Simulation of CXRS spectra using Raysect and CHERAB frameworks

Aleksei Shabashov

Contents

1 Purpose

2 Scope

This document is applicable to the 55.EC CXRS Edge diagnostic. The CXRS Edge system is an optical diagnostic that collects the light emitted by the plasma upon interaction with the Diagnostic Neutral Beam (DNB) and analyses this light to extract the ion temperature, plasma rotation velocities and impurity content of the plasma. An overview of the CXRS Edge system is given in the Design Description document (DDD) [1].

3 Definitions

For a complete list of ITER abbreviations see [2]. Below abbreviations used in this document are given.

CXRS	Charge eXchange Recombination Spectroscopy
DNB	Diagnostic Neutral Beam

4 Introduction

4.1 CXRS Diagnostics

4.1.1 CXRS for Plasma Measurements

The Charge Exchange Recombination Spectroscopy (CXRS) diagnostic measures line emissions of several low Z impurities in the plasma due to interaction with a neutral beam.

The Charge Exchange Recombination Spectroscopy (CXRS) diagnostics measures line emissions of several impurity isotopes in the plasma excited by charge exchange reactions with neutral hydrogen atoms injected into the plasma by the diagnostic neutral beam (DNB). This line emission provides essential information for plasma control and physics studies: ion temperature T_i, toroidal (v_{tor}) and poloidal (v_{pol}) plasma rotation velocity, helium ash and low Z impurity densities (beryllium, carbon, neon etc.) and the derived quantities such as Zeff. The CXRS Edge diagnostic is a distributed system with components throughout the ITER tokamak complex. The primary viewing components (front-end optics) are installed in Equatorial port 3 (EP3), viewing the Diagnostic Neutral Beam (DNB) that enters the plasma in the neighboring section 4. Front-end optics consists of two light collecting systems: Upper and Lower. The scope of the 55.EC CXRS Edge diagnostic for Upper system ends at the image plane in the port interspace (11-L1-C03) where the collected signal is coupled into optical fibre bundles. For Lower system, however, 55.EC CXRS Edge diagnostic is responsible for light collection system, light transportation and detection. The collected signal is transported through optical fibre bundles to the Tritium building (Building 14 – Level 2 – Room 4), where the detection systems are located.

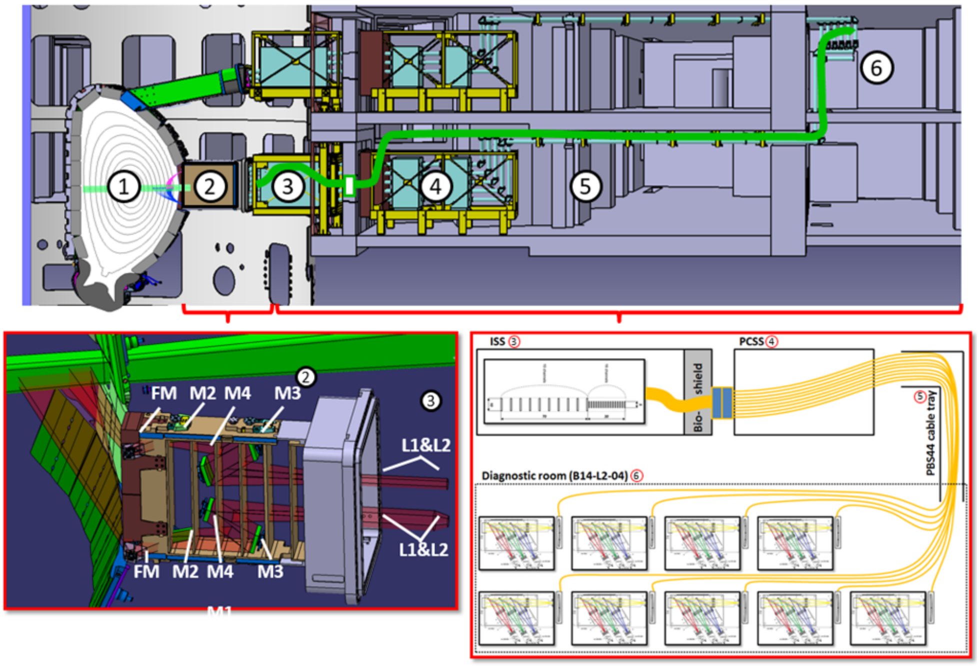


Figure 1. An overview of the CXRS Edge diagnostic design.

Measured parameters:

- Line's width – ion temperature:

$$kT_{\text{ion}} = mc^2 \frac{\Delta\lambda^2_{\text{Dopp}}}{\lambda_0^2} \tag{1}$$

- Line center's shift — toroidal and poloidal velocities:

$$v_{\text{rot}} = c\frac{\Delta\lambda_{\text{rot}}}{\lambda_0} \tag{2}$$

- Line's intensity — impurity's density and Z_{eff}:

$$n_{\text{imp}} = \frac{4\pi \int I(\lambda)\, d\lambda}{n_{\text{b}} Q^{\text{eff}}_{\text{CX}}(v_{\text{b}})\, dl} \tag{3}$$

Ion	Transition	Wavelength, nm
BeIV	6 → 5	465.8 nm
BeIV	8 → 6	468.5 nm
HeII	4 → 3	468.5 nm
ArXVIII	16 → 15	522.5 nm
NeX	11 → 10	524.9 nm
CVI	8 → 7	529.1 nm
Hα		656.3 nm
MSE		659.1 nm

Table 1. Spectroscopic lines observed by CXRS

$$X^{Z+} + H^0 \to X^{(Z-1)+}(n_2) + H^+ \tag{4}$$

$$X^{(Z-1)+}(n_2) \to X^{(Z-1)+}(n_1) + h\nu \tag{5}$$

4.1.2 CXRS in ITER

Geometry

Equipment

IMAS Database

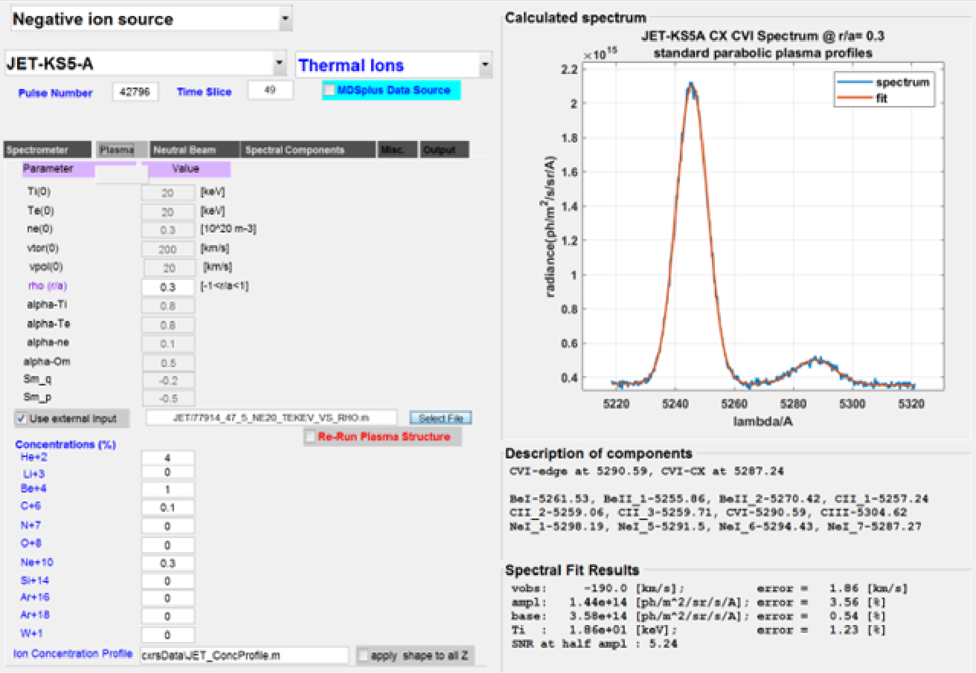
4.2 Development of the New Simulation Code

4.2.1 Existing Code

Simulation of Spectra (SOS) code by M. G. von Hellermann [3]

Features:

- Simulation takes into account many physical effects (halo effect, crossection effect, plume effect and others);
- Written in Matlab;
- Has Graphical User Interface (fig. 2).



Negative ion source

JET-KS5-A

Thermal Ions

Pulse Number: 42796 Time Slice: 49 □ MDSplus Data Source

Parameter	Value	
Ti(0)	20	[keV]
Te(0)	20	[keV]
ne(0)	0.3	[10^20 m-3]
vtor(0)	200	[km/s]
vpol(0)	20	[km/s]
rho (r/a)	0.3	[-1<r/a<1]
alpha-Ti	0.8	
alpha-Te	0.8	
alpha-ne	0.1	
alpha-Om	0.5	
Sm_q	-0.2	
Sm_p	-0.5	

☑ Use external input: JET/77914_47_5_NE20_TEKEV_VS_RHO.m

□ Re-Run Plasma Structure

Concentrations (%)	
He+2	4
Li+3	0
Be+4	1
C+6	0.1
N+7	0
O+8	0
Ne+10	0.3
Si+14	0
Ar+16	0
Ar+18	0
W+1	0

Ion Concentration Profile: cxrsData\JET_ConcProfile.m □ apply shape to all Z

Description of components

```
CVI-edge at 5290.59, CVI-CX at 5287.24

BeI-5261.53, BeII_1-5255.86, BeII_2-5270.42, CII_1-5257.24
CII_2-5259.06, CII_3-5259.71, CVI-5290.59, CIII-5304.62
NeI_1-5298.19, NeI_5-5291.5, NeI_6-5294.43, NeI_7-5287.27
```

Spectral Fit Results

```
vobs:     -190.0 [km/s];              error =   1.86 [km/s]
ampl:   1.44e+14 [ph/m^2/sr/s/A]; error =   3.56 [%]
base:   3.58e+14 [ph/m^2/sr/s/A]; error =   0.54 [%]
Ti  :   1.86e+01 [keV];               error =   1.23 [%]
SNR at half ampl : 5.24
```

Figure 2. SOS interface.

4.2.2 Motivation

Existing code (Simulation of Spectra - SOS) lacks some features:

- Simplified plasma, tokamak and diagnostic geometry
  (e.g. elliptical plasma, point emission and others);
- Does not take reflections into account;
- Cannot use data from IMAS directly;
- Requires Matlab license, hard to extend by new developers.

The goal was to create an open and extensible simulation code using Python.

Sub goals:

- Implement interaction with IMAS database (read and write);
- Use IMAS data to create a plasma and diagnostic beam with spatial distributions;

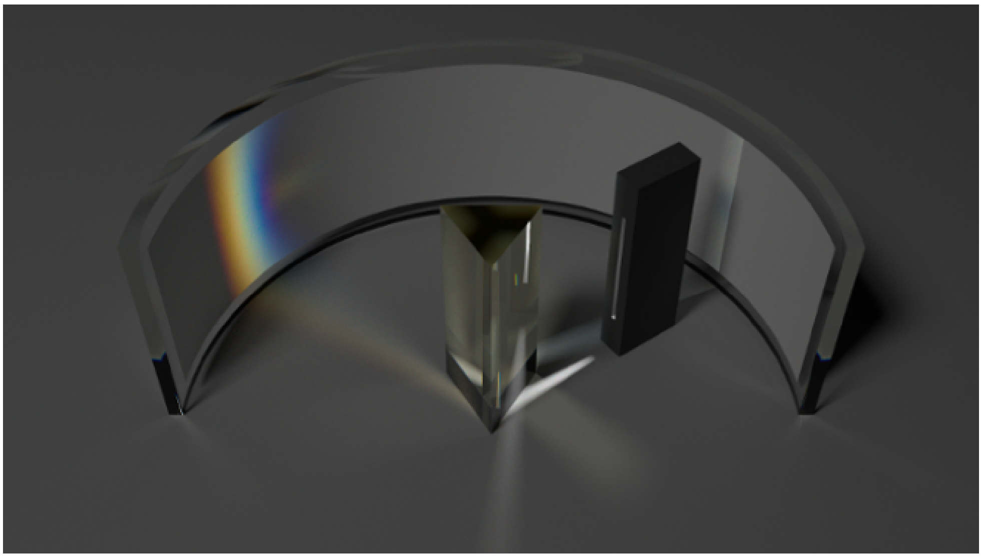


Figure 3. Demonstration of Raysect features.

- Use a ray-tracing engine to simulate spectra, this includes how reflections affect simulated spectra;
- Ensure that emission models include all physics already captured by SOS.

4.3 Raysect and CHERAB

Raysect [4] is a ray-tracing framework for Python designed for scientific purposes.

- Supports scientific ray-tracing of spectra from physical light sources such as plasmas.
- Easily extensible, written with user customisation of materials and emissive sources in mind.
- Different observer types supported such as pinhole cameras and optical fibres.

CHERAB [5] is a Python library for forward modelling diagnostics based on spectroscopic plasma emission which provides physical models for Raysect. Provided models for Raysect:

- Tools for plasma and diagnostic beam simulations;
- Physical emission models (active charge exchange, bremsstrahlung and more).

5 For Users

5.1 Installation

Clone this repository then go to its root folder:

```
git clone ssh://git@git.iter.org/diag/cxrs.git
cd cxrs
```

At first you have to load all modules needed:

```
source env.sh
```

This will purge all loaded modules and will load modules necessary to install and use cxrs. To install cxrs on your computer run

```
pip install --user .
```

while in the the cxrs root directory.

Note: you can omit python -m part later by adding $HOME/.local/bin to your PATH. To do so, locate file .bashrc in your home directory and add the following line at the end of the file:

```
export PATH="$PATH:/home/ITER/<username>/.local/bin"
```

where <username> is your username at ITER GPC. You can look at it by executing next command via command line:

```
echo $USER
```

5.2 Preparations for the first run

Step 1: Create environment file

In order to create environment file, run

```
python -m cxrs create-env
```

This will create the default environment file, that loads all needed modules. To do so, run

```
source env.sh
```

It will purge all loaded modules and will load modules necessary to use cxrs.

Note: you have to do this for every new session on ITER GPC.

Step 2: Create configuration file

cxrs behaviour controlled by configuration file which is necessary to run the code. To create default configuration file, run

```
python -m cxrs create-config
```

This will create a configuration file with a name config.xml in current directory.

Step 3: Populate local atomic database

cxrs uses a local atomic database. To create one, run

```
python -m cxrs populate
```

Atomic data will be copied to the $HOME/.cherab directory.

5.3 Usage

5.3.1 Performing a simulation

Main part of cxrs is simulation routine, to use it run

```
python -m cxrs simulate -s <shot> -r <run> -t <time> -c <config>
```

It will perform a simulation and will store result in newly created folder cxrs_output.

time can be omitted. In that case code will use a time slice in the middle of available time range for a simulation.

For more information use help:

```
python -m cxrs simulate --help
```

5.3.2 Print plasma profiles

cxrs info subprogram can be used to inspect different distributions of plasma and DNB parameters. To use it, run

```
python -m cxrs info -s <shot> -r <run>
```

Results will be stored in cxrs_output folder.

Note: 1D profiles present values at the DNB axis.

5.3.3 Print plasma composition

You can quickly look at plasma composition for requested pulse by using

```
python -m cxrs composition -s <shot> -r <run>
```

Omit arguments to print composition for all pulses available by scenario_summary program.

5.3.4 Configuration

cxrs accepts user's configurations as xml-file.

To change option's value, change value attribute in double quotes corresponding to the option.

For more information on the meaning of certain parameter, look at its description.

Note: type attribute is actually important. For string values use str, for floating-point values use float and int for integers.

Warning: Do not change tags, like <user_options> or others, this will stop the program from
.

5.3.5 Emission parameters

Section `emission_parameters` controls which emission types are used during a simulation. Each of them can be turned on or off separately.

5.3.6 Plasma, beam, fibre, optics, camera, scanner and spectrometer parameters

Each of those sections controls appropriate aspect of simulation. For information on each parameter look at its description in configuration file.

5.3.7 Emission lines

List of emission lines suited for observations is placed in `<emission_parameters>` section:

```
<emission_lines description="CXRS elements to observe.">
    <HI>
        <name
            description="Name of the element."
            unit="n.a."
            type="str"
            value="hydrogen"/>
        <charge
            description="Charge of the ion."
            unit="n.a."
            type="int"
            value="0"/>
        <transition_levels
            description="Transition levels [upper, lower]."
            unit="n.a."
            type="int"
            value="[3, 2]"/>
    </HI>
    ... more lines omitted
</emission_lines>
```

You can change and add new entries in this section following given template and added emission lines will be simulated if atomic data exists for them.

Note: isotopes have to stated as separate structure. For example, line of helium-4 can be added as

```
<He4II>
    <name
```

```
        description="Name of the element."
        unit="n.a."
        type="str"
        value="helium4"/>
    <charge
        description="Charge of the ion."
        unit="n.a."
        type="int"
        value="1"/>
    <transition_levels
        description="Transition levels [upper, lower]."
        unit="n.a."
        type="int"
        value="[4, 3]"/>
</He4II>
```

5.3.8 DNB

DNB section of the configuration file contains parameters for Diagnostic Neutral Beam. Structure of this section replicates structure of nbi IDS.

5.3.9 Wavelength ranges and geometry

These two sections contain all information for CXRS diagnostics.

6 For Developers

6.1 Project Structure

6.2 Reading data from IMAS

6.2.1 `equilibrium` IDS

6.2.2 `core_profiles` IDS

6.2.3 `edge_profiles` IDS

6.2.4 `charge_exchange` IDS

6.2.5 `nbi` IDS

6.2.6 Supplementary Functions

6.3 Setting the Wall

6.4 Observers

6.4.1 Base Class

6.4.2 Sightlines

6.4.3 Optics

6.4.4 Fibres

Separate Fibres

Fibre Bundle

6.4.5 Camera

6.4.6 Others

Scanner

Total Radiance

Spectrometer

6.5 Populating CHERAB Atomic Database

6.6 Utility Functions

6.6.1 Parsing XML Configuration File

6.6.2 Setting Emission Parameters

6.6.3 Math Functions

6.6.4 Others

References

[1] Zvonkov A., Serov S., and Tugarinov S. *System Design Description (DDD) 55.EC CXRS Edge*. Version 4.1. Apr. 2020.

[2] *ITER Abbreviations*. Version 1.17. Mar. 2018.

[3] Manfred von Hellermann et al. "Simulation of Spectra Code (SOS) for ITER Active Beam Spectroscopy". In: *Atoms* 7.1 (Mar. 2019), p. 30. DOI: 10.3390/atoms7010030.

[4] Dr Alex Meakins and Matthew Carr. *raysect/source: v0.5.2 Release*. Version v0.5.2. Aug. 2018. DOI: 10.5281/zenodo.1341376. URL: https://doi.org/10.5281/zenodo.1341376.

[5] Dr Carine Giroud et al. *CHERAB Spectroscopy Modelling Framework*. Version v0.1.0. Mar. 2018. DOI: 10.5281/zenodo.1206142. URL: https://doi.org/10.5281/zenodo.1206142.