Portland State University
PDXScholar

University Honors Theses | University Honors College

2015

Towards Graphene Remote Plasma Interfacial Growth and Auger Spectra Simulation

Alex Jacobson
Portland State University

Follow this and additional works at: https://pdxscholar.library.pdx.edu/honorstheses
Let us know how access to this document benefits you.

Recommended Citation
Jacobson, Alex, "Towards Graphene Remote Plasma Interfacial Growth and Auger Spectra Simulation" (2015). *University Honors Theses.* Paper 194.
https://doi.org/10.15760/honors.196

This Thesis is brought to you for free and open access. It has been accepted for inclusion in University Honors Theses by an authorized administrator of PDXScholar. Please contact us if we can make this document more accessible: pdxscholar@pdx.edu.

Towards Graphene Remote Plasma Interfacial Growth and Auger Spectra Simulation

by

Alex Jacobson

An undergraduate honors thesis submitted in partial fulfillment of the

requirements for the degree of

Bachelor of Science

in

University Honors

and

Physics

Thesis Adviser

Jun Jiao

Portland State University
2015

Abstract

Graphene, the single layer of carbon atoms which form a honeycomb lattice, has excellent physical properties that make it an well known material for fundamental and applied research. A surface sensitive technique known as Auger electron spectroscopy is ideal for studying graphene due to the thickness of the graphene. A complete understanding of how the various peak in the graphene Auger spectra occur is not yet available. Towards explaining these peaks, a simulation which predicts the Auger energies of neutral free carbon has been scripted. Growth of graphene, which is mostly likely to see the greatest implementation in industry occurs in chemical vapor deposition systems. In order to avoid time-consuming and potentially damaging transfer processes, a type of graphene growth which occurs at the interface of a nickel catalyst and the underlying substrate which requires no transfer is experimented with. Here, growths have occurred on a remote plasma inductively coupled plasma chemical vapor deposition system at various gas ratios of methane to hydrogen, temperatures, and plasma conditions to explore growth optimization. The growths are subsequently analyzed using Raman spectroscopy, X-Ray photoelectron spectroscopy, and Auger electron spectroscopy. In general growths without plasma tended to be more graphite-like and growths with plasma tended to be more diamond-like.

Contents

List of Figures

1. Introduction

Graphene is prevalent in the literature of physicists and material scientists for its electrical properties, strength, and thermal conductivity among other properties. Such properties are reviewed by Novoselov (2012) and continue to be expanded upon in each following year. They report a room temperature electron mobility of 2.5 x 10^5 $cm^2 V^{-1}s^{-1}$ which is two orders of magnitude higher than that of Si (Callister 2007), a Young's modulus of 1 TPa which is five times greater than steel (Halliday 1997) and thermal conductivity of 3000 W mK^{-1} which is about 7.5 times greater than silver (Young 1992) have been observed in high quality mechanically exfoliated graphene. Graphene is a single atomic layer of carbon which forms a honeycomb lattice. Implementation of graphene in technologies is mostly in the research phase due to lack of performance quality graphene synthesis at the mass production level. Proposed developed technologies within the next twenty years include photodetectors, optical modulators, conductive inks, composite materials, solar cells, and gas sensors to name a few as reported in Novoselov (2012).

When developing a material it is necessary to receive feedback to understand what has been made, how much of the material has been made, and the quality of the material. Raman spectroscopy, X-ray photoelectron spectroscopy (XPS), and Auger electron spectroscopy (AES) are some of the methods used to receive such feedback. In Raman spectroscopy the main peaks of interest for graphene are the D, G, and G' (2D) peaks which are found near 1350 cm^{-1}, 1580 cm^{-1} and 2800 cm^{-1}, respectively. Typical figures of merit are the $I_{D/G}$ ratio for defect density, the $I_{2D/G}$ ratio for identification of single layer graphene and the peak shape of the 2D peak for determining the number of layers in few layer graphene. Specifically AES brings chemical environment information and electronic structure.

The research questions that this work seeks to address is: What process parameters will achieve the growth of graphene at a silica/Ni interface in a remote inductively coupled plasma CVD system?

Also, how can Auger spectra of graphene be modeled to arrive at electron orbital hybridization information?

1.1 The Auger Process

From an electron shell perspective, the Auger process occurs when an electron is kicked out of an atom due to an internal photoelectric effect. The inner shell of the atom loses an electron due to either X-ray or electron bombardment. Because of the instability of this state, an electron from a higher shell fills in the inner shell. The resulting excess energy is then enough to kick out an electron from an even higher shell. Auger energies are conditional on the atomic number of the element and what electron orbital from the atom the Auger electrons came from. Thus, the energies of Auger electrons can be used to identify the elements in a compound and the type of bonding that these elements have. Auger electrons are low energy electrons (100s of eV) which makes Auger spectroscopy a surface sensitive technique and thus ideal for probing 2D crystals such as graphene. A type of Auger transition, the Coster-Kronig transition, occurs when the initial vacancy due to electron bombardment is in the L shell or higher.

1.2 Description of ICP CVD System and CVD Growth Mechanism of Graphene

The inductively coupled plasma breaks down the desired precursor via ionization and, in the case of methane, by dehydrogenating the molecules. Typically, chemical vapor deposition works by flowing gases where one of the gases is the precursor, or source, of the desired deposition, an inert gas is used to dilute the precursor, and in the case of carbon growths a gas that is a catalyst for the reaction is flowed. Depending on temperature and pressure, gas types and different pressures, different compounds will form. If methane, argon, and nitrogen are flowed at low pressure and the platen is heated to a high temperature, graphene can be formed. The hydrogen is an activator of surface bound carbon necessary for monolayer growth and it controls the grain shape and dimensions by etching the weaker carbon-carbon bonds (Vlassiouk 2011). The literature reveals that many researchers have

accomplished CVD growth for graphene yet not many have done CVD growth which consists of intentionally growing the graphene at the interface between the catalysts and the underlying substrate. Most growths are intended to occur at the top of the interface. The reason that it may be more advantageous to grow graphene at the interface is to avoid transfer methods. Transfer methods take a lot of time and their effect on growth quality is not always favorable. Metal catalysts such as Ni are often used to facilitate growth nucleation sites.

2. Methods

2.1 Growth

Growths occurred in an inductively coupled plasma chemical vapor deposition system. The growths are conducted at temperatures that range from 600 ºC to 800 ºC chosen according to Kato (2012). The growths were conducted at a gas ratio of methane to hydrogen of 9:1. The gas ratio parameters were explored and gas ratios of 1:1 and of 1:5 were also explored. Growths were also conducted without plasma. Growths were conducted in an ICP CVD chamber where the inductively coupled plasma coil is in the cylindrical geometry and it is remote from the growth site. There is a shower head for a specific distribution of the gases into the main chamber. At first different gas ratios were explored based on what seemed favorable for graphene growth as reported in Kato (2012). The platen height with respect to the bottom of the chamber is 2.6 inches. The growths had a 5 minute anneal stage and a 5 minute growth stage. During the anneal stage the argon and hydrogen gases were flowed and the platen was at the temperature set point. During the growth stage methane gas was also flowed. With the plasma enhanced growths the RF power supplied was 250 W with a reflected power of 30 W for the gas series growths and 1 W for the temperature series. The reduced reflected power is due to replacement of one of the RF connectors which was arcing. The pressure during the anneal stage was on the 10^{-2} torr order of magnitude. The pressure during the growth stage was10^{-1} torr order of magnitude. The heating of the platen originates from high temperature bulbs beneath the stage. Thus the system is a cold wall CVD reactor. In order to prepare the samples for analysis, the following transfer method was employed.

Transfer method:

1. Polymethyl methacrylate (PMMA) was spin coated on the sample at 1000 rpm for 65 seconds.
2. The sample was placed in a 5:1 aqueous dilution of Transene C-100 ($FeCl_2$ and HCl)solution until the Ni was etched. What are the concentrations of the iron chloride and the Hcl?

3. The top layer/PMMA film was removed and placed on a Si/SiO_2 wafer. The film/SiO_2 was rinsed in two water baths.
4. The transferred top layer/PMMA was then dried very gently with Nitrogen to thin film interference.
5. The transferred top layer/PMMA was placed on a hotplate at 180 ^oC for 1 hour to flatten the PMMA.
6. The PMMA was then removed from the top layer transfer by placing it in acetone for 2.5 hours at room temperature.
7. The transferred top layer was then taken out of the acetone, rinsed with acetone, rinsed with isopropyl aclcohol (IPA) and nitrogen dried.
8. The interface layer was rinsed in two water baths, rinsed with IPA, and nitrogen dried.

Kato (2012) has proposed a method by which the Carbon arrives at the silica substrate, namely that it diffuses through the Ni layer. Kato (2012) . The way that the temperature is measured and how that may or may not affect the reported temperatures due to the variation in the temperatures. The thickness of the Ni film was chosen to be 100 nm so graphene would be most likely to form. The metal catalyst used when growths occurred at the interface in the literature tended to be nickel, thus nickel was chosen to attempt graphene growths. Kato (2012) reported that their attempts at using copper as the catalyst for interface growths was unsuccessful.

2.2 Auger Spectra Simulation

The simplest known method of predicting the Auger energies for a given transition is:

$$E_v - E_x - E_y = E_{Auger} \quad (1)$$

Where all the energies are for a given free atom, $E(VXY)$ is the energy of ejected Auger electrons, $E(V)$ and $E(X)$ are the energies of their respective levels and $E(Y')$ is the energy for an atom already singly

ionized in an inner shell and bears a resemblance to the energy of the Y level. Equation 1 can be modified to include more accurate representations of some of the energies by accounting for effective charge and independence of the manner of transition. Calculation of the Coster-Kronig energies can be calculated using the empirical formula:

$$E_{valence\,3} - E_{valence\,2} - E_{valence\,1} = E_{Coster\ Kronig} \quad . \tag{2}$$

A equation 2 is used to calculate the Auger Coster-Kronig energies for a free carbon atom using data available from NIST as compiled by Kramida (2014). The data from NIST gives the 282 energy levels of neutral free carbon categorized according to spin-orbital momentum splitting and orbital-orbital angular momentum splitting. Free carbon in this model is where there is one carbon atom which is isolated from any bonding with other carbon atoms. Taking into consideration the bonds between the carbon atoms would lead to a more accurate simulation of the Auger spectra for graphene. These energy levels are categorized according to what shell they are in so that the modified form of equation 1 can be used. From there, an algorithm which accounts for such a categorization is developed to systematically calculate all the possible Auger transitions with an initial vacancy in the L shell according to equation 1. The simulation computes the Auger energies at a greater resolution than present experimental ability by considering the possible Auger transitions between the valence electron configurations of free carbon.

A histogram of the calculated Auger energies gives information about which energies have the highest number of transitions that can yield those Auger energies. For those Auger energies which have many transitions backing it the relative height of the peak in that area will be much greater.

Fermi's “Golden rule no. 2” from perturbation theory can give the transition probability per unit time when it is summed over all possible final states. The rule is,

$$w_{fi} = 2\pi \left| \int \psi_f^* V \psi_i \, d\tau \right|^2 \rho(E_f) \tag{3}$$

where ψ_i is the initial state wavefunction, $V=\sum_{i\neq j}(e^2/r_{ij})$, ψ_f is final state wavefunction, and $\rho(E_f)$ is the density of final states for energy E_f. The Auger process between two states can occur either as a direct or exchange process. In the direct process the initial vacancy is filled in by the valence shell closest to the core shell and an electron from a higher valence shell fills in the vacancy in the first valence shell. The exchange process is the reverse where the higher valence shell fills in the vacancy in the core shell. Following the math of Bambynek (1972) the matrix element for the direct process is

$$D=\int\int \psi_a^*(1)\psi_b^*(2)[e^2/(|r_1-r_2|)]\psi_c(1)\psi_d(2)d\tau_1\tau_2 \quad (4)$$

and the exchange matrix element is,

$$E=\int\int \psi_a^*(2)\psi_b^*(1)[e^2/(|r_1-r_2|)]\psi_c(1)\psi_d(2)d\tau_1\tau_2 \quad . \quad (5)$$

With this level of specificity the transition probability per unit time is written as

$$w_{fi}=(2\pi/\hbar)|D-E|^2\rho(E_f) \quad . \quad (6)$$

The continuum wavefunction can be normalized to give the transition probability as,

$$w_{fi}=(1/\hbar^2)|D-E|^2 \quad . \quad (7)$$

Using the continuum wavefunction the matrix elements can be analytically solved out. Bambynek (1972) and Kostroun (1968) both solve out the direct and exchange matrix elements using slightly different assumptions. Solutions from both papers are multiple line equations that will not be repeated here.

This work could be built upon by computing the Auger spectra to predict how the spectra changes for different chemical environments that the graphene is in. Also calculating the Auger energies for free carbon that are for the K shell and then the spectra for free carbon, as well as the spectra for graphene. Such a calculation could also involve an empirical equation from Thompson which would need its own algorithm to be calculated.

3. Growths

Raman shifts of the growths were detected using a Horiba Jobin Yvon HR800 Raman spectrometer equipped with a 532 nm laser. Typical acquisition times were 60 seconds or less and the Raman was acquired under dark room conditions.

It was determined that gas ratios of less than or equal to 1:1 methane to hydrogen were not able to produce sp^2 hybridized carbon and that a gas ratio of 9:1 produces sp^2 hybridized carbon. So future growths only considered a gas ratio of 9:1. This is interesting because specifically what happened in the other growths is that there was no G peak and additionally that carbon appeared to be hydrogenated.

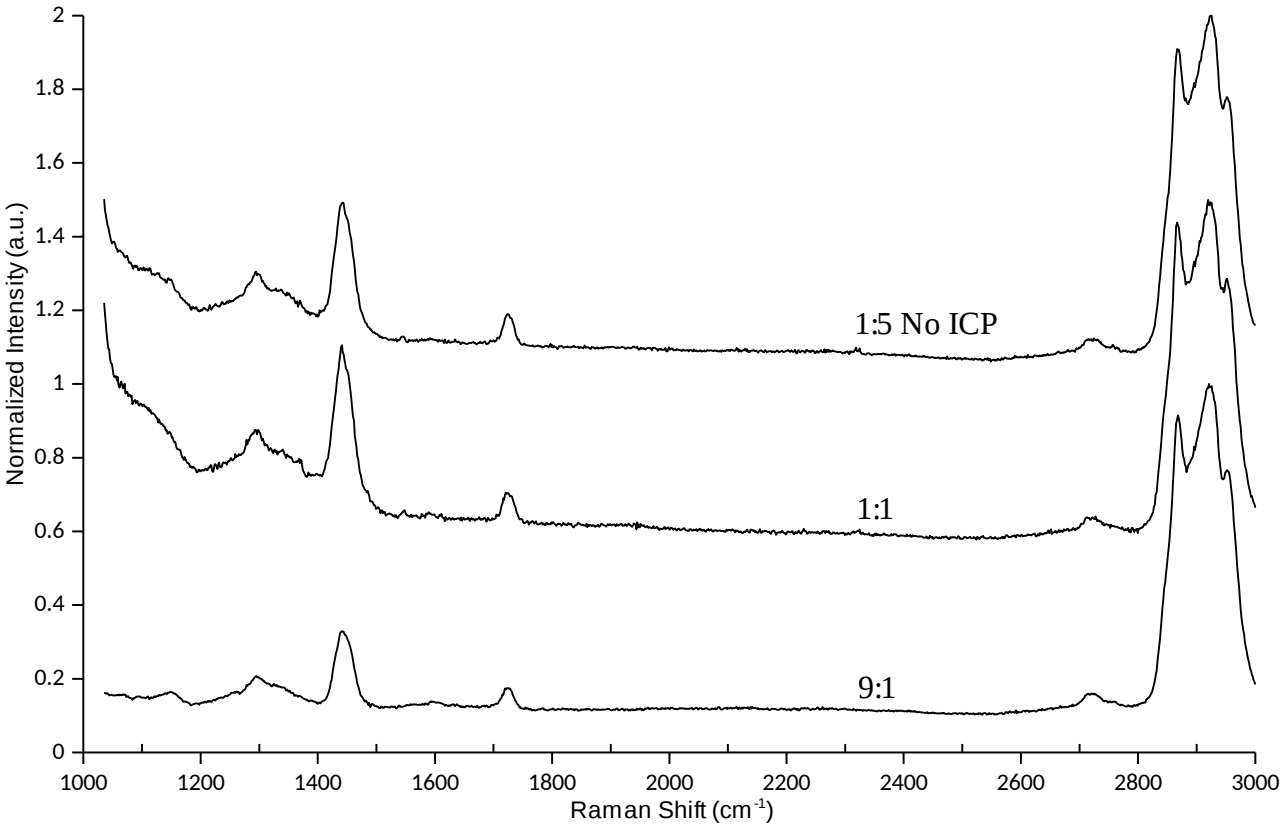

Figure 1: Raman of interface thermal growths

Figure 1 shows the Raman spectra of the interface thermal growths for a gas ratio series. In the interface layer Raman spectra the grouped peaks in the range of 2800-3000 cm^{-1} have been deconvolved

using a pseudo-voigt function to resolve peaks at ~2870 cm^{-1}, ~2920 cm^{-1}, and ~2960 cm^{-1}. These peaks are indicative of carbon hydrogenation as evident by the work of Geraud-Grenier (2004).

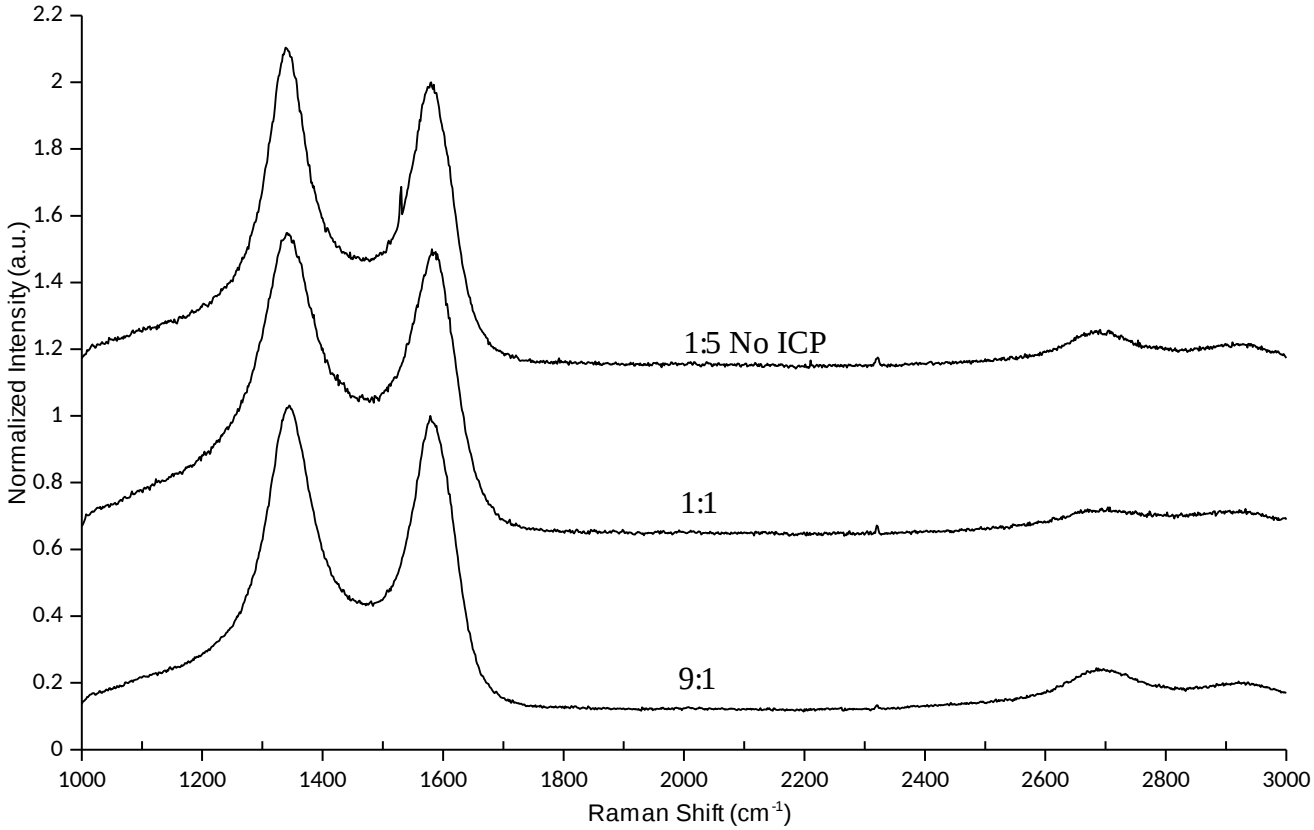

Figure 2: Raman of top layer gas series growths

Figure 2 shows the Raman spectra for the top layer gas ratio series growths. The top layer of graphene shows peaks near 1350 cm^{-1}, 1580 cm^{-1}, and 2700 cm^{-1} which suggests that the layer is disordered graphite.

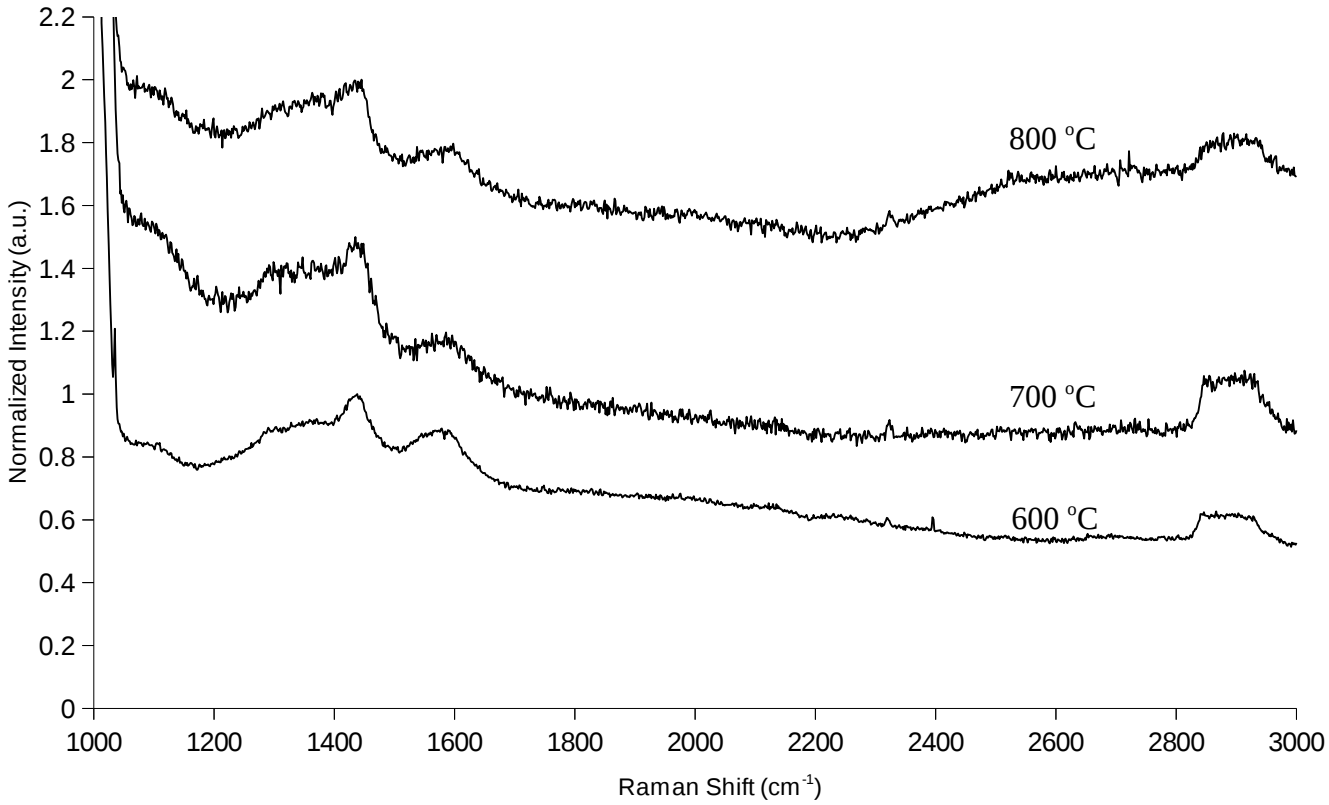

Figure 3: Raman spectra of interface thermal growths

Figure 3 shows the Raman spectra for the thermal growths at the interface of the catalyst and the underlying substrate. When deconvolved, the 600 °C growth has two peaks that are attributable two silica or Si. One of them is at 1288.4 cm^{-1} and the other is at 1439.0 cm^{-1} as reported to the truncated tenth place. The 600 °C growth has a slight up-shifted G peak at 1585.6 cm^{-1} and two possible hydrogenated carbon peaks at 2846.8 cm^{-1} and 2900.4 cm^{-1}. The 700 °C growth exhibits a silica or Si peak at 1439.3 cm^{-1} , a possible D peak at 1372.4 cm^{-1} and a G peak which is up-shifted more than the 600 °C growth at 1594.7 cm^{-1} . The 700 °C growth also shows two possible hydrogenated carbon peaks. One of the peaks is at 2850.4 cm^{-1} and the other is at 2899.4. The 800 °C growth has a possible silica or Si peak at 1278.7 cm^{-1} and a definite silica or Si peak at 1438.8 cm^{-1} . The 800 °C growth contains a possible D peak at 1383.6 cm^{-1} and a G peak at 1585.6 cm^{-1} . The 800 °C growth also shows a possible and a definite hydrogenated peak at 2856.3 cm^{-1} and 2919.2 cm^{-1} , respectively.

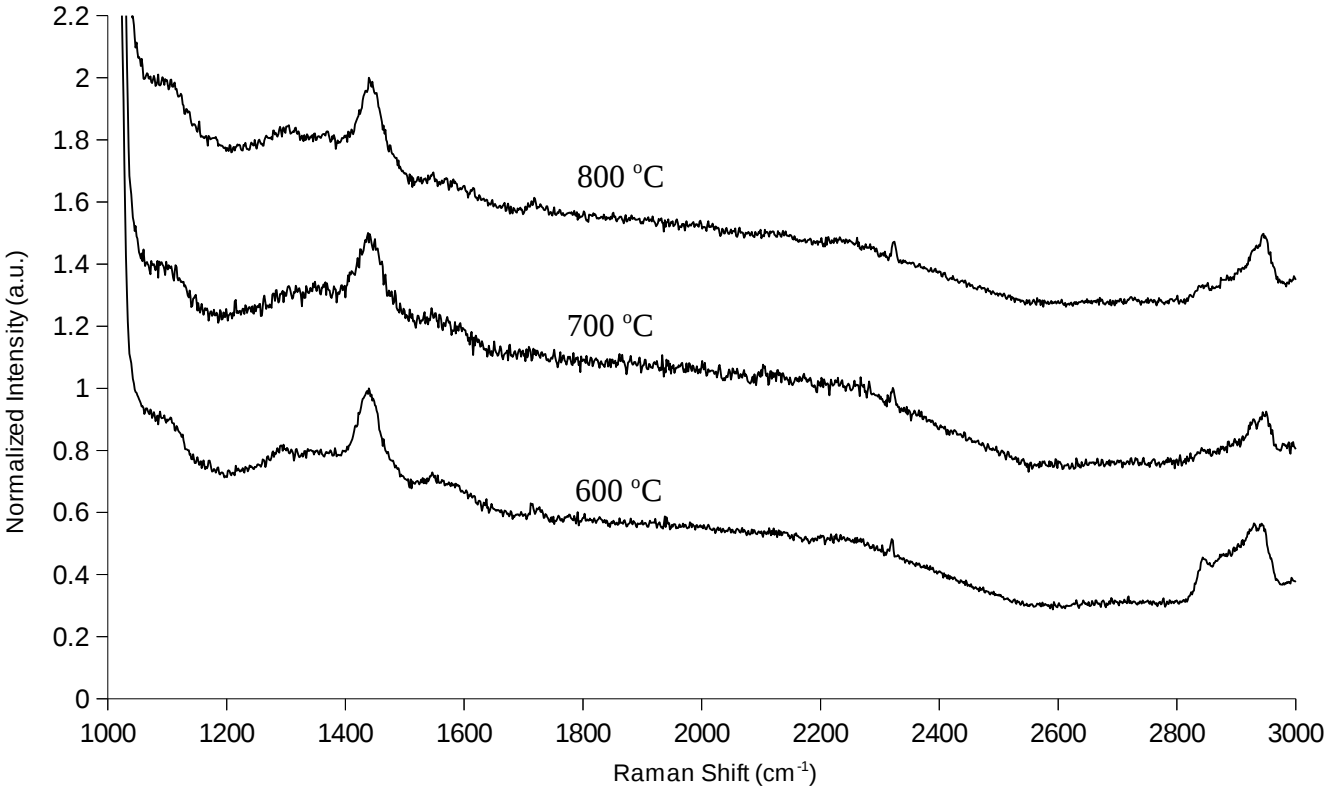

Figure 4: Raman spectra of top layer thermal growths

Figure 4 is a plot of the Raman spectra for the thermal growths that occurred on top of the Ni catalyst. Deconvolution of peaks for the 600 °C growth leas to two silica or Si peaks at 1294.1 cm^{-1} and at 1439.9 cm^{-1}. A G peak is present at 1565.7 cm^{-1} and a hydrogenated carbon peak is at 2875.3 cm^{-1}. There are also two other possible hydrogenated carbon peaks, one at 2929 cm^{-1} and the other at 2946.8 cm^{-1}. The 700 °C growth recorded a silica or Si peak at 1442.6 cm^{-1} and a possible D peak at 1317.6 cm^{-1}. The G peak for this growth occurred at 1565.6 cm^{-1} and two possible hydrogenated carbon peaks in this spectrum are at 2842.4 cm^{-1} and 2929.8 cm^{-1}. The 800 °C growth spectrum points to a silica or silicon peak at 1444.0 cm^{-1} and two possible hydrogenated carbon peaks at 2928.6 cm^{-1} and 2943.6 cm^{-1}. The 800 °C growth contains a D peak at 1363.7 cm^{-1} and a G peak at 1577.8 cm^{-1}.

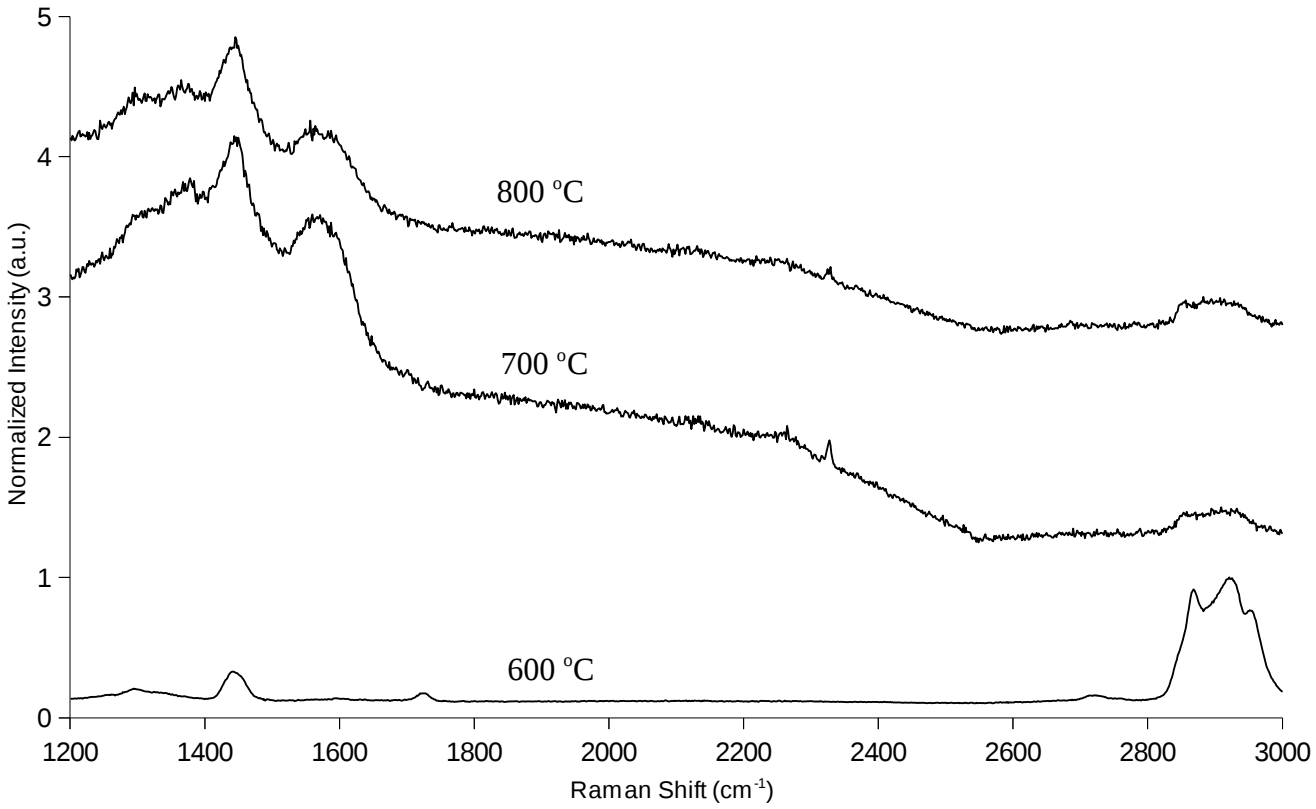

Figure 5: Interface plasma temperature series growths

Figure 5 shows the Raman spectra for the plasma interface growths. The 600 °C spectra has a possible D peak at 1334.73 cm^{-1} and a G peak at 1597.0 cm^{-1}. The spectrum contains a silica or Si peak at 1444.6 cm^{-1} and three hydrogenated carbon peaks at 2864.1 cm^{-1}, 2922.3 cm^{-1}, and 2961 cm^{-1}. The 700 °C growth contains a silicon peak at 1444.3 cm^{-1} and a possible D peak at 1370.1 cm^{-1}. The Raman spectrum also contains a G peak at 1569.6 cm^{-1} and a possible hydrogenated carbon peak at 2907.1 cm^{-1} . The possible peaks that the 800 °C growth contains is a D peak at 1365.9 cm^{-1} and a hydrogenated carbon peak at 2854.0 cm^{-1}. The 800 °C growth contains a silicon peak at 1444.8 cm^{-1} and a G peak at 1576.8 cm^{-1}.

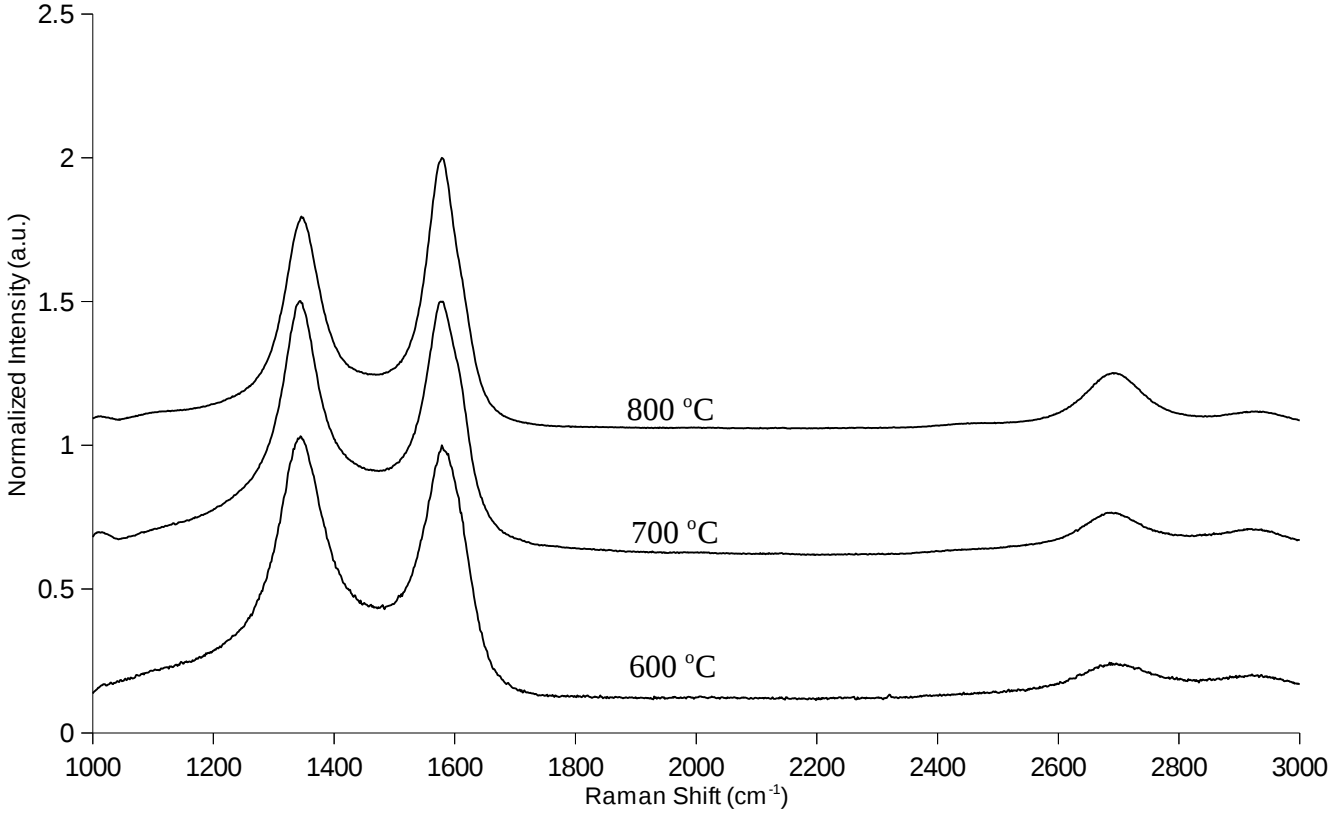

Figure 6: Raman spectra of the top layer plasma growths

In the Raman spectra of the top layer plasma growths as shown in figure 6, four main peaks were resolved for each temperature. The 600 ºC growth has a D peak at 1349.1 cm^{-1} and a G peak at 1580.2 cm^{-1}. It also has a possible 2D peak 2696.1 cm^{-1} and a possible hydrogenated carbon peak at 2934.8 cm^{1}. The 700 ºC growth Raman spectra shows a D peak at 1347.1 cm^{-1} and a G peak at 1578.1 cm^{-1}. The 700 ºC spectra may also contain a 2D peak at 2690.8 cm^{-1} and a hydrogenated carbon peak at 2907.1 cm^{-1}. The 800 ºC growth Raman spectra consists of a D peak at 1350.3 cm^{-1}, a G peak at 1578.5 cm^{-1}, and a possible 2D peak at 2692.5 cm^{-1}. It also has a hydrogenated carbon peak at 2924.4 cm^{-1}.

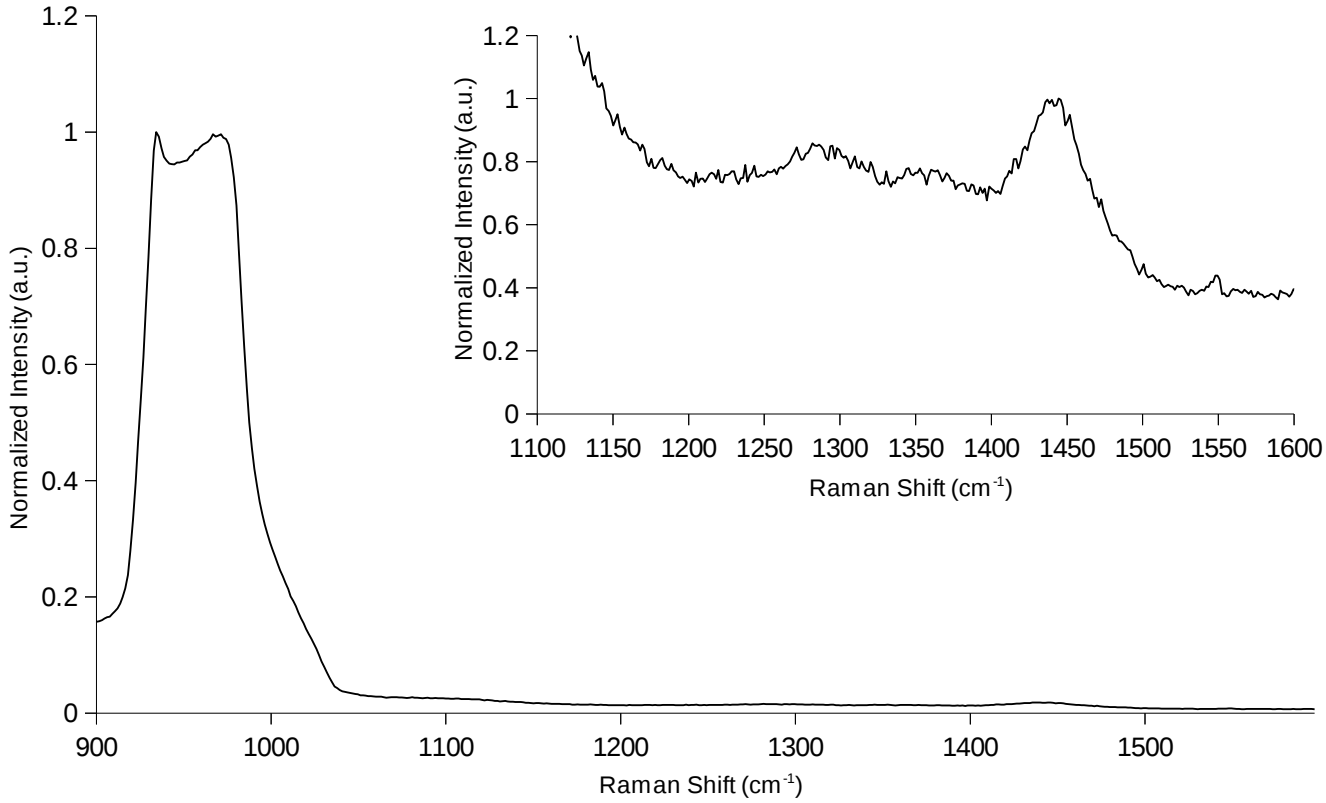

Figure 7: Raman of silica and Si for 60 second exposure time. Inset is 1100 to 1600 wavenumbers normalized to the peak at ~1450.

The Raman spectrum in figure 7 is that of the bare silca-Si wafer. The spectrum was taken under the same conditions as the spectra of the growths. From these spectrum, three principle peaks which occur are also present in the Raman of the growths can be deconvolved: 1291.4 cm^{-1}, 1366.7 cm^{-1}, and 1441.8 cm^{-1}.

3.2 XPS of Thermal Growths

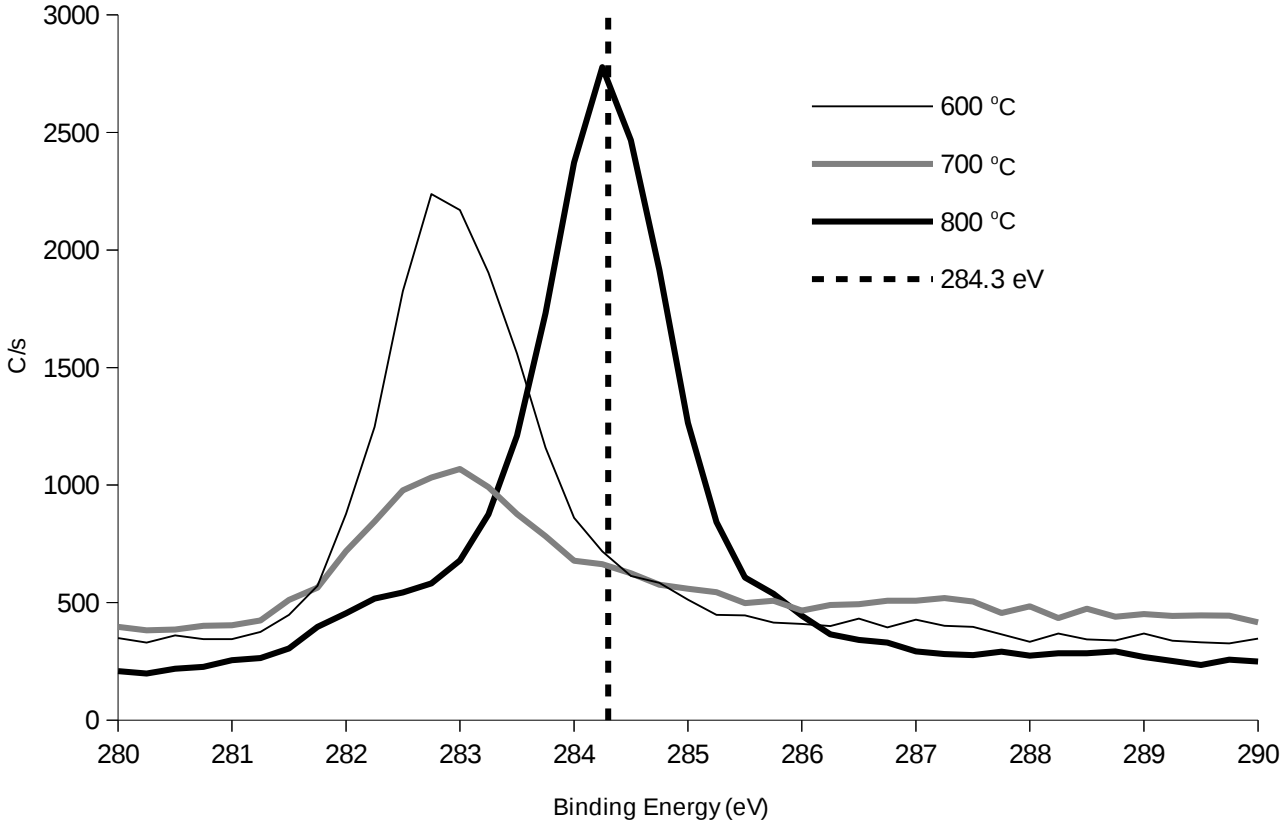

Figure 8: XPS spectra of the thermal growths at the interface

Measurements of XPS and AES spectra occurred on a PHI Versaprobe II. The XPS spectra for the interface thermal growth series was deconvolved using a combination of Gaussian and Lorentzian functions. According to Bourgoin (1999) the principle peak for graphite like (sp^2 hybridized) carbon is found at 284.3 eV. The 700 °C and 800 °C thermal growths have peaks with energies near that of the graphite like carbon with binding energies of 284.36 eV and 284.26 eV, respectively. The most intense peaks in the 600 °C and 700 °C thermal growths both with binding energies of 282.84 eV are of lower energy than seen in the literature regarding carbon XPS.

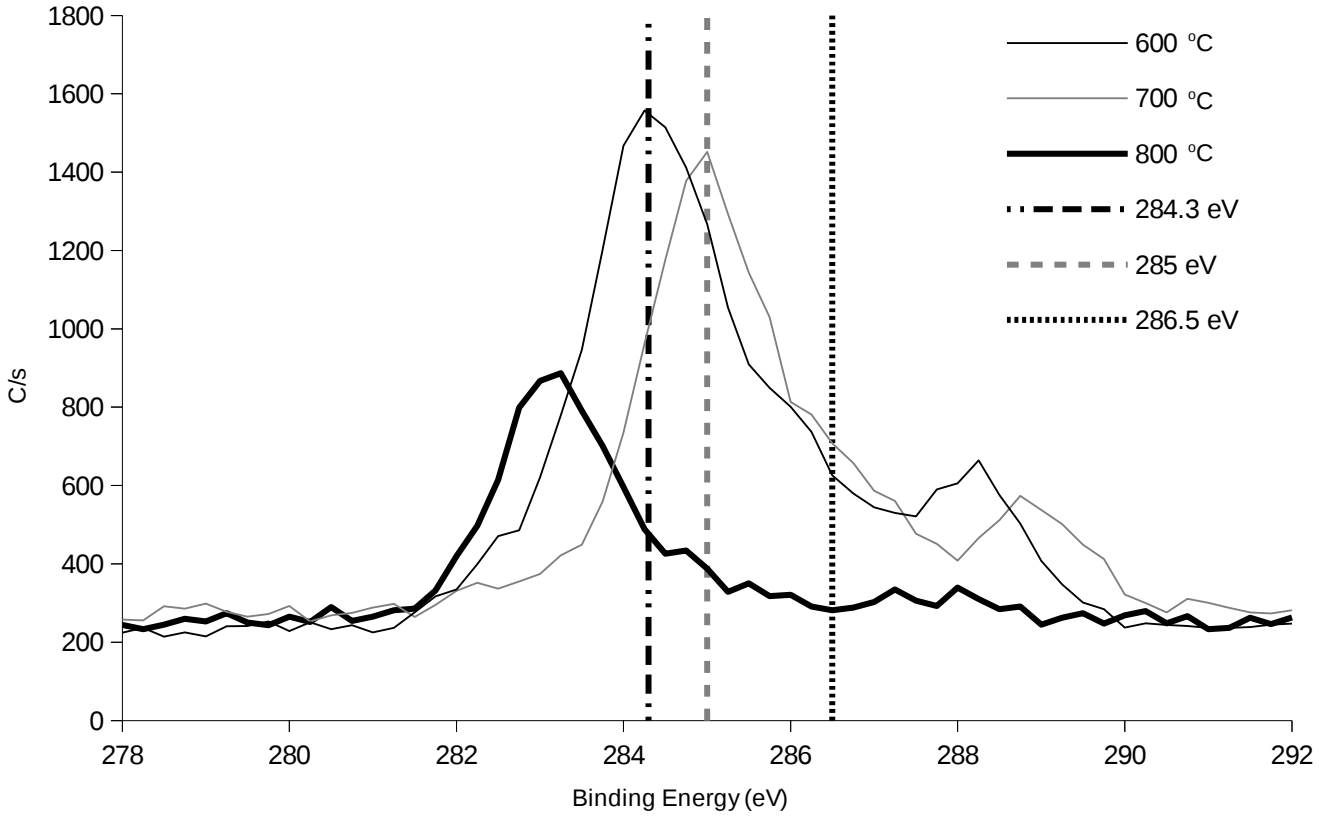

Figure 9: XPS spectra of the thermal growths on the top layer

The XPS spectra for the temperature series of the growths on top of the catalyst tended to exhibit higher energy peaks when deconvolved as compared to the interface growths. The 600 °C growth on top of the catalyst consisted of graphite like carbon as evident by the peak at 284.3 eV. The 600 °C growth also has one lower energy and two higher energy peaks unaccounted for. The 700 °C growth on top of the catalyst consisted of diamond-like carbon (sp^3 hybridization), a 285 eV peak according to Bourgoin, as evident by the peak at 284.9 eV. The 700 °C growth also has a peak 286.5 which can be attributed to a carbon-oxygen bond which may be due to prolonged exposure to the air. The growth on top of the catalyst at 800 °C consists of only two peaks, a peak at 283.1 which may be graphite like and a diamond-like peak at 284.7 eV. The percent area of the C 1s spectra for the 800 °C growth that is diamond-like is 11.87%.

In terms of graphite like carbon both the 800 °C growth at the interface had the highest peak

and the 600 ºC growth on top of the catalyst had the highest peak. Thus towards production of graphene at the interface higher temperature growths should be attempted even though a resulting loss in graphite like carbon in the growth on top of the catalyst may occur. In addition to these Ni based growth, Cu growths were also grown under these same conditions and could be analyzed via Raman, XPS and AES.

3.3 AES of Thermal Growths

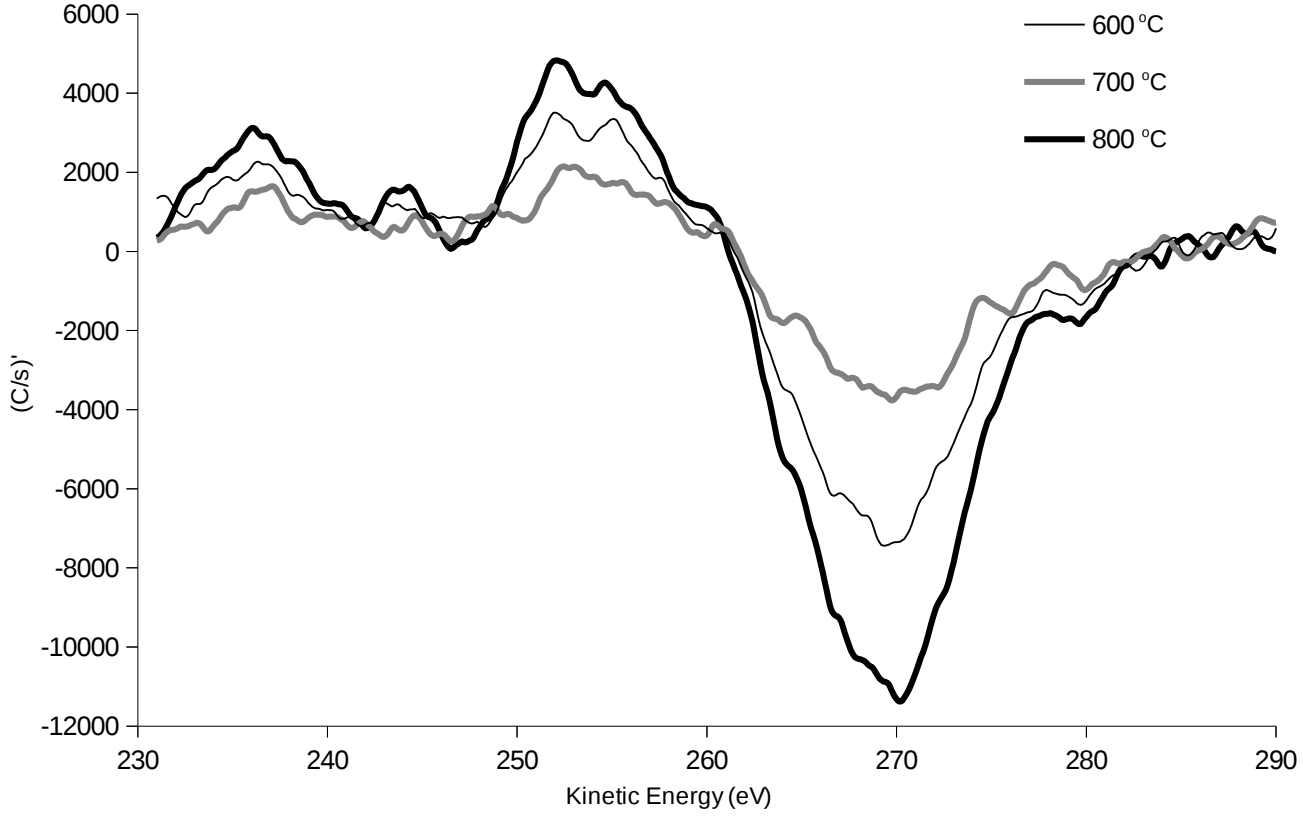

Figure 10: Interface thermal AES in the carbon range

Figure 10 shows the first derivative of the AES spectra for the interface thermal growths in the energy window of carbon. A metric known as the D parameter, defined by Kaciulis (2013) as the distance between the most positive maximum and the most negative minimum, has been calculated for these spectra. The D parameter for the 600 ºC growth is 17.4 eV, for the 700 ºC growth is 17.3 eV, and for the 800 ºC growth is 18.1 eV. So, there is very little change in the D parameter between the two

lowest temperature growths and there is some change in the D parameter between the highest temperature growth and the two lowest temperature growths.

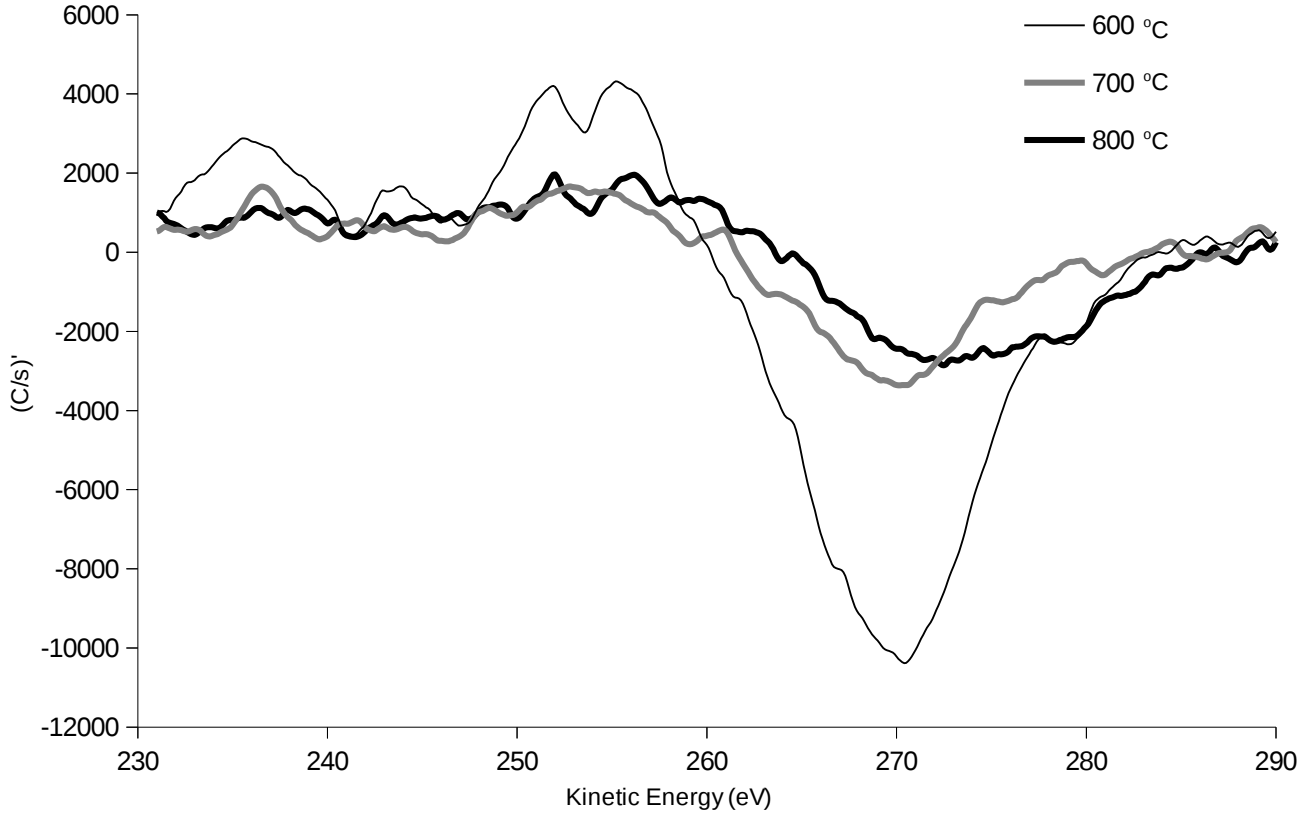

Figure 11: Top layer thermal AES in the carbon range

Figure 11 shows the AES spectra in the energy range of the carbon KVV Auger transitions. The D parameter has been calculated for these spectra and is 15.2 eV for the 600 °C growth, 17.4 eV for the 700 °C growth, and 20.5 eV for the 800 °C growth. The difference in the D parameter is much larger than that of the thermal interface growths. The parameter also changes in value by several eVs for each growth whereas in the thermal interface growths, the D parameter changed by only 0.1 eV between the two lower temperature growths.

3.4 XPS of Plasma Growths

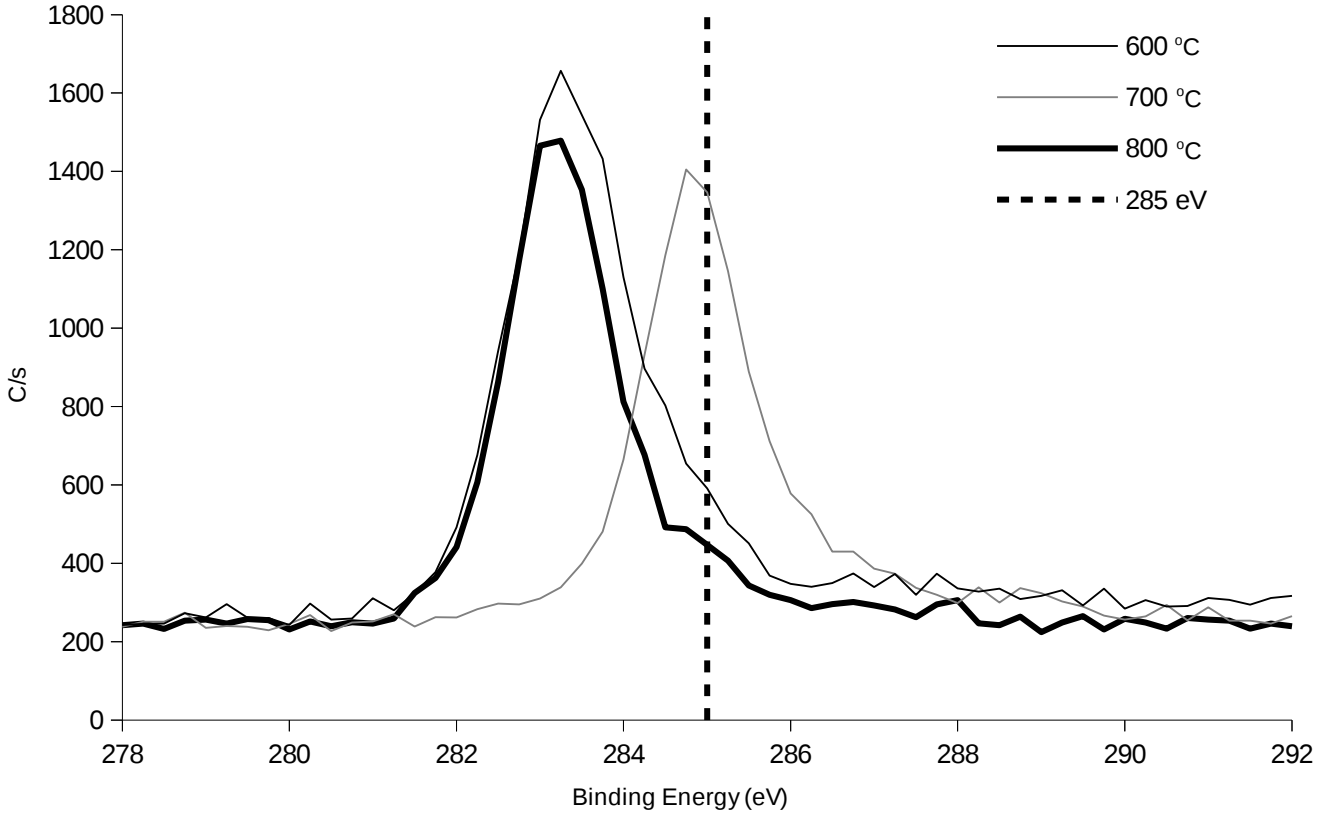

Figure 12: XPS of plasma interface growths

The XPS of the plasma based CVD growths at the interface of the nickel and the underlying substrate are shown in figure 12. The XPS of the 600 °C plasma interface growth consists of an unknown peak at 283.3 eV and a diamond-like peak at 284.83 eV. The XPS of the 700 °C growth deconvolve to a peak at 284.5 eV which is due to diamond-like carbon, a peak at 286.5 eV which is due to oxygen bound carbon and an unknown peak at 288.8 eV. XPS of the 800 °C growth consists of an unknown peak at 283.2 eV and a diamond-like peak at 284.9 eV. The 600 °C growth has a diamond-like carbon peak with area of 10.1 %, the 700 °C growth has a percent area of 79.8 % for the diamond-like carbon peak, and the 800 °C growth with a percent area of 6.22 % for the diamond-like carbon peak.

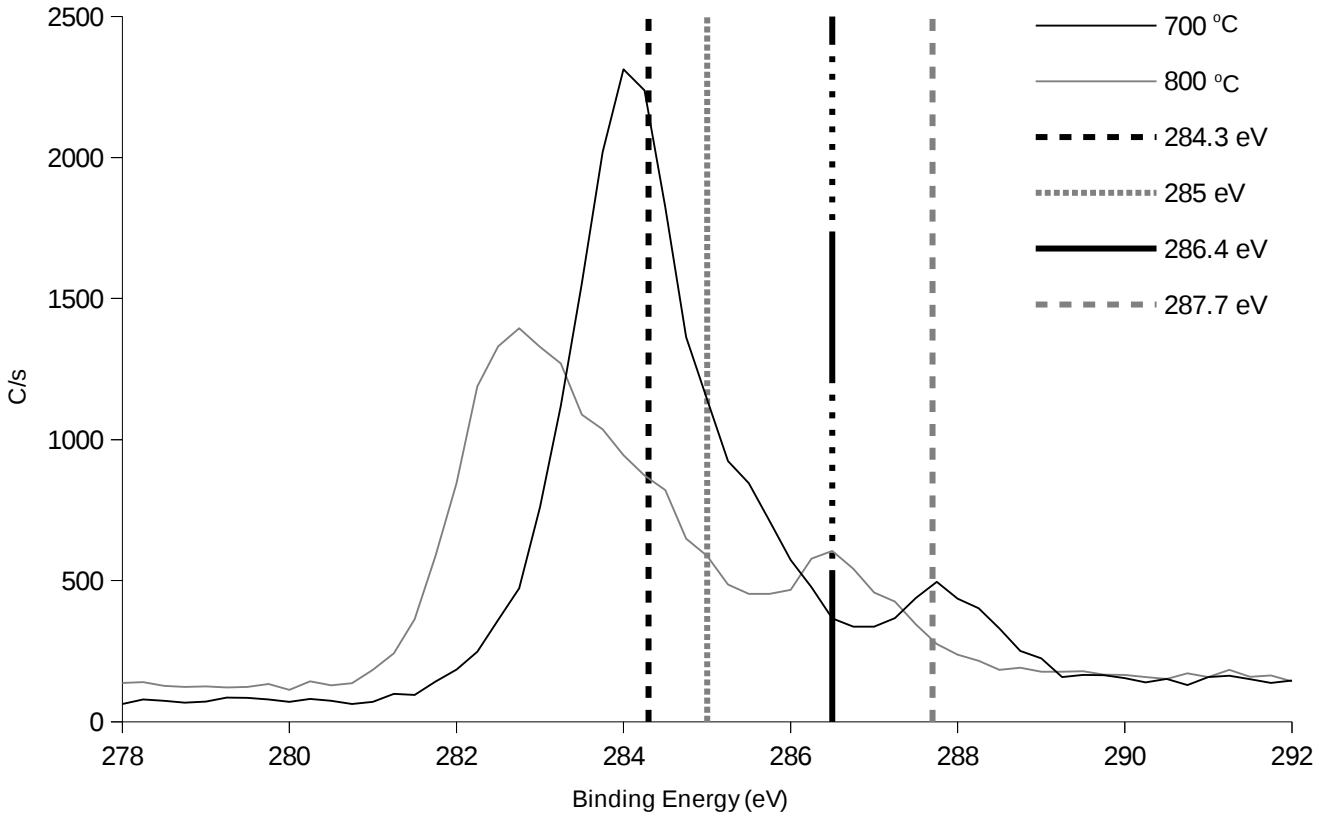

Figure 13: XPS of plasma top layer growths

The 700 °C top layer plasma growth consists of an unknown peak at 283.1 eV a graphite like peak at 284.0 eV, a diamond-like peak at 285.3 eV and a double bonded carbon-oxygen peak at 287.8 eV. The 800 °C growth top layer has an unknown peak at 282.6 eV, a graphite like peak at 284 eV, and a carbon-oxygen peak at 286.6 eV. The carbon-oxygen peaks may be due to air exposure contamination.

3.5 AES of Plasma Growths

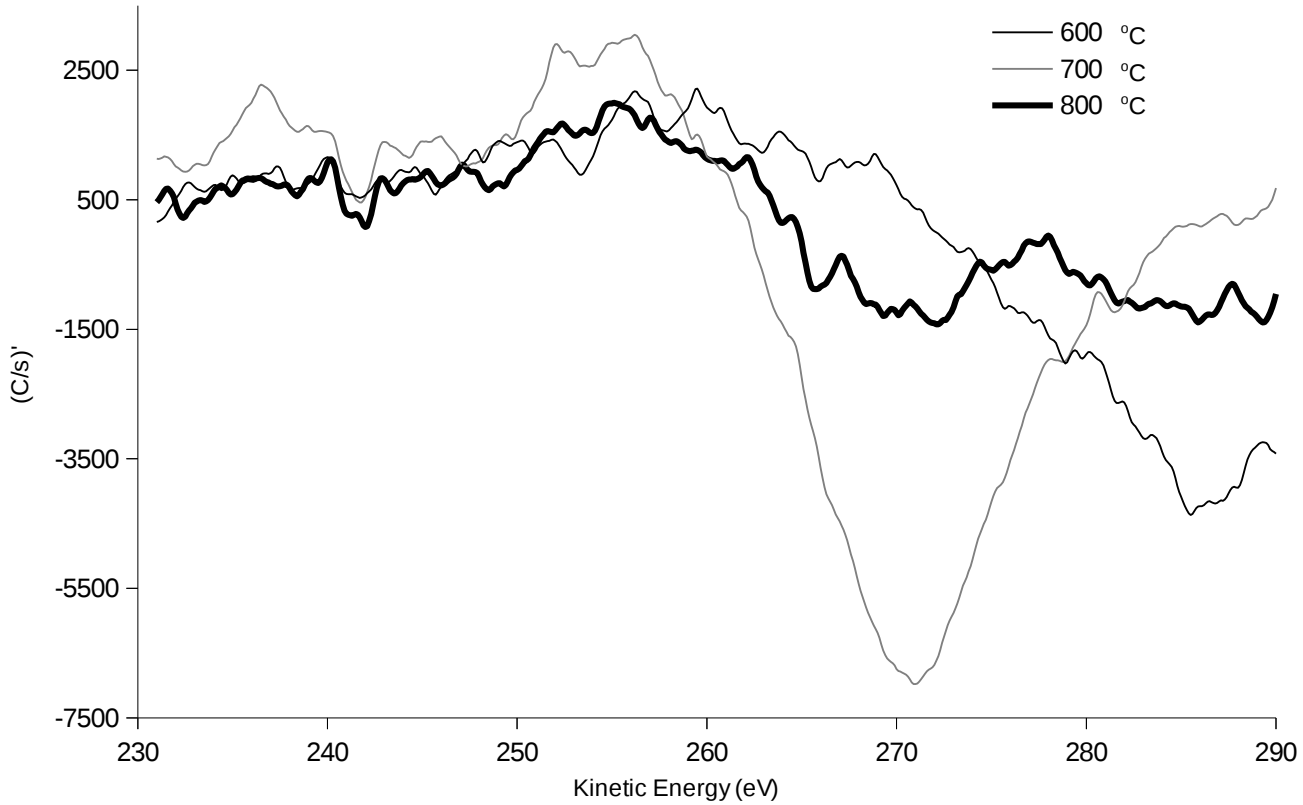

Figure 14: Carbon KVV AES for plasma growth interface layer

In figure 14, the carbon KVV AES spectra are shown. The D parameter for these spectra has been calculated to be 26 eV for the 600 °C growth, 14.7 eV for the 700 °C growth and 17 eV for the 800 °C growth. This differs from the thermal growth interface layer AES spectra which had similar D parameters for the 600 °C and 700 °C growths and an increase for the 800 °C growth.

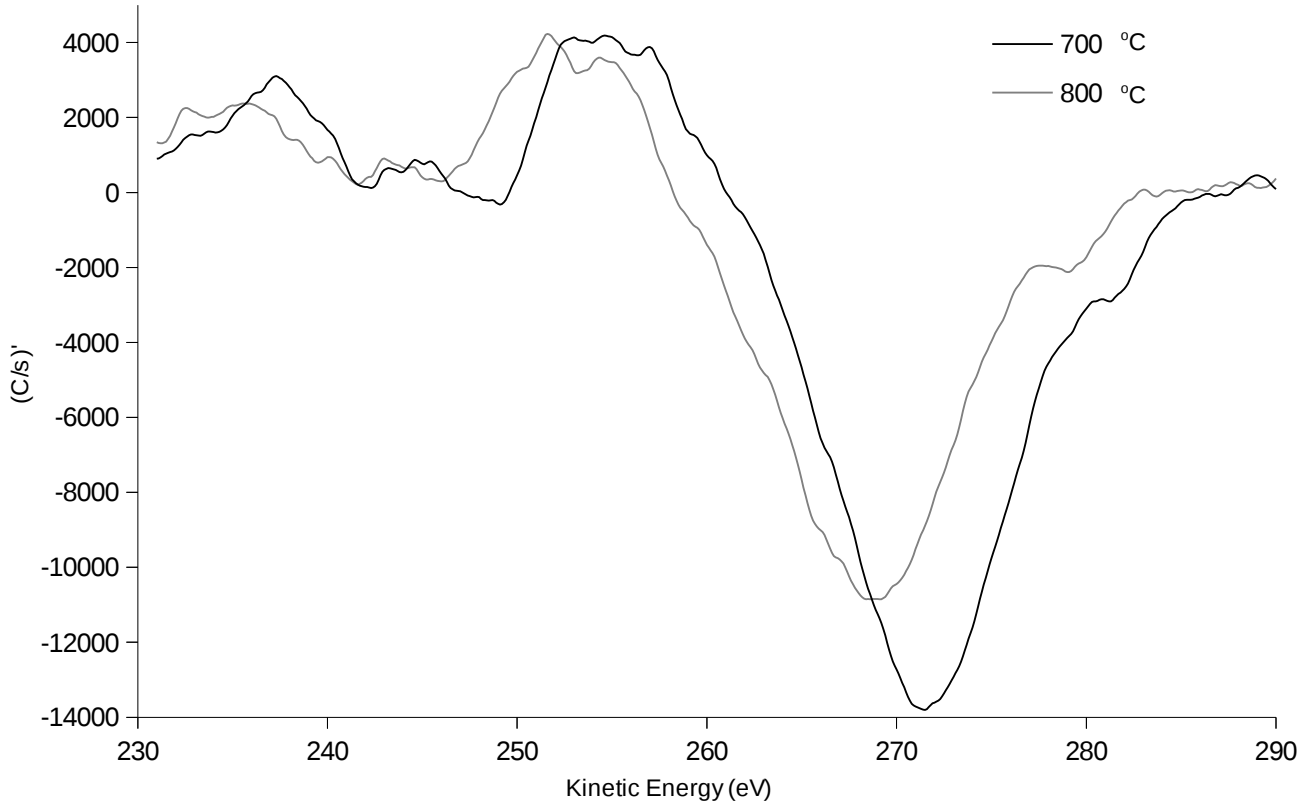

Figure 15: Carbon KVV AES for plasma growth top layer

Figure 15 show the KVV carbon AES spectra for the plasma growths on top of the nickel catalyst. The D parameter was calculated to be 16.9 eV for the 700 °C growth and 16.8 eV for the 800 °C growth. This differs from the thermal top growths where the D parameter had a 3.1 eV difference between the 700 °C growth and the 800 °C growth.

4. Simulated Free Carbon Auger Energies

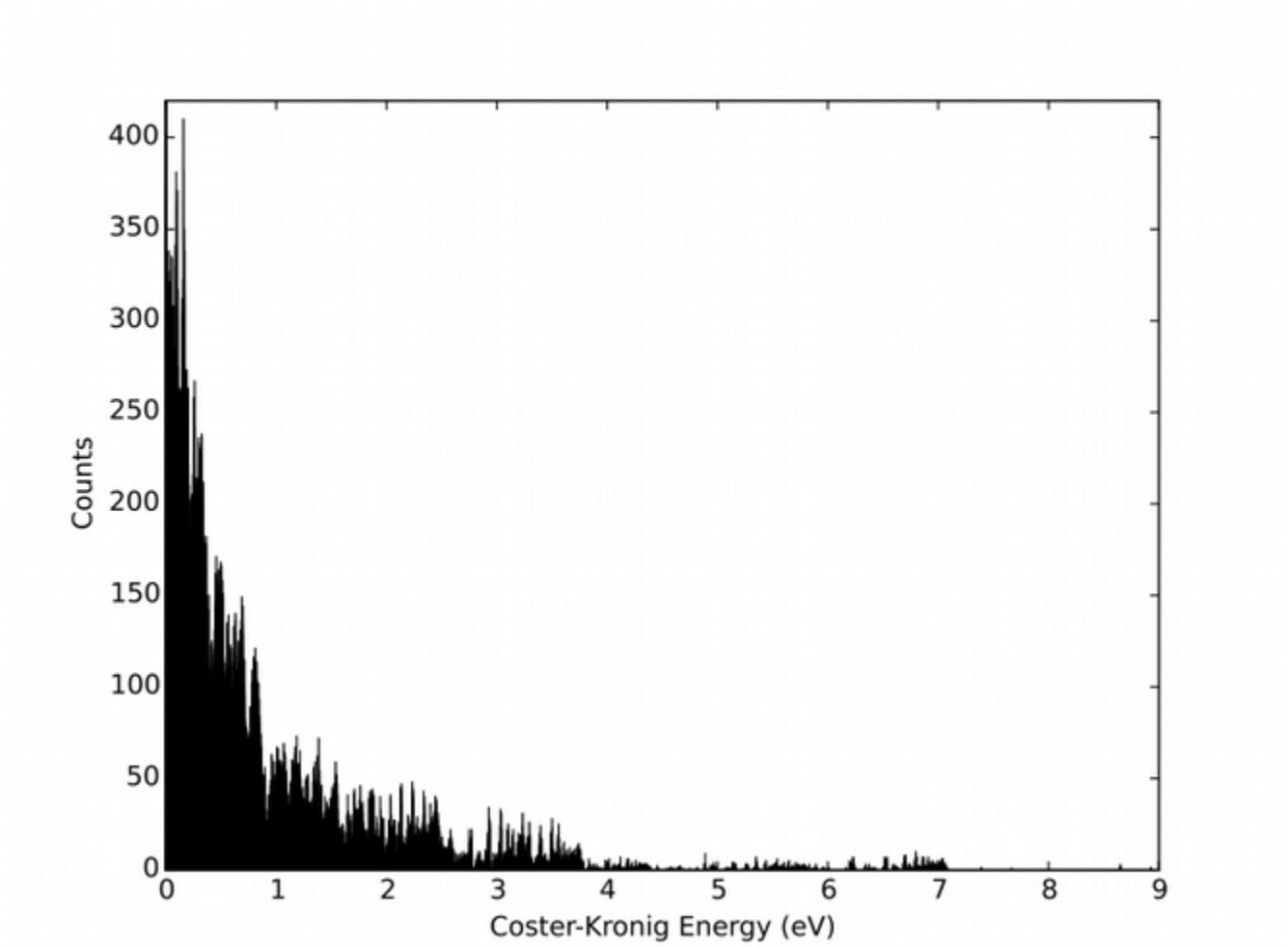

Figure 16: Histogram of Coster-Kronig energies for neutral carbon with 5000 bins.

These energies were calculated using the python script in the appendix. The Coster-Kronig auger energies with the highest counts are those that lie between 0 and 1 eV. The principle peaks from an experiment could be found and compared to the calculated energies from the python script. The difference between the experimental peaks and the those from the python script could be found. Observations could be made about the differences to determine if there are any systematic differences or if the differences appear to have no pattern. The values of the energies point to specific transitions taking place to yield those energies. Those transitions occur based on certain electron configurations.

So, if an experimental spectra of Coster-Kronig transitions could be compared to the computed one, it would be possible to determine what electron configurations the carbon has.

Appendix

```
#Python script for computing Auger energies

from pandas import Series, DataFrame
import matplotlib.pyplot as plt
import pandas as pd
import itertools

#Get and categorize energy levels accoring to electron configuration, e stands for to the power in the
electron configurations, div stands for /,that is, divide by
# ll is ( and rr is  )

#Test of how to put non continous elements in a list
#test = [5,6,7,8]
#list_test = test[0],test[3]
#print list_test
#list(c1elvls[:])
#Reads the first line in the code as a header skipping over enteringing it into the series
#When calling multiple elements from c1elvls the ending value must be 1 more than the desired value
c1elvls = pd.read_table('c1energylevels.txt', squeeze = True)
#c1energylevels.txt is a text file of neutral carbon energy levels retrieved from the NIST atomic spectra
database.
c1econfig_2se22pe2 = list(c1elvls[0:5])
c1econfig_2s2pe3 = [c1elvls[5], list(c1elvls[10:13]),list(c1elvls[23:26]), c1elvls[276]]
c1econfig_2se22p3s = list(c1elvls[6:10])
c1econfig_2se22p3p = list(c1elvls[13:23])
c1econfig_2se22p3d = [c1elvls[26], list(c1elvls[30:36]), list(c1elvls[37:42])]
c1econfig_2se22p4s = [list(c1elvls[27:29]),c1elvls[36]]
c1econfig_2se22p4p = list(c1elvls[42:52])
c1econfig_2se22p4d = [c1elvls[52], list(c1elvls[56:62]), c1elvls[67],c1elvls[74], list(c1elvls[77:80]) ]
c1econfig_2se22p5s = [list(c1elvls[53:56]), c1elvls[62]]
c1econfig_2se22pll2p1div2rr4f = list(c1elvls[63:67])
c1econfig_2se22pll2p3div2rr4f = [list(c1elvls[68:74]),list(c1elvls[75:77])]
c1econfig_2se22p5p = list(c1elvls[80:93])
c1econfig_2se22p6s = [list(c1elvls[93:96]), c1elvls[99]]
c1econfig_2se22p5d = [list(c1elvls[96:99]), c1elvls[100],c1elvls[105], list(c1elvls[108:111])]
c1econfig_2se22pll2p3div2rr5f = [list(c1elvls[101:105]), list(c1elvls[106:108])]
c1econfig_2se22p6p = list(c1elvls[111:120])
c1econfig_2se22p6d = [list(c1elvls[121:124]),list(c1elvls[127:130]),
c1elvls[133],list(c1elvls[136:140])]
c1econfig_2se22p7s = [list(c1elvls[124:127]), c1elvls[132]]
c1econfig_2se22pll2p1div2rr6f = list(c1elvls[130:132])
c1econfig_2se22pll2p3div2rr6f = [list(c1elvls[134:136]), list(c1elvls[140:142])]
c1econfig_2se22p7p = list(c1elvls[142:151])
```

```
c1econfig_2se22p7d = [list(c1elvls[151:155]),list(c1elvls[158:161]), c1elvls[163],
list(c1elvls[166:169]),c1elvls[170]]
c1econfig_2se22p8s = [list(c1elvls[155:158]), c1elvls[162]]
c1econfig_2se22pll2p1div2rr7f = c1elvls[161]
c1econfig_2se22pll2p3div2rr7f = [list(c1elvls[164:166]), c1elvls[169], c1elvls[171]]
c1econfig_2se22p8p = list(c1elvls[172:179])
c1econfig_2se22p8d = [list(c1elvls[179:183]), list(c1elvls[185:188]), c1elvls[191],
list(c1elvls[193:197])]
c1econfig_2se22p9s = [list(c1elvls[183:185]), c1elvls[190]]
c1econfig_2se22pll2p1div2rr8f = list(c1elvls[188:190])
c1econfig_2se22pll2p3div2rr8f = c1elvls[192]
c1econfig_2se22p9p = list(c1elvls[197:201])
c1econfig_2se22p9d = [list(c1elvls[201:204]), list(c1elvls[205:208]), list(c1elvls[209:212])]
c1econfig_2se22p10s = [c1elvls[204], c1elvls[208]]
c1econfig_2se22p10p = list(c1elvls[212:215])
c1econfig_2se22p10d = list(c1elvls[215:221])
c1econfig_2se22p11p = c1elvls[221]
c1econfig_2se22p11d = list(c1elvls[222:228])
c1econfig_2se22p12d = [list(c1elvls[228:230]), list(c1elvls[231:233])]
c1econfig_2se22p13s = [c1elvls[230], list(c1elvls[236:238])]
c1econfig_2se22p13d = list(c1elvls[233:235])
c1econfig_2se22p14s = c1elvls[235]
c1econfig_2se22p14d = list(c1elvls[238:242])
c1econfig_2se22p15d = [c1elvls[242],list(c1elvls[244:246])]
c1econfig_2se22p16d = [c1elvls[243],list(c1elvls[247:249])]
c1econfig_2se22p17d = [c1elvls[246],list(c1elvls[250:252])]
c1econfig_2se22p18d = [c1elvls[249],list(c1elvls[254:256])]
c1econfig_2se22p19d = [c1elvls[252],list(c1elvls[257:259])]
c1econfig_2se22p20d = [c1elvls[253], list(c1elvls[260:262])]
c1econfig_2se22p21d = [c1elvls[256], c1elvls[262]]
c1econfig_2se22p22d = [c1elvls[259], c1elvls[263]]
c1econfig_2se22p23d = c1elvls[264]
c1econfig_2se22p24d = c1elvls[265]
c1econfig_2se22p25d = c1elvls[266]
c1econfig_2se22p26d = c1elvls[267]
c1econfig_2se22p27d = c1elvls[268]
c1econfig_2se22p28d = c1elvls[269]
c1econfig_2se22p29d = c1elvls[270]
c2econfig_2pe2p1div2 = c1elvls[271]
c2econfig_2pe2p3div2 = c1elvls[272]
c1econfig_2s2pe2ll4Prr3s = list(c1elvls[273:276])

#Place the C I energy configurations in a list.
#Do not include the core shell energies for they will be appended later
c1econfig = []
c1econfig.append(c1econfig_2s2pe3)
c1econfig.append(c1econfig_2se22p3s)
```

```
c1econfig.append(c1econfig_2se22p3p )
c1econfig.append(c1econfig_2se22p3d )
c1econfig.append(c1econfig_2se22p4s)
c1econfig.append(c1econfig_2se22p4p)
c1econfig.append(c1econfig_2se22p4d )
c1econfig.append(c1econfig_2se22p5s)
c1econfig.append(c1econfig_2se22pll2p1div2rr4f )
c1econfig.append(c1econfig_2se22pll2p3div2rr4f )
c1econfig.append(c1econfig_2se22p5p)
c1econfig.append(c1econfig_2se22p6s)
c1econfig.append(c1econfig_2se22p5d )
c1econfig.append(c1econfig_2se22pll2p3div2rr5f)
c1econfig.append(c1econfig_2se22p6p)
c1econfig.append(c1econfig_2se22p6d)
c1econfig.append(c1econfig_2se22p7s)
c1econfig.append(c1econfig_2se22pll2p1div2rr6f)
c1econfig.append(c1econfig_2se22pll2p3div2rr6f)
c1econfig.append(c1econfig_2se22p7p)
c1econfig.append(c1econfig_2se22p7d)
c1econfig.append(c1econfig_2se22p8s)
c1econfig.append(c1econfig_2se22pll2p1div2rr7f)
c1econfig.append(c1econfig_2se22pll2p3div2rr7f)
c1econfig.append(c1econfig_2se22p8p)
c1econfig.append(c1econfig_2se22p8d)
c1econfig.append(c1econfig_2se22p9s)
c1econfig.append(c1econfig_2se22pll2p1div2rr8f)
c1econfig.append(c1econfig_2se22pll2p3div2rr8f)
c1econfig.append(c1econfig_2se22p9p)
c1econfig.append(c1econfig_2se22p9d)
c1econfig.append(c1econfig_2se22p10s)
c1econfig.append(c1econfig_2se22p10p)
c1econfig.append(c1econfig_2se22p10d )
c1econfig.append(c1econfig_2se22p11p)
c1econfig.append(c1econfig_2se22p11d)
c1econfig.append(c1econfig_2se22p12d)
c1econfig.append(c1econfig_2se22p13s)
c1econfig.append(c1econfig_2se22p13d)
c1econfig.append(c1econfig_2se22p14s )
c1econfig.append(c1econfig_2se22p14d )
c1econfig.append(c1econfig_2se22p15d )
c1econfig.append(c1econfig_2se22p16d)
c1econfig.append(c1econfig_2se22p17d)
c1econfig.append(c1econfig_2se22p18d)
c1econfig.append(c1econfig_2se22p19d)
c1econfig.append(c1econfig_2se22p20d )
c1econfig.append(c1econfig_2se22p21d)
c1econfig.append(c1econfig_2se22p22d)
```

```
c1econfig.append(c1econfig_2se22p23d)
c1econfig.append(c1econfig_2se22p24d)
c1econfig.append(c1econfig_2se22p25d )
c1econfig.append(c1econfig_2se22p26d)
c1econfig.append(c1econfig_2se22p27d )
c1econfig.append(c1econfig_2se22p28d)
c1econfig.append(c1econfig_2se22p29d)
c1econfig.append(c2econfig_2pe2p1div2)
c1econfig.append(c2econfig_2pe2p3div2 )
c1econfig.append(c1econfig_2s2pe2ll4Prr3s )

c1econfig_names = []
c1econfig_names = []
c1econfig_names.append("c1econfig_2s2pe3")
c1econfig_names.append("c1econfig_2se22p3s")
c1econfig_names.append("c1econfig_2se22p3p" )
c1econfig_names.append("c1econfig_2se22p3d ")
c1econfig_names.append("c1econfig_2se22p4s")
c1econfig_names.append("c1econfig_2se22p4p")
c1econfig_names.append("c1econfig_2se22p4d" )
c1econfig_names.append("c1econfig_2se22p5s")
c1econfig_names.append("c1econfig_2se22pll2p1div2rr4f" )
c1econfig_names.append("c1econfig_2se22pll2p3div2rr4f" )
c1econfig_names.append("c1econfig_2se22p5p")
c1econfig_names.append("c1econfig_2se22p6s")
c1econfig_names.append("c1econfig_2se22p5d ")
c1econfig_names.append("c1econfig_2se22pll2p3div2rr5f")
c1econfig_names.append("c1econfig_2se22p6p")
c1econfig_names.append("c1econfig_2se22p6d")
c1econfig_names.append("c1econfig_2se22p7s")
c1econfig_names.append("c1econfig_2se22pll2p1div2rr6f")
c1econfig_names.append("c1econfig_2se22pll2p3div2rr6f")
c1econfig_names.append("c1econfig_2se22p7p")
c1econfig_names.append("c1econfig_2se22p7d")
c1econfig_names.append("c1econfig_2se22p8s")
c1econfig_names.append("c1econfig_2se22pll2p1div2rr7f")
c1econfig_names.append("c1econfig_2se22pll2p3div2rr7f")
c1econfig_names.append("c1econfig_2se22p8p")
c1econfig_names.append("c1econfig_2se22p8d")
c1econfig_names.append("c1econfig_2se22p9s")
c1econfig_names.append("c1econfig_2se22pll2p1div2rr8f")
c1econfig_names.append("c1econfig_2se22pll2p3div2rr8f")
c1econfig_names.append("c1econfig_2se22p9p")
c1econfig_names.append("c1econfig_2se22p9d")
c1econfig_names.append("c1econfig_2se22p10s")
c1econfig_names.append("c1econfig_2se22p10p")
c1econfig_names.append("c1econfig_2se22p10d ")
```

```
c1econfig_names.append("c1econfig_2se22p11p")
c1econfig_names.append("c1econfig_2se22p11d")
c1econfig_names.append("c1econfig_2se22p12d")
c1econfig_names.append("c1econfig_2se22p13s")
c1econfig_names.append("c1econfig_2se22p13d")
c1econfig_names.append("c1econfig_2se22p14s ")
c1econfig_names.append("c1econfig_2se22p14d ")
c1econfig_names.append("c1econfig_2se22p15d ")
c1econfig_names.append("c1econfig_2se22p16d")
c1econfig_names.append("c1econfig_2se22p17d")
c1econfig_names.append("c1econfig_2se22p18d")
c1econfig_names.append("c1econfig_2se22p19d")
c1econfig_names.append("c1econfig_2se22p20d" )
c1econfig_names.append("c1econfig_2se22p21d")
c1econfig_names.append("c1econfig_2se22p22d")
c1econfig_names.append("c1econfig_2se22p23d")
c1econfig_names.append("c1econfig_2se22p24d")
c1econfig_names.append("c1econfig_2se22p25d" )
c1econfig_names.append("c1econfig_2se22p26d")
c1econfig_names.append("c1econfig_2se22p27d" )
c1econfig_names.append("c1econfig_2se22p28d")
c1econfig_names.append("c1econfig_2se22p29d")
c1econfig_names.append("c2econfig_2pe2p1div2")
c1econfig_names.append("c2econfig_2pe2p3div2" )
c1econfig_names.append("c1econfig_2s2pe2ll4Prr3s" )

#Loop structure looks through each element in c1econfig. If the element contains a list
#continually look inside that list for lists. When a list is found remove that element and then
#return it as seperate elements.
for i in range(len(c1econfig)):
    n = 0
    if isinstance(c1econfig[i],list):
        while n < len(c1econfig[i]):
            if isinstance(c1econfig[i][n], list):
                poppedelement = c1econfig[i].pop(n)
                for j in range(len(poppedelement)):
                    c1econfig[i].insert(j+n, poppedelement[j])
            n = n+1


#Create a list of possible Auger transistions
transistions = []
transistion_names = []
place = []
for i in range(len(c1econfig)):
#    print c[i]
#    print "----"
```

```
#    print " ---- one i iteration -----"
    place.append("*-----*")
    place.append(i)
    if i+len(c1econfig) <= len(c1econfig):
        #For the special case where i=0. If the iterator is less than or equal to the number of electron
configurations
    #append the ith element of c1econfig plus one less than the number of electron configurations to
element 0 of
    #the variable transitions. Append the ith element of c1econfig to element 1 of the variable
transistions.
#        print i
#        print i +len(c) -1
        transistions.append([c1econfig[i+len(c1econfig)-1], c1econfig[i]])
        transistion_names.append([c1econfig_names[i+len(c1econfig)-1], c1econfig[i]])
    for j in range(1,len(c1econfig)):
#        print " ---- one j iteration -----"
#        print i
#        print j
    #For the cases when i is not equal to zero. When the sum of the iterators is less than or equal to
    #the number of electron configurations append to transistions. Append to element zero of transistions
    #the element of c1econfig that is one less than the sum of the iterators. Append to element one of
transistions
    #the ith element of c1econfig.
        place.append("---")
        place.append(j)
        if i+j <= len(c1econfig):
            transistions.append([c1econfig[i+j-1], c1econfig[i]])
            transistion_names.append([c1econfig_names[i+j-1], c1econfig_names[i]])


#Determine if an energy configuration has nested loops. Use XOR to do so.
#for n in range(len(transistions)):
#This for loop is about calculating the auger energies w/o the core. Notice that the transistions have
#already been figured out according to the above.
auger_energies_wo_core = []
auger_energies_wo_core_onelist = []
auger_energies_wo_core_twolists = []
checktypeor = []
checktypeand = []
checktypenor = []
finalchecktypeor = []
finalchecktypenor = []
types = []
difference = []

for s in range(len(transistions)):
    checktypeor.append(isinstance(transistions[s][0], list) or isinstance(transistions[s][1], list))
```

```
    checktypenor.append(not checktypeor[s])
    checktypeand.append(isinstance(transistions[s][0], list) and isinstance(transistions[s][1], list))
    finalchecktypeor.append(checktypenor[s] or checktypeand[s])
    finalchecktypenor.append(not finalchecktypeor[s])

    if checktypeand[s] == True:
        #print n
        types.append( "Two lists")
        start = len(auger_energies_wo_core_twolists)
#       print start
        for t in range(len(transistions[s][0])):
            for u in range(len(transistions[s][1])):

                auger_energies_wo_core_twolists.append(transistions[s][0][t]-transistions[s][1][u])
        stop = len(auger_energies_wo_core_twolists)
#       print stop
#       print "--------"
#       print s
#       print "--------"
        transistion_names[s].append(auger_energies_wo_core_twolists[start:stop])
    elif finalchecktypenor[s] == True:
        types.append( "One list")
        start = len(auger_energies_wo_core_onelist)
        #If there is one list within a transistion determine which element is the list.
        #Iterate through the list and compute the auger_energies_wo_core_onelist
        if isinstance(transistions[s][0], list):
            for v in range(len(transistions[s][0])):
                auger_energies_wo_core_onelist.append(transistions[s][0][v]-transistions[s][1])
        elif isinstance(transistions[s][1], list):
            for v in range(len(transistions[s][1])):
                auger_energies_wo_core_onelist.append(transistions[s][0]-transistions[s][1][v])
        stop = len(auger_energies_wo_core_onelist)
        transistion_names[s].append(auger_energies_wo_core_twolists[start:stop])
    else:
      start = len(auger_energies_wo_core)
      types.append( "Zero lists")
      auger_energies_wo_core.append(transistions[s][0]-transistions[s][1])
      stop = len(auger_energies_wo_core)
      transistion_names[s].append(auger_energies_wo_core_twolists[start:stop])

#Put the augers_energies_wo_core lists together and remove any negative values
for w in range(len(auger_energies_wo_core_twolists)):
    auger_energies_wo_core.append(auger_energies_wo_core_twolists[w])
for x in range(len(auger_energies_wo_core_onelist)):
    auger_energies_wo_core.append(auger_energies_wo_core_onelist[x])
#Use a while loop with an explicit iterator to avoid issues when auger_energies_wo_core changes size
#Any time an element is popped the element after it could be negative. Therefore it is necessary to
```

```
make
#the progression of the iterator contingent on the element's value being greater than or equal to zero.
y = 0
while y < len(auger_energies_wo_core):
    if auger_energies_wo_core[y] < 0:
        auger_energies_wo_core.pop(y)
    elif auger_energies_wo_core[y] >= 0:
        y= y+1

y1 = 0
y2 = 0
while y1 < len(transistion_names):
    while y2< len(transistion_names[y1][2]):
        if transistion_names[y1][2][y2] < 0:
            transistion_names[y1][2].pop(y2)
        elif transistion_names[y1][2][y2] >= 0:
            y2 = y2 +1
    y1 = y1 +1
    y2 = 0

#Account for the core hole in calculation of the Auger energy
auger_energies = []
for z in range(len(auger_energies_wo_core)):
    for aa in range(len(c1econfig_2se22pe2)):
        auger_energies.append(auger_energies_wo_core[z]-c1econfig_2se22pe2[aa])

auger_energies_w_labels = []
auger_energies_w_labels_calculated = []
for z2 in range(len(transistion_names)):
    auger_energies_w_labels.append(transistion_names[z2][0:2])
for z2 in range(len(transistion_names)):
    start = len(auger_energies_w_labels_calculated)
#    print start
    for ab in range(len(transistion_names[z2][2])):
        for ac in range(len(c1econfig_2se22pe2)):
            auger_energies_w_labels_calculated.append(transistion_names[z2][2][ab]-
c1econfig_2se22pe2[ac])
    stop = len(auger_energies_w_labels_calculated)
#    print stop
#    print "---------"
    auger_energies_w_labels[z2].append(auger_energies_w_labels_calculated[start:stop])

#Take out any negative values from the Auger energy list.
m = 0
while m < len(auger_energies):
    if auger_energies[m] < 0:
        auger_energies.pop(m)
```

```
    elif auger_energies[m] >= 0:
        m= m+1

m1 = 0
m2 = 0
while m1 < len(auger_energies_w_labels):
#    print m1
    while m2 < len(auger_energies_w_labels[m1][2]):
        if auger_energies_w_labels[m1][2][m2] < 0:
            auger_energies_w_labels[m1][2].pop(m2)
        elif auger_energies_w_labels[m1][2][m2] >= 0:
            m2 = m2 + 1
    m1 = m1 + 1
    m2 = 0

#Plot a histogram of Coster-Kronig transitions
plt.hist(auger_energies,bins = 5000, color = 'black')
plt.xlabel('Coster-Kronig Energy (eV)')
plt.ylabel('Counts')
plt.ylim(0,420)


#Write Auger energies to file
f = open('auger_energies', 'w')
for line in range(len(auger_energies)):
    f.write(str(auger_energies[line])+ "\n")
f.close
```

Acknowledgements

A.J. would like to thank the Semiconductor Research Corporation Education Alliance through the Undergraduate Research Opportunity for financial support. A.J. thanks Lester Lampert for his constant mentoring, valuable discussions, and enthusiasm for research. A.J. thanks Andrew Barnum for valuable discussions and his operation of the PHI Versa Probe II for XPS and AES measurements. A.J would also like to thank professor Jun Jiao for the opportunity to be part of her lab group.

References

Bambynek, W. *et al.* X-Ray Fluorescence Yields, Auger, and Coster-Kronig Transition Probabilities. *Rev. Mod. Phys.* **44,** 716–813 (1972).

Bourgoin, D., Turgeon, S. & Ross, G. G. Characterization of hydrogenated amorphous carbon films produced by plasma-enhanced chemical vapour deposition with various chemical hybridizations. *Thin Solid Films* **357,** 246–253 (1999).

Callister, William D. *Materials Science and Engineering: An Introduction.* (Wiley , 2007).

Géraud-Grenier, I., Massereau-Guilbaud, V. & Plain, A. Characterization of particulates and coatings created in a 13.56 MHz radiofrequency methane plasma. *Surface and Coatings Technology* **187,** 336–342 (2004).

Halliday, D., Resnick, R. & Walker, J. *Fundamentals of Physics.* (Wiley, 1997).

Kaciulis, S., Mezzi, A., Calvani, P. & Trucchi, D. M. Electron spectroscopy of the main allotropes of carbon. *Surf. Interface Anal.* **46,** 966–969 (2014).

Kato, T. & Hatakeyama, R. Direct Growth of Doping-Density-Controlled Hexagonal Graphene on SiO2 Substrate by Rapid-Heating Plasma CVD. *ACS Nano* **6,** 8508–8515 (2012)

Kramida, A., Ralchenko, Yu., Reader, J. and NIST ASD Team (2014). *NIST Atomic Spectra Database* (version 5.2), National Institute of Standards and Technology, Gaithersburg, MD.

Kostroun, V. O. Studies in nuclear and atomic physics utilizing high-resolution semiconductor photon spectrometers. (University of Oregon, 1968).

Novoselov, K. S. *et al.* A roadmap for graphene. *Nature* **490,** 192–200 (2012).

Thompson, M, *Auger Electron Spectroscopy*. A Wiley-Interscience publication, 1985.

Vlassiouk, I. *et al.* Role of Hydrogen in Chemical Vapor Deposition Growth of Large Single-Crystal Graphene. *ACS Nano* **5,** 6069–6076 (2011).

Young, Hugh D. *University Physics*, 7th Ed. (Addison Wesley, 1992).