SIS20 Primer

Hiro Yamamoto / LIGO

- Introduction
  - Real world vs simulation
- Simple system
  - FP
    - Field, power distribution, etc
  - FP with maps and point absorber
    - Round trip loss, modes, mirror surface, etc
- LLO
  - DRFPI with point absorber
  - Round trip loss, PRG, etc

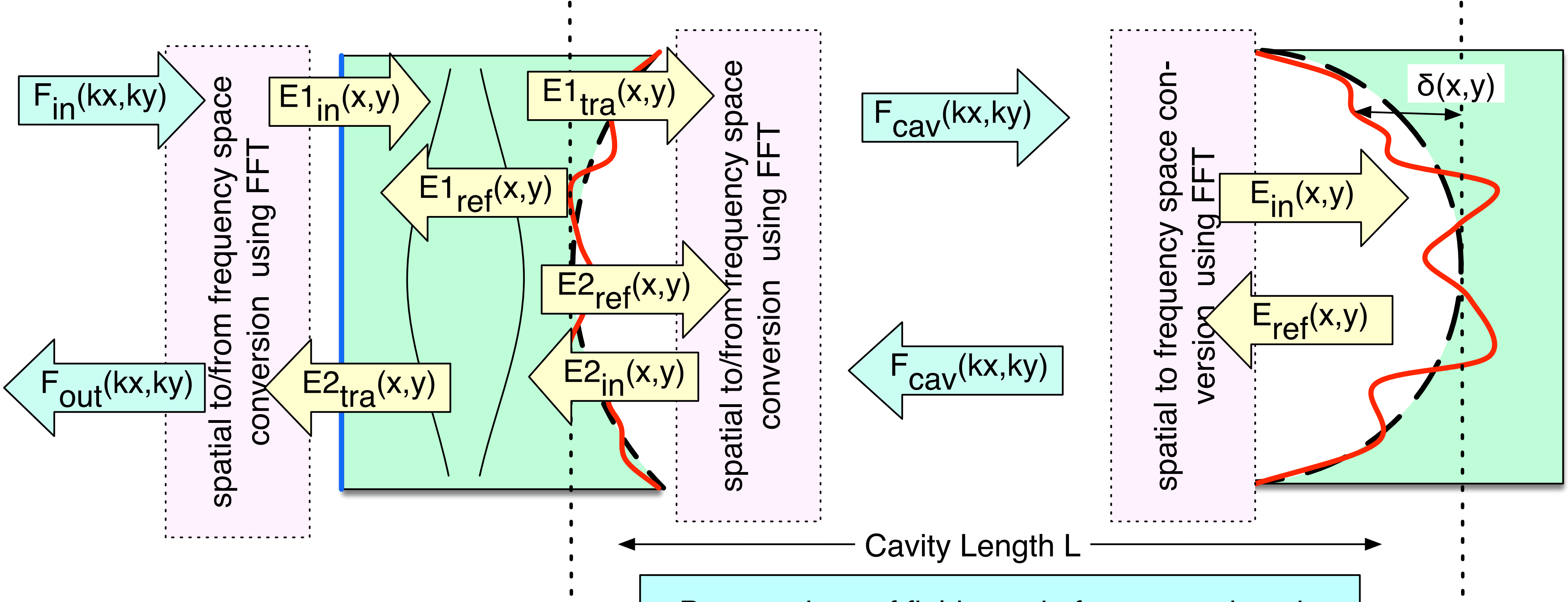
real hardware vs SIS

	Real HW	Simulation
Building block	Laser, Mirror, Cavity	IFOBuilder : tool package FFTIFO (base class), LaserObj, MirrorObj, PropObj
IFO	40m, iLIGO, aLIGO	DRFPM, FP, Mirror (derived class of FFTIFO using objects defined in IFOBuilder)
Operation	Laser on, locking	lock, calc (calculate fields with / without cavity locking)
Analysis	Measure error signal, modes in cavity, power on baffle	demod, power, gaussFit (FFT fields) HGPower, mainLGs (HG and LG mode analysis) PointScatter, getField (large angle scattering)

FFT Basic :
Field on optics + propagation

Interaction between fields and optics are in spatial domain

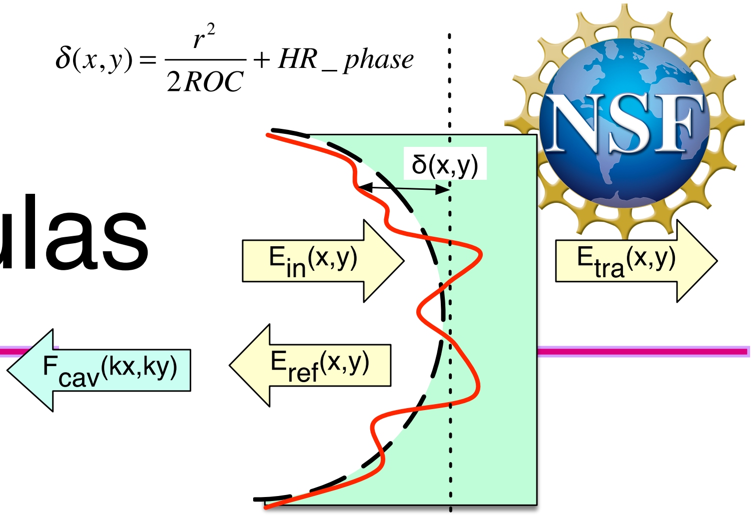
$$\delta(x,y) = \frac{r^2}{2ROC} + HR\_phase$$



Propagations of fields are in frequency domain

FFT is used for the conversions between spatial and frequency domain

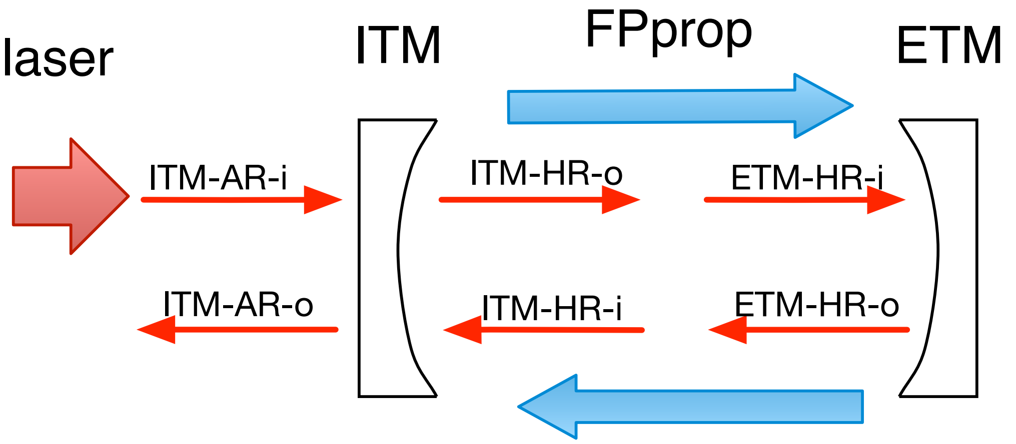
Three basic formulas



- Transmission
  - » $E_{tra}(x,y) = t_{opt} \cdot \exp(i\varphi_t) \cdot \exp(-\mathrm{ik}(\mathrm{n}-1)\delta) \cdot E_{in}(x,y)$
  - » t_{opt} =amplitude transmittance, φ_t =transmission phase, k=$2\pi/\lambda$, n = refractive index, $\delta = surface\ height$, RoC=radius of curvature
- Reflection
  - » $E_{ref}(x,y) = r_{opt} \cdot \exp(2\mathrm{ik}\delta) \cdot E_{in}(x,y)$
  - » r_{opt} =amplitude reflectance
- Propagation
  - » $E_{out}(x,y) = IFFT[\,Prop(fx,fy,L,n_0)\ FFT[E_{in}(x,y)]\,]$
  - » $FFT(IFFT)$ = (inverse) Fast Fourier Transform, L = propagation distance, n_0=refractive index of space

Simple FP simulation

`FPIFO0.m`



```
classdef FPIFO0 < FFTIFOAcc

    methods
        function obj = FPIFO0( varargin )
            obj@FFTIFOAcc( varargin{:} );
        end

        function defineIFO( obj, varargin )
            %% add optics
            % laser source
            obj.addLaser( 'laser' );
            % ITM
            obj.addMirror( 'ITM', 'invRoC', 1/1934, 'Aperture', 0.34, 'T', 0.014, 'Thick', 0.2 );
            % ETM
            obj.addMirror( 'ETM', 'invRoC', 1/2245, 'Aperture', 0.34, 'T', 5e-6 );

            %% define connections among optics
            obj.addProp( '', 'laser', 'ITM-AR' );
            obj.addProp( 'FPprop', 'ITM-HR', 'ETM-HR', 4000);
        end
    end
End
```

Examine FP cavity
same procedure for any

```
* Build an IFO
fp = FPbasic;                   % define FP

* Lock the cavity
fp.lock;                        % lock FP

* Study system
fp.power('ITM-HR-o');           % power of outgioing field from ITM HR
fp.roundtripLoss(‘ITM-HR-o’);   % round trip loss
fp.gaussFit('ITM-HR-o')         % gaussfit of a field
fp.HGCoef('ITM-HR-o',0:1,0:1)   % HG(0,0), (1,0), (0,1), (1,1) amplitude

fp.printAll                     % print all information about fp

% field of incoming field to ETM HR
% E(j,i) is the amplitude of the field at (x,y) = (x(i), x(j))
[E,x,y] = fp.getField('ETM-HR-i');
semilogy( x.^2, abs(E(end/2,:)).^2 )
```

Outputs

```
fp = FPbasic;     (the mode is defined using FP cavity RoCs and length)
Cavity mode defined using field 'ITM-HR-o' with w = 0.0534211, RoC = -1934

fp.lock;
  1 (FFT      8) : 'ITM-HR-o' Pwr= 2.832e+02, Err= 9.930e-01, …
  (change cavity length until converges)
  9 (FFT     72) : 'ITM-HR-o' Pwr= 2.835e+02, Err= 8.077e-07, … delL=-3.376e-16

fp.roundtripLoss(‘ITM-HR-o’)
  6.9348e-07

fp.gaussFit(‘ITM-HR-o’) : (gauss fit of the field)
  '(x,y) = (1.79476e-08, -1.80293e-08), w = 0.05342810192, R = -1934.086631

fp.HGCoef('ITM-HR-o',0:1,0:1)  % HG(0,0), (0,1); (1,0), (1,1) amplitude
   1.6837e+01 + 3.1260e-03i   2.7750e-10 - 1.9355e-10i
   6.3478e-11 + 1.8906e-10i   2.0643e-11 - 1.3251e-11i
```

fp.printAll outputs

```
=== Information about fields ===
3) name: ITM-HR-i   ====   Power = 283.481, E(analytic) = -16.8377, [status = 1024]
RoC = 1934, w = 0.0534211, z = 1837.22, z0 = 421.68, 1/phiCorr = 1844.22
q = 1837.22 + i 421.68, 1/q = 0.000517063 - i 0.000118677

=== Information about optics ===
2) name: 'ITM',  Wfft = 0.854768,  Nfft = 256
inPorts  = [ 3 4 ]  loss for each input port = [ 2.10749e-07 1.66283e-09 ]
outPorts = [ 5 6 ]

loss 3.91984e-05 in 'ITM-HR-i'+'ITM-AR-i'->'ITM-HR-i'+'ITM-AR-i'

R = 0.986, T = 0.014, L = 0
Roc(HR) = 1934, Aperture = 0.340000, Thickness = 0.200000, n = 1.449630, …

=== Information about propagators ===
3) name: FPprop   prop loss = 4.85674e-07
Prop from 'ITM-HR-o'[5] to 'ETM-HR-i'[7]
L0 = 4000, propL0 = 4000, delL = 4.61187e-07, lockPhase/pi = 0, gouy00 = 2.72342
nRef = 1, ratio = -1.16894, CLa = -0.00046387, (-k*L+(1+n+m)gouy00)/pi = 1.83876e-08
```

FP with surface aberration

FPwMaps.m

- Variables can be set at run time
  - fp = FPwMaps('ITMID', 4, 'ETMID', 10, 'xPoE', 0.02, 'pointPwrE', 0.03);
- Mirrors can be specified by ID
  - Data files should be in ./Data0/
- Simple ring heater
- Thermal distortions by coating absorption
- Point absorbers at multiple locations
- Spatial resolution can be specified if you need fine structure
- Poorman's WFS
- Support tools
  - Mode analysis
  - Inspection of surface map

Implementing surface aberration

```
% dat = calcThermal( elph, r, Psub, Pcoat, w, thickness, radius )
%    thermal deformation by absorption using Hello-Vinet formula
%
% del = pointAbs( x, y, x0, y0, Pabs, absRadius )
%    parametrization of surface aberration by point absorber
%    (x0, y0) : location of point absorber in m
%    Pabs : amount of absorption in W
%    multiple absorbers : x0 = [0.02, 0.03], y0 = [-0.01, 0.02],
%                         Pabs = [0.04, 0.02]

% ITM HR surface

obj.setHRfiles( 'ITM', ITMmap, ...
   'map + rsq*delRoCI/2 + calcThermal(0, r, 0, PcoatI, wI, 0.2, 0.175 )
        + pointAbs(x,y,xPoI, yPoI, pointPwrI)', ...
   0.16, ITMRoC, 0, ...
   'delRoCI', delRoCI, 'PcoatI', PcoatI, 'wI', 0.053,
   'xPoI', xPoI, 'yPoI', yPoI, 'pointPwrI', pointPwrI );
```

LLO simulation

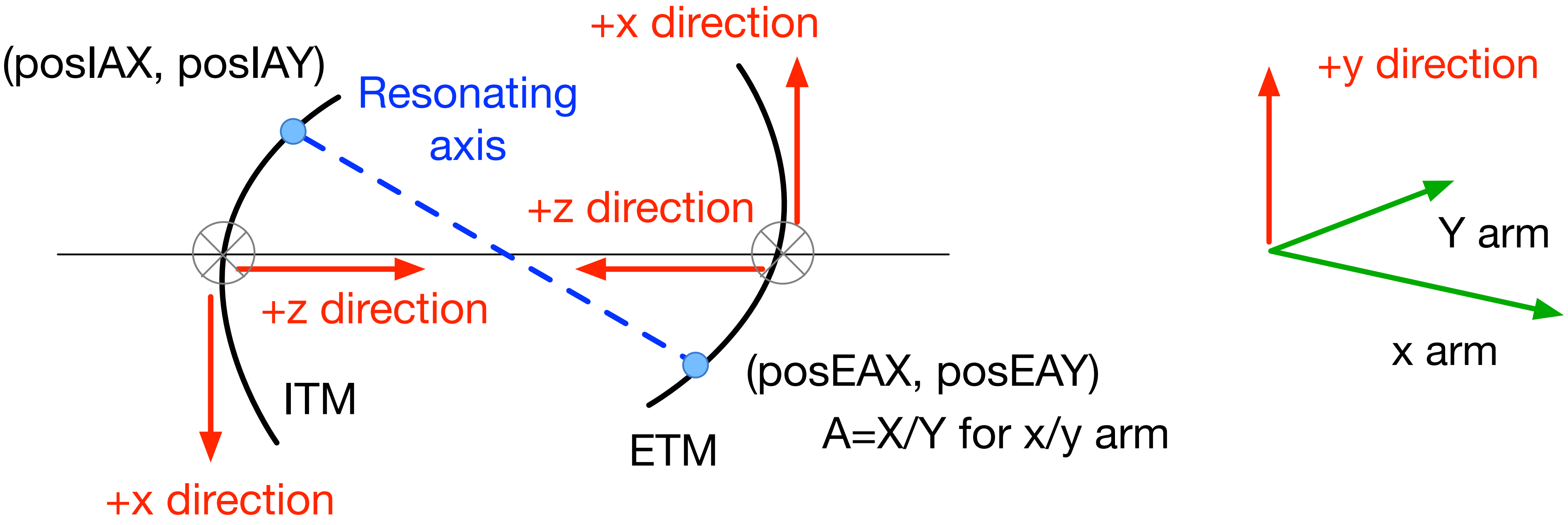
LLODRFPMAcc.m

- Dual recycled FP Michelson with LLO maps
  - Stable power recycling and signal recycling cavities
- All the map aberrations in FPwMaps are included for both arms
  - Effects of point absorbers
- Beam off-centering in the arm – tricky way
- Misc
  - Extra arm loss of 30ppm added by hand to make the PRG to be 50

LLO runs with point absorber

```
llo = LLODRFPMAcc;  llo.lock(‘errorLimit’,1e-3)

: 'PRM-HR-o' Pwr= 5.186e+01, Err= 2.278e-04, delL=-1.231e-12
: 'SRM-HR-o' Pwr= 5.986e-02, Err= 2.950e-04, delL=-1.203e-11
: 'ITMX-HR-o' Pwr= 6.948e+03, Err= 7.878e-04, delL= 1.626e-15
: 'ITMY-HR-o' Pwr= 6.785e+03, Err= 7.715e-04, delL= 1.577e-15

llo.roundtripLoss('ITMX-HR-o’) : 5.315445166031996e-05
llo.roundtripLoss('ITMY-HR-o’) : 5.931360270183816e-05

% with point absorber on ETMY
llo = LLODRFPMAcc('xPoY', 0.02, 'yPoY', 0, 'PpointEY', 0.05 );

: 'PRM-HR-o' Pwr= 3.013e+01, Err= 7.783e-05, delL= 2.503e-12
: 'SRM-HR-o' Pwr= 9.942e-02, Err= 3.720e-04, delL= 2.699e-11
: 'ITMX-HR-o' Pwr= 3.996e+03, Err= 8.830e-04, delL=-3.660e-17
: 'ITMY-HR-o' Pwr= 3.932e+03, Err= 9.296e-04, delL=-4.301e-15

llo.roundtripLoss('ITMX-HR-o’) : 5.317286612804839e-05
llo.roundtripLoss('ITMY-HR-o’) : 1.597188171539310e-04
```

LLO beam tilt



```
llo = LLODRFPMAcc('res', 2e-3, 'armPower', 0,
      'xPoY', 0.02, 'yPoY', 0, 'PpointEY', 0.05,
      'posEYX', -0.02);
```

No thermal by coating absorption

Point absorber on ETMY at (2cm,0) 50mW

Beam on ETMY at (-2cm,0)

PRG and arm loss

Point absorber on ETMY	Beam center on ETMY	PRG	X arm loss	Y arm loss
none	(0,0)	51.0	53 ppm	59 ppm
(2cm,0) 50mW	(0,0)	29.1	53 ppm	160 ppm
(2cm,0) 50mW	(-2cm,0)	32.8	53 ppm	108 ppm
(2cm,0) 50mW	(2cm,0)	33.5	53 ppm	141 ppm
(2cm,-1cm) 30mW (0,-1cm) 50mW	(2cm,0)	28.2	53 ppm	155 ppm