The CXSFIT code development

Allan Whiteford (ADAS)

M G von Hellermann (FOM/Jülich)

L D Horton, C F Maggi, J Schirmer (IPP-Garching)

K-D Zastrow, T M Biewer*, C Giroud and A G Meigs (UKAEA/JET)

* Permanent institution: ORNL

13th November 2006

Contents

- Background.
- KS4FIT.
- CXSFIT.
- Graphical Interface.
- Fitting options available.
- Fit history methodology and automatic processing.
- Benchmarking and testing.

Background

- Fitting charge exchange spectra has always been a complicated process:
  - on the fly wavelength calibration (based on Be line position),
  - the passive signals means a one Gaussian fit is usually not possible,
  - other lines make it difficult to fit the background,
  - using carbon temperatures to aid in the fitting of helium spectra,
  - coupling line positions together based on known wavelengths,
  - and many many more.

- von Hellermann and co-workers developed advanced techniques to solve all of these issues at JET — resulted in a computer code called KS4FIT.

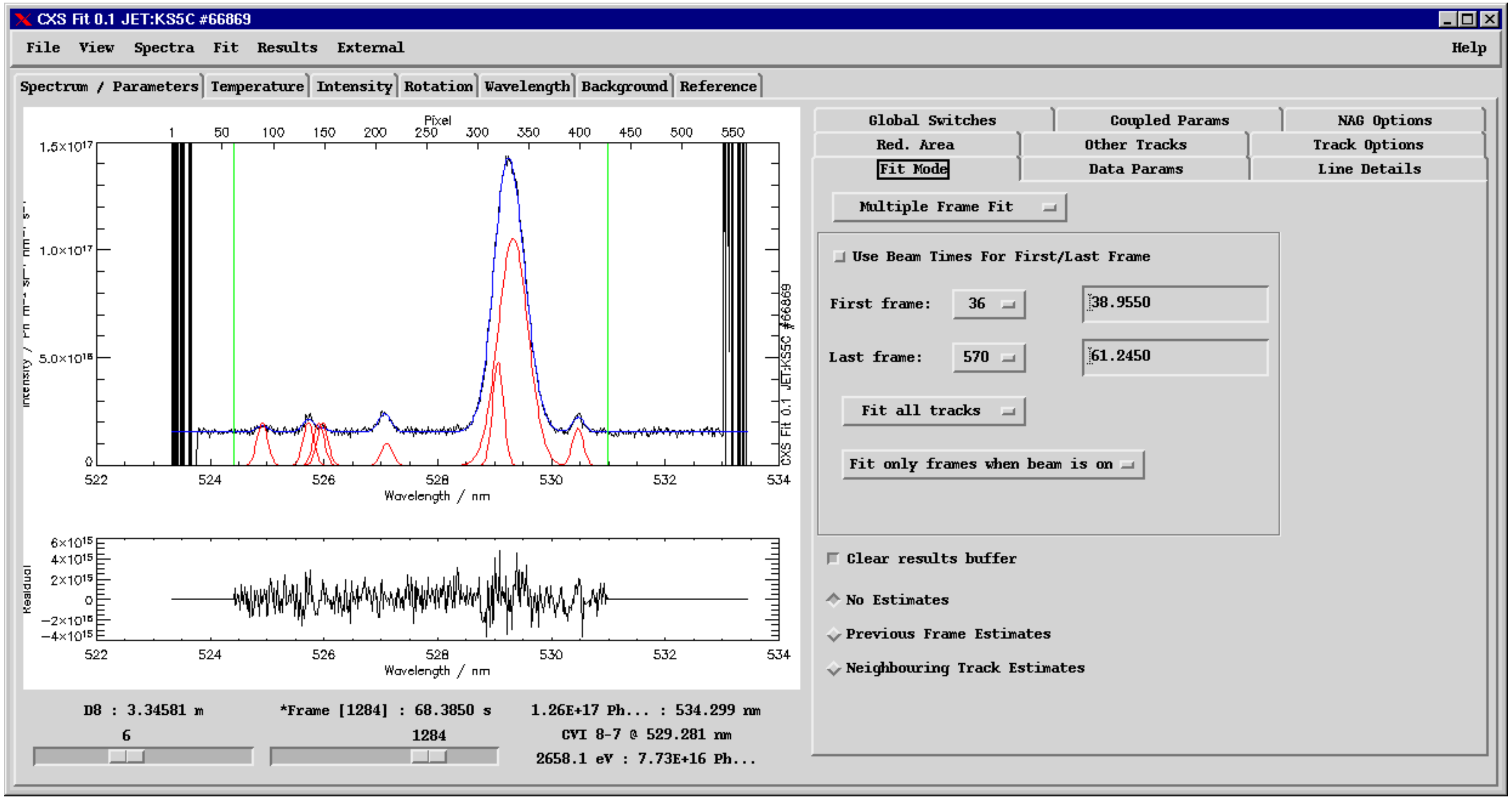
KS4FIT

- Practical implementation of charge exchange fitting at JET is called KS4FIT:
  - originally written for the IBM but ported to Linux,
  - reads spectra (JPFs) and writes results (PPFs),
  - has a TSO-style interface.
- However, KS4FIT exists outside of JET where it is a different thing:
  - same core fitting algorithm,
  - input and output are completely different,
  - packaged along with InSPECtor (JAVA fitting code).
- It's the latter version of KS4FIT (the pure fitting engine) which CXSFIT uses.

CXSFIT

- CXSFIT is a joint development between ADAS, FOM, Garching, Jülich and UKAEA to provide a universal interface to KS4FIT.
- Graphical user interface written in IDL.
- Contains all of the features present in KS4FIT.
- Provides visualisation of each fit and of the overall results.
- Almost all of the code is machine independent:
  - Machine specific reading/writing routines need to supplied.

The Graphical Interface



Dropdown menu overview

- Spectra Menu:
  - UTC-style Load / Save / Save as procedures,
  - Read AUG, JET or TEXTOR spectra.

- Fit Menu:
  - UTC-style Load / Save / Save as procedures,
  - Fit now (including single frame override),
  - Options relating to history (covered later).

- Results Menu:
  - UTC-style Load / Save / Save as procedures,
  - Write AUG, TEXTOR or JET results.

Fitting options (1/3)

- Fit Mode:
  - type of fit (single, multiple, repair etc.),
  - time window of fit,
  - automatic estimates (previous frame, neighbouring tracks etc.).
- Data Params:
  - pixel ranage to fit (also adjustable by dragging bars),
  - selection of CX and Reference lines.
- Global Switches:
  - ability to turn on and off various options (11 in total),
  - e.g. fixed values, parameter bounds etc.

Fitting options (2/3)

- Line Details:
  - naming and initial estimates for each line (also adjustable by dragging lines),
  - parameter bounds and selection of fixed values,
  - specification of theoretical wavelength and transition (upper/lower n etc.).
- Coupled Params:
  - couple different lines together,
  - temperature, width, position and height,
  - can be set the same, or with multiplicative factors or additive constants.
- Reduced Areas:
  - blank out regions of the spectra (also adjustable by dragging bars).

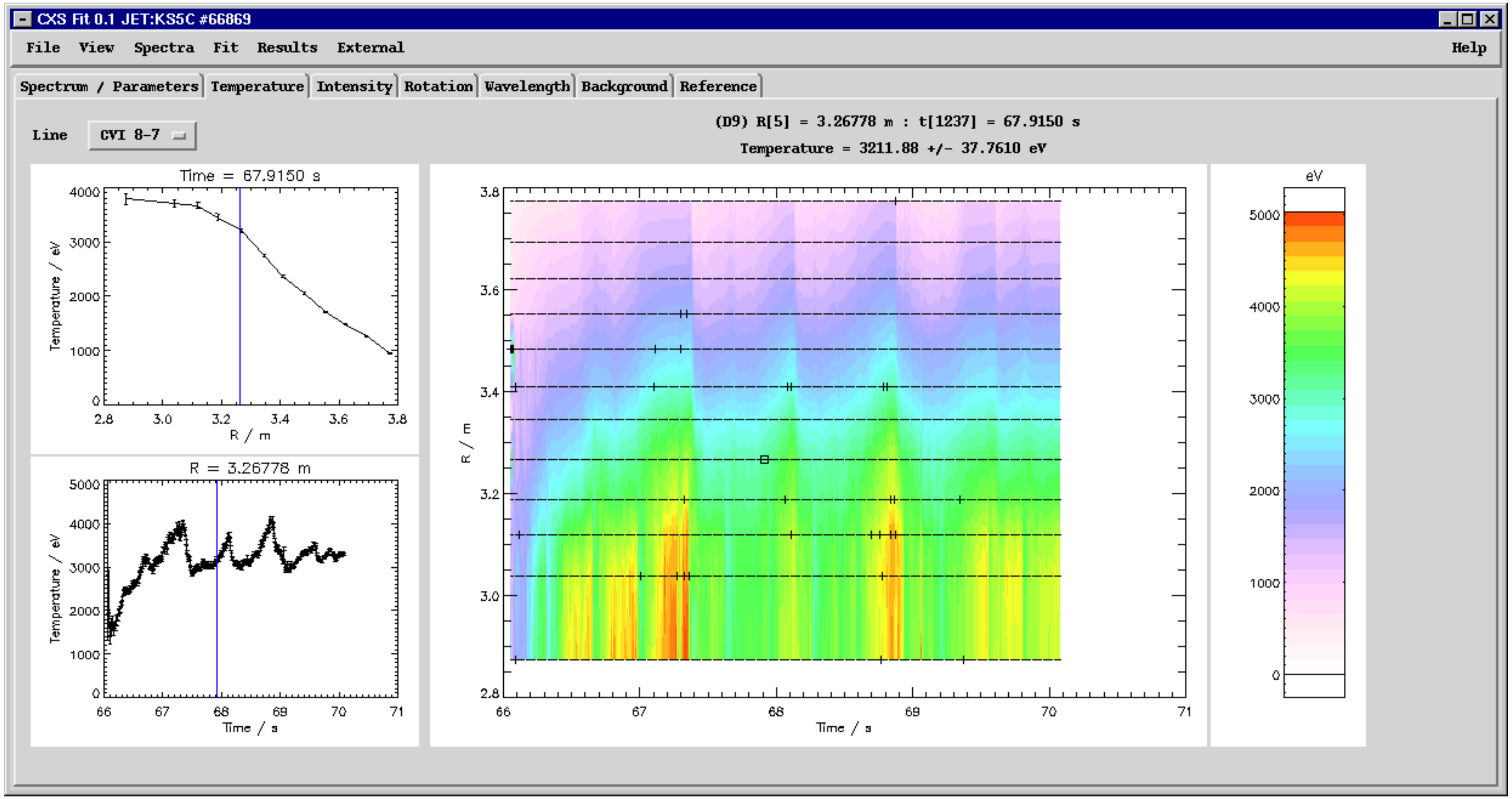
Fitting options (3/3)

- Other Tracks:
  - allows estimates of one parameter from the results of another,
  - typically used for estimating the passive emission from the edge and using it at the core.

- Track Options:
  - option to fit a track or not,
  - pixel corrections on a track by track basis to wavelength calibration.

- NAG Options:
  - allows control of underlying NAG routines.

Preview of results



Repairing spectra

- Even with all the options available, usually a few of the fits don't converge.
- Marked with crosses (poor fits) or stars (failed fits) in the output preview.
- Options exist to refit the failed or poor frames with new fit options:
  - 'standard' current procedure at JET is to use the previous frame estimates,
  - sometimes necessary to do individual fits by hand,
  - conceivable that different time zones may need different setup entirely.
- However, this means that to reproduce results or use the same recipe on a different shot there are multiple parameter settings which need to be used.

Fit History

- CXSFIT stores the fit history, i.e.:
  - The list of steps used to produce the current fit in terms of what the fit options were set to each time a fit was initiated.
- Fit histories can be saved as default recipes (also written to default output).
- Very simple history might be:
  1. fit all frames with beam on for all times,
  2. fit any failed frames using previous frame estimates,
  3. fit any remaining failures using neighbouring tracks option.
- Allows the user to load in a spectrum, load a recipe and then “replay” the history. Standard recipes can be developed for particular instruments.

The command line

- CXSFIT can be controlled from the command line for convenience or for batch processing, examples are:

- Load a spectrum at startup:
  - `cxsfit cer:17148`

- Do batch processing using the same recipe:
  - `cxsfit ks5c:66869 ks5c_carbon8-7.fit replay save ks5c_66869.cxf quit`
  - `cxsfit ks5c:66870 ks5c_carbon8-7.fit replay save ks5c_66870.cxf quit`
  - `cxsfit ks5c:66871 ks5c_carbon8-7.fit replay save ks5c_66871.cxf quit`

- Reload a previously saved setup:
  - `cxsfit ks5c_66869.cxf`

Testing and benchmarking

- Benchmarking between CXSFIT and the TSO KS4FIT has been done at JET.
- Results were almost identical (expected since the core code is the same).
- Extensive testing has been done on AUG for a number of spectra. Including helium spectra using external estimates.
- External estimates not tested at JET yet but should work.
- The output of CXSFIT at JET can be processed by the current version of CHEAP after some development.
- Possible CHEAP re-development a much bigger issue.

Current issues and future work

- Non-constant dispersion.
- Track dependent instrument functions.
- Pixel dependent instrument functions:
  - Probably will be automatically solved when non-constant dispersion is handled.
- Necessary for efficient treatment of KS5D and KS5E:
  - track to track instrument functions are currently handled inefficiently by CXSFIT (multiple calls to KS4FIT),
  - non-constant dispersion is taken account of by remapping the intensity and errors on to an artificially constant dispersion grid as the data are read.

Conclusions

- All of the KS4FIT functionality is present in CXSFIT.
- Portable to other machines, just require specific reading/writing routines.
- Tested at JET and AUG, results are good.
- Maintained alongside ADAS at Strathclyde.
- Can be distributed with ADAS?
- A laptop will be running with a demonstration at this meeting.