Lab #11: Macroscopic Methods for Measuring Planck's Constant

Jake Waksbaum and Oriel Farajun

Due on Friday, May 12

Abstract

In this paper we present two methods for measuring Planck's constant h. The first, based on earlier work (Zhou and Cloninger 2008), measures the stopping voltage of LEDs to determine the energy of the photons produced and fits the result from many LEDs to the equation $E = h\nu$. The second is novel, and involves measuring the radiation spectrum of the sun using a photodiode and light filters, and fitting it to the blackbody radiation curve predicted by Planck's Law. We measured h to be $6.27810(12064) \times 10^{-34}$ and $7.78680(60495) \times 10^{-34}$ using the two methods respectively.

Introduction

Since it's development in the beginning of the 20th century, quantum mechanics has revolutionized physics by overturning widely held beliefs about the nature of the universe and solving previously intractable problems. At the same time, it shifted the focus of physics from the realm of common experience to the quantum world for which most people have no intuition. Our goal was to measure Planck's constant, one of the fundamental constants of quantum mechanics, by observing familiar phenomena in the macroscopic world.

One of the problems that originally motivated the development of quantum mechanics was the paradox of blackbody radiation. Classical mechanics predicted a blackbody would radiate infinite energy. Planck introduced the idea of intensity of light of wavelength λ radiated by a blackbody at some temperature T that became known as Planck's Law:

$$I_T(\lambda) = \frac{2hc^2}{\lambda^5} \frac{1}{\exp\left(\frac{hc}{k_B T} \frac{1}{\lambda}\right) - 1} \tag{1}$$

where $k_B = 1.38064852 \times 10^{-23}$ is Boltzmann's constant.

Another one of the early successes of quantum mechanics was the explanation of the photoelectric effect due to Einstein. The photoelectric effect is the production of a current due to incident light, and according to classical mechanics the kinetic energy of the electrons should increase with the intensity of the incident light. However, experiments showed that the kinetic energy of the electrons depended on the frequency of the incident light. Einstein explained this phenomenon using the idea of light as discrete quanta called photons, and derived the Planck-Einstein relation for the energy of a photon:

$$E = h\nu \tag{2}$$

where h is the same constant introduced by Planck in explaining blackbody radiation.

Our first method for measuring Planck's constant is related to the photoelectric effect. An LED produces light by a mechanism that is almost the reverse of the photoelectric effect: electrons travel across a voltage drop and emit photons of a specific frequency following the Planck-Einstein relation. The stopping voltage

of an LED is the lowest voltage at which it produces light, and at that voltage the voltage drop across the LED is equal to the voltage drop of the electron and the emitted photon. Therefore we have

$$E = q_e V_{\text{stop}} = h\nu \tag{3}$$

where $q_e = 1.60217662 \times 10^{-19}\,\text{C}$ is the charge on an electron and ν is the frequency.

Determining when an LED just begins to produce light is subjective and error-prone. Instead, one can use an RC circuit in which a capacitor discharges to power an LED (Fig. 1b). Assuming that the voltage drop across the LED is constant, the voltage drop across the capacitor will be given by

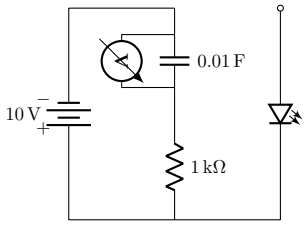
$$V(t) = \left(\frac{Q_0}{C} - V_{\text{stop}}\right) \exp\left(-\frac{t}{RC}\right) + V_{\text{stop}} \tag{4}$$

Fitting voltage data to the form $V = A\exp(Bt) + C$ allows us to determine the stopping voltage, and fitting the stopping voltages versus the wavelengths of the LEDs using the Planck-Einstein relation allows us to determine h.

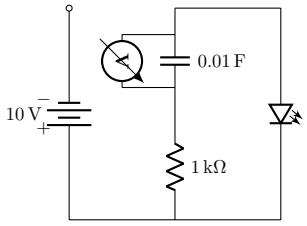
Our second method is related to blackbody radiation. The sun is approximately a blackbody radiating at 5778 K. We attempt to measure its radiation spectrum using a photodiode circuit (Fig. 2) that produces voltage in response to the intensity of incident light. By covering the photodiode with filters that admit only certain bands of light, we can measure the relative intensity of light at different frequencies. Fitting this curve to the curve predicted by Planck's Law allows us to determine h.

Materials and Methods

The circuit shown in Fig. 1 was used to first charge the capacitor (Fig. 1a) and then to discharge the capacitor through the LED (Fig. 1b) for approximately $15\tau = 150\,\text{s}$. The voltmeter was connected to a computer that recorded the voltage drop every 0.01 s. This process was repeated with all of the LEDs available.



(a) Charging



(b) Discharging

Figure 1: LED RC Circuit

The circuit shown in Fig. 2 was connected to a multimeter measuring the voltage difference between V_{out} and ground. The $\pm 15\,\text{V}$ voltages were altered from 9–27 V as the photodiode's generated voltage reached the upper bound determined by the input voltage.

The photodiode was covered with various filters and exposed to the sun's light, and the generated voltage was recorded. The date, time, and latitude and longitude were also recorded for each trial.

The photodiode produces voltage more readily in response to light of different frequencies; to correct for this, the spectral response curve for the photodiode was obtained from the manufacturer and the intensities were divided by the corresponding responsivity.

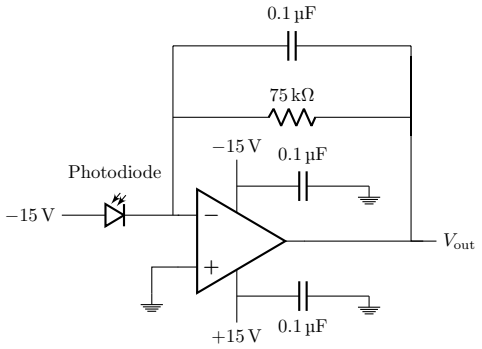


Figure 2: Photodiode Circuit

The atmosphere itself absorbs light of different wavelengths in different ways. To correct for this, we followed the recommendations of Mecherikunnel and Richmond 1980. We used the date, time, and location to calculate the air mass using formulas from *NOAA Solar Calculator* 2005. We used ngstrom turbidity coefficients $\alpha = 1.3$ and $\beta = 0.085$.

Results and Discussion

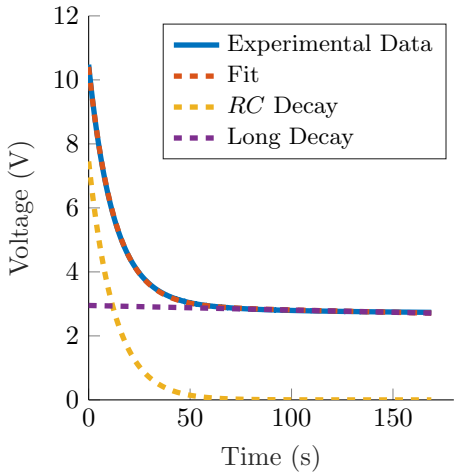
Fig. 3 shows the change in voltage over time for a specific trial. When attempting to fit the data to a decay curve, we first tried a fitting it to a single decay $V = A\exp(Bt) + C$ as predicted by theory. However, we noticed that in-between charging and discharging, the capacitor experienced some voltage leakage, decaying even though it was not attached to any circuit. In addition, plotting the residual yielded a non-random pattern indicating that the fit did not capture some underlying phenomenon occurring in the circuit.

This motivated us to try a fit with two exponentials to account for the leakage in the form $V = A\exp(Bt) + C\exp(Dt)$. We expected D to be very small to correspond to the very slow decay of the leakage. We took C to be the stopping voltage because this is the constant voltage other than the main decay that is present before the leakage; in this way we try to ignore the leakage. However, Fig. 3b shows that the residual is still non-random, similar to the residual with the single exponential fit. This may be due to a non-constant voltage across the LED or to other phenomena we have not considered.

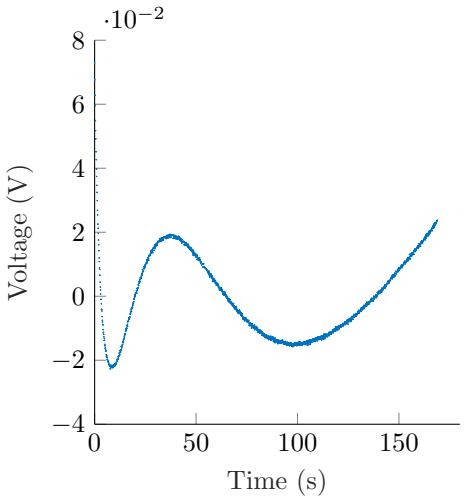
Fig. 4 shows the energy of the light at different frequencies, and has a slope of h. Our experimental value of $h = 6.27810(12064) \times 10^{-34}$ was close to the theoretical value, with only 6.04 % error. Perhaps voltage leakages from the capacitor caused a divergence from the ideal RC circuit that our fit didn't capture, yielding some error in the fit's approximation and in the calculation of Planck's constant. We assumed that the voltage drop across the LED was constant, which could also have resulted in poor fits. In addition, a definitive source of error arises from relying on the wavelengths of the LEDs reported by the manufacturer.

Fig. 5 shows the blackbody radiation curve generated by our data. From the fit, we computed $h = 7.78680(60495) \times 10^{-34}$. This is within 17.52 % of the accepted value, which is better than we had hoped for a method with so many possible sources of noise. Possible sources of error include imperfect blocking of external light, imprecise orientation of the filter resulting in reduced light intensity, errors in correcting for the spectral absorption of the photometer and atmospheric absorption, and deviations in the sun's true radiation spectrum from an ideal blackbody at 5778 K.

Table 1 compares the measured values of Planck's constant from both methods with the accepted value. We can see that the LED method was more accurate, which is consistent with our expectations: the LED



(a) Experimental data with fit to $V = A\exp(Bt) + C\exp(Dt)$ with $r^2 = 0.9999$. *RC decay* shows just the first exponential $A\exp(Bt)$ while *Long Decay* shows the second exponential $C\exp(Dt)$.



(b) Residual of $V = A\exp(Bt) + C\exp(Dt)$ fit.

Figure 3: Double Exponential Fit: Data from 400 nm trial

method is simpler than the solar radiation method because relies on our ability to model electronics in the lab, while the solar radiation depends on our ability to model the sun, the atmosphere, and the photodiode.

	h (J s)	Percent Error
LED	$6.27810(12064) \times 10^{-34}$	6.04 %
Sun	$7.78680(60495) \times 10^{-34}$	17.52 %

Table 1: Computed Values for Planck's Constant

Conclusion

Inherent in both the electron and in the sun, in the atomic and in the celestial, can be found one of the fundamental constants of the universe: Planck's constant. We used two methods of experimentally determining this constant that were related to two of the earliest successes of the quantum theory. The LED method proved to be more successful than the blackbody method, which we ascribe to the controlled environment in which it was performed; the blackbody experiment had many more uncontrolled variables. We consider it a significant success that that in both experiments the error was below 20 %. To improve our results for the LED method, we would first measure the frequencies of the LEDs directly, since the frequencies listed by the manufacturer as the frequencies of the LEDs themselves have errors. In addition, we would try to correct for any voltage leakage in the capacitor and LED to more nearly approximate an ideal RC-circuit. To improve our results for the blackbody experiment, we would build an apparatus that would secure the photometer and the filter to point in the same angle so that all the light transmitted through the filter would hit the photometer directly.

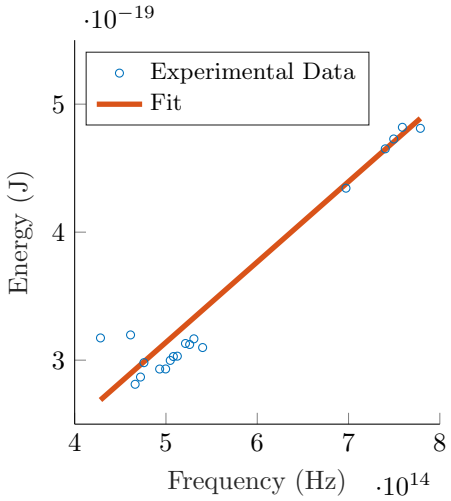


Figure 4: Energy vs. Frequency: Stopping voltages are the constant C obtained from the fits to the voltage data and wavelengths are the manufacturer reported wavelengths of the LEDs. Fit is linear with $r^2 = 0.8664$.

Acknowledgments

Thanks to Jenn for obtaining so many LEDs, for allowing us to use the filters in the filter cubes. Thanks also to Quan for providing us with more filters from his lab and for discussing how to deal with leakage in the capacitor.

This paper represents my own work in accordance with University regulations – /s/ Jake Waksbaum and Oriel Farajun

References

(1) CODATA Value: Boltzmann constant., `http://physics.nist.gov/cgi-bin/cuu/Value?k`.

(2) CODATA Value: elementary charge., `http://physics.nist.gov/cgi-bin/cuu/Value?e`.

(3) CODATA Value: Planck constant., `http://physics.nist.gov/cgi-bin/cuu/Value?h`.

(4) CODATA Value: speed of light in vacuum., `http://physics.nist.gov/cgi-bin/cuu/Value?c`.

(5) Mecherikunnel, A. T.; Richmond, J. *Spectral distribution of solar radiation*; Technical Memo TM-82021; Greenbelt, MD, United States: NASA Goddard Space Flight Center, 1980.

(6) Meeus, J., *Astronomical algorithms*; Richmond, Va.: Willmann-Bell: 1998; Vol. 2nd ed., 2nd English ed.

(7) NOAA Solar Calculator., `https://www.esrl.noaa.gov/gmd/grad/solcalc/`.

(8) The Editors of Encyclopdia Britannica photoelectric effect., `https://www.britannica.com/science/photoelectric-effect`.

(9) Zhou, F.; Cloninger, T. *The Physics Teacher* **2008**, *46*, 413–415.

Supplementary Information

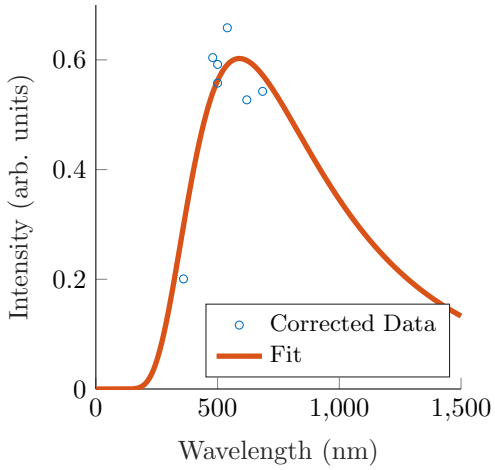


Figure 5: Solar Emission Spectrum: *Corrected Data* shows the intensity data after corrections for the photodiode spectral response and atmospheric absorption. The fit represents a fit to the shape of the curve in Planck's law, $I(\lambda) = A\lambda^{-5}\big(\exp\big(B\lambda^{-1} - 1\big)\big)^{-1}$ and has $r^2 = 0.8152$.

```
%% Settings
% Disable text interpreter, that will be handled by LaTeX
set(groot, 'defaultAxesTickLabelInterpreter','none');
set(groot, 'defaultLegendInterpreter','none');
set(groot, 'defaultTextInterpreter','none');
% Hide figures
visible = 'off';

%% Data Analysis
% Read location coordinates
latlong = readGeoPoint(fullfile('data', 'sun', 'latlong.txt'));
% Read wavelength intensities
data = readtable(fullfile('data', 'sun', 'sun.csv'));
% Read spectral response curve
specResponse = readtable(fullfile('data', 'sun', 'specresponse.csv'));

% Convert to actual datetimes
data.DateTime = parseDateISO(data.DateTime, '-04:00');
% Calculate the airmass
data.AirMass = airMass(data.DateTime, latlong);
% Normalize for width of wavelength range passed by filters
data.VoltagePerNM = data.Voltage ./ data.Width;
% Calculate the spectral response factors for the wavelengths
responses = findSpecResponse(specResponse, data.Wavelength);
% Correct for spectral response
data.PhotodiodeCorrected = data.VoltagePerNM ./ responses;

% Calculate coefficients c1 and c3
[c1,c3] = getC13(data.Wavelength);
data.C1 = c1;
data.C3 = c3;
% Correct for atmospheric absorption
data.AtmosphereCorrected = correctAtmosphere(data,data.PhotodiodeCorrected);
```

```

% Calculate h from fitting to blackbody radiation
[h, fitresult, gof, stdev] = calcSunH(data);
% Compare with accepted value
err = percentErr(h, Constants.H);

%% Figures
% Atmospheric Correction and Fit
fig = figure('Visible', visible);
hold on
scatter(data.Wavelength, data.AtmosphereCorrected);
w = linspace(0, 1500, 1000);
i = (fitresult.a .*1e-29 ./ (w.*1e-9).^5) .*...
    (1./(exp(fitresult.k./(w.*1e-9)) - 1));
plot(w, i, 'LineWidth', 2)
xlabel('Wavelength (\si{\nano\meter})')
ylabel('Intensity (arb. units)')
legend('Corrected Data', 'Fit', 'Location', 'SouthEast')
saveFig(fig, 'sun.tex', '2in', '2in')
```

```
function [h, data, fitresult, gof] = calcLEDH(data)
%CALCLEDH Fits a line to E = hf to determine h
data.Freq = Constants.C ./ (data.Wavelength * 1e-9);
data.Energy = Constants.E .* data.VF;
ft = fittype(@(c,x) c .* x);
[fitresult,gof] = fit(data.Freq, data.Energy, ft, 'StartPoint', 1e-34);
h = fitresult.c;
end
```

```
function vf = getVF(trial)
%GETVF Calculate the stopping voltage by fitting to two exponentials
options = optimset('Display', 'off');
fitfunc = @(c,t) c(1) .* exp(-t./c(2)) + c(3)*exp(-t./c(4));
startpoints = [7 10 1 1e4];
[coeffs,~,~] = lsqcurvefit(fitfunc, startpoints,    ...
                           trial.Time, trial.Voltage, ...
                           [], [], options);
vf = coeffs(3);
end
```

```
%% Settings
% Disable text interpreter, that will be handled by LaTeX
set(groot, 'defaultAxesTickLabelInterpreter','none');
set(groot, 'defaultLegendInterpreter','none');
set(groot, 'defaultTextInterpreter','none');
% Hide figures
visible = 'off';

%% Data Analysis
% Get a list of all the data files
files = dir(fullfile('data', 'leds', '*.txt'));

% For each of the trials...
trials = struct();
```

```
for i=1:length(files)
    % Get the path to the file
    path = fullPath(files(i));
    % Parse the file name
    [~,wavelengthStr,~] = fileparts(path);

    % Convert file name to a number to get the wavelength
    trials(i).wavelength = str2double(wavelengthStr);
    % Read the data file and store it
    trials(i).data = readtable(path);
end

% Create a table of wavelength vs stopping voltage
wavelengthVF = vfTable(trials);

% Calculate h by fitting E = hf
[h, wavelengthVF, fitresult, gof] = calcLEDH(wavelengthVF);

% Calculate standard deviation from confidence interval
stdev = diff(confint(fitresult)) ./ 4;

% Compare with accepted value
err = percentErr(h, Constants.H);

%% Figures

figWavelegnth = trials(3).wavelength;
figData = trials(3).data;
t = linspace(min(figData.Time), max(figData.Time), 1000);

% Double Exponential Fit
fig3 = figure('Visible', visible);
hold on
plot(figData.Time, figData.Voltage, 'LineWidth', 2)
fitfunc2 = @(a,b,c,d,x) a .* exp(-x./b) + c .* exp(-x./d);
startpoint = [7 10 1 1e4];
ft = fittype(fitfunc2);
[fitresult, gof] = fit(figData.Time, figData.Voltage, ft, ...
    'StartPoint', startpoint);
coeffs = [fitresult.a fitresult.b fitresult.c fitresult.d];
fitVals = fitfunc2(coeffs(1), coeffs(2), coeffs(3), coeffs(4), t);
plot(t, fitVals, '--', 'LineWidth', 2)
plot(t, coeffs(1) .* exp(-t./coeffs(2)), '--', 'LineWidth', 2)
plot(t, coeffs(3) .* exp(-t./coeffs(4)), '--', 'LineWidth', 2)
xlabel('Time (\si{\second})')
ylabel('Voltage (\si{\volt})')
legend('Experimental Data', 'Fit', '$RC$ Decay', 'Long Decay')
saveFig(fig3, 'complex_fit.tex', '2in', '2in')

% Double Exponential Residual
fig4 = figure('Visible', visible);
resid = figData.Voltage - fitfunc2(coeffs(1), coeffs(2), coeffs(3), coeffs(4),
    figData.Time);
```

```
scatter(figData.Time(1:10:end), resid(1:10:end),1,'x')
xlabel('Time (\si{\second})')
ylabel('Voltage (\si{\volt})')
saveFig(fig4, 'complex_resid.tex', '2in', '2in')

% E = hf
fig5 = figure('Visible', visible);
hold on
scatter(wavelengthVF.Freq, wavelengthVF.Energy)
f = linspace(min(wavelengthVF.Freq), max(wavelengthVF.Freq), 1000);
plot(f, h.*f, 'LineWidth', 2)
xlabel('Frequency (\si{\hertz})')
ylabel('Energy (\si{\joule})')
legend('Experimental Data', 'Fit', 'Location', 'NorthWest')
saveFig(fig5, 'evsf.tex', '2in', '2in')
```

```
function data = vfTable(trials)
%VFTABLE Calculates the stopping voltage for each trial and creates a table
% of wavelength vs. stopping voltage
data = table;
data.Wavelength = [trials.wavelength]';
data.VF = cellfun(@(trial) getVF(trial), {trials.data})';
end
```

```
function dt = parseDateISO(dateStr, tz)
%PARSEDATEISO Creates a Matlab datetime object from a datetime string in
%ISO format, for example 2000-01-01T12:00
dt = datetime(dateStr, 'InputFormat', 'uuu-MM-dd''T''HH:mm', 'TimeZone', tz);
end
```

```
classdef Constants
    %CONSTANTS Physical constants and accpeted values
    properties (Constant)
        E = 1.6021766208e-19; % Electron charge
        H = 6.62607004e-34;   % Planck's constant
        C = 299792458;        % Speed of light
        SUNTEMP = 5778;       % Temprature of the Sun in Kelvin
        KB = 1.38064852e-23;  % Boltzman constant
    end
end
```

```
function m = airMass(dt, latlong)
%AIRMASS Calculate the air mass from the date, time, and location

% Calculate the hour angle and solar declination angle
[ha, da] = hourDecAngle(dt, latlong.Longitude);
da = deg2rad(da);
ha = deg2rad(ha);

theta = deg2rad(latlong.Latitude); % Latitude angle

% Calculate the air mass
m = 1 ./ (sin(theta) .* sin(da) + cos(theta) .* cos(da) .* cos(ha));
end
```

```
function [h, fitresult, gof, stdev] = calcSunH(data)
%CALCSUNH Calculate h by fitting to the shape of a blackbody radiation
%curve at the sun's temperature
wavelength = data.Wavelength .* 1e-9;        % Convert to SI nm
intensity = data.AtmosphereCorrected * 1e29; % Scale up to help fit
[xData, yData] = prepareCurveData(wavelength, intensity);

% Set up fittype and options.
ft = fittype( '(a./x.^5) .* (1./(exp(k./x) - 1))', 'independent', 'x', 'dependent'
    , 'y' );
opts = fitoptions('Method', 'NonlinearLeastSquares');
opts.Display = 'Off';
opts.StartPoint = [2.5 1e-06];

% Fit model to data.
[fitresult, gof] = fit( xData, yData, ft, opts );
h = fitresult.k .* Constants.KB .* Constants.SUNTEMP ./ Constants.C;
% Calculate standard deviation from confidence interval
stdev = diff(confint(fitresult)) ./ 4 * Constants.KB .* Constants.SUNTEMP ./
    Constants.C;
end
```

```
function corrected = correctAtmosphere(data, intensity)
%CORRECTATMOSPHERE Correct for the atmospheric absorption of different
%wavelengths

c1 = data.C1;
alpha = 1.3;
beta  = 0.085;
c2 = beta .* (data.Wavelength*1e3) .^(-alpha);
c3 = data.C3;
m = data.AirMass;

corrected = intensity ./ exp(-(c1 + c2 + c3).*m);
end
```

```
function responses = findSpecResponse(specResponse, wavelengths)
%FINDSPECRESPONSE Returns the spectral response factor corresponding to the
%given wavelengths
findingFunc = @(w) find(round(specResponse.Wavelength) == w, 1);
matchingIndices = arrayfun(findingFunc, wavelengths);
responses = specResponse.Responsivity(matchingIndices);
end
```

```
function [c1, c3] = getC13(wavelength)
%GETC13 Calculate c1 and c3 by fitting given data with splines and
%sampling the fit and desired wavelength
data = readtable(fullfile('data', 'sun', 'c13.csv'));

ppc1 = csaps(data.Wavelength, data.C1, 0.31673854109269317); % c1 fit
ppc3 = csaps(data.Wavelength, data.C3, 0.9);                 % c3 fit

% Sample fits at desired wavelength
c1 = fnval(ppc1, wavelength);
```

```
11  c3 = fnval(ppc3, wavelength);
12  end
```