Table of Contents

[Red Tide Spectrum Meter Control via Matlab 2](#_Toc346537934)

[Introduction 2](#_Toc346537935)

[Ocean Optics Spectrometer Support from MATLAB 2](#_Toc346537936)

[Installation Instructions 3](#_Toc346537937)

[Step 1: Omni Driver 3](#_Toc346537938)

[Step 2: Perforce 3](#_Toc346537939)

[Step 3: Matlab R2012b 3](#_Toc346537940)

[Step 4: Configure MATLAB to use the Ocean Optics driver in R2012b: 3](#_Toc346537941)

[Using Matlab Driver 4](#_Toc346537942)

[Introduction 4](#_Toc346537943)

[Control Property Descriptions (Read/Write) 5](#_Toc346537944)

[Status Property Descriptions (Read Only) 5](#_Toc346537945)

[Control Property Descriptions (Read/Write) 9](#_Toc346537946)

[Control Property Descriptions (Read/Write) 14](#_Toc346537947)

# Red Tide Spectrum Meter Control via Matlab

## Introduction

## Ocean Optics Spectrometer Support from MATLAB



Matlab supports Ocean Optics USB spectrometers. Ocean Optics manufactures a broad line of USB powered spectrometers covering the visible, near IR and UV portions of the spectrum.  These spectrometers can be used from MATLAB.

## Installation Instructions

### Step 1: Omni Driver

Please install OmniDriver from Ocean Optics, it is required to interface with matlab. You can find the latest driver from here:

<http://www.oceanoptics.com/technical/softwaredownloads.asp>

### Step 2: Perforce

Wrapper code can be found in perforce, please check out to a local directory of your choice.

//software/projects/isp\_tools/RedTide/

### Step 3: Matlab R2012b

Ensure you have at least matlab 2012b installed on your machine.

### Step 4: Configure MATLAB to use the Ocean Optics driver in R2012b:

The OmniDriver provided by Ocean Optics has a java API. To use this API in a MATLAB session you must add two files to the preference directory. These file are found in in perforce.

javaclasspath.txt

javalibrarypath.txt

The content of these files are the perforce as described in step 2.

Open MATLAB and execute the following command.

*>> prefdir*

The prefdir command will print out the preference directory used by MATLAB. Please copy javaclasspath.txt and javalibrarypath.txt from the perforce directory to the preference directory.

**Note:** The contents of the javalibrarypath.txt and javaclasspath.txt are basically the OmniDriver installation folder location. If you have changed the location of the driver installation folder please edit the content of these files appropriately.

**Step 5: Restart.**

Restart MATLAB.

## Using Matlab Driver

### Introduction

There are three main objects in this program set. The RedTide driver, the plotSpectrum and Spectrum\_RT. A description of each block is below. The objects have a simplified interface which allows you to merely set/get the properties of the object to control behaviour.

#### RedTide (Object Description)

This is the wrapper for the red tide driver. It allows you to change the driver settings such as the integration time. It will report back the wavelengths and spectrum data. It has a timer integrated into the object which allows you to use it in single shot mode or continuous mode to allow a real time display. The RedTide does not have a GUI. It merely controls and buffers the wavelength and spectrum data.

### Control Property Descriptions (Read/Write)

This section provides a description of properties. Curly braces { } enclose default values.

##### realtime\_Enable

##### on | {off}

The mechanism is to control the internal timer object which will update the wavelengths and spectralData properties. If this is ‘on’ the properties will be update at a frequency set by the UpdateRate property. If this is set to *‘off’* the wavelength and spectralData will only be read which the properties are interrogated. This allows support for single and continuous modes.

##### integrationTime

##### double

This is the integration time of the sensor. The default value is 50.

##### UpdateRate

##### double

This property is used when the realTime\_Enable is set to ‘on’. This parameter controls the frequency of the update to wavelength and spectralData. The default value is 1.

### Status Property Descriptions (Read Only)

This section provides a description of properties. Curly braces { } enclose default values.

##### Wavelengths

The property reports the wavelength of the light bins. Typical this will report values between 300 to 1000 nm. If realTime\_Enable is set to ‘on’ this property will update at the frequency set by UpdateRate.

##### spectralData

The property reports the energy of the light found at different wavelengths. Typical this will report the energies between 300 to 1000 nm. If realTime\_Enable is set to ‘on’ this property will update at the frequency set by UpdateRate.

##### instrumentType

In this case the instrument type is a spectrometer.

##### driverType

In this case the instrument type is a MATLAB generic.

##### driverName

This is the name of the driver used. In this case the driver name is “OceanOptics\_OmniDriver.mdd”. This file can be found in your perforce directory.

##### deviceName

In this case the instrument type is a USB650.

##### deviceSerialNumber

The device serial Number is different for every piece of hardware. This property can be used for tracking purpose for calibration or other. The unit held in Cambridge has a serial number 'USB2G48769’.

##### numOfSpectrometers

This property report the number of device attached. As we only have one device in Cambridge I can only guarantee that this supports a single device. If multiple devices are required further work would be required to develop the driver further.

##### hardwareVersion

In this case the instrument type is a 'Version1'.

##### communicationState

This describes the communication state with the driver. The classdef open the device when it loads and close the communication when the classdef is cleared. So this value should always read ‘open’. If this value is every read as ‘closed’. Please report [Bryant@broadcom.com](mailto:Bryant@broadcom.com).

##### Example 1: Single Shot Plot of Spectrum

**M Code:**

% Ensure workspace is clear

close all

clear classes

% Load RedTide object

obj = RedTide( 'integrationTime', 50);

% Read wavelength and spectralData from device

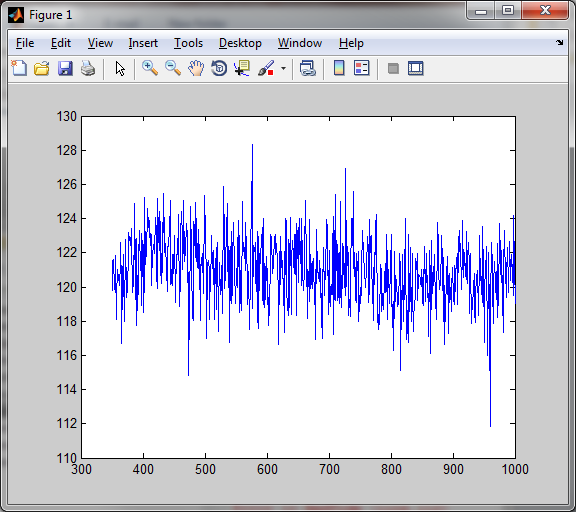
wavelengths = obj.wavelengths;

spectralData = obj.spectralData;

% plot data

plot(wavelengths,spectralData);

**Output:**



##### Example 2: Inspecting the available properties object

Like any classdef we can inspect the properties and methods to see get an idea of how to configure the object. In this case all the behaviour of the object is controlled using the properties.

**M Code:**

% Inspect the properties of a classdef

properties(obj)

methods(obj)

disp(obj)

**Output:**

#### plotSpectrum (Object Description)

This object allows plotting of spectrum data. This has been packaged into a classdef which allow this to be controlled in a similar way to matlab’s inbuilt plots meaning that you can set/get the properties to control the way the plot appears. Another key consideration on this code was to allow the plot to be updated with new wavelength and spectrumData only to allow for fast real-time display.

By avoiding the “plot” commands which re-generates the figure and updating the XDATA and YDATA properties of the figure and better real-time performance was experience.

### Control Property Descriptions (Read/Write)

This section provides a description of properties. Curly braces { } enclose default values.

##### Title

The default title is ‘Transmission spectrum’

##### illuminant

Some inbuilt spectrums. The default value is D65.

A The standard tungsten filament lamp (2856K)

D65 Medium daylight with UV component (6500K)

EE Theoretical equal-energy illuminant

##### spectrumSTD

Some inbuilt spectrum colour map standards, see below. The default value is ‘1964\_full’. This will change the way the Colour map is generate on the X axis of the figure.

CIE\_1931 CIE 1931 2-degree, XYZ

1931\_FULL CIE 1931 2-degree, XYZ (at 1nm resolution)

CIE\_1964 CIE 1964 10-degree, XYZ

1964\_FULL CIE 1964 10-degree, XYZ (at 1nm resolution)

Judd CIE 1931 2-degree, XYZ modified by Judd (1951)

Judd\_Vos CIE 1931 2-degree, XYZ modified by Judd (1951) and Vos (1978)

Stiles\_2 Stiles and Burch 2-degree, RGB (1955)

Stiles\_10 Stiles and Burch 10-degree, RGB (1959)

##### Wavelengths

The property reports the wavelength of the light bins. Typical this will report values between 300 to 1000 nm. Internally to this object there is a listener on this property. If the properties is set then the plot will update automatically.

##### spectralData

The property reports the energy of the light found at different wavelengths. Typical this will report the energies between 300 to 1000 nm. Internally to this object there is a listener on this property. If the properties is set then the plot will update automatically.

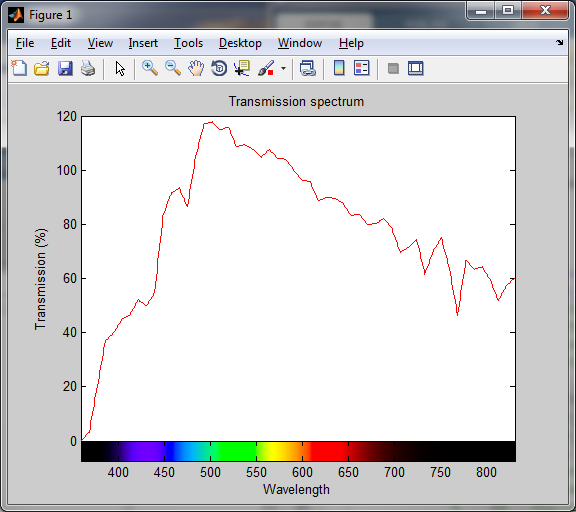
##### Example 1: Plot the Spectrum with default values:

**M Code:**

% plot D65 from defaults

obj = plotSpectrum();

**Output:**

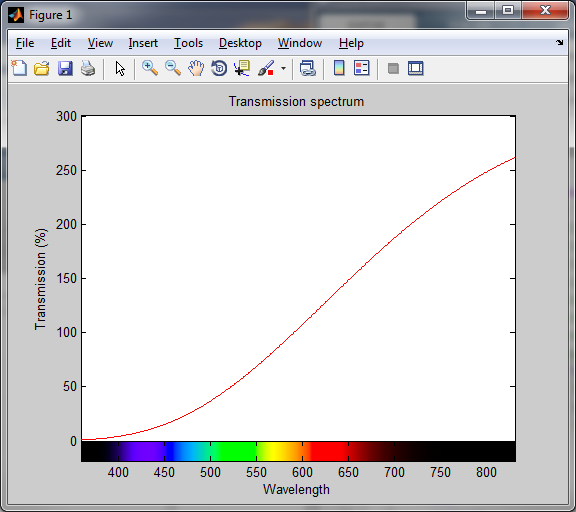


**M Code:**

% plot A

obj = plotSpectrum('illuminant','A');

**Output:**



##### Example 2: Plot single capture from red tide

**M Code:**

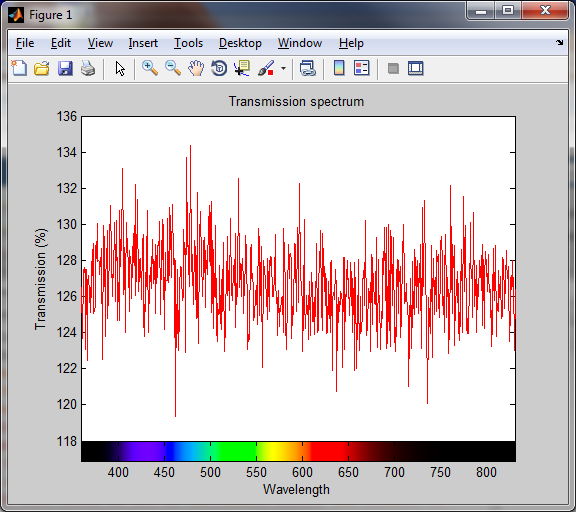
% plot from red tide

objRT = RedTide( 'integrationTime', 50);

obj = plotSpectrum( 'wavelengths', objRT.wavelengths, ...

'spectralData', objRT.spectralData);

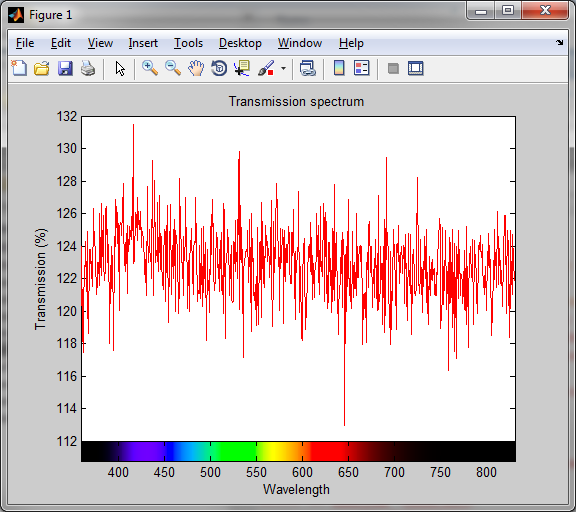
**Output:**



**M Code:**

% Since the object are already loaded you can update the spectrum plot from red tide like so:

obj.spectralData = objRT.spectralData;

**Output:** 

#### ***Spectrum***\_RT

This object is the glue that links the plotSpectrum and RedTide buffer together. It allow the Spectrum data from the RedTide to be displayed real-time. The properties can control the timer which is located in the RedTide driver object. It’s this timer that forces the buffers in the red-Tide module to update which then force the plot to update.

### Control Property Descriptions (Read/Write)

This section provides a description of properties. Curly braces { } enclose default values.

##### integrationTime

##### double

This is the integration time of the sensor. The default value is 50.

##### UpdateRate

##### double

This property is used when the realTime\_Enable is set to ‘on’. This parameter controls the frequency of the update to wavelength and spectralData. The default value is 1.

##### Example 1: Run the spectrum plot real-time.

**M Code:**

%% Real Time Example Link to Spectrum plot

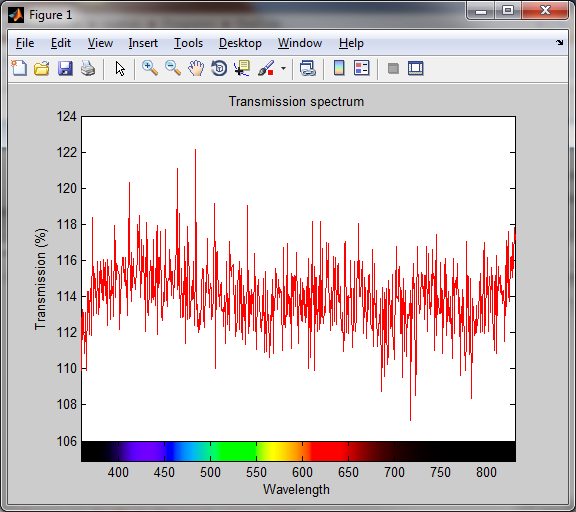
% This will open object. By default the timer is not on.

close all

clear classes

obj = Spectrum\_RT( 'integrationTime', 50, ...

'updateRate', 0.1);

**Output:** 

**M Code:**

To turn on and off the real time plot (Enables and disables the timer)

%% start

obj.handles.objRT.realtime\_Enable = true

%% stop

obj.handles.objRT.realtime\_Enable = false