Step Fourier Transform Spectroscopy in the Mid Infra-red

Senior Thesis by Jeffrey N Stout -- May 2005
University of Denver Dept. Physics & Astronomy.
Faculty mentor: Prof. Stencel

Chapter One—Discovery:

Mid-IR spectral polarimetry is the study of polarization within a given spectrum. In other words, an analysis of exactly what colors are in a rainbow, and the polarization of a given color of light. This is a particularly useful technique in the mid-IR because the wavelength is the right length to interact via emission or scattering with dust grains.[1] As such, the polarization portion of this technique provides data leading to descriptions of possible grain alignment due to magnetic fields, grain size, grain shape and even dimension. The spectral component provides data indicating the composition of the grains.[2] This in combination with imaging can potentially lead to very specific information about the structure and formation of the stellar envelope.

The theory and techniques involved in spectral polarimetry are not simple, especially considering that the spectral data is being gathered by interferometry. Interferometry is a relative of spectroscopy, but through Fourier analysis, a mathematical technique made easier today by the numerical capability of computers, an interferogram can be transformed into a spectrum. The polarization measurements are accomplished by use of Stokes parameters (intensity measurements) which are then reduced. A basic understanding of the various components of this process is both necessary, and important for exploring the more exciting but rigorous numerical methods necessary for data reduction.

[1] Jurgenson, C. A., Stencel, R. E., Theil, D. S., Klebe, D. I. and Ueta, T. 2002, Astrophysical Journal, 582, L35
[2] Bohren, C. F. & Huffman, D. R. 1983, Absorption and scattering of light by small particles (New York: Wiley)

Spectroscopy is simply an analysis of the individual monochromatic components of light emitted from a source. Not so simple is the process of collecting the spectral data and the analysis of the spectrum once collected. In the case of SIFTIR (Spectro-polarimetric Imaging Fourier Transform Spectrometer for the InfraRed), an instrument being developed by PhD candidate Colby Jurgenson and Dr. Robert Stencel in the University of Denver Department of Physics and Astronomy, this spectral data is collected through a process of interference. Interference simply defined is the interaction between two or more light waves yielding a resultant irradiance that deviates from the sum of the component irradiances. Speaking in terms of sound interference is the beat frequency musicians listen for when they are tuning their instruments. There are two types of interference, wave front and amplitude. Wave front interference occurs when a wavefront is divided into sources, or optical devices produce virtual sources. Amplitude interference occurs when two beams traversing two different path lengths are recombined to produce interference. Interference only takes place between two beams of nearly the same frequency, since beams of dissimilar frequency will create a unstable interference pattern which when averaged over some observing time will go to zero intensity. It is also important, when discussing interference to keep the idea of coherence in mind. Coherence, as Hecht puts it, describes all the waves in a beam marching in step, but not necessarily together[3]—the frequency is not fluctuating over time. In brief, coherence dictates both the stability and resolution of the interferogram. Without coherence one is quickly left with no observable interferogram since the frequency and wavelength of the waves no longer correspond and thus do not interfere observably.

[3] Hect, E. 2002, Optics 4th Edition (Addison Wesley, Reading Mass) 390

The most famous interference experiment was Young's double slit experiment. He used the sun as his source, but to make it coherent and point-like allowed it to shine through a pin hole. This source was then used to illumine two slits which mimicked two coherent secondary sources separated by some distance much less than the distance to the screen, the point of observation. At the screen one then sees an interference pattern. This pattern (a series of bright and dark lines corresponding to path length differences of half wavelengths) results from the distances which the light from each slit traverses to reach the point of observation. This experiment established both the wave nature of light, and was the first example of an interferometer, other than those which exist in nature, for example soap bubbles and oil slicks.

The interferogram that is created under various conditions is of primary interest. Under coherent monochromatic illumination by a source such as a laser, the interference pattern will be a sinusoidal distribution. Depending on the coherence length (for a laser this is very long, several km) the distribution can have many peaks of almost equal amplitude on each side of the central maximum (in spectral terms, a single wavelength). In coherent broad bandwidth or white light conditions there will be a large central peak at path difference zero, and then each component frequency will interfere at maximums in the same way as a laser. Each higher order maxima, therefore, will show a spread of wavelengths. In reality, white light sources such as incandescent bulbs radiate a spatially coherent beam that is too short to observe more than few peaks about the central maximum (in spectral terms, a continuous spectrum). These interferograms can be analyzed using Fourier methods to produce spectral data, at least in principal.

The Michelson interferometer is the basic example of the interferometer used in SIFTIR. As opposed to the diffractive double slit interferometer, the Michelson interferometer is reflective. The path difference and resulting interference is adjusted by means of a movable mirror. A single source is split into two by means of a beam splitter (a partially reflective mirror) and the two arms of the instrument are nearly identical except that one has a compensator which ensures that each beam effectively travels through the glass of the beam splitter the same number of times. The interferogram is created when the two beams are recombined and because of their different path lengths an interference pattern is generated. The mirror can be moved at a constant rate, called scanning, or moved to a particular place and then stopped, called stepping. Stepping used to be nearly impossible because it requires very high precision motion control. SIFTIR is able to both step and scan, as it has a laser reference signal that will constantly determine the actual location of the moving mirror, by counting laser interference fringes from zero path difference.

Once the interferogram has been collected, by moving the mirror specific positions and reading the signal from the detector, the interferogram must be converted into a spectrum. This conversion is done by a Fourier transform. By application of a Fourier Transform (FT), one can transform the intensity versus path difference function of the interferogram into a function of luminance depending on frequency. This FT successfully bridges the gap between interferometry and spectroscopy.

Non-Fourier spectroscopy requires successive examination of every spectral element. Fourier Transform Spectroscopy (FTS) allows every spectral element to be collected at once. There is some amount of time that is required to complete a scan with the interferometer, but at every point were data is taken, information about every spectral element is collected. Also, FTS allows imaging. If spectral data were obtained using diffraction or refraction techniques one would be left with spectral data, but no way to take an image. Michelson himself raised the possibility of FTS, but before the advent of high-speed digital computers taking the FT was next to impossible. We now perform the FT using the computer, the requisite integral is evaluated numerically. The transformation is done under several non-ideal conditions. Two that cannot be avoided are finite maximum path length and discrete sampling. The most important consideration with these two constraints in mind is that the resolution of the resulting spectrogram is determined by a direct relation to the maximum path difference. A longer path difference creates a higher resolution. In SIFTIR this is about 2cm, leading to a resolution of 10,000 at 2 microns.

The theory of the FT is quite straight forward when described verbally. The actual FT and the analysis of it are different matters. In a translated work by Janine Connes, Spectroscopic Studies Using Fourier Transformation, Connes, outlines the ideal and actual constraints on the data collection in the path difference domain and the resulting implications in the frequency domain. These constraints quickly add together to make the FT, data reduction and analysis more difficult than this simple treatment suggests. I take up the mathematical discussion of the Fourier Transform in Chapter Two: Experimental Methods.

So far, however, this discussion has only addressed half of capability of the SIFTIR instrument. SIFTIR, while collecting spectral data also collects data about the polarization of the incoming light. Polarization is of three types, planar, circular and elliptical. Each type refers to the orientation of the electric field vector when one looks at the light wave along the propagation direction. In planar polarized light (P-state) the electric field vector oscillates in a plane that is parallel to the propagation direction. In circular (R and L-states) and elliptically (E-state) polarized light the electric field vector traces a circle or an ellipse respectively. It is obvious thus that the light might be right (R) or left (L) circularly polarized depending upon the direction in which the electric field traces an ellipse or circle.

In the P-state the light can be always be treated as two terms of orthogonal P-states. Perfectly linearly (plane) polarized light is defined as light that is transmitted in only one P-state. In the elliptical case more generally, the light wave can be considered as linear combination of two waves with orthogonal electric fields which oscillate out of phase with one another by some constant. If this phase difference is exactly 90 degrees or an odd multiple of 90 degrees we then again have circularly polarized light. If the phase difference is 180 or 360 degrees we have linearly polarized light. If it is anything else, we obtain some sort of left or right elliptical polarization. (NB right and left are defined by handedness, thumb in direction of propagation the way ones fingers curl gives right with the right hand, and left with the left hand)

Polarization can be determined by use of some media that blocks certain polarizations of light and allows others to be transmitted called a polarizer. The most common polarizer is a Polaroid. A Polaroid allows only one polarization of visible light through. If two Polaroids are stacked with their transmission axis offset by 90 degrees no light will pass through. The first Polaroid polarizes the light in a single P-state which when it is incident on the second Polaroid is blocked as it is polarized perpendicularly to the transmission axis.

In the IR the polarizer of choice is a wire grid. A wire grid is made of many closely spaced conductors (wires) that thus absorb the component of the electric field parallel to the wires and allow the perpendicular component to pass through relatively unaffected. The electric field parallel to the wires is absorbed since the E-field transfers its energy to the electrons in the wire, creating a current which is dissipated as joule heat (electrons colliding with the lattice atoms). The spacing of the wires in a grid must be on the order of the wavelength of the light, so this polarizer becomes practical only at wavelengths larger than the visible spectrum.

In instruments it is sometimes necessary to compensate for the rotation in polarization caused by reflection or transition through optically active (meaning polarization affecting) optical elements. This can be done by use of a retarder. Since polarized light can always be considered as two waves whose E-fields are orthogonal, the light's polarization can be affected if one of those two waves is retarded with reference to the other. In other words a retarder affects the phase difference discussed above, making adjustment to the polarization. Retarders are called wave plates, and are often used to precisely manipulate the polarization of light as it passes through the instrument.

Polarized light is created in a number of ways. It is created when light is emitted from a single dipole emitter, or when light is scattered off of an atom. For the latter case the incoming beam of light induces vibrations in the atom parallel to the E-field, the atom then emits light, which varies in polarization based on the direction it is emitted. 90

degrees perpendicular to the illuminating beam, the light will be linearly polarized, directly in line with the beam, the light will appear unchanged with respect to polarization. Polarized light is also created by reflection. When a ray is incident to a surface the reflected ray, due to interactions with the surface is polarized in the plane of incidence.

The polarization of light is affected by magnetic fields as well. This is called the Faraday affect. The amount of rotation of the polarization has been experimentally determined to be related to the intensity of the magnetic field, the distance through which the light wave passes through the field and the frequency of the light by means of a constant that is also material dependent. The frequency dependence is interesting since this effect falls off rapidly with decreasing frequency. IR is susceptible to the Faraday Effect, as are radio waves.

All this polarization data, when combined with the spectroscopy and imaging capability of SIFTIR allows a more thorough examination of the dust clouds surrounding stars or other clouds illuminated in the IR, than could ever be preformed before. The truly unique aspect of SIFTIR, beyond this combination of polarimetry and spectroscopy, is the imaging capability. Charged Coupled Devices (CCDs) have made astronomical imaging much easier than it used to be. SIFTIR data is collected by a 128x128 mid-IR CCD array. This resolution is very low by modern visible light CCD standards, but the larger wavelength of IR (5 microns compared to 400 nm) and the custom fabrication of these arrays makes a higher resolution array largely unnecessary and prohibitively expensive. The imaging capability of SIFTIR allows it to collect data about the variations in spectrum and polarization throughout an extended body such as a dust cloud. On top of being able to gather information about dust particle size shape and compositions and the polarization of various spectral features, information can also be gathered about the magnetic fields aligning the dust grains, or other macro-level factors in the structure of extended bodies. In short, a more complete picture, quite literally, is captured by SIFTIR's ability to simultaneously perform spectropolarimetry in conjunction with imaging.

Sources used throughout:

Coones, Janine, Spectroscopic Studies using Fourier Transformation (translated and published at China Lake, 1963)

Hect, Eugene, Optics 4th Edition (Addison-Wesley, Reading Massachusetts, 2002).

Chapter Two—Experimental Methods:

Performing mid-IR spectral-polarimetry is difficult in terms of experimental method: a faint source must be picked out of a sky that glows more brightly in the mid-IR than itself; Fourier transform spectrometry is demanding of both the experimental apparatus and the computer that performs the Fourier transfer algorithm used to extract meaningful spectral data from an interferogram; collecting data using a charge-coupled device (CCD) presents its own array of peculiar hardware and software demands; and every optic in the device affects the polarization of the light which we are trying to measure and glows in the mid-IR. Each of these difficulties has been considered in the SIFTIR design choices. To accommodate the unique demands of CCD array readout and control, along with the Fourier transform spectrometer (FTS) control, a National Instruments Field Programmable Gate Array (FPGA) was selected to control the entire instrument. This unique device practically eliminates the electronics normally necessary to interface the FTS and array readout with the computer. The FPGA is programmed using LabVIEW, a graphical development environment that provides for data acquisition and control. What follows is a description of the experimental setup of SIFTIR and a detailed description of the LabVIEW programs necessary to control SIFTIR. This chapter concludes with some remarks about the mathematical theory behind FTS, including the sampling theorem used for SIFTIR. Data reduction methods and calibration discussion are presented in the next chapter.

Experimental Setup:

The experimental setup consists of several components in addition to the detector (TNTCAM2) and the Fourier transform spectrometer. These include computer

interface components, the FPGA, the co-adder, and Sutter filter wheel control box. TNTCAM2 is a cryogenic mid-infrared camera capable of imaging and polarimetric analysis. SIFTIR has TNTCAM2 at its heart, but its mind is the FTS. The FTS facilitates the collection of imaging data which contains spectral information. This combination of spectropolarimetry means that
given pixel in an image captured by SIFTIR includes both polarization and spectral information which must be analyzed. Ideally the polarization of a single spectral feature can be analyzed.

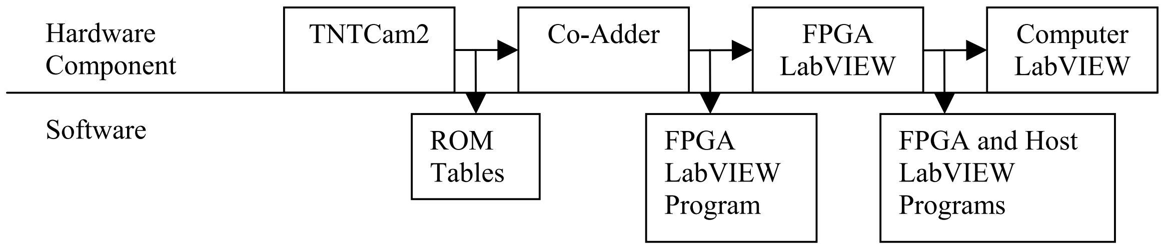
To understand more completely the entire SIFITR instrument package, what follows is a description of the instrument in terms of the light beam path. First, light enters the telescope aperture where is focused into a culminated beam which is sent into the FTS. In the FTS the light is split and sent down the two arms of the Michelson interferometer. The beam is recombined creating the interference necessary for the transform spectrometry. The beam is then sent into TNTCAM2 where it passes through a wire grid and then a rotatable waveplate before it is imaged onto a CCD array.

Next, following the electrical signal, the signal is read from the CCD into the co-adder. The co-adder reads from the CCD constantly and co-adds the requisite number of frames before the co-added data is read out of the co-adder by the FPGA, which in turn passes the data to the computer.

The FTS used in SIFTIR is capable of stepping and scanning. It is very impressive to be able to perform stepped interferometry. The FTS can control the motion of the mirror in discrete steps of 632.8nm and hold its position to +/-20nm. The interferometer is a basic Michelson type with corner cubes as the moving and fixed mirrors to allow for imaging. The moving mirror has a maximum travel of one centimeter, equating to a maximum optical path difference of two centimeters. The precision motion control is achieved through the observation of Helium Neon (HeNe) laser interference fringes. A 45° polarized HeNe laser is shown into the interferometer. The laser is split by the beam splitter and goes through a ¼ waveplate phase rotator in the moving mirror arm, and a ¼ waveplate compensator in the fixed mirror arm. The two lasers traverse their respective paths and are recombined at the beam splitter. When the 45° polarized light goes through the ¼ waveplate phase rotator the outgoing light is circularly polarized (each field component is 90° out of phase). The circularly polarized light is then split by a beam splitter cube into its two different components. The two component beams are monitored by encoders. The mirror can then be driven in precise laser interference fringe steps, and depending on the phase relation between the two component signals, the direction of the mirror displacement is determined (Saptari). The IR beam from a source is passed through the FTS to TNTCAM2.

Within TNTCAM2 the beam is first passed through a wiregrid and then a waveplate which can be rotated to specific orientations. By rotating the waveplate the Stokes parameters discussed in Appendix One on the mathematical description of polarized light can be measured directly and the light polarimetrically analyzed. It is important to note that to get the Stokes parameters an image must be captured for each of 8 waveplate positions.

Figure 2.1: SIFTIR Hardware/Software Overview



LabVIEW Implementation:

LabVIEW is a data-driven graphical development environment. The structure of a LabVIEW program is built around data flow; structures wait for all their inputs to be fed data before executing. LabVIEW is ideal for experiment control and signal processing because of its data-centric nature. Further, with the National Instruments (NI) FPGA that is programmed using the LabVIEW, computer interfacing between instruments, in our application the Fourier Transform Spectrometer and Array Control Electronics, and the computer (where all signal processing save analog to digital conversion takes place) can be accomplished without the electronics normally necessary.

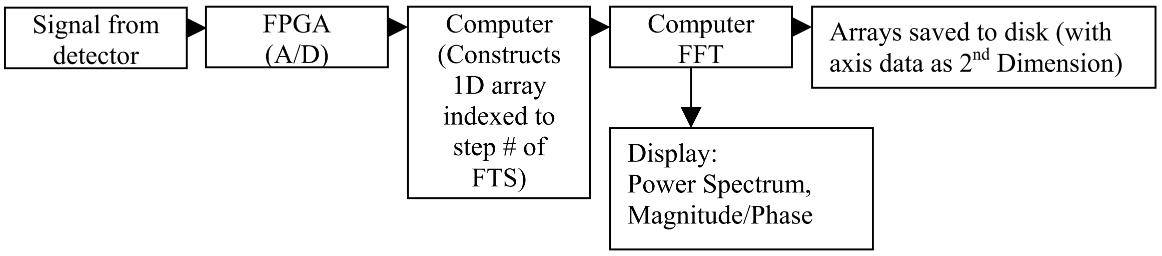
This exposition of the Host and FPGA programs for array control, and host element of spectrometer control, is written assuming an intermediate knowledge of LabVIEW programming and FPGA development.

Theory of Operation:

The LabVIEW programming written for SIFTIR falls into two categories: FPGA programming and host computer programming. The interaction of these two types of programs (Indicators in the FGPA program can be read by the host at any moment, controls in the FPGA program can be written to by the computer at any moment) dictates the structure of each. The SIFTIR control software was designed for use with the first version of the FPGA released by NI. This version of the FPGA does not support direct memory access (DMA) thus the transfer of data from the FPGA to the computer must be accomplished with either polling (computer requested, FPGA served, computer read) transfer or interrupt (FPGA interrupt, computer read, computer acknowledged) transfer. With SIFTIR array data transfer is accomplished via the polling method. This is convenient for at no time do the host and the FPGA rely on each other for operation; they operate independently.

The single pixel detector program set is designed for FTS calibration; it takes no polarimetry data. In the single pixel detector calibration programs, the detector is monitored by the FPGA where the signal is digitized, and then read off by the computer. The FPGA controls the FTS in its entirety. Thus, the computer's interaction with the FPGA is limited to reading the signal data and requesting certain FTS operations (mirror step, scan, toggle etc.). Once data from a scan of the FTS is collected, the Fourier transform (fast Fourier transform, FFT) is performed in LabVIEW, and the original interferogram and FFT saved to the hard disk.

Figure 2.2: Signal Path for Single Pixel Detector Set-up


Signal from detector
FPGA (A/D)
Computer (Constructs 1D array indexed to step # of FTS)
Computer FFT
Arrays saved to disk (with axis data as 2nd Dimension)
Display: Power Spectrum, Magnitude/Phase

The basic operation of the single detector program is as follows:

1. The computer calculates the number of steps in a scan and the mirror displacement per step via the inputted maximum mirror displacement and band pass.

2. The computer is told to move the mirror to the position for the start of the scan and it tells the FPGA program to drive the FTS mirror the desired number of fringe steps. The mirror is allowed to settle for one second.
3. The signal from the detector is monitored by the FPGA and the computer reads the signal as reported by the FPGA. The computer forms an array indexed to step number and detector signal.
4. The computer commands the FPGA to move the mirror to the next position, the mirror is allowed to settle, the signal is read and the process is repeated for a scan. Normally about 3000 steps are needed in a scan.
5. Once the scan is completed the array, an interferogram, is processed by the native LabVIEW FFT algorithm.
6. The outputted power spectrum, and magnitude and phase data is saved to the computer hard drive.

A complete scan is estimated to take about 20 minutes. The data is outputted into a spreadsheet format, so that is can be viewed later in a spreadsheet program such as excel.

The program designed for array data capture (using the TNTCAM2) is much more complicated, but parallels the basic data flow structure above. The main differences are:

- the FPGA program depends on the computer host program to read the 128x array elements from the FPGA and that during this time the host and FPGA programs are essentially devoted to the transfer

- the FPGA program is handling both the FTS control and the TNTCAM2 readout
- the FPGA can hold only one frame's worth of array elements in memory at a time, thus an array read out from the Co-Adder must be transferred to the computer (a process which takes approximately 500-600 milliseconds) before the next frame can be read from the co-adder
- the host program controls the Sutter filter wheel control box

Figure 2.3: Hardware Interfaces and Software or Firmware managing each for TNTCAM2 and FTS

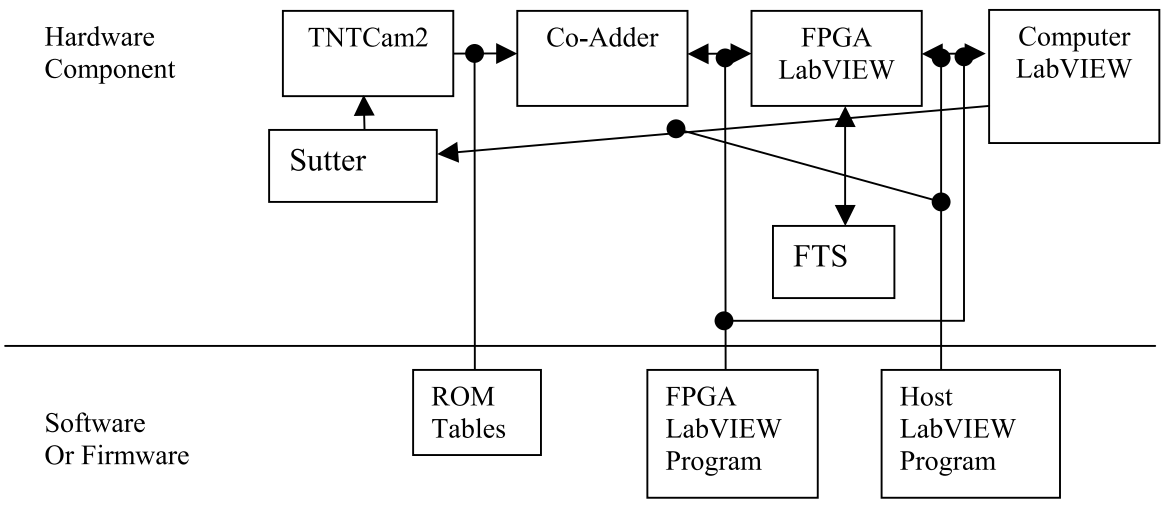
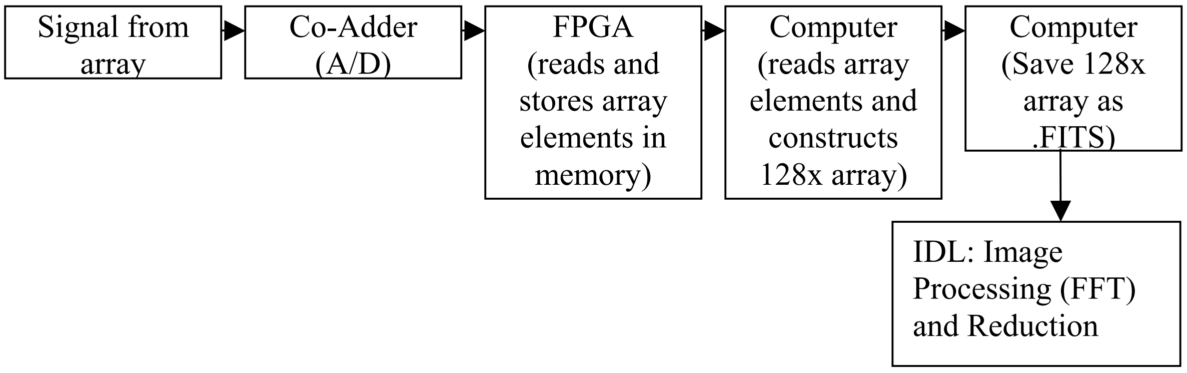


Figure 2.4: The Basic Signal Flow for TNTCAM2 Set-up



Signal from array
Co-Adder (A/D)
FPGA (reads and stores array elements in memory)
Computer (reads array elements and constructs 128x array)
Computer (Save 128x array as .FITS)
IDL: Image Processing (FFT) and Reduction

The basic operation of a spectropolarimetric scan using TNTCAM2 as a detector is as follows:

1. When the LabVIEW host program is started the Co-Adder is programmed through the FPGA instructing TNTCAM2 to take images of a desired integration time and to coadd a certain number of those images.
2. The Sutter filter wheel control box is initiated, and the wave plate moved to position U0 (22.5°).
3. The computer calculates the number of steps in a scan and the mirror displacement per step via the inputted maximum mirror displacement and band pass.
4. The computer is told to move the mirror to the position for the start of the scan and it tells the FPGA program to drive the FTS mirror the desired number of fringe steps to the starting position. The mirror is allowed to settle for one second.
5. The data series is started. A chop foreground/back ground pair of images is read consecutively from the Co-Adder to the FPGA to the computer.
6. The wave plate is moved to position U2 (45°).
7. A chop foreground/back ground pair of images is read consecutively from the Co-Adder to the FPGA to the computer.
8. Steps 6 and 7 are repeated for wave plate position sequence: U0 (0°), U2 (45°), U2, U0, Q1 (22.5°), Q3 (67.5°), Q3, Q1.
9. At the end of this wave plate rotation sequence, after all 16 frames are captured and saved to the hard disk of the computer as .FITS files. The

telescope is nodded and another wave plate rotation, image capture sequence is conducted.

10. This completes all the data collection that must be accomplished at a single FTS mirror position so the mirror is driven to the next step position. The mirror is allowed to settle for one second, and another 32 frames are captured corresponding to the same chop foreground/background, wave plate rotation, nod sequence as before.
11. The scan completes stopping the request number of times.
12. The saved data from the scan is analyzed later.

It is estimated that each FTS mirror position will be held between 20 and 40 seconds to accommodate data collection (these values are approximate because it is unclear what the total duration of the attributing delays are). An average scan of 300 steps means that a scan will take approximately 1.6 hours. This scan time could be greatly reduced by use of the next generation FPGA device supporting DMA for FPGA to computer data transfer. Also limited by the data readout from TNTCAM2 to computer rate:

- chop frequency to approximately 2-1Hz.
- wave plate rotation frequency to approximately .5-.25Hz
- nod period minimum 18 seconds

Program Notes: (Appendix Two provides some screen shots of the programs)

In the single pixel detector setup this program running on the FPGA does two things. First it controls the FTS using a proportional-integral-derivative (PID) motion control program designed by Colby Jurgensen. The FTS control program is one big while loop which on each iteration (it runs at approx. 200 kHz, A/D conversion at 100 kHz) checks the position of the FTS mirror and makes appropriate adjustments to the induction motor. It is a proportional-integral-derivative control based program. Whenever the program is started or stopped the FTS control lines are set to zero so that the FTS is started and stopped in a neutral state, in other words all the analog lines on FPGA connector 0 are set to zero.

Second, the program monitors the input from the MCT detector. This is accomplished via an analogue line into the FPGA coming from a preamp which is between the detector and the FPGA. The signal from the preamp is an amplitude modulated ~300Hz square wave, resulting from a chopper placed between the calibration glow bar and beam into the FTS. The 300Hz frequency is necessary because MCT (Mercury Cadmium Telluride) detectors are normally used in scanning spectrometers where the signal would be on the 100Hz order, but in the FTS step mode we use with SIFTIR the signal must be artificially modulated at a frequency above 100Hz. The FPGA program is synched with the chopper by monitoring a photogate, which comes into the FPGA via DIO line 1 on connector 1. When the line goes high the chopper is in pass mode, low in block mode. Thus the FPGA program waits until the chopper is passing light, then starts monitoring the detector signal waiting for a maximum value. When a maximum is found, this value is written to an indicator to be read by the computer host program. The computer readout is keep in synchronization with the FPGA using an interrupt line.

The host program for the single pixel detector setup is built with a single event structure and an outside while loop which monitors all the FTS FPGA indicators (fringe count etc) at a speed of 2Hz. The event structure is a powerful LabVIEW feature that controls the program based on front panel actions. For example, when the user inputs a ‘step size’ for the FTS and presses the ‘Step’ button the event structure responds to the ‘Step’ button value change by writing the ‘step size’ value to the FPGA and sets the ‘Step’ control in the FPGA program to true, driving the FTS to step accordingly. Each interaction with the front panel is wired appropriately. Most events are very straightforward and need not be discussed in this overview, however, the ‘start scan’ event requires some discussion.

Before the user presses the ‘start scan’ button, a number of parameters must first be entered. These include the ‘mirror displacement,’ the ‘minimum wave number’ that is passed through any filters, or that the detector is sensitive to, the ‘Coadd #’ which is how many readings from the detector is taken at each mirror step and the ‘settling time’ that the mirror needs to make a single step and settle on the next fringe. The first two of these parameters, ‘mirror displacement’ and ‘minimum wave number,’ provide the necessary data from which the LabVIEW program calculates the ‘step size,’ number of steps in one complete scan (#Samples) and the ‘start fringe’ from which the scan should commence. This calculation (described in more detail in the sampling theorem exposition) is performed each time either ‘mirror displacement’ or ‘minimum wave number’ is changed by the user. After these two values are inputted, the user must step the mirror to the ‘start fringe’ using the manual FTS controls. If the user desires to save the data from a scan the ‘file save’ button should be set true and then the ‘start scan’ button pressed.

The ‘start scan’ event structure does the following:

1. The sampling calculation is repeated, and the values sent to the FPGA appropriately.
2. A while loop is initiated which runs once for each mirror step:
    a. The host waits for the interrupt from the FPGA.
    b. The computer acknowledges the interrupt, causing the FPGA program to advance.
    c. The FPGA reads the detector signal each time the chopper opens and co-adds the proper number of frames.
    d. This co-added value is written to an indicator.
    e. The FPGA sends an interrupt to the computer.
    f. The computer reads the co-added value and acknowledges the interrupt, causing the FPGA program to step the mirror on the FTS.
    g. This process repeats for each step in the scan.
3. The host program displays the interferogram in real time on the front panel.
4. Once the scan has been completed the data array is saved for latter data analysis.

If at any point in the above process the scan must be halted, the ‘abort scan’ button can be pressed to do so. The end position of the mirror is displayed by the ‘fringe count’ indicator. Another scan can be started immediately by driving the mirror back to the ‘start fringe’ value and pressing the ‘start scan’ button again. The program should be terminated only through the large ‘STOP’ button on the front panel. This tells the FPGA program halt which sets the FTS control lines to zero so that the FTS is left in a neutral state.

The array control program is significantly more complicated than the single-pixel detector program, but the basic structure is the same. Although every attempt has been made to verify that the array control program is fully functional, it has not been tested with the actual IR-CCD chip, nor have ROM tables (which program the co-adder) been developed. Thus, the array control program, although seemingly fully functional, will need some modification when the FTS and TNTCAM2 are mated and calibrated together. The program notes below are not intended to explain the entire operation of the FPGA and host array control programs. There are ample notes in each program explaining details of operation. Offered here is a general outline of program functionality and some justifications of major decisions made while programming.

The host program starts the FPGA program after calculating the bit-stream necessary to program the co-adder. This bit-stream is calculated in a sub-VI designed by Jurgensen based on the documentation for the co-adder provided by Wallace Instruments (WI). The host sends the bit stream to the FPGA which then programs the co-adder via the Prg Coadd sub-VI. The Prg Coadd sub-VI follows the WI documentation, writing the program using three digital lines. After the co-adder has been programmed the FPGA program waits to read a frame of data from the co-adder. Since the FTS Control loop runs parallel to the Array Control loop, the FTS can be controlled at any moment regardless of the status of the Array Control loop. The host program controls the FTS in the same way that the single pixel detector programs do: an event structure in the host sends commands to the FPGA program whenever certain changes are made on the front panel. These changes to the FTS control loop in the FPGA program control the FTS appropriately.

The major complexity in the array program pair are the steps necessary to capture an array to the computer. The programming to accomplish this was restrained by a number of factors including: the FPGA's limited on-board memory, the FPGA's inability to support DMA, polling transfer being preferable to interrupt transfer since it is faster. Figure 4 offers some idea of the steps necessary for this transfer. It should be noted that this program structure has the most potential for streamlining if new array control electronics are used, the FPGA is upgraded to a newer version with greater on board memory or DMA support, or the co-adder is controlled directly by the computer through serial cards as was done in the previous TNTCAM2 setup.

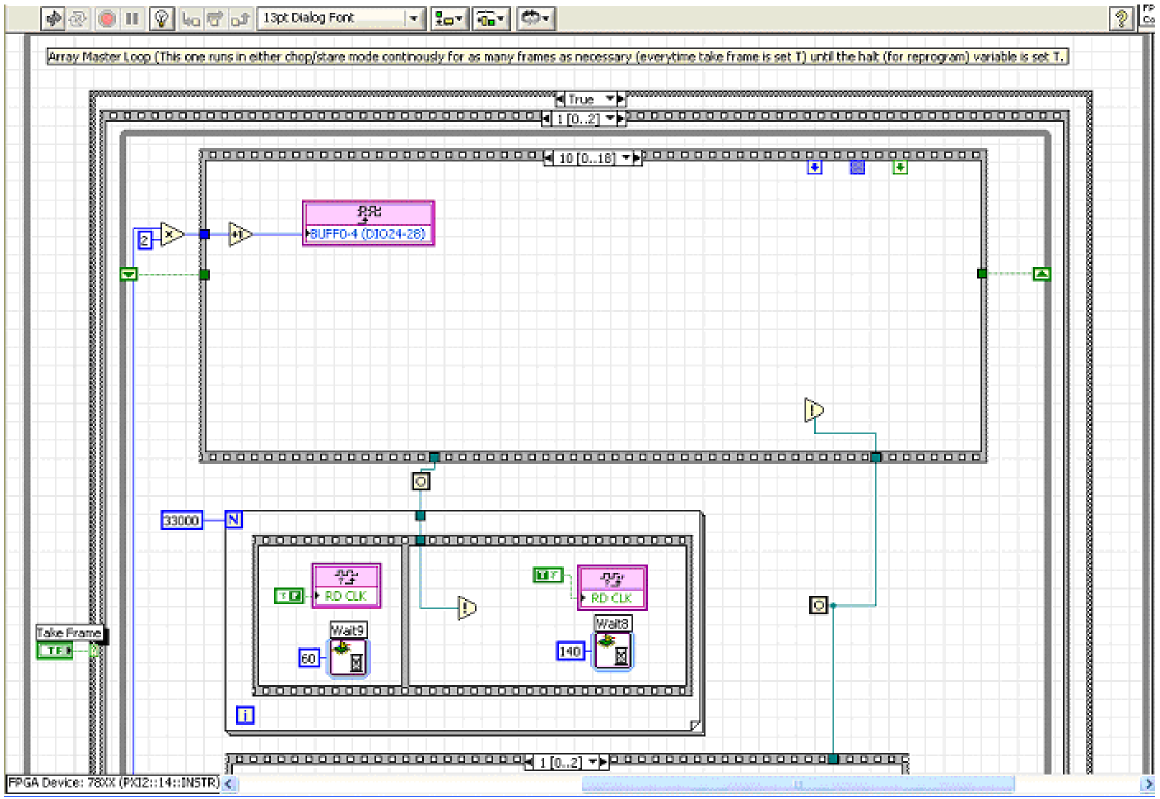
As programmed, to transfer an array from the co-adder the following steps occur:

1. The host commands the FPGA program to take a frame.
2. The FPGA program waits for the co-adder buffer line to change states (indicating which buffer is currently available to be read from) and begins reading the frame from the co-adder.
    a. As outlined in the WI documentation, the sub-buffer address from which the first pixel is read is written to the FPGA (in almost every case 0).
    b. The clear line is set high then low.
    c. The lowest 16 bits of the first pixel are read on the digital lines 0-15.
    d. The clock line is toggled high then low and the high bits of the first pixel are read on the digital lines 0-15.

   e. The high and low 16 bits numbers are combined into a 32-bit unsigned number and saved in the 0 addressed FPGA memory location.
   f. The clock line is toggled, low bits of pixel 2 are read.
   g. The clock line is toggled, high bits of pixel 2 are read.
   h. The two numbers are combined and saved in memory address.
   i. The steps f-h above are repeated to read out half the 128x128 array.
   j. The next consecutive sub-buffer address (almost always 1) is written to the sub-buffer address lines.
   k. The clear line is toggled.
   l. Steps c-i are repeated. The complete array has been transferred to the FPGA memory.
3. The FPGA signals the host that an array is waiting to be read by setting the Ready_to_Read indicator high.
4. The host requests the first pixel to be written to the Dataout_host indicator by setting the Data_Read control on the FPGA to high.
5. The first pixel is read.
6. Steps 4-5 are repeated for every pixel of the 128x128 array (16,384 pixels).
7. The computer checks to make sure it has read every pixel from the FPGA and then either calls for the next frame to be read, or waits for user input.

All the synchronization necessary in the various loops of the FPGA program are controlled by occurrences. This allows, for example, precise timing of the clock line toggle. Since parallel loops on the FPGA take no extra time, it is a much faster program than if all the functionality of the FPGA program were implemented in a single sequence and loop (Figure 5). The data is transferred to the computer using a polling method which takes roughly 300ms to transfer a frame. It is unclear whether DMA would speed this transfer up or not. This speed is the limiting factor in how fast every other process of SIFTIR might run. This 300ms, as stated previously, will lead to a scan time (300 FTS steps) of approximately 1.5 hours. The FPGA program is essentially dumb in that is delivers a frame when asked, and otherwise waits on the host, except in the case where foreground and background chop frames are taken. In this case the FPGA is set to take two frames as fast as it can, starting to read the next frame immediately after transferring the first to the computer. In the chop it is vital the frames read out are one foreground and background image, not two foreground or background images. To avoid this behavior the ROM tables must be programmed to ensure the FPGA and Host can complete two complete array reads in the length one co-adder buffer cycle. If the warning lights on the front panel indicate a buffer toggle error the ROM tables must be adjusted, if the FPGA appears to be behaving properly.

Figure 2.5: Screen Shot illustrating parallel loops synchronized with occurrences in the FPGA array program.


Array Master Loop (This one runs in either chop/stare mode continously for as many frames as necessary (everytime take frame is set T) until the halt (for reprogram) variable is set T.

The FPGA program controls only the co-adder and the FTS. The host controls the Sutter Filter Wheel control. For guidance in programming the ROM tables, the following timing estimates for the steps two retrieve a chop set of images is offered:

1. Host tells FPGA to take frame to background frame coming into the computer—300ms
2. To computer reads second, foreground, frame—800ms
3. To host sends signal to rotate the waveplate and verify waveplate rotation—860ms
4. To computer tells FPGA to take next frame—900ms

In practice, the total time to take a chop image set and rotate the waveplate appears to be on the order of 1.2 seconds, but this is unverified.

Pressing take series on the front panel starts the following sequence of events (N.B. the waveplate must be moved to the U0 position, and the FTS to its starting position at the start of a scan). The synchronization is handled using occurrences:

1. The host commands the FPGA to take a chop pair of images.
2. The waveplate is rotated.
3. The FPGA takes another chop pair.
4. Steps 2-3 are completed for every position of in the waveplate sequence, U0 (0°), U2 (45°), U2, U0, Q1 (22.5°), Q3 (67.5°), Q3, Q1, moving the waveplate as needed.
5. The FTS is driven to the next step.
6. Steps 2-4 are completed at the step.
7. Steps 5-6 are completed until the scan is complete.

During a scan the Sutter control echo should update along with the FPGA fringe number, indicating the progress of the scan.

The array data read from the co-adder to the computer is in a 1-D indexed format that is meaningless until reassembled into a 128x128 array. This array assembly is handled by two sub-VIs documented in Appendix 3.

Mathematical Theory for FTS (FFT and Sampling Theory):

Fourier Transform:

Fourier decomposition allows the frequency distribution, B(σ), to be measured in the form of an interferogram, an intensity distribution, I(x). The basic Fourier decomposition integral is:

$$B(\sigma) := \int_{-\infty}^{\infty} I(x)\cdot e^{-i\cdot 2\cdot\pi\cdot\sigma\cdot x}\,dx$$

the convolution would be:

$$I(x) := \int_{-\infty}^{\infty} B(\sigma)\cdot e^{i\cdot 2\cdot \pi\cdot \sigma\cdot x}\, d\sigma$$

An analysis of the spectra derived from an interferogram, must proceed along the lines of Fourier analysis. At first glance this appears straight forward, take the integral. However, because of physical restrictions such as signal noise, limited sampling range and others that exist even in an ideal instrument, a Fourier analysis is not quite so straight forward. Put simply, "the process of observing and recording a signal does not simply duplicate the incident signal but "folds in" the properties of the instrument" (Davis et. al.), and thus, in order to say something meaningful about the incident signal, we must be able to describe what happens to the signal in the instrument. The following is an outline of some of the aspects of developing this instrument function.

Due to this "folding" the observed signal is described as a convolution integral where h(x) is the recorded signal, f(u) the instrument function and g(x-u) the incident signal convolved would be:

$$h(x) := \int_{-\infty}^{\infty} f(u)\cdot g(x-u)\, du$$

This is an accurate portrayal of the recorded signal, as long as the instrument function is not dependent on the independent variable x.

On top of this instrument response is the unavoidable necessity to measure the interferogram at discrete points. This can be modeled mathematically by multiplying the instrument response function (this might include the incident signal) and the Dirac comb function III(x). Thus the function is sampled at regular intervals. It should be noted that the transform of the Dirac comb function is another comb function with a different spacing between deltas.

$$\mathrm{III}(x)\, f(x) := \sum_{n=-\infty}^{\infty} f(n)\cdot\delta(x-n)$$

It is appropriate to say a few words about the practical implications of this necessity to sample discretely. It is important, according to the Nyquist-Shannon sampling theorem, that the sampling frequency be at least twice the highest frequency present in the analog signal. In SIFTIR's case, we apply a fairly narrow bandpass filter to the incoming signal, and then sample at the appropriate frequency. It is critical that under sampling is avoided, since then high frequencies are aliased, and appear in the spectrum as lower frequencies. Another problem might occur if the comb spacing in the interferogram space is not small enough. Then spectral features cannot be resolved adequately. Needless to under sampling makes the data difficult to use meaningfully.

In a real world instrument there are two factors that serve to further complicate matters: finite path length in the interferometer and finite optical size. This has the effect of turning a Dirac delta in the spectral domain into a Dirac delta convolved with a sinc function and a rectangular function. The result is a Dirac delta that is far from the ideal. First an illustration of the effects of finite path length:

Consider the interferogram of perfect monochromatic light over infinite path length:

$$I(x) := \cos\left(2\pi\cdot\sigma_0\cdot x\right)$$

where σ is the frequency of the light then the transform is indeed a perfect delta function:

$$B(\sigma) := \delta(\sigma - \sigma_0)$$

However, as has been established, this is impractical. Mathematically the finite path length of the interferometer of, say, -L<x<L can be represented as a rectangular function multiplied by the infinite interferogram:

$$I_{obs}(x) := \cos(2\pi \cdot \sigma_0 \cdot x) \cdot \Pi\left(\frac{x}{2L}\right)$$

(where Π is the rectangular function)

Thus in the spectral domain, we have the Dirac delta convolved with a sinc function.

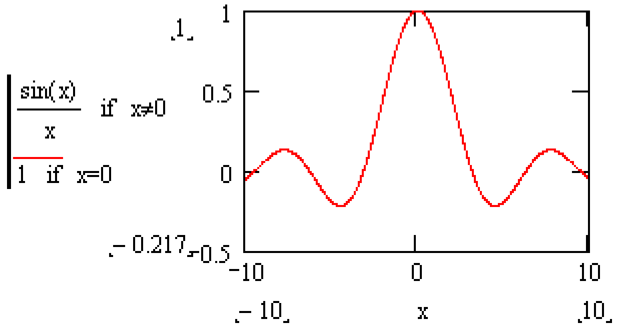
$$B_{obs}(\sigma) := B(\sigma) * \cdot 2L \cdot \mathrm{sinc}(2L\sigma)$$

(* represents convolution)

The resulting profile of the ideal delta function is a line of finite width (and thus height), where the width at half max is:

$$FWHM := 1.207 \cdot \left(\frac{1}{2L}\right)$$

The sinc function produces a spectrum that includes negative amplitudes in the sidelobes, which are non-physical. The side lobes are the symptoms of ringing. Ringing is exacerbated by spectral lines that are narrower than a resolution element (the min width of a spectral element that can be resolved) of the instrument.

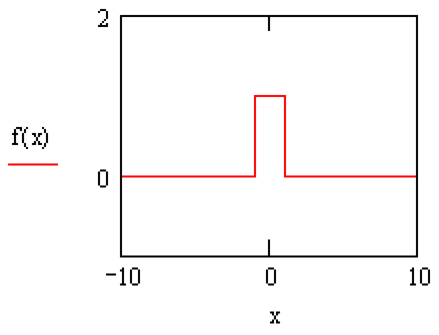


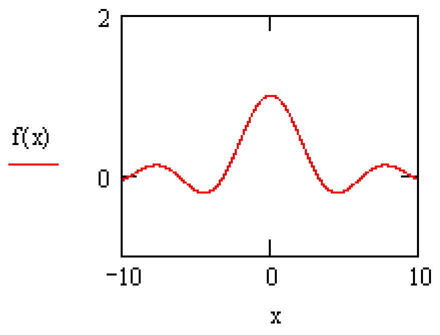
The above exemplifies the process of developing an instrument function. In short, the ideal spectrum from a given source is known, the spectrum after decomposition does not match, therefore adjust spectrum to match. This process is summarized:

$$B_{obs}(\sigma) := B(\sigma)\cdot{*}\cdot O(\sigma)$$

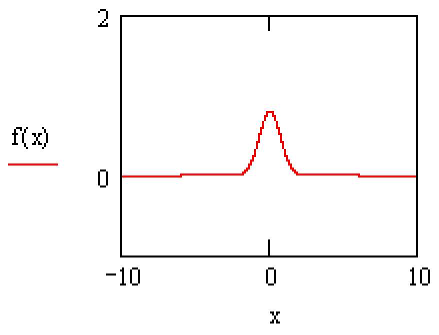
Where the observed spectrum is the convolution of the incident signal and the instrument lineshape function, O(σ).

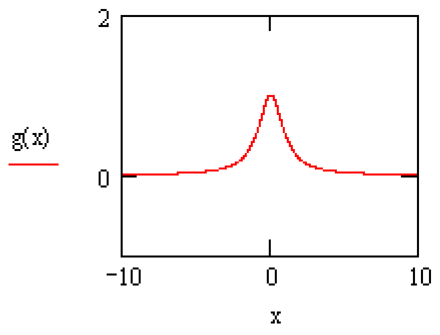
So that a basic mental image of Fourier transforms might be gained, some Fourier transform pairs are listed below, please note that this transforms work in either direction:



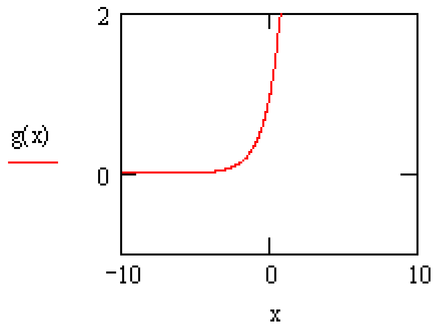


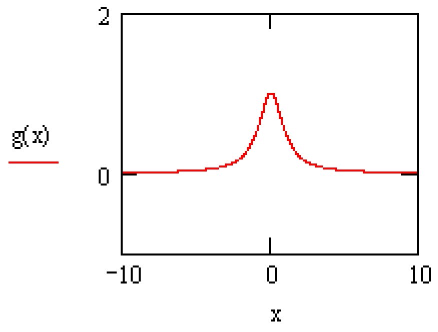
Box transforms to Sinc



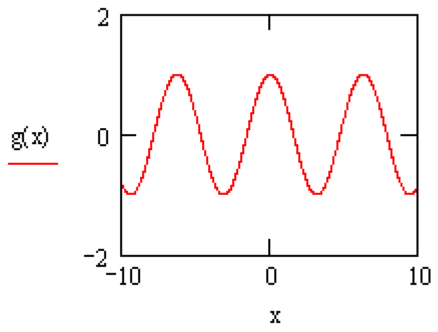


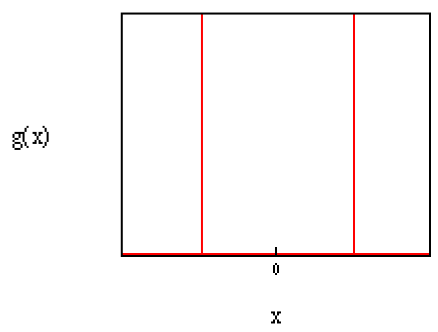
Gaussian transforms to Gaussian



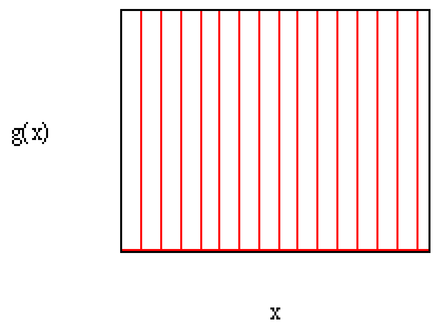


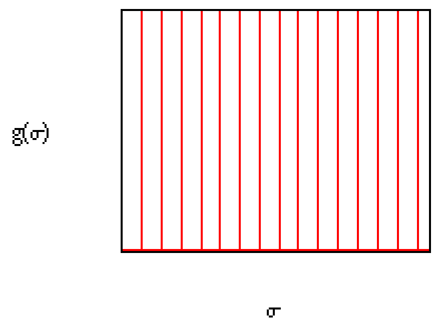
Exponential transforms to Lorentzian





Cosine transforms to Dirac Pair





Dirac Comb transforms to Dirac Comb (1/x spacing)

White noise transforms to White noise

Sources:

Coones, Janine, <u>Spectroscopic Studies using Fourier Transformation</u> (translated and published at China Lake, 1963)

Davis, S. P. et al., Fourier transform spectrometry (Academic Press, San Diego, 2001)

Sampling Theorem:

In order to minimize the number of data points necessary to obtain a spectrum with sufficient resolution and frequency range some a sampling theorem has to be developed. The two elements of the sampling theorem are that: one, the Nyquist condition that samples must be taken more often than twice the bandpass, and two, the scan needs to be long enough (long enough path difference) to obtain the adequate resolution. The key to understanding the sampling theorem is that the data from the FTS is essentially the original interferogram multiplied by a dirac comb. In the spectral space then, the spectrum is the original spectrum convolved with the Dirac comb. If the interferogram is sampled every HeNe wavelength (632.8×10^{-7} cm) the frequency bins in the Fourier space are $1/(632.8 \times 10^{-7} * N)$ cm, or $632.8 \times 10^{7}/N$ wavenumbers. In other words, the number of samples in normal space is inversely proportional to the wavenumber resolution of the Fourier space. The following details the sampling theorem used for SIFTIR:

$\Delta\sigma := \sigma_M - \sigma_m$ where σ_M = maximum frequency sampled in wavenumbers (cm^-1)

σ_m = minimum frequency sampled in wavenumbers (cm^-1)

$\Delta\sigma$ is thus the bandpass (cm^-1)

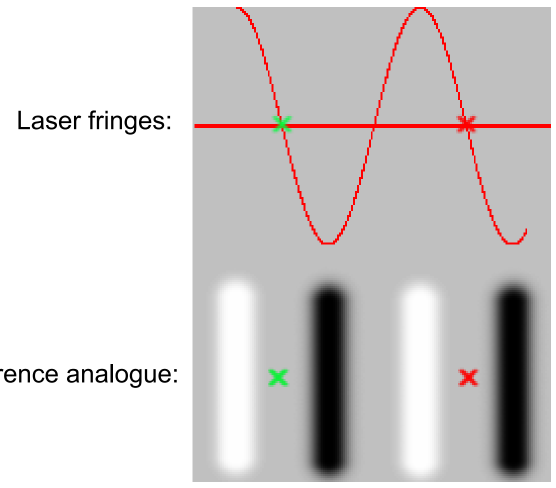
The Nyquist statement is that $\frac{1}{h} := 2 \cdot \Delta\sigma$ where h is the distance between samples in path length (cm)

When the value of $\frac{\sigma_M}{\Delta\sigma} := \chi + f$ some integer, χ, plus some remainder, f, the modified bandpass for the sake of this sampling theorem is: $\Delta\sigma' := \frac{\sigma_M}{\chi}$ and $h' := \frac{1}{2\Delta\sigma'}$

If L is the total path length for some resolution $\frac{1}{L}$ then the total number of samples taken througl

some pathlength is: $\frac{L}{h}$

In the case of SIFTIR, however, steps can only be made in two laser zero crossings (a fringe), or in terms of path length 632.8nm, discrete steps:



From the green starting x to the red stopping x the pathlength has changed by one HeNe wavelength (one constructive, one destructive interference cycle), and the mirror has moved half of a wavelength.

Thus, if one divides: $\frac{h}{632.8 \, 10^{-7}}$ where 632.8x10^-7 is HeNeλ in cm, and then rounds down to the nearest integer to get: N_f one gets the number of fringes the mirror may step and still maintain the Nyquist theorem. Taking: $\frac{L}{632.8 \, 10^{-7} \cdot N_f}$ then gives the number of samples, of Nf fringe steps one takes in a scan. These can be displaced to either side ZPD, as long as ZPD is a part of the sample.

Chapter Three—Conclusion and Discussion:

Data Reduction:

When data is collected by the FTS it is an interferogram. The spectrum must be constructed from this data using Fourier analysis. It was decided to use Mathcad to compute the data reduction algorithm discussed below. The bulk of the programming in Mathcad was completed by Jurgenson, with some assistance from me, the complete data reduction Mathcad worksheet is provided in Appendix Four. There are three main parts to this data reduction scheme data preparation, apodization and phase correction. The first two steps involve reductions performed on the raw data from the FTS. The third step is performed after the Fourier transform.

Mathcad has several a Fourier transform algorithms. The best one for our use is the fast Fourier transform (FFT). The FFT is 'fast' because it performs significantly fewer calculations ($2N\log_2 N$ versus N^2) than the direct transform[4]. It can only except a data set of 2^n values, however, so the data set must be zero filled to raise the number of data points to some 2^n value. Before zero filling, however, the mean of the data set is subtracted out in order to reduce low frequency distortion.[5] This mean subtraction has the practical effect of moving translating the interferogram so that the tails become roughly zero.

The next step is apodization. According to Davis this literally means, "cutting the feet off."[6] If it were possible to collect an interferogram of infinite path difference there would be no need to perform apodization. However, since the data can only be

[4] Brault, J. W. and White, O. R. 1971, Astron. & Astrophys., "The Analysis and Restoration of Astronomical Data via the Fast Fourier Transform", 13, 176
[5] Davis, S. P. et al., Fourier transform spectrometry (Academic Press, San Diego, 2001) 98
[6] ibid. 93

collected in a finite range the resulting spectrum will have ringing. This ringing results from the virtual box function multiplied with the interferogram. A box function produces ringing (a sinc function convolved with the spectrum[7]) because of its sharp vertical edges. In order to decrease this ringing a smoother function can be multiplied by the interferogram to decrease this ringing. On a practical note, some self-apodization occurs simply by taking an interferogram of a resolution (1/MOPD) high enough that the interferogram goes to zero on each side of the central interference pattern. As a result it might not be necessary to perform any apodization on our collected interferograms, which in the later versions of the data reduction algorithm is the case.[8] To reduce noise, however, a Gaussian apodization may be performed. As seen in the previous chapter, a Gaussian function in the retardation domain transforms into a Gaussian. The adverse effects on the spectrum are therefore minimal although some line broadening[9] occurs.

Next, phase correction is performed. Phase correction in simple terms means combining the real and imaginary parts of the Fourier transform in order to get back the true spectrum. This combination, however, is not simply the modulus of the complex FFT if the spectrum is not phase corrected as this would introduce a significant amount of noise into the spectrum.[10] Phase correction compensates for asymmetries in the interferogram resulting from instrumentation effects.[11] The phase correction is performed in the retardation domain. Moving the ZPD central bright to one side of the interferogram essentially eliminates the phase error. To accomplish this, the interferogram is split at its maximum value corresponding to ZPD. Both the right hand

[7] ibid. 93
[8] ibid. 96
[9] ibid. 96
[10] Ibid. 102.
[11] Ibid. 101

side (RHS) and the left hand side (LHS) side are flipped so that the interferogram has phase zero. The RHS and LHS are then joined back together (by their tails) with a string of zeros inserted in between to bring the total samples up to the nearest 2^n level.

The FFT is then performed and the resulting magnitude plotted as the spectrum. The phase is calculated by taking the inverse tangent of the ratio of the imaginary to the real part of the FFT. Where the phase varies rapidly the interferogram should display only noise. If the phase varies slowly with wave number and is close to zero this indicates a low noise and well phase corrected spectrum.

Calibration Results:

In progress

The Big Picture:

In progress

Next Steps:

In progress

Appendix One—Polarization Mathematical Discussion

The Polarization Ellipse- A description of polarized light.
The transverse components of an optical field are:

$$E_x(z,t) := E_{0x}\cdot\cos\left(\tau + \delta_x\right)$$
$$E_y(z,t) := E_{0y}\cdot\cos\left(\tau + \delta_y\right)$$

With a trigonometric identity and some algebra:

$$\frac{E_x}{E_{0x}} := \cos(\tau)\cdot\cos\left(\delta_x\right) - \sin(\tau)\cdot\sin\left(\delta_x\right)$$
$$\frac{E_y}{E_{0y}} := \cos(\tau)\cdot\cos\left(\delta_y\right) - \sin(\tau)\cdot\sin\left(\delta_y\right)$$

Or writing the above in the form of an ellipse:

$$\frac{{E_x}^2}{{E_{0x}}^2} + \frac{{E_y}^2}{{E_{0y}}^2} - 2\cdot\frac{E_x}{E_{0x}}\cdot\frac{E_y}{E_{0y}}\cdot\cos(\delta) := \sin(\delta)^2 \quad \text{where} \quad \delta := \delta_x - \delta_y$$

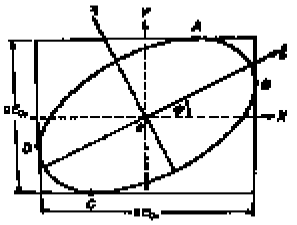
This the the equation for the polarization ellipse, which at any instant in time is the locus of points described by the optical field as it propagates.
A few of the special cases to consider and define:

Linear horizontally polarized light: $E_{0y} := 0$

Linear vertically polarized light: $E_{0x} := 0$

+/-45 degree linear polarized light: $\delta := 0, 45$

Circularly polarized light: $E_{0x} := E_0$ $\quad E_{0y} := E_0$ and $\delta := \frac{\pi}{2}, \frac{3\pi}{2}$



In general the ellipse is rotated by some angle ψ. It can also be characterized by some elliptically angle χ.

By examining the rotated ellipse, and writing down the equations necessary to make the rotation, one derives that:

$$\tan(2\cdot\psi) := \frac{2\cdot E_{0x}\cdot E_{0y}\cdot\cos(\delta)}{{E_{0x}}^2 - {E_{0y}}^2}$$

It is interesting to observe that ψ is zero only when either δ=90 or 270 degrees or Eox or Eoy is zero.

and

$$\sin(2\chi) := \sin(2\alpha)\sin(\delta) \quad \text{where} \quad \tan(\alpha) := \frac{E_{0y}}{E_{0x}}$$

The Stokes Parameters-
There is no way to observe the polarization ellipse directly, but a complete description of polarized light is still possible using the Stokes parameters. These will be developed below. In general two orthogonal plane waves are described as:

$$E_x(t) := E_{0x}(t)\cos\left(\omega t + \delta_x(t)\right)$$

$$E_y(t) := E_{0y}(t)\cos\left(\omega t + \delta_y(t)\right)$$

The polarization ellipse then for monochromatic radiation would be:

$$\frac{E_x(t)^2}{E_{0x}^{\,2}} + \frac{E_y(t)^2}{E_{0y}^{\,2}} - 2\cdot\frac{E_x(t)}{E_{0x}}\cdot\frac{E_y(t)}{E_{0y}}\cdot\cos(\delta) := \sin(\delta)^2$$

For observation take the time average, so we are working with intensity that can be measured.

$$\frac{\left(E_x(t)^2\right)\cdot \text{avg}}{E_{0x}^{\,2}} + \frac{\left(E_y(t)^2\right)\cdot \text{avg}}{E_{0y}^{\,2}} - 2\cdot\frac{\left(E_x(t)\cdot E_y(t)\right)\text{avg}}{E_{0x}\cdot E_{0y}}\cdot\cos(\delta) := \sin(\delta)^2$$

where $\left(E_x(t)\cdot E_y(t)\right)\text{avg} := \lim_{T\to\infty}\int_0^T E_i(t)\,E_j(t)\,dt$

We can find that:

$$\left(E_x(t)^2\right)\cdot \text{avg} := \frac{1}{2}E_{0x}^{\,2}$$

$$\left(E_y(t)^2\right)\cdot \text{avg} := \frac{1}{2}E_{0y}^{\,2}$$

$$\left(E_x(t)\cdot E_y(t)\right)\text{avg} := \frac{1}{2}E_{0x}\cdot E_{0y}\cdot\cos(\delta)$$

Substituting all these into the time avg ellipse equation, and manipulating with a bit of algebra:

$$\left(E_{0x}^{\,2} + E_{0y}^{\,2}\right)^2 - \left(E_{0x}^{\,2} - E_{0y}^{\,2}\right)^2 - \left(2E_{0x}\cdot E_{0y}\cdot\cos(\delta)\right)^2 := \left(2\cdot E_{0x}\cdot E_{0y}\cdot\sin(\delta)\right)^2$$

and there they are the stokes parameters, namely:

$$S_0 := E_{0x}^{\,2} + E_{0y}^{\,2}$$

$$S_1 := E_{0x}^{\,2} - E_{0y}^{\,2}$$

$$S_2 := 2E_{0x}\cdot E_{0y}\cdot\cos(\delta)$$

note: these are all intensities

$$S_3 := 2E_{0x}\cdot E_{0y}\cdot\sin(\delta)$$

$$S_0^{\,2} := S_1^{\,2} + S_2^{\,2} + S_3^{\,2}$$

The Stokes Parameters Continued-
Each of the parameters describes a certain aspect of the polarization:
S0 - total intensity
S1 - the amount of linear (horizontal of vertical) polarization
S2 - the amount of linear (+/-45 degree) polarization
S3 - the amount of r-l circular polarization

In stokes parameters ψ and χ are represented:

$$\tan(2\psi) := \frac{S_2}{S_1} \qquad \sin(2\chi) := \frac{S_3}{S_0}$$

The degree of polarization is defined as:

$$P := \frac{I_{pol}}{I_{tot}} \qquad I_{pol} := \left(S_1{}^2 + S_2{}^2 + S_3{}^2\right)^{\frac{1}{2}} \qquad I_{tot} := S_0$$

Appendix Two—LabVIEW Screen Shots:

In Progress

Appendix Three—Array Builder and Block Builder VI

These programs were designed to manipulate pixel data from a IR CCD array. It was necessary to recombine pixel elements in a specific sequence documented in fig. 1 below.

The Block and Array Builder programs work on the same theory. Using the Replace Array Subset sub vi, the pixel elements can be substituted into a 128x128 array initialized to some value. The array is compiled within a for loop, whose index value is manipulated to index the Replace Array Subset vi. The result including the Block Builder vi is a stacked series of for loops that combine the 2 pixel wide chunks into blocks of pixels, and then blocks into the full 128x128 array.

Array construction diagram:

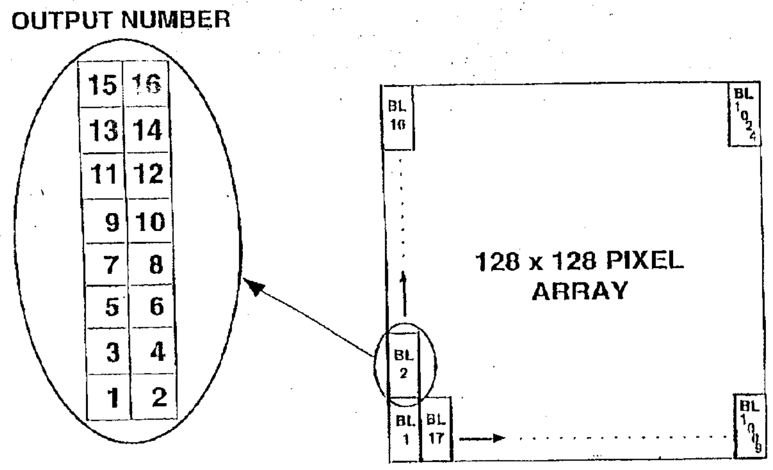


FIGURE 1

Appendix Four—Data Reduction Mathcad Worksheet

When data is collected by the FTS it is an interferogram. The spectrum must be constructed from this data using Fourier analysis. It was decided to use Mathcad to compute the data reduction algorithm presented below. The bulk of the programming in Mathcad was completed by Jurgenson, with some assistance from me. (This is a Mathcad document best viewed in landscape format. It begins on the next page.)

Appendix Five—Common Abbreviations:

CCD — Charge Coupled Device
DMA — Direct Memory Access
FPGA — Field Programmable Gate Array
FT — Fourier Transform
FFT — fast Fourier transform
FTS — Fourier Transform Spectrometer (or in some case FT Spectroscopy)
LHS — left hand side
MCT — Mercury Cadmium Telluride
MOPD — maximum optical path difference
NI — National Instruments
PID — proportional-integral-derivative
RHS — right hand side
SIFTIR — Spectro-polarimetric Imaging Fourier Transform Spectrometer for the InfraRed
TNTCAM2 — Ten and Twenty μm Camera
ZPD — zero path difference