMfromITM
15.fromETMtoITM	16.toITMfromETM	17.transmitted
18.toMMT2fromRM	19.fromMMT2toMMT3	20.toMMT3fromMMT2
21.fromMMT3toBS	22.toMMT3fromBS	23.fromMMT3toMMT2
24.toMMT2fromMMT3	25.fromMMT2toRM	

9.1.2 summary file example

```
Summary of FP cavity "FPcav"

cavity length = 4.6009048921436e-07 = 4.600905007607e-07(delL) - 0(ITMz) - 1.1546333643549e-14(ETMz)
power in cavity = 4168.3615958045, reflected power = 14.540713468391, leakPower / Power00 = 14.540713468391 / 14.53467696518
Diffractive loss = 0.86595485482288, total loss = 3.8109067558301

Summary of Coupled cavity "CCav"

cavity length = 9.083003897165e-08 = 7.688e-08(delL_RM) - -1.395003897165e-08(RMz) + 0(n*ITMz)
Beam splitter wedge = 0.04 degree, which magnifies 1.0008579124112 in inline and 0.99913926986321 in offline from HR to AR side
power in cavity = 14.857463549585, reflected power = 0.65640620958983

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
   FP cavity modal analysis using cold optics parameters

ROC(ITM) = 1934, ROC(ETM) = 2245, Cavity lenggth = 3994.499999672
Fval(ITM) = -4301.3143743072, OPL(ITM) = 0.22764414887384, Fval(ETM) = -4992.994269126
waist size = 0.01203704073212, waist position from ITM = 1834.2198819617, Rayleigh range = 427.80682127602

Mode parameters of cavity fields
ETM AR (out base) : w = 0.0619634      R = 1548.671       z  = 1520.299       z0 = 207.6871       w0 = 0.008386884
       (out fit ) : NULL Field
ETM HR (in  base) : w = 0.0619634      R = 2245           z  = 2160.28        z0 = 427.8068       w0 = 0.01203704
       (in  fit ) : (wX,wY)=(0.061951, 0.061948) R(x/y) = ( 2245.196,  2245.199) (x0,y0)=( 0         ,  0         )  power / HMfrac =
4168.362 / 5.111e-07
       (out fit ) : (wX,wY)=(0.061951, 0.061948) R(x/y) = (-2244.804, -2244.801) (x0,y0)=( 0         ,  0         )  power / HMfrac =
4168.192 / 7.195e-07
ITM HR (in  base) : w = 0.05299391     R = 1934           z  = 1834.22        z0 = 427.8068       w0 = 0.01203704
       (in  fit ) : (wX,wY)=(0.053004, 0.052998) R(x/y) = ( 1933.94 ,  1933.946) (x0,y0)=( 0         ,  0         )  power / HMfrac =
4168.192 / 7.195e-07
       (out fit ) : (wX,wY)=(0.053004, 0.052997) R(x/y) = (-1934.118, -1934.174) (x0,y0)=( 0         ,  0         )  power / HMfrac =
4168.362 / 5.111e-07
ITM AR (in  base) : w = 0.05300295     R = -1334.355      z  = -1300.696      z0 = 209.2372       w0 = 0.008418125
       (in  fit ) : (wX,wY)=(0.052986, 0.052906) R(x/y) = (-1338.063, -1342.353) (x0,y0)=( 0         ,  0         )  power / HMfrac =
14.85746 / 0.0004086
       (out fit ) : (wX,wY)=( 0.05304, 0.053172) R(x/y) = ( 1337.955,  1342.128) (x0,y0)=( 0         ,  0         )  power / HMfrac =
14.54071 / 0.0004151
----------------------------------------------------------------------------------------------------------------------

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
   Coupled cavity modal analysis for Stable cavity configuration  using cold optics parameters

ROC(RM) = -5.6938       , ROC(MMT2) = -6.427        , ROC(MMT3) = 36
Length : RM-MMT2 = 15.726, MMT2-MMT3 = 15.460699886708, MMT3-BS = 19.368, BS-ITM = 4.8096
GouyPhase : RM-MMT2 = 0.31669677759007, MMT2-MMT3 = 0.011792112582659, MMT3-ITM = 0.0041864295672143, Total = 0.33267531973995
OPL(RM) = 0.05173705340095, OPL(BS) = 0.094825104568678
Fval(RM) = 12.663300936325
Mode parameters of cavity fields
```

```
ITM AR       (in  base) : w = 0.05300295     R = -1334.355      z  = -1300.696      z0 = 209.2372       w0 = 0.008418125
             (in  fit ) : (wX,wY)=(0.053024, 0.052884) R(x/y) = (-1337.256, -1342.226) (x0,y0)=( 0          ,  0          )  power /
HMfrac = 14.85746 / 0.0004032
             (out fit ) : (wX,wY)=(0.053013, 0.053168) R(x/y) = ( 1337.674,  1342.153) (x0,y0)=( 0          ,  0          )  power /
HMfrac = 14.53949 / 0.0004158
BS AR        (in  base) : w = 0.05319401     R = 1339.041       z  = 1305.506       z0 = 209.2372       w0 = 0.008418125
             (in  fit ) : (wX,wY)=( 0.05322, 0.053352) R(x/y) = ( 1342.741,  1347.225) (x0,y0)=( 0          ,  0          )  power /
HMfrac = 14.53949 / 0.0004235
             (out fit ) : (wX,wY)=(0.053209, 0.053083) R(x/y) = (-1342.366, -1346.874) (x0,y0)=( 0          ,  0          )  power /
HMfrac = 14.85746 / 0.000409
BS HR        (in  base) : w = 0.05319777     R = -1339.133      z  = -1305.601      z0 = 209.2372       w0 = 0.008418125
             (in  fit ) : (wX,wY)=(0.053165, 0.053083) R(x/y) = (-1337.716, -1346.874) (x0,y0)=( 0          ,  0          )  power /
HMfrac = 14.8677 / 0.000452
             (out fit ) : (wX,wY)=(0.053066, 0.053349) R(x/y) = ( 1338.009,  1347.205) (x0,y0)=( 0          ,  0          )  power /
HMfrac = 14.52527 / 0.0005348
MMT3 -> BS   (in  base) : w = 0.05396732     R = 1358.011       z  = 1324.969       z0 = 209.2372       w0 = 0.008418125
             (in  fit ) : (wX,wY)=(0.053843,  0.05408) R(x/y) = ( 1356.9  ,  1366.352) (x0,y0)=( 0          ,  0          )  power /
HMfrac = 14.52527 / 0.0005348
             (out fit ) : (wX,wY)=( 0.05393, 0.053901) R(x/y) = (-1356.517, -1365.221) (x0,y0)=( 0          ,  0          )  power /
HMfrac = 14.8677 / 0.000452
MMT3 -> MMT2 (in  base) : w = 0.05396732     R = 18.24179       z  = 18.24171       z0 = 0.03869572     w0 = 0.0001144794
             (in  fit ) : (wX,wY)=(0.054562, 0.054096) R(x/y) = ( 18.31708,  18.31532) (x0,y0)=( 0          ,  0          )  power /
HMfrac = 14.86845 / 0.0001942
             (out fit ) : (wX,wY)=(0.053107, 0.053614) R(x/y) = (-18.31705, -18.3151 ) (x0,y0)=( 0          ,  0          )  power /
HMfrac = 14.52508 / 0.0002455
MMT2 -> MMT3 (in  base) : w = 0.00822827     R = -2.781545      z  = -2.781007      z0 = 0.03869572     w0 = 0.0001144794
             (in  fit ) : (wX,wY)=(0.0081937, 0.0082386) R(x/y) = (-2.780078, -2.781758) (x0,y0)=( 0          ,  0          )  power /
HMfrac = 14.525 / 0.0002441
             (out fit ) : (wX,wY)=(0.0082067, 0.0082433) R(x/y) = ( 2.780138,  2.782084) (x0,y0)=( 0          ,  0          )  power /
HMfrac = 14.86859 / 0.0001946
MMT2 -> RM   (in  base) : w = 0.00822827     R = 20.69314       z  = 20.47376       z0 = 2.119331       w0 = 0.0008472182
             (in  fit ) : (wX,wY)=(0.0082059, 0.0082434) R(x/y) = ( 20.62998,  20.70724) (x0,y0)=( 0          ,  0          )  power /
HMfrac = 14.86859 / 0.0001747
             (out fit ) : (wX,wY)=(0.0081949, 0.0082387) R(x/y) = (-20.62716, -20.6903 ) (x0,y0)=( 0          ,  0          )  power /
HMfrac = 14.525 / 0.0002235
RM HR        (in  base) : w = 0.002078462    R = -5.6938        z  = -4.747762      z0 = 2.119331       w0 = 0.0008472182
             (in  fit ) : (wX,wY)=(0.0020451, 0.002084) R(x/y) = (-5.680142, -5.677736) (x0,y0)=( 0          ,  0          )  power /
HMfrac = 14.525 / 0.0002235
             (out fit ) : (wX,wY)=(0.002049, 0.0020833) R(x/y) = ( 5.70585 ,  5.708103) (x0,y0)=( 0          ,  0          )  power /
HMfrac = 14.86859 / 0.0001747
RM AR        (in  base) : w = 0.002078462    R = 3.927761       z  = 3.587582       z0 = 1.104726       w0 = 0.0006116784
             (in  fit ) : w = 0.002078462    R = 3.927761       (x0,y0)=( 0          ,  0          )  power / HMfrac = 1 / 3.116e-13
             (out fit ) : (wX,wY)=(0.0020156, 0.0020901) R(x/y) = (-3.908724, -3.912966) (x0,y0)=( 0          ,  0          )  power /
HMfrac = 0.6564062 / 0.000989
-------------------------------------------------------------------------------------------------------------------

***** Object definitions *****

Printing Values for TELESCOPE
**************************
**************************

Printing Values for ITM
**************************
ITM.aperture = 0.336
ITM.thickness = 0.33
ITM.Wfft = 0
ITM.mech.x = 0
ITM.mech.y = 0
ITM.mech.z = 0
ITM.mech.tX = 0
ITM.mech.tY = 0
ITM.mech.tZ = 0
ITM.opt.T = T_ITM
ITM.opt.R = 1 - ITM.opt.T - extraLoss / 2
ITM.opt.ROC = 1934
ITM.opt.refrIndex = 1.44963
ITM.opt.HR_phase = 0
ITM.opt.trans_phase = 0
ITM.opt.trans_loss = (null)
ITM.opt.AR_trans = 1
ITM.oscillation.amplitude = 1e-15
ITM.oscillation.phase = 3.1415926535898
ITM.incident.theta = 0
ITM.incident.phi = 0
**************************

Printing Values for ETM
**************************
ETM.aperture = 0.336
```

```
ETM.thickness = 0.2
ETM.Wfft = 0
ETM.mech.x = 0
ETM.mech.y = 0
ETM.mech.z = 0
ETM.mech.tX = 0
ETM.mech.tY = 0
ETM.mech.tZ = 0
ETM.opt.T = T_ETM
ETM.opt.R = 1 - ETM.opt.T - extraLoss/2
ETM.opt.ROC = 2245
ETM.opt.refrIndex = 1.44963
ETM.opt.HR_phase = 0
ETM.opt.trans_phase = (null)
ETM.opt.trans_loss = (null)
ETM.opt.AR_trans = (null)
ETM.oscillation.amplitude = 1e-15
ETM.oscillation.phase = 0
ETM.incident.theta = 0
ETM.incident.phi = 0
**************************

Printing Values for FPcav
**************************
FPcav.L0 = 3994.5
FPcav.delL = 0
FPcav.matchToInput = 3
FPcav.Nfft = 512
FPcav.Ncutoff1 = 0
FPcav.Ncutoff2 = 0
FPcav.lockFP = 1
FPcav.adaptiveGrid = 0
**************************

Printing Values for BS
**************************
BS.aperture = 0.366
BS.thickness = 0.06
BS.Wfft = 0
BS.mech.x = 0
BS.mech.y = 0
BS.mech.z = 0
BS.mech.tX = 0
BS.mech.tY = 0
BS.mech.tZ = 0
BS.opt.T = BS_T
BS.opt.R = 1 - BS.opt.T
BS.opt.ROC = 1e+20
BS.opt.refrIndex = 1.44963
BS.opt.HR_phase = (null)
BS.opt.trans_phase = (null)
BS.opt.trans_loss = (null)
BS.opt.AR_trans = (null)
BS.oscillation.amplitude = (null)
BS.oscillation.phase = 0
BS.incident.theta = 0
BS.incident.phi = 0
BS.centerOffset = 1000
BS.flat = 0
BS.WedgeAngle = 0.00069813170079773
BS.baffleX = 2.1
BS.baffleXOffset = 0
BS.baffleY = 2.6
BS.baffleYOffset = 0
**************************

Printing Values for RM
**************************
RM.aperture = 0.15
RM.thickness = 0.075
RM.Wfft = 0.025
RM.mech.x = 0
RM.mech.y = 0
RM.mech.z = 0
RM.mech.tX = 0
RM.mech.tY = 0
RM.mech.tZ = 0
RM.opt.T = T_RM
RM.opt.R = 1 - RM.opt.T
RM.opt.ROC = -5.6938
```

```
RM.opt.refrIndex = 1.44963
RM.opt.HR_phase = (null)
RM.opt.trans_phase = (null)
RM.opt.trans_loss = (null)
RM.opt.AR_trans = (null)
RM.oscillation.amplitude = (null)
RM.oscillation.phase = 0
RM.incident.theta = 0
RM.incident.phi = 0
**************************

Printing Values for MMT2
**************************
MMT2.aperture = 0.15
MMT2.thickness = 0.075
MMT2.Wfft = 0.08
MMT2.mech.x = 0
MMT2.mech.y = 0
MMT2.mech.z = 0
MMT2.mech.tX = 0
MMT2.mech.tY = 0
MMT2.mech.tZ = 0
MMT2.opt.T = 0
MMT2.opt.R = 1
MMT2.opt.ROC = -6.427
MMT2.opt.refrIndex = 1.44963
MMT2.opt.HR_phase = (null)
MMT2.opt.trans_phase = (null)
MMT2.opt.trans_loss = (null)
MMT2.opt.AR_trans = (null)
MMT2.oscillation.amplitude = (null)
MMT2.oscillation.phase = 0
MMT2.incident.theta = 0.015184364492351
MMT2.incident.phi = 0
**************************

Printing Values for MMT3
**************************
MMT3.aperture = 0.261
MMT3.thickness = 0.1014
MMT3.Wfft = 0
MMT3.mech.x = 0
MMT3.mech.y = 0
MMT3.mech.z = 0
MMT3.mech.tX = 0
MMT3.mech.tY = 0
MMT3.mech.tZ = 0
MMT3.opt.T = 0
MMT3.opt.R = 1
MMT3.opt.ROC = 36
MMT3.opt.refrIndex = 1.44963
MMT3.opt.HR_phase = (null)
MMT3.opt.trans_phase = (null)
MMT3.opt.trans_loss = (null)
MMT3.opt.AR_trans = (null)
MMT3.oscillation.amplitude = (null)
MMT3.oscillation.phase = 0
MMT3.incident.theta = 0.013700834628155
MMT3.incident.phi = 0
**************************

Printing Values for CCav
**************************
CCav.RM_MMT2 = 15.726
CCav.MMT2_MMT3 = 15.460699886708
CCav.MMT3_BS = 19.368
CCav.NfftProp23 = 2048
CCav.RM_beamSize = 0
CCav.delRM = 7.688e-08
CCav.BS2ITM = 4.8096
CCav.config = 2
CCav.matchToInput = 0
**************************

Printing Values for inputBeam
**************************
inputBeam.beamType = LG
inputBeam.power = 1
inputBeam.RF = 0
inputBeam.index1 = 0
```

```
inputBeam.index2 = 0
inputBeam.waistSize = 0.00061167843916231
inputBeam.waistPosition = 3.587581999999
inputBeam.matchToCavity = 1
inputBeam.fldAmp = (null)
inputBeam.fldPhase = (null)
**************************

***** Database definitions *****

++++++++++++++++++++
0> Database ID[ 1 ] : name { ".//sisDB_SCC.mcr$MAINDB" }
 * "ArmPower" = 0
 * "BS.WedgeAngle" = 0.000698132
 * "BS.Wfft" = 0
 * "BS.aperture" = 0.366
 * "BS.baffleX" = 2.1
 * "BS.baffleXOffset" = 0
 * "BS.baffleY" = 2.6
 * "BS.baffleYOffset" = 0
 * "BS.centerOffset" = 1000
 * "BS.flat" = 0
 * "BS.incident.phi" = 0
 * "BS.incident.theta" = 0
 * "BS.mech.tX" = 0
 * "BS.mech.tY" = 0
 * "BS.mech.tZ" = 0
 * "BS.mech.x" = 0
 * "BS.mech.y" = 0
 * "BS.mech.z" = 0
 * "BS.opt.ROC" = 1e+20
 * "BS.opt.refrIndex" = 1.44963
 * "BS.oscillation.phase" = 0
 * "BS.thickness" = 0.06
 * "BSShift" = 0
 * "BSWedge" = 0.04
 * "BS_T" = 0
 * "BX" = 2.1
 * "BY" = 2.6
 * "CCav.BS.WedgeAngle" = 0.000698132
 * "CCav.BS.prismInScale" = 1.00086
 * "CCav.BS.prismOffScale" = 0.999139
 * "CCav.BS2ITM" = 4.8096
 * "CCav.FPcav._cavLeng" = 4.6009e-07
 * "CCav.FPcav._diffLoss" = 0.865955
 * "CCav.FPcav._power" = 4168.36
 * "CCav.FPcav._totalLoss" = 3.81091
 * "CCav.MMT2_MMT3" = 15.4607
 * "CCav.MMT3_BS" = 19.368
 * "CCav.NfftProp23" = 2048
 * "CCav.RM_MMT2" = 15.726
 * "CCav.RM_beamSize" = 0
 * "CCav._cavLeng" = 9.083e-08
 * "CCav._diffLoss" = 2064.4
 * "CCav._power" = 14.8686
 * "CCav._totalLoss" = 2090.49
 * "CCav.config" = 2
 * "CCav.delRM" = 7.688e-08
 * "CCav.matchToInput" = 0
 * "CP_THICKNESS" = 0.13
 * "Config" = 2
 * "D_BS_CP" = 4.8046
 * "D_BS_PCP" = 4.8497
 * "D_BS_SCP" = 4.8046
 * "D_CP_ITM" = 0.005
 * "D_MMT2_MMT3" = 15.4607
 * "D_MMT3_BS" = 19.368
 * "D_PR2_PR3" = 16.1558
 * "D_PR3_BS" = 19.5384
 * "D_PRM_PR2" = 16.6037
 * "D_RM_MMT2" = 15.726
 * "D_SR2_SR3" = 15.4607
 * "D_SR3_BS" = 19.368
 * "D_SRM_SR2" = 15.726
 * "Design" = 0
 * "ETM.Wfft" = 0
 * "ETM.aperture" = 0.336
 * "ETM.incident.phi" = 0
 * "ETM.incident.theta" = 0
 * "ETM.mech.tX" = 0
```

```
* "ETM.mech.tY" = 0
* "ETM.mech.tZ" = 0
* "ETM.mech.x" = 0
* "ETM.mech.y" = 0
* "ETM.mech.z" = 0
* "ETM.opt.ROC" = 2245
* "ETM.opt.refrIndex" = 1.44963
* "ETM.oscillation.phase" = 0
* "ETM.thickness" = 0.2
* "FPcav.L0" = 3994.5
* "FPcav.Ncutoff1" = 0
* "FPcav.Ncutoff2" = 0
* "FPcav.Nfft" = 512
* "FPcav.adaptiveGrid" = 0
* "FPcav.delL" = 0
* "FPcav.lockFP" = 1
* "FPcav.matchToInput" = 3
* "HV_all_thermal" = 3
* "HV_lens_thermal" = 2
* "INVROC_HOT_WAT" = 625000
* "ITM.Wfft" = 0
* "ITM.aperture" = 0.336
* "ITM.incident.phi" = 0
* "ITM.incident.theta" = 0
* "ITM.mech.tX" = 0
* "ITM.mech.tY" = 0
* "ITM.mech.tZ" = 0
* "ITM.mech.x" = 0
* "ITM.mech.y" = 0
* "ITM.mech.z" = 0
* "ITM.opt.ROC" = 1934
* "ITM.opt.refrIndex" = 1.44963
* "ITM.oscillation.phase" = 3.14159
* "ITM.thickness" = 0.33
* "MASS_THICKNESS" = 0.2
* "MMT2.Wfft" = 0.08
* "MMT2.aperture" = 0.15
* "MMT2.incident.phi" = 0
* "MMT2.incident.theta" = 0.0151844
* "MMT2.mech.tX" = 0
* "MMT2.mech.tY" = 0
* "MMT2.mech.tZ" = 0
* "MMT2.mech.x" = 0
* "MMT2.mech.y" = 0
* "MMT2.mech.z" = 0
* "MMT2.opt.ROC" = -6.427
* "MMT2.opt.refrIndex" = 1.44963
* "MMT2.oscillation.phase" = 0
* "MMT2.thickness" = 0.075
* "MMT3.Wfft" = 0
* "MMT3.aperture" = 0.261
* "MMT3.incident.phi" = 0
* "MMT3.incident.theta" = 0.0137008
* "MMT3.mech.tX" = 0
* "MMT3.mech.tY" = 0
* "MMT3.mech.tZ" = 0
* "MMT3.mech.x" = 0
* "MMT3.mech.y" = 0
* "MMT3.mech.z" = 0
* "MMT3.opt.ROC" = 36
* "MMT3.opt.refrIndex" = 1.44963
* "MMT3.oscillation.phase" = 0
* "MMT3.thickness" = 0.1014
* "MirrorScale" = 1
* "Nfft" = 1024
* "PRM_X_Conf" = 0
* "PRM_Y_Conf" = 1
* "PcoatRate" = 5e-07
* "Pin" = 0
* "PowerArmGain" = 283.725
* "PowerInSubstrateUnit" = 0.00189133
* "PowerRecyclingGain" = 94.5664
* "PowerTotalGain" = 26830.8
* "PsubsRate" = 2e-05
* "RM.Wfft" = 0.025
* "RM.aperture" = 0.15
* "RM.incident.phi" = 0
* "RM.incident.theta" = 0
* "RM.mech.tX" = 0
* "RM.mech.tY" = 0
```

```
 * "RM.mech.tZ" = 0
 * "RM.mech.x" = 0
 * "RM.mech.y" = 0
 * "RM.mech.z" = 0
 * "RM.opt.ROC" = -5.6938
 * "RM.opt.refrIndex" = 1.44963
 * "RM.oscillation.phase" = 0
 * "RM.thickness" = 0.075
 * "ROC_ETM" = 2245
 * "ROC_ITM" = 1934
 * "ROC_MMT2" = -6.427
 * "ROC_MMT3" = 36
 * "ROC_PR2" = -4.555
 * "ROC_PR3" = 36
 * "ROC_PRM" = -10.997
 * "ROC_RM" = -5.6938
 * "ROC_SR2" = -6.427
 * "ROC_SR3" = 36
 * "ROC_SRM" = -5.6938
 * "RecPower" = 0
 * "SISAppFrame_TAGVALUE" = 1.25866e+09
 * "SISFatalErrorCount" = 0
 * "SRM_X_Conf" = 2
 * "SRM_Y_Conf" = 3
 * "T_ETM" = 5e-06
 * "T_ITM" = 0.014
 * "T_PRM" = 0.03
 * "T_RM" = 0.2
 * "T_SRM" = 0.2
 * "TestMass_Aperture" = 0.34
 * "Wfft" = 0.7
 * "adjust23" = 1
 * "armLeng" = 3994.5
 * "bevel" = 0.002
 * "dBX" = 0
 * "dBY" = 0
 * "del23perW" = -0.0010274
 * "extraLoss" = 7e-05
 * "folded_IFO" = 10
 * "iLoop" = 0
 * "iLoopEnd" = 0
 * "inMode" = 1
 * "initDelRM" = 7.688e-08
 * "inputBeam.RF" = 0
 * "inputBeam.index1" = 0
 * "inputBeam.index2" = 0
 * "inputBeam.matchToCavity" = 1
 * "inputBeam.power" = 1
 * "inputBeam.waistPosition" = 3.58758
 * "inputBeam.waistSize" = 0.000611678
 * "no_thermal" = 0
 * "poormans_thermal" = 1
 * "saveTemp" = 0
 * "straightDesign" = 1
 * "straight_IFO" = 0
 * "thermalModel" = 0
 * "theta2" = 0.87
 * "theta2_PRC" = 0.79
 * "theta2_SRC" = 0.87
 * "theta3" = 0.785
 * "theta3_PRC" = 0.615
 * "theta3_SRC" = 0.785
 * "w0In_0" = 0.000603
 * "w0In_1" = 1.3e-08
 * "wOnETM" = 0.0619634
 * "wOnITM" = 0.0529939
 * "zIn_0" = 3.3467
 * "zIn_1" = -3.35e-05

++++++++++++++++++++
1>     Database ID[ 2 ] : name { "$defaultDataBase" }
     * "AVOGADRO_NUMBER" = 6.02214e+23  " avogadro number "
     * "BOLTZMANN_CONST" = 1.38066e-23  [ m^2 s^-2 kg K^-1 ]  " Boltzman constant "
     * "FALSE" = 0
     * "GRAVITATIONAL_CONST" = 6.6726e-11  [ m^3 s-2 kg-1 ]  " gravitational constatnt "
     * "GRAV_ACCEL" = 9.80665  [ m s^-2 ]  " standard grav. accel. at sea level "
     * "LIGHT_SPEED" = 2.99792e+08  [ m s^-1 ]  " speed of light "
     * "NDYAG_WAVELENGTH" = 1.064e-06  [ m s^- ]  " NDYAG wavelength "
     * "PI" = 3.14159
     * "PLANCK_CONSTANT" = 6.62608e-34  [ m^2 s^-1 kg ]  " Planck constant "
```

```
* "SIO2_REFRACTIVE_INDEX" = 1.44963  " Nominal SiO2 refractive index "
* "TRUE" = 1
* "false" = 0
* "true" = 1
```

10 Objects

10.1 primitive (070228)

10.1.1 mirror2x2

aperture = 0.34

thickness = 0.2

mech.[x, y, z] = [0,0,0]

mech.[tX, tY, tZ] = [0, 0, 0]

opt.[T, R] = [0.005, 0.995]

opt.ROC = 2076

opt.reflIndex = 1.45

opt.HR_phase = (null)

opt.trans_phase = (null)

opt.AR_trans = (null)

10.1.2 cavity

cavity.L0 = 4000

cavity.delL = 0

cavity.matchToInput = 1

10.1.3 inputBeam

inputBeam.[beamType, inex1, index] = [LG, 0, 0]

inputBeam.[waistSize, waistPosition] = [0.01, -2000]

inputBeam.matchToCavity = 1

10.2 Properties

10.2.1 Thermal deformation (070429)

THERMOELASTIC(Psubs, Pcoat, waist [, T0])

THERMALPHASE(Ps,ubs Pcoat, waist [, T0])

- Psubs : Power absorbed by the substrate. If the value is negative, it is interpreted as an absorption rate. The power on the HR side multiplied by the power transmissivity and the power on AR side are added and this sum is multiplied by this absorption rate to calculate the power absorbed by the substrate.

- Pcoat : Power absorbed by the coating. If the value is negative, it is interpreted as an absorption rate, and the power on the HR surface is multiplied by this value to calculate the power absorbed by the coating.
- waist : beam waist size. If the value is genative, the actual beam profile is used to calculate coefficients of Deni series expansion of the absorbed power.
- T0 [300] : temperature in Kelvin

10.2.2 to be described

```
% ITM.opt.HR_phase = d0Thr * exp( -w0Thr * rr ) * ( 1 + b2Thr * rr + b4Thr * rr * rr )
% ITM.opt.HR_phase = DATAFILE( “thermalDeform_N256_W700.txt” )
% ITM.opt.HR_phase = "if ( r < (aperture/2-delR), 0, -ampRoll*pow( (r-aperture/2+delR)*delR0/delR, 4 ) )"
% ITM.opt.HR_phase = "0 * NOISESPEC( 123, 0.5e-9, 1, -3 )"
```

II. Physics in SIS

11 General

11.1 general

All quantities shown as f(x,y) can be specified by formulas which contains

(1) generic expressions using C syntax, like “thickness * absorption”

(2) data stored in files, as DATAFILE(“filename”, lowPassOption, lowPassParam)

(3) random 2d distribution, as NOISESPEC(seed, rms, power, WykoIndex)

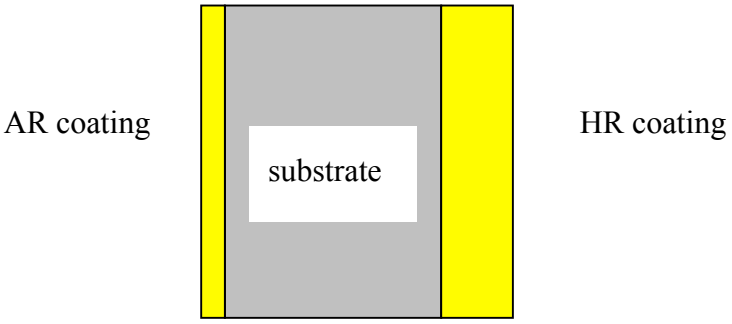
(4) Thermal deformations, as THERMOELASTIC(Psubs, Pcoat, Wbeam, T0) and THERMALPHASE(Psubs, Pcoat, Wbeam, T0)

These data can be combined, like 2*DATAFILE(“a”)*exp(-2 * (x*x+y*y) / (W*W)). This expression is evaluated at each pixel point.

12 Optics

12.1 mirror

12.1.1 mirror basic for 2x2 and 4x4 mirrors



(1) AR coating :

a. d(x,y) : surface hight

b. rf(x,y), rb(x,y), t(x,y) : reflectance and tramsmissivity

(2) HR coating:

same as AR coating

(3) substrate: $E_{out} = E_{in} \times \exp(i \, \text{phase}(x,y)) \times \sqrt{1\text{-Loss}}$

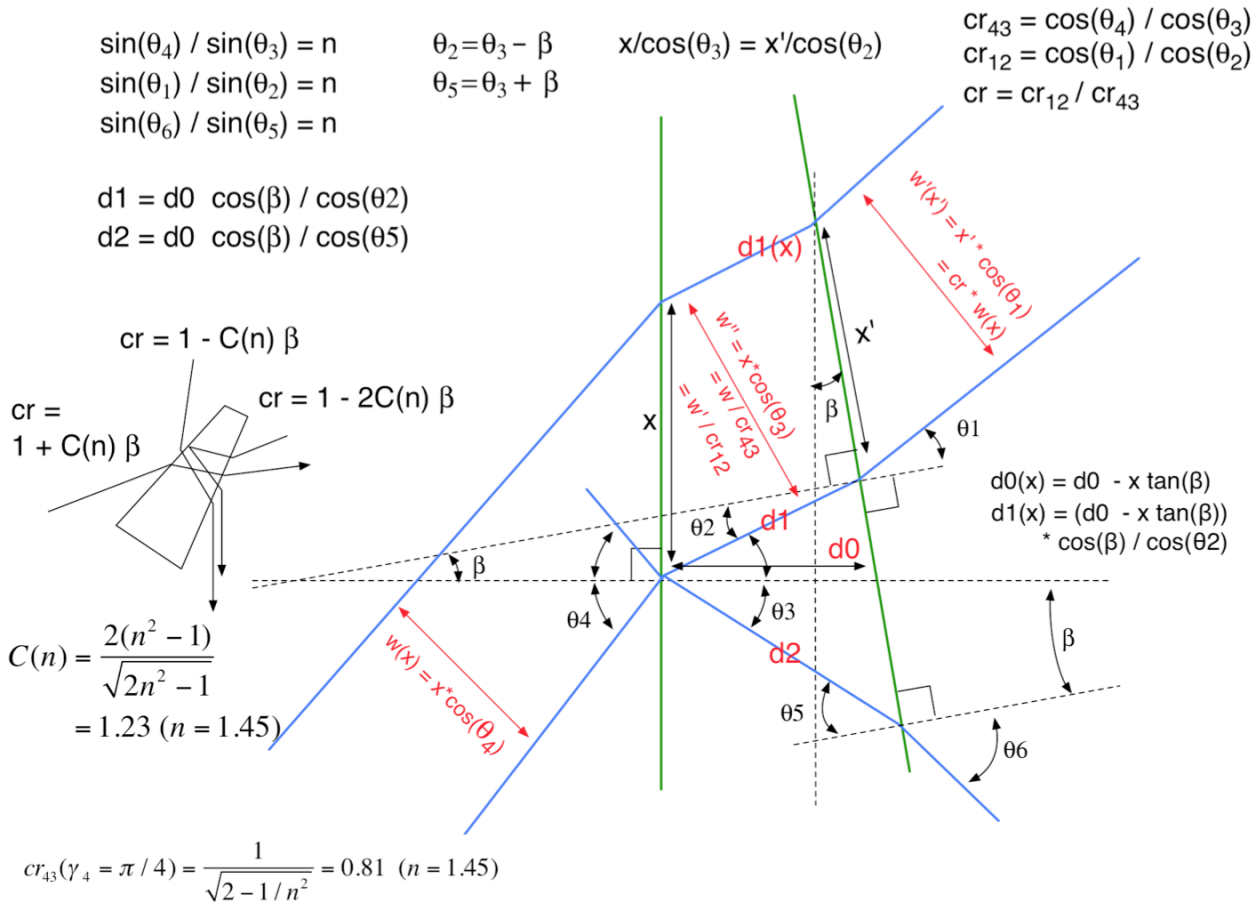
c. Loss(x,y)

d. phase(x,y)

e. Nominal optical thickness is set to be an integer multiple of wave length.

12.1.2 Wedge angle

The wedge angle changes the direction of the beam. This is effectively included in the simulation when a propagator is attached to the surface. The effect beyond this direction change is related to the change of the beam size in the optic. ~~For now, the thickness of the optic is assumed to be smaller than the Rayleigh length and the beam size change will be O(thickness / rayleigh)^2, and so no explicit wedge effect is not implemented in the simulation. When the simulation of beams induced by the wedge angle is needed, they can be added.~~



$$\frac{\sin(\theta_2)}{\sin(\theta_1)} = \frac{\sin(\theta_3)}{\sin(\theta_4)} = \frac{1}{n}$$

$$L = \frac{d_2}{\cos(\theta_2)}$$

$$r = d_2 \cdot \tan(\theta_2)$$

$$d_1 = d_2 + r \cdot \tan(\phi)$$

12.1.3 ABCD matrix of prims

In this subsection, the implication of the ABCD matrix is discussed. *In the SIS simulation, this formulation is not used, and the recipe in SIS is explained in the following sections.* These two approaches are compared where appropriate to understand the physics effect.

ABCD matrix of a prism is given as

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix} = \begin{pmatrix} \frac{\cos(\theta_2)}{\cos(\theta_1)}\frac{\cos(\theta_4)}{\cos(\theta_3)} & \frac{d}{n}\frac{\cos(\theta_1)}{\cos(\theta_2)}\frac{\cos(\theta_4)}{\cos(\theta_3)} \\ 0 & 1/A \end{pmatrix}$$

where θ_I's are the incident and internal angles from the normal in the order that the field passes through the prism of refractive index n and the physical path length through the prism is d.

The change of q ($\equiv z + i\, z_0$) through a prism is calculated as follows:

$$z' + i \cdot z_0' = (r_{21} \cdot r_{43})^2 \left(z + \frac{L}{n \cdot r_{21}^2} + i \cdot z_0\right)$$

$$r_{ij} = \frac{\cos(\theta_i)}{\cos(\theta_j)}$$

The effective distance of propagation for transverse profile is scaled by $1/(n\, r_{21}^2)$, and the overall transverse spread is scaled by the square of the ratio of the beam size which is determined by the change of the propagation direction.

When the incident angle is close to $\pi/4$ and when the wedge angle is small, the matrix element A can be approximated as follows taking the lowest order of these angles.

$$A = 1 + \varepsilon_w, \varepsilon_w = \frac{2(n^2 - 1)}{\sqrt{2n^2 - 1}} \times wedge = 1.231 \times wedge$$

The wedge angle is positive when $\theta_1 < \theta_4$. For the reflection by the HR surface, the magnitude is doubled. The dependence on the change of the incident angle around $\pi/4$ is assumed to be small, and it is neglected in the calculation.

For 0.8 degree wedge angle, this correction is 0.0172.

The contribution of the B term is O(thickness / ROC), and it can be neglected for BS with d = O(10cm). With this approximation, the prism effect can be summarized as follows.

$$ROC' = A / D \times ROC = (1 + 2\varepsilon_w) \times ROC$$

$$w' = \sqrt{A / D} \times w = (1 + \varepsilon_w) \times w$$

When the field goes through the BS with a wedge angle of 0.8 degree, the ROC changes by 3.4%, and the field goes from ITMx to SRM, ROC changes by 6.9%.

The latest design as of August 4th, the wedge angle of BS is 0.04, and all corrections quoted will be down by 20.

12.2 Field

12.2.1 Field propagation in space and in frequency domain

The Fresnel approximation of Huygen's Integral is given by the following equation, product of two terms, the primary propagation factor, exp(-ikΔz) and the remaining term E_t, which is mainly related to the transverse profile:

$$E(x,y,z) = \exp(-ik\Delta z)\cdot E_t(x,y,z)$$

$$E_t(x,y,z) \equiv \frac{i}{\Delta z\cdot\lambda}\iint dx_0\, dy_0 E_0(x_0,y_0,z_0)\exp(-ik\frac{\Delta x^2+\Delta y^2}{2\Delta z})$$

$$\Delta x = x - x_0, \Delta y = y - y_0, \Delta z = z - z_0, k = 2\pi/\lambda$$

In the limit of $\lambda\,\Delta z / w^2 \to 0$ (w being the size of the field), this expression becomes the following simple form

$$E(x,y,z) = \exp(-ik\Delta z)\cdot E_0(x,y,z)$$

by using this limiting identity:

$$\frac{i}{\varepsilon}\exp(-i\frac{\pi}{\varepsilon}(x^2+y^2)) \underset{\varepsilon\to 0}{\to} \delta(x)\delta(y)$$

E_t is a convolution of the source distribution E_0 and the paraxial diffraction kernel,

$$K(x,y,\Delta z) = \frac{i}{\Delta z\cdot\lambda}\exp(-ik\frac{x^2+y^2}{2\Delta z}) = \iint dp\, dq \exp(-i\cdot 2\pi\cdot(p\cdot x + q\cdot y))\cdot\tilde{K}(p,q,\Delta z)$$

$$\tilde{K}(p,q,\Delta z) = \exp(i\frac{\Delta z(p^2+q^2)}{2k})$$

So the frequency space formula of the Fresnel equation becomes

$$\tilde{E}_t(p,q,z) = \tilde{K}(p,q,\Delta z)\cdot\tilde{E}_t(p,q,z_0)$$

$$E_t(x,y,z) = \iint dp\, dq \exp(-i\cdot 2\pi\cdot(p\cdot x + q\cdot y))\cdot\tilde{E}_t(p,q,z)$$

The propagation of a field in a free space is done in the follwoing steps.

1) apply FFT to convert field at (x_0, y_0, z_0) to distribution in frequency domain, $E_t(p,q,z_0)$
2) multiply frequency domain paraxial diffraction kernel, $\tilde{K}(p,q,\Delta z)$
3) apply inverse FFT to convert $\tilde{E}_t(p,q,z)$ back to spatial distribution
4) multiply the longitudinal propapation phase, exp(-ikΔz)

When calculating the field propagation for a short distance, the following approximation can be used.

$$
\begin{aligned}
E_t(x,y,z,\Delta z) &= \frac{1}{4\pi^2}\iint dp\,dq \exp(ipx+iqy)\overline{E_t}(p,q,z,\Delta z) \\
&= \frac{1}{4\pi^2}\iint dp\,dq \exp(ipx+iqy)\overline{E_t}(p,q,z)\cdot\exp(i\frac{\Delta z}{2k}(p^2+q^2)) \\
&\approx \frac{1}{4\pi^2}\iint dp\,dq \exp(ipx+iqy)\overline{E_t}(p,q,z)(1+i\frac{\Delta z}{2k}(p^2+q^2)) \\
&= E_t(x,y,z) - i\frac{\Delta z}{2k}(\frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2})E_t(x,y,z)
\end{aligned}
$$

Because the transverse component satisfied the following equation,

$$
(\frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2})E_t(x,y,z) = 2ik\frac{\partial}{\partial z}E_t(x,y,z)
$$

the above approximation is the same as

$$
E_t(x,y,z,\Delta z) \approx E_t(x,y,z) + \Delta z\frac{\partial}{\partial z}E_t(x,y,z)
$$

This simpler form is the naive extrapolation formula.

12.2.2 Scattering by a small anomaly

When there is an anomaly of the surface shape or loss in a small area on a mirror surface, the effect can be expressed as follows.

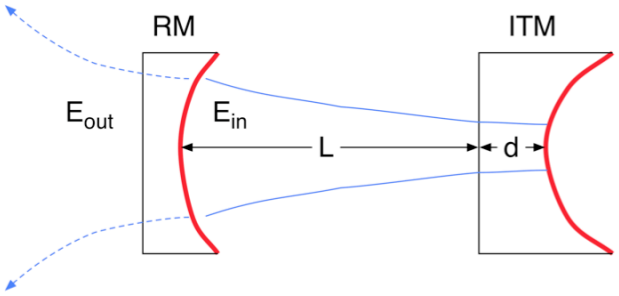
12.2.3 Field propagation in substrate

The effect of a substrate for the field propagation is derived starting from the Huygen's integral. In a substrate with refractive index n, the wavelength λ is replaced by λ / n, and the above Fresnel equation becomes as follows:

$$
E(x,y,z) = \exp(-ikn\Delta z)\cdot E_t(x,y,z,\Delta z/n)
$$

The propagation phase changes from exp(-ikΔz) to exp(-ik nΔz), i.e., the spatial distance effectively becomes n times longer. On the other hand, the dependence on Δz in the transverse profile, E_t, becomes Δz/n.

From this result, one can calculate a field in a cavity containing substrate in it, e.g., a cavity formed by a recycling HR side and ITM HR side, in the following way.

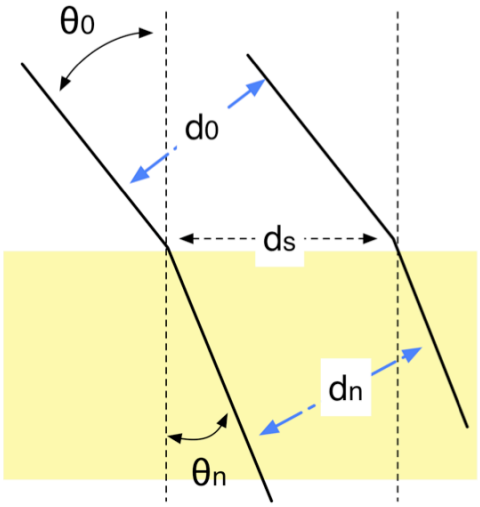


The propagation phase is calculated using L+d*n and the rest of the field propagation is calculated using L+d/n.

12.2.4 Field propagation in substrate with non-perpendicular incidence

When a field goes into a substrate with refractive index n with an angle θ_0, the field propagation direction changes to θ_n, which is related to the incident angle as follows:

$$\sin(\theta_0)/\sin(\theta_n) = n$$



Due to this bent, the beam size changes by the ratio of r_{n0} defined as:

$$r_{n0} = d_n / d_0 = \cos(\theta_n)/\cos(\theta_0)$$

The transverse component, E_t, becomes as follows:

$$E_t(x,y,z,\Delta z) = \frac{i}{\Delta z/n\cdot\lambda}\iint dx_0\, dy_0 E_0(x_0/r_{n0}, y_0, z_0)\exp(-ik\frac{(x-x_0)^2+(y-y_0)^2}{2\Delta z/n})$$

I.e., the beam size is scaled by r_{n0}. When $E_0(x)\sim\exp(-x^2/w^2)$, $E_0(x/r)\sim\exp(-x^2/(r\,w)^2)$.

$$
\begin{aligned}
E_t(r_{n0}x,y,z,\Delta z) &= \frac{i}{\Delta z/n\cdot\lambda}\iint dx_0\,dy_0 E_0(x_0/r_{n0},y_0,z_0)\exp(-ik\frac{(r_{n0}x-x_0)^2+(y-y_0)^2}{2\Delta z/n}) \\
&= \frac{i\cdot r_{n0}}{\Delta z/n\cdot\lambda}\iint dx_0'\,dy_0 E_0(x_0',y_0,z_0)\exp(-ik\frac{(r_{n0}x-r_{n0}x_0')^2+(y-y_0)^2}{2\Delta z/n}) \\
&= \sqrt{\frac{i}{\Delta z/n\cdot\lambda}}\int dy_0\exp(-ik\frac{(y-y_0)^2}{2\Delta z/n})\sqrt{\frac{i}{\Delta z/(r_{n0}^2 n)\cdot\lambda}}\int dx_0'\exp(-ik\frac{(x-x_0')^2}{2\Delta z/(r_{n0}^2 n)})E_0(x_0',y_0,z_0) \\
&\approx E_0(x,y,z_0)
\end{aligned}
$$

$\exp(-(r\,x)^2/w_n^2) = \exp(-x^2/(w_n/r)^2) = \exp(-x^2/w_0^2)$, i.e., $w_n = r\,w_0$.

When compared with the ABCD formulation, the formulation using Fresnel approximation gives same result as the ABCD matrix.

When the field goes out to vacuum from the other side, the change of direction introduces another factor r_{n1},

$$r_{n1} = d_n/d_1 = \cos(\theta_n)/\cos(\theta_1)$$

But this time, the beam becomes narrower, instead of broadening at the entrance.

$$E_1(x,y,z,\Delta z) \approx E_0(x\cdot r_{n1}/r_{n0},y,z_0)\cdot\exp(-ikn\cdot\Delta z)$$

where $n\,\Delta z$ is the optical path length at each path.

12.2.5 FP cavity with substrate

The stationary field calculation in the cavity can be done only using the nominal cavity length between two HR surfaces.

When calculating the transmission from AR side of ITM to calculate the source field on the HR side or calculating the leak field on the AR side of ITM or ETM from the field on the AR side, small correction in the transverse direction needs to be calculated, in addition to the longitudinal phase change.

In order to calculate the extrapolation, the following formula is numerically calculated.

$$E_t(x,y,z,n\Delta z) - E_t(x,y,z) = -i\frac{\Delta z/n}{2k}(\frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2})E_t(x,y,z)$$

The two dimensional derivative was calculated numerically by minimizing Δ in the following equation.

$$
\begin{aligned}
E(x,y) &= a_{00} + a_{10}\delta\cdot x + a_{01}\delta\cdot y + a_{20}\delta^2\cdot x^2 + 2a_{11}\delta^2\cdot xy + a_{02}\delta^2\cdot y^2 \\
\Delta(x,y) &= \sum_{nx=-2}^{2}\sum_{ny=-2}^{2}\left|E(x+\delta\cdot nx,y+\delta\cdot ny) - E_0(x+\delta\cdot nx,y+\delta\cdot ny)\right|^2/(nx^2+ny^2+1)
\end{aligned}
$$

In this equation, E_0 is the field calculated on one side of the surface, δ is the grid spacing. Using this result, the two dimensional derivative is given by the following equation.

$$(\frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2})E = 2(a_{20}+a_{02})$$

12.3 Cavity

12.3.1 Focusing cavity simulation

Simulation of the focusing cavity in the stable recicling cavity is discussed in T0900640. This formulation accelerates the simulation speed by factor 10 by allowing to use smaller grid sizes.

12.4 Lock

12.4.1 FP

A FP system is locked using an error signal calculated by a promptly reflected CR field and the total reflected field. Here, the promptly reflected component plays the role of the SB field in the actual experiment. The error signal is defined to be proportional to the imaginary part of CR(prompt)* x CR(total reflection).

12.4.2 Coupled Cavity

FP arm composed of ITM and ETM and Michelson cavity composed of RM and FP composite optic are clocked individually until a stationary condition is found.

12.5 FP

12.5.1 dynamics

12.5.2 static

12.5.2.1 rough estimation of reflectance using static quantities

$$\frac{E_{ref}}{E_{in}} = (1-L_{subs})\frac{r_{ITM}-r_{ETM}(r_{ITM}^2+t_{ITM}^2)}{1-r_{ITM}r_{ETM}}$$
$$\approx -1+\frac{L_{ITM}+L_{ETM}+T_{ETM}}{T_{ITM}}$$

12.5.2.2 diffractive loss

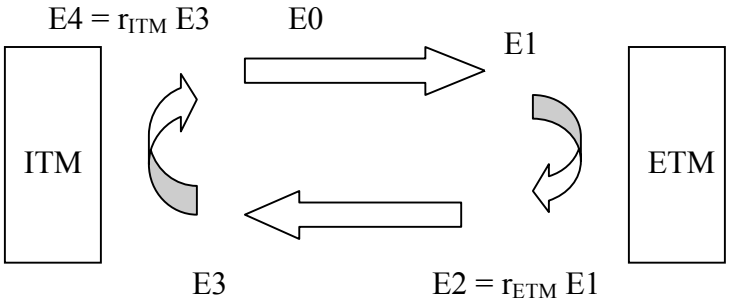
The diffractive loss of a FP cavity is calculated as follows.

(1) propagate a field from ITM to ETM (E0->E1),

(2) reflect the field by ETM (E1->E2),

(3) propagate back to ITM (E2->E3),

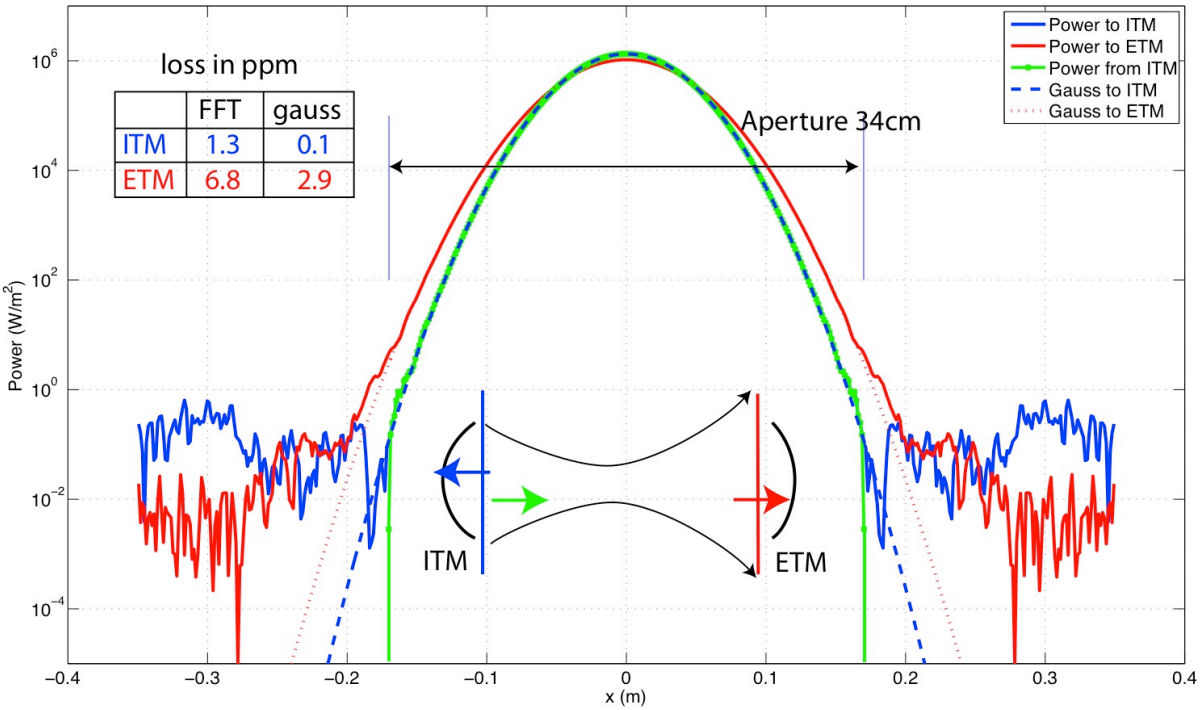
(4) reflect the field by ITM (E3->E4)



And the diffractive loss is defined to be

$$diffractive\ loss = (P0 - \frac{P4}{R_{ITM} R_{ETM}}) / P0$$

~~When we refer to the loss of an individual mirror (ITM, e.g.), the diffractive loss of the cavity is calculated and the loss of the other mirror (ETM) is subtracted from this total loss. Practically, this is done by setting the aperture of the other mirror (ETM) large so that the diffractive loss of the other mirror (ETM) is negligible.~~ THIS IS WRONG!!!



Power distribution of fields in a FP cavity

ROC(ITM) = 1971, ROC(ETM) = 2191, mirror aperture = 34cm, cavity length = 3995m

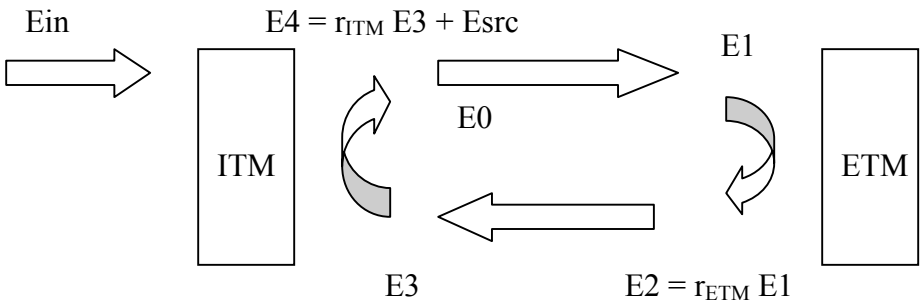
The solid blue line is the power distribution of the field going to ITM in a FP with finite aperture (34cm). The dashed blue line is the power distribution of the same field in a FP cavity with infinite aperture size, and its distribution is the Gaussian shape. The diffractive loss is the power of these fields outside of the aperture. As can be seen from the comparison of these two blue lines, the diffractive loss estimated using the Gaussian shape underestimates the loss, because the actual distribution has larger diffractive tail and this tail is outside of the mirror aperture.

The green line is the power distribution of the field going from ITM toward ETM. Because of the finite mirror size, the green line shows sharp cut offs at the mirror radius. In a FP cavity with infinite mirror aperture, the same field has the Gaussian shape, same as the glue dashed line. When a Gaussian beam propagates, the Gaussian shape is preserved and the power distribution of the field on ETM is of the Gaussian shape as is shown in dotted red lines. But, in a FP cavity with finite aperture, the initial condition is not of Gaussian shape, i.e., the green line with cut off edges, and the field on ETM is not of Gaussian shape, but a Gaussian with long diffractive tail.

The diffractive loss estimated using the Gaussian shape tends to underestimate the actual loss, as can be seen by comparing blue solid line and blue gashed line or red solid line and red dotted line. Because the diffractive loss is initiated by the sharp cut off of the field at the edge of the mirror located on the other side, it is not appropriate to calculate the diffractive loss of a mirror with finite aperture by paring with a mirror with infinite aperture.

12.5.2.3 Round trip cavity loss, total loss and mode matching

The round trip loss is defined in the following way.



For a given source field, E_{src}, the stationary field, E_4, in a cavity is calculated numerically by iteration. This automatically includes the effect of the mode coupling of the source field and the cavity mode, diffractive loss and any other possible interactions and causes of degrading the power.

The round trip loss, L_{cav}, is defined as follows for a locked cavity, where the round trip phase is an integer multiple of 2 π.

$$E_4 = r_{ITM} r_{ETM} \sqrt{1 - L_{cav}} E_4 + E_{src}$$

$$E_4 = \frac{E_{src}}{1 - r_{ITM} r_{ETM} \sqrt{1 - L_{cav}}}$$
$$\approx \frac{E_{src}}{1 - r_{ITM} r_{ETM}} (1 - \frac{L_{cav}}{2(1 - r_{ITM} r_{ETM})})$$
$$\approx \frac{E_{src}}{1 - r_{ITM} r_{ETM}} (1 - \frac{L_{cav}}{T_{ITM}})$$

If the total loss, L_{total}, is defined by

$$P_{cav}(L_{cav}) = (1 - L_{total})\, P_{cav}(L_{cav}=0)$$

the total loss, L_{total}, and the round trip cavity loss, L_{cav}, have the following relation.

$$L_{total} = \frac{2}{T_{ITM}} L_{cav}$$

When all other losses are neglected, like mode mismatch, the source field and the input field are related as

$$E_{src} = t_{ITM}\, E_{in}$$

and the cavity power becomes

$$P_{cav} = \frac{4}{T_{ITM}} \cdot (1 - \frac{T_{ITM}}{2}) \cdot (1 - 2\frac{L_{cav} + T_{ETM}}{T_{ITM}}) \cdot P_{in}$$

where terms up to the first order of T_{ITM} and the first order of $(L_{cav} + T_{ETM})/ T_{ITM}$ are kept.

Within the same order, i.e., up to $O(T_{ITM})$ and $O((L_{cav} + T_{ETM})/ T_{ITM})$, the transmitted power, P_{trans}, the reflected power, P_{refl}, and other losses accounted for by L_{cav}, P_{loss}, are given by the following equations.

$$P_{trans} = P_{cav} \cdot T_{ETM} = 4\frac{T_{ETM}}{T_{ITM}} P_{in}$$
$$P_{loss} = P_{cav} \cdot L_{cav} = 4\frac{L_{cav}}{T_{ITM}} P_{in}$$
$$P_{refl} = (1 - 4\frac{L_{cav} + T_{ETM}}{T_{ITM}}) P_{in}$$

This shows that the energy conservation is explicitly manifested, $P_{in}=P_{trans}+P_{loss}+P_{refl}$.

The ratio of the two powers, the stationary cavity power, Pcav, and the source power, Psrc, are related by the following expression.

$$\sqrt{\frac{P_{src}}{P_{cav}}} = 1 - r_{ITM} r_{ETM} \sqrt{1 - L_{cav}}$$

$$L_{cav} = 1 - (\frac{1 - \sqrt{P_{src} / P_{cav}}}{r_{ITM} r_{ETM}})^2$$

When expressing Esrc as a sum of the resonating component, Eres, and the non resonating component, Enr, E4 can be shown in the following way:

$$E_4 = \frac{1}{1 - r_{ITM} r_{ETM}} E_{res} + \frac{1}{1 - r_{ITM} r_{ETM} \exp(i\varphi)} E_{nr}$$
$$\approx \frac{1}{1 - r_{ITM} r_{ETM}} E_{res}$$

or

$$P_{cav} = \frac{(1 - \varepsilon_{nr}) P_{src}}{(1 - r_{ITM} r_{ETM})^2}$$
$$\varepsilon_{nr} = P_{nr} / P_{src} = P_{nr} / (P_{nr} + P_{res})$$

where $1-\varepsilon_{nr}$ is the fraction of the source field energy resonating in the cavity. If ITM does not change the mode, this is the mode coupling of the input field and the cavity resonant mode. When this expression is put in the expression of L_{cav}, one gets the following relation between the mode matching and the round trip loss.

$$L_{cav} \approx \frac{(1 - r_{ITM} r_{ETM})}{r_{ITM} r_{ETM}} \varepsilon_{nr} \approx \frac{T_{ITM}}{2} \varepsilon_{nr}$$

In the limit of $\varepsilon_{nr} = 0$,

$$P_{cav} = (\frac{2}{T_{ITM}})^2 P_{src}$$

i.e., the source field adds up $2/T_{ITM}$ times to form the resonating field. So, when there is a non resonant loss ε_{nr}, the loss per round trip becomes the expression shown above.

An additional explanation will help to relate the normal mode matching concept and the loss calculation in the simulation. When a field goes into a FP cavity, all components, resonant and non resonant, goes through ITM. The resonant component adds up constructively in the cavity to become a major component in the cavity. Because the amplitude in the cavity is large, the leak from the ITM back to the incident direction is still noticeable even when a small transmittance, T_{ITM}, is multiplied. This way, the resonant component behaves like it is stored in the cavity. The non resonant components adds up destructively in the cavity and the resultant amplitude is small and is a minor, negligible component in the cavity. Because the amplitude is small, the leak from ITM is negligible, and it looks like the non resonant component is not in the cavity.

The simulation does not distinguish the resonant and non resonant component. The transmission treats both components equally. When the field builds up in the cavity, only the resonant component remains large and the non resonant component becomes negligible.

Mathematically, this calculation based on realistic process and another calculation using mode matching – just use the resonant component of the input field to calculate the cavity field – give the same answer.

12.5.2.4 incident and stationary fields

For a nominal advLIGO cavity, (ROC=2076m, cavity length = 4km), the diffractive losses for the first incident field and for the stationary field were calculated with distorted input beams. The input beam is a Hermite Gaussian field, which matches to the cavity eigen mode, but it goes through a BS mirror placed in front of the FP cavity, i.e., filtered by an oval mask.

BS aperture	first incident	stationary
400mm	1.6 ppm	0.25 ppm
350mm	15.3 ppm	0.44 ppm

This one shows that the diffractive loss depends how you calculate the loss. For the first incident field, the beam is distorted (the more for smaller BS) and the loss is large. The field in the cavity is filtered to become a nice Gaussian shape when the field becomes stationary. Because of this, the loss of the stationary field is small, and the dependence on the input beam shape is smaller than the difference seen for the first incident case.

When the ray tracing code is used to calculate the geometrical effect to calculate the diffractive loss, the loss calculated corresponds to the calculate for the first incident case. This shows a limitation of the ray tracing program, like ZEMAX, when there is a resonator in the optics setup being calculated.

13 Thermal distortion

Hello-Vinet analytic formula in the code for mirrors except for BS.

14 Hankel transformation

14.1 Basic expressions

14.1.1 basic math

$g(p) = \int dx \exp(i p x) f(x) / \sqrt{2}\pi$, $f(x) = \int dp \exp(-i p x) g(p) / \sqrt{2}\pi$

$\delta(x) = \int dx \exp(i p x) / 2\pi$

$J_0(x) = \int dt \exp(i x \cos(t)) / 2\pi$

14.1.2 Hankel math

ζk : zeros of $J1(z) \sim \pi(k + 0.25)$

$\zeta 0 = 0$, $\zeta 1 = 3.8317$, $\zeta 2 = 7.0156$, $\zeta 3 = 10.1734$, $\zeta 4 = 13.324$, $\zeta 5 = 16.471$

$r_k = a\, \zeta_k / \zeta_N$: spatial sampling

$\rho_k = \zeta_k / a$: spatial frequency sampling

14.1.3 Conversion between x and f

$$H_{\alpha\beta}^{(+)} = \frac{2a^2 J_0(\zeta_\alpha \zeta_\beta / \zeta_N)}{\zeta_N^2 \cdot J_0^2(\zeta_\beta)} : f \leftarrow x$$

$$H_{\alpha\beta}^{(-)} = \frac{2J_0(\zeta_\alpha \zeta_\beta / \zeta_N)}{a^2 \cdot J_0^2(\zeta_\beta)} \quad : x \leftarrow f$$

14.2 Physics element

Propagator

$$\tilde{G}_\alpha = \exp(i \frac{z}{2k} \rho_\alpha^2)$$

$$P_{\alpha\beta} = \sum_{\sigma=0}^{N} H_{\alpha\sigma}^{(-)} \tilde{G}_\sigma H_{\sigma\beta}^{(+)}$$

Mirror reflection by front size

$$M_\alpha = -r \cdot \exp(i2k \frac{r_\alpha^2}{2R} + 2ik\delta(r_\alpha))$$

Mirror reflection by back size

$$M_\alpha = r \cdot \exp(-i2k \frac{n \cdot r_\alpha^2}{2R} + 2ik\phi(r_\alpha)) = r \cdot \exp(-i2k(\frac{n}{2R} - \frac{1}{2f}) r_\alpha^2)$$

Transmission

$$T_\alpha = t(r_\alpha)\cdot\exp(i\phi(r_\alpha)) = t(r_\alpha)\cdot\exp(-ik(\frac{(n-1)}{2R}-\frac{1}{2f})r_\alpha^2)$$

14.3 Cavity

fA : reflection by he front side of mirror A

bA : reflection by the back side of mirror A

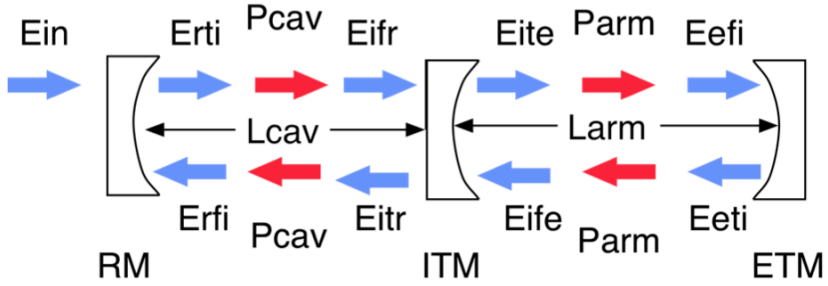
tA : transmission through mirror A

P : propagator

FP cavity

$$E_{cav} = \frac{1}{1-e^{i\phi}C}\cdot tITM\cdot E_{in}$$
$$C = fITM\cdot P\cdot fETM\cdot P$$

Coupled Cavity



$$Erti = (1-Ccav-fRM\cdot Pcav\cdot tITM\cdot Parm\cdot fETM\cdot Parm\frac{1}{1-Carm}tITM\cdot Pcav)^{-1}\cdot tRM\cdot Ein$$

$$Eite = (1-Carm)^{-1}\cdot tITM\cdot Pcav\cdot Erit$$

$$Ccav = fRM\cdot Pcav\cdot bITM\cdot Pcav$$

$$Carm = fITM\cdot Parm\cdot fETM\cdot Parm$$

15 FFT, PSD and ASD

$$H(f) = \int_{-\infty}^{+\infty} h(t)e^{i2\pi ft}\,dt$$

$$h(t) = \int_{-\infty}^{+\infty} H(f)e^{-i2\pi ft}\,df$$

$$\int_{-\infty}^{+\infty}\exp(i2\pi t\xi)d\xi = \delta(t)$$

$$Total\ \ power = \int_{-\infty}^{+\infty} |h(t)|^2 dt = \int_{-\infty}^{+\infty} |H(f)|^2 df$$

$$h_k \equiv h(t_k), \quad t_k \equiv k\Delta, \quad \Delta \equiv W/N, \quad k = 0,1,...,N-1$$

$$H_n \equiv \sum_{k=0}^{N-1} h_k e^{i2\pi k\frac{n}{N}}, \quad H(f_n) \approx \Delta H_n, \quad f_n = \frac{n}{N\Delta} = \frac{n}{W} \quad (n = -\frac{N}{2},...,\frac{N}{2}-1)$$

$$h_k = \frac{1}{N}\sum_{n=0}^{N-1} H_n e^{-i\ 2\pi\ kn/N}$$

$$\int |h(t)|^2\ dt \approx \frac{W}{N}\sum |h_k|^2 = \int |H(f)|^2\ df \approx \frac{1}{W}\sum |H(f_n)|^2 = \frac{\Delta^2}{W}\sum |H_n|^2 = \frac{W}{N^2}\sum |H_n|^2$$

$$\int \sum |h_k|^2 = \frac{1}{N}\sum |H_n|^2$$

$$\text{FFTW FFT} = \sum_{0}^{N-1} X_k e^{\pm i\ 2\pi\ kn/N}$$

$$\int_{-\infty}^{+\infty} |h(t)|^2 dt = W\sigma^2 = \int_{-\infty}^{+\infty} |H(f)|^2 df = W\int_{-\infty}^{+\infty} |ASD(f)|^2 d$$

$$ASD(f_n) = \frac{1}{\sqrt{W}} H(f_n) = \frac{\Delta}{\sqrt{W}} H(f_n) = \frac{\sqrt{W}}{N} H_n$$

$$h_k = \frac{1}{N}\sum H_n e^{-i\ 2\pi\ kn/N} = \frac{1}{N}\frac{N}{\sqrt{W}}\sum ASD(f_n)\, e^{-i\ 2\pi\ kn/N} = \frac{1}{\sqrt{W}}\sum ASD(f_n)\, e^{-i\ 2\pi\ kn/N}$$

III. Simulation coding details

16 Circular boundary calculation