# Temperature and Species Density Measurements Using Fourier Transform Infrared Spectroscopy

A thesis submitted in partial fulfillment of the requirement for the degree of Bachelor of Science with Honors in Physics from the College of William and Mary in Virginia,

by

Jason D. Hoffman

Accepted for

Advisor: Dr. Manos

Advisor: Dr. Danehy

Dr. Bynum

Dr. Hoatson

Williamsburg, Virginia
May 2003

## Abstract

We developed a routine to model absorption spectra measured with a Fourier transform infrared spectrometer based on spectroscopic parameters from the HiTran database and local temperature and pressure information. This model was then validated using an air-filled gas cell at room temperaturewith path length 15 cm. Quantitative measurements of the temperature and pressure within the cell reveal a that pressure measurements are accurate to within 15 Torr (0.3 PSI), while the temperature is systematically over-predicted by 20 K. The standard deviation in the pressure measurements was found to be 9.3 Torr (0.18 PSI) at 770 Torr (14.9 PSI), while for the temperature measurements it was 1.2 K at 298 K.

## Acknowledgements

I would like to thank Dennis Manos for the constant encouragement and for introducing me to ”real” physics. I would also like to thank Paul Danehy for a wonderful opportunity to do research. Without their support none of this would have been possible.

# Contents

# 1 Introduction

## 1.1 Motivation

Testing in wind tunnels, such as those at NASA Langley Research Center (LaRC), is the primary method for evaluating new aerospace vehicle concepts. Model performance is judged by measuring the aerodynamic forces such as lift, drag, and wing deformation acting on or within the model. To fully understand these measurements, knowledge of the temperature, pressure, velocity, and composition of the gas flowing in the tunnel is necessary.

Optical measurement techniques are preferred, especially in supersonic facilities, because they are non-perturbative, unlike probe-based techniques in use today. While many optical measurement options exist, we are examining Fourier transform infrared (FT-IR) spectroscopy because it is a relatively low-cost, easy-to-use, and well understood technique, that uses commerically available hardware. However, we believe that our application to studying temperature and species densities in wind tunnel flows is novel.

In our work we hope to measure both rotational and vibrational temperature, as well as species density, from which pressure may be inferred. The measurements will be path-averaged across the tunnel, though tomography could be used on axisymmetric flows to generate two-dimensional temperature and pressure profiles. Furthermore, FT-IR data are time averaged over several seconds, making this technique unsuitable for the study of unsteady flows. Another important aspect of FT-IR spectroscopy is that only a fraction of molecules are infrared active. These include many species of practical interest ($CO_2$, $H_2O$, etc...) but excludes $O_2$, $H_2O$, Ar, etc....

FT-IR spectroscopy data may also be used to validate or refine the input parameters currently used in Computational Fluid Dynamics (CFD) calculations. At present, these conditions are obtained from probe-based instruments. Furthermore, the data may be used diagnostically to determine if the tunnel is operating at desired conditions. If possible, near real-time analysis would display freestream conditions while the tunnel was operating.

Numerous experimental opportunities for this method have been investigated at LaRC. Combustion facilities such as the 8-ft High Temperature Test Facility and the Direct Connect Supersonic Test Facility produce both carbon dioxide and water vapor while running. The National Transonic Facility has contaminant oils and water vapor, whose concentration could be quantified. The Unitary Plan Wind Tunnel, the 15-inch Mach 6, 20-inch Mach 6, and 31-inch Mach 10 tunnels all run with air, which contains trace quantities of carbon dioxide. Seeding of an external gas is not necessary because FT-IR is sensitive to a wide-range of species already present in the facilities. Additionally, FT-IR could be extended to study the water vapor and carbon dioxide in scramjet engines.

## 1.2 Overview

A brief outline of the experimental setup and data analysis methodology are constructive prior to presenting a theoretical basis for the work. As we show in subsequent sections, sample pressure and temperature are important parameters governing the appearance of an absorption spectrum. However, because these variables (along with others) influence the spectrum in a complex manner they may not be obtained directly from experiment. Rather, an experimental spectrum is compared, via least-squares fitting, to a library of theoretical spectra. The library is generated using our program **specgen** (see §3 and appendix B), which relies on environmental (e.g. temperature and pressure) and experimental (e.g. instrument resolution and absorption path length) conditions designated by the user, in addition to spectroscopic parameters from the HiTran database [1]. The temperature and pressure used to produce the theoretical spectrum that best approximates the experimental spectrum are presumed to be the conditions at which the experiment was conducted.

The work completed to date has centered on developing and validating the **spec-gen** software. While both the development and validation processes will be elaborated later, we provide a brief description of the experimental setup used to validate the algorithm, so that the reader may better conceptualize the experiment, and thereby connect the theoretical development to something concrete.

Broadband radiation is passed through a Michelson interferometer, the output of which passes through the gas sample, downstream of which detection occurs. The gas sample, air for the validation tests, is contained in a custom-made metal cell, measuring one inch in diameter and seven inches in length, with one inch diameter potassium chloride windows on either end. The cell is connected to a vacuum pump that allows us to evacuate the cell to 0.75 Torr for background scans, or to partially evacuate the cell to validate the model from 0.75 Torr to one atmosphere. The ratio of the background and experimental spectra gives the absorption spectrum for the gas in the cell. If this technique were to be extended to studying pressure and temperature in a wind tunnel, the gas cell would be replaced by the free stream flow. Additionally, fiber optics would be necessary to carry the radiation from the interferometer to the sample, and from the sample to the detector.

## 2 Theory

### 2.1 Fourier Transform Spectrometers

The Fourier transform infrared spectrometer consists of two plane mirrors, $M_1$ and $M_2$ in figure 1, oriented perpendicularly and forming a right angle. At the vertex of the right angle is a 50-50 beamsplitter, rotated 45° relative to the mirrors. Incident radiation, $B_0$, from a broadband source, impinges on the beamsplitter such that equally intense beams $B_1$ and $B_2$ propagate to $M_1$ and $M_2$ respectively. The two beams recombine at the beamsplitter, to form two new beams $B_4$ and $B_5$. Beam $B_4$ is directed back at the source and not of interest; $B_5$ is received by the detector. Translation of mirror $M_1$ along the axis of the corresponding arm causes interference in beam $B_5$. This modulation is slow relative to the frequency of the radiation in $B_0$ and is detected as an interferogram, that is a series of peaks and valleys as a function of time (or mirror displacement if the mirror has a constant velocity). The interferogram is written $I_0(t) \simeq I_0(\delta)$, where $t$ is time and $\delta$ is the mirror displacement (usually in centimeters).

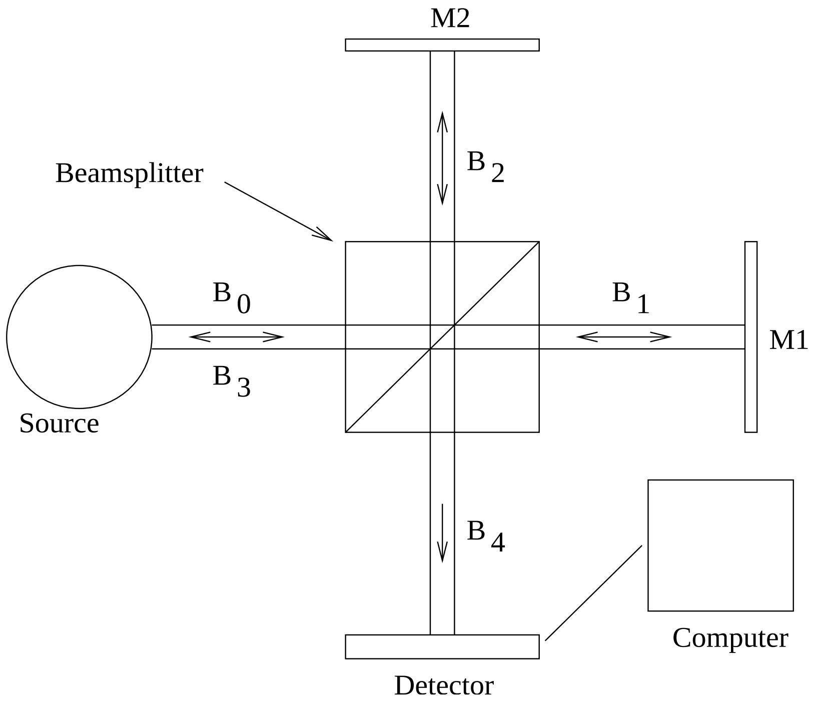

Figure 1: Schematic of the components of a Fourier transform infrared spectrometer.

Considering two interfering waves of intensities $B_a$ and $B_b$, the resulting wave has intensity $I$, such that

$$I = B_a + B_b + 2\sqrt{B_a B_b}\cos d, \tag{1}$$

where $d$ is the phase difference between the two waves. In the case $B_a = B_b = B_\ominus$, the intensity of the interfered wave is,

$$I = 2B_\ominus(1 + \cos d). \tag{2}$$

Returning to the Michelson interferometer depicted in figure 1, and assuming a hypothetical, monochromatic source emitting at wavenumber $\omega_0$, which corresponds to wavelength $\lambda_0$, the phase difference between $B_1(\omega_0)$ and $B_2(\omega_0)$ when they recombine is,

$$d = \frac{4\pi\delta}{\lambda_0} = 4\pi\delta\omega_0. \tag{3}$$

Furthermore, if the beamsplitter divides $B_0(\omega_0)$ equally, such that $B_1(\omega_0) = B_2(\omega_0) = B_0(\omega_0)/2$, then (2) predicts that $I_0(\delta) = B_0(\omega_0)(1 + \cos d)$. Substitution of (3) into this expression gives,

$$I_0(\delta) = B_0(\omega_0)[1 + \cos(4\pi\delta\omega_0)]. \tag{4}$$

However, as we stated previously, the source used for FT-IR is broadband, with spectral intensity distribution $B_0(\omega)$. Therefore, the interferogram recorded by the detector is a superposition of cosines,

$$I_0(\delta) = \int_0^\infty B_0(\omega)[1+\cos(4\pi\delta\omega)]d\omega = \int_0^\infty B_0(\omega)d\omega + \int_0^\infty B_0(\omega)\cos(4\pi\delta\omega)d\omega. \tag{5}$$

Evaluating (5) at $\delta = 0$, we find that,

$$I_0(0) = 2\int_0^\infty B_0(\omega)d\omega. \tag{6}$$

Using the two previous equations to remove the constant term,

$$I(\delta) = I_0(\delta) - \frac{1}{2}I_0(0) = \int_0^\infty B_0(\omega)\cos(4\pi\delta\omega)d\omega, \tag{7}$$

where the right hand side contains all the spectral information, and is the well-known cosine Fourier integral for the distribution $B_0(\omega)$. The inverse Fourier transform may be used to recover $B_0(\omega)$, with the result

$$B_0(\omega) = \int_0^\infty I(\delta) \cos(4\pi\delta\omega) d\delta. \tag{8}$$

There is a hidden subtlety in (8), as pointed-out by Davis, *et al.*, whose treatment we have been following to this point [2]. Fourier analysis produces an unphysical mirror-image of $B_0(\omega)$ at negative frequencies, that is $B_0(-\omega)$. Because cosine is an even function, both $B_0(\omega)$ and $B_0(-\omega)$ map to identical interferograms. We may eliminate this inconsistency with our physical intuition by constructing distribution $B(\omega) = \frac{1}{2}[B_0(\omega) + B_0(-\omega)]$. Changing our definitions to incorporate all frequencies,

$$I(\delta) = \int_{-\infty}^\infty B(\omega) \cos(4\pi\delta\omega) d\omega, \tag{9}$$

and,

$$B(\omega) = \int_{-\infty}^\infty I(\delta) \cos(4\pi\delta\omega) d\delta. \tag{10}$$

## 2.2 Spectroscopy

### 2.2.1 Overview

Spectroscopy is a powerful tool for the study of molecular structure. Analysis of a molecular spectrum yields information on discrete energy level values, from which the molecule's electron configuration, in addition to molecule's rotation and vibration can be deduced. Furthermore, it is possible to make quantitative determinations of the molecule's environment, including temperature and pressure. This is done by studying the strength of each spectral line, and through detailed analysis of the environmental perturbations to the ideal spectrum. These perturbations manifest themselves as broadening of the spectral lines.

The spectroscopy of a polyatomic molecule may be arbitrarily complex, due to transitions between electronic, vibrational, and rotational levels. Because our work involves the exclusive study of carbon dioxide, we may simplify the problem by considering only the spectroscopy of linear, triatomic molecules (e.g. $CO_2$). Transitions between electronic states result, for the most part, in absorption lines in the visible or ultraviolet region, and consequently may be ignored for our work, which is limited to the infrared. According to Herzberg, in the ground electronic state the angular momentum about the internuclear axis of a linear polyatomic molecule is exactly zero [3]. In this approximation, the rotational energy levels as a function of rotational quantum number, $J$, is given by

$$F(J) = \frac{E_r}{hc} = BJ(J+1) - DJ^2(J+1)^2 + \cdots, \tag{11}$$

where $B$ is the rotational constant,

$$B = \frac{h}{8\pi^2 c I_B}. \tag{12}$$

In (12), $I_B$ is the moment of inertia about an axis perpendicular to the internuclear axis and going through the molecule's center of mass. For a symmetric, linear molucule, such as Y-X-Y, $I_B$ is given by $2mr^2$, where $m$ is the mass of atom Y and $r$ is the X-Y bond length. The second term in (11) arises due to molecular stretching induced by the centrifugal force exerted on a rotating molecule. In the harmonic oscillator approximation, $D$ may be related to $B$ and to the vibrational frequency by,

$$D = \frac{4B^3}{\omega^2}. \tag{13}$$

Higher order terms exist in (11), though they are exceedingly small, and may be neglected for $CO_2$. By enforcing the selection rule for rotational transitions, $\Delta J = \pm 1$, we are able to derive the infrared spectrum for a non-rigid rotator,

$$\nu = F(J-1) - F(J) = 2B(J+1) - 4D(J+1)^2. \tag{14}$$

As we have alluded to, by inclusion of the constant $D$ in the preceding work, molecules undergo periodic vibrations while rotating. Considering only symmetric, linear molecules, three fundamental frequencies are observed, corresponding to the $\nu_1$ symmetric stretch, $\nu_2$ perpendicular, and $\nu_3$ asymmetric stretch. These three vibrational modes are shown in figure 2.2.1.

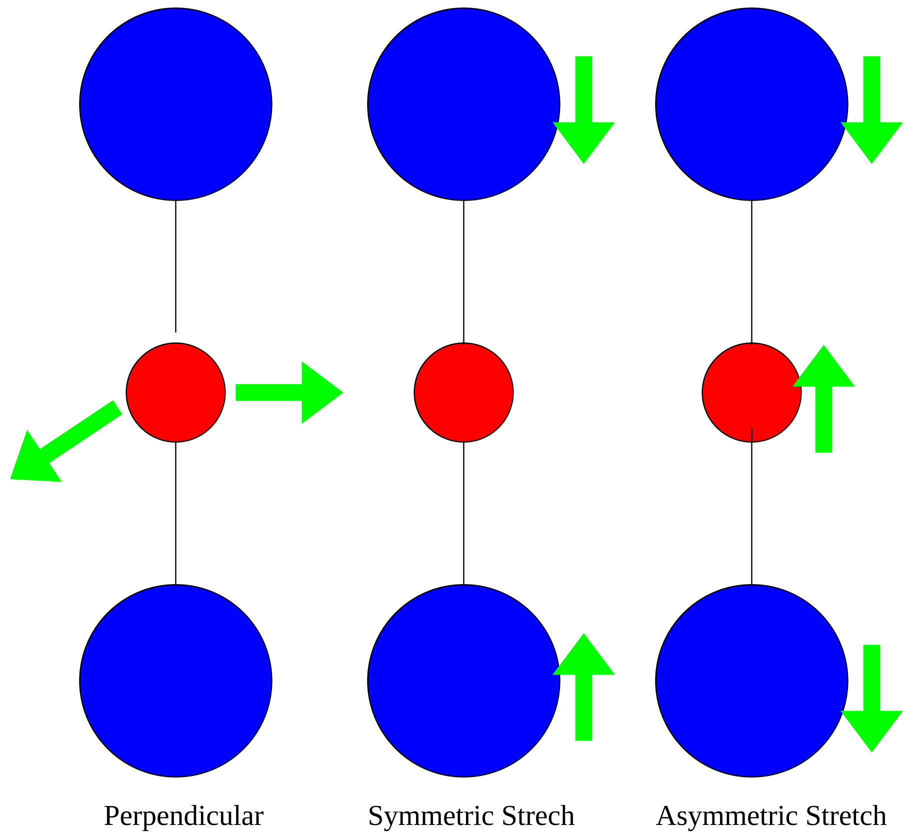

Figure 2: Vibrational modes of $CO_2$. From left to right, the $\nu_2$, $\nu_1$, and $\nu_3$ modes at 667.3 cm$^{-1}$, 1388.3 cm$^{-1}$, and 2349.3 cm$^{-1}$ respectively. Only the $\nu_2$ and $\nu_3$ modes are infrared active.

In addition to the three fundamental frequencies, overtone, difference, and combination bands also exist. For our work, we are interested in the $\nu_3$ band, centered at 2349.3 cm$^{-1}$. This band is very stong, resulting in measurable absorption even with low $CO_2$ concentrations.

### 2.2.2 Natural Line Broadening

Natural line broadening results from the inevitable decay of electrons from an excited state to a lower one. Our discussion of this process at a basic level follows that of Silfvast [9]. In a simple two-level model, for isolated (non-interacting) atoms, with upper state $u$, and energy $\epsilon_u$, and lower state $l$ with energy $\epsilon_l$ ($\epsilon_l < \epsilon_u$) this process may be expressed mathematically as,

$$\frac{dN_u}{dt} = -A_{ul}N_u \tag{15}$$

where $N_u$ is the number of electrons in the upper state[1], $A_{ul}$ is the radiative transition rate from state $u$ to $l$ [2]. The solution to equation (15) is found to be,

$$N_u(t) = N_u(0)e^{-A_{ul}t} = N_u(0)e^{-\frac{t}{\tau_u^{rad}}}. \tag{16}$$

The radiative lifetime of state $u$, $\tau_u^{rad}$, is defined to be the reciprocal of the radiative transition rate, $A_{ul}$.

In a multi-level model, with upper state $u$ and lower states $i, j, k, \ldots$ equation (15) must be written as,

$$\frac{dN_u}{dt} = -(A_{ui} + A_{uj} + A_{uk} + \ldots)N_u = -\sum_{\alpha<u} A_{u\alpha}N_u, \tag{17}$$

where $N_u$ has the same meaning as above, and $A_{u\alpha}$ is the radiative transition rate from state $u$ to state $\alpha$. The summation is carried out over **all** states with energy less than $\epsilon_u$. Analogous to the result obtained previously, the solution to (17) is,

$$N_u(t) = N_u(0)\exp\Big[-\Big(\sum_{\alpha<u} A_{u\alpha}\Big)t\Big] = N_u(0)e^{-\frac{t}{\tau_u^{rad}}}, \tag{18}$$

[1] $N_u$ must be large enough that statistical fluctuations may be overlooked and, simultaneously, small enough to preclude interaction between the molecules.

[2] $A_{ul}$ is also known as the Einstein coefficient for spontaneous emission or the Einstein A coefficient.

where the radiative lifetime of state $u$ has been defined by (19),

$$\tau_u^{rad} \equiv \frac{1}{\sum_{\alpha<u} A_{u\alpha}}. \tag{19}$$

For succinctness and consistency with later notation, we make the following definition for the half width of the line in wavenumbers,

$$\gamma_u^{rad} \equiv \frac{1}{4c\pi\tau_u^{rad}}. \tag{20}$$

Invoking the Heisenberg Uncertainty Principle, $\Delta\epsilon\Delta t \approx \hbar$, and equating $\tau_u^{rad}$ with the uncertainty in time, we conclude that

$$\Delta\epsilon_u = \frac{\hbar}{\tau_u^{rad}} = \hbar\sum_{\alpha<u} A_{u\alpha}. \tag{21}$$

By the same reasoning, the lower energy level $l$ would also have a finite width, given by

$$\Delta\epsilon_l = \frac{\hbar}{\tau_l^{rad}} = \hbar\sum_{\beta<l} A_{l\beta}. \tag{22}$$

The total effective energy width for the transition from $u$ to $l$ is found by adding the two separate widths, yielding

$$\Delta\epsilon_{tot} = \Delta\epsilon_u + \Delta\epsilon_l = \hbar\Big(\sum_{\alpha<u} A_{u\alpha} + \sum_{\beta<l} A_{l\beta}\Big) = h\Delta\nu_{ul}^{rad}, \tag{23}$$

where $\Delta\nu_{ul}^{rad}$ represents the uncertainty in the emitted (or absorbed) frequency. Rewriting (21 - 23),

$$\Delta\nu_{ul}^{rad} = \frac{1}{2\pi}\Big(\sum_{\alpha<u} A_{u\alpha} + \sum_{\beta<l} A_{l\beta}\Big) = \frac{1}{2\pi}\Big(\frac{1}{\tau_u^{rad}} + \frac{1}{\tau_l^{rad}}\Big) = 2c(\gamma_u^{rad} + \gamma_l^{rad}) = 2c\gamma^{rad}, \tag{24}$$

where $\gamma^{rad}$ is proportional to the total relaxation rate from states $u$ and $l$.

This result has been confirmed through a more rigorous, quantum-mechanical approach by V. Weisskopf and E. Wigner [4]. They begin by showing that for an electron in stationary state $m$ (energy $\epsilon_m$), the probability that its energy lies between $\epsilon$ and $\epsilon + d\epsilon$ is given by,

$$W(\epsilon)d\epsilon = \frac{c\gamma_m^{rad}}{\pi}\frac{h}{(hc\,\gamma_m^{rad})^2 + (\epsilon - \epsilon_m)^2}d\epsilon. \tag{25}$$

Consequently, the simultaneous probability that the upper state has energy between $\epsilon$ and $\epsilon + d\epsilon$ and the lower state has energy between $\epsilon'$ and $\epsilon' + d\epsilon'$ is given by $W(\epsilon)W(\epsilon')d\epsilon d\epsilon'$. Additionally, we have the constraint that $h\nu = \epsilon - \epsilon'$, so that for fixed values of $\epsilon$,

$$|d\epsilon'| = h|d\nu|. \tag{26}$$

The probability of emission (or absorption) occurring at a frequency between $\nu$ and $\nu + d\nu$, denoted by $J'(\nu)d\nu$, is found by replacing $\epsilon'$ and $d\epsilon'$ in our expression for $W(\epsilon)W(\epsilon')d\epsilon d\epsilon'$ according to (26), then integrating over all possible values of $\epsilon$. Thus, we obtain for the normalized line shape function

$$J'(\nu)d\nu = J(|\nu - \nu_{ul}|)d\nu = \frac{1}{\pi}\frac{c\gamma^{rad}}{(\nu - \nu_{ul})^2 + (c\gamma^{rad})^2}d\nu. \tag{27}$$

This intensity distribution may alternatively be found by Fourier analysis of a classical oscillator with damping constant $\gamma^{rad}$ [5, 6, 7]. From equation (27) we find that $J(\nu)$ falls to one-half of its peak value at frequency $\nu'$, such that

$$|\nu' - \nu_{ul}| = \frac{1}{4\pi}\Big(\sum_{\alpha<u} A_{u\alpha} + \sum_{\beta<l} A_{l\beta}\Big) = c(\gamma_u^{rad} + \gamma_l^{rad}) = c\gamma^{rad}. \tag{28}$$

Comparison of (24) and (28) reveals that $\Delta\nu_{ul}^{rad} = 2c\gamma^{rad}$ represents the FWHM of the Lorentzian profile defined by (27), while $|\nu' - \nu_{ul}| = c\gamma^{rad}$ represents the HWHM [8].

Conversion from frequency ($\nu$) to wavenumber ($\omega$) space, via $\nu\lambda = c \rightarrow \nu = c\omega$ transforms (27) into

$$J(|\omega - \omega_{ul}|)d\omega = \frac{1}{\pi}\frac{(\gamma^{rad})}{(\omega - \omega_{ul})^2 + (\gamma^{rad})^2}d\omega. \tag{29}$$

Consequently, the natural line half-width (HWHM) in wavenumbers is defined to be

$$|\omega' - \omega_{ul}| = (\gamma_u^{rad} + \gamma_l^{rad}) = \gamma^{rad}. \tag{30}$$

### 2.2.3 Collisional Line Broadening

Molecular collisions are a complex, quantum-mechanical process that alter the system's energy distribution. Two limiting types of collisions may be identified, depending on the assumptions employed. Strong collisions, which are generally regarded as adiabatic, (i.e. occurring over a short time interval compared to the period of oscillation) are such that the molecule is presumed to be distributed according to the Boltzmann Law after the collision. That is to say, the molecule's post-collision state is independent of its state prior to the collision. Weak interactions arise when a single collision does not impart enough energy for the system to jump to the next higher energy level. Under these conditions an appreciable change in the system's state may only occur as the result of a large number of collisions; though sufficient deformation may occur in a weak collision to cause the central frequency to shift [10].

Early attempts by Lorentz to model the quantum-mechanical processes responsible for strong collisional broadening were based on the Fourier analysis of the classical oscillator undergoing collisions. However, this theory was unable to predict the correct line shape for some regions of the electromagnetic spectrum under certain pressure and temperature conditions. Ben-Reuven was able to develop a significantly more generalized line shape than Lorentz by considering collisions that changed the oscillator's amplitude, phase, orientation, and momentum. The Ben-Reuven line shape is given by the expression[3]

$$\widetilde{J}'(\omega)d\omega = \frac{2(\gamma^{col} - \xi)\omega^2 + 2(\gamma^{col} + \xi)[(\omega_{ul} + \hat{\delta})^2 + (\gamma^{col})^2 - \xi^2]}{[\omega^2 - (\omega_{ul} + \hat{\delta})^2 - (\gamma^{col})^2 + \xi^2]^2 + 4\omega^2(\gamma^{col})^2} d\omega \tag{31}$$

where $\gamma^{col}$ is proportional to the collisional damping rate of the oscillator, and therefore related to the pressure-broadened line width ($\Delta\omega_{ul}^{col}$), $\xi$ describes the mean rate of momentum-reverting collisions (also known as the coupling coefficient or coupling element), and $\hat{\delta}$ measures the pressure induced shift in the line center that results

[3]The notation $\widetilde{J}$ has been used to indicate that this expression is not normalized to unit area.

from weak interactions [11, 12, 13, 14].

If there are no momentum-reverting collisions, that is $\xi = 0$, then (31) reduces to the Van Vleck-Weisskopf line shape function [10, 11, 15]

$$J'(\omega)d\omega = \frac{1}{2\pi}\Big[\frac{\gamma^{col}}{(\omega - \omega_{ul} - \hat{\delta})^2 + (\gamma^{col})^2} + \frac{\gamma^{col}}{(\omega + \omega_{ul} + \hat{\delta})^2 + (\gamma^{col})^2}\Big]d\omega. \tag{32}$$

Formally, equation (32) predicts a "physical" resonance at $(+\omega_{ul} + \hat{\delta})$ and an "unphysical" resonance at $(-\omega_{ul} - \hat{\delta})$, with a tail extending into the "physical" region, $\omega > 0$. If we concern ourselves with a small region around $+\omega_{ul}$, and consider a small pressure shift, then the first term in (32) is negligible, yielding the familiar Lorentzian profile

$$J'(\omega)d\omega = \frac{1}{\pi}\frac{\gamma^{col}}{(\omega - \omega_{ul} - \hat{\delta})^2 + (\gamma^{col})^2}d\omega. \tag{33}$$

This is the same result found by Lorentz using a classical oscillator model, with the addition of the pressure induced shift term $\hat{\delta}$. Furthermore, this outcome is logical if we view collisions as a process that increases the decay rate, thereby reducing the state's lifetime. Consistent with §2.2.2, this process yields a Lorentzian line shape with pressure-broadened half-width $\gamma^{col}$ determined by the mean stimulated decay rate; while previously the radiatively-broadened half-width $\gamma^{rad}$ was determined by the mean spontaneous decay rate.

Analogous to (20), $\gamma^{col}$ is defined by,

$$\gamma^{col} \equiv \frac{1}{4c\pi\bar{\tau}^{col}}, \tag{34}$$

where $\bar{\tau}^{col}$ is the mean time between collisions. This time may be found by assuming that a collision occurs each time two molecules, $A$ and $B$, approach each other within a certain distance, referred to as the optical collision diameter. The optical collision diameter, $d_{AB}$, is given by

$$d_{AB} = \frac{1}{2}(d_A + d_B), \tag{35}$$

where $d_A$ and $d_B$ are the optical diameters of molecules $A$ and $B$ respectively. The corresponding optical cross section is $\pi(d_{AB}^2)$. The mean time between collisions of a single molecule of species $B$ and *all* $A$ molecules, $\bar{\tau}_{AB}^{col}$ equals $[n_A\pi(d_{AB}^2)\bar{c}]^{-1}$, where $n_A$ is the number density of species $A$, $\bar{c}$ is the mean relative speed,

$$\bar{c} = \Big(\frac{8kT}{\pi\mu_{AB}}\Big)^{\frac{1}{2}}, \tag{36}$$

and $\mu_{AB}$ is the reduced mass of $A$ and $B$. The mean time between collisions for a single molecule of species $B$ and all collision partners, $\bar{\tau}_B^{col}$ is found by summation,

$$\bar{\tau}_B^{col} = \Bigg[\sum_A n_A(\sigma_{AB}^2)\Big(\frac{8\pi kT}{\mu_{AB}}\Big)^{\frac{1}{2}}\Bigg]^{-1}. \tag{37}$$

Employing the ideal gas law, $p = nkT$, we find:

$$\bar{\tau}_B^{col} = \Bigg[p\sum_A \chi_A(\sigma_{AB}^2)\Big(\frac{8\pi}{\mu_{AB}kT}\Big)^{\frac{1}{2}}\Bigg]^{-1}, \tag{38}$$

where $\chi_A$ is the mole fraction of collision partner $A$. Substituting (38) into (35), we find that the pressure broadened half-width is,

$$\gamma^{col} = \frac{p}{4c\pi}\Bigg[\sum_A \chi_A(\sigma_{AB}^2)\Big(\frac{8\pi}{\mu_{AB}kT}\Big)^{\frac{1}{2}}\Bigg]. \tag{39}$$

The important thing to notice in (39) is the dependence of $\gamma^{col}$ on $p$ and $T$; that is,

$$\gamma^{col} \propto \frac{p}{T^{\frac{1}{2}}}. \tag{40}$$

More generally, the half-width is assumed to vary as

$$\gamma^{col} \propto \frac{p}{T^n}, \quad \text{with } 0 < n < 1, \tag{41}$$

where the coefficient of temperature dependence, $n$, is adjusted empirically [7]. The discrepancy between (40) and (41) results from the approximations of kinetic theory employed in developing (40), as opposed to a more rigorous, quantum-mechanical calculation.

The HiTran database, which predicts infrared absorption, uses a similar scheme to approximate the pressure broadening. The pressure broadening is characterized as coming from two sources: self-broadening (i.e. coming from like molecules) and air-broadening (i.e. coming from foreign molecules). Thus, the total pressure-broadened half width is a linear combination of self-broadened and air-broadened half-width terms, weighted by the partial pressures of "like" and "foreign" gases respectively. Furthermore, the entire expression is scaled from a measured half-width at $p_{ref} = 1$ atm and $T_{ref} = 296$ K to the desired temperature $T$ $[K]$, and pressure $p$ [atm], according to

$$\gamma^{col}(p,T) = \Big(\frac{T_{ref}}{T}\Big)^n \Big[\gamma^{col}_{air}(p_{ref},T_{ref})(p-p_s) + \gamma^{col}_{self}(p_{ref},T_{ref})p_s\Big], \tag{42}$$

where $p_s$ is the partial pressure of the gas [1]. In **specgen** we use (42) to calculate the pressure-broadened half-width for each line, employing transition-dependent values of $\gamma^{col}_{air}$, $\gamma^{col}_{self}$, and $n$ supplied by HiTran.

### 2.2.4 Doppler Line Broadening

Neglecting for a moment the previously discussed broadening mechanisms, we motivate the presence of Doppler broadening. If, for a gas of stationary molecules, a transition between states $u$ and $l$ results in the emission (or absorption) of light at frequency $\nu_{ul}$, then the *actual* frequency of light emitted (or absorbed) by molecules moving with velocity $v_x$ in the direction of the line of sight is shifted to $\nu$ according to the Doppler principle such that,

$$\nu = \nu_{ul}\Big(1 - \frac{v_x}{c}\Big). \tag{43}$$

For a gas in thermal equilibrium, the distribution of velocities is spherically symmetric, and given by the Maxwell velocity distribution,

$$P(v_x)dv_x = \Big(\frac{m}{2\pi k_B T}\Big)^{\frac{1}{2}} \exp\Big(-\frac{m}{2k_B T}v_x^2\Big)dv_x, \tag{44}$$

which gives the probability that a molecule of mass $m$ and temperature $T$ moving in direction $x$ has a velocity between $v_x$ and $v_x + dv_x$ [8].

The probability of observing emission (or absorption) in a frequency range between $\nu$ and $\nu + d\nu$ is found by first solving (43) for $v_x$,

$$v_x = \frac{c}{\nu_{ul}}(\nu - \nu_{ul}) \tag{45}$$

and subsequently differentiating (45),

$$dv_x = \frac{c}{\nu_{ul}} d\nu \tag{46}$$

and squaring (45),

$$v_x^2 = \frac{c^2}{\nu_{ul}^2}(\nu - \nu_{ul})^2. \tag{47}$$

Ultimately, the desired expression may be found via substitution of (46) and (47) into (44), with the result

$$J'(\nu)d\nu = J(|\nu - \nu_{ul}|)d\nu = \frac{c}{\nu_{ul}}\Big(\frac{m}{2\pi kT}\Big)^{\frac{1}{2}} \exp\Big[-\frac{mc^2(\nu - \nu_{ul})^2}{2kT\nu_{ul}^2}\Big]d\nu. \tag{48}$$

Re-expressing (48) in wavenumbers,

$$J'(\omega)d\omega = J(|\omega - \omega_{ul}|)d\omega = \frac{c}{\omega_{ul}}\Big(\frac{m}{2\pi kT}\Big)^{\frac{1}{2}} \exp\Big[-\frac{mc^2(\omega - \omega_{ul})^2}{2kT\omega_{ul}^2}\Big]d\omega. \tag{49}$$

The half-width of the Gaussian profile defined by (49) is found to be

$$|\omega' - \omega_{ul}| = \omega_{ul}\sqrt{\frac{2kT\ln(2)}{mc^2}} = \gamma^{dop}, \tag{50}$$

where $\omega'$ is defined by the relationship $J'(\omega') = J(0)/2$.

### 2.2.5 Combined Effects

Comparison of equations (20), (41), and (50) reveals that under different conditions of temperature and pressure the natural, collisional, and Doppler broadening mechanisms may have different relative strengths. The natural line broadening is dependent on the quantum-mechanical properties of the transition, collisional broadening is determined by both temperature and pressure, while Doppler broadening is controlled by temperature and transition frequency. When one effect is dominant, that is it produces a half-width significantly greater than that produced by the other two mechanisms, the weaker effects may safely be neglected. However, when two or more effects are of similar strength, care must be taken when constructing the combined line profile. The total line shape is given by the convolution of the individual line shapes.

Convolution is a smoothing operation, that may act on discrete data sets or continuous functions. The discrete case, also known as a serial product, may be thought of as a weighted moving average. For $b$ a set of $2i+1$ points, $a = b \otimes_{discrete} g$ is found by,

$$a_n = \sum_{m=-i}^{+i} b_{n-m} g_m. \tag{51}$$

For the functions $a$, $b$, and $g$ of continuous variables $x$ and $x'$, the convolution $a = b \otimes g$ is defined according the formula

$$a(x) = \int_{-\infty}^{+\infty} b(x - x') g(x') dx'. \tag{52}$$

However, because our experimental and theoretical data are stored as discrete samplings of continuous functions, eq. (51) proves to be a much more useful formula. Convolutions are covered in greater detail in a number of sources, though Jansson is particularly thorough [16].

For simplicity, we begin by considering combined natural and collisional broadening, in the limit both mechanisms are modeled with a Lorentzian line shape. It may be shown that the convolution of two Lorentzians, $\phi_1(\omega)$ and $\phi_2(\omega)$, of the form

$$\phi(\omega) = \frac{1}{\pi} \frac{\gamma}{(\omega - \omega_0)^2 + \gamma^2}, \tag{53}$$

with half widths $\gamma_1$ and $\gamma_2$ respectively, produce a third Lorentzian with half width $\gamma_3 = \gamma_1 + \gamma_2$. For our work, the total width is therefore proportional to the sum of the spontaneous decay rate and the collision rate. However, for infrared transitions of gases at p $\gtrsim$ 1 atm the lifetime of an excited level is significantly longer than the mean time between collisions, making the effects of natural line broadening negligible compared to pressure broadening.

A much more difficult situation arises when trying to calculate the line shape resulting from the convolution of Gaussian and Lorentzian profiles of comparable widths. Because our research takes place across a wide-range of temperatures and pressures, we are unable to neglect one of these effects. The convolution of such profiles generates a Voigt profile [17]

$$J'(\omega)d\omega = \left\{ \frac{1}{\gamma^{dop}} \sqrt{\frac{\ln 2}{\pi}} \left[ \frac{y}{\pi} \int_{-\infty}^{+\infty} \frac{\exp(-t^2)}{y^2 + (x-t)^2} dt \right] \right\} d\omega, \tag{54}$$

where

$$x \equiv \frac{\omega - \omega_{ul}}{\gamma^{dop}} \sqrt{\ln 2} \tag{55}$$

converts the wave number scale to units of Doppler half widths, and

$$y \equiv \frac{\gamma^{rad} + \gamma^{col}}{\gamma^{dop}} \sqrt{\ln 2} \simeq \frac{\gamma^{col}}{\gamma^{dop}} \sqrt{\ln 2} \tag{56}$$

gives the ratio of Lorentz to Doppler widths. In (54), the expression inside the square brackets is referred to as the Voigt function, while the terms preceding it are for normalization. Calculation of eq. (54) is inherently difficult, as no closed form solution of the integral is known. However, due to its spectroscopic importance a number of expansions have been developed dependent on the values of $x$ and $y$ [8, 18, 19, 21]. The paper by Asfaw (ref. [19]) appears to contain significant errors, though much of the basic framework is correct. The **specgen** program uses a slightly modified version of the algorithm developed by Wells, which uses a different expansion of the Voigt function dependent upon the Voigt $x$ and $y$ parameters [21].

### 2.2.6 Instrumental Line Broadening

In practice, our ability to measure an interferogram is inherently constrained by $|\delta| \leq \Delta$ where $\delta$ is the mirror displacement and $\Delta$ is the maximum mirror displacement.

This limitation is equivalent to the multiplication of the interferogram, $I(\delta)$, by a truncation function, such as the rectangle (window) function,

$$\prod_{2\Delta}(\delta) \equiv \begin{cases} 1 & |\delta| \leq \Delta \\ 0 & |\delta| > \Delta. \end{cases} \tag{57}$$

The convolution theorem states: *Convolution in one domain corresponds to multiplication in the other (Fourier) domain* [22]. Therefore, the product $I(\delta)\prod_{2\Delta}(\delta) = B(\omega) \otimes S(\omega)$, where the instrument function $S(\omega)$ is the Fourier transform of $\prod_{2\Delta}(\delta)$. In other words, the spectrum obtained from FT-IR spectroscopy is the convolution of the true spectrum $B(\omega)$ with $S(\omega)$. The instrument function, plotted in figure 3, is easily calculated,

$$S(\omega) = 2\Delta\left[\frac{\sin(2\pi\omega\Delta)}{2\pi\omega\Delta}\right] = 2\Delta\text{sinc}(2\pi\omega\Delta). \tag{58}$$

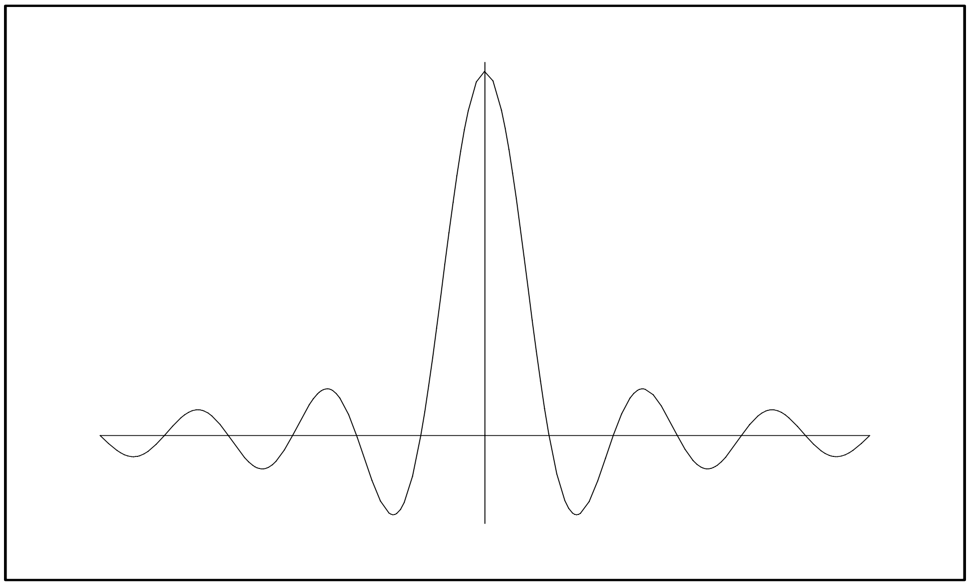

Figure 3: The sinc instrument function results from finite mirror travel in the interferometer. The sinc function must be convolved with the absorption line profile if apodization is not used.

Figure 3 reveals two problems with the instrument function as it stands. First the magnitudes of the primary sidelobes[4] are considerable, measuring $\sim 21\%$ of the

[4]Sidelobes correspond to the Gibb's phenomenon in a finite Fourier sum. Truncation of the interferogram is an

central peak height. Furthermore, the sidelobes are negative - an unphysical effect. These two effects introduce significant distortion to the recorded spectrum, though it may be reduced by way of apodization. Apodization, Greek for "little feet", is a technique by which the truncation function (57) is supplanted by an apodization function that has a more favorable instrument function. While this step may seem like black magic, it is legitimate provided the apodization function takes values between zero and one (inclusive) in the region $-\Delta \leq \delta \leq \Delta$, and is uniformly zero outside that region. One popular apodization function is the triangle function defined by,

$$\bigwedge_{2\Delta}(\delta) \equiv \begin{cases} 1 - |\frac{\delta}{\Delta}| & |\delta| \leq \Delta \\ 0 & |\delta| > \Delta. \end{cases} \tag{59}$$

The corresponding instrument function,

$$S(\omega) = \Delta\left[\frac{\sin^2(\pi\omega\Delta)}{(\pi\omega\Delta)^2}\right] = \Delta\ \text{sinc}^2(\pi\omega\Delta), \tag{60}$$

is shown in figure 4.

The sinc-squared instrument function is used in our modeling because it has small sidelobes (only $\sim$ 4.5% of peak height), is uniformly positive, and may be implemented with little computational overhead[5]. Additionally, triangular apodization is available on most FT-IR spectrometers. Therefore, we are able to apodize both the experimental and theoretical spectra in an identical manner. However, the improvement in appearance that comes from apodization is not free.

Convolution of the instrument function and absorption line profile results in an absorption line profile broader (HWHM) than either of the pre-convolved functions. As shown in figures 3 and 4, the half-widths of the sinc and sinc-squared instrument functions are,

$$\gamma^{sinc} = \frac{1.207}{2\Delta} \tag{61}$$

equivalent process.

[5]For a discussion of other popular apodization functions, see the classic text by Griffiths and de Haseth [23].

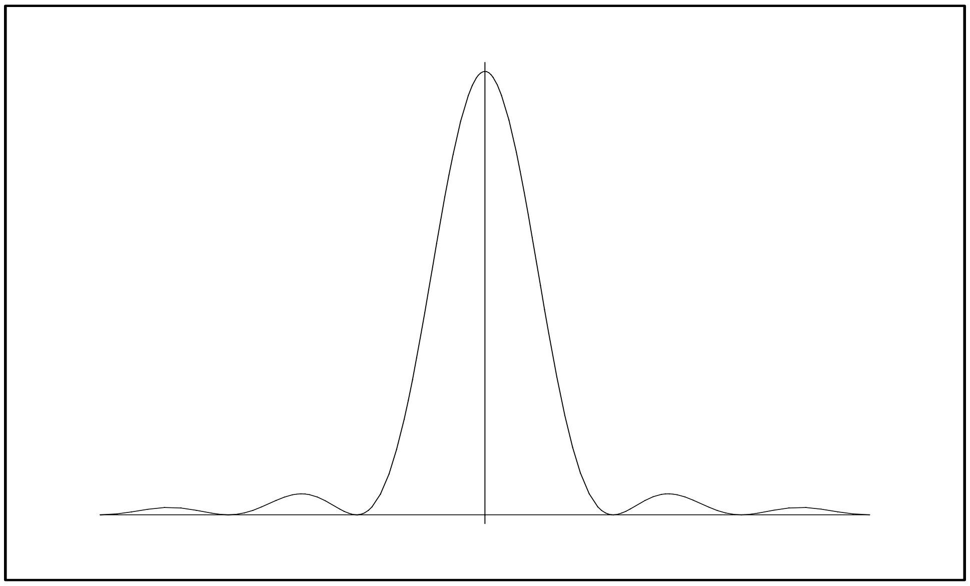

Figure 4: The sinc-squared instrument function results from triangular apodization of the interferogram. The sinc-squared function is convolved with the absorption line profile in the **specgen program because the experimental spectra were triangularly apodized.**

and,

$$\gamma^{sinc-squared} = \frac{1.772}{2\Delta} \tag{62}$$

respectively.

Because the area of the absorption line profile does not change in the convolution process, it is normalized to unit area before and after, the peak height of the profile is reduced by apodization because the resulting line is wider. Furthermore, instrumental broadening means that two, closely-spaced lines will no longer be distinguishable.

The ability of a spectrometer to separate two, closely-spaced lines is referred to as its resolution. While several criteria have been proposed to define an instrument's resolution, the most useful for triangularly apodized lines is the Rayleigh criterion. This condition states that two adjacent lines of equal intensity, with sinc-squared instrument functions, are fully resolved when the center of one line line is at the same frequency as the first zero of the other. In practice, this condition is satisfied such that

$$\text{resolution} \simeq \frac{1}{\Delta}, \tag{63}$$

where resolution is measured in wavenumbers and $\Delta$ is the maximum mirror displacement in centimeters.

#### 2.2.7 Quantitative Spectroscopy

The Lorentzian, Gaussian, and Voigt profiles considered previously give, for a transition centered at $\omega_{ul}$, the probability of observing emission (or absorption) between $|\omega - \omega_{ul}|$ and $|\omega - \omega_{ul}| + d\omega$. Since we assume a transition has occurred we have normalized all three to unit area. In practice, the population of the states involved and other factors contribute to a transition's intensity, $S_{ul}$. The quantity $S_{ul}$ is provided for each transition by HiTran, after being scaled to the correct temperature.

The Beer-Lambert law relates the intensity of light at frequency $\omega$ that is transmitted through a sample with absorption coefficient $\kappa(\omega)$, concentration $c$, and path length $l$ to the incident intensity, $I_0$, by the formula,

$$I(\omega) = I_0(\omega)\exp(\kappa(\omega)cl). \tag{64}$$

The ratio $\frac{I(\omega)}{I_0(\omega)}$ is determined from the experimental spectrum, $l$ may be measured, $c$ varies in a known manner with temperature and pressure, and $\kappa(\omega)$ is given by

$$\kappa(|\omega - \omega_{ul}|) = S_{ul} J(|\omega - \omega_{ul}|). \tag{65}$$

### 2.3 Fluid Dynamics

For our purposes, the discussion of fluid dynamics may be limited to the very specific case of one-dimensional, isentropic flow. In this approximation, mass, momentum, and energy conservation laws yield three important equations for determining approximate temperature, pressure, and density within the free stream. The ratio of specific heats for the sample gas is given by,

$$\gamma_0 \equiv \frac{C_p}{C_v}, \tag{66}$$

and Mach number M, the ratio of total (reservoir) temperature, $T$, pressure, $p$ , and density, $\rho$ to static (free stream) conditions are given by [20],

$$\frac{T_0}{T} = 1 = \frac{\gamma_0 - 1}{2} M^2, \tag{67}$$

$$\frac{p_0}{p} = \left(1 + \frac{\gamma_0 - 1}{2} M^2\right)^{\frac{\gamma_0}{\gamma_0 - 1}}, \tag{68}$$

and,

$$\frac{\rho_0}{\rho} = \left(1 + \frac{\gamma_0 - 1}{2} M^2\right)^{\frac{1}{\gamma_0 - 1}} \tag{69}$$

respectively. These equations are used by our program **specgen** to convert user supplied reservoir conditions to free stream conditions based upon the Mach number at which the tunnel is operating. The absorption spectrum in modeled at free stream conditions.

## 3 Software

Two complementary pieces of software were vital to this project: the publicly available HiTran Database[6] and our program, **specgen**[7] Each of these programs will be discussed in turn.

### 3.1 HiTran

HiTran is a searchable collection of spectroscopic parameters that may be used to simulate the absorption of light in the atmosphere. The user selects the molecules

[6]The HiTran Database and support files may be downloaded from the ftp server at ftp://cfa-ftp.harvard.edu/pub/hitran.

[7]See C++ source code listing in Appendix B.

and isotopes of interest. HiTran then returns a listing of parameters for each transition above a user specified threshold intensity and within the frequency interval specified. The parameters are automatically scaled to reflect the temperature entered by the user. A sample HiTran output file listing is shown in (Appendix B - table 3). The HiTran output file is parsed by **specgen**, to which we now turn.

## 3.2 specgen

Our program **specgen** computes a FT-IR absorption spectrum, taking into account the effects of Mach number, temperature, pressure, path length, apodization, and instrument resolution. Mach number is needed in wind tunnel flows because the stagnation (reservoir) temperature and pressure are altered as the gas accelerates through the tunnel, as explained in §2.3. After reading the HiTran output file into a vector, static values for the temperature, pressure, and density are calculated according to eqs. (67 - 69). The reservoir temperature and pressure in degrees Rankine and PSI[8] respectively are necessary inputs to the program. The concentration is computed from the density and mole fraction of the gas of interest.

The bulk of the program determines the absorption coefficient. To do so, a second vector, `freq`, is created, containing the frequencies at which the theoretical spectrum will be evaluated. The entries in this vector are centered on the frequencies listed in the HiTran output file, though for reasons that will be made apparent shortly, they extend to somewhat higher and lower frequencies. The instrument function, (60), is then evaluated at discrete spacings equal to the frequency interval between consecutive entries in vector `freq`.

Using the static temperature and pressure determined previously, coefficients from HiTran, and equations (42) and (50), the pressure and Doppler broadened half widths

[8]These units are used, as opposed to metric ones, for consistency with literature published by the Langley Wind Tunnel Enterprise.

are calculated. Subsequently, these values are used to compute the Voigt $x$ and $y$ parameters, which are passed to the function `Humlik`. This function uses an optimized version of Humlíček's W4 algorithm to approximate the Voigt function. This routine was adapted from one published in Fortran by Wells [21]. Conversion of this function to C++ was entirely syntactical. The Voigt function, as returned by `Humlik` must be properly normalized still, in accordance with (54).

Convolution of the instrument function and Voigt line shape is carried out through the discrete convolution formula, (51). Due to end-effects inherent to the discrete convolution operation, the `freq` vector was created with a buffer at either end, such that the middle section of the line profile was not distorted. While this adds to the computational time, it is necessary to prevent spectral anomalies.

The absorption coefficient is found by multiplying the post-convolved Voigt line shape by the transition intensity, $S_{ul}$, given by HiTran. The Beer-Lambert law (64) is used to calculate the amount of absorption, using the user supplied path length, the previously calculated concentration, and the absorption coefficient just computed. This process is carried out for each entry in the vector `freq`, and for each line in the HiTran data file. The HiTran file used in the validation work consisted of 555 transitions between 2200cm$^{-1}$ and 2500cm$^{-1}$ with intensity greater than $1 \times 10^{-21}$cm$^{-1}$/(molecule$-$cm$^{-2}$).

## 4 Experimental Setup

The appeal of FT-IR spectroscopy, lies, at least in part, in the ease at which commercially available systems may be adopted to our needs. This results in a clean, yet rugged system that may be transported from one facility to another with ease. The setup's simplicity is reflected in figures 5 and 6, which show the layout used for validation and the proposed layout for wind tunnel measurements.

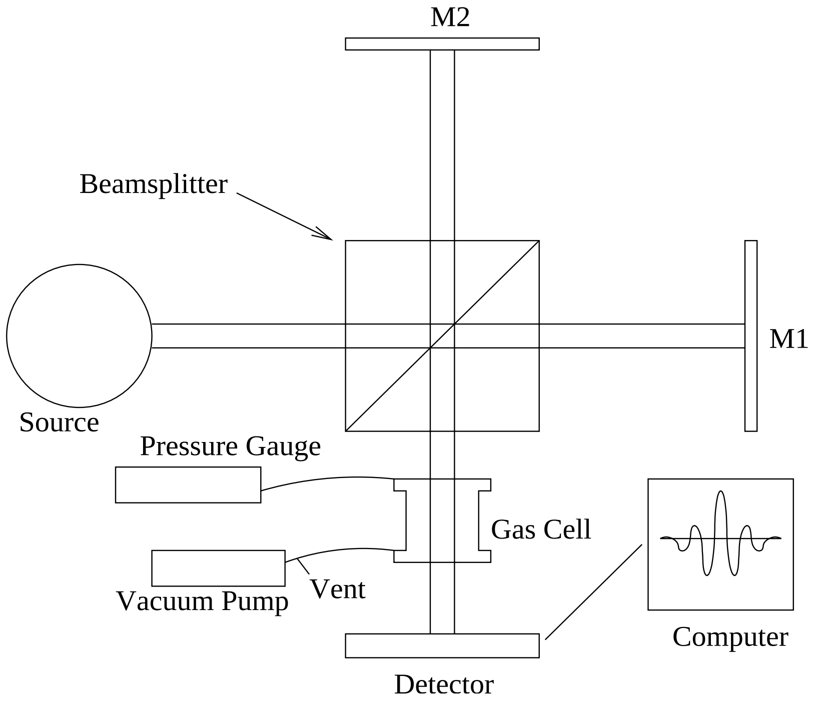

Figure 5: Experimental setup used for validation of the **specgen** program. (not to scale)

The FT-IR assembly includes a Nicolet Magna-IR 750 FT-IR spectrometer interfaced to a PC running Microsoft Windows 98 and manufacturer supplied software to control the instrument. The spectrometer consists of a light source, a beamsplitter made of KBr, and a Mercury Cadmium Telluride (MCT) detector, in a nitrogen-purged housing. This instrument is typically used to study the spectra of thin-films, though the sample chamber was large enough to house a custom made gas cell from MDC Vacuum Products Corp. This cell was 7.0 inches in length, with an interior path length of 5.9$\pm$0.1 in., and had a diameter of 1.0 inches. Infrared transparent windows ($\sim$ 90 % transmission per cm in the region of interest), made of 5 mm thick KCl, were installed at both ends. Finally, ports near either end of the cell allowed a vacuum pump and pressure gauge to be attached. For background scans the pump was capable of evacuating the cell to pressures of less than 1.0 Torr in only several seconds. System leaks caused the cell pressure to rise at a peak rate of $\sim$ 0.5 Torr per minute when the vacuum pump was turned off. This effect could be neglected though, as a single spectrum took less than five seconds to collect.

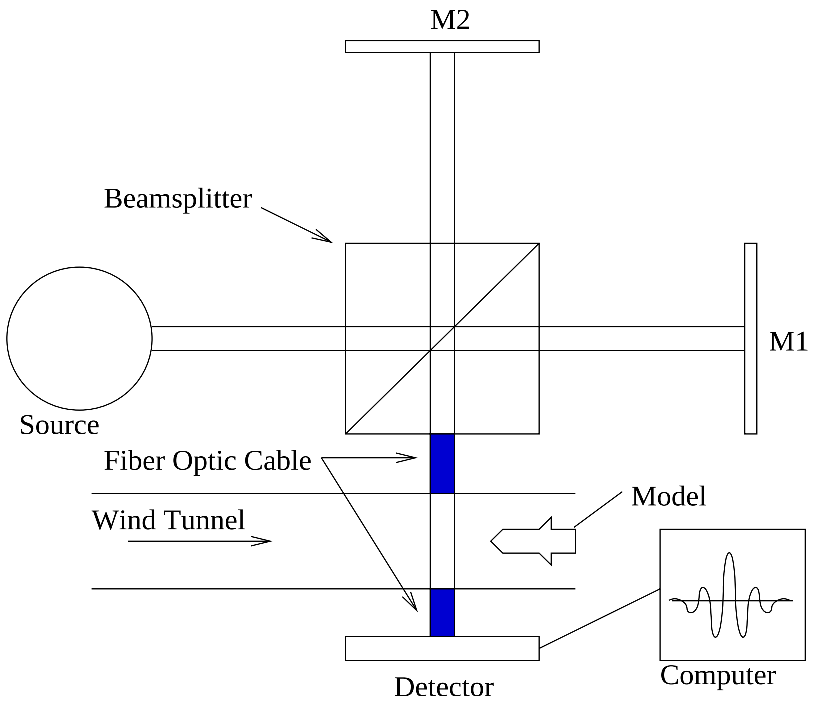

Figure 6: Experimental setup proposed for wind tunnel testing. (not to scale)

The layout used to model validation would require only slight modification to be used in gas flow studies. These adjustments consist of coupling fiber optics to the source and the detector, and are commercially available as customer-installed kits from many manufacturers. Fiber optics are necessary to prevent absorption of the infrared beam due to carbon dioxide present in the room air. The beam path (either external or sample chamber) is controlled via personal computer.

# 5 Results

## 5.1 Accuracy

The data used for quantitative analysis consisted of 11 experimental spectra and a single background spectrum, all ranging from 600cm$^{-1}$ to 4000cm$^{-1}$. The data were collected at 0.5cm$^{-1}$ resolution, room-temperature ($\sim$ 298 K), and varying pressures from 0.7 Torr to 773.5 Torr (0.01 PSI to 14.96 PSI). Additionally, a background spectrum was taken at 0.85 Torr (.02 PSI). Furthermore, each of the spectra was an average of 32 individual scans to improve the signal-to-noise. The experimental spectra were automatically divided by the background spectra to eliminate of absorption due to the cell windows and carbon-dioxide remaining within the system, in addition to correcting for the frequency dependent intensity of the light source. The absorption path length was presumed to be the interior length of the cell, 15.1 $\pm$ 0.25cm.

Experimental spectra were written to individual files that contained columns for frequency (wavenumbers) and the corresponding transmission (percent transmission). Visual inspection of the data revealed minor systematic errors. The average transmission in sections of the spectra in which HiTran did not predict strong transitions was greater than 100 %. We believe that gradual warming of the cooled detector was responsible, for the effect was found to vary with the amount of time that had passed since the background scan was taken. The region between 2390cm$^{-1}$ and 2400cm$^{-1}$ was chosen as a reference region, to be normalized to 100 % transmission. This section was selected because it was free of strong transitions according to HiTran. Normalization was conducted on a spectrum-by-spectrum basis through multiplication of the spectra by a constant, $\epsilon$,

$$\epsilon \equiv \frac{100}{\bar{T}}, \tag{70}$$

where $\bar{T}$ is the average percentage transmission of the forty data points in the reference region. Values of $\epsilon$ ranged from 0.9887 to 0.9992.

Spectral library files, with similar format to the experimental spectra (columns for wavenumber and fractional absorption) were generated at 2 K temperature increments and 0.1 PSI pressure increments using our code **specgen**. The library files contained five frequency data points per experimental point. To ease comparison between theory and experiment the frequency of every fifth library point equals a frequency in the experimental data. This arrangement is illustrated in figure 7 for a single absorption line of the experimental spectrum collected at 773.5 Torr. In this figure we show the normalized experimental data, using $\epsilon = 0.99284$, and the "best-fit" spectrum generated at 318 K and 15.0 PSI.

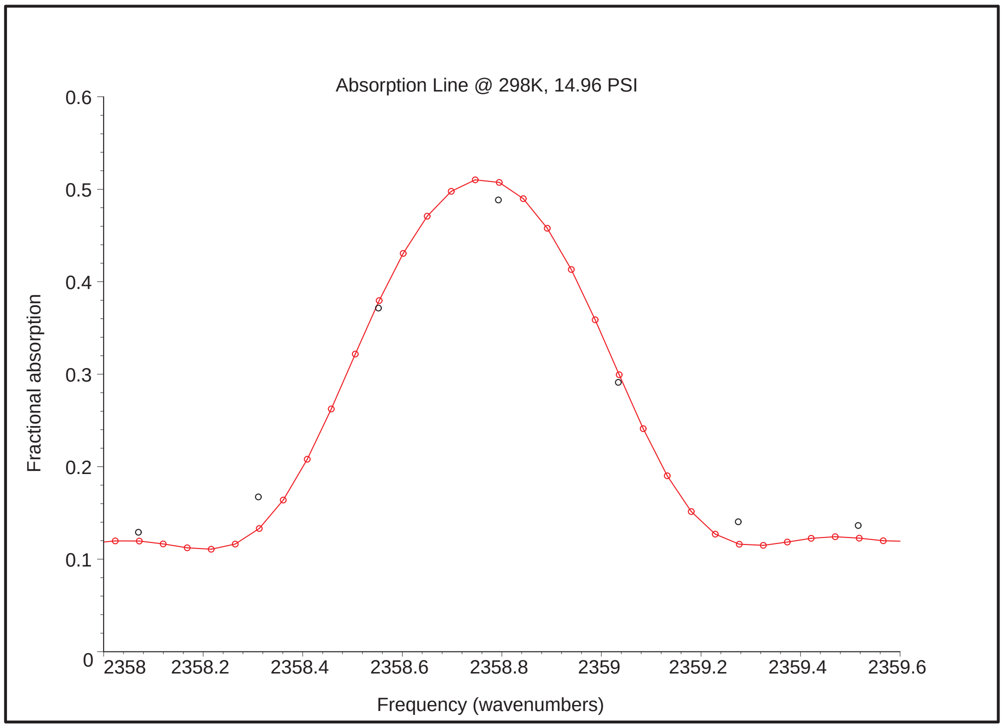

Figure 7: A single absorption line in both theory and experiment. Notice that every fifth theoretical point (hollow red circles and line) corresponds to an experimental one (hollow black circles).

The normalized experimental spectrum acquired at 773.5 Torr was sequentially compared to each spectrum in the library. A "goodness of fit," labeled $\chi^2$, was determined for each library entry by summing the squared-difference in absorption in the library entry and the normalized experimental data file at each of the common frequencies between 2237.78 cm$^{-1}$ and 2462.45 cm$^{-1}$. The temperature and pressure used to generate the library entry that minimized $\chi^2$ were alleged to be the temperature and pressure at which the experimental spectrum was obtained.

Initial fitting of the 773.5 Torr experimental spectrum to the library over-predicted the pressure and temperature, selecting the entry generated using a pressure of 15.4 PSI (= 796.4 Torr) and 318 K. To compensate, we adjusted the atmospheric concentration of carbon dioxide to a mole-fraction of 0.000393 from .00036, as was reported by Kraushaar and Ristinen for the observatory on Mauna Loa, Hawaii. Justification for this adjustment is provided, in part, by the above authors, who report an annual increase in the concentration of $CO_2$ of ∼1.4ppm (0.4%), as well as seasonal variations of up to 12 ppm [24]. Furthermore, research by Nasrallah, *et al.*, shows slight daily fluctuations (∼ 3ppm) and geographical variations in the atmospheric carbon dioxide concentration, with measurements of up to 650 ppm for urban Phoenix, Arizona [25, 26]. The corrected concentration was selected such that the entry with the minimum $\chi^2$ was generated with the proper pressure, 15.0 PSI (=775.3 Torr).

Using the adjusted carbon dioxide concentration, we analyzed the remaining experimental spectra. As shown in table 1, the pressure calculated using FT-IR spectroscopy agrees with the pressure gauge measurement for pressures between 1.5 PSI (= 77.6 Torr) and 15.0 PSI (=775.3 Torr). Below this point FT-IR spectroscopy no longer predicts the pressure accurately, due to errors in the **specgen** program. The frequency interval between adjacent theoretical points is large relative to the width of the low pressure lines. This problem could be resolved by increasing the sampling density, though doing so would add significant computational time to library generation. Figure 8 shows the pressure inferred from FT-IR spectroscopy versus the

| | Gauge | FT-IR | | | | |
|---|---|---|---|---|---|---|
| | Pressure | Pressure | Temperature | | | Peak |
| Order | (PSI) | (PSI) | K | $\chi^2$ | $\epsilon$ | Absorption |
| 10 | 0.01 | 0.0 | — | 0.000423 | 0.9962 | 0.0023 |
| 11 | 0.11 | 0.0 | — | 0.000828 | 0.9956 | 0.0054 |
| 1 | 0.33 | 0.0 | — | 0.008394 | 0.9992 | 0.0161 |
| 2 | 0.67 | 0.1 | 318 | 0.029322 | 0.9987 | 0.0315 |
| 3 | 1.65 | 1.5 | 320 | 0.074622 | 0.9978 | 0.0741 |
| 4 | 2.72 | 2.6 | 320 | 0.057583 | 0.9973 | 0.1208 |
| 5 | 5.44 | 5.4 | 318 | 0.028684 | 0.9959 | 0.2444 |
| 6 | 7.75 | 7.5 | 318 | 0.035354 | 0.9948 | 0.3204 |
| 7 | 9.72 | 9.4 | 318 | 0.045539 | 0.9945 | 0.3734 |
| 8 | 13.00 | 12.9 | 318 | 0.067254 | 0.9935 | 0.4540 |
| 9 | 14.96 | 15.0 | 318 | 0.076935 | 0.9928 | 0.4970 |

Table 1: Comparison of pressure gauge measurements and temperature and pressure obtained by analysis of FT-IR spectra. The measured temperature for each run was 298±0.5 K. Order gives the order in which the spectra were acquired, $\chi^2$ measures the goodness of fit, and $\epsilon$ is the normalization constant.

pressure measured via gauge.

Adjustment of the $CO_2$ concentration did not affect the system's temperature accuracy. As table 1 shows that the fitting scheme consistently selects a temperature hotter than that measured with a thermocouple, 298±0.5 K. Because no additional free parameters, like concentration, exist in our model, we are unable to normalize the theory to a reference point, as we did for pressure. The model is not sensitive to temperature for the three lowest pressures, because it selects a best-fit pressure of 0.0

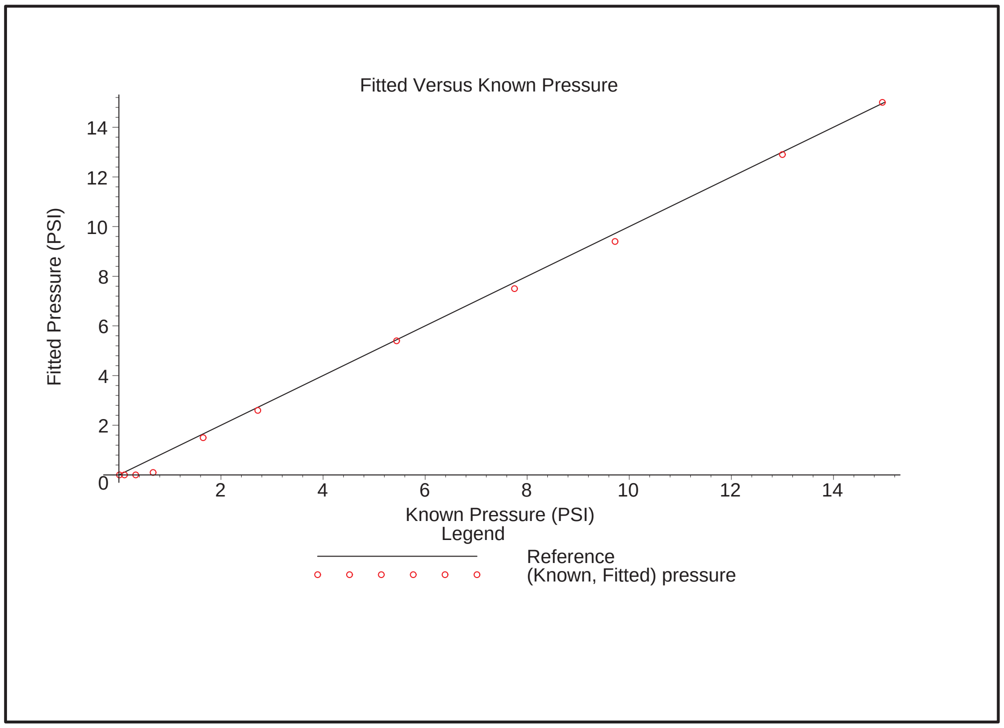

Figure 8: The fitted pressure plotted against the measured pressure.

PSI. In this situation, the theoretical spectrum shows no absorption at any frequency, regardless of the temperature used in its generation. Therefore, the model is unable to determine the temperature of the experimental spectrum.

The source of the consistent over-prediction of the fitted temperature is unknown. Visual comparison of the experimental spectra, their best-fit counterparts, and theoretical spectra generated at the known pressure and temperature reveals that the experimental spectra (and their best-fit analogues) display noticeably more absorption in the high $J$ lines than the theoretical spectra generated at the known conditions, as seen in figure 9. This phenomenon results from the greater population of the high $J$ states at higher temperatures as compared to low temperatures. Furthermore, inspection of the line intensities given by HiTran appear to support a higher sample temperature. This may suggest an error in the way **specgen** implements temperature dependence, though no specific errors have been identified.

Figure 9 shows theoretical spectra for both 298 and 318 K. These spectra were choosen to show the difference between known and fitted temperatures. Though the two spectra look similar, we notice that the lower temperature spectrum has more absorption than the higher temperature spectrum near the band center, while the opposite is true far from the center.

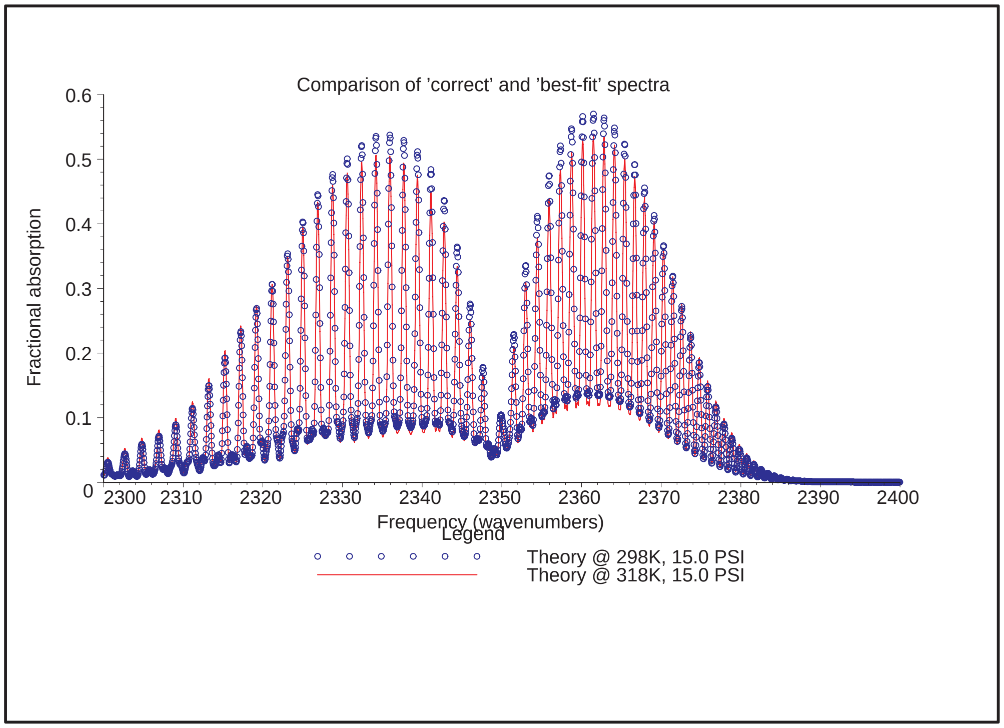

Figure 9: Absorption spectra at 15.0 PSI, 298 K and 15.0 PSI, 318 K.

Figures 10 - 13 show FT-IR absorption spectra taken over a range of pressures. Also shown in each figure is the "best-fit" theoretical spectrum that minimizes $\chi^2$ for the data. Notice that the theoretical spectra at low pressure do not agree well with the experimental ones. This results from the **specgen** program calculating the absorption at points that are spaced too widely relative to the pressure-broadened half width. This could be corrected easily, though at the expense of higher computational time.

The theoretical spectrum in figure 10 does not show the expected distribution of intensities that we expected, due to errors in our program **specgen**. This error is not due to the model we are using, but rather a poor choice of sampling density. The same effect may be seen in figure 11, though a larger number of lines are observed.

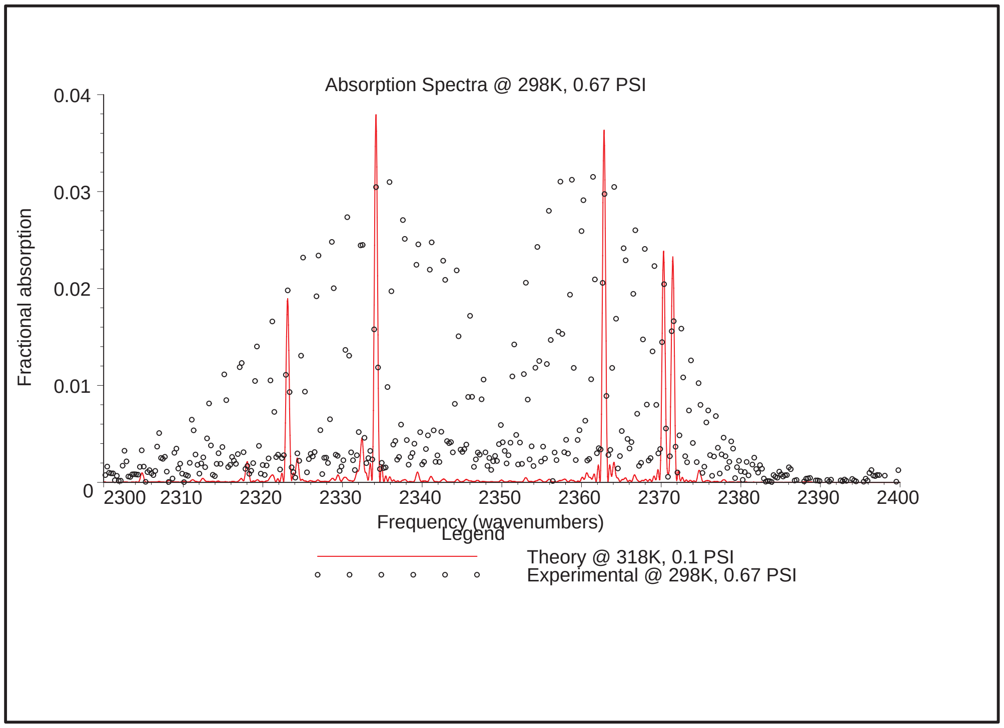

Figure 10: Absorption spectra at 0.67 PSI, 298 K.

At higher pressures the absorption lines are broader. Consequently, the frequency spacing used in **specgen** is sufficient for accurate spectra modeling, as seen in figures 12 and 13. The line intensities follow the expected Maxwell distribution in both of these cases.

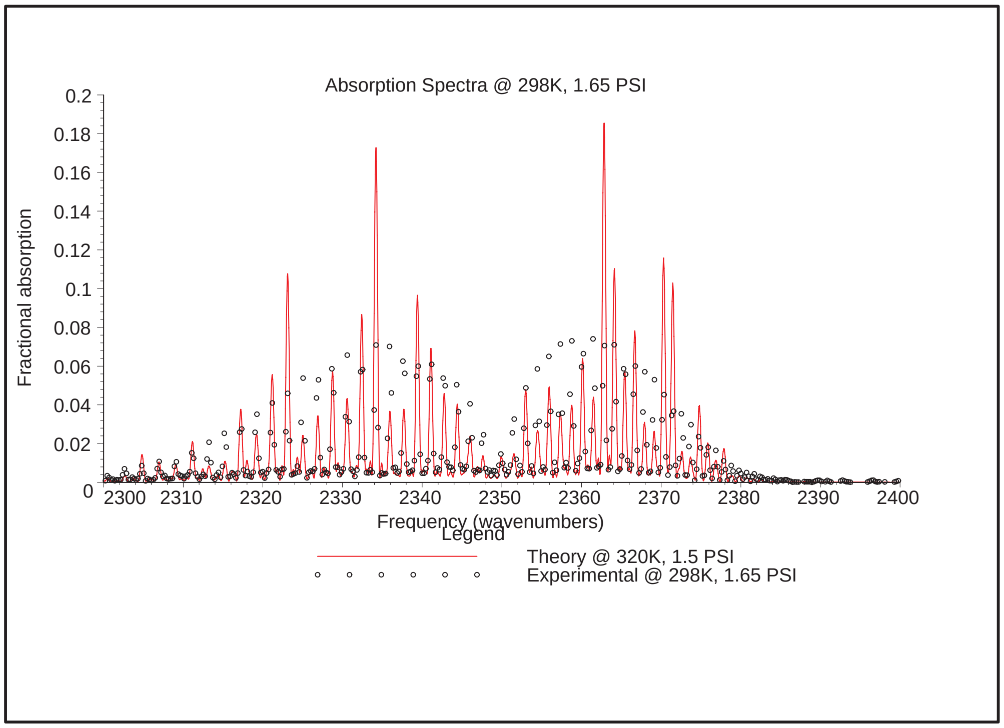

Figure 11: Absorption spectra at 1.65 PSI, 298 K.

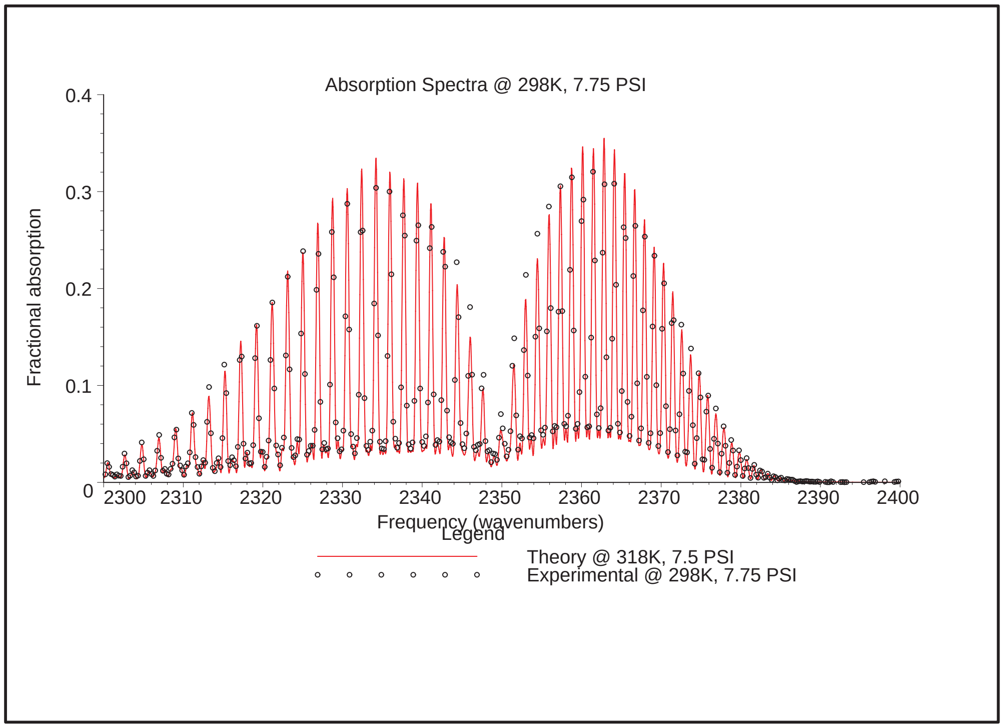

Figure 12: Absorption spectra at 7.75 PSI, 298 K.

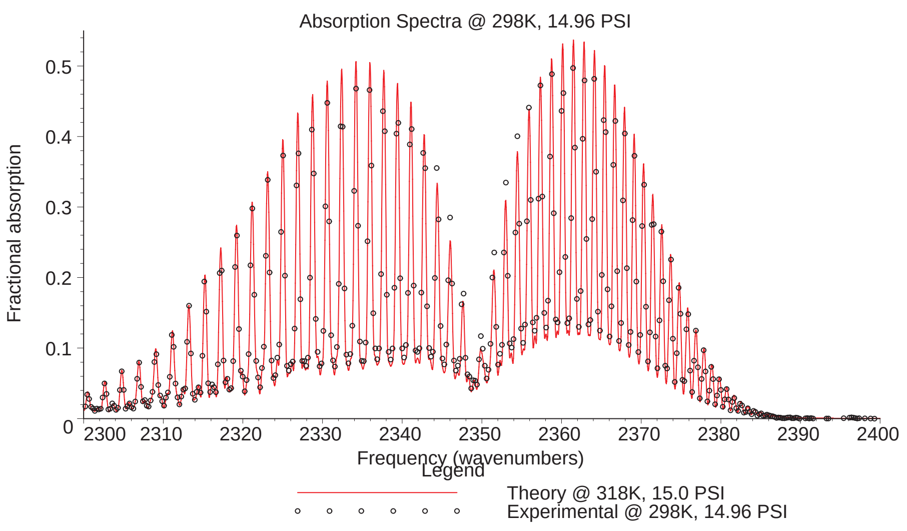

Figure 13: Absorption spectra at 14.96 PSI, 298 K.

## 5.2 Precision

Additional data, consisting of 10 single-scan spectra, were acquired to estimate measurement precision. These spectra each required three seconds to obtain, and were all taken at approximately the same pressure, 770.3 - 771.1 Torr (=14.90 - 14.91 PSI), and the same temperature, $298 \pm 0.5K$ K. These spectra were also collected at a resolution of $0.5\text{cm}^{-1}$ and with triangular apodization. The fitted pressures and temperatures are shown in table 2, along with the $\epsilon$ used for normalization and the peak absorption after normalization.

| Pressure (PSI) | Temperature K | $\epsilon$ | Peak Absorption |
|---|---|---|---|
| 17.6 | 319.0 | 0.98950 | 0.53959 |
| 17.6 | 319.5 | 0.98957 | 0.54304 |
| 17.6 | 319.5 | 0.98869 | 0.54052 |
| 17.3 | 316.5 | 0.99012 | 0.54302 |
| 17.3 | 316.5 | 0.99012 | 0.53929 |
| 17.6 | 318.5 | 0.98967 | 0.54269 |
| 17.4 | 317.0 | 0.99100 | 0.54147 |
| 17.3 | 317.0 | 0.99175 | 0.54172 |
| 17.2 | 316.5 | 0.99267 | 0.54522 |
| 17.1 | 317.0 | 0.99629 | 0.54938 |
| 0.18 | 1.21 | Standard Deviation | |

Table 2: Temperature and pressure calculated from 10 different FT-IR spectra taken at the same conditions, 14.9 PSI,2198 K.

We used this data to calculate the standard deviation for both pressure and temperature for measurements taken at $\sim$ 14.9 PSI and 298 K. It is likely that the standard deviation will be different at other conditions. Deviations of 0.18 PSI (1.2%) and 1.21 K (0.4%) were found for pressure and temperature respectively. The standard deviation in temperature is small compared to other optical techniques such as CARS and PLIF, which have reported deviations 2% and 4% respectively [27].

Two things should be noted about the data and the results used. First, each spectrum was composed of a single scan, giving a worse signal-to-noise ratio than the spectra created by averaging 32 individual scans. Secondly, we notice a systematic error in temperature of $\sim 20$ K and in pressure of $\sim 2.5$ PSI. The error in temperature is consistent with our earlier work. The error in pressure is difficult to explain. A background scan was taken immediately prior to the obtaining the spectra, eliminating effects due to detector drift. Approximately one hour passed between when these spectra were acquired and when the ones described in table 1 were taken; too little time for significant changes in the atmospheric concentration of $CO_2$.

# 6 Future Work and Conclusion

A novel use of FT-IR spectroscopy to determine the temperature and pressure in wind tunnel free streams has been considered. A program to model the absorption spectra produced by FT-IR spectroscopy has been written and validated at a range of pressures. Comparison of theoretical and experimental carbon dioxide rotational spectra of the $\nu_3$ asymmetric stretch band was used to determine temperature and pressure. Laboratory studies, conducted in a gas cell indicate that this technique yields accurate pressure measurements when the $CO_2$ concentration is such that peak absorption is greater than 3 percent. We expect this condition to be met in most LaRC facilities. In the lab, atmospheric $CO_2$ calibration was completed by forcing the algorithm to pick the correct pressure for the spectrum taken at 773.5 Torr. The correct concentration of carbon dioxide within the wind tunnel could be determined by forcing the algorithm to select the appropriate pressure for a "wind-off" spectrum. A significant systematic error of 20 Kelvins was present. The source of this error is under continued investigation, though if eliminated, this technique could also be used to make rapid, accurate, automated temperature measurements.

The main drawbacks of using FT-IR spectroscopy to make quantitative measurements are that it is time-averaged over several seconds and path-averaged across the flow. However, these disadvantages are offset by the potential of FT-IR spectroscopy to make real-time measurements of free stream conditions in a large number of facilities, without modification to the system.

Future work is needed to study system performance over a range of temperatures. Hopefully, these measurements will enable us to resolve the present systematic temperature error. Additionally, extension from a laboratory setting to the wind tunnel will require some work. Problems resulting from the incorporation of fiber optics are expected. When these obstacles are surmounted, FT-IR spectroscopy will give

LaRC the ability to make unperturbed free stream measurements necessary for better understanding of model performance, the refinement of CFD code input parameters, and as a diagnostic tool.

## A HiTran Output

| Mol | $\omega_{ul}$ | $S_{ul}$ | $\Re_{ul}$ | $\gamma_{air}^{col}$ | $\gamma_{self}^{col}$ | $\epsilon_l$ | n | $\delta$ | iv$_u$ | iv$_l$ | q$_u$ | q$_l$ | ierr | iref |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 91 | 2487.526800 | 9.074E-22 | 2.932E-04 | .0943 | .3900 | 116.8625 | .75 | .000000 | 13 | 118 | 118 19 | 119 | 002 | 8 5 1 |
| 41 | 2487.807000 | 2.555E-21 | 1.581E-04 | .0664 | .0000 | 415.4855 | .82 | .000000 | 11 | 1 |  | R 31E | 000 | 0 0 0 |
| 91 | 2488.177600 | 9.036E-22 | 2.921E-04 | .0943 | .3900 | 105.2994 | .75 | .000000 | 13 | 117 | 017 18 | 018 | 002 | 8 5 1 |
| 41 | 2488.576800 | 2.342E-21 | 1.584E-04 | .0663 | .0000 | 442.2791 | .82 | .000000 | 11 | 1 |  | R 32E | 000 | 0 0 0 |
| 162 | 2489.044050 | 1.289E-21 | 0.000E+00 | .0769 | .1344 | 166.83380 | .500 | .000000 | 2 | 1 | Q 4.5 | P 4 | 452 | 5 1 1 |

Table 3: Sample HiTran output file format.

| | | | |
|---|---|---|---|
| Mol | Molecule and isotope numbers | n | Coefficient of temperature dependence |
| $\omega_{ul}$ | Frequency | $\delta$ | Air-broadened pressure shift |
| $S_{ul}$ | Intensity | iv$_u$, iv$_l$ | Upper and lower state global quanta indexes |
| $\Re_{ul}$ | Weighted transition moment-squared | q$_u$, q$_l$ | Upper and lower state local quanta indexes |
| $\gamma_{air}^{col}$ | Air-broadened half width | ierr | Accuracy indexes |
| $\gamma_{self}^{col}$ | Self-broadened half width | iref | Indices for references |

## B specgen.cpp

```
//  * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
// *                                                                          *
//  * Specgen.cpp - Generates FT-IR spectra based on temperature, pressure    *
// *  path length, etc... selected by the user, and an output file from the    *
//  * program hitran (available at ftp://cfa-ftp.harvard.edu/pub/hitran).     *
// *  Based on method documented by L. Rothman and S.S. Penner.                *
//  * Additional effects due to doppler broadening are accounted for, giving  *
// *  a Voigt function.  Also, instrument effects are account for, with user   *
//  * selected instrument function.  Further information available in honors  *
// *  thesis:  Non-intrusive Temperature and Species Density Measurements      *
//  * using FT-IR Spectroscopy, Jason Hoffman, The College of William and     *
// *  Mary, 2003.                                                              *
//  *                                                                         *
// *  Written by: Jason D. Hoffman   -   Decemeber 23, 2002                    *
//  *                                                                         *
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *


#include <iostream>                          // for input and output

#include <fstream>                           // for file input

#include <string>                            // for strings

#include <vector>                            // for vectors


#include <iomanip>                           // for Humlik()

#include <cmath>                             // for Humlik()


#define PI 3.14159265359


using namespace std;                         // initialize global namespace


struct linedata

{

    double frequency;                        // frequency in wavenumbers
    double intensity;                        // intensity @ 296 Kelvins in wavenumbers per
                                                (molecule * wavenumbers-squared)
    double wtms;                             // weighted transition moment-squared in Debye-squared
    double abhw;                             // air-broadened halfwidth @ 296 Kelvins in wavenumbers
                                                per atmosphere
    double sbhw;                             // self-broadened halfwidth @ 296 Kelvins in wavenumbers
                                                per atmosphere
    double cotd;                             // coeffficient of temperature dependence
    double abps;                             // air-broadened pressure shift @ 296 Kelvins in
                                                wavenumbers per atmosphere
};


void readin(const char* filename, vector<linedata>& hitran);
int print_help(void);
void HUMLIK(int N, vector<long double>& X, long double Y, vector<long double>& K);
void error(const string p, const char* p2, bool quit);
void error(const string p, bool quit);


int main(int argc, char* argv[])
{
    double reference_temperature = 296.0;    // reference temperature in Kelvins for Rothman, et al
    const char* filename;                    // name of hitran data file (c-string)
    vector<linedata> hitran;                 // vector of lines read in from hitran file

    // default values for experiment
    double total_pressure_psi = 14.8875;      // total pressure in pounds per square inch
    double total_temperature_rankines = 536.85;// total temperature in Rankine
    double path_length_centimeters = 15.094; // path length in centimeters
    double resolution_wavenumbers = 1.000;   // resolution in wavenumbers
    double mach_number = 0.0;                // mach number of flow
    // default properties for gas
    double gamma = 1.4;                      // ratio of specific heats
    double specific_gas_constant = 287.0;    // specific gas constant in Joules / (kilogram * Kelvin)
    double molecular_mass_environment = 28.84;// molecular mass of environemnt in grams per gram-mole

    double molecular_mass_gas = 43.98983;     // molecular mass of gas in grams per gram-mole

    double mole_fraction = .000393;            // mole fraction


    switch(argc)

    {

    case 1: // get all information from user ** implement this section as a do-while menu, that it

               uses when the wrong comamnd line arguments given

    {

string userfilename;                  // user entered filename

cout << "Please enter the name of the HiTran input file: ";

cin >> userfilename;                  // user enters name of hitran file

filename = userfilename.c_str();      // it is then converted to c-string

cout << endl << "Please enter the total pressure in pounds per square inch: ";

cin >> total_pressure_psi;            // user enters total pressure in atmospheres

cout << endl << "Please enter the total temperature in degress Rankine: ";

cin >> total_temperature_rankines;    // user enters total temperature in Rankine

cout << endl << "Please enter the path length in centimeters: ";

cin >> path_length_centimeters;       // user enters path length in centimeters

cout << endl << "Please enter the resolution in wavenumbers: ";

cin >> resolution_wavenumbers;        // user enters resolution in wavenumbers

cout << endl << "Please enter the mach number of the flow: ";

cin >> mach_number;                   // user enters mach number

cout << endl << "Please enter gamma, the ratio of specific heats: ";

cin >> gamma;                         // user enters gamma, ratio of specific heats

cout << endl << "Please enter the specific gas constant in Joules / (kilogram * Kelvin): ";

cin >> specific_gas_constant;         // user enters specific gas constant in Joules

                                              per (kilogram * Kelvin)

cout << endl << "Please enter the molecular mass of environment in grams per gram-mole: ";

cin >> molecular_mass_environment;    // user enters molecular mass in grams per gram-mole

cout << endl << "Please enter the molecular mass of gas in grams per gram-mole: ";

cin >> molecular_mass_gas;            // user enters molecular mass in grams per gram-mole

cout << endl << "Please enter the mole fraction: ";
```

```
cin >> mole_fraction;                 // user enters mole fraction

break;

    }

    case 2: // print help message, or use defaults

    {

// print help message

if( ((argv[1])[0] == '-') && ((argv[1])[1] == 'h') )

{

    print_help();

    exit(-1);

} // end if


// use default data values

if( ((argv[1])[0] == '-') && ((argv[1])[1] == 'd') )

{

    // use a defautlt filename and convert to c-string

    string defaultfilename = "default.out";

    filename = defaultfilename.c_str();

}

else

    error("invalid command line argument", true);

break;

    }

    case 7: // read in half from command line, half from user

    {

filename = argv[1];                   // filename is read from command line

total_pressure_psi

    = (double)atof(argv[2]);          // total pressure is read from command line

total_temperature_rankines

    = (double)atof(argv[3]);          // total temperature is read from command line

path_length_centimeters

    = (double)atof(argv[4]);          // path length is read from command line

resolution_wavenumbers
```

```
    = (double)atof(argv[5]);          // resolution is read from command line

mach_number

    = (double)atof(argv[6]);          // Mach number is read from command line


cout << endl << "Please enter gamma, the ratio of specific heats: ";

cin >> gamma;                         // user enters gamma, ratio of specific heats

cout << endl << "Please enter the specific gas constant in Joules / (kilogram * Kelvin): ";

cin >> specific_gas_constant;         // user enters specific gas constant in

                                                Joules per (kilogram * Kelvin)

cout << endl << "Please enter the molecular mass of environment in grams per gram-mole: ";

cin >> molecular_mass_environment;    // user enters molecular mass in grams per gram-mole

cout << endl << "Please enter the molecular mass of gas in grams per gram-mole: ";

cin >> molecular_mass_gas;            // user enters molecular mass in grams per gram-mole

cout << endl << "Please enter the mole fraction: ";

cin >> mole_fraction;                 // user enters mole fraction

break;

    }

    case 8: // read in half from command line, half from defaults

    {

if( ((argv[1])[0] == '-') && ((argv[1])[1] == 'd') )

{

    filename = argv[2];               // filename is read from command line

    total_pressure_psi

= (double)atof(argv[3]);      // total pressure is read from command line

    total_temperature_rankines

= (double)atof(argv[4]);      // total temperature is read from command line

    path_length_centimeters

= (double)atof(argv[5]);      // path length is read from command line

    resolution_wavenumbers

= (double)atof(argv[6]);      // resolution is read from command line

    mach_number

= (double)atof(argv[7]);      // Mach number is read from command line

} // end if
```

```
else
    error("invalid command line argument", true);
break;
    }
    case 12: // read in everything from the command line
    {
filename = argv[1];                   // filename is read from command line
total_pressure_psi
    = (double)atof(argv[2]);          // total pressure is read from command line
total_temperature_rankines
    = (double)atof(argv[3]);          // total temperature is read from command line
path_length_centimeters
    = (double)atof(argv[4]);          // path length is read from command line
resolution_wavenumbers
    = (double)atof(argv[5]);          // resolution is read from command line
mach_number
    = (double)atof(argv[6]);          // Mach number is read from command line
gamma
    = (double)atof(argv[7]);          // gamma is read from command line
specific_gas_constant
    = (double)atof(argv[8]);          // gas constant is read from command line
molecular_mass_environment
    = (double)atof(argv[9]);          // molecular mass of environtment is read from command line
molecular_mass_gas
    = (double)atof(argv[10]);         // molecular mass of gas is read from command line
mole_fraction
    = (double)atof(argv[11]);         // mole fraction is read from command line
break;
    }
    default:
error("invalid number of command line arguments", true);
    } // end switch(argc)
    readin(filename, hitran);                 // read in the data from hitran file
    int numlines = hitran.size();             // number of lines read in from hitran file

    if(numlines == 0)
error("no input read in from file,", filename, true);

    // convert total pressure and temperature into standard units
    double total_pressure_pascals = total_pressure_psi * 6894.75729317;
    double total_temperature_kelvins = total_temperature_rankines / 1.8;

    // find the total density in kilograms per cubic meter using ideal gas law
    double total_density =
total_pressure_pascals / (specific_gas_constant * total_temperature_kelvins);

    // convert total pressure, temperature, and density to static pressure, temperature, and density
    double static_temperature_kelvins =
total_temperature_kelvins / ( 1 + 0.5 * (mach_number * mach_number) * (gamma - 1) );
    double static_pressure_pascals = total_pressure_pascals /
pow( ( 1 + 0.5 * (mach_number * mach_number) * (gamma - 1) ) , (gamma / (gamma - 1)) );
    double static_density = total_density /
pow( ( 1 + 0.5 * (mach_number * mach_number) * (gamma - 1) ) , (1 / (gamma - 1)) );

    // convert static pressure to atmospheres for consistency with Rothman, et al.
    double static_pressure_atmospheres = static_pressure_pascals / 101325.0;

    // calculate the number density of particles in particles per cubic meter
    double number_density = static_density * 1000 * 6.0221367 * pow(10.0, 23.0)
                            / molecular_mass_environment;

    // calculate the concentration of specific gas in particles per cubic meter
    double concentration = number_density * mole_fraction;

    // ** note, if number_of_frequencies is too small, numerical integration will not yield area ~1.0
    int number_of_frequencies = 10000;        // the number of frequency points to be evaluated

    // define variables needed to set up the vector of frequency points to be evaluated

    long double freq_range = hitran[numlines-1].frequency - hitran[0].frequency;

    // handle poor resolution correctly

    if(freq_range < 8 * resolution_wavenumbers)

  freq_range = 8 * resolution_wavenumbers;

    long double freq_delta = 2.0 * freq_range / (double)number_of_frequencies;

    long double freq_current = hitran[0].frequency - 0.5 * freq_range;


    vector<long double> freq;                 // vector of frequency points to be evaluated

    // set up the vector of frequency points to be evaluated

    freq.push_back(freq_current);

    for(int i = 1; i < number_of_frequencies; i++)

freq.push_back(freq_current += freq_delta);


    // define variables regarding instrumental lineshape

    long double retardation_centimeters = 1 / resolution_wavenumbers;

    long double instrument_lineshape_function = 0.0;

    vector<long double> ilsf;                 // instrument lineshape function


    freq_current = (-number_of_frequencies / 8.0) * freq_delta;


    // calculate the instrument function

    // currently uses a sinc-squared function - need to change to user choice

    for(int i = 0; i <= number_of_frequencies / 4; i++)

    {

// calculate the instrument lineshape function

instrument_lineshape_function = retardation_centimeters *

    (pow( (sin(PI * freq_current * retardation_centimeters)) , 2 )) /

    (pow( (PI * freq_current * retardation_centimeters), 2 ));


// add the instrument lineshape function to vector

ilsf.push_back(instrument_lineshape_function);
```

```
freq_current += freq_delta;           // update the evaluation frequency

    } // end for(i)


    // vectors to store information on environmental lineshape

    vector<long double> pblh;                 // vector of pressure broadened line halfhwidths

    vector<long double> dblh;                 // vector of doppler broadened line halfhwidths

    vector<long double> pstf;                 // vector of pressure shifted transition frequencies

    vector<long double> voigty;               // vector of voigt y parameters (Wells)

    vector<long double> voigtx;               // vector of voigt x parameters for a single transition

    vector<long double> voigtk;               // vector of voigt function values returned from Humlik

    vector<long double> nlsf;                 // vector of normalized lineshape functions (Weisstein)


    vector<long double> conv;                 // vector of convolved voigt and instrument functions

    vector<long double> mac;                  // vector of monochromatic absorption coefficients

    vector<long double> tmac;                 // vector total monochromatic absorption coefficients


    // define variables regarding environmental lineshape

    long double pressure_broadened_line_halfwidth = 0.0;

    long double doppler_broadened_line_halfwidth = 0.0;

    long double pressure_shifted_transition_frequency = 0.0;

    long double voigt_x = 0.0;                // the voigt x parameter (Wells)

    long double voigt_y = 0.0;                // the voigt y parameter (Wells)

    long double normalized_lineshape_function = 0.0;


    long double preconvolved_area = 0.0;      // the area of the line before convolution

    long double postconvolved_area = 0.0;     // the area of the line after convolution


    long double convolution = 0.0;            // value of convolution

    long double monochromatic_absorption_coefficient = 0.0;


    for(int i = 0; i < numlines; i++)

    {
```

```
// calculate the pressure shifted transition frequency

pressure_shifted_transition_frequency = hitran[i].frequency +

    hitran[i].abps * static_pressure_atmospheres;


// add pressure shifted transition frequency to vector

pstf.push_back(pressure_shifted_transition_frequency);


// calculate the pressure_broadened_line_halfwidth (Rothman eqn. A12)

pressure_broadened_line_halfwidth =

    pow( (reference_temperature / static_temperature_kelvins), hitran[i].cotd) *

    static_pressure_atmospheres * ( (1.0 - mole_fraction) * hitran[i].abhw +

    mole_fraction * hitran[i].sbhw );


// calculate the doppler_broadened_line_halfwidth (Penner eqn. 3.30)

doppler_broadened_line_halfwidth = 3.5811735 * pow(10.0, -7.0) * pstf[i] *

    sqrt(static_temperature_kelvins / molecular_mass_gas);


// add pressure and doppler broadened line halfwidths to vectors

pblh.push_back(pressure_broadened_line_halfwidth);

dblh.push_back(doppler_broadened_line_halfwidth);


// calculate the voigt y parameter

voigt_y = sqrt(log(2.0)) * pressure_broadened_line_halfwidth /

    doppler_broadened_line_halfwidth;


// add voigt y to vector;

voigty.push_back(voigt_y);


voigtx.clear();                        // clean up the old vector

for(int j = 0; j < number_of_frequencies; j++)

{

    // calculate the voigt x parameter

    voigt_x = sqrt(log(2.0)) * (freq[j] - pstf[i]) / dblh[i];
    // add the current parameter to vector

    voigtx.push_back(voigt_x);

} // end for(j)


voigtk.clear();                         // clean up the old vector

// call HUMLIK function to calculate voigt function (Wells)

HUMLIK(number_of_frequencies, voigtx, voigt_y, voigtk);


nlsf.clear();                           // clean up the old vector

// normalize the voigt function (Weisstein)

for(int j = 0; j < number_of_frequencies; j++)

{

    // calculate the normalized lineshape function

    normalized_lineshape_function = (1 / doppler_broadened_line_halfwidth) *

    sqrt( log(2.0) / PI) * voigtk[j];


    // add the normalized lineshape function to vector

    nlsf.push_back(normalized_lineshape_function);

} // end for(j)


// calculate the area of the preconvolved line (~1.0)

preconvolved_area = 0.0;                // reset the area to 0.0

for(int j = 0; j < number_of_frequencies; j++)

    preconvolved_area += freq_delta*nlsf[j];


// if the area of the line is not ~1.0, print error

if(preconvolved_area < 0.9 || preconvolved_area > 1.1)

    error("Spectral line area not normalized.  Increase number_of_frequencies!", false);


conv.clear();                           // clean up the old vector

// fill front of vector with 0.0

for(int j = 0; j < number_of_frequencies / 8; j++)

    conv.push_back(0.0);
```

```
// convolve the line with the instrument function

for(int j = number_of_frequencies / 8; j < 7 * number_of_frequencies / 8; j++)

{

    convolution = 0.0;                // reset to 0.0

    // perform discrete convolution

    for(int k = j - number_of_frequencies / 8; k < j + number_of_frequencies / 8; k++)

convolution += ( ilsf[j - k + number_of_frequencies / 8] * nlsf[k] );

    conv.push_back(convolution);

} // end for(j)


// fill end of vector with 0.0

for(int j = 0; j < number_of_frequencies / 8; j++)

    conv.push_back(0.0);


// calculate the area of the postconvolved line (~1.0?)

postconvolved_area = 0.0;              // reset the area to 0.0

for(int j = 0; j < number_of_frequencies; j++)

    postconvolved_area += freq_delta*conv[j];


// renormalize the line after convolution

for(int j = 0; j < number_of_frequencies; j++)

    conv[j] = conv[j] * preconvolved_area / postconvolved_area;


// calculate the monochromatic absorption coefficient (Rothman)

mac.clear();                           // clean up the old vector

for(int j = number_of_frequencies / 8; j < 7 * number_of_frequencies / 8; j++)

{

    monochromatic_absorption_coefficient = conv[j]*hitran[i].intensity;

    mac.push_back(monochromatic_absorption_coefficient);

}


// calculate the total monochromatic absorption coefficient
```

```
if(i == 0)                              // first time through

{

    // initialize with correct values

    for(int j = 0; j < 3 * number_of_frequencies / 4; j++)

tmac.push_back(mac[j]);

} // end if(i)

else

{

    // update the array

    for(int j = 0; j < 3 * number_of_frequencies / 4; j++)

tmac[j] += mac[j];

} // end else(i)

    } // end for(i)


    // define variables for total absorption

    vector<long double> absorb;                 // vector to hold total absorption

    long double total_absorption = 0.0;       // the total absorption at a frequency


    for(int j = 0; j < 3 * number_of_frequencies / 4; j++)

    {

total_absorption= 1.0-exp(-1.0*tmac[j]*concentration*0.000001*path_length_centimeters);

absorb.push_back(total_absorption);

    }


    cout.precision(10);

    for(int j = 0; j < 3 * number_of_frequencies / 4; j++)

cout << freq[j + number_of_frequencies / 8] << "  " << absorb[j] << endl;


    return 0;

} // end int main()


//  *    read in the data from hitran file      *

void readin(const char* filename, vector<linedata>& hitran)
```

```
{

    string line;                              // line read in from file to be parsed

    ifstream input(filename);                 // open hitran data file

    if (!input)                               // exit with error if can't open file

error("cannot open input file", filename, true);


    linedata current;                         // struct for current line data


    while(input)

    {

getline(input, line);                 // read in the line to be parsed

if(input)                             // handle last line correctly

{

    //  read in data from file and store it in growing vector

    current.frequency = atof(line.substr(3,12).c_str());

    current.intensity = atof(line.substr(15,10).c_str());

    current.wtms = atof(line.substr(25,10).c_str());

    current.abhw = atof(line.substr(35,5).c_str());

    current.sbhw = atof(line.substr(40,5).c_str());

    current.cotd = atof(line.substr(55,4).c_str());

    current.abps = atof(line.substr(59,8).c_str());


    hitran.push_back(current);        // push the current data onto vector

} // end if(input)

    } // end while(input)

} // end void readin()


//  *    print the help message with options    *

int print_help(void)

{

    cout << "Usage:  specgen" << endl;

    cout << "    --  allows user to enter all data" << endl << endl;
```

```
   cout << "Usage:  specgen -h" << endl;

   cout << "    --  get help" << endl << endl;


   cout << "Usage:  specgen -d" << endl;

   cout << "    --  use all default values" << endl << endl;


   cout << "Usage:  specgen filename, pressure, temperature, path length,"

<< endl << "        resolution, mach number" << endl;

   cout << "    --  reads filename, total pressure in pounds per square inch, total temperature

     in degrees Rankine,"

     << endl << "        path length in centimeters, resolution in wavenumbers, and Mach

     number from command line"

<< endl << "    --  all other values are entered by user" << endl << endl;


   cout << "Usage:  specgen -d filename, pressure, temperature, path length"

<< endl << "        resolution,  mach number" << endl;

   cout << "    --  reads filename, total pressure in pounds per square inch, total temperature

     in degress Rankine,"

<< endl << "        path length in centimeters, resolution in wavenumbers and Mach

     number from command line."

<< endl << "    --  all other values assume default value" << endl << endl;


   cout << "Usage:  specgen filename, pressure, temperature, path length, resolution,

         mach number,"

<< endl << "        gas constant, gamma, molecular mass of environement,

        molecular mass of gas, mole fraction" << endl;

   cout << "    --  reads filename, total pressure in pounds per square inch,

        total temperature in degrees Rankine,"

<< endl << "        path length in centimeters, resolution in wavenumbers,

        mach number,"

<< endl << "        gamma = ratio of specific heats, gas constant in Joules

        per (kilogram * Kelvin),"

<< endl << "        molecular mass of environment in grams per gram-mole, molecular
```

```
            mass of gas in grams per gram-mole,"
 << endl << "        and mole fraction from command line." << endl;
    return 0;
} // end print_help


// ******************************************************************************
//  This code is based on Fortan code published by R.J. Wells
//  "Rapid Approximation to the Voigt/Faddeeva Function and its Derivatives"
//  Journal of Quantitative Spectroscopy and Radiative Transfer 62.
//
//  Translation by: Jason Hoffman - August 2002
//
//  Updated to support vectors instead of arrays for compatability
//
// ******************************************************************************


//       To calculate the Faddeeva function with relative error less than 10^(-R).
//       R0=1.51*EXP(1.144*R) and R1=1.60*EXP(0.554*R) can be set by the the user
//       subject to the constraints 14.88<R0<460.4 and 4.85<R1<25.5


// Global Constants
long double RRTPI = 0.56418958;                                        // 1/SQRT(pi)
long double Y0 = 1.5;
long double Y0PY0 = Y0+Y0;
long double Y0Q = Y0*Y0;                                               // for CPF12 algorithm


long double C[6] = { 1.0117281, -0.75197147,  0.012557727,
                     0.010022008, -0.00024206814,   0.00000050084806 };
long double S[6] = { 1.393237,   0.23115241, -0.15535147,
                     0.0062183662, 0.000091908299, -0.00000062752596 };
long double T[6] = { 0.31424038, 0.94778839,  1.5976826,
                     2.2795071,    3.0206370,       3.8897249 };
```

```
void HUMLIK(int N, vector<long double>& X, long double Y, vector<long double>& K)

{

  long double R0 = 146.7;                                              // Region boundaries

  long double R1 = 14.67;                                              // for R=4


  // Local variables

  int RG1 = 0;

  int RG2 = 0;

  int RG3 = 0;                                                         // y polynomial flags


  long double ABX = 0;                                                 // |x|

  long double XQ = 0;                                                  // x^2

  long double YQ = 0;                                                  // y^2

  long double YRRTPI = 0;                                              // y/SQRT(pi)


  long double XLIM0 = 0;

  long double XLIM1 = 0;

  long double XLIM2 = 0;

  long double XLIM3 = 0;

  long double XLIM4 = 0;                                               // |x| on region boundaries


  long double A0 = 0;

  long double D0 = 0;

  long double D2 = 0;

  long double E0 = 0;

  long double E2 = 0;

  long double E4 = 0;

  long double H0 = 0;

  long double H2 = 0;

  long double H4 = 0;

  long double H6 = 0;                                                  // W4 temporary variables


  long double P0 = 0;
  long double P2 = 0;

  long double P4 = 0;

  long double P6 = 0;

  long double P8 = 0;

  long double Z0 = 0;

  long double Z2 = 0;

  long double Z4 = 0;

  long double Z6 = 0;

  long double Z8 = 0;


  long double XP[6], XM[6], YP[6], YM[6];                         // CPF12 temporary values

  long double MQ[6], PQ[6], MF[6], PF[6];


  long double D, YF, YPY0, YPY0Q;


  // ***** Start of executable code *****************************************


  RG1 = 1;                                                        // Set flags

  RG2 = 1;

  RG3 = 1;

  YQ  = Y*Y;                                                      // y^2

  YRRTPI = Y*RRTPI;                                               // y/SQRT(pi)


  // Region boundaries when both K and L are required or when R<>4

  XLIM0 = R0 - Y;

  XLIM1 = R1 - Y;

  XLIM3 = 3.097*Y - 0.45;


  XLIM2 = 6.8 - Y;

  XLIM4 = 18.1*Y + 1.65;

  if ( Y <= 0.000001 )                                            // When y<10^-6

    {

      XLIM1 = XLIM0;                                              // avoid W4 algorithm
```

```
      XLIM2 = XLIM0;

    } // end if


  for(int i = 0; i < N; i++)                                                    // Loop over all points

    {

      ABX = fabs( X[i] );                                                       // |x|

      XQ  = ABX*ABX;                                                            // x^2

      if ( ABX > XLIM0 )                                                        // Region 0 algorithm

  K.push_back(YRRTPI / (XQ + YQ));

      else

{

  if ( ABX > XLIM1 )                                                    // Humlicek W4 Region 1

    {

      if ( RG1 != 0 )                                                   // First point in Region 1

{

  RG1 = 0;

  A0 = YQ + 0.5;                                                // Region 1 y-dependents

  D0 = A0*A0;

  D2 = YQ + YQ - 1.0;

} // end if RG1 != 0

      D = RRTPI / (D0 + XQ*(D2 + XQ));

      K.push_back(D*Y*(A0 + XQ));

    } // end if ABX > XLIM


  else

    {

      if ( ABX > XLIM2 )                                                // Humlicek W4 Region 2

{

  if ( RG2 != 0 )                                               // First point in Region 2

    {

      RG2 = 0;

      H0 =  0.5625 + YQ*(4.5 + YQ*(10.5 + YQ*(6.0 + YQ)));            // Region 2 y-dependents

      H2 = -4.5    + YQ*(9.0 + YQ*( 6.0 + YQ* 4.0));
      H4 = 10.5    - YQ*(6.0 - YQ*  6.0);

      H6 = -6.0    + YQ* 4.0;

      E0 =  1.875  + YQ*(8.25 + YQ*(5.5 + YQ));

      E2 =  5.25   + YQ*(1.0  + YQ* 3.0);

      E4 =  0.75*H6;

    } // end if RG2 != 0

  D = RRTPI / (H0 + XQ*(H2 + XQ*(H4 + XQ*(H6 + XQ))));

  K.push_back(D*Y*(E0 + XQ*(E2 + XQ*(E4 + XQ))));

} // end if ABX > XLIM2


      else

{

  if ( ABX < XLIM3 )                                       // Humlicek W4 Region 3

    {

      if ( RG3 != 0 )                                      // First point in Region 3

{

  RG3 = 0;

// Region 3 y-dependents

  Z0 = 272.1014 + Y*(1280.829 + Y*(2802.870 + Y*(3764.966 + Y*(3447.629 +

Y*(2256.981 + Y*(1074.409 + Y*(369.1989 + Y*(88.26741 + Y*(13.39880 + Y)))))))));

  Z2 = 211.678 + Y*(902.3066 + Y*(1758.336 + Y*(2037.310 + Y*(1549.675 +

Y*(793.4273 + Y*(266.2987 + Y*(53.59518 + Y*5.0)))))));

  Z4 = 78.86585 + Y*(308.1852 + Y*(497.3014 + Y*(479.2576 + Y*(269.2916 + Y*(80.39278 + Y*10.0)))));

  Z6 = 22.03523 + Y*(55.02933 + Y*(92.75679 + Y*(53.59518 + Y*10.0)));

  Z8 = 1.496460 + Y*(13.39880 + Y*5.0);

  P0 = 153.5168 + Y*(549.3954 + Y*(919.4955 + Y*(946.8970 + Y*(662.8097 +

Y*(328.2151 + Y*(115.3772 + Y*(27.93941 + Y*(4.264678 + Y*0.3183291))))))));

  P2 = -34.16955    + Y*(-1.322256+ Y*(124.5975 + Y*(189.7730 + Y*(139.4665 +

Y*(56.81652 + Y*(12.79458 + Y*1.2733163))))));

  P4 = 2.584042     + Y*(10.46332 + Y*(24.01655 + Y*(29.81482 + Y*(12.79568 + Y*1.9099744))));

  P6 = -0.07272979  + Y*(0.9377051+ Y*(4.266322 + Y*1.273316));

  P8 = 0.0005480304 + Y*0.3183291;

} // end if
```

```
      D = 1.7724538 / (Z0 + XQ*(Z2 + XQ*(Z4 + XQ*(Z6 + XQ*(Z8+XQ)))));

      K.push_back(D*(P0 + XQ*(P2 + XQ*(P4 + XQ*(P6 + XQ*P8)))));

    } // end if ABX < XLIM3


  else                                                          // Humlicek CPF12 algorithm

    {

      YPY0 = Y + Y0;

      YPY0Q = YPY0*YPY0;

      K.push_back(0.0);

      for(int j = 0; j < 6; j++)

{

  D = X[i] - T[j];

  MQ[j] = D*D;

  MF[j] = 1.0 / (MQ[j] + YPY0Q);

  XM[j] = MF[j]*D;

  YM[j] = MF[j]*YPY0;

  D = X[i] + T[j];

  PQ[j] = D*D;

  PF[j] = 1.0 / (PQ[j] + YPY0Q);

  XP[j] = PF[j]*D;

  YP[j] = PF[j]*YPY0;

} // end for j


      if ( ABX < XLIM4 )                                        // Humlicek CPF12 Region I

{

  for(int j = 0; j < 6; j++)

    K[i] = K[i] + C[j]*(YM[j]+YP[j]) - S[j]*(XM[j]-XP[j]);

} // end if ABX < XLIM4


      else                                                      // Humlicek CPF12 Region II

{

  YF = Y + Y0PY0;

  for(int j = 0; j < 6; j++)
```

```
    K[i] = K[i] + (C[j]*(MQ[j]*MF[j]-Y0*YM[j]) + S[j]*YF*XM[j]) /
(MQ[j]+Y0Q) + (C[j]*(PQ[j]*PF[j]-Y0*YP[j]) - S[j]*YF*XP[j]) / (PQ[j]+Y0Q);
  K[i] = Y*K[i] + exp( -XQ );
} // end else
    } // end else
} // end else
    } // end else
} // end else
//      cout << "here " << K[i] << endl;
    } // end for
} // end humlik


void error(const string p, const char* p2, bool quit)
{
    cerr << p << ' ' << p2 << endl;
    if(quit)
exit(1);
}


void error(const string p, bool quit)
{
    cerr << p << endl;
    if(quit)
exit(1);
}
```

## References

[1] L. Rothman, *et al.*, "The HiTran Molecular Spectroscopic Database and HAWKS (HiTran Atmospheric Workstation): 1996 Edition," J. Quant. Spectros. & Radiat. Transfer. **60**, 665 (1998).

[2] S. Davis, *et al. Fourier Transform Spectroscopy.* Academic Press, Inc., New York, 2001.

[3] G. Herzberg. *Molecular Spectra and Molecular Structure II.* D. Van Nostrand Co. Inc., New York, 1945.

[4] V. Weisskopf, E. Wigner, "Berechnung der natürlichen Linienbreite auf Grund der Diracschen Lichttheorie," Z. Phys. **63**, 54 (1930).

[5] F. Hoyt, "The Structure of Emission Lines," Phys. Rev. **36**, 860 (1930).

[6] H. Margenau, W. W. Watson, "Pressure Effects on Spectral Lines," Rev. Mod. Phys. **8**, 22 (1936).

[7] J. Lenoble. *Atmospheric Radiative Transfer.* A. Deepak Pub, Hampton, VA, 1993.

[8] S. Penner. *Quantitative Molecular Spectroscopy and Gas Emissivities.* Addison-Wesley Pub. Co. Inc., Reading, MA, 1959.

[9] W. Silfvast. *Laser Fundamentals.* Cambridge U.P., UK, 1996.

[10] J. Van Vleck, V. Weisskopf, "On the Shape of Collision-Broadened Lines," Rev. Mod. Phys. **17**, 227 (1945).

[11] A. Ben-Reuven, "Transition from Resonant to Nonresonant Line Shape in Microwave Absorption," Phys. Rev. Lett. **14**, 349 (1965).

[12] A. Ben-Reuven, "Impact Broadening of Microwave Spectra," Phys. Rev. **145**, 7 (1966).

[13] A. Ben-Reuven, “The Meaning of Collisional Broadening of Spectral Lines: The Classical-Oscillator Analog,” Adv. At. Mol. Phys. **5**, 201 (1969).

[14] J. Waters, “Absorption and Emission by Atmospheric Gases,” In *Methods of Experimental Physics*, **12B** (M.L. Meeks, Ed.). Academic Press, New York, 1976.

[15] M. Harmony. *Introduction to Molecular Energies and Spectra.* Holt, Rinehart and Winston, Inc., New York, 1972.

[16] P. Jansson. *Deconvolution: With Applications to Spectroscopy.* Academic Press, Inc., New York, 1984.

[17] B. Armstrong, “Spectrum Line Profiles: the Voigt Function,” J. Quant. Spectros. & Radiat. Transfer. **7**, 62 (1967).

[18] V. Faddeyeva, N. Terent’ev. *Tables of Values of the Function* $w(z) = e^{-z^2}\left(1 + \frac{2\imath}{\sqrt{\pi}} \int_0^z e^{t^2} dt\right)$ *for Complex Argument.* Pergamon Press, New York, 1961.

[19] A. Asfaw, “A Fast Method of Modeling Spectral Lines,” J. Quant. Spectros. & Radiat. Transfer. **70**, 129 (2001).

[20] J. Anderson. *Modern Compressible Flow.* McGraw Hill, New York, 2003.

[21] R. Wells, “Rapid Approximation to the Voigt/Faddeeva Function and its Derivatives,” J. Quant. Spectros. & Radiat. Transfer. **62**, 29 (1999).

[22] J. Kauppinen, J. Partanen. *Fourier Transforms in Spectroscopy.* Wiley-VCH, New York, 2001.

[23] P. Griffiths, J. de Haseth. *Fourier Transform Infrared Spectroscopy.* John Wiley & Sons, Inc., New York, 1986.

[24] J. Kraushaar, R. Ristinen. *Energy and Problems of a Technical Society.* John Wiley & Sons, Inc., New York, 1993.

[25] H. Nasrallah, *et al.*, "Temporal Variations in Atmospheric $CO_2$ Concentrations in Kuwait City, Kuwait with Comparisons to Phoenix, Arizona, USA." Environmental Pollution, **121**, 301 (2003).

[26] C. Idso, *et al.*, "An Intensive Two-week Study of an Urban $CO_2$ Dome in Phoenix, Arizona, USA." Atmospheric Environment, **6**, 995 (2001).

[27] A. Eckbreth, *Laser Diagnostics for Combustion Temperature and Species.* Gordon and Breach Publishers, New York, 1996.