Efficient optimization of multilayer coatings for ultrafast optics using analytic gradients of dispersion

Jonathan R. Birge and Franz X. Kärtner

A fully analytic method for computing gradients of dispersion (to any order) for a dielectric multilayer coating is developed, and it is demonstrated 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. © 2007 Optical Society of America

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.

Although 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 focus explicitly only on gradients of group delay.

Finally, we show an example gradient computation for a chirped mirror used in a mode-locked laser cav-

The authors are with the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, Massachusetts 02139, USA. J. R. Birge's e-mail address is birge@mit.edu.

Received 11 August 2006; revised 5 January 2007; accepted 15 January 2007; posted 17 January 2007 (Doc. ID 73992); published 23 April 2007.

0003-6935/07/142656-07$15.00/0

ity. 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, 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 GD can be optimized directly by automatically including an error minimizing constant delay as follows.

Consider the portion of the merit function due to GD, assuming weighted least-squares minimization:

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

with $\tau_g(k)$ being the group delay of the filter under optimization, $\tau_{g0}(k)$ being the ideal delay, and w_i being the weighting for the error evaluated at n_k points. We have 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 Eq. (1):

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

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

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

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

with W the weighting normalization $\sum_i w_i$ and where we have 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 GDs, an intuitive result. Substituting the above into Eq. (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 Eq. (5) with respect to the vector of layer thicknesses $\mathbf{d}$ yields

$$\nabla_{\mathbf{d}} z_{\mathrm{GD}} = -2\sum_i w_i\big[\tau_g(k_i) - \tau_{g0}(k_i) + \overline{\tau_g} - \overline{\tau_{g0}}\big]\big[\nabla_{\mathbf{d}}\tau_g(k_i) + \overline{\nabla_{\mathbf{d}}\tau_g}\big]. \tag{6}$$

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 a linear term and thus require solving a 2D 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 cover only computation of first-order phase derivatives (GD). 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.

A. General Case

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

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

In our notation, $\mathbf{T}_l$ 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. Refering to Fig. 1, we'll write $\mathbf{T}_{(l_2,l_1)} \equiv \mathbf{T}_{l_2} \ldots \mathbf{T}_{l_1+2}\mathbf{T}_{l_1+1}$ 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.

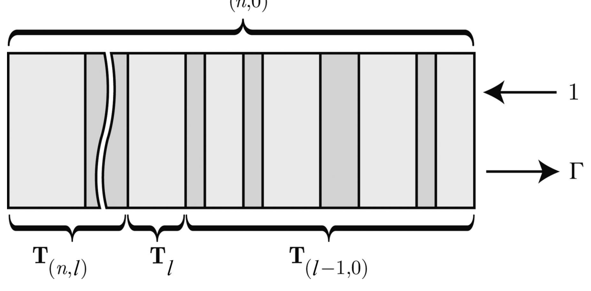

Fig. 1. Diagram showing transfer matrix notation.

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 $\mathbf{D}_l$, which handles the transfer across the interface, followed by a diagonal matrix $\mathbf{P}_l$ that propagates through the layer:

$$\mathbf{T}_l \equiv \mathbf{P}_l\mathbf{D}_l \tag{8}$$

$$= \begin{bmatrix} e^{-i\tilde{n}_l(k)d_lk} & 0 \\ 0 & e^{i\tilde{n}_l(k)d_lk} \end{bmatrix} \times \frac{1}{2}\begin{bmatrix} 1+p_l(k) & 1-p_l(k) \\ 1-p_l(k) & 1+p_l(k) \end{bmatrix}, \tag{9}$$

where $\tilde{n}_l(k) \equiv n_l(k)\cos\theta_l$ is the effective index (which takes into account the propagation angle θ_l of the wave), d_l is the layer thickness, and $p_l(k)$ is the ratio

$$p_l(k) \equiv \begin{cases} \dfrac{\tilde{n}_{l-1}(k)}{\tilde{n}_l(k)} & \text{TE polarization,} \\ \dfrac{\tilde{n}_{l-1}(k)n_l^2(k)}{\tilde{n}_l(k)n_{l-1}^2(k)} & \text{TM polarization.} \end{cases} \tag{10}$$

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 $\mathbf{T}_{(l,0)}$ for l from one to n. In the case of GD, for example, this means repeatedly computing matrices of the form

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

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

B. Constant Coupling Approximation

Assuming $p_l'(k) \to 0$ 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 $\mathbf{D}_l(k)$ 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 Eq. (8) under the approximation

$$\mathbf{T}_l'(k) \approx \mathbf{P}_l'\mathbf{D}_l \tag{15}$$

$$= \begin{bmatrix} -id_l[\tilde{n}_l(k) + k\tilde{n}_l'(k)] & 0 \\ 0 & id_l[\tilde{n}_l(k) + k\tilde{n}_l'(k)] \end{bmatrix}\mathbf{P}_l\mathbf{D}_l \tag{16}$$

$$\equiv -id_l[\tilde{n}_l(k) + k\tilde{n}_l'(k)]\boldsymbol{\sigma}_3\mathbf{T}_l(k). \tag{17}$$

For convenience, we have used the Pauli matrix

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

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 $\mathbf{P}_l$ is diagonal, and thus 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 need to keep track of only 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 T(*k*)

At the core of computing the gradient of GD 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. To begin with, we factor the total transfer matrix to isolate the lth layer:

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

The lth gradient element is then simply

$$\frac{\partial\mathbf{T}}{\partial d_l} = \mathbf{T}_{(n,l)}\frac{\partial\mathbf{T}_l}{\partial d_l}\mathbf{T}_{(l-1,0)}, \tag{20}$$

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

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

An inspection of Eq. (21) immediately suggests the general method for computing the gradients: If we precompute all the front matrices $\mathbf{T}_{(l,0)}$ as well as the back matrices $\mathbf{T}_{(n,l)}$ 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 Subsection 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 naive finite difference gradient.

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

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 Eq. (19), the matrix product rule gives us the following decomposition:

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

The lth gradient element is then

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

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

$$\frac{\partial \mathbf{T}'(k)}{\partial d_l} = -ik\tilde{n}_l[\mathbf{T}'_{(n,l)}\boldsymbol{\sigma}_3\mathbf{T}_{(l,0)} + \mathbf{T}_{(n,l)}\boldsymbol{\sigma}_3\mathbf{T}'_{(l,0)}] + \frac{1}{2p}\mathbf{T}_{(n,l)}\begin{bmatrix} 2p(d_lk\tilde{n}+i)(\tilde{n}+k\tilde{n}') - ik\tilde{n}p' & -ik\tilde{n}p'e^{-i2d_lk\tilde{n}} \\ ik\tilde{n}p'e^{i2d_lk\tilde{n}} & 2p(d_lk\tilde{n}-i)(\tilde{n}+k\tilde{n}') + ik\tilde{n}p' \end{bmatrix}\mathbf{T}_{(l,0)}, \tag{24}$$

where all primes refer to wavenumber derivatives, and we have dropped some of the unambiguous l subscripts for convenience. In arriving at the above, we solved for the matrix that takes the simultaneous wavenumber and layer thickness derivative of $\mathbf{T}_l$.

The front and back derivative matrices, $\mathbf{T}'_{(l,0)}$ and $\mathbf{T}'_{(n,l)}$, 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 Subsection 5.C.)

Higher-order dispersion terms beyond what we have shown here can be computed similarly, and the computation of the front and back matrices will scale as $O[nm^2]$, according to Section 3. However, the number of matrix multiplications required for the final matrices [e.g., Eq. (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 GD) is rather complex as can be seen from Eq. (24). In the next subsection, 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.

B. Constant Coupling Approximation

We have already seen how the assumption that $\mathbf{D}'_l(k) \to 0$ greatly speeds up the computation of the front and back matrices, as needed for Eq. (24). However, it also greatly simplifies the final terms in Eq. (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 Eq. (24) becomes

$$\frac{\partial \mathbf{T}'(k)}{\partial d_l} = -ik\tilde{n}_l[\mathbf{T}'_{(n,l)}\boldsymbol{\sigma}_3\mathbf{T}_{(l,0)} + \mathbf{T}_{(n,l)}\boldsymbol{\sigma}_3\mathbf{T}'_{(l,0)}] + \frac{i(\tilde{n}+k\tilde{n}')}{2p}\mathbf{T}_{(n,l)}\boldsymbol{\sigma}_3\mathbf{T}_{(l,0)} + d_lk\tilde{n}(\tilde{n}+k\tilde{n}')\mathbf{T}, \tag{25}$$

where the third term is a scalar multiplication of the zeroth-order gradient element from Eq. (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., Eq. (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 GD, 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 speedup of roughly a factor of four. Taking another k derivative of Eq. (25) and combining like terms gives

$$\begin{aligned}\frac{\partial \mathbf{T}''(k)}{\partial d_l} = {} & -i[(n+kn')/2p - n][\mathbf{T}'_{(n,l)}\boldsymbol{\sigma}_3\mathbf{T}_{(l,0)} \\ & + \mathbf{T}_{(n,l)}\boldsymbol{\sigma}_3\mathbf{T}'_{(l,0)}] - 2ikn[\mathbf{T}'_{(n,l)}\boldsymbol{\sigma}_3\mathbf{T}'_{(l,0)} \\ & + \mathbf{T}''_{(n,l)}\boldsymbol{\sigma}_3\mathbf{T}_{(l,0)} + \mathbf{T}_{(n,l)}\boldsymbol{\sigma}_3\mathbf{T}''_{(l,0)}] \\ & + \frac{in'}{2p}\mathbf{T}_{(n,l)}\boldsymbol{\sigma}_3\mathbf{T}_{(l,0)} + (d_l\tilde{n}^2 + 2d_lnn'k)\mathbf{T} \\ & + d_lnk(\tilde{n}+\tilde{n}'k)\mathbf{T}'. \end{aligned} \tag{26}$$

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 below, there is little reason to use exact GDD computations except perhaps as a final refinement step.

C. Efficient Computation of Back Derivative Matrices

An element of extending the constant coupling method of Subsection 3.B to gradients, which is not straightforward, is the issue of efficiently handling the back matrices, $\mathbf{T}_{(n,l)}$, 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 **PD** layers, as in Eq. (8). Were we to simply compute the matrices by using right multiplications and working our way backward from the end, 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,

$$\mathbf{T}^R_{n-l+1} = \mathbf{P}_l^\dagger\,\frac{\mathbf{D}_{l-1}^\dagger}{|\mathbf{D}_{l-1}|}. \tag{28}$$

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

If we were to have to compute all of the determinants arising from Eq. (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 $\mathbf{T}_{(n,l)}$ in terms of the reverse stack:

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

$$= \frac{\left[\dfrac{\mathbf{D}_{l-1}^\dagger}{|\mathbf{D}_{l-1}|}\left(\mathbf{P}_{l-1}^\dagger\dfrac{\mathbf{D}_l^\dagger}{|\mathbf{D}_l|}\right)\ldots\left(\mathbf{P}_{n-1}^\dagger\dfrac{\mathbf{D}_n^\dagger}{|\mathbf{D}_n|}\right)\right]^\dagger}{\left|\dfrac{\mathbf{D}_{l-1}^\dagger}{|\mathbf{D}_{l-1}|}\left(\mathbf{P}_{l-1}^\dagger\dfrac{\mathbf{D}_l^\dagger}{|\mathbf{D}_l|}\right)\ldots\left(\mathbf{P}_{n-1}^\dagger\dfrac{\mathbf{D}_n^\dagger}{|\mathbf{D}_n|}\right)\right|}. \tag{30}$$

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

$$\mathbf{T}_{(n,l)} = \frac{\dfrac{[\mathbf{D}_{l-1}^\dagger(\mathbf{P}_{l-1}^\dagger\mathbf{D}_l^\dagger)\ldots(\mathbf{P}_{n-1}^\dagger\mathbf{D}_n^\dagger)]^\dagger}{|\mathbf{D}_{l-1}|\,|\mathbf{D}_l|\,|\mathbf{D}_n|}}{\dfrac{|\mathbf{D}_{l-1}^\dagger(\mathbf{P}_{l-1}^\dagger\mathbf{D}_l^\dagger)\ldots(\mathbf{P}_{n-1}^\dagger\mathbf{D}_n^\dagger)|}{(|\mathbf{D}_{l-1}|\,|\mathbf{D}_l|\,|\mathbf{D}_n|)^2}}. \tag{31}$$

The determinants then all cancel (the determinant of a propagation matrix is 1) yielding a very simple and direct way to go from the individual layer matrices to the back matrices,

$$\mathbf{T}_{(n,l)} = [\mathbf{D}_{l-1}^\dagger(\mathbf{P}_{l-1}^\dagger\mathbf{D}_l^\dagger)\ldots(\mathbf{P}_{n-1}^\dagger\mathbf{D}_n^\dagger)]^\dagger. \tag{32}$$

This expression has two advantages. First, and most important, it allows us to build up the back matrices using successive left multiplications of **PD** matrix pairs, enabling the use of the fast approximate algorithm discussed in Subsection 3.B to find the k derivatives of Eq. (32). Second, everything on the right hand side of Eq. (32) has already been computed in finding the front matrices, $\mathbf{T}_{(l,0)}$. 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 GD:

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

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

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

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

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

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

$$\begin{aligned}\nabla\phi'(k) = \frac{1}{|\Gamma|^2}\Bigg[&\Im[\nabla\Gamma']\Re[\Gamma] - \Re[\nabla\Gamma']\Im[\Gamma] \\ &- \nabla\phi(\Re[\Gamma']\Re[\Gamma] - \Im[\Gamma']\Im[\Gamma]) + \frac{\phi'\nabla|\Gamma|^2}{2}\Bigg].\end{aligned} \tag{39}$$

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 $\mathbf{P}_l$ and $\mathbf{D}_l$ and their derivatives with respect to k.
2. Iteratively compute front transfer matrices $\mathbf{T}_{(l,0)}$ and their derivatives for $n - 1 > l > 1$, as demonstrated in Section 3 and Ref. 11.
3. Compute back transfer matrices $\mathbf{T}_{(n,l)}$ and derivatives by repeating Step 2 for the reversed stack and then taking the Hermitian transpose, as shown in Eq. (32) and explained in Subsection 5.C.
4. Loop through all layers, computing matrix gradient elements of $\nabla\mathbf{T}$ and $\nabla\mathbf{T}'$ as per the exact expression Eq. (21) or the approximate expression Eq. (25).
5. Translate matrix gradients into reflectivity and GD 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 GD 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 Figs. 2 and 3, respectively. The gradient of group delay at 800 nm is shown with and without the constant coupling approximation in Fig. 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 GD), 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 GD 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.

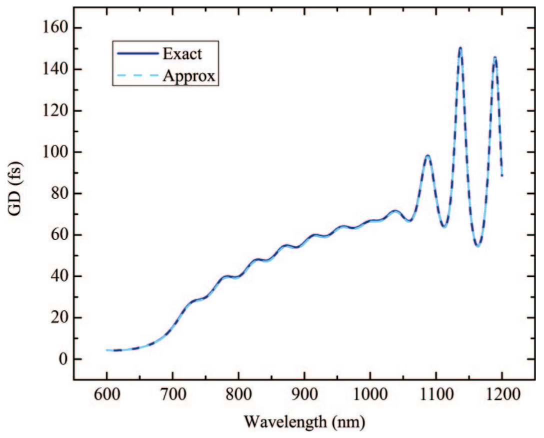

Fig. 2. (Color online) Spectral group delay of an example chirped mirror. A portion of the response past the high reflectivity region (wavelengths greater than approximately 1050 nm) is shown to demonstrate that the approximation holds even when the group delay is rapidly varying.

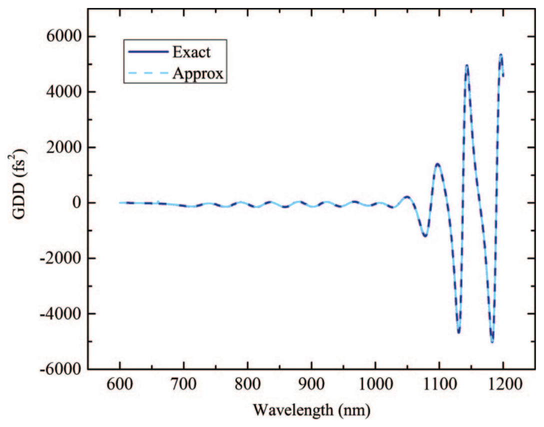

Fig. 3. (Color online) Spectral group delay dispersion of the chirped mirror shown in Fig. 2.

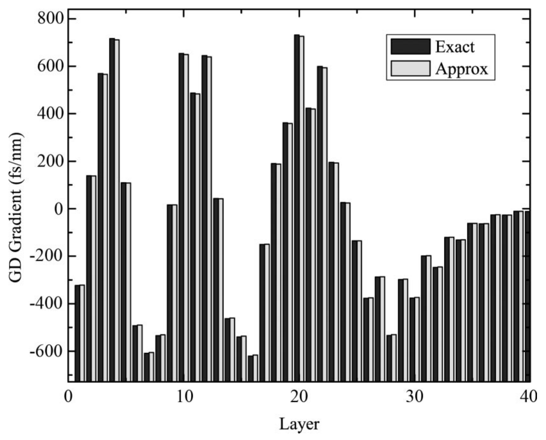

Fig. 4. Exact and approximate gradient of the group delay of the filter shown in Fig. 2 at 800 nm. The gradient approaches zero for layers past the 40th as virtually all of the light is reflected by that point.

9. MATLAB and C Code

To simplify the 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 Ref. 11 to the $O[n]$ computation of group delay gradients and have shown how dispersion optimization can be done by 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 inherent in finite differences and saves the user from being concerned with issues of optimal grid spacing or phase unwrapping.

The authors thank Christian Jirauschek for helpful discussions. This work was supported in part by National Science Foundation grant ECS-0501478 and Defense Advanced Research Projects Agency 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).
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. Quantum Electron. **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,** 5163–5169 (2005).
7. A. V. Tikhonravov, "Some theoretical aspects 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 *Optical Interference Coatings Topical Meeting* (Optical Society of America, 2004), p. ThA6.
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, 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,** 10888–10894 (2005).
14. J. Kong, *Electromagnetic Theory* (EMW, 2001).
15. A. A. Tovar and L. W. Casperson, "Generalized reverse theorems for multipass applications in matrix optics," J. Opt. Soc. Am. A **11,** 2633–2642 (1994).