Numerical Analysis for Computer Scientists
FMN011, Lund University 2012
Project #2
Finding the line strength of stars for an astronomer

Erik Westrup, <ada09ewe@student.lu.se>

May 24, 2012

1 Introduction and Problem Background

This project is about solving real-world problems with their accompanied complexities and ambiguities. The best method must be determined and its limitations and possible errors must be known. In this project intensity of light from stars will be studied. Stars produces a continuous curve of light over a range of frequencies called the *spectrum* for that star. It can be measured in *relative intensity* ($\frac{\mathrm{W}}{\mathrm{m}^2 \times \mathrm{Hz}}$) as a function of the frequency (Hz) which is defined as the spectrum. We want to measure in relative intensity because the energy is higher at higher frequencies and that known property is not the one we want to study. Therefore we normalize it with the frequency.

What is interesting is that the surrounding gases of a star will either absorb (cold gas) or emit (warm) light which is visible at discrete fixed frequencies in the spectrum as *spectral lines*. By identifying these spectral lines the components of the star can be found because each chemical composition have a unique fingerprint of discrete frequencies [1] [2].

In this project a measured continuous spectrum for a star is given which contains six spectral lines evenly divided in absorption lines and emissions lines. The task is to find the total intensity for each of these spectral lines called the *line strength* measured in $\frac{\mathrm{M}}{\mathrm{m}^2}$ [3]. This is achieved by finding the area under the curve at the peak-frequencies i.e. integrating the relative intensity over these frequencies.

The areas can not be found using analytical methods since we don't have the real spectrum and spectrum line functions – just observations from it. We must find a way of finding the area from just these observations.

The given data is a file with 4000 measurements of *Specific Intensity* in $\frac{\mathrm{W}}{\mathrm{m}^2 \times \mathrm{Hz}}$ at frequencies in the range $[3.50 \times 10^{14}, 4.29 \times 10^{14}]$.

2 Numerical Considerations

For this problem there was no hints on what methods to be used. Therefore an iterative approach was used to find methods that works. The process will be describe in section 3. As described there, I used some *MATLAB*® features including polynomial fitting in the least squares sense with the command *polyfit* and estimation of areas using the trapezoidal method with the command *trapz*. As later described there was a problem with an ill-conditioned matrix which was solved my normalization.

3 Results & Analysis

All code used to get the results can be seen in Appendix B. The first thing to do is, as always, to get an understanding of what are the given. Since studying the raw data is hard a graphical plot over the curve is desired to grasp the main characteristics of what we have. The relative intensity spectrum can be seen in figure 1 and the spectrum itself in figure 2. Looking at the spectrum (no the relative) and compare to the spectrum lines in [1] we see that it looks like the curve for body with temperature 4000 Kelvin for the human visible wavelengths. In these two plots the six spectral lines are clearly visible. Just because it is interesting, the intensities are also show as a function of the wave lengths in figure 3 using the *MATLAB*® function *spectrumLabel* [4].

The first naïve thing to do is to see if we can fit a curve the data directly. Just by trying out some, one seen in figure 4, we can conclude that no polynomial can fit both the spectrum and the spectrum lines and give a good representation of the real functions. From this the main idea for solving this project was developed. We want to find a function representing the spectrum and then approximate the area between this function and the peaks/dips.

To find this function we must first remove the distracting points from the spectrum lines and extrapolate the missing parts of the spectrum line. By plotting the data points as a function of their index in a graph the beginning, midpoints and endpoints for peaks/dips could be found. One had to zoom in very much in the graph to really see the hidden details of the slope. From these points some polynomial fittings was done as seen in figure 5. This of course depends on eye measurements which is not exact so I made sure to have some small marginals by removing possibly more points than needed. This is not a problem since this area is small and will not affect the end results. There are probably a better way of finding these points but I considered that this would require too much effort and that this method was good enough.

It was found that a polynomial of degree 4 followed the curve good enough. Higher degrees did not to much better. The tricky point is the small and steep segment between the first dip and the second peak. Because of the large variety in the frequencies the matrix A used by *polyfit* is ill-conditioned $cond(A^T \times A = 1.681060E + 27)$ giving results that can not be trusted the frequencies had to be normalized before fitting.

For all but two spectrum lines we can now calculate the area by using the trapezoidal method by giving the start and end frequencies and the corresponding difference between the fitted curve and the measure relative intensities to the $MATLAB^{®}$ command *trapz*. The results can be seen in table 1.

Spectral Line	Area (full)
1	3.555511E+06
2	5.030370E+06
3	1.586236E+06
4+5	2.127647E+07
6	3.250223E+06

Table 1: Full areas for the spectrum lines (double dip excluded).

But what about the double dip? We're missing one boundary value for each of the dips and can thus not calculate the areas. Therefore we now assume that the distribution of spectrum frequencies are symmetric. By looking at the peaks/dips that we have we can see that they all seems to follow this. If we assume this the areas for the two dips can be found by finding the area for half of the line and multiply it by two. This was done for all points, in figure 2, so the results can be compared to the one in table 1

It is interesting to compare the full areas and the areas found by multiplying by two. So the infinity norm of the difference in areas (double dip excluded) was found to be $4.091669E + 05$ and was for the last peak, spectral line 6. If we divide that with the corresponding more accurate area (the one found by using the whole known interval) we get that the quota in percent is 12.589. This must be considered high, also relative to the other differences but is has an explanation. By zooming in on the last peak (figure 6) we can see that there is no data point in the middle and the closes two points are not laid out symmetrically around the peak. Therefore it is hard to know where the mid point actually is. I chose to to take the point closest to the real top and not spend time on approximating the real one. This is because there is no real need to find the area by by using the symmetry for this point since both end and start

Spectral Line	Area (symmetry method)
1	3.546553E+06
2	5.039057E+06
3	1.576005E+06
4	1.416181E+07
5	9.521223E+06
6	2.841056E+06

Table 2: Areas found by calculating half the area and multiplying by two.

points are known. The only reason for calculating the area in this way is for reference but we have 3 other peaks/dips that could be used to compare the full area with the symmetry methods that have clear points defining the middle.

But for the two dips with missing start/end points we some how have to guess leading us to approximate by calculate the half we know. There are of course other methods that might give worse or better guesses. But all boils down to making smart guesses – we can not recover unknown information.

The difference between the full double dip area and sum of the symmetrically found individual areas are $2.406557E + 06$. The sum is 11.311% larger than the full double dip area. This difference is of course excepted since the areas overlaps for the two dips leading to a smaller area for the combined full dip.

For all results above we must know the all are affected by approximations and can not be taken for being exact. The first source of error is when I select begin and endpoints for the peaks/dips with eye-measurements. Comparison with other the area results from friends that chose similar but not the exactly the same points gave no noticeable difference in the results. So hopefully the choose of points does not affect so much be is still a source of error. The curve fitting of the remaining points is also not perfect introducing errors in the final area difference. The trapezoidal method is also an approximation so it does not give the exact area since we only have a finite set of data points to work with. For the two dips that overlaps can possibly have some more errors since half of the area is doubled to get the total area. If there is a large error in the half area (caused by any of the two previously mentioned sources) it will be doubled.

To sum up, the final result for this project are the areas $1 - 3, 6$ from table 1 and areas $4 - 5$ from table 2 show in table 3.

Spectral Line	Area
1	3.555511E+06
2	5.030370E+06
3	1.586236E+06
4	1.416181E+07
5	9.521223E+06
6	3.250223E+06

Table 3: The final areas forming the answer for this project.

4 Lessons Learned

In this project I practiced on solving real problems as an engineer by combining previously learned knowledge in this and other courses to solve new types of problems. It was fun and instructive to finding methods of solving the problem. I've also learned to use some new tools in *MATLAB*® like *trapz*, *setdiff* and how to import data from proprietary formats like *Microsoft Excel*®. Also I've practiced on normalizing to avoid ill-conditioning.

5 Acknowledgments

I worked tightly with Oscar Olsson, Tommy Ivarsson and Jonas Klauber when finding out the overall solution method. I also discussed the report with Fredrik Petterson and Simon Thörnqvist.

References

[1] N. Strobel, "Production of light." `http://www.astronomynotes.com/light/s4.htm`. Accessed 2012-04-21.

[2] N. Strobel, "Discrete spectrum." `http://www.astronomynotes.com/light/s5.htm`. Accessed 2012-04-21.

[3] S. D. S. Survey, "Spectral types." `http://cas.sdss.org/dr6/en/proj/advanced/spectraltypes/`. Accessed 2012-04-21.

[4] J. Mather, "Spectral and xyz color functions." `http://www.mathworks.com/matlabcentral/fileexchange/7021-spectral-and-xyz-color-functions`. Downloaded 2012-04-21.

Appendix

A Figures

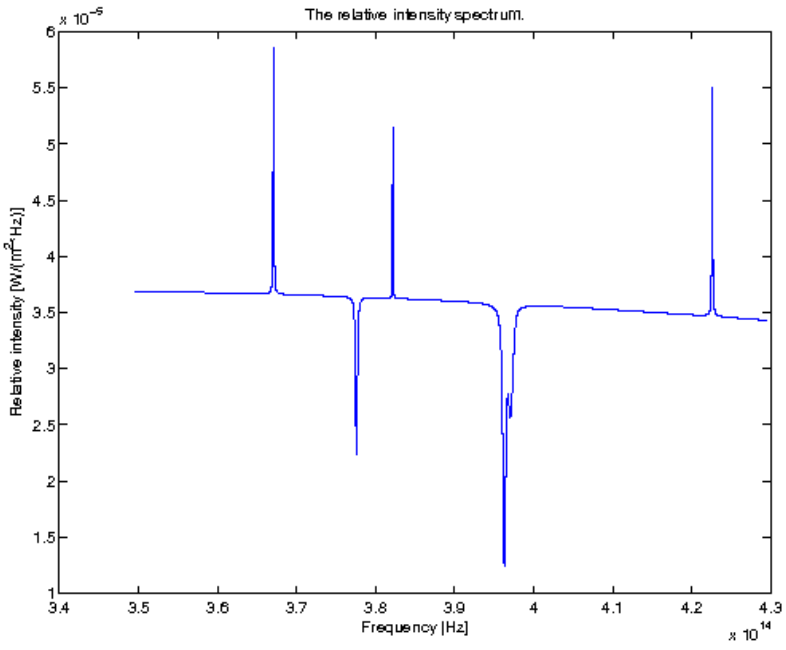

Figure 1:

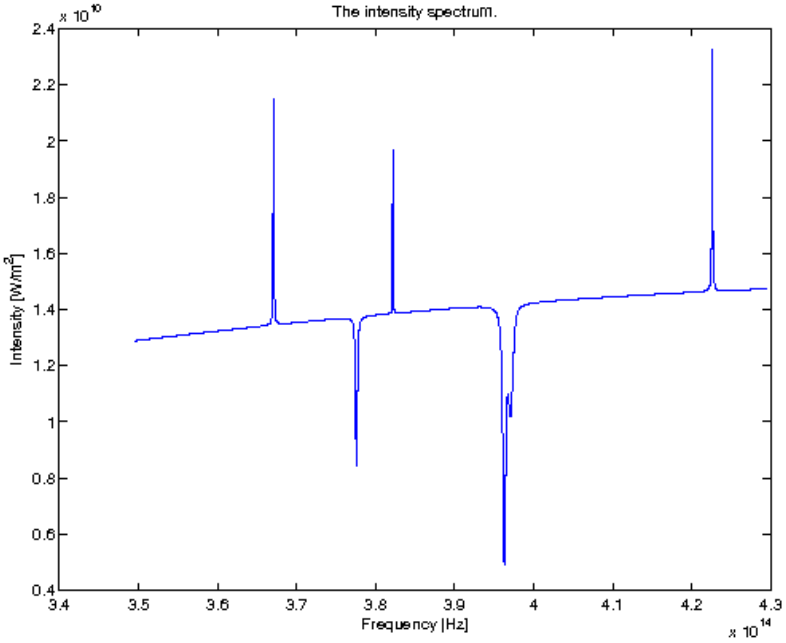

Figure 2:

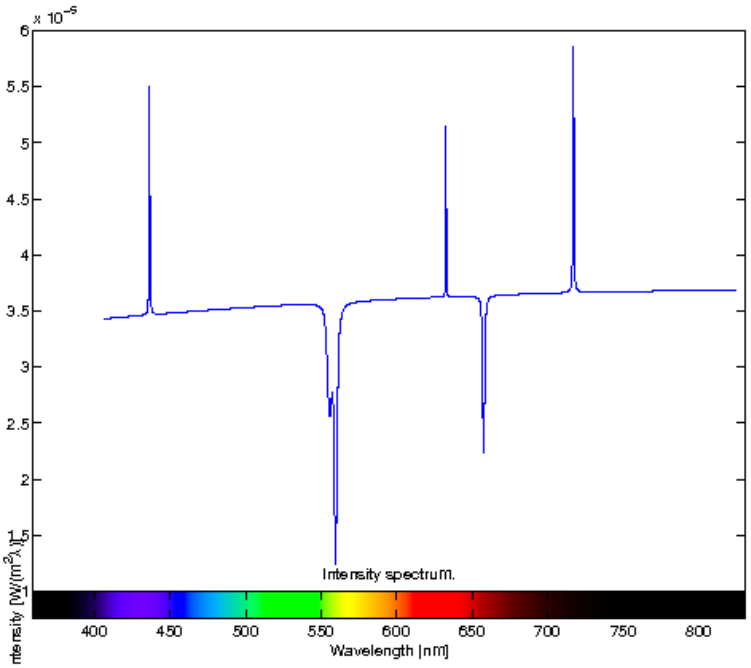

Figure 3:

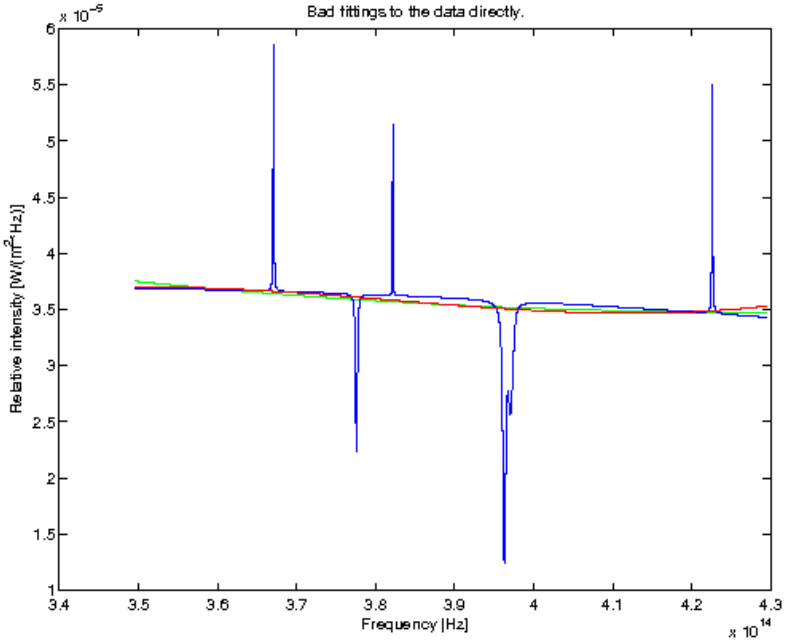

Figure 4:

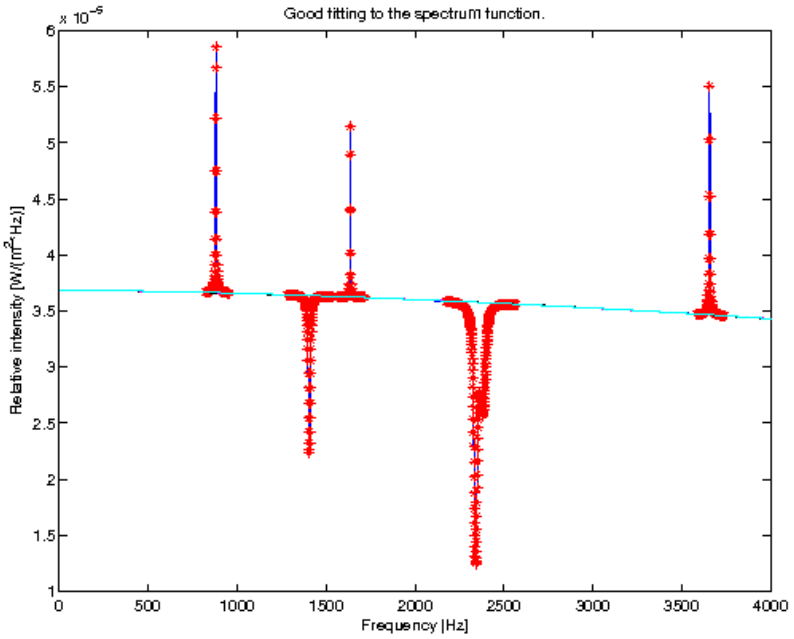

Figure 5:

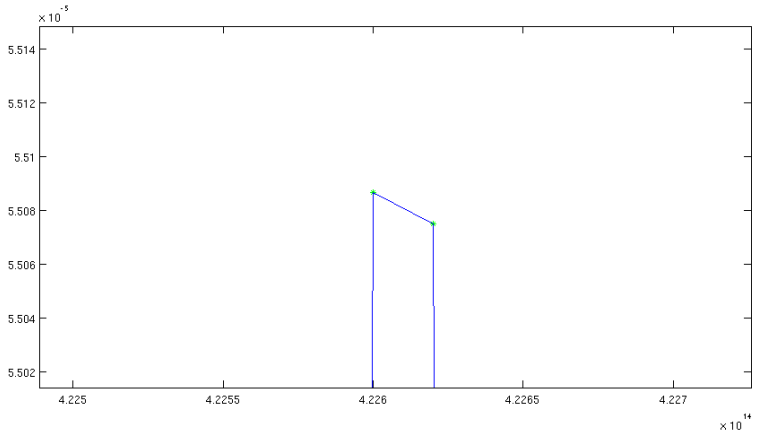

Figure 6: The two highest data points for the spectral line.

B Program listings

Here the *MATLAB*® functions and scripts used to achieve the results above are listed.

B.1 Scripts

The following scrips are the drivers for producing the results found in this report.

../src/pr2.m

```
clc % Clear command screen.
format long % Format of floating point numbers.
%format short e % Don't factor our common parts of numbers in
    matrices.
%clf % Clear current figure
close all  % Close all figures.
fprintf(1, '-->Project #2.\n');
clear all
%warning on verbose % See MSGIDs for warnings.
%global d_freq, d_intens;
[ d_freq,  d_intens ] = pr2.import_data('../spectrum_data.xls');

% Plot relative spectrum.
fig = figure('visible','off'); % Don't display the plot.
plot_relspectrum = plot(d_freq, d_intens, 'b');
xlabel('Frequency [Hz]')
ylabel('Relative intensity [W/(m^2*Hz)]')
title('The relative intensity spectrum.')
saveas(plot_relspectrum, '../img/spectrum_relative.eps', 'eps')
saveas(plot_relspectrum, '../img/spectrum_relative.png', 'png')
set(fig ,'visible','on') % Enable plots again.
close(fig);

% Plot spectrum
fig = figure('visible','off'); % Don't display the plot.
plot_spectrum = plot(d_freq, d_freq .* d_intens, 'b');
xlabel('Frequency [Hz]')
ylabel('Intensity [W/m^2]')
title('The intensity spectrum.')
saveas(plot_spectrum, '../img/spectrum.eps', 'eps')
saveas(plot_spectrum, '../img/spectrum.png', 'png')
set(fig ,'visible','on') % Enable plots again.
close(fig)

% Plot wavelengths spectrum.
fig = figure('visible','off'); % Don't display the plot.
ax = axes();
d_wavelen = 2.998e8 ./ d_freq; % wavelength = c / freq.
%d_wintens = d_wavelen .* d_intens;
plot_wave = plot(d_wavelen, d_intens, 'b');
spectral_color.spectrumLabel(ax);
xlabel('Wavelength [nm]')
ylabel('Intensity [W/(m^2\lambda)]')
title('Intensity spectrum.')
saveas(plot_wave, '../img/spectrum_wave.eps', 'eps')
saveas(plot_wave, '../img/spectrum_wave.png', 'png')
set(fig ,'visible','on') % Enable plots again.
close(fig)

% Try to fit polynomials to the data directly.
fig_badfit = figure('visible','off'); % Don't display the plot.
plot_badfit = plot(d_freq, d_intens, 'b');
```

```
52 hold on
53 xlabel('Frequency [Hz]')
54 ylabel('Relative intensity [W/(m^2*Hz)]')
55 title('Bad fittings to the data directly.')
56
57
58 warning off MATLAB:polyfit:RepeatedPointsOrRescale  % I know about
       the problem.
59 badfit2 = polyfit(d_freq, d_intens, 2);
60 warning on MATLAB:polyfit:RepeatedPointsOrRescale
61 badfit2_v = polyval(badfit2, d_freq);
62 plot(d_freq, badfit2_v, 'g')
63 plot_badfit = plot(d_freq, d_intens, 'b');
64 hold on
65
66 warning off MATLAB:polyfit:RepeatedPointsOrRescale
67 badfit3 = polyfit(d_freq, d_intens, 3);
68 warning on MATLAB:polyfit:RepeatedPointsOrRescale
69 badfit3_v = polyval(badfit3, d_freq);
70 plot(d_freq, badfit3_v, 'r')
71
72 saveas(plot_badfit, '../img/bad_fits.eps', 'eps')
73 saveas(plot_badfit, '../img/bad_fits.png', 'png')
74 set(fig_badfit ,'visible','on') % Enable plots again.
75 close(fig_badfit)
76
77
78
79 %% Try to find a fit for the underlying spectrum function by
       deleting the peaks and dips.
80 % Plot relative spectrum.
81 d_indices = 1:length(d_freq); % Plot by index for easier
       adjustments later.
82 fig_goodfit = figure('visible','off');
83 %fig_goodfit = figure();
84 plot_goodfit = plot(d_indices, d_intens, 'b');
85 hold on
86 xlabel('Frequency [Hz]')
87 ylabel('Relative intensity [W/(m^2*Hz)]')
88 title('Good fitting to the spectrum function.')
89
90 peaks = []; % Row i is peak/dip i counted from the left with col1
       the left border and col2 the right border of the peak/dip.
91 % Found graphically with MATLABs data cursor in plots.
92 peaks(end+1,:) = [820 960];
93 peaks(end+1,:) = [1275 1540]; % Watch out for right boundary, deep
       zoom reveals heavy slope!
94 peaks(end+1,:) = [1560 1720];
95 peaks(end+1,:) = [2175 0];
96 peaks(end+1,:) = [0 2575];
97 peaks(end+1,:) = [3580 3745];
98
99 midpoints = [];
100 midpoints(end+1) = [881];
101 midpoints(end+1) = [1406];
102 midpoints(end+1) = [1636];
103 midpoints(end+1) = [2341];
104 midpoints(end+1) = [2380];
105 midpoints(end+1) = [3656];
106
107
108 spectrum_indices = [1:peaks(1,1)]; % Indices excluding the peaks
```

```
    and dips.
for i = 2:length(peaks)
  if i ~= 5 % No segment between these two dips.
    spectrum_indices = [spectrum_indices peaks(i-1,2):peaks(i,1)];
  end
end

spectrum_indices = [spectrum_indices peaks(length(peaks),2):
    d_indices(end)];
peak_indices = setdiff(d_indices, spectrum_indices);
spectrum_v = d_intens(spectrum_indices)';
peak_v = d_intens(peak_indices);
%plot(spectrum_indices, spectrum_v, 'g*'); % Plot the spectrum
    points.
plot(peak_indices, peak_v, 'r*'); % Plot points in the peak ranges,
     now, adjust the boudaries manually.

%% Try fitting some lines to the spectrum.
spectrum_line2 = polyfit(spectrum_indices, spectrum_v, 2);
spectrum_line2_v = polyval(spectrum_line2, spectrum_indices);
plot(spectrum_indices, spectrum_line2_v, 'k') % black

warning off MATLAB:polyfit:RepeatedPointsOrRescale  % I know about
    the problem now.
spectrum_line3 = polyfit(spectrum_indices, spectrum_v, 3);
warning on MATLAB:polyfit:RepeatedPointsOrRescale
spectrum_line3_v = polyval(spectrum_line3, spectrum_indices);
plot(spectrum_indices, spectrum_line3_v, 'm') % magenta


warning off MATLAB:polyfit:RepeatedPointsOrRescale
spectrum_line4 = polyfit(spectrum_indices, spectrum_v, 4);
warning on MATLAB:polyfit:RepeatedPointsOrRescale
%spectrum_line4 = polyfit((spectrum_indices - mean(spectrum_indices
    )) / std(spectrum_indices), spectrum_v, 4);
spectrum_line4_v = polyval(spectrum_line4, spectrum_indices);
plot(spectrum_indices, spectrum_line4_v, 'c') % cyan

% Check conditioning, see Sauer p.203.
A = [ones(length(spectrum_indices),1) spectrum_indices'
    spectrum_indices.^2' spectrum_indices.^3' spectrum_indices
    .^4'];
condition = cond(A'*A); % Ill conditioned!
fprintf(1, 'The conditon of A for the normal eqation for a
    polynomial fit of degree %i is %E\n', 4, condition);

% Centering and normalization to solve bad condition.
% Normalize frequencies.
d_freq_norm = (d_freq - mean(d_freq)) / std(d_freq);

saveas(plot_goodfit, '../img/spectrum_goodfit.eps', 'eps')
saveas(plot_goodfit, '../img/spectrum_goodfit.png', 'png')
set(fig_goodfit ,'visible','on') % Enable plots again.
close(fig_goodfit);
close all

spectrum_line = polyfit(d_freq_norm(spectrum_indices), d_intens(
    spectrum_indices), 4);
line_v = polyval(spectrum_line, d_freq_norm);
peak_intens = d_intens - line_v; % We want the area between the
    curves.

```

```
% Assume symmetrical spectral lines.
half_areas = zeros(6,1);
for i = 1:length(peaks)
  if i == 5 % We only have right start point
    half_areas(i) = trapz(d_freq(midpoints(i):peaks(i,2)),
        peak_intens(midpoints(i):peaks(i,2)));
  else
    half_areas(i) = trapz(d_freq(peaks(i,1):midpoints(i)),
        peak_intens(peaks(i,1):midpoints(i)));
  end
end

areas_symm = abs(2 .* half_areas);
areas_symm_str = sprintf('\t%E\n', areas_symm);
fprintf(1, 'Areas in W/m^2 found using symmetry is: \n[\n%s]\n',
    areas_symm_str);

% Calculate full areas for the 4 peaks we have left and right
    border for. The double dip (index 4) is couned as one dip.
full_areas = zeros(6,1);
for i = 1:length(peaks)
  if ~(i == 4 || i == 5) % Skip the double dip.
    full_areas(i) = trapz(d_freq(peaks(i,1):peaks(i,2)),
        peak_intens(peaks(i,1):peaks(i,2)));
  elseif i == 4
    full_areas(i) = trapz(d_freq(peaks(i,1):peaks(i+1,2)),
        peak_intens(peaks(i,1):peaks(i+1,2)));
  end
end

% Areas found by using trapezoidal method for the whole peak/dip
    interval.
areas = abs(full_areas);
areas_str = sprintf('\t%E\n', areas);
fprintf(1, 'Areas found using thw whole peak/dip interval is: \n[\n
    %s]\n', areas_str);

% Find the largest difference in area between full and half.
areas_symm_parts = areas_symm([1:3 6]); % Skip double dip.
areas_parts = areas([1:3 6]); % Skip double dip.
area_diff = areas_symm_parts - areas_parts;
%area_norm = norm(area_diff , Inf);
[area_norm norm_pos] = max(abs(area_diff));
norm_quota = (area_norm / areas_parts(norm_pos)) * 100;
fprintf(1, 'The infinity norm of the differnce in the two set of
    areas (double dip excluded) is %E. That is %.3f%% of the full
    area.\n', area_norm, norm_quota);

dip_diff = abs(areas(4) - (areas_symm(4) + areas_symm(5)));
dip_diff_quota = (dip_diff / areas(4)) * 100;
fprintf(1, 'The difference between the full double dip area and sum
     of the symmetrically found individual areas are %E. The sum is
     %.3f%% larger than the full double dip area.\n', dip_diff,
    dip_diff_quota);
```

B.2 Functions

The following functions implements the algorithms and the rest serves as helper functions to these algorithms and the scripts. To distinguish these from other *MATLAB*®-functions in the global namespace these reside in a own package called *pr2*.

../src/+pr2/import_data.m

```
function [ d_freq,  d_intens ] = import_data( filename )
% Reads the data from the excel file filename and returns the
    frequencies
% d_freq and the intensities d_intens.
    warning off MATLAB:xlsread:ActiveX % I don't have Excel or
        Windows.
    num = xlsread(filename);
    d_freq = num(:,1);
    d_intens = num(:,2);
end
```