**Requirements for New Atomic Spectroscopy Data Analysis Software**

*Initial draft by Gillian Nave, 1st Sept, 2022*

**Background**

Spectra from the FTS and grating spectrometers at NIST and ICL are currently analyzed with a diverse set of programs written in languages from 1970’s FORTRAN to python3. Many of these programs are hand-crafted by the individual researcher to match their needs, but this makes it difficult to get new summer students up and running rapidly, and hampers people from taking over an existing project to complete or extend it. In addition, each researcher has their own conventions for file names, variable names and short-hand spectroscopic notation that can be hard for other researchers to understand. Most of the analyzed data are ASCII files, sometimes with different versions in different directories.

This document outlines the requirements and decisions needed to replace this diverse set of programs with a more homogeneous set of programs and data, with the goal of making it easier for new people to begin a project or take over an existing one. In addition, the new software should encode as much as possible of the existing knowledge of senior researchers, particularly with the evaluation of uncertainties and in line identification techniques.

**Current Data Files and Software**

The following data files are used for the start of the analysis of FTS data. The names will vary between researchers, so these are GN’s (current) conventions. All but the first are made by Xgremlin.

|  |  |
| --- | --- |
| **Filename** | **Description** |
| <filename>.dat,  <filename>.hdr | Original interferograms from FTS, created by alice data acquisition software. .dat files are 32-bit floating point numbers. .hdr files are ASCII. |
| <filename>\_r.dat, .hdr,  <filename>\_c.dat, .hdr,  .badpt and .phase files. | Spectra from FT of original interferograms, transformed by Xgremlin. .dat files are 32-bit floating point numbers, others are ASCII. |
| <filename>.lin | Binary format file of all the lines in a spectrum. Format is described in linelist.h in the ‘tools’ directory in the xgremlin source tree. |
| <filename>.lst | ASCII format linelist, uncalibrated. |
| <filename>.wavcorr | Wavelength calibrated linelist, using ‘wavcorr’ variable in the header file |
| <filename>.intcorr | Intensity calibrated linelist, using response function. Only the column ‘EW’ (integrated intensities) is calibrated, ‘peak’ is left alone. |
| <filename>.wintcorr | Wavelength and intensity calibrated linelist. |
| <filename>.hfslin | Binary file of hyperfine structure fits. |
| <filename>.hfsasc | ASCII file of hyperfine structure fits. |

There could be a lot of sense in replacing these separate files with one hdf file, including the files needed to create them. Possible groups and datasets include:

* response curves
* reference wavelengths and plots
* phase curves, plots and bad point files
* original lamp calibration data
* Intensity calibration spectra taken at same time as spectrum

**Things we do to the data and how they’re done now**

1. Wavelength calibration of each spectrum. A set of reference wavelengths are taken and used to make a wavelength calibration plot, from which a calibration curve (for grating) or wavenumber correction factor (for FTS) is derived. A variety of scripts are used for this, usually different ones for each researcher. FT spectra will usually only have a single wavenumber correction factor derived from the weighted mean and standard deviation of the calibration. Grating spectra use a polynomial – the old FORTRAN program ‘polyfit’ has been used for this in the past.
2. Intensity calibration of a spectrum. Xgremlin is used to derive a response curve using spectra of a radiometric standard lamp taken at the same time as the spectrum of interest. Sometimes heavy filtering of the spectrum is required to get reasonable results. The response curve is read into the Xgremlin ‘r’ array and applied to the ‘EW’ (integrated intensity) column of the linelist, with the ‘peak’ column left unchanged. This makes it possible to use the SNR in the ‘peak’ column to estimate uncertainties.

Line identification and Term analysis:

1. Preparation of merged linelist. This involves taking all the original spectra and constructing a single list of all of the observed lines in the spectrum. Both FTS and grating data may go into this list, with FTS data usually preferred if a spectral line is observed in both – this choice should be based on the estimated uncertainty. If lines are to be taken from several different spectra, weighted mean parameters should be derived. Creation of this list has involved a variety of techniques in the past, including old C programs to calculate the weighted means of the parameters. I’m just looking back at the Mn II analysis and already it’s not clear to me exactly how this was done for this spectrum, at least for the FTS portion. It could be useful to include the intensities of the lines measured from different sources and different source conditions here, as that could help distinguish between different ionization stages.
2. Preliminary match of observed lines to energy level differences and to previous identifications from either another spectrum analysis or from impurity and gas lines. This is currently done using the old ‘strans’ FORTRAN program, with various wrappers written in python, awk, etc. to hide the inner workings. Christian’s TAME program is a great start to replacing this mess.
3. Sorting out the correct line identifications from the spurious coincidences. This is the bulk of the hand-work and relies on information from a variety of sources, including gf-values from atomic structure calculations, the observed – Ritz wavenumber, the line intensity. TAME could be extended to include a lot of this information, which I’ve currently included in the past using hand-written additions to printed lists sorted by upper level. The flexibility to include data from a variety of different sources could be very useful here. Some manual input to this is inevitable, and it is going to be an iterative process as the level values are improved, more identifications are added from new energy levels, and previous mistakes are corrected.
4. Level optimization. Currently done with Sasha’s LOPT program and we probably want to keep that.

HFS structure analysis

1. I’ve used Xgremlin for this in the past but am curious about Milan’s program and how it could be incorporated into the suit of programs that we’re proposing. Xgremlin’s way of dealing with linelists generated through HFS is not well integrated with its usual way of making linelists.

Transition probability (TP) measurements

1. I’ve developed the ‘pybranch’ program for this. It operates on identified lines files generated from the ‘.intcorr’ or ‘.wintcorr’ files using a python script that just matches up the lines to a previous reliable linelist. It also taken the results from atomic structure calculations to calculate a residual branching ratio and to compare to the measured results.

**Decisions to be made**

1. User interface: Qt, wxpython … There seems to be consensus on Qt.
2. Format for files: There seems to be consensus on moving to hdf5, but we need to decide when that is implemented. Should the original spectrum already be in hdf5, or do we implement it later? If we move to hdf5 for the spectrum, do we want to modify Xgremlin to be able to read it or will we just put up with the problem of keeping two different formats for the files, one readable by Xgremlin and another readable by everything else? I think keeping the original NSO6 format for the intereferogram is probably wise until we’re prepared to ditch Xgremlin altogether.
3. Naming scheme for files. This might not matter if we move to hdf5, but then a naming scheme for groups and datasets would probably be needed instead.
4. Where to host? Currently we have programs on various personal Githubs, but we probably want to move to something more central.
5. License? Anything I write is going to be public domain.
6. Naming scheme for energy levels, and how to match them up between the different sources – ASD, atomic structure calculations etc. This is one that always trips me up, and I find I have to use a separate key and lookup table to match levels from different sources. I’m curious to know if anyone has a better way. I also prefer to have a shorter version of the energy level than is used by ASD, but I do like to see some information about the configuration and terms. So some flexibility here would be useful.
7. It’s not clear to me exactly how all of these programs and files will fit together. HFS, term analysis, and TP measurements all use the original spectra from xgremlin, but each could benefit from the analysis from the other programs. Some way of feeding the results of the term analysis back to the original spectra, for example, could be useful, but everything gets meshed together because we have many different spectra going into the analysis and into the TP and HFS programs.
8. Some entry in the original files that the spectrum was used in a particular analysis and wound up in a particular paper? I’m trying to specify in all my papers exactly what spectra were used in the analysis and I think it might be a good idea to have the reverse as well.