CXSFIT - Charge Exchange Fitting

Allan Whiteford (ADAS)

M G von Hellermann, Thijs Versloot (FOM/Jülich)

L D Horton, C F Maggi, J Schirmer, T Pütterich, R McDermott (IPP-Garching)

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

* Permanent institution: ORNL

15th September 2009

ADAS-EU Training Course

Contents

- Background.
- KS4FIT.
- CXSFIT.
- Graphical Interface.
- Feature highlights.
- 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,
  - now we have tungsten contaminating the spectra,
  - 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.

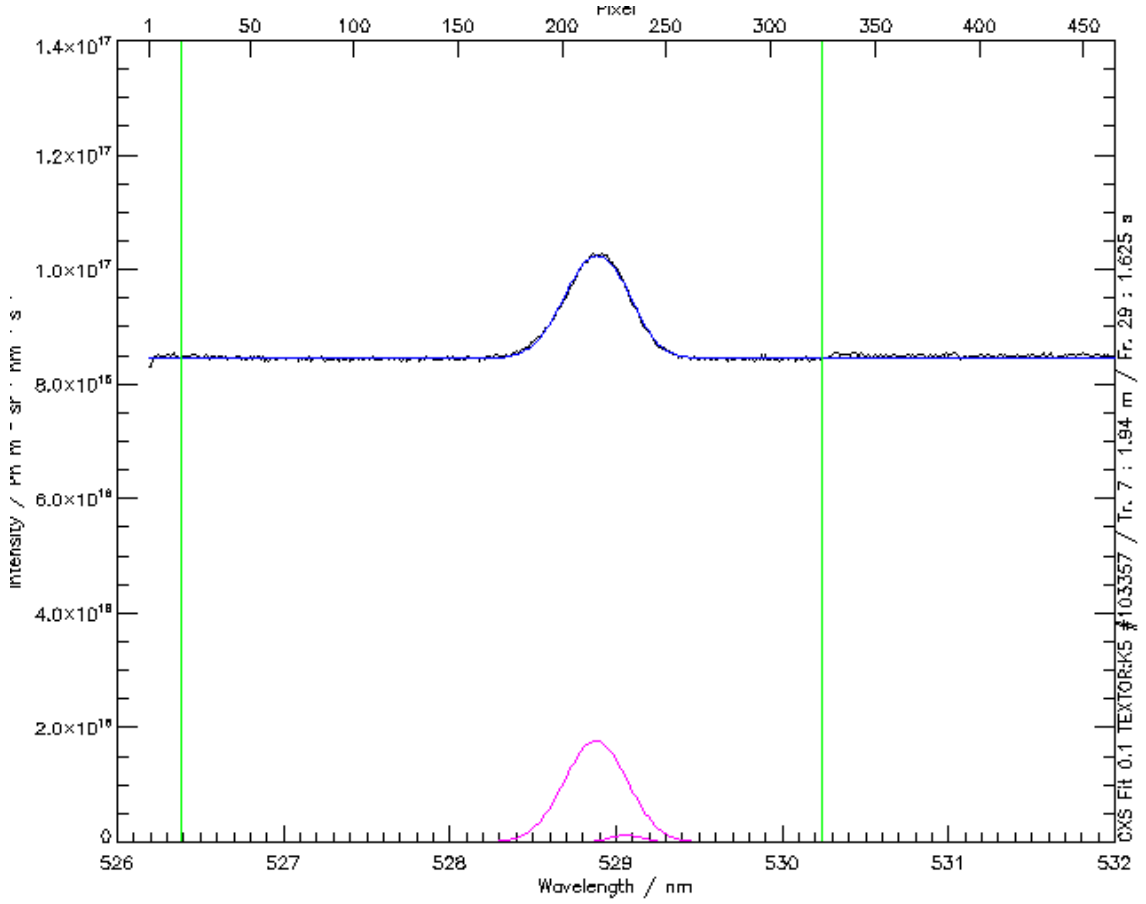
KS4FIT

- Practical implementation of charge exchange fitting at JET was called KS4FIT:
  - originally written for the IBM but ported to Linux,
  - reads spectra (JPFs) and writes results (PPFs),
  - had a TSO-style interface,
  - now replaced by CXSFIT.
- 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.

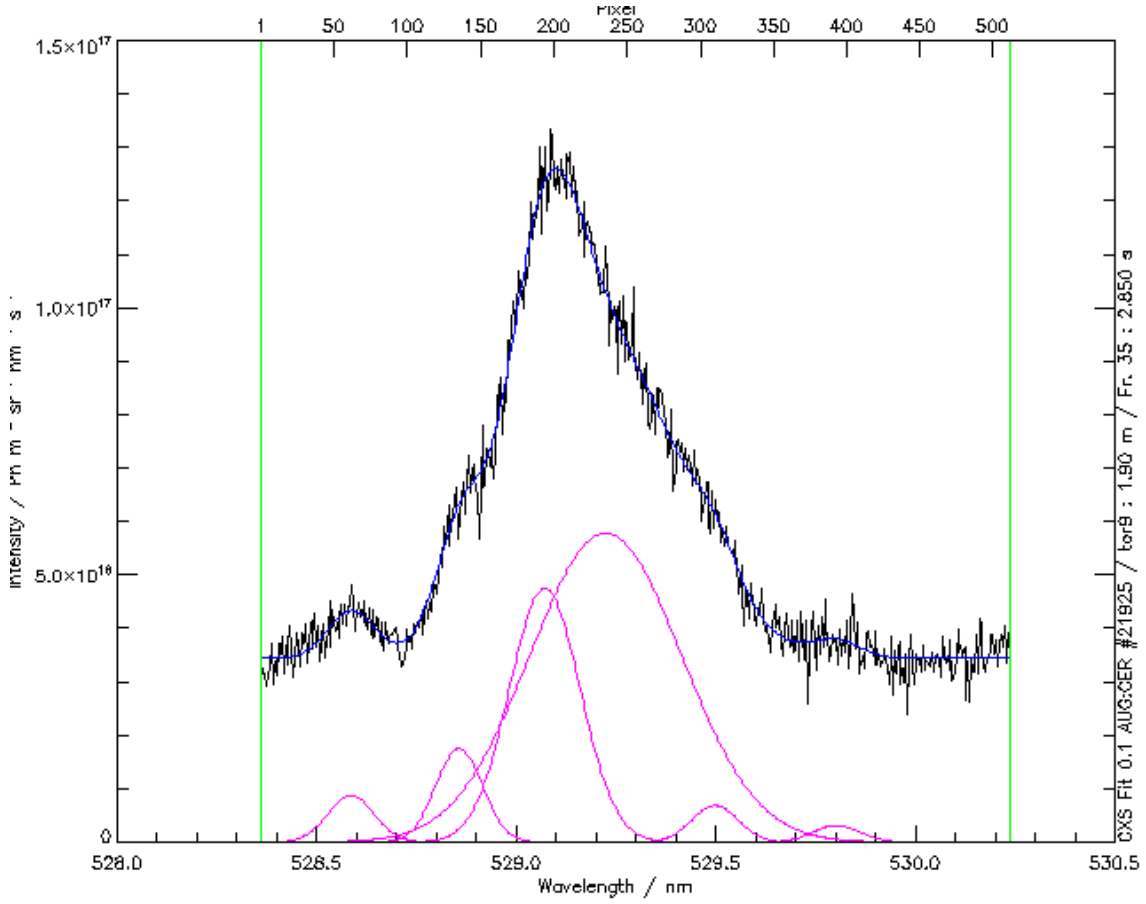
CXSFIT

- 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.
  - We really do have an identical code running on AUG, JET and TEXTOR!

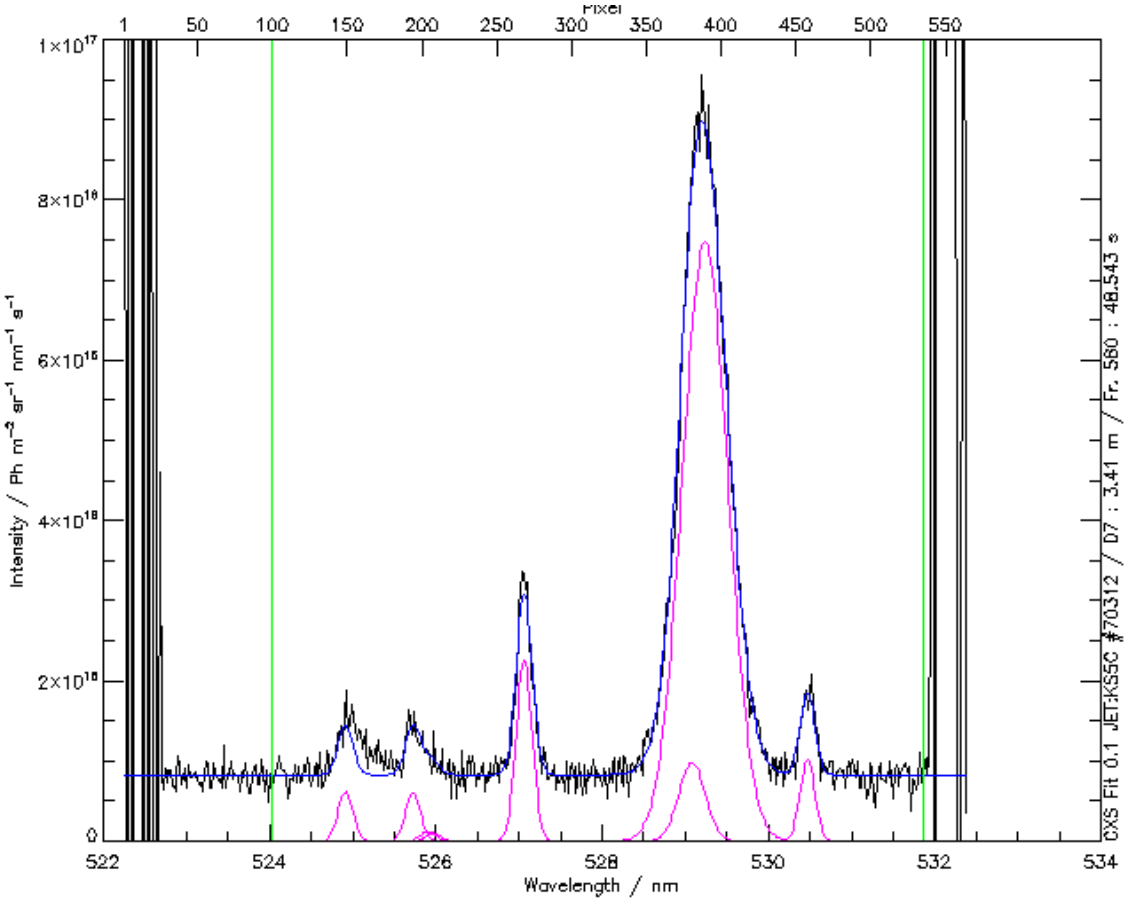
Example spectra — TEXTOR



Example spectra — AUG



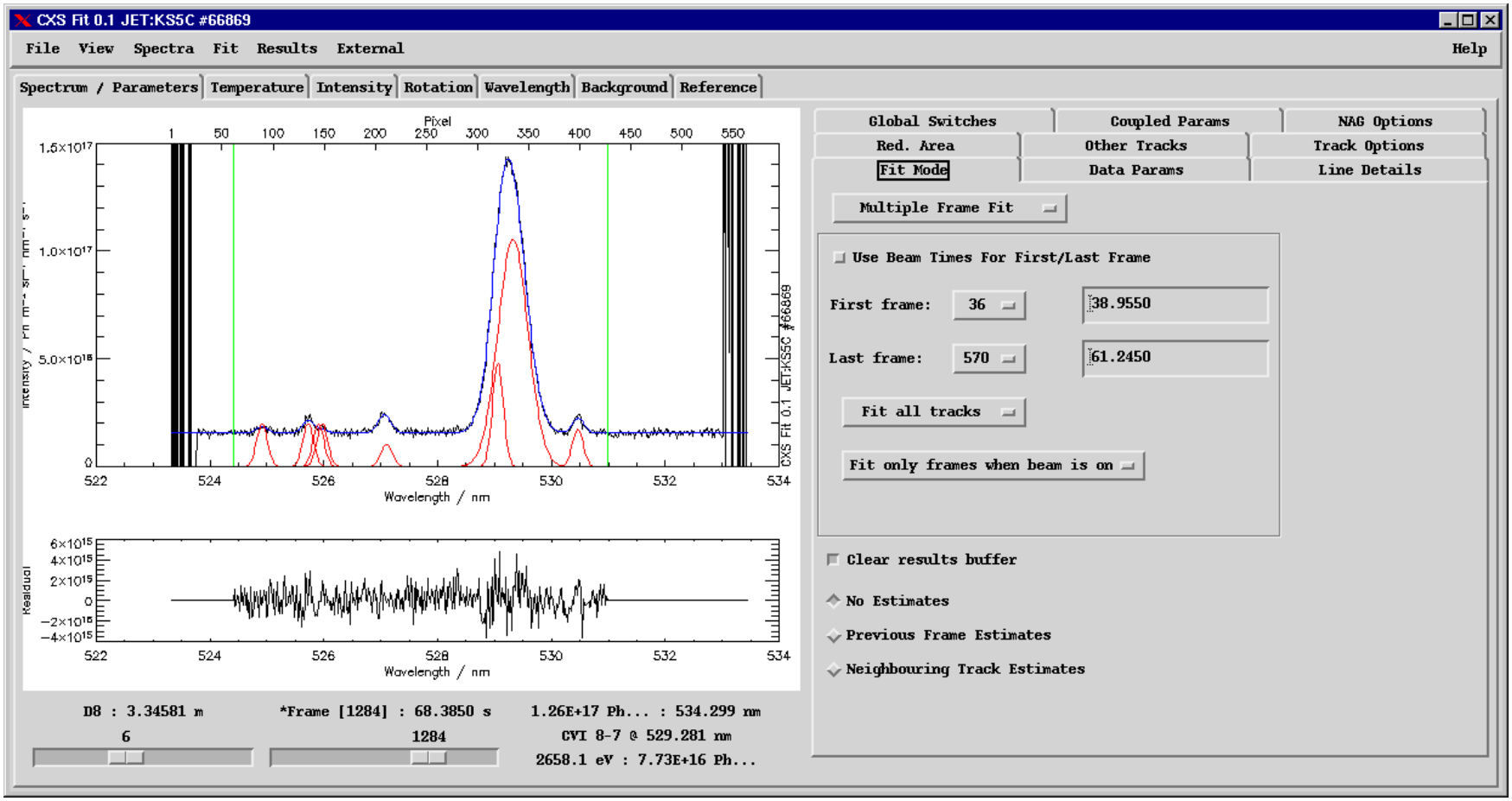
Example spectra — JET



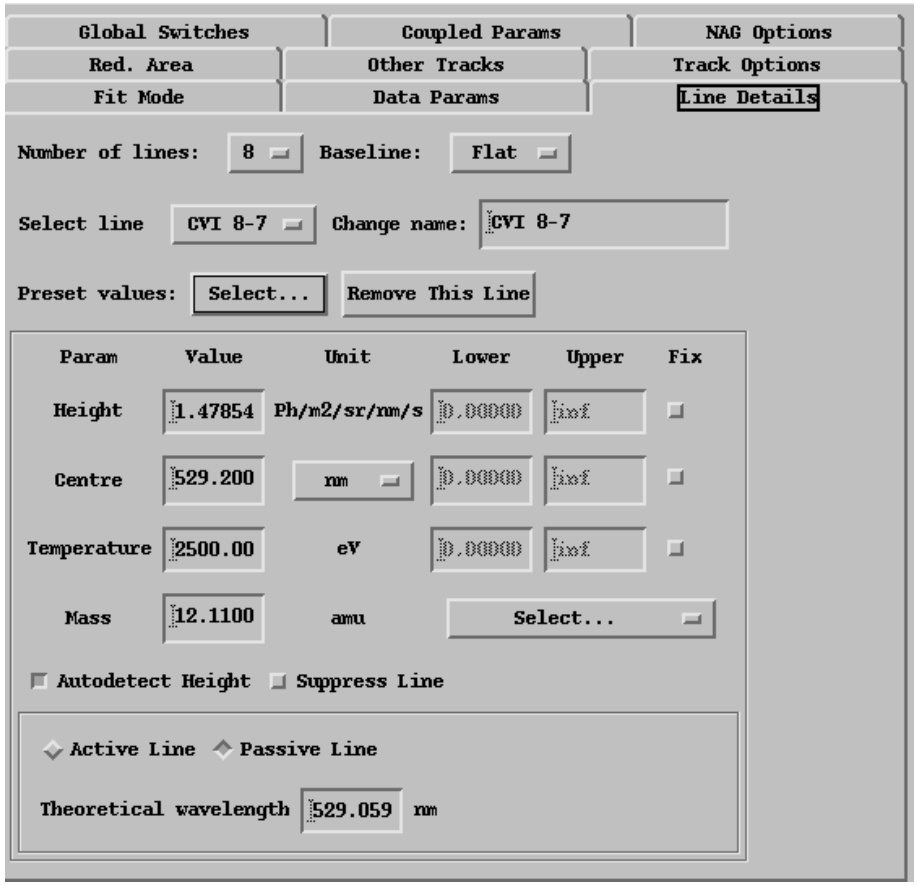
Current usage and deployment

- Used exclusively on all JET core spectroscopy systems (KS5A-E),
  - Reads from JPFs and takes geometry etc. from central location.
  - Writes directly to PPFs.
- Used at AUG on CER and CHR core systems plus CHZ edge system,
  - Reads level-0 shotfiles (spectra) and geometry etc. from central location.
  - Writes directly to level-1 shotfiles compatible with cview.
- Replaced InSPECtor at TEXTOR on all Kameras (1-6).
  - Reads directly from TWU (both spectra and all other information).
  - Writes directly to TEXTOR TPD system.
- First usage at CEA Cadarache last month.

The Graphical Interface

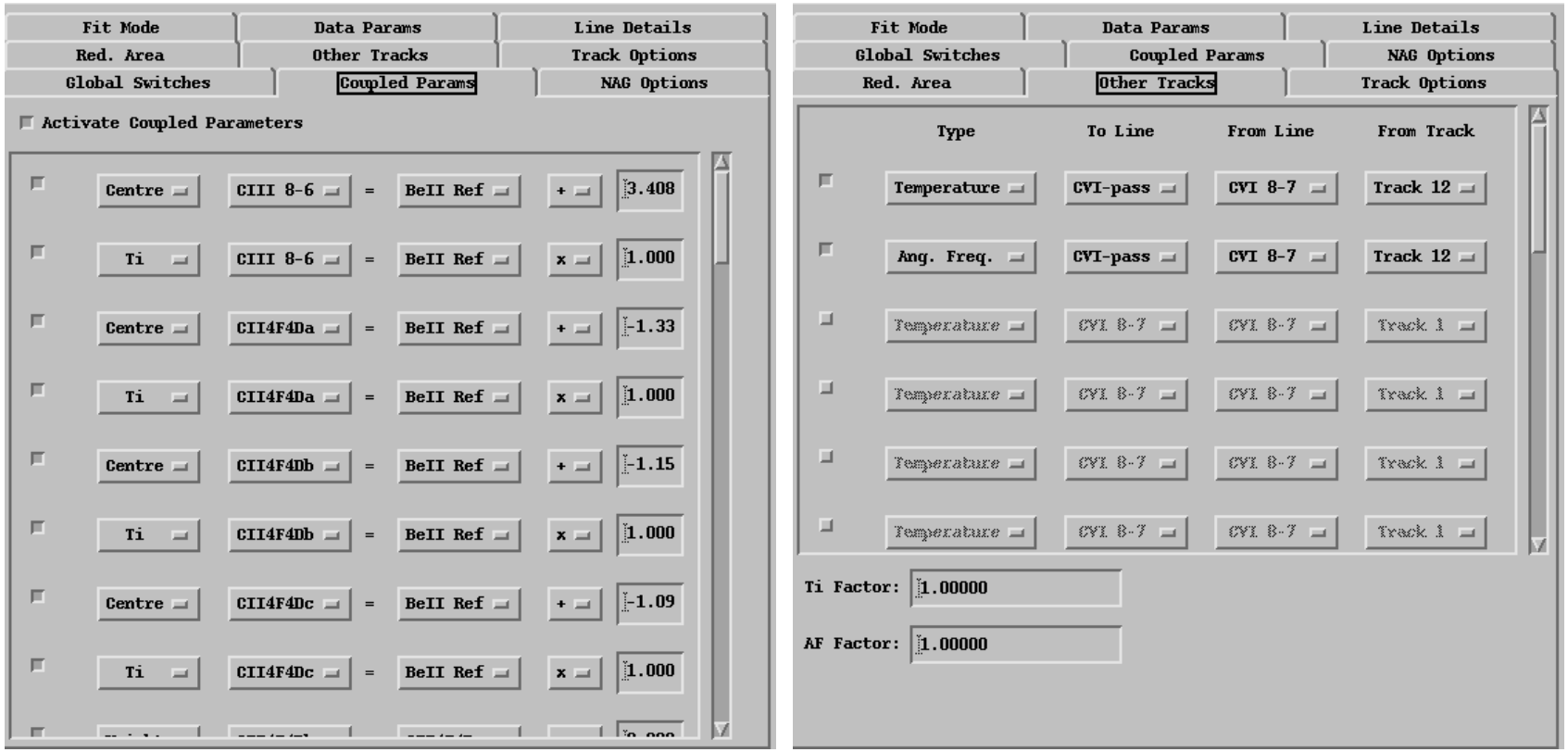


Feature Highlights: In-built line database



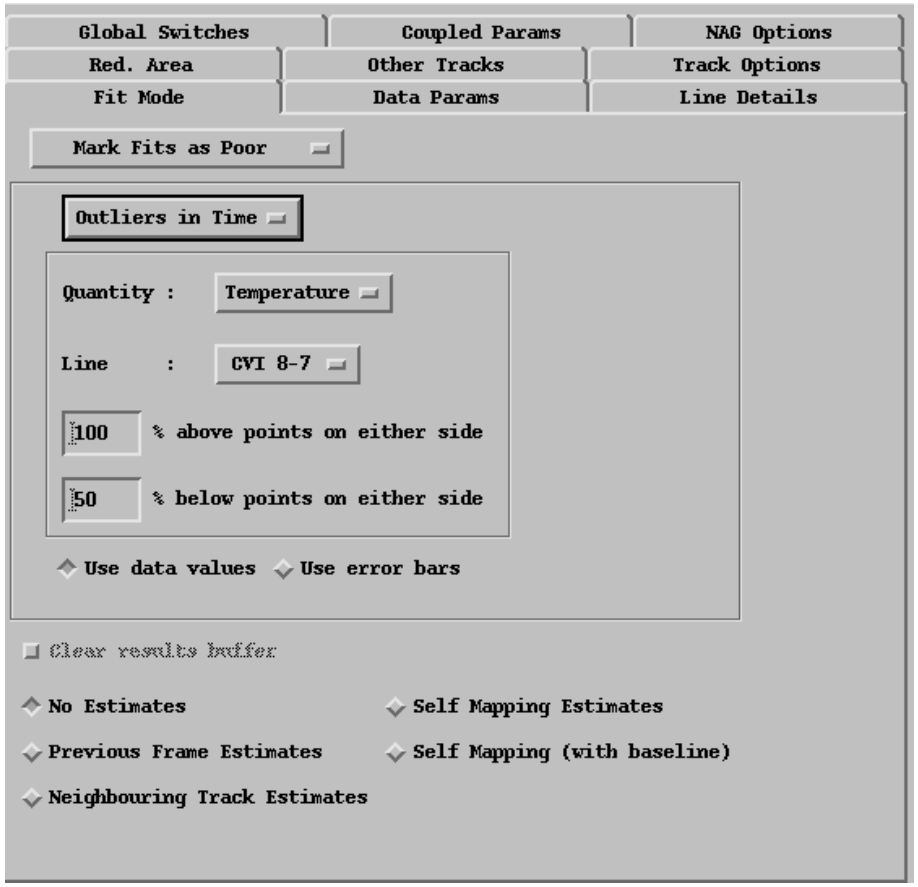
- Very high quality observed wavelengths included in program.
- Gives best quality rotations and on-the-fly calibration.
- If wanted, can be selected automatically based on line position.
- Can be overridden by the user if necessary.

Feature Highlights: Complex coupling



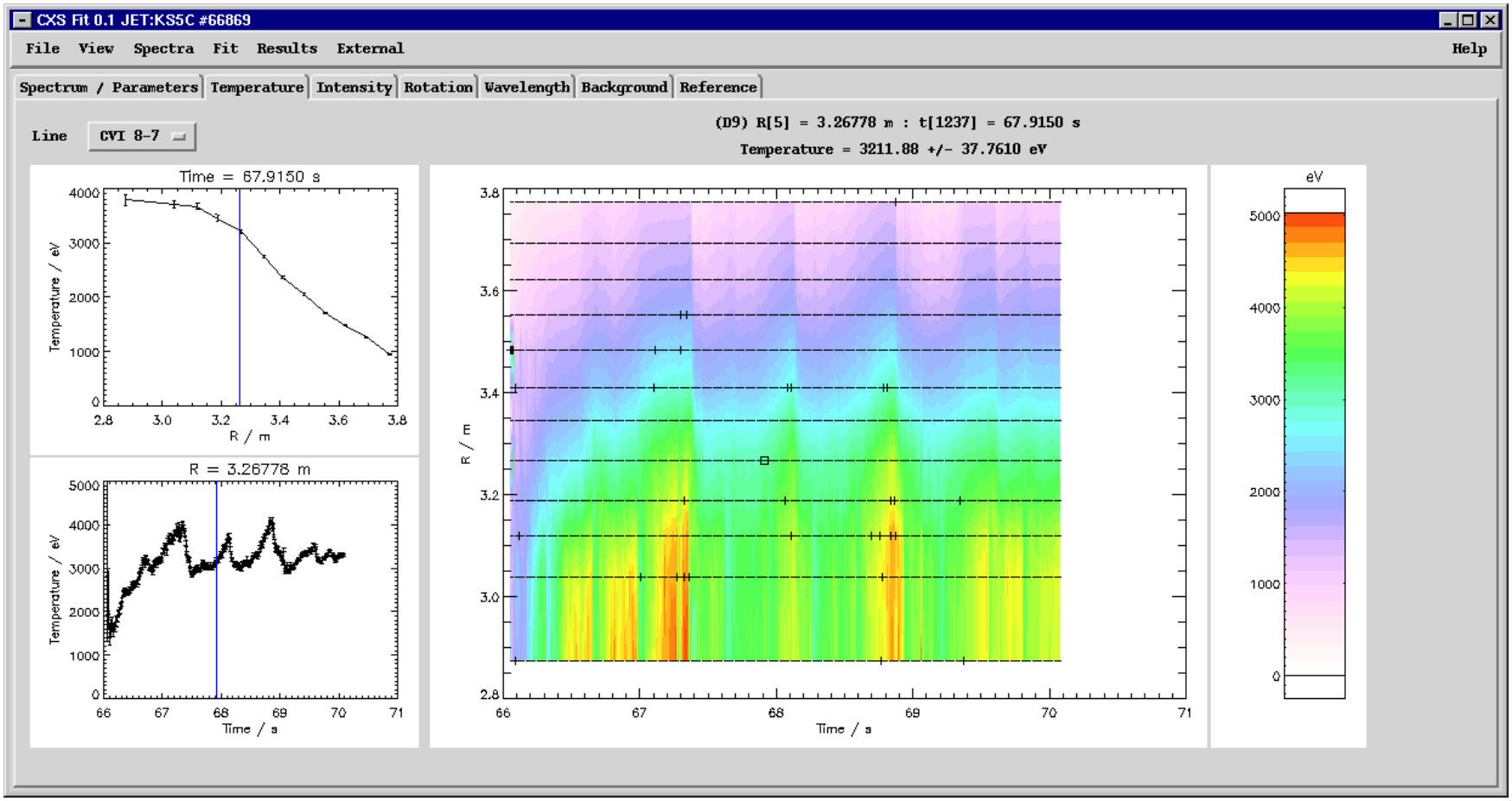
Can couple inside a spectrum or across tracks.

Highlights: detecting poor data



- Can do analysis of fit results to check quality/sanity
- Outliers in time or non-physical T_i profiles
- Necessary on systems with high temporal resolution.
- Can be used as part of a more complex analysis.

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 temporal regions 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 k5_103357.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.
- Testing at TEXTOR gave very similar results to InSPECtor implementation (expected that the CXSFIT results are better due to the treatment of errors).
- The output of CXSFIT at JET can be processed by the current (IBM-esque) version of CHEAP. Similarly for IDL-CHEAP at AUG and Matlab-CHEAP at TEXTOR.
- Possible CHEAP re-development and rationalisation is an issue.

CXSFIT can be run downstairs, please feel free to have a go. Tutorial sheets are available.

Thank you