Contents


```
% main function which excludes negative coefficients
% This version of simulation has the following features:
% delta_L is rounded off to 0.1e-9 (0.1 nm). The resulting effect is
% that spectrum recovery is stable up to **Order = 65 points**. This is
% improved from **Order = 20** of last version, which is a huge
% improvement.

% New feature updated from last version, which is **Verion 00**: The
% parameter **omega_d**, which is for the candidate frequencies is
% preselected first. Preselection kind of sets the resolution of FOS. FOS
% searches through the candidate frequencies and the number of searched
% points is **Order**, which can be changed from run to run.

% The process of using this code is:
% Step 1: Choose the number of MZI in **theory_sim_sigma**
% Step 2: Set delta=1 in **main**
% Step 3: Run **main**
% Step 4: Choose a large integer for the variable **order**
% Step 5: From the simulation result in step 4, go back to step 2, choose a
% different value for **delta = 1401/m** (the vale at which simulation
% stopped in step 4. The **delta** value can be inferred from the final
% plot. The magnitude of the peak/peaks shoud be roughly the value
% Step 6: Re-run the simulation with the new **delta** value. In this step,
% a smooth curve should be obtained.
```

Important parameters to bear in mind

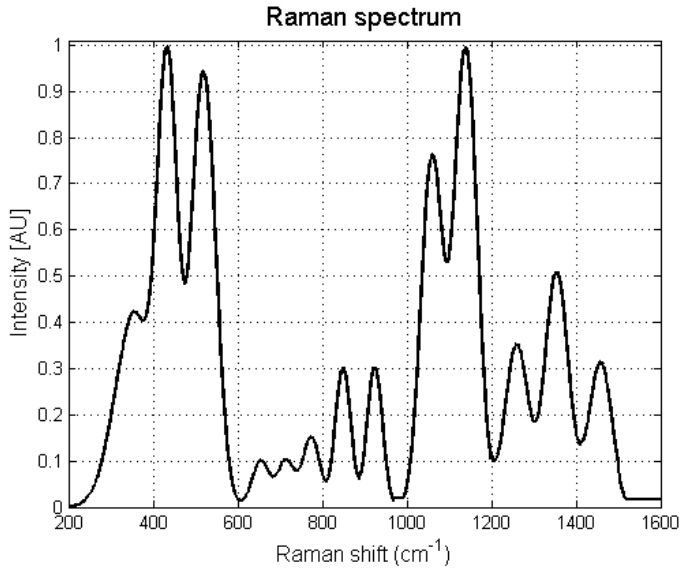
Line 121 in KOR89_exclude_negative.m determines error term. The magnitude of the peak/peaks shoud be roughly the value of ***delta***. Specifically the ***delta*** value should be just above the peak value, and as close as possible.

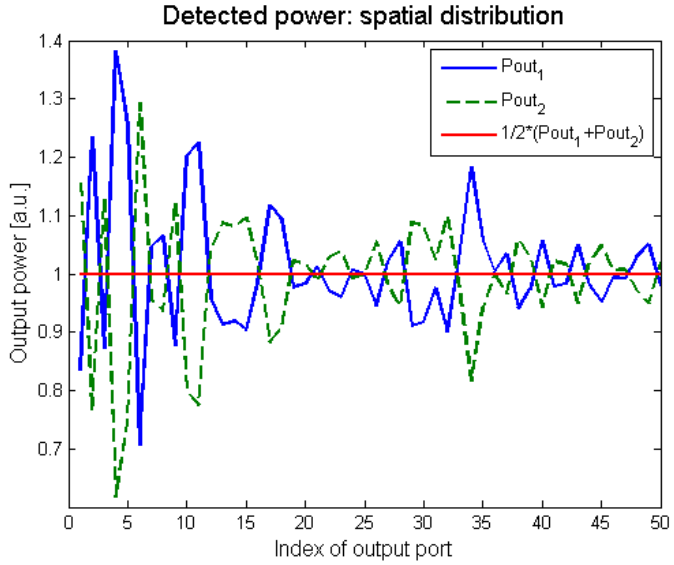
```
clc; clear all; close all;
```

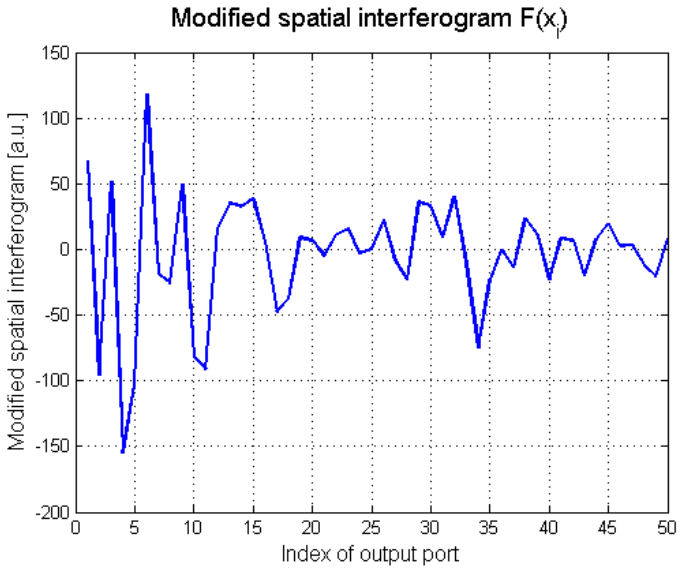
Interferogram, spectrum and MZI layout.

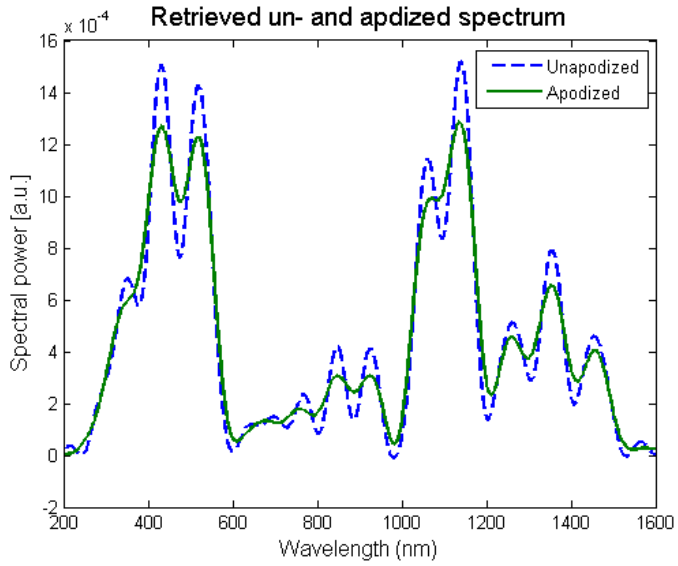
the functions theory_sim_sigma.m and KOR89.m are called prompt = 'Number of MZIs in the device?\n MZInumber = '; MZInumber = input(prompt);

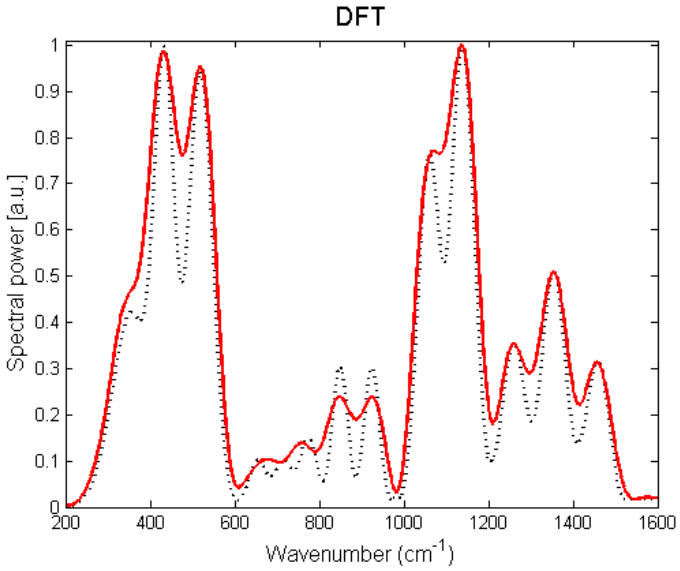
```
MZInumber = 50;

[ F, p_in, resol, sigma, delta_Li, neff, p_in_bar_apodized] = theory_sim_sigma(MZInumber);
% F: the interferogram
% p_in: the glucose Raman spectrum
% resol: the resolution of the Raman spectrum, = cm^{-1}
% sigma: wavenumber of the glucose Raman spectrum
% delta_Li: the unbalance of the MZI's two arms
% neff: modal effective index
```


Raman spectrum
Intensity [AU]
Raman shift (cm-1)


Detected power: spatial distribution
Pout1
Pout2
1/2*(Pout1 +Pout2)
Output power [a.u.]
Index of output port


Modified spatial interferogram F(xi)
Modified spatial interferogram [a.u.]
Index of output port





Random selection of interferogram data points.

M = importdata('myFile.txt'); y = F(M); n = delta_Li(M);

Full information of interferogram data points.

```
y = F;
n = delta_Li;
delta = 32;
% prompt = 'Resolution of FOS?(Depends on number MZIs. To start, try *1*.)\n delta = ';
% delta = input(prompt);
omega_d = 2*pi*sigma(1:delta:end)*neff;
% y: LHS of the Equations (1) and (22)
% n: integer quantities in Equations (20), (21), and (22) etc.
% delta: "= 1" means all data is included. None is skipped.
% omega_d: candidate freq's, i.e. in Equation (20)
```

KOR89 algorithm to get g and alpha.

```
tic;

[g, alpha,index, order,error_reduction_return] = KOR89_exclude_negative(y,n,omega_d);
% g: is the coefficient on Page 221 of the reference paper.
% alpha: is the coefficient on Page 221, Equation (5).
% index: the indices of the "omega_d" entries found through FOS.
% order: the total number of frequency components to be found.

toc;
```

```
Elapsed time is 0.547949 seconds.
```

Convert coefficient g to a.

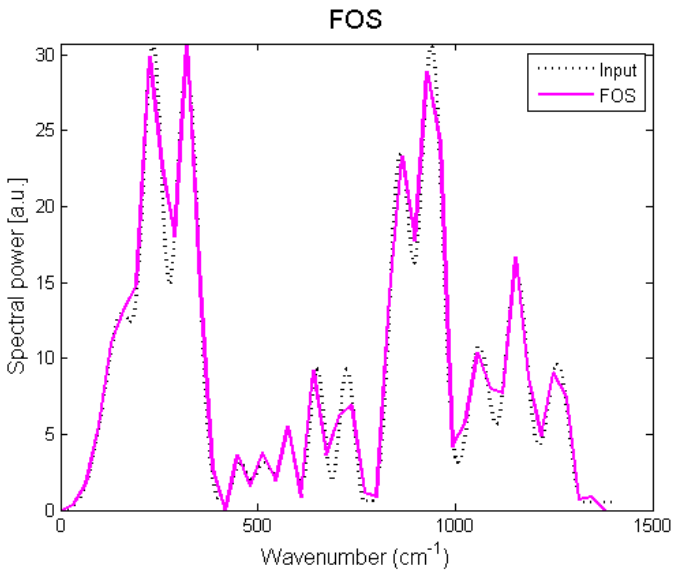
```
% a is the coefficient on Page 222, Equation (15).
a = zeros(1,length(g));
a(length(g)) = g(length(g));

for m = (length(g)-1):-1:1
    v = zeros(1,length(g));
        v(m) = 1;
        a(m) = g(m)*v(m);
    for i = (m+1):length(g)
        for r = m:(i-1);
            v(i) = v(i) - alpha(i,r)*v(r);
        end
        a(m) = a(m)+ g(i)*v(i);
    end

end
```

Plot spectrum

```
FOS_spectrum = zeros(1, length(omega_d));
% FOS_spectrum: spectrum retrieved using FOS method

for i = 1:(order);

    FOS_spectrum(index(i)) = a(i+1);
    % Because coefficient a includes a constant term (zero frequency),
    % FOS_spectrum does NOT include constant terms.

%     spectrum(index(i)) = a(2*i); % this is for KOR89_incld_sin

end
```

Before normalization

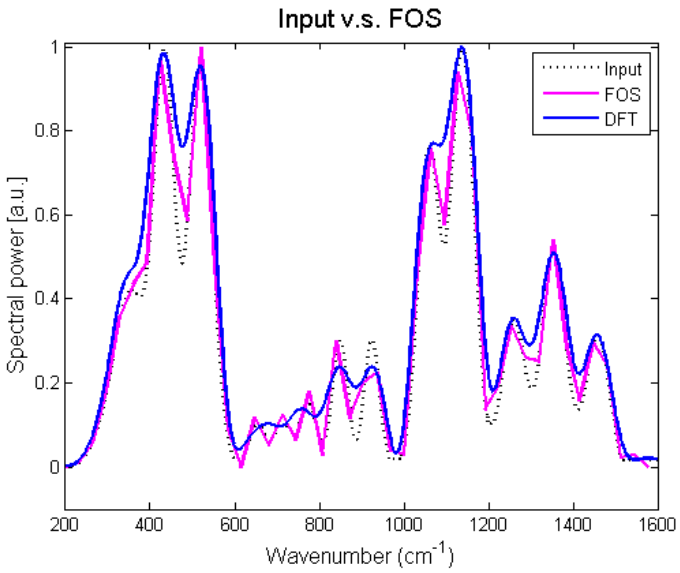
```
figure;
plot(p_in*max(FOS_spectrum),':k','linewidth',2);
hold on;

plot((1:delta:1401),FOS_spectrum,'-m','linewidth',2);

ylim([min(FOS_spectrum) max(FOS_spectrum)]);
xlabel('Wavenumber (cm^{-1})','FontSize',12);
ylabel('Spectral power [a.u.]','FontSize',12);
title('FOS','FontSize',14);
legend('Input','FOS');

hold off;
```



Comparison between FOS and DFT

```
figure;
plot(199+(1:1:1401),p_in,':k',199+(1:delta:1401),FOS_spectrum/max(FOS_spectrum),'-m',199+(1:1:1401),p_in_bar_apodized/max(p_in_bar_apodized),'b','LineWidth',2);
xlabel('Wavenumber (cm^{-1})','FontSize',12);
ylabel('Spectral power [a.u.]','FontSize',12);
xlim([200 1600]);
```

```
ylim([-0.1 1.01]);
legend('Input','FOS','DFT');
title('Input v.s. FOS','FontSize',14);
```



Published with MATLAB® R2013a