APPLICATION NOTE

Group Delay Ripple Compensation

Scope and Overview

Many useful optical components such as diffraction gratings and fiber Bragg gratings possess, by design, a frequency dependent group delay. This control over group delay enables dispersion compensation, a necessity for today's optical communication networks.

However, the actual group delay profile produced by such components typically does not perfectly match the desired response. The experimental deviation in group delay of the component from the desired profile is known as group delay ripple (GDR).

By using the phase modulating capabilities of the WaveShaper, which is capable of producing arbitrary group delay profiles of up to ±25 ps, it is possible to compensate for GDR and significantly improve the accuracy of the component. This compensation is achieved by configuring the WaveShaper to produce an equal but opposite GDR that counteracts the GDR of the component.

This application note explains how suitable group delay profiles can be calculated, and subsequently uploaded to the WaveShaper.

Applicable Documents

- WaveShaper Manual
- WaveShaper API Programming Manual
- WaveShaper API header file (ws_api.h)
-

1. Theory

Group delay (τ) is defined as

$$\tau(\omega) = -\frac{\mathrm{d}\phi}{\mathrm{d}\omega}, \qquad \ldots (1)$$

with angular frequency ω, and phase ϕ. Suppose an optical component imposes a wavelength dependent GDR of $\tau_c(\omega)$. To compensate this, the WaveShaper must introduce its own GDR $\tau_w(\omega)$ which is equal but opposite to τ_c. For convenience, the WaveShaper will be made to reproduce the GDR profile directly rather than its negative, noting that the WaveShaper's ability to match a GDR profile will also correspond with its ability to compensate this same profile.

With τ_c measured experimentally, the phase profile $\phi_w(\omega)$ required for the WaveShaper can be calculated from Eq. 1.

$$\phi_w = -\int \tau_c \,.\mathrm{d}\omega, \qquad \ldots (2)$$

2. Method

In this example, a WaveShaper Profile (.wsp) is generated which configures the WaveShaper to reproduce the GDR profile of a chirped fiber Bragg grating (CFBG). The GDR profile is shown in Figure 1 (blue). The optical resolution of the WaveShaper is around 10 GHz, and any features in the GDR profile which are smaller cannot be reproduced. Filtering these features out provides improved results. The filtered GDR is shown in Figure 1 (red).

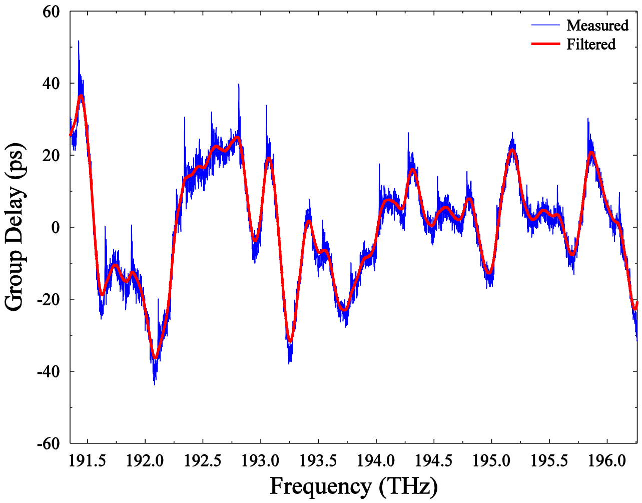


Figure 1: Unfiltered (blue) and filtered (red) GDR profile of a chirped fiber Bragg grating.

The following outlines the main steps required to match the GDR of the WaveShaper with the profile of the CFBG. The profile data to be matched is interpolated to match the .wsp format. E.g. minimum and maximum frequency of 191.250 THz and 196.275 THz respectively, with data equally spaced in 0.001 THz intervals.

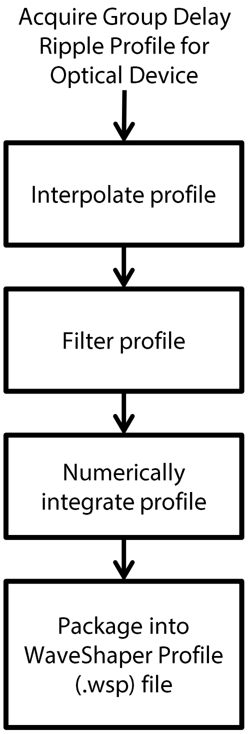
This data is smoothed to remove oscillations which reduce the compensation accuracy. This step is further explained below in Section 5.

The filtered profile is numerically integrated to produce the phase profile required for the WaveShaper.

The phase profile is packaged into a WaveShaper Profile (.wsp) file, and sent to the Waveshaper.

(Optional) The attenuation is flattened.

These steps are described in more detail with code examples in Section 3.


Acquire Group Delay Ripple Profile for Optical Device
Interpolate profile
Filter profile
Numerically integrate profile
Package into WaveShaper Profile (.wsp) file

3. Implementation

Any suitable programming language can be used to perform the following example. The code examples provided are written in MATLAB.

3.1 Interpolation

In this stage, the GDR data is converted to a format native to the WaveShaper. Three steps are performed:

A. The data is loaded from a file named 'toMatch.csv'. The method in which the GDR data is loaded will depend on the format in which it is stored. In this example, the data is stored in a comma separated variable (.csv) file with wavelength in nanometers stored in the 1st column, and GDR in picoseconds stored in the 7th column.

B. The data is converted from wavelength (nm) to frequency (THz).

C. The data is interpolated to match the .wsp format of the WaveShaper. For more information about the .wsp format, see Section 5.3 in the Waveshaper User Manual. In this example, the minimum and maximum frequencies were set to 191.250 THz and 196.275 THz respectively, with an equal spacing of 0.001 THz.

```
%Load Data
rawData = dlmread('toMatch.csv', ',', 1, 0);
lambda = rawData(:,1);
gdr = rawData(:,7);

%Convert to Frequency (THz)
c = 2.99792458e8;
freq = c./lambda*1e-3;

%Interpolate Data

wsFreq = [191.25:0.001:196.275]';
wsGdr = interp1(freq, gdr, wsFreq, 'linear', 0 );
```

3.2 Filtering

Next, the smoothing filter is applied. The details of the following filter are described in more detail in Section 5.

```
CutoffRatio = 0.01;


%Define the Butterworth transfer function
B = (1+(linspace(0,2,length(wsGdr))/CutoffRatio)
.^4).^-0.5;



%Apply the filter
gdr_fft = fft(wsGdr).*B';
gdr_fft(round(length(wsGdr)/2+1):end) = 0;

%inverse FFT to produce the filtered GDR Data
wsFilteredGdr = 2*real(ifft(gdr_fft));
```

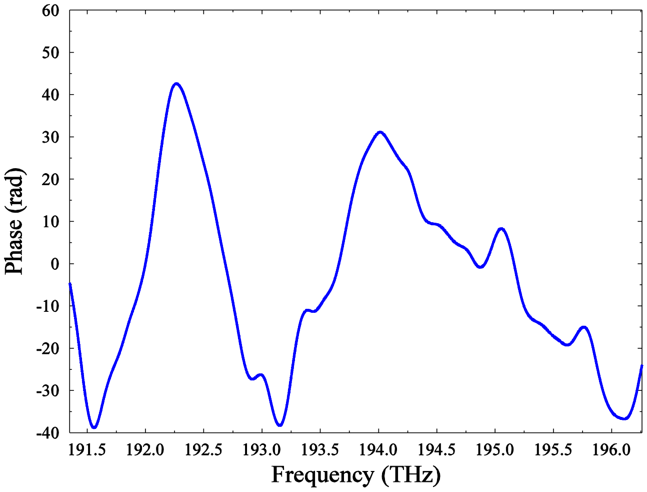


Figure 2: Phase profile generated after numerical integration of filtered GDR profile.

3.3 Profile Integration

In this part, the data must be numerically integrated to produce the **phase** profile required for the WaveShaper. Correct units must be used in the integration, noting that the WaveShaper receives phase information in radians. Eq. 2 can be rewritten in terms of frequency f,

$$\phi_w = -2\pi \int \tau_c.\mathrm{d}f \qquad \ldots (3)$$

In this example, group delay (τ_c) is in units of ps, while frequency (f) is in cycles/ps. The integral itself will be in units of cycles. Multiplying by 2π expresses the final phase profile in radians. A cumulative numerical integration is performed using the trapezoidal rule via the MATLAB function `cumtrapz`.

```
  %Numerically integrate the filtered GDR profile
  wsPhase = -2*pi*cumtrapz(wsFreq, wsFilteredGdr);
```

3.4 WaveShaper Profile creation

This phase data is now ready to be packaged into a .wsp file. The attenuation column is set to either 0 or the attenuation profile generated in Section 6. The port column is set to 1. This example creates the file *GDRCompProfile.wsp*.

```
  %Create the attenuation and port columns.
  wsAttn = zeros(size(wsFreq));
  wsPort =  ones(size(wsFreq));

  %Create the .wsp file
  Filename = 'GDRCompProfile.wsp';
  id = fopen(Filename, 'w+');
  for i=1:length(wsFreq)

      fprintf(id, '%.3f\t%.3f\t%.3f\t%d\n',
wsFreq(i), wsAttn(i), wsPhase(i), wsPort(i));
  end
  fclose(id);
```

4. Results

The effect of filtering (Section 3.2) can be seen in Figure 1 (red). It demonstrates that the profile has been smoothed to a function that can be replicated by a WaveShaper. Figure 2 shows the generated phase profile after numerically integrating the filtered GDR (Section 3.3).

After generating the .wsp file (Section 3.4), it is ready to be sent to the WaveShaper 1000S. The GDR produced by this profile was measured experimentally using a LUNA OVA. Figure 3 shows that the GDR profile of the WaveShaper matches the CFBG profile to within around ±5 ps.

5. Filtering

The GDR profile being compensated may contain smaller details than 10 GHz, the resolution of the WaveShaper. To achieve more accurate results, it is strongly recommended to filter out these details.

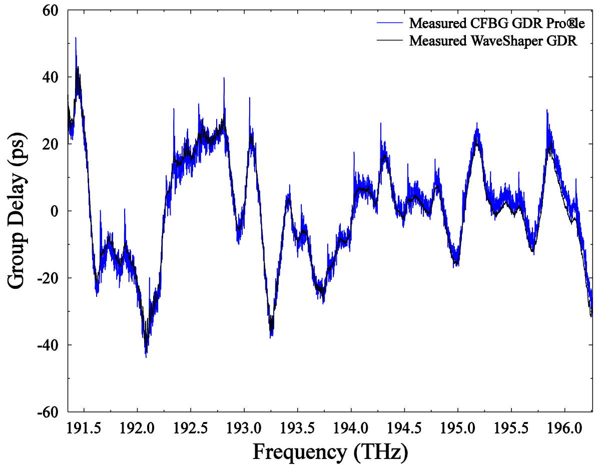


Figure 3: (blue) Original CFBG GDR profile. (black) WaveShaper GDR profile measured using a LUNA OVA.

There are many possible ways to remove such high order details. In this example, a 2nd order Butterworth filter is used. Consider a GDR profile $g(f)$ which, when Fourier transformed, is represented by $G(\alpha)$. It is filtered using a 2nd order Butterworth filter with transfer function:

$$B(\alpha)^2 = \frac{1}{1+(\frac{\alpha}{r\alpha_{max}})^4}, \qquad \ldots (4)$$

where α is the spatial frequency produced by the transform, α_{max} is the maximum spatial frequency, and r is the cutoff ratio. Note that the f dimension is discretised using 1 GHz spacing, so the maximum spatial frequency is then $\alpha_{max} = 1/1GHz$. The cutoff ratio r represents where the -1.5 dB position lies. I.e. features on the GDR profile with a spatial frequency of $r/1GHz$ will suppressed by 1.5 dB. In this example, the cutoff frequency was selected to be $r = 0.01$. This means that a spatial frequency of $0.01/1GHz$, or features of around 100 GHz in size, will be suppressed by 1.5 dB. Features of size 76 GHz are reduced by 3 dB.

The filtered GDR data is found by multiplying G and B and applying the inverse Fourier transform:

$$A(f) = F^{-1}\{G(\alpha)B(\alpha)\}, \qquad \ldots (5)$$

The filtered data $A(f)$ is shown in Figure 1 (red).

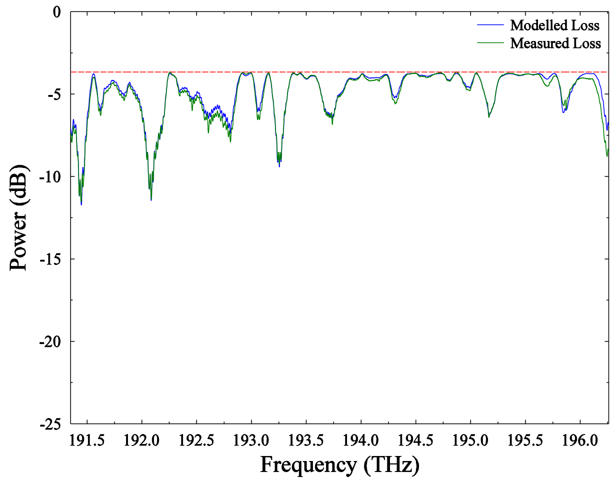


Figure 4: (red) Expected attenuation profile. (blue) Attenuation profile predicted using the WaveShaper API Model. (green) WaveShaper attenuation profile measured experimentally using a LUNA OVA.

6. Attenuation Flattening

Any group delay created using the WaveShaper also undergoes group delay dependent attenuation and is caused by the internal optical components in the WaveShaper. Figure 4 (blue) shows how the WaveShaper experiences parasitic attenuation as a result of the imposed group delay, even when configured with a zero attenuation profile.

The attenuation is approximately proportional to the square of group delay, and turns out to be $A \cong 0.0048\tau^2$ for attenuation A in dB and group delay τ in ps. This attenuation is less than 3 dB for group delays from 0 up to ±25 ps. When compensating for larger group delay ripples such as in the previous example, this attenuation may need to be flattened using the modeling functions found in the API.

By generating a specific attenuation profile, it is possible to flatten the power spectrum of the WaveShaper. The following steps describe the process of generating the required profile:

1. The optical response of the WaveShaper is modeled using the WaveShaper API.

Attenuation data is collected and inverted.

The data is packaged back into the .wsp file (Section 3.4)

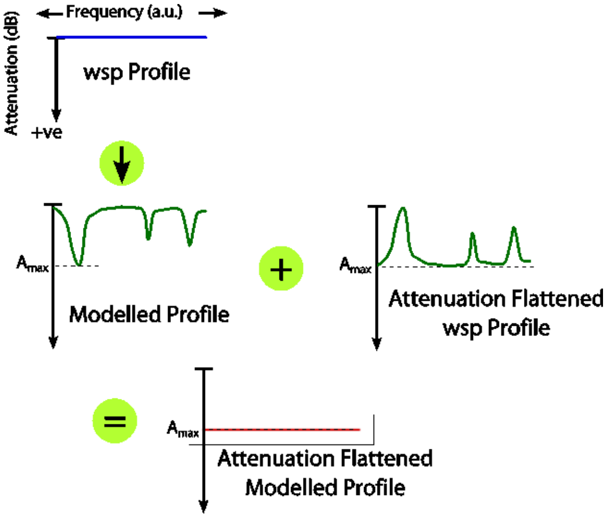


Figure 5: Schematic showing the process of attenuation flattening. (Top) Attenuation profile without attenuation flattening. (Middle Left) After modeling using the API. (Middle Right) Attenuation flattened profile. (Bottom) Attenuation flattened profile modeled using the API.

6.1 Attenuation Modeling

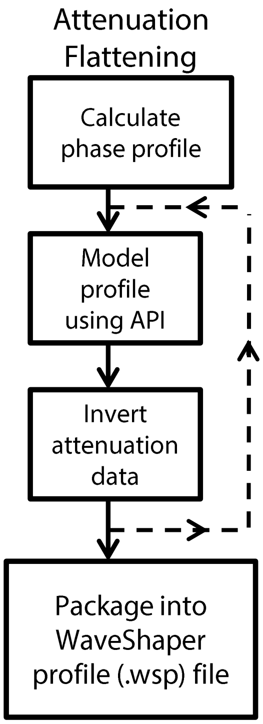
Through use of the Waveshaper API, it is possible to model the optical response of a WaveShaper for a specific profile; this functionality is specifically tailored for a given unit. The attenuation due to phase modulation was modeled, Figure 4 (blue), and closely matches experimental data.

The following steps involve using the WaveShaper API. Please consult the WaveShaper API Programming Manual to determine how to configure your preferred programming language to use the WaveShaper API.

Once a WaveShaper instance has been created using the `ws_create_waveshaper` function, the following steps must be taken to obtain the modeled data using the API:

1. The GDR compensating wsp file (GDRCompProfile.wsp) is turned into a string and loaded using the ws_load_profile_for_modeling function. The input output arguments for this function can be found in the WaveShaper C API header file (ws_api.h), and can be called in a similar fashion to the ws_load_profile function.

Once the profile has successfully loaded, the ws_get_model_profile function is called to generate and return the modeled data. This function will return the WSP representation of the modeled data and behaves in the same way as the ws_get_profile function.



6.2 Inverting Attenuation Data

The second column of the output of the `ws_get_profile` function contains the expected attenuation spectrum. The WaveShaper is a passive device, and can only attenuate light. Thus, additional attenuation must be introduced in order to produce a flat attenuation spectrum. Calculating the correct attenuation profile to use is shown conceptually in Figure 5, where A_{max} is the maximum attenuation (and the minimum power transmitted). Any frequencies attenuating less than this will be made to attenuate to this value. This is written mathematically as

$$A_{new} = A_{max} - A_{model}, \qquad \dots (6)$$

where A_{new} is the flattened attenuation profile and A_{model} is the attenuation profile obtained using the model.

6.3 Re-packaging the wsp file

The same process described in Section 3.5 is performed here using the newly calculated attenuation profile.

6.4 Attenuation Flattening Results

To determine the accuracy of the attenuation flattened profile, it can be passed back into the API modeling function. The modeled results can be seen in Figure 6 (blue), and is flat to within 0.5 dB**.** The attenuation flattened wsp file was then loaded into the WaveShaper, and its optical properties were measured experimentally, shown in Figure 6 (green). Experimental results show that this method has flattened the GDR compensated power spectrum of the WaveShaper to within 2 dB.

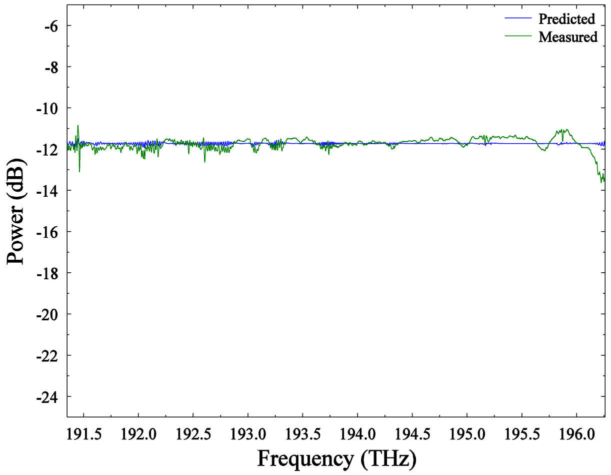


Figure 6: (blue) Predicted power spectrum after attenuation flattening. (green) Power spectrum measured experimentally using LUNA OVA.

6.5 Iterative Approach

When the attenuation flattened profile is modeled by the WaveShaper, it is found that the resulting power spectrum is flat to within 0.5 dB (Figure 6 blue). This result can be improved by repeating step 6.2 iteratively. This iterative process can be described mathematically by the following:

$$\begin{aligned} B_{n-1} &\equiv A_{n-1} - MA_{n-1} \\ A_n &= B_{n-1} - \min(B_{n-1}) \end{aligned} \qquad \dots (7)$$

Where n is the iteration index, and M is an operator corresponding to modeling the data. This enables the flattened profile to be further optimized with each iteration.

Three iterations are applied here and are shown in **Error! Reference source not found.**, with each iteration showing an improvement over the previous. It should be noted that the difference between the modeled and experimental results is larger than the error in the model. This means that even a perfectly modeled flat attenuation profile ($n \to \infty$) will not improve the experimental results. In fact, it was found that more than one iteration did not further improve the experimentally measured power spectrum of the WaveShaper.

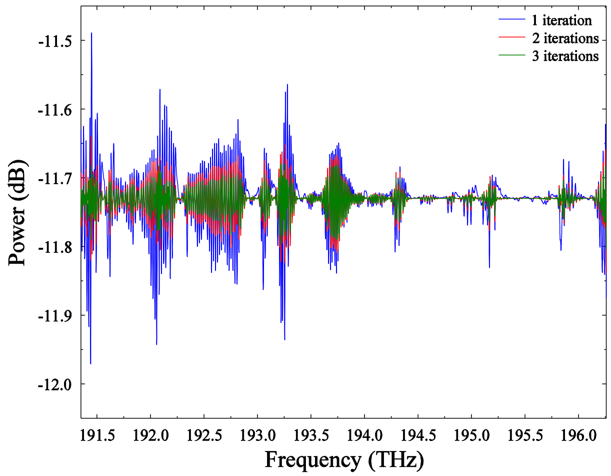


Figure 7: Modeled power spectrum after successive iterations. Blue, red and green represent one, two and three iterations respectively.

It should also be noted that an iterative approach which uses a direct experimental measurement of the optical power spectrum rather than the model could improve results.

1389 Moffett Park Drive
Sunnyvale, CA 94089, USA
Tel.: +1-408-548-1000
Fax: +1-408-541-6138
waveshaper@finisar.com
http://www.finisar.com/optical-instrumentation