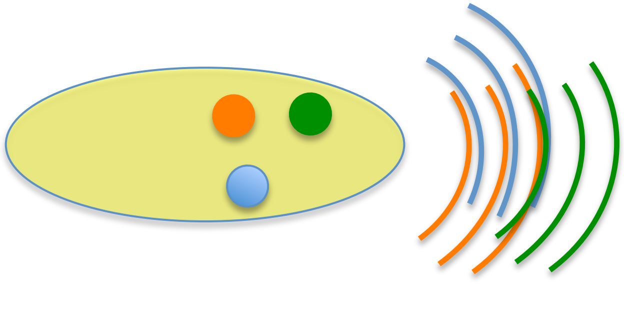
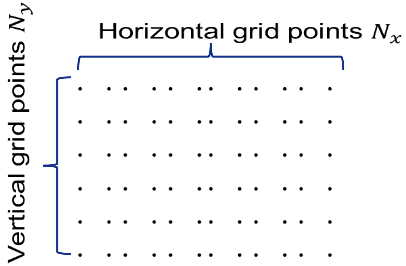
COMSYL

Coherent Modes of Synchrotron Light

(calculates the coherent modes for undulators in storage rings)

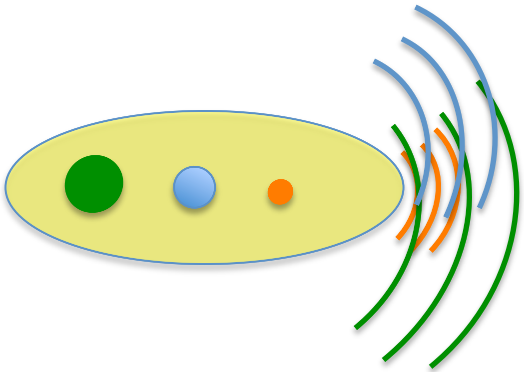
$$W(x_1, y_1, x_2, y_2, z, \omega) = \langle E^*(x_1, y_1, z, \omega) E(x_2, y_2, z, \omega) \rangle$$





$N_x, N_y \in [100,1000]$.

W ~ 10^8 – 10^{12} (Gb-Tb)
Propagate: 4D integrals



4D function

2D functions

$$W(x_1, y_1, x_2, y_2, \omega) = \sum_m \lambda_m(\omega) \phi_m^*(x_1, y_1, \omega) \phi_m(x_2, y_2, \omega)$$

Store m x N x N Propagate: 2D integrals

CROSS SPECTRAL DENSITY

- *Mutual* Coherent Function (t-dependency) or Cross *Spectral* Density (w-dependency)

$$W(x_1, y_1, x_2, y_2, z, \omega) =$$
$$= \langle E^*(x_1, y_1, z, \omega) E(x_2, y_2, z, \omega) \rangle$$

- 8D function that propagates using a doble wave equation

- Spectral Density "intensity"

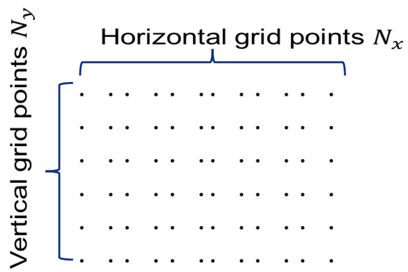
$$S(x, y, z, \omega) = W(x, y, x, y, z, \omega)$$

- Wide sense stationary: *W*≠0 if $w_1 = w_2 = w$
  Long bunch length, *high* w , *not to small* Dw
  Geloni et al. NIM A 588 463 (2008)

- Spectral Degree of Coherence

$$\mu(x_1, y_1, x_2, y_2, z, \omega) = \frac{W(x_1, y_1, x_2, y_2, z, \omega)}{\sqrt{S(x_1, y_1, z, \omega)}\sqrt{S(x_2, y_2, z, \omega)}}$$

- Decoupling z: 4D function (for given z and w)



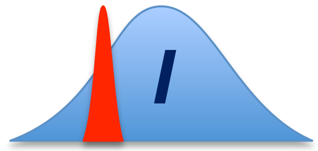
$N_x, N_y \in [100{,}1000]$.

W ~ 10^8 – 10^{12} (Gb-Tb)
Propagate: 4D integrals

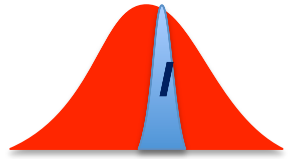
$$\underbrace{W(x_1, y_1, x_2, y_2, \omega)}_{\text{4D function}} = \sum_m \lambda_m(\omega) \underbrace{\phi_m^*(x_1, y_1, \omega) \phi_m(x_2, y_2, \omega)}_{\text{2D functions}}$$

SIMPLE CASE: 1D Gaussian SHELL-MODEL

$$W(x_1, x_2, \omega) = A^2 e^{-(x_1^2+x_2^2)/(4\sigma_I^2)} e^{-(x_2-x_1)^2/(2\sigma_\mu^2)}$$



Both intensity (Spectral Density) and correlation (Spectral Degree of Coherence) are Gaussians



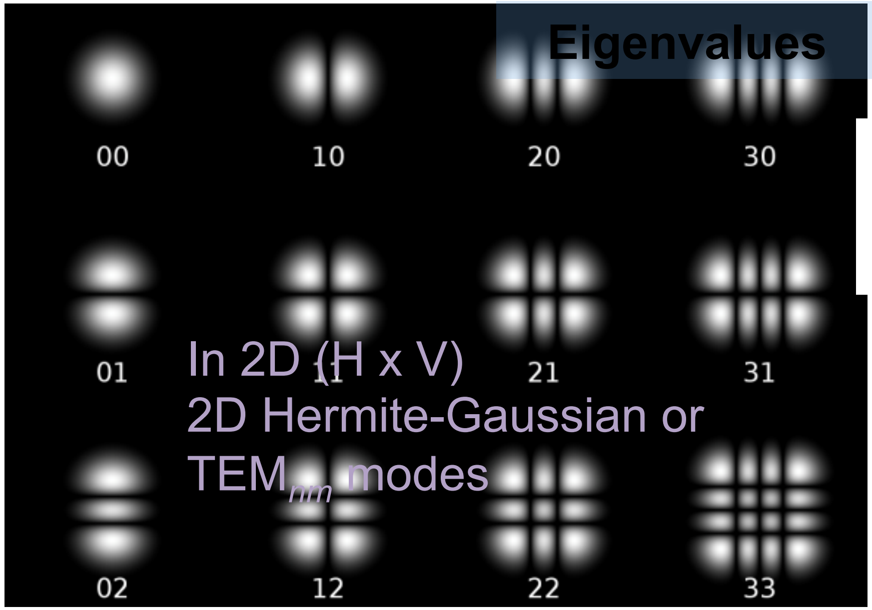
$S_I >> S_m$ the source is mostly incoherent (quasi homogeneous)

$S_I << S_m$ is mostly coherent

$$W(x_1, x_2, \omega) = \sum_n \boxed{\lambda_n(\omega)} \boxed{\phi_n{}^*(x_1, \omega)} \boxed{\phi_n(x_2, \omega)},$$

$$\beta = \frac{\sigma_\mu}{\sigma_I}$$

Eigenvalues



In 2D (H x V)
2D Hermite-Gaussian or
TEM_{nm} modes

Eigenfunctions
(Hermite-Gaussian modes)

$$\phi_n(x) = \left(\frac{2c}{\pi}\right)^{1/4} \frac{1}{\sqrt{2^n n!}} H_n(x\sqrt{2c})e^{-cx^2}$$

Magic property: Propagation invariance

In the first mode (Gaussian) : $\boxed{\sigma_x \sigma_\theta = \frac{\lambda}{4\pi}}$

The spectrum of coherent modes

Eigenvalues

$$\lambda_n = A\left(\frac{\pi}{a+b+c}\right)^{1/2}\left(\frac{b}{a+b+c}\right)^n$$

$$a(\omega) = \frac{1}{4\sigma_I{}^2(\omega)}, \qquad b(\omega) = \frac{1}{2\sigma_\mu{}^2(\omega)} \qquad c = (a^2 + 2ab)^{1/2}$$

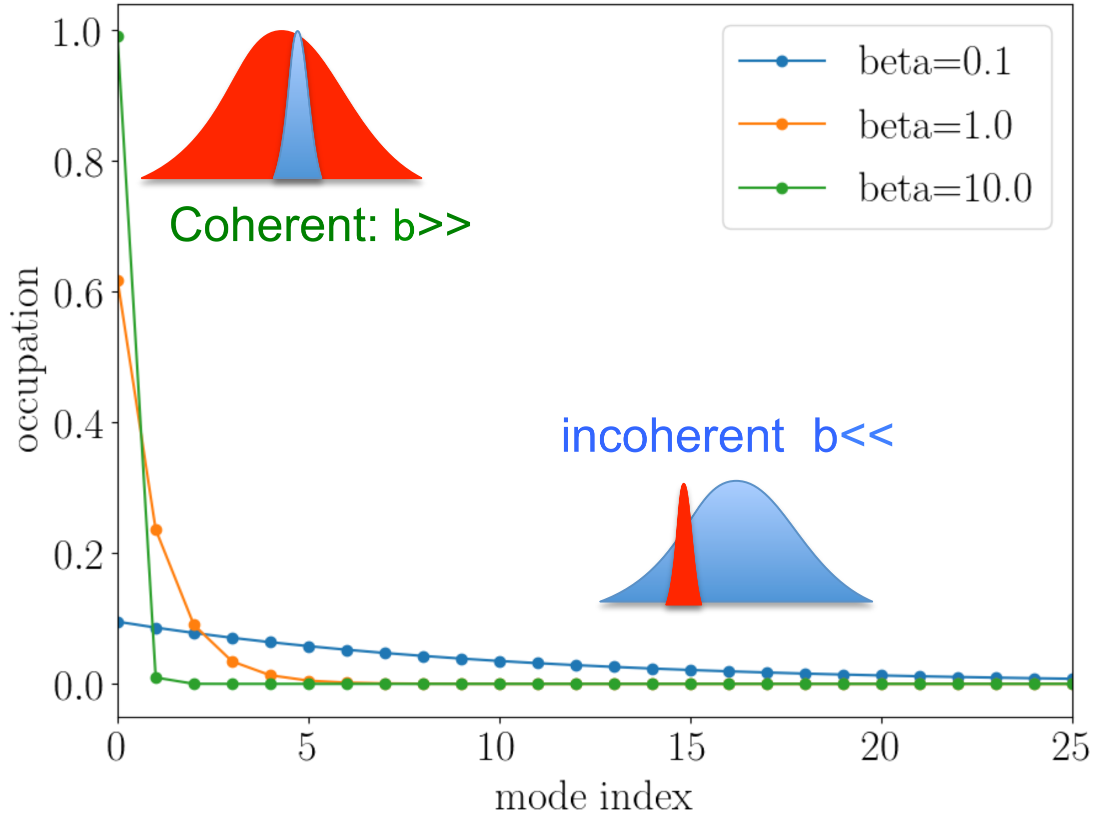
$$\beta = \frac{\sigma_\mu}{\sigma_I}$$

Mode occupation:

$$\eta_i(\omega) = \frac{\lambda_i(\omega)}{\sum_{n=0}^{\infty}\lambda_n(\omega)}$$

Coherent fraction:

$$CF = \frac{\lambda_0}{\sum \lambda_n}$$



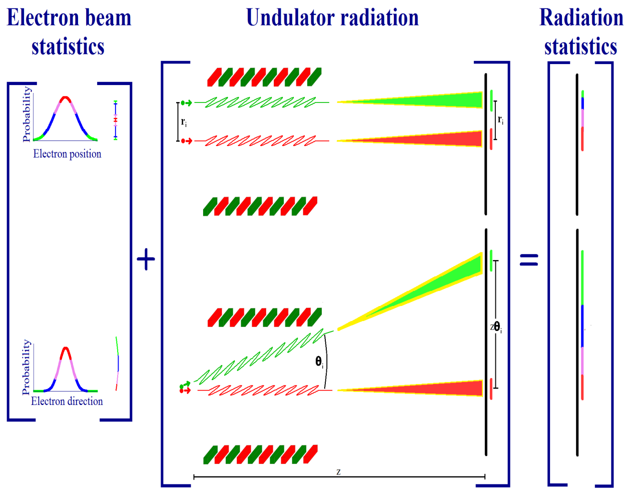
In general we cannot apply Gaussian Shell-model to synchrotron

COMSYL (Coherent Modes for Synchrotron Light)

SRW → Undulator radiation → Calculate CSD / Coherent mode decomposition

COMSYL https://github.com/mark-glass/comsyl

Glass & Sanchez del Rio (2017) EPL 119 p 34004



K.-J. Kim Proc. SPIE 0582 (1986)

Propagate CSD along the beamline

Rediagonalize CSD

OASYS

Friedholm equation:

$$A_W[\phi_n] = \lambda_n \phi_n$$

$$A_W[f](\boldsymbol{r}_2) = \int W(\boldsymbol{r}_1, \boldsymbol{r}_2, \omega) f(\boldsymbol{r}_1) d\boldsymbol{r}_1$$

$$\mathrm{M}_{ij} = \langle b_i | A_W | b_j \rangle$$

Diagonalize M with iterative solver slepc

- Large memory.
- Parallel computers using MPI.
- Large clusters (ESRF). Cloud computing (AWS, etc.)

Info

- Repository:
  https://github.com/mark-glass/comsyl
- Installation:
  https://github.com/mark-glass/comsyl/wiki

Install Comsyl-Oasys (not for COMSYL calculations, only for displaying and propagating results)

- pip install oasys-comsyl
- Start OASYS
- Download files (BIG!!) from: http://ftp.esrf.eu/pub/scisoft/comsyl/

- For coherent mode decomposition using COMSYL follow installation instructions in previous slide

Run

Run Comsyl

OASYS and Comsyl