Design Proposal – Bragg Grating Resonators

Patrick Wilson - 66081258 – ELEC 413 2022W

***Abstract*—This document describes the design process of a waveguide Bragg grating resonator with a Fabry Perot cavity.**

I. INTRODUCTION

THIS document describes the design process for a waveguide Bragg grating resonator with a Fabry Perot cavity. Described below, are the numerous models used to parametrize the device for use at 1310nm. The goal is to design a resonator with a large quality factor (QF), and a low free spectral range (FSR). We divide the model into several sections to describe each attribute of the device more accurately. The sections are as follows:

- Waveguide properties (dimensions, polarization, materials)
- Effective and group indices (mode profile, compact model)
- Transfer function
- Parameter variations
- Transmission spectrum

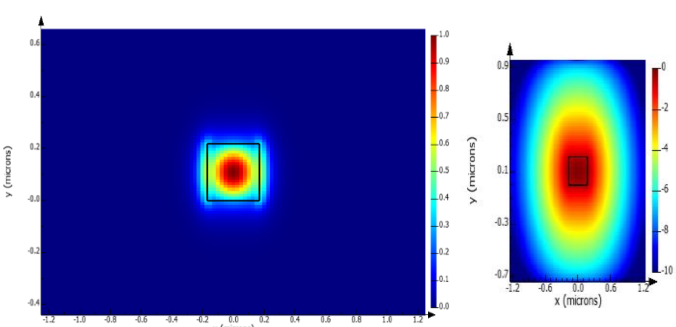
As stated above, the large QF, low FSR requirement will make this resonator suitable for laser-on-chip applications.

II. WAVEGUIDE

We begin by defining the waveguide geometry. This is a 350x220nm silicon waveguide with silicon dioxide cladding. It is TE polarized at a central wavelength of 1310nm.

To parametrize the waveguide, it is first assumed that shrinkage will occur during the manufacturing process. All waveguide simulations will be performed with dimensions of 335x220nm to account for this. Using Ansys Lumerical MODE, a rectangular waveguide is created with regions of Si, SiO2 in appropriate dimensions. *Material explorer* is then applied to the silicon region to fit a multi coefficient model representing index vs wavelength. Next, the simulation is run. The mode profile of the waveguide is seen in *figure 1*.

Fig 1. *Waveguide Mode Profile*



Once the modes have been calculated, a frequency sweep is performed on the fundamental mode (mode 1). Analyzing the plots of effective index and group index vs wavelength (λ) reveals the operating point of the waveguide at 1310nm.

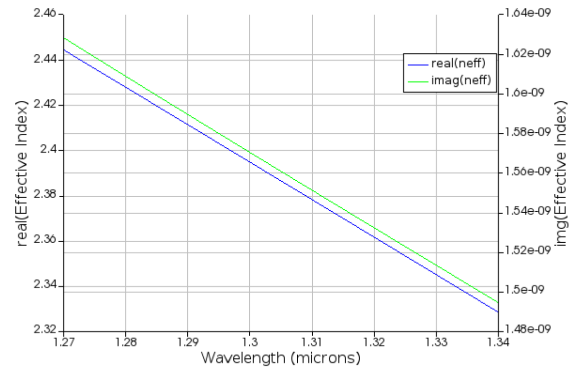


Fig.2. *Effective Index vs Wavelength*

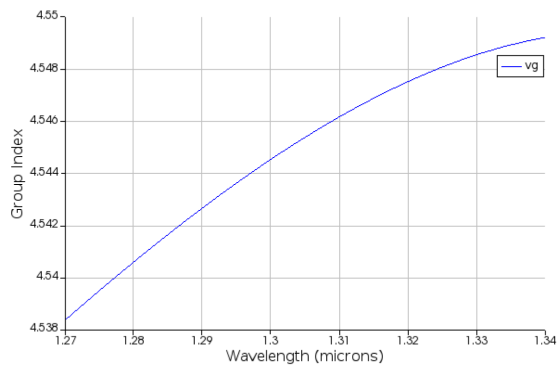


Fig.3. *Group Index vs Wavelength*

At 1310nm, the group index of a 335x220nm rectangular waveguide is found to be **4.5462** with an effective index of **2.3782**. The effective index can be used to find the Bragg period of a unit cell Bragg structure via:

$$\Lambda = \frac{\lambda_B}{2 * n_{eff}} \qquad (1)$$

The Bragg period is **275.4**nm.

To find the compact model of the wavelength dependent effective index, the frequency sweep data is exported to MATLAB and the provided *phot1x_fit_wg_compactmodel.m* script fits the data. The resulting model is as follows:

$$n_{eff}(\lambda) = 2.3782 - 1.6552(\lambda - 1.31)^2 - 0.0386(\lambda - 1.31)$$

III. Bragg Grating Unit Cell

Using Lumerical FDTD, the waveguide model can be expanded to represent a unit cell (one period long) Bragg grating. The width is kept constant at 335nm and the period remains 275nm as found in section II. The grating width, expressed as a difference in overall width dW, is swept from 5nm to 120nm. The central wavelength (λ_B), bandwidth ($\Delta\lambda$), and the coupling coefficient Kappa (κ) can be extracted from the simulation. The relationship between these three parameters is as follows:

$$\kappa = \frac{\pi \Delta\lambda}{\lambda_B^2} \qquad (2)$$

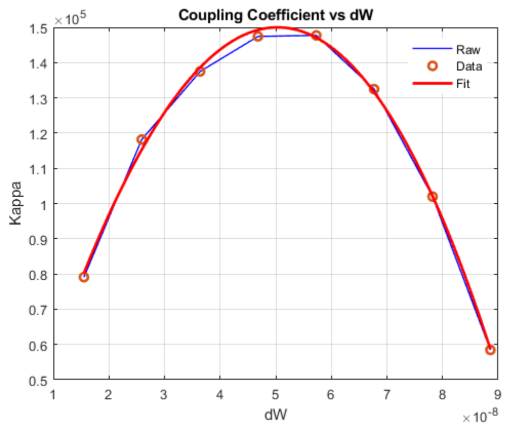


Fig.4. *Coupling Coefficient* (κ) *vs dW*

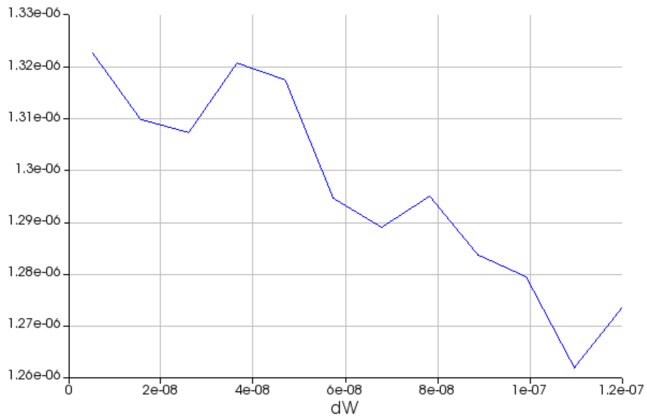


Fig.5. *Central Wavelength vs dW*

Seen in (4) is the relationship between κ and dW as plotted via MATLAB polyfit. Figure (5) shows the relationship between central wavelength and dW found in FDTD. Note that (5) was taken with only twelve data points. Increasing the number of points may have increased the quality of the dataset. We observe however, that the target wavelength, 1310nm is achievable with dW=[5e-06,6e-06]. There is also a region at dW < 2e-06 with $\lambda_B \approx 1310nm$ however given the curve in (4), bandwidth suffers at low values of dW and thus the solution is rejected. A dW value of **55nm** is selected for the design.

IV. Transfer Matrix Method

To combine Bragg waveguide models into a cavity resonator, the transfer matrix method can be utilized. The transfer matrix method allows for a model of the resonator as the product of three matrices:

$$T_{left} * T_{Cavity} * T_{Right} \qquad (3)$$

Where

$$T_{left} = (T_{hw2} * T_{is21} * T_{hw1} * T_{is12})^{NG-1} \qquad (4)$$

$$T_{Right} = (T_{is21} * T_{hw1} * T_{is2} * T_{hw2})^{NG-1} \qquad (5)$$

$$T_{Cavity} = \qquad (6)$$
$$T_{hw2} * T_{is21} * T_{hw1} * T_{is1c} * T_{hwC} * T_{isc1} * T_{hw1} * T_{is12} * T_{hw2}$$

Note that T_{hw*} and T_{is*} represent matrices for homogenous waveguides, and step changes in index between regions. NG-1 gratings are used, in addition to two unit cells on either side of the cavity. The total number of gratings is NG. The homogenous and unit step matrices are as follows:

$$T_{hw*} = \begin{bmatrix} e^{j\beta L} & 0 \\ 0 & e^{-j\beta L} \end{bmatrix} \qquad (7)$$

$$T_{is*} = \begin{bmatrix} \frac{n_1+n_2}{2\sqrt{n_1 n_2}} & \frac{n_1-n_2}{2\sqrt{n_1 n_2}} \\ \frac{n_1-n_2}{2\sqrt{n_1 n_2}} & \frac{n_1+n_2}{2\sqrt{n_1 n_2}} \end{bmatrix} \qquad (8)$$

$$\beta = \frac{2\pi n_{eff}}{\lambda} \qquad (9)$$

In the above, L is the length of the waveguide and β is the propagation constant. We find n_1 and n_2 using the change in effective index:

$$\Delta n = \frac{\kappa \lambda_B}{2} \qquad (10)$$

$$n_1 = n_{eff} - \frac{\Delta n}{2} \qquad (11)$$

$$n_1 = n_{eff} - \frac{\Delta n}{2} \qquad (12)$$

The TMM calculations are performed via python script. The polyfit results shown in (4) are used to get a compact model for kappa:

$$\kappa(dW) = -6.0156E19 * (dW)^2 + 5.9885E12(dW) + 1092.5 \qquad (13)$$

And (10) is used to obtain Δn, which can be used to find n_1 and n_2 in T_{is} . Computing both the transmission and reflection for parameters: *NG=75, dW=55e-9*, $\Lambda = 275e\text{-}9$, λ_B*=1310e-9* yields:

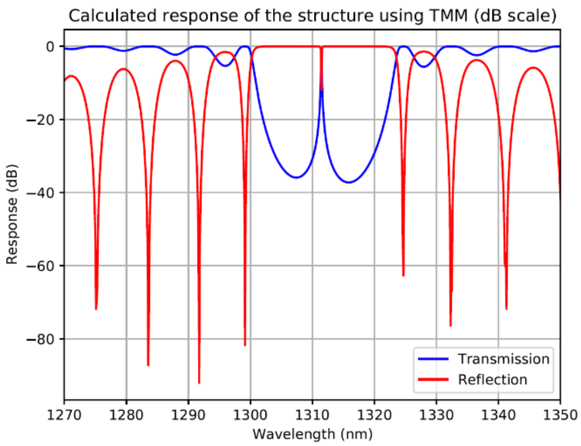


Fig.6. Transfer Method Spectrum (dB)

Sweeping *NG* from 10-200 reveals that transmission decays with increasing *NG*.

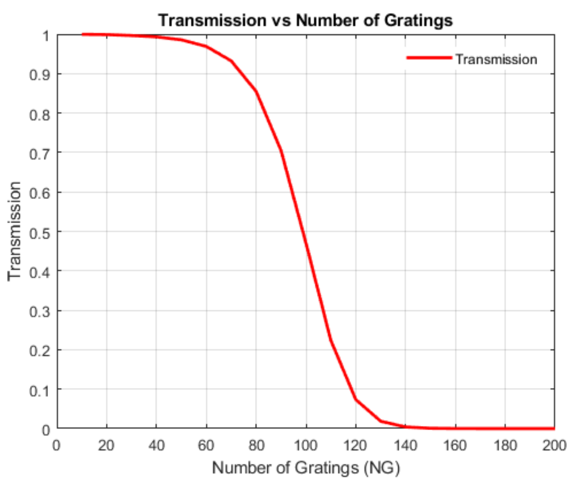


Fig.7. Transmission vs NG

For the design, NG will vary between **30-80** to preserve the transmission peak.

V. Q, FSR and Expected performance

Quality factor and Free Spectral Range are defined as follows:

$$Q = \frac{w}{\Delta w_{\frac{1}{2}}} = \frac{2\pi n_g}{\lambda * \alpha} \qquad (14)$$

$$FSR = \Delta\lambda_{\frac{1}{2}} * F \qquad (15)$$

Where

$$F = \frac{\pi\sqrt{AR}}{1 - AR} \qquad (16)$$

The expected performance of these gratings will be experimentally confirmed, however the design will focus on maintaining a high Quality factor, between 5000 and 50 000. Low to medium risk devices will likely have Q= [5000,14 000] whereas medium-high risk devices will have Q= [14 000, 50 000]. In total there will be 14 designs manufactured, with each varying a different set of parameters. Set A will vary *NG* from 40-70 microns in 5-micron increments, with set B varying resonator cavity length from 30 microns to a maximum of 137.5 microns (*period/2).* For set B, *NG* will remain fixed. the proposed designs are below:

CL *[Microns]*	NG	dW *[nm]*	λ_B *[nm]*	T_{peak} [dB]
75	40	55	1310	-0.0410
75	45	55	1310	-0.0871
75	50	55	1310	-0.1711
75	55	55	1310	-0.3099
75	60	55	1310	-0.5775
75	65	55	1310	-1.0863
75	70	55	1310	-2.0047

Table.1. Design Set A: *Number of Gratings [40,70]*

CL *[Microns]*	NG	dW *[nm]*	λ_B *[nm]*	T_{peak} [dB]
30	70	55	1310	-2.0047
50	70	55	1310	-2.0047
70	70	55	1310	-2.0047
90	70	55	1310	-2.0047
110	70	55	1310	-2.0047
130	70	55	1310	-2.0047
137.5	70	55	1310	-2.0047

Table.2. Design Set B: *Cavity Length [30, 137.5]*

VII. Layout

Each design set will exist within a footprint of 605x410nm. The designs include three 350nm grating couplers that act as injection, transmission, and reflection ports respectively. For project 2, one design from the above table will be selected for connection to the tunable laser.

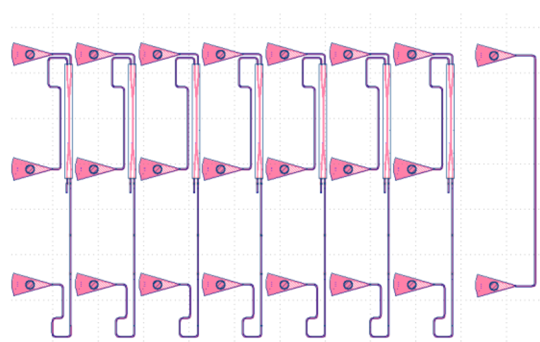


Fig.8. Design structure layout (Approximate)

The tunable laser operates in the O range (1270-1330nm) and is tunable within 0.2nm. This means the selected design should have an FSR of at most 0.2nm. Additional requirements are as follows:

- One resonator
- One Y-branch splitter

- One output waveguide
- FSR smaller than current tuning range
- Large mirror bandwidth (Δw)

It is estimated that each resonator will see -21dBm. The selected design will be from the low-medium risk group such that it will likely work with the laser setup. Estimated Q factor in the 5000-15000 range with FSR $\approx 0.2nm$.

Layout is done using *Klayout* and verification can be performed in *Lumerical INTERCONNECT.*

VI. CONCLUSION

This document describes the design and layout of various Bragg grating resonators and describes the integration to a tunable laser experimental setup. All calculations and figures were produced using a variety of tools including MATLAB, Lumerical MODE, FDTD, Python, and Klayout. The final design document will follow.

APPENDIX

Extraction of compact model coefficients.

Provided by Dr .L. Chrostowski

```
% example 2
% TM polarization for a 350 x 220 nm waveguide
% data was saved as a MATLAB file, and is loaded
% this requires that you either have the file on the
internet and use the "websave" command to access it
% (this is necessary if you are using the in-browser edX
MATLAB)
% or run this script on your own computer and load the
file from your local disk.

load('Neff.mat');


neff = real(neff);  % take the real part of the effective
index.

c=299792458;  % speed of light, m/s
lambdas = c ./ f;  % f is the matrix of frequency points,
                   % where the effective index is
recorded.
lambdas = lambdas * 1e6;  % convert to microns.
lambda0 = 1.310;   % replace with desired centre
wavelength

figure; plot (lambdas, neff,'o','MarkerSize',10); hold
on;

% use Matlab anonymous function for the effective index
expression:
neff_eq = @(nx, lambda) ...
                (nx(1) + nx(2).*(lambda-lambda0) +
nx(3).*(lambda-lambda0).^2);

% initial guess.
% The X matrix is defined as follows:  n1 = X(1), n2 =
X(2), n3 = X(3)
X=[2.384 0 0];

plot ( lambdas, neff_eq(X, lambdas), 'r')

% curve fit to find expression for neff.
format long
X = lsqcurvefit (neff_eq, X, lambdas, neff);

disp (['n1 = ' num2str(X(1)) ', n2 = ' num2str(X(2)) ',
n3 = ' num2str(X(3))]);

r=corrcoef(neff,neff_eq(X, lambdas));
r2=r(1,2).^2;
disp (['Goodness of fit, r^2 value: ' num2str(r2) ])

lambdas2=linspace(min(lambdas), max(lambdas), 100);

plot ( lambdas2, neff_eq(X, lambdas2), 'k')
xlabel ('Wavelength [nm]');
ylabel ('Effective Index');

legend ('Data','Initial Guess','Curve Fit')


period=1.31/(2*X(1));
```

Bragg Bandstructure MAIN: FDTD Script

Provided by Dr. L. Chrostowski

Written by M. Hamood

```
#
# Bragg grating Lumerical simulation flow
# see https://github.com/mustafacc/SiEPIC_Photonics_Package/ for
documentation
#
# Author: Mustafa Hammood ; mustafa@siepic.com ; mustafa@ece.ubc.ca
# SiEPIC Kits Ltd. 2020 ; University of British Columbia
#
# (c)2020

newproject;
save("Bragg_Bandstructure.fsp");
clear;

##############################
# Simulation parameters #
##############################

wl_min = 1.27e-6; # simulation wavelength start
wl_max = 1.6e-6; # simulation wavelength stop

pol = 'TE'; # simulation polarization

mesh_y = 5e-9;
mesh_x = 5e-9;
mesh_z = 20e-9;

sim_time = 1500e-15; #E-15 is femto...
mesh = 4;

##############################
# Device geometry #
##############################

ng = 4.5462; # group index of the waveguide (average width)
W = 335e-9; # uncorrugated waveguide width
dW = 50e-9; # waveguide corrugation
period = 275e-9; # corrugations period
rib = false; # enable or disable rib layered waveguide type (do not enable with
TM mode)
sidewall_angle = 90;

thickness_device = 220e-9; # waveguide full thickness
thickness_rib = 90e-9; # waveguide rib layer thickness
thickness_superstrate = 2e-6; # superstrate thikness
thickness_substrate = 2e-6; # substrate thickness
thickness_handle = 300e-6; # handle substrate thickness

mat_device = 'Si (Silicon) - Dispersive & Lossless'; # device material
mat_superstrate = 'SiO2 (Glass) - Palik'; # superstrate material
mat_substrate = 'SiO2 (Glass) - Palik'; # substrate material
mat_handle = 'Si (Silicon) - Dispersive & Lossless'; # handle substrate
material

Bragg_draw;
Bragg_simulate;
#Bragg_analysis;
```

Bragg_TMM Python script:

Provided by Dr. L. Chrostowski
Written by M. Hamood
Modified by P. Wilson

```
"""
SiEPIC Photonics Package

Author:     Mustafa Hammood
            Mustafa@siepic.com

            https://github.com/SiEPIC-Kits/SiEPIC_Photonics_Package

fixed by Lukas Chrostowski, 2020/02

Solvers and simulators: Bragg simulator using transfer matrix method (TMM) approach

"""

#%% dependent packages
import numpy as np
import math, cmath, matplotlib
import matplotlib.pyplot as plt
from numpy.lib.scimath import sqrt as csqrt
import scipy.io as sio

#%% user input

# set the wavelength span for the simultion
wavelength_start = 1309e-9
wavelength_stop = 1312e-9
resolution = 0.1

# Grating waveguide compact model (cavity)
# these are polynomial fit constants from a waveguide width of 500 nm
n1_wg = 2.38782
n2_wg = -1.6552
n3_wg = -0.038638

# Cavity waveguide compact model (cavity)
# these are polynomial fit constants from a waveguide width of 500 nm
n1_c = 2.38782
n2_c = -1.6552
n3_c = -0.038638

# grating parameters
period = 275e-9     # period of pertrubation
n_delta = .0      # effective index pertrubation
lambda_Bragg = 1310e-9
dw = 55e-9
kappa = -6.0156e19 * (dw)**2+ 5.9885e12 * (dw) +1.0952e03
n_delta = kappa * lambda_Bragg / 2
print(n_delta)
NG=70
N_left = NG          # number of periods (left of cavity)
N_right = NG          # number of periods (right of cavity)

# Cavity Parameters
alpha_dBcm = 3     # dB per cm
alpha = np.log(10)*alpha_dBcm/10*100. # per meter

L = 70e-6    # length of cavity

#%% Analysis


def HomoWG_Matrix( wavelength, neff, l):
    beta = 2*math.pi*neff/wavelength-j*alpha/2
    v = [np.exp(j*beta*l), np.exp(-j*beta*l)]
    T_hw = np.diag(v)

    return T_hw

def IndexStep_Matrix(neff1, neff2):
    a=(neff1+neff2)/(2*np.sqrt(neff1*neff2))
    b=(neff1-neff2)/(2*np.sqrt(neff1*neff2))

    T_is=[[a, b],[b, a]]
    return T_is

def Grating_Matrix( wavelength, n1, n2, l ):
    T_hw1=HomoWG_Matrix(wavelength, n1, l)
    T_is12=IndexStep_Matrix(n1,n2)
    T_hw2=HomoWG_Matrix(wavelength, n2, l)
    T_is21=IndexStep_Matrix(n2,n1)


    type = 'Cavity'


    if type == 'Waveguide':
        # 1 cm
        T = HomoWG_Matrix(wavelength, n1, 0.01)

    if type == 'Bragg_left':
        Tp1 = np.matmul(T_hw1, T_is12)
        Tp2 = np.matmul(T_hw2, T_is21)
        Tp_Left = np.matmul(Tp1, Tp2)
        T = np.linalg.matrix_power(Tp_Left,N_left)

    if type == 'Bragg_right':
        Tp1 = np.matmul(T_hw1, T_is12)
        Tp2 = np.matmul(T_hw2, T_is21)
        Tp_Right = np.matmul(Tp1, Tp2)

        T = np.linalg.matrix_power(Tp_Right,N_right)

    if type == 'Cavity':
        Tp1 = np.matmul(T_hw1, T_is12)
        Tp2 = np.matmul(T_hw2, T_is21)
        Tp_Left = np.matmul(Tp1, Tp2)

        T_cavity = HomoWG_Matrix(wavelength, n1, l)

        Tp1 = np.matmul(T_hw1, T_is12)
        Tp2 = np.matmul(T_hw2, T_is21)
        Tp_Right = np.matmul(Tp1, Tp2)

        T = np.matmul(np.matmul( np.linalg.matrix_power(Tp_Left,N_left), T_cavity), np.linalg.matrix_power(Tp_Right,N_right))

    return T

def Grating_RT( wavelength, n1, n2, l ):
    M = Grating_Matrix( wavelength, n1, n2, l )
    T = np.absolute(1 / M[0][0])**2
    R = np.absolute(M[1][0]/M[0][0])**2.  # or M[0][1]?
    return [T,R]

j = cmath.sqrt(-1)

l = period/2
lambda_0 = np.linspace(wavelength_start, wavelength_stop, round((wavelength_stop-wavelength_start)*1e9/resolution))
neff0 = (n1_wg + n2_wg*(lambda_0*1e6-lambda_Bragg*1e6) + n3_wg*(lambda_0*1e6-lambda_Bragg*1e6)**2)

n1 = neff0 - n_delta/2
n2 = neff0 + n_delta/2

print(n1)
```

```
# Grating length
L_left = N_left * period
L_right = N_right * period

R = []
T = []

for i in range(len(lambda_0)):

    [t,  r]  = Grating_RT(lambda_0[i],  n1[i],  n2[i],
1)

    R.append(r)
    T.append(t)


sio.savemat('bragg_tmm_1310.mat',   {'R':R,   'T':T,
'lambda_0': lambda_0})

sio.loadmat('bragg_tmm_1310.mat')
Tmax=max(T);
print(Tmax)
#%% plot spectrum
plt.figure(0)
fig1    =    plt.plot(lambda_0*1e9,10*np.log10(T),
label='Transmission', color='blue')
fig2    =    plt.plot(lambda_0*1e9,10*np.log10(R),
label='Reflection', color='red')
plt.legend(loc=0)
plt.grid(True)
plt.ylabel('Response (dB)', color = 'black')
plt.xlabel('Wavelength (nm)', color = 'black')
plt.setp(fig1, 'linewidth', 1.5)
plt.setp(fig2, 'linewidth', 1.5)
plt.xlim(round(min(lambda_0*1e9)),round(max(lambda_0
*1e9)))
plt.title("Calculated  response  of  the  structure
using TMM (dB scale)")
plt.savefig('bragg_tmm_dB_1310'+'.pdf')
```

Plot Transmission and dB spectrum/extract max T:

Written by P.Wilson

```
clear;

load('bragg_tmm_1310.mat');

fig=figure;
plot(lambda_0,R,'r','LineWidth',2);
hold on
grid on
plot(lambda_0,T,'b','Linewidth',2);
legend Transmission Reflection
title('Transmission Spectrum');
xlabel('Wavelength [microns]');
ylabel('Transmission');

hold off;

fig2=figure;
plot(lambda_0,10*log10(R),'r','LineWidth',2);
hold on
grid on
plot(lambda_0,10*log10(T),'b','Linewidth',2);
legend Transmission Reflection
title('Transmission Spectrum [dB]');
xlabel('Wavelength [microns]');
ylabel('Transmission');

lambda_0=transpose(lambda_0);
T=transpose(T);
L=table(lambda_0,T,'VariableNames',{'lambda',
'Transmission'});

Tmax=max(T);
Maxdb=10*log10(Tmax);
idx=L.Transmission==Tmax;
Lam=L.lambda(idx);

% Find the half-maximum value
half_max = Tmax/ 2;

disp(Tmax);
disp(Maxdb);
```

Plot T vs Grating Number

Written by P.Wilson

```
clear;

t=readtable('TvNG.csv');
NG=t.NG;
T=t.T;

fig=figure;
plot(NG,T,'r','LineWidth',2);
grid on;
title('Transmission vs Number of Gratings');
xlabel ('Number of Gratings (NG)');
ylabel ('Transmission');
legend Transmission;
hold off;
```

Find Polynomial Fit of Kappa

Written by P.Wilson

```
clear;

t= readtable('kappdw.csv');
x=t.dW;
y=t.Y;
b=t.B;

fig1=figure;
plot(x,y,'b','LineWidth',1);
xlabel('dW');
ylabel('Kappa');
grid on;
hold on;

% Fit a 3rd degree polynomial to the data
p=polyfit(x,y,2);
pn=(p.*b)./2;
p1=polyfit(x,b,3);

%s = sprintf('%g+%g*x^3 + %g*x^2 + %g*x', p(1), p(2),
p(3), p(4));

%s1 = sprintf('%g+%g*x^3 + %g*x^2 + %g*x', p1(1), p1(2),
p1(3), p1(4));

% Evaluate the polynomial fit at x values
x_fit = linspace(min(x), max(x));
y_fit = polyval(p, x_fit);

dn_fit=(y_fit.*b)./2;

x_fit = linspace(min(x), max(x));
b_fit = polyval(p1, x_fit);

% Plot the data and the fit
plot(x, y, 'o', x_fit, y_fit,'r','LineWidth',2);
title('Coupling Coefficient vs dW')
legend Raw Data Fit;
hold off;

%disp(s);
```