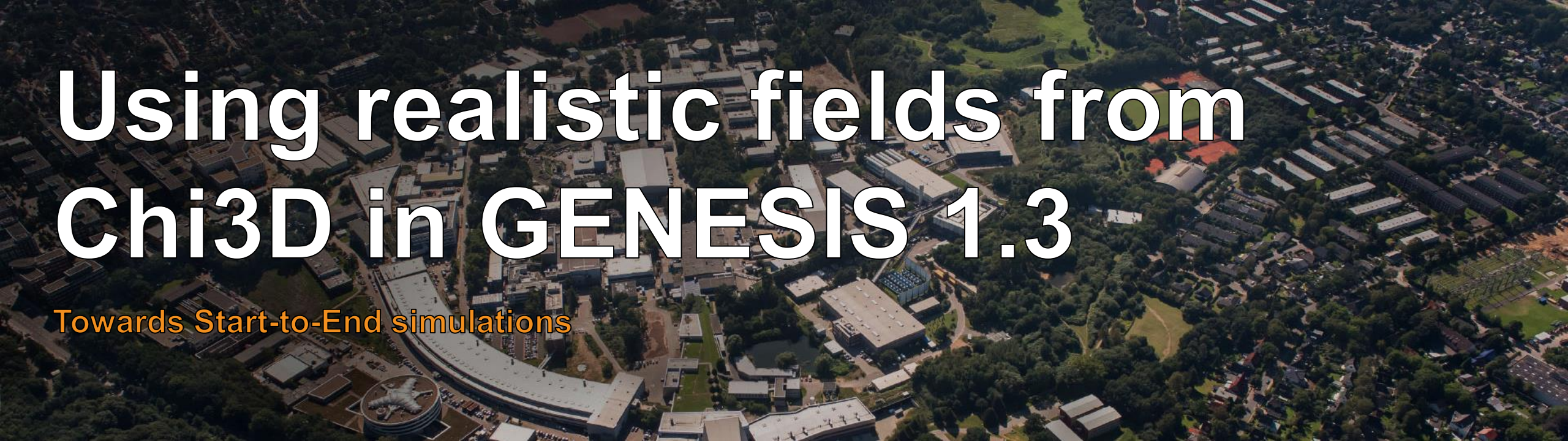


Using realistic fields from Chi3D in GENESIS 1.3

Towards Start-to-End simulations

Sven Ackermann

Hamburg, December 16th, 2021

HELMHOLTZ RESEARCH FOR GRAND CHALLENGES

DESY.

GENESIS1.3, version 4

A short reminder

- GENESIS1.3 is the name of the code. For convenience, one refers to the code by Genesis or G4.
- Time dependent
- 3D
- Undulator-period averaged (UPA)
- Uses MKS system
- Coordinate system is based on slices
- Version 4 finally implements:
  - HPC compatibility: Runs smoothly on Maxwell, Jules, …
  - Uses HDF5 instead of propietary binary file format
  - „Single electron“ mode uses one simulation particle per electron („one4one“)
  - Easier input formats (especially lattice- and simualtion configuration)

Sliced approach

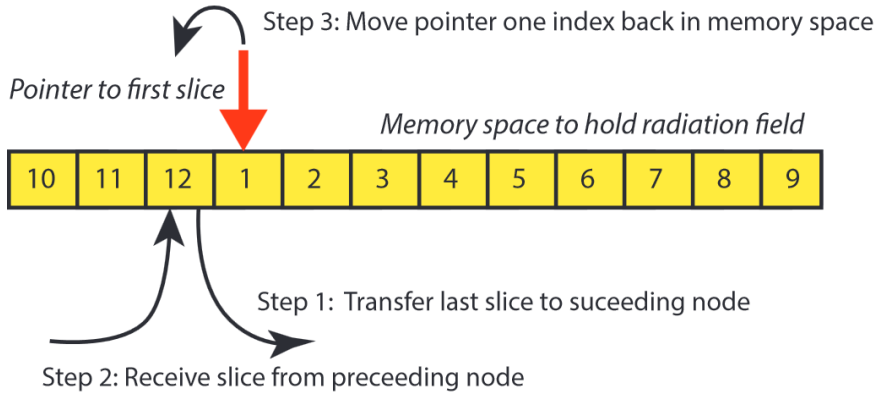
Electron beam and justification

- In Genesis, the electron bunch consists of slices
- Each slice is one radiation wavelength λ_{Rad} long
- The position of particles inside a slide N_{Slice} is given by its ponderomotive phase Θ
- $z = \left(\frac{\Theta - \pi}{2\pi} + N_{\mathrm{Slice}}\right) \cdot \lambda_{\mathrm{Rad}}$

Sliced approach

Why (it is elegant!)

- Due to the UPA approach, this is very elegant
  - The slippage is one radiation wavelength per undulator period
  - One only has to reduce N_{Slice} by one → Realized by moving the pointer to the memory address
  - Radiation field is considered being at a fixed position in time
- Particles with $(\Theta \lesssim 0) \vee (\Theta \gtrsim 2\pi)$ are transferred to next slice (in one4one mode)
  - Needed for the oversheering in the EEGH scheme
  - In former versions, electron with such phase would stay in the slice, like ~ $\Theta_{\mathrm{new}} = \mathrm{mod}(\Theta_{\mathrm{old}}, 2\pi)$
- This reduces a lot of overhead time for moving data between memory addresses



Sven Reiche: *Update on the FEL code GENESIS1.3*
in: Proceedings of FEL2014, Basel, Switzerland (TUP019)

Sliced approach

Photon field

- The photon field is calculated with the same longitudinal granulosity
- UPA approach → Field is averaged over one period
- The simulation takes place on a rectangular grid defined by
  - Grid size
  - Number of grid points (best practise: Odd number to have a center gridpoint)
- Each gridpoint contains the complex field amplitude
- Units are W and $\frac{\mathrm{W}}{\mathrm{m}^2}$

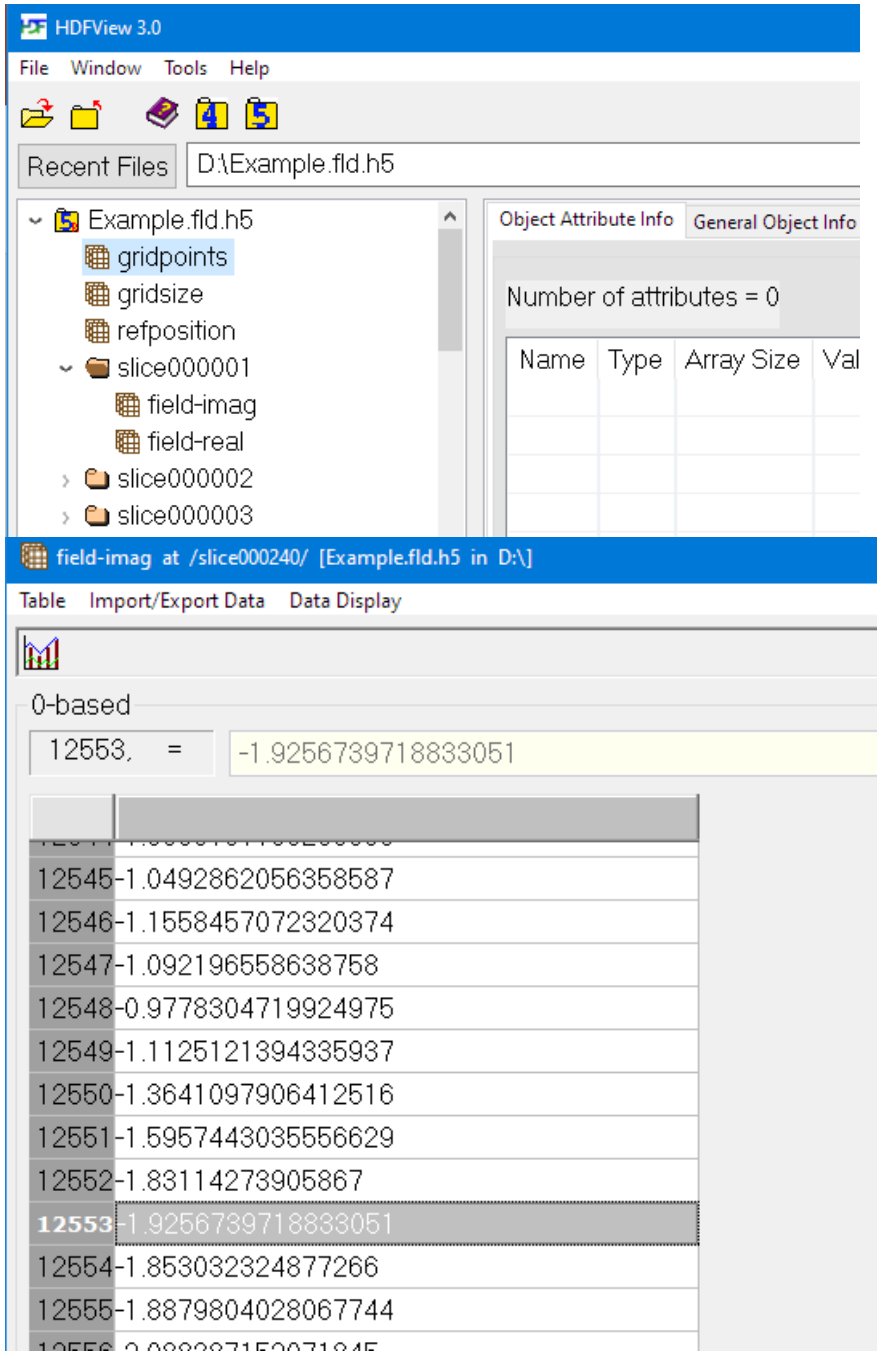
File format

HDF5 file for the radiation field

- The HDF5 file contains metadata
  - Gridpoints
  - Gridsize
  - Number of slices
  - Radiation wavelength
- And real data, organized in groups
  - \slice000001
    - field-imag ($\Im(E)$)
    - field-real ($\Re(E)$)
- The data is stored in 1D lists

1	2	3
4	5	6
7	8	9

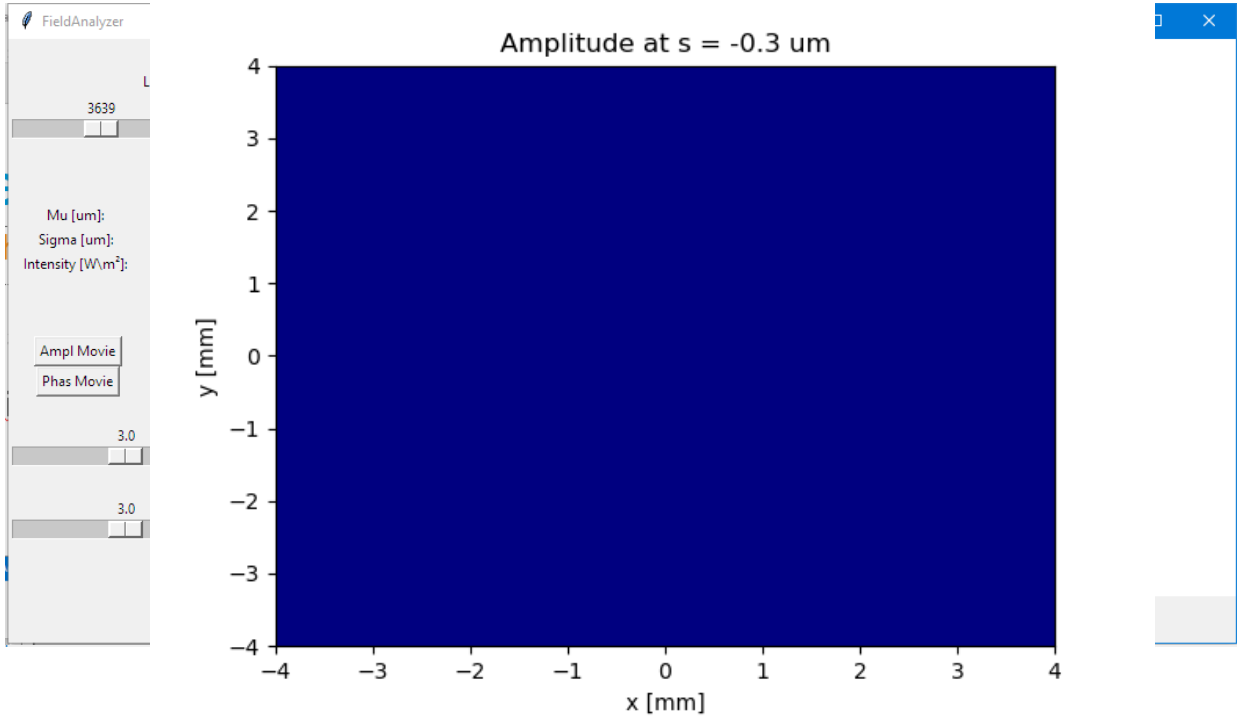
➔ (1, 2, 3, 4, 5, 6, 7, 8, 9)



File analysis

- A slice-wise viewer has been developed (Field.py)
- It is, together with all other tools, available publicly (without any guarantee) at

  https://www.desy.de/~ackerm/G4-Tools/




```
Filename      : D:/rad.fld.h5
Gridpoints    : 301
Gridsize      : 3.3333333333333333e-06
Ref.position  : 0.0
No. of slices : 21952
Slicespacing  : 7e-09
Wavelength    : 7e-09
```

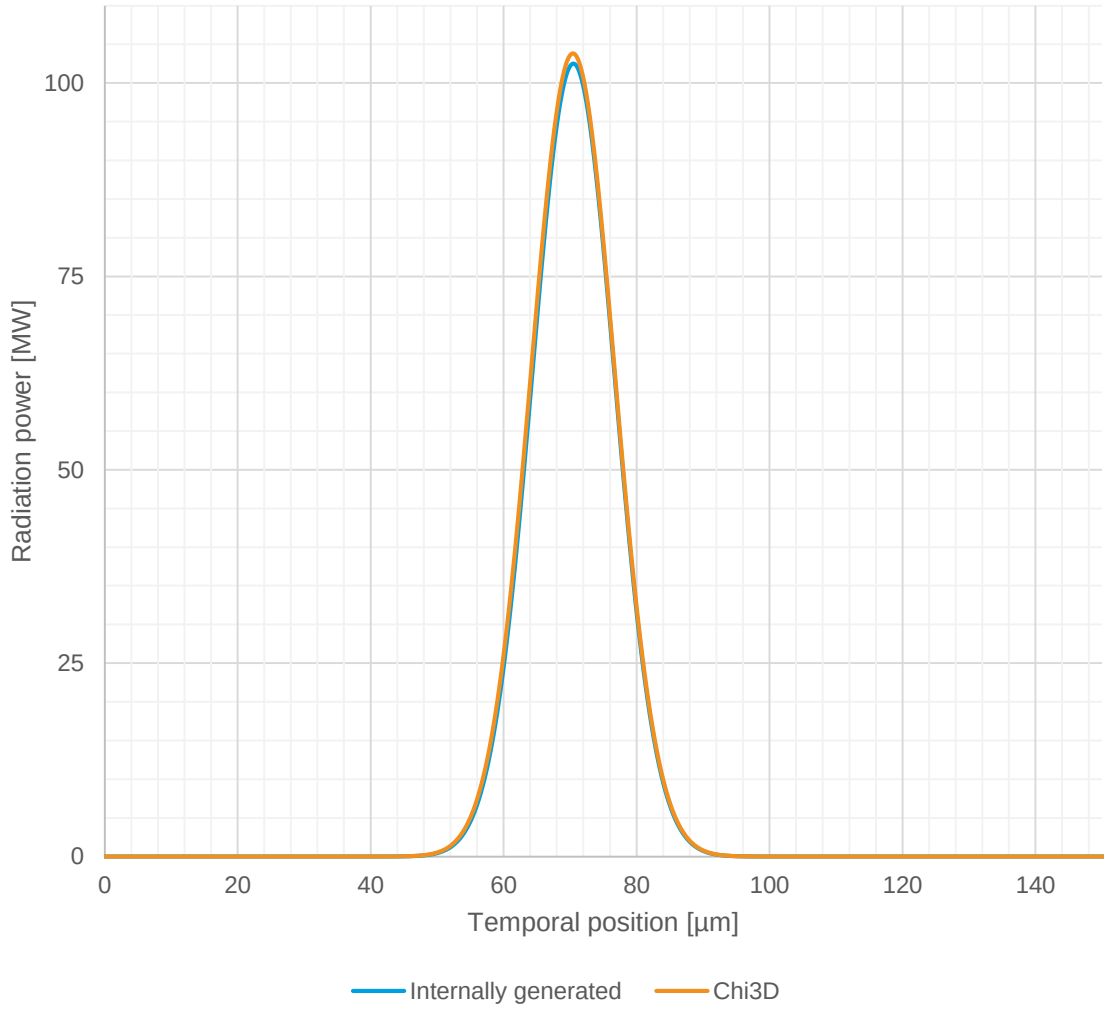
Conversion

- Thanks to HDF5 standard, little to no effort has to be taken into the mechanics storing data
- Arrange it in the way Genesis „expects“ the data
- Luckily, Genesis accepts additional fields → Field properties and additional information can be stored
- To test the conversion, the procedure is the following:
  - Generate field using Chi²
  - Convert the field towards Genesis field file in HDF5 format
  - Use that field file in a zero-length simulation and have Genesis save a field file
  - Compare the file after the simulation and the one before conversion
  - If they match, the conversion was done correctly

```
%HDF5 conversion
clc
name = fullfile('results',['chi3D_',datestr(now,'yymmdd_hhMMss'),'.h5']);%
mkdir('results')

h5create(name,'/gridpoints',1);%'Datatype','single'
h5write(name,'/gridpoints',gridpoints)

h5create(name,'/gridsize',1);
h5write(name,'/gridsize',obj.constVars.dx); %the exact gridsize in defined by the conversion process

h5create(name,'/refposition',1);
h5write(name,'/refposition',refposition)

h5create(name,'/slicecount',1);
h5write(name,'/slicecount',N_slices)

h5create(name,'/wavelength',1);
h5write(name,'/wavelength',obj.constVars.dt*3e8); %due to convertion the defined center wavelength can differ slight

h5create(name,'/slicespacing',1);%
h5write(name,'/slicespacing',obj.constVars.dt*3e8) %the exact slicespacing in defined by the conversion process

f = waitbar(0,'convert chi3D to HDF5...');
for i=1:N_slices
    waitbar(i/N_slices,f,'convert chi3D to HDF5...');

    N=gridpoints^2;
    Exy_i = gather(squeeze(Etxy(i,:,:)));

    h5create(name,['/slice',num2str(i,'%06i'),'/field-imag'],N);
    h5create(name,['/slice',num2str(i,'%06i'),'/field-real'],N);

    h5write(name,['/slice',num2str(i,'%06i'),'/field-imag'],imag(Exy_i(:))');
    h5write(name,['/slice',num2str(i,'%06i'),'/field-real'],real(Exy_i(:))')
end

pulseProperties = fields(obj.simResults.UV);
for i=1:length(pulseProperties)
    h5create(name,['/pulseProperties/',pulseProperties{i}],1);
    h5write(name,['/pulseProperties/',pulseProperties{i}],obj.simResults.UV.(pulseProperties{i}))
end

waitbar(1,f,'done');
% h5disp(name)
```

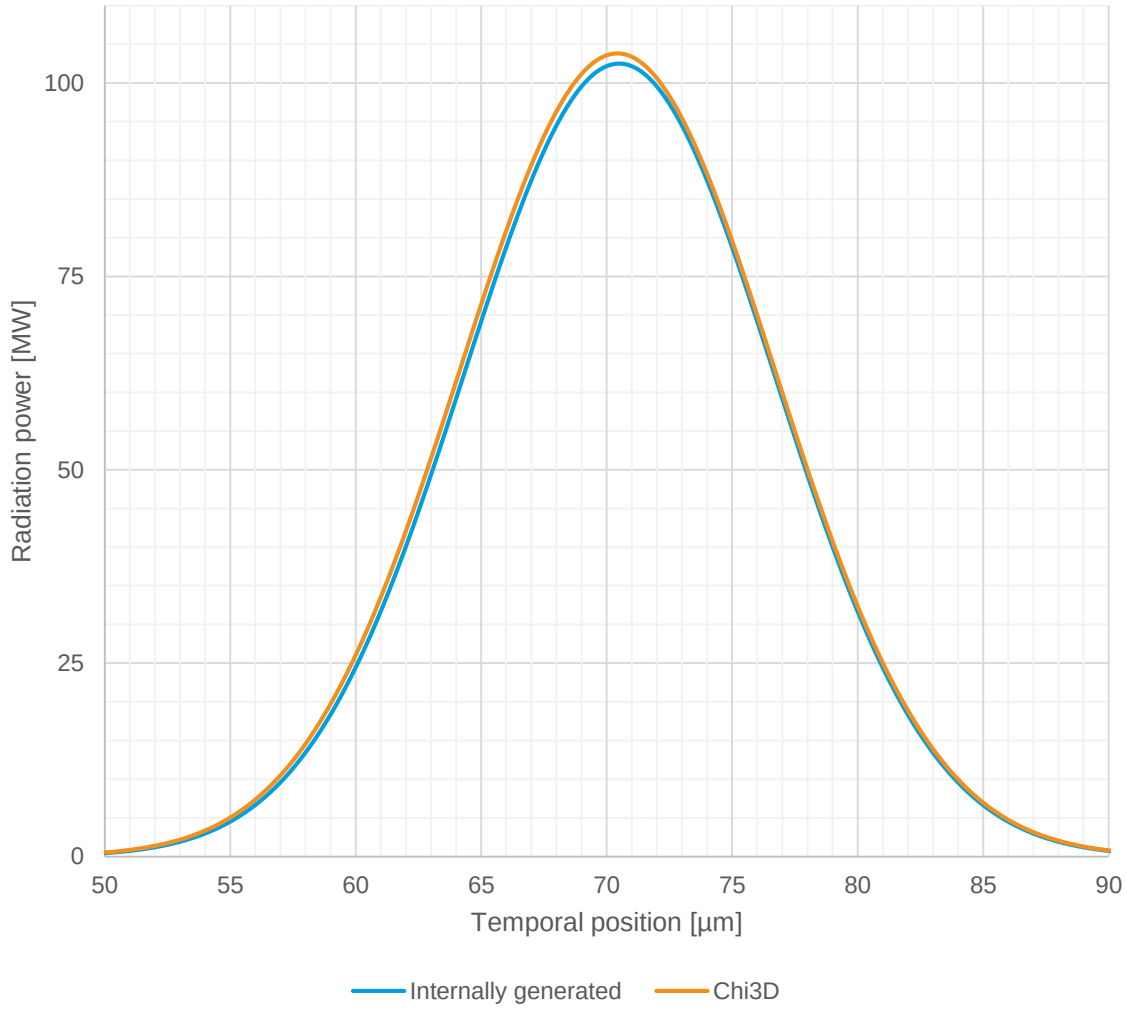
Current state

- Comparison between internal generation and Chi3D
- First successful run yesterday:
  - Peak power: 102,51 MW
  - Center at 70.5 µm
  - Sigma at 6.2 µm
- Still some small deviation, but <1%



Current state

- Comparison between internal generation and Chi3D
- First successful run yesterday:
  - Peak power: 102,51 MW
  - Center at 70.5 µm
  - Sigma at 6.2 µm
- Still some small deviation, but <1%



Conclusion

- We have powerful tools
  - Chi3D for photon fields
  - GENESIS for the FEL process
- We can user our code to transfer fields between them
- Not everything is perfect (yet)
- First real simulations can now start using realistic laser beam profiles

➔ Thanks to everyone who contributed to the process (especially Tino, Eugenio, Fabian)

Contact

DESY. Deutsches
Elektronen-Synchrotron

www.desy.de

Sven Ackermann
FS-FLASH
sven.ackermann@desy.de
+49 (0)40 8998-6239