Efficient Optimization of Multilayer Coatings for Ultrafast Optics using Analytic Gradients of Dispersion

Jonathan R. Birge and Franz X. Kärtner

We develop a fully analytic method for computing gradients of dispersion (to any order) for a dielectric multilayer coating, and demonstrate how group delay gradients can be used to optimize the dispersion of such a filter. The algorithm complexity is linear with the number of layers and quadratic in dispersion order. To our knowledge, this is the first published algorithm for computing exact analytic gradients of dispersion. We show an approximation that speeds up the computation significantly, making it linear in dispersion order. MATLAB and C code implementing the algorithms are made available.

OCIS codes: 310.6860, 310.0310, 320.0320

1. Introduction

Dispersion compensating dielectric mirrors[1,2] have played a critical role in the development of mode-locked lasers, with state-of-the-art mirror pairs allowing for femtosecond group delay control over nearly an octave of bandwidth.[3] Such precise control of optical phase has enabled pulses containing only a few optical cycles directly from an oscillator.[4,5] Furthermore, the compression or manipulation of pulses outside of the laser cavity requires the design of mirrors with prescribed group delay dispersion over extremely wide bandwidths.[6] Thus, the synthesis of multilayer filters with prescribed phase properties has received increasing interest in the past decade or so.[7,8]

When numerically optimizing a thin film structure, the majority of the computational effort is dedicated to repeatedly computing the gradient of the merit function, and perhaps also the merit function alone (such as during line searches with numerical derivatives). Should the merit function include the spectral dispersion, one must be able to compute the gradient of phase derivatives. While analytic methods have been published for computing gradients of simple reflectivity,[9] no work has been shown on analytically computing gradients of dispersion. To our knowledge, this is the first published algorithm for computing analytic gradients of dispersion, approximate or otherwise.

We recently demonstrated an inductive method[10,11] for computing analytic derivatives of multilayer phase to any order. Here, we extend this method to computing the full analytic gradient of such phase derivatives. To our knowledge, this is the first published

algorithm to compute the exact analytic gradient of dispersion. The method is $O[nm^2]$ in terms of matrix multiplications, where n is the number of layers and m is the dispersion order. Furthermore, we show an approximation that allows for the accurate computation of dispersion gradients in only $O[nm]$, with significant improvement in practice even for $m = 1$ (group delay).

Computing analytic gradients is important for optimizing multilayer coatings for two reasons. First, the use of analytic derivatives avoids the issues of numerical stability associated with finite differences, improving accuracy and convergence.[12] Second, and perhaps most importantly, computing gradients using finite differences results in a gradient algorithm that scales as $O[n^2]$ in the number of layers, making it rather inefficient for complicated mirror systems.

While the general scheme shown in this paper can be applied to the computation of any order of dispersion, the implementation complexity increases for higher orders. Fortunately, group delay alone can be used in a least squares optimization of dispersion, as shown in Section 2. Thus, in this paper we only focus explicitly on gradients of group delay.

Finally, we show an example gradient computation for a chirped mirror used in a mode-locked laser cavity. We give relative timing results for the gradient with and without the aforementioned approximation, and show the resulting difference in computed group delay (GD) and group delay dispersion (GDD).

2. Optimization of Dispersion Using Group Delay Gradients

In most optical systems, zeroth- and first-order phase are irrelevant, representing a carrier phase-shift and overall time-shift, respectively. For this reason, group delay dispersion (GDD) or higher-order dispersion have been typically used to optimize the phase response of filters. However, in the case where weighted least squares minimization is used (at least for dispersion errors) the spectral group delay can be optimized directly by automatically including an error minimizing constant delay as follows.

Consider the portion of the merit function due to group delay, assuming weighted least squares minimization:

$$z_{\mathrm{gd}} \equiv -\sum_{i=1}^{n_k} w_i \big(\tau_g(k_i) - \tau_{g0}(k_i) + \tau^*\big)^2, \tag{1}$$

with $\tau_g(k)$ the group delay of the filter under optimization, $\tau_{g0}(k)$ the ideal delay, w_i the weighting for the error evaluated at n_k points. We've also introduced an arbitrary constant group delay τ^* which will be chosen to minimize the error. To find τ^* we find the stationary point of (1):

$$\frac{\partial z_{\mathrm{gd}}}{\partial \tau^*} = -\sum_i 2w_i \left[\tau_g(k_i) - \tau_{g0}(k_i) + \tau^*\right] = 0. \tag{2}$$

Because of the squared error, the offset is easily isolated. Solving for τ^* gives

$$\tau^* = \frac{1}{W}\sum_i w_i\big(\tau_g(k_i) - \tau_{g0}(k_i)\big), \tag{3}$$

$$\equiv \overline{\tau_g} - \overline{\tau_{g0}}, \tag{4}$$

with W the weighting normalization $\sum_i w_i$ and where we've used an overbar to denote the weighted mean. Thus, the delay offset which minimizes the squared error is simply the difference between the weighted means of the actual and ideal group delays, an intuitive result. Substituting the above into (1) gives

$$z_{\mathrm{gd}} \equiv -\sum_{i=1}^{n_k} w_i\big(\tau_g(k_i) - \tau_{g0}(k_i) + \overline{\tau_g} - \overline{\tau_{g0}}\big)^2. \tag{5}$$

Taking the gradient of (5) with respect to the vector of layer thicknesses $\mathbf{d}$ yields

$$\nabla_{\mathbf{d}}\, z_{\mathrm{gd}} = -\sum_i w_i\big(\tau_g(k_i) - \tau_{g0}(k_i) + \overline{\tau_g} - \overline{\tau_{g0}}\big)\left(\nabla_{\mathbf{d}}\tau_g(k_i) + \overline{\nabla_{\mathbf{d}}\tau_g}\right). \tag{6}$$

This gradient can be used to optimize toward a desired spectral GDD (or whatever order dispersion) by computing the associated ideal GD.

It should be noted that to the extent that the final design has a finite error, minimizing GDD error is not generally the same as minimizing GD error. However, it can be argued that for wide bandwidths, minimizing spectral GD error is preferable to minimizing GDD error. In fact, ideally one would optimize for least magnitude squared error of complex reflectivity (or transmission) modulo a zeroth- and first-order phase term, as that would minimize error energy. However, this would require removing both an error minimizing constant and linear term and thus require solving a two-dimensional system of equations at each optimization step. Furthermore, it would reintroduce the problem of phase unwrapping. However, it is potentially worth pursuing in the future, especially given recent work showing the ineffectiveness of GDD in optimizing chirped mirrors.[13]

3. Analytic Computation of Stack Phase Derivatives

Here, we briefly review the analytic computation of phase derivatives developed previously,[11] including the so-called constant coupling approximation. We leave out most details here, and only cover computation of first-order phase derivatives (group delay). However, the method can be directly extended to any order of dispersion. Further details and a discussion of the validity of the constant coupling approximation can be found in Ref. 11.

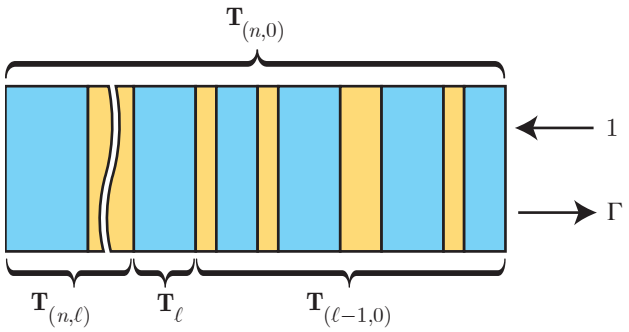

Fig. 1. Diagram showing transfer matrix notation.

3.A. General Case

In this paper we will follow the convention established in the Ref. 11 and consider a dielectric stack whose total transfer matrix is written as

$$\mathbf{T}(k) = \begin{pmatrix} T_{11}(k) & T_{12}(k) \\ T_{12}^*(k) & T_{11}^*(k) \end{pmatrix}. \tag{7}$$

In our notation, $\mathbf{T}_\ell$ refers to the transfer matrix of the ℓth layer, which is defined to include only the interface reflection between it and the previous medium and propagation through the layer. We'll write $\mathbf{T}_{(\ell_2,\ell_1)} \equiv \mathbf{T}_{\ell_2} \cdots \mathbf{T}_{\ell_1+2}\mathbf{T}_{\ell_1+1}$ to refer to the matrix that goes from the end of layer ℓ_1 to the end of layer ℓ_2. The substrate can be handled as a final layer with a thickness of zero.

For convenience, we depart from computing phase derivatives in terms of frequency, as was done in Ref. 11, and use vacuum wavenumber, $k = \omega/c$, instead. This simply avoids having c appear in intermediate formulas, which will help when considering gradients. This also more closely matches the way computation is done in practice, making it easier to compare the results of this paper with the code provided.

In our notation, the transfer matrix operates on a vector whose components are the forward and reverse propagating wave amplitudes, respectively.[14] For reasons that will become clear later, we will write the matrix for the ℓth layer as the product of a full matrix $\mathbf{D}_\ell$, which handles the transfer across the interface, followed by a diagonal matrix $\mathbf{P}_\ell$ that propagates through the layer:

$$\mathbf{T}_\ell \equiv \mathbf{P}_\ell \, \mathbf{D}_\ell \tag{8}$$

$$= \begin{pmatrix} e^{-i\tilde{n}_\ell(k)d_\ell k} & 0 \\ 0 & e^{i\tilde{n}_\ell(k)d_\ell k} \end{pmatrix} \times \frac{1}{2} \begin{pmatrix} 1+p_\ell(k) & 1-p_\ell(k) \\ 1-p_\ell(k) & 1+p_\ell(k) \end{pmatrix}, \tag{9}$$

where $\tilde{n}_\ell(k) \equiv n_\ell(k)\cos\theta_\ell$ is the effective index (which takes into account the propagation

angle θ_ℓ of the wave) and $p_\ell(k)$ is the ratio

$$p_\ell(k) \equiv \begin{cases} \frac{\tilde{n}_{\ell-1}(k)}{\tilde{n}_\ell(k)} & \text{TE polarization,} \\ \frac{\tilde{n}_{\ell-1}(k) n_\ell^2(k)}{\tilde{n}_\ell(k) n_{\ell-1}^2(k)} & \text{TM polarization.} \end{cases} \tag{10}$$

The complex transmission and reflection coefficients are given from the elements of the transfer matrix (8) by

$$\Gamma(k) = -\frac{T_{12}^*(k)}{T_{11}^*(k)}, \tag{11}$$

$$T(k) = T_{11}(k) - \frac{|T_{12}(k)|^2}{T_{11}^*(k)}, \tag{12}$$

respectively.

To determine the mth frequency derivative of phase (either in reflection or transmission) one must know the zeroth through mth derivatives of the elements of the transfer matrix. As discussed in Ref. 11, this can be done in $O[nm^2]$ operations by inductively computing the matrices $\mathbf{T}_{(\ell,0)}$ for ℓ from one to n. In the case of group delay, for example, this means repeatedly computing matrices of the form

$$\mathbf{T}_{(\ell,0)} = \mathbf{T}_\ell \mathbf{T}_{(\ell-1,0)}, \tag{13}$$

$$\frac{\partial \mathbf{T}_{(\ell,0)}}{\partial k} = \frac{\partial \mathbf{T}_\ell}{\partial k} \mathbf{T}_{(\ell-1,0)} + \mathbf{T}_\ell \frac{\partial \mathbf{T}_{(\ell-1,0)}}{\partial k}. \tag{14}$$

3.B. Constant Coupling Approximation

Assuming $p'_\ell(k) \to 0$ yields a significant decrease in complexity for computing dispersion, from $O[m^2]$ to $O[m]$, as experimentally verified in Table 1 of Ref. 11. This implies that we neglect the derivatives of the $\mathbf{D}_\ell(k)$ matrices that couple between forward and backward waves, hence the name. The reason why this is more efficient can be seen by considering the derivative of (8) under the approximation,

$$\mathbf{T}'_\ell(k) \approx \mathbf{P}'_\ell \, \mathbf{D}_\ell \tag{15}$$

$$= \begin{pmatrix} -id_\ell\big[\tilde{n}_\ell(k) + k\tilde{n}'_\ell(k)\big] & 0 \\ 0 & id_\ell\big[\tilde{n}_\ell(k) + k\tilde{n}'_\ell(k)\big] \end{pmatrix} \mathbf{P}'_\ell \, \mathbf{D}_\ell \tag{16}$$

$$\equiv -id_\ell\big[\tilde{n}_\ell(k) + k\tilde{n}'_\ell(k)\big] \boldsymbol{\sigma}_3 \mathbf{T}_\ell(k). \tag{17}$$

For convenience, we have used the Pauli matrix

$$\boldsymbol{\sigma}_3 \equiv \begin{pmatrix} +1 & 0 \\ 0 & -1 \end{pmatrix}, \tag{18}$$

though there is obviously no implied connection between the present application and spinors except where the commutation relations may prove useful (e.g two such derivative approximations in succession cancel to a scalar). The reason for the simple form is

that $\mathbf{P}_\ell$ is diagonal, and so taking the derivative of it is equivalent to left multiplying it with another diagonal matrix. Since the symmetry of transfer matrices is such that we only need to keep track of one row, left multiplication by a diagonal matrix is computationally equivalent to a single scalar multiplication (though it still does not commute, of course, so it cannot be lumped with other scalars). The net result is that the first derivative matrix can be computed using only one matrix multiplication instead of two.

4. Gradients of $\mathbf{T}(k)$

At the core of computing the gradient of group delay is the problem of computing the gradients of $\mathbf{T}_{(n,0)}(k)$ and $\mathbf{T}'_{(n,0)}(k)$ with respect to the n layer thicknesses, denoted as d_ℓ. To begin with, we factor the total transfer matrix to isolate the ℓth layer:

$$\mathbf{T} = \mathbf{T}_{(n,\ell)}\,\mathbf{T}_\ell\,\mathbf{T}_{(\ell-1,0)}. \tag{19}$$

The ℓth gradient element is then simply

$$\frac{\partial \mathbf{T}}{\partial d_\ell} = \mathbf{T}_{(n,\ell)}\,\frac{\partial \mathbf{T}_\ell}{\partial d_\ell}\,\mathbf{T}_{(\ell-1,0)} \tag{20}$$

$$= -ik\tilde{n}_\ell \mathbf{T}_{(n,\ell)}\,\boldsymbol{\sigma}_3\,\mathbf{T}_{(\ell,0)}, \tag{21}$$

where we have used (8) to compute the derivative.

Inspection of (21) immediately suggests the general method for computing the gradients: If we precompute all the front matrices $\mathbf{T}_{(\ell,0)}$ as well as the back matrices $\mathbf{T}_{(n,\ell)}$ for each layer ℓ, then the gradients can be computed trivially in n matrix multiplications. (The multiplication by the Pauli matrix $\boldsymbol{\sigma}_3$ is not counted as it is computationally equivalent to a scalar multiplication, as explained in the previous section.) More importantly, the front matrices can be computed in n matrix multiplications by simply computing them inductively as shown in Section 3. The same is true of the back matrices, though there are some complications that will be covered in Section 5.C. The entire gradient can thus be computed in $O[n]$ matrix multiplications, a significant improvement over the $O[n^2]$ operations required for a naïve finite difference gradient.

5. Gradients of $\mathbf{T}'(k)$

5.A. General Method

The scheme outlined in the previous section can be applied in a straightforward way to find gradients of any order wavenumber derivative, albeit with significant growth in complexity as higher derivatives are used. As justified in Section 2, we will simply demonstrate the method for the first wavenumber derivative used to compute GD. Taking the k derivative of (19), the matrix product rule gives us the following decomposition:

$$\mathbf{T}'(k) = \mathbf{T}'_{(n,\ell)}\,\mathbf{T}_\ell\,\mathbf{T}_{(\ell-1,0)} + \mathbf{T}_{(n,\ell)}\,\mathbf{T}'_\ell\,\mathbf{T}_{(\ell-1,0)} + \mathbf{T}_{(n,\ell)}\,\mathbf{T}_\ell\,\mathbf{T}'_{(\ell-1,0)}. \tag{22}$$

The ℓth gradient element is then

$$\frac{\partial \mathbf{T}'(k)}{\partial d_\ell} = -ik\tilde{n}_\ell \mathbf{T}'_{(n,\ell)}\,\boldsymbol{\sigma}_3\,\mathbf{T}_{(\ell,0)} + \mathbf{T}_{(n,\ell)}\,\frac{\partial \mathbf{T}'_\ell}{\partial d_\ell}\,\mathbf{T}_{(\ell-1,0)} - ik\tilde{n}_\ell \mathbf{T}_{(n,\ell)}\,\boldsymbol{\sigma}_3\,\mathbf{T}'_{(\ell,0)}, \tag{23}$$

where we've applied the result in (21) to simplify the outer two terms. From the definition of T_ℓ in (8) we obtain

$$\begin{aligned}\frac{\partial \mathbf{T}'(k)}{\partial d_\ell} = &-ik\tilde{n}_\ell \left[\mathbf{T}'_{(n,\ell)}\,\boldsymbol{\sigma}_3\,\mathbf{T}_{(\ell,0)} + \mathbf{T}_{(n,\ell)}\,\boldsymbol{\sigma}_3\,\mathbf{T}'_{(\ell,0)}\right] + \\ &\frac{1}{2p}\mathbf{T}_{(n,\ell)}\begin{pmatrix} 2p(d_\ell k\tilde{n}+i)(\tilde{n}+k\tilde{n}') - ik\tilde{n}p' & -ik\tilde{n}p'e^{-i2d_\ell k\tilde{n}} \\ ik\tilde{n}p'e^{i2d_\ell k\tilde{n}} & 2p(d_\ell k\tilde{n}-i)(\tilde{n}+k\tilde{n}') + ik\tilde{n}p' \end{pmatrix}\mathbf{T}_{(\ell,0)},\end{aligned} \tag{24}$$

where all primes refer to wavenumber derivatives and we've dropped some of the unambiguous ℓ subscripts for convenience. In arriving at the above, we solved for the matrix which takes the simultaneous wavenumber and layer thickness derivative of $\mathbf{T}_\ell$.

The front and back derivative matrices, $\mathbf{T}'_{(\ell,0)}$ and $\mathbf{T}'_{(n,\ell)}$ respectively, are found using the exact methods of Section 3. The front matrices are available directly as they are the intermediate results of computing $\mathbf{T}'$. The back derivative matrices require extra computation, however, and can be found by proceeding through the stack in reverse, doing right multiplications in lieu of left multiplications. (In the case where the constant coupling approximation is used, there is a more efficient way to compute the back matrices, discussed in Section 5.C.)

Higher order dispersion terms beyond what we've shown here can be computed similarly, and the computation of the front and back matrices will scale as $O[nm^2]$, as per Section 3. However, the number of matrix multiplications required for the final matrices [e.g. equation (24)] grows exponentially, as $O[3^m]$, and the complexity of the *elements* in each matrix increases considerably. Thus, higher order dispersion quickly becomes infeasible with this method, and even the $m = 1$ case (for group delay) is rather complex, as can be seen from (24). In the next section, we will show how to use the constant coupling approximation to greatly simplify the gradient computation, significantly speeding up low order dispersion and enabling the gradient computation of higher order dispersion.

5.B. Constant Coupling Approximation

We have already seen how the assumption that $\mathbf{D}'_\ell(k) \to 0$ greatly speeds up the computation of the front and back matrices, as needed for (24). However, it also greatly simplifies the final terms in (24). Under the constant coupling approximation, the off diagonal terms in the last product term vanish, leaving a trivial scalar multiplication in lieu of a matrix

product. Thus, equation (24) becomes

$$\frac{\partial \mathbf{T}'(k)}{\partial d_\ell} = -ik\tilde{n}_\ell \left[\mathbf{T}'_{(n,\ell)}\boldsymbol{\sigma}_3\mathbf{T}_{(\ell,0)} + \mathbf{T}_{(n,\ell)}\boldsymbol{\sigma}_3\mathbf{T}'_{(\ell,0)}\right] + \frac{i(\tilde{n}+k\tilde{n}')}{2p}\mathbf{T}_{(n,\ell)}\boldsymbol{\sigma}_3\mathbf{T}_{(\ell,0)} + d_\ell k\tilde{n}(\tilde{n}+k\tilde{n}')\mathbf{T}, \quad (25)$$

where the third term is a scalar multiplication of the zeroth-order gradient element from (24), and the last term is just a scaling of the total transfer matrix.

In general, the simplification afforded by the constant coupling approximation not only takes the front and back matrix computation from $O[nm^2]$ to $O[nm]$, but also makes the final matrix expression [e.g. (25)] scale as $O[2^m]$ instead of $O[3^m]$, as all terms consist of only two full matrices. In the specific case of group delay, the total speedup in practice is roughly a factor of two, assuming n is large enough such that the computation of front and back matrices dominate.[11]

Finally, the approximation makes it reasonable to compute GDD gradients, affording a speed-up of roughly a factor of four. Taking another k derivative of (25) and combining like terms gives

$$\frac{\partial \mathbf{T}''(k)}{\partial d_\ell} = -i[(n+kn')/2p - n]\left[\mathbf{T}'_{(n,\ell)}\boldsymbol{\sigma}_3\mathbf{T}_{(\ell,0)} + \mathbf{T}_{(n,\ell)}\boldsymbol{\sigma}_3\mathbf{T}'_{(\ell,0)}\right] - 2ikn\left[\mathbf{T}'_{(n,\ell)}\boldsymbol{\sigma}_3\mathbf{T}'_{(\ell,0)} + \mathbf{T}''_{(n,\ell)}\boldsymbol{\sigma}_3\mathbf{T}_{(\ell,0)} + \mathbf{T}_{(n,\ell)}\boldsymbol{\sigma}_3\mathbf{T}''_{(\ell,0)}\right] + \frac{in'}{2p}\mathbf{T}_{(n,\ell)}\boldsymbol{\sigma}_3\mathbf{T}_{(\ell,0)} + (d_\ell\tilde{n}^2 + 2d_\ell nn'k)\mathbf{T} + d_\ell nk(\tilde{n}+\tilde{n}'k)\mathbf{T}'. \quad (26)$$

The individual terms above are all derived in Ref. 11. While it is certainly possible to compute GDD gradients without the constant coupling approximation, the final gradient terms become extremely cumbersome. Fortunately, given the accuracy of the constant coupling approximation for GDD, as demonstrated in Fig. 3, there is little reason to use exact GDD computations except perhaps as a final refinement step.

5.C. Efficient Computation of Back Derivative Matrices

An element of extending the constant coupling method of Section 3.B to gradients that is not straightforward is the issue of efficiently handling the back matrices, $\mathbf{T}_{(n,\ell)}$, which take the fields from the interior of the stack to the end. The efficiency of the constant coupling approximation hinges on the fact that we build the full matrix from successive left multiplications of $\mathbf{PD}$ layers, as in (8). Were we to simply compute the matrices by using right multiplications and working our way backwards from the end, therefore, the constant coupling approximation would not yield any advantage.

The way around this is to actually compute the back matrices as the "front" matrices for the reversed stack. This can be done without having to recompute any of the individual

transfer matrices by using the reversal theorem of transfer matrices[15]

$$\mathbf{T}^R = \frac{\mathbf{T}^\dagger}{|\mathbf{T}|}, \tag{27}$$

where $\mathbf{T}^R$ denotes the transfer matrix for the reversed stack. In terms of a specific layer, we must also take into account the fact that the propagation must occur after the boundary. With this in mind, we can write a single layer of the reversed stack in terms of the components of the original stack,

$$\mathbf{T}^R_{n-\ell+1} = \mathbf{P}^\dagger_\ell \frac{\mathbf{D}^\dagger_{\ell-1}}{|\mathbf{D}_{\ell-1}|}. \tag{28}$$

Note that the propagation matrix is once again exposed on the left side as in (15).

If we were to have to compute all of the determinants arising from (28), the extra complexity would mitigate any advantage of the constant coupling approximation. Fortunately, however, we can safely ignore the determinants as they cancel in the end. To see how, consider the computation for $\mathbf{T}_{(n,\ell)}$ in terms of the reverse stack,

$$\mathbf{T}_{(n,\ell)} = \frac{\left(\mathbf{T}^R_{(\ell,0)}\right)^\dagger}{\left|\mathbf{T}^R_{(\ell,0)}\right|} \tag{29}$$

$$= \frac{\left[\frac{\mathbf{D}^\dagger_{\ell-1}}{|\mathbf{D}_{\ell-1}|}\left(\mathbf{P}^\dagger_{\ell-1}\frac{\mathbf{D}^\dagger_\ell}{|\mathbf{D}_\ell|}\right)\cdots\left(\mathbf{P}^\dagger_{n-1}\frac{\mathbf{D}^\dagger_n}{|\mathbf{D}_n|}\right)\right]^\dagger}{\left|\frac{\mathbf{D}^\dagger_{\ell-1}}{|\mathbf{D}_{\ell-1}|}\left(\mathbf{P}^\dagger_{\ell-1}\frac{\mathbf{D}^\dagger_\ell}{|\mathbf{D}_\ell|}\right)\cdots\left(\mathbf{P}^\dagger_{n-1}\frac{\mathbf{D}^\dagger_n}{|\mathbf{D}_n|}\right)\right|}. \tag{30}$$

The groups in parentheses represent the individual layer matrices of the reversed stack. Moving the determinants in the denominator outside the surrounding determinant yields the product of the squared determinants (since we are dealing with 2×2 matrices), giving us

$$\mathbf{T}_{(n,\ell)} = \frac{\frac{\left[\mathbf{D}^\dagger_{\ell-1}\left(\mathbf{P}^\dagger_{\ell-1}\mathbf{D}^\dagger_\ell\right)\cdots\left(\mathbf{P}^\dagger_{n-1}\mathbf{D}^\dagger_n\right)\right]^\dagger}{|\mathbf{D}_{\ell-1}||\mathbf{D}_\ell||\mathbf{D}_n|}}{\frac{\left|\mathbf{D}^\dagger_{\ell-1}\left(\mathbf{P}^\dagger_{\ell-1}\mathbf{D}^\dagger_\ell\right)\cdots\left(\mathbf{P}^\dagger_{n-1}\mathbf{D}^\dagger_n\right)\right|}{(|\mathbf{D}_{\ell-1}||\mathbf{D}_\ell||\mathbf{D}_n|)^2}}. \tag{31}$$

The determinants then all cancel (the determinant of a propagation matrix is one) yielding a very simple and direct way to go from the individual layer matrices to the back matrices,

$$\mathbf{T}_{(n,\ell)} = \left[\mathbf{D}^\dagger_{\ell-1}\left(\mathbf{P}^\dagger_{\ell-1}\mathbf{D}^\dagger_\ell\right)\cdots\left(\mathbf{P}^\dagger_{n-1}\mathbf{D}^\dagger_n\right)\right]^\dagger \tag{32}$$

This expression has two advantages. First, and most importantly, it allows us to us to build up the back matrices using successive left multiplications of $\mathbf{PD}$ matrix pairs, enabling the use of the fast approximate algorithm discussed in Section 3.B to find the k derivatives of (32). Second, everything on the right hand side of (32) has already been computed in finding the front matrices, $\mathbf{T}_{(\ell,0)}$. Consult the code referenced in Section 9 for further details and a demonstration.

6. Dispersion Gradients from Matrix Gradients

Having computed gradients of the full transfer matrix and its k derivatives, the final step in any optimization will be the translation of those values into gradients of dispersion for use in the merit function gradient computation. For reference, we provide formulas for the case of reflection group delay:

$$\Gamma'(k) = \frac{T_{12}T'_{11} - T'_{12}T_{11}}{T_{11}^2}, \tag{33}$$

$$\phi'(k) = \frac{\Im[\Gamma']\Re[\Gamma] - \Re[\Gamma']\Im[\Gamma]}{|\Gamma|^2}, \tag{34}$$

$$\nabla\Gamma(k) = -\frac{1}{T_{11}}(\Gamma\,\nabla T_{11} + \nabla T_{12}), \tag{35}$$

$$\nabla\,|\Gamma(k)|^2 = 2\left(\Re[\nabla\Gamma]\Re[\Gamma] + \Im[\nabla\Gamma]\Im[\Gamma]\right), \tag{36}$$

$$\nabla\Gamma'(k) = \frac{1}{T_{12}^2}\left[\nabla T_{12}T'_{11} + \nabla T_{11}T'_{12} + \Gamma(2\nabla T_{11}T'_{11} - T_{11}\nabla T'_{11} - T_{11}\nabla T'_{12})\right], \tag{37}$$

$$\nabla\phi(k) = \frac{\Im[\nabla\Gamma]\Re[\Gamma] - \Re[\nabla\Gamma]\Im[\Gamma]}{|\Gamma|^2}, \tag{38}$$

$$\nabla\phi'(k) = \frac{1}{|\Gamma|^2}\left[\Im[\nabla\Gamma']\Re[\Gamma] - \Re[\nabla\Gamma']\Im[\Gamma] - \nabla\phi\big(\Re[\Gamma']\Re[\Gamma] - \Im[\Gamma']\Im[\Gamma]\big) + \frac{\phi'\nabla\,|\Gamma|^2}{2}\right]. \tag{39}$$

In the above, all gradients are with respect to the layer thicknesses, and all primes refer to k derivatives.

7. Algorithm Overview

Here we summarize the overall dispersion gradient algorithm. The general steps are:

1. Precompute all individual layer transfer matrices $\mathbf{P}_\ell$ and $\mathbf{D}_\ell$ and their derivatives with respect to k, .
2. Iteratively compute front transfer matrices $\mathbf{T}_{(\ell,0)}$ and their derivatives for $n-1 > \ell > 1$, as demonstrated in Section 3 and Ref. 11.
3. Compute back transfer matrices $\mathbf{T}_{(n,\ell)}$ and derivatives by repeating Step 2 for the reversed stack and then taking the hermitian transpose, as shown in (32) and explained in Section 5.C.
4. Loop through all layers, computing matrix gradient elements of $\nabla\mathbf{T}$ and $\nabla\mathbf{T}'$ as per the exact expression (21) or the approximate expression (25).
5. Translate matrix gradients into reflectivity and group delay gradients as done in Section 6.

6. Finally, compute merit function gradients from the matrix gradients found in step 5, using the results from Section 2 when group delay error is part of the merit function.

8. Example Gradient Computation

To demonstrate the efficacy of the constant coupling approximate gradient algorithm, we have computed the gradient of group delay at a central wavelength for a typical chirped mirror. For reference, the chirped mirror spectral GD and GDD are plotted in Figures 2 and 3, respectively. The gradient of group delay at 800 nm is shown with and without the constant coupling approximation in Figure 4, illustrating how close the approximate version is to the exact computation.

When computing the gradient at 256 wavelengths, the constant coupling gradient was empirically found to be 63% faster than the exact gradient algorithm, demonstrating that computational savings in the gradient extend from the algorithms given in Ref. 11. We thus estimate that our GD gradient method is roughly four times faster than computing gradients using GDD.

Note that the gradient itself is exact (i.e. it is an exact analytic gradient of an approximate group delay) so using the approximation should not affect the convergence rate, but may have a small effect on the final solution. We expect that, on average, optimizations will be made correspondingly faster. Should final group delay error be so small that the constant coupling approximation error is no longer negligible, a few refinement steps can be performed at the end with the exact gradient to converge at the exact solution.

9. MATLAB and C Code

To simplify use of these algorithms, we have provided MATLAB and C code of both the GD gradient algorithms discussed in this paper, as well as the GD and GDD algorithms from Ref. 11. The C codes are written as MATLAB MEX functions to enable their use in optimization routines within MATLAB, though each function is available as a stand-alone for use in user programs. Several ancillary MATLAB functions that facilitate mirror optimization using the Optimization Toolbox are also included. The code is available at *http://www.mit.edu/~birge/dispersion*.

10. Conclusions

We have demonstrated the extension of the fast group delay algorithm in Reference 11 to the $O[n]$ computation of group delay gradients, and have shown how dispersion optimization can be done using only group delay instead of costly group delay dispersion calculations. In addition to providing a significant speed advantage, this method uses analytic derivatives of both phase and layer thickness. This avoids the numerical difficulties

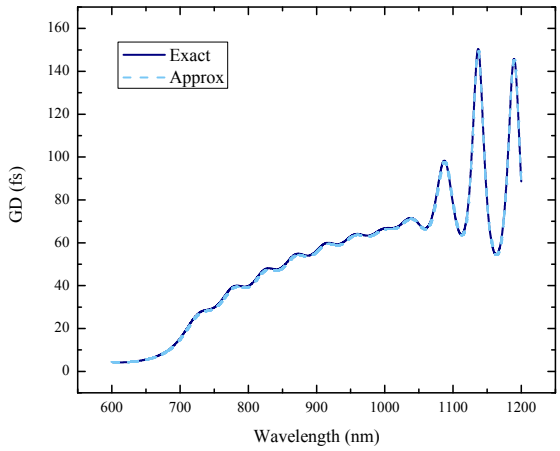

Fig. 2. (Color online) Spectral group delay of example chirped mirror. A portion of the response past the high reflectivity region (wavelengths greater than about 1050 nm) is shown to demonstrate that the approximation even holds when the group delay is rapidly varying.

inherent with finite differences, and saves the user from being concerned with issues of optimal grid spacing or phase unwrapping.

11. Acknowledgements

The authors thank Christian Jirauschek for helpful discussions. This work was supported, in part, by NSF grant ECS-0501478 and DARPA grant HR0011-05-C-0155.

References

1. A. Stingl, M. Menzner, C. Spielmann, F. Krausz, and R. Szipöcs, "Sub-10-fs mirror-dispersion-controlled Ti:sapphire laser," Opt. Lett. **20**, 602–604 (1995).
2. F. X. Kärtner, N. Matuschek, T. Schibli, U. Keller, H. A. Haus, C. Heine, R. Morf, V. Scheuer, M. Tilsch, and T. Tschudi, "Design and fabrication of double-chirped mirrors," Opt. Lett. **22**, 831–833 (1997).
3. F. X. Kaertner, U. Morgner, T. R. Schibli, E. P. Ippen, J. G. Fujimoto, V. Scheuer, G. Angelow, and T. Tschudi, "Ultrabroadband double-chirped mirror pairs for generation of octave spectra," J. Opt. Soc. Am. B **18**, 882–885 (2001).
4. U. Morgner, F. X. Kärtner, S. H. Cho, Y. Chen, H. A. Haus, J. G. Fujimoto, and E. P. Ippen, "Sub-two-cycle pulses from a Kerr-lens mode-locked Ti:sapphire laser," Opt. Lett. **24**, 411–413 (1999).

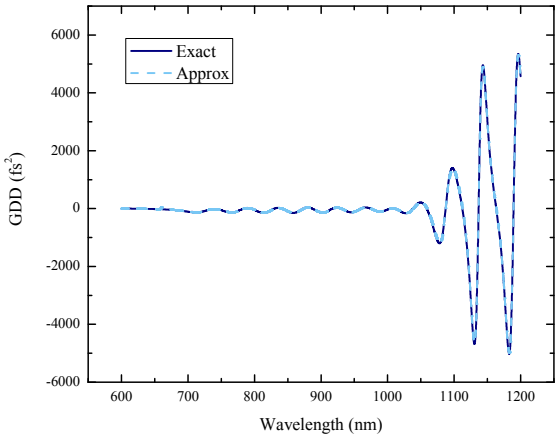

Fig. 3. (Color online) Spectral group delay dispersion of chirped mirror shown in Fig. 2.

5. T. R. Schibli, O. Kuzucu, J. Kim, E. P. Ippen, J. G. Fujimoto, F. X. Kärtner, V. Scheuer, and G. Angelow, "Towards single-cycle laser systems," IEEE J. Sel. Top. Quant. Elec. **9**, 990–1001 (2003).
6. O. D. Mücke, R. Ell, A. Winter, J. Kim, J. R. Birge, L. Matos, and F. X. Kärtner, "Self-referenced 200 MHz octave-spanning Ti:saphhire laser with 50 attosecond carrier-envelope phase jitter," Opt. Express **13**(5163) (2005).
7. A. V. Tikhonravov, "Some theoretical apsects of thin-film optics and their applications," Appl. Opt. **32**, 5417–5426 (1993).
8. A. V. Tikhonravov, P. W. Baumeister, and K. V. Popov, "Phase properties of multilayers," Appl. Opt. **36**, 4382–4392 (1997).
9. C. J. v. d. Laan and H. J. Frankena, "Fast computation method for derivatives of multilayer stack reflectance," Appl. Opt. **17**, 538–541 (1978).
10. J. R. Birge, C. Jirauschek, and F. X. Kärtner, "Efficient analytic computation of group delay dispersion from optical interference coatings," in *OSA Opt. Interference Coatings Top. Mtg.*, p. ThA6 (Tucson, 2004).
11. J. R. Birge and F. X. Kärtner, "Efficient analytic computation of dispersion from multilayer structures," Appl. Opt. **45**, 1478–1483 (2006).
12. K. Atkinson, *An Introduction to Numerical Analysis* (Wiley, New York, 1989).
13. P. Dombi, V. S. Yakovlev, K. O'Keeffe, T. Fuji, M. Lezius, and G. Tempea, "Pulse compression with time-domain optimized chirped mirrors," Opt. Express **13**, 10,888–10,894 (2005).
14. J. Kong, *Electromagnetic Theory* (EMW, 2001).
15. A. A. Tovar and L. W. Casperson, "Generalized reverse theorems for multipass applications

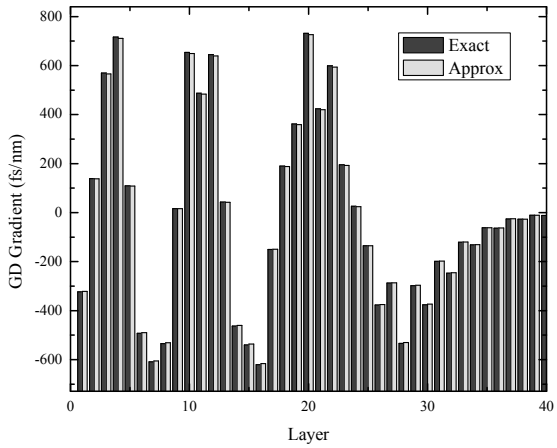

Fig. 4. Exact and approximate gradient of the group delay of the filter shown in Fig. 2 at 800 nm. The gradient approaches zero for layers past the 40th as virtually all of the light is reflected by that point.

in matrix optics," J. Opt. Soc. Am. A **11**, 2633–2642 (1994).