Recent Progress Toward Real-Time Measurement of Ultrashort Laser Pulses

Daniel J. Kane

(Invited Paper)

***Abstract*—Frequency-resolved optical gating (FROG) is a technique that produces a spectrogram of an ultrashort laser pulse optically. While a great deal of information about the pulse can be gleaned from its FROG trace, often it is desirable to obtain all of the pulse information immediately, in real time. Quantitative information about the pulse is not readily obtainable from its spectrogram without the use of a two-dimensional phase retrieval algorithm. While current algorithms are quite robust, retrieval of all the pulse information can be slow. In this paper, I describe a recently developed FROG trace inversion algorithm called Principal Component Generalized Projects that is fast, robust, and can invert FROG traces in real time. A femtosecond oscilloscope based on second-harmonic generation FROG is also described that uses this new algorithm to rapidly (up to 2.3 Hz) and continuously display the intensity and phase of ultrashort laser pulses.**

***Index Terms*— Phase retrieval, pulse measurement, ultrafast lasers, ultrashort pulses.**

I. Introduction

FREQUENCY-RESOLVED optical gating (FROG) is used to characterize ultrashort laser pulses [1]–[12]. It optically obtains a spectrogram of an input pulse by interacting one or more pulses in a nonlinear medium to form a gate that interacts with the input pulse. The interaction forms a signal pulse which is spectrally resolved and recorded as a function of delay between the input pulse and the gate. The spectrogram (FROG trace) is a plot of signal intensity versus frequency and time. The target information, the temporal and spectral profiles of the input pulse (intensity and phase), can be obtained from the FROG trace using two-dimensional (2-D) phase-retrieval methods [4], [13], [14].

FROG is experimentally simple and data acquisition can be rapid: less than 70 ms using a video camera and frame grabber. The resulting spectrogram provides immediate qualitative information about the pulse. Quantitative pulse characteristics require up to a few minutes to obtain (depending on the required resolution) because of the iterative nature of the phase-retrieval calculation [4], [15]–[18]. Thus, FROG's usefulness as a real-time diagnostic for ultrashort laser pulses depends on the speed and robustness of the phase-retrieval algorithm used. In this paper, I discuss the development of a new algorithm to obtain quantitative pulse characteristics, in real time, from experimental FROG traces. The paper begins with a brief summary of inversion algorithm development. The discussion continues with the description of a new inversion algorithm, called the Principal Component Generalized Projections Algorithm (PCGPA) [17], [18], that is very fast for some common FROG geometries. Later, I combine the PCGPA with data acquisition in a multishot second-harmonic generation (SHG) FROG [19], [20] device to develop a femtosecond oscilloscope that demonstrates real-time pulse measurement, displaying the intensity and phase of the extracted pulse at rates up to 2.3 Hz [18].

Manuscript received October 6, 1998; revised December 23, 1998. The work of D. J. Kane was supported by the National Science Foundation under Grant III-9 361 715 and Grant DMI-9 801 116.

The author is with Southwest Sciences, Inc., Santa Fe, NM 87505 USA.

Publisher Item Identifier S 0018-9197(99)02548-8.

II. FROG Inversion Algorithms

The first step in all inversion algorithms is to construct a spectrogram mathematically that mimics a physical FROG device (see Fig. 1). An input pulse can be represented by the equation

$$E(t) = \mathrm{Re}\left\lfloor \sqrt{I(t)} \exp\left(i\omega_0 t - i\varphi(t)\right)\right\rfloor \tag{1}$$

where $I(t)$ and $\varphi(t)$ are the time-dependent intensity and phase, respectively, and ω_0 is the carrier frequency. Upon entering the FROG device, the pulse is split into two identical pulses via a beam splitter. The identical pulses are combined in a nonlinear material producing a signal with the mathematical form

$$E_{\mathrm{sig}}(t,\tau) = E(t)\Gamma[E(t-\tau)] \tag{2}$$

where $E(t)$ is referred to as the *probe*, and Γ is the gate function that converts the pulse into the gate, which depends on the nonlinear interaction used. For the FROG device depicted in Fig. 1, $\Gamma[E(t-\tau)] = E(t-\tau)$, because an SHG crystal is used and the second harmonic is collected. Another nonlinear interaction commonly used in FROG devices is the optical Kerr effect in the polarization-gate (PG) geometry [2], [3]. In that case, $\Gamma[E(t-\tau)] = |E(t-\tau)|^2$.

A spectrometer spectrally resolves the signal; that is mimicked in the algorithm via a Fourier transformation into the frequency domain. A detector, such as a CCD array, obtains the FROG trace by recording spectral intensity of the signal at each time delay. These data can be represented as the magnitude squared of the Fourier transform of $E_{\mathrm{sig}}(t-\tau)$

$$I_{\mathrm{FROG}}(\omega,\tau) = \left| \int_{-\infty}^{\infty} E(t)\Gamma[E(t-\tau)] \exp\left(-i\omega t\right) dt \right|^2. \tag{3}$$

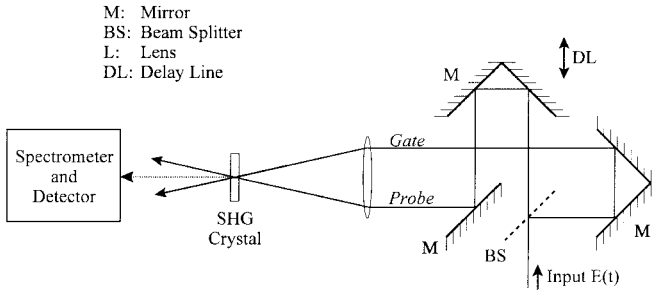


Fig. 1. Schematic of an SHG FROG device. A beamsplitter splits the input $E(t)$ into probe and gate beams. The two beams are focused into an SHG crystal. The spectrum of the second harmonic is collected as a function of delay.

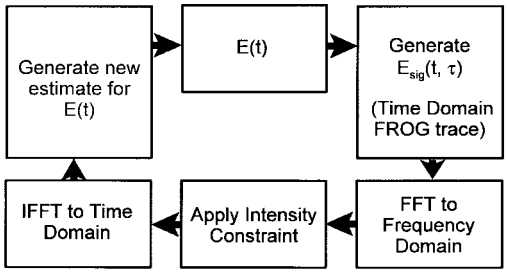


Fig. 2. All FROG trace inversion algorithms work by iterating between two constraints related by a transformation with an inverse. Paramount to the algorithm's performance is how it obtains the estimate of $E(t)$ for the next iteration.

$I_{\mathrm{FROG}}(\omega,\tau)$ is a real quantity; therefore, it has no direct phase information. The goal of the FROG inversion algorithm is to determine the phase by solving the equation

$$\sqrt{I_{\mathrm{FROG}}(\omega,\tau)}\phi(\omega,\tau)=\int_{-\infty}^{\infty}E(t)\Gamma[E(t-\tau)]\exp(-i\omega t)dt \tag{4}$$

for $\phi(\omega,\tau)$ which is a complex function of unity magnitude.

Thus, (2) and (3) define the two constraints common to all FROG algorithms that must be satisfied [4], [21]. Equation (2) is called the physical constraint and is used in the FROG algorithms both to obtain the next guess for $E(t)$ and to construct the new signal field. It is applied in the time domain. The result from (3), the FROG trace, is the intensity constraint which is applied in the frequency domain (4). The goal of the algorithm is to minimize the difference between the measured FROG trace and the FROG trace calculated from the current pulse $E(t)$ [see (5)] [4].

Fig. 2 shows the general form of FROG trace inversion algorithms. An initial guess is provided for $E(t)$ to get the algorithm started [4]. $E_{\mathrm{calc}}(t,\tau)$, a guess for $E_{\mathrm{sig}}(t,\tau)$, is calculated and Fourier transformed into $\sqrt{I_{\mathrm{calc}}(\omega,\tau)}\phi_{\mathrm{calc}}(\omega,\tau)$. $\sqrt{I_{\mathrm{cal}}(\omega,\tau)}$ is replaced by the square root of the measured FROG trace, $\sqrt{I_{\mathrm{FROG}}(\omega,\tau)}$. The next guess for $E(t)$ is then calculated from $\sqrt{I_{\mathrm{FROG}}(\omega,\tau)}\phi_{\mathrm{calc}}(\omega,\tau)$. Consequently, the phase-retrieval problem looks much like a recursive equation with N^2 variables (where N is the number of time points and frequency points).

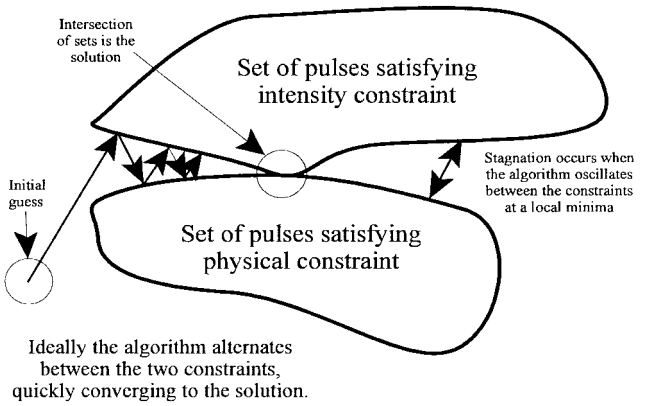


Fig. 3. Figure showing the ideal operation of a FROG inversion algorithm.

Once the new estimate for the $E(t)$ is obtained, a new spectrogram is constructed. The process is repeated (see Fig. 2) until the spectrogram error ϵ_{TF} (equivalent to the FROG trace error) reaches an acceptable minimum

$$\epsilon_{\mathrm{FROG}}\equiv\left[\frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}[I_{\mathrm{CALC}}(\omega_i,\tau_j)-I_{\mathrm{FROG}}(\omega_i,\tau_j)]^2\right]^{1/2} \tag{5}$$

where ϵ_{FROG} represents the rms error per element of the spectrogram, $I_{\mathrm{CALC}}(\omega_i,\tau_j)$ is the current iteration of the spectrogram, $I_{\mathrm{FROG}}(\omega_i,\tau_j)$ is the measured spectrogram, and ω_i and τ_j are the ith frequency and jth delay in the frequency and delay vectors, respectively [4].

The various FROG algorithms differ in how $E(t)$ is calculated from $\sqrt{I_{\mathrm{FROG}}(\omega,\tau)}\phi_{\mathrm{cal}}(\omega,\tau)$ [4], [15]–[18], [21]. If convergence time was of no concern, the first try at an algorithm might be a simulated annealing algorithm where the next guess for $E(t)$ is obtained randomly from the previous guess for $E(t)$. If only we had forever! A better approach is to produce the next guess from the previous guess in a systematic way, but how?

Ideally, after each iteration of the algorithm, a slightly better estimate for the phase of the spectrogram is obtained until convergence (Fig. 3). The original FROG inversion algorithm integrated the *time domain* FROG trace, $E_{\mathrm{sig}}(t,\tau)$, with respect to τ, the time delay, to obtain subsequent guesses for $E(t)$ (so-called "vanilla" or "basic" algorithm) [4], [15], [21]. The integration effectively reduces the gate function to a constant, yielding $\beta E(t)$, where β is the integration constant and $E(t)$ is the next guess for the electric field. While fast, this algorithm stagnates easily and fails to invert spectrograms of double pulses [4], [15], [21]. In an attempt to overcome stagnation problems and increase robustness, the vanilla algorithm was used to provide an initial guess to a brute force minimization of the rms difference between the retrieved FROG trace and the experimental FROG trace [15]. While this method is robust, converging in most cases, it is very slow.

A major advance in the FROG inversion algorithm came with the development of the generalized projections algorithm [16], [19], [22]–[24]. First, it virtually guarantees that the error always decreases for each iteration [16], [24]. Second, the

generalized projections algorithm is very robust [16], [17], [21]. Third, it is reasonably fast, being much faster than brute force minimization [16]. Last, it converges well even in the presence of noise. A complete discussion, along with derivatives used in the minimization, appears in a review article by Trebino *et al*. [21].

Like the previous algorithms, generalized projections works by alternating between two (or more) sets, S_1 and S_2 [21]. (For FROG trace inversion, S_1 is the set of all $E_{\text{sig}}(t,\tau)$'s satisfying the nonlinear material response, and S_2 is the set of all complex functions with a magnitude $I_{\text{FROG}}(\omega,\tau)$.) Unlike the previous algorithms, generalized projects finds the next guess by insuring that the distance between S_1 and S_2 is minimized for each iteration of the algorithm. DeLong *et al*. developed a generalized projections algorithm for FROG by using a minimization algorithm to find the $E(t)$ that minimizes the Euclidian distance between the signal field constructed from $E(t)$ and the signal field that satisfies the intensity constraint $E'_{\text{sig}}(t,\tau)=\sqrt{I_{\text{FROG}}(\omega,\tau)}\phi_{\text{calc}}(\omega,\tau)$. That is, the following equation:

$$Z=\sum_{t,\tau=1}^{N}\left|E'_{\text{sig}}(t,\tau)-E(t)\Gamma[E(t-\tau)]\right|^2 \tag{6}$$

is minimized with respect to $E(t)$ to obtain the next estimate of $E(t)$, and $\Gamma[\ ]$ is the function that converts $E(t)$ into the gate function [16], [19], [21].

The algorithm developed by DeLong *et al*. directly follows the definition of generalized projections [22]–[24]. The implementation is very powerful; it can be used for any FROG geometry and can include material response [25], but it is too slow for real-time inversion of FROG traces [18]. For common FROG geometries, such as SHG and PG, a new generalized projections algorithm, called Principal Component Generalized Projections (PCGP) [17], has recently been developed that does not require a minimization step, increasing the iteration rate by nearly a factor of two [18]. Because the PCGPA code is compact, it easily fits into inexpensive digital signal processing boards, allowing simultaneous data acquisition and FROG trace inversion. Using such a scheme, this new algorithm has been used to invert FROG traces in real time and has been demonstrated in a femtosecond oscilloscope that can continuously and indefinitely characterize ultrashort laser pulses in real time [18]. Sections III–V discuss the derivation of this new algorithm.

III. Principal Component Generalized Projections Algorithm

In FROG, there is always an assumed relationship between the probe and the gate. However, there is another, less common, but more general case where the gate is entirely independent of the probe. This has been called TREEFROG (Twin Retrieval of Excitation Electric fields FROG) by the more acronymous among us [26]. I shall refer to this general case as blind-FROG because it makes no *a priori* assumptions about the relationships between the probe and the gate [17]. At first, this may seem obtuse, but it allows the development of a generalized projections algorithm that does not require a minimization step, ultimately speeding algorithm execution and simplifying programming. (This becomes important later when the inversion algorithm is placed entirely on a digital signal processing board.)

Thus, the blind-FROG spectrogram, $I_{\text{FROG}}(\omega,\tau)$, is

$$I_{\text{FROG}}(\omega,\tau)=\left|\int_{-\infty}^{\infty}E(t)G(t-\tau)\exp(-i\omega t)\,dt\right|^2 \tag{7}$$

where the gate is represented as $G(t-\tau)$ and the probe as $E(t)$. The function that produces $G(t-\tau),\Gamma[\ ]$, is not required. A single time slice of the blind-FROG trace is the intensity spectrum of the product of these two functions where the gate is delayed relative to the probe by τ. The complete spectrogram is obtained when the gate is scanned in time across the probe, $E(t)$.

Equation (1) is the magnitude squared of the Fourier transform of the product $E(t)G(t-\tau)$ with respect to t. Virtually all practical data collection methods rely on discretizing τ and t. Suppose $E(t)$ and $G(t)$ are sampled at given values of t with a constant spacing of Δt. Then $E(t)$ and $G(t)$ can be thought of as vectors of length N whose elements sample E and G at discrete times

$$\begin{aligned}E_{\text{Probe}}=\Bigg[&E\left(-\frac{N}{2}\Delta t\right),E\left(-\left(\frac{N}{2}-1\right)\Delta t\right),\\&E\left(-\left(\frac{N}{2}-2\right)\Delta t\right),\cdots,E\left(\left(\frac{N}{2}-1\right)\Delta t\right)\Bigg]\\E_{\text{Gate}}=\Bigg[&G\left(-\frac{N}{2}\Delta t\right),G\left(-\left(\frac{N}{2}-1\right)\Delta t\right),\\&G\left(-\left(\frac{N}{2}-2\right)\Delta t\right),\cdots,G\left(\left(\frac{N}{2}-1\right)\Delta t\right)\Bigg].\end{aligned} \tag{8}$$

For simplicity, the vectors are written as

$$\begin{aligned}E_{\text{Probe}}&=[E_1,E_2,E_3,\cdots,E_N]\\E_{\text{Gate}}&=[G_1,G_2,G_3,\cdots,G_N].\end{aligned} \tag{9}$$

The outer product of E_{Probe} and E_{Gate} is (Fig. 4)

$$\begin{bmatrix}E_1G_1&E_1G_2&E_1G_3&E_1G_4&\cdots&E_1G_N\\E_2G_1&E_2G_2&E_2G_3&E_2G_4&\cdots&E_2G_N\\E_3G_1&E_3G_2&E_3G_3&E_3G_4&\cdots&E_3G_N\\E_4G_1&E_4G_2&E_4G_3&E_4G_4&\cdots&E_4G_N\\\cdot&\cdot&\cdot&\cdot&\cdots&\cdot\\\cdot&\cdot&\cdot&\cdot&\cdots&\cdot\\\cdot&\cdot&\cdot&\cdot&\cdots&\cdot\\E_NG_1&E_NG_2&E_NG_3&E_NG_4&\cdots&E_NG_N\end{bmatrix}. \tag{10}$$

This matrix will be referred to as the *outer product form*.

The outer product contains all of the points required to construct the time domain FROG trace because it contains all of the interactions between the pulse and gate for the discrete delay times. Consequently, a one-to-one mapping of the elements of the outer product can transform the outer product into the time domain of the FROG trace. This is the key to the PCGP algorithm. Because the mapping is one-to-one, it is invertible; transformations can be made from the outer product form to the time domain FROG trace and *vice*

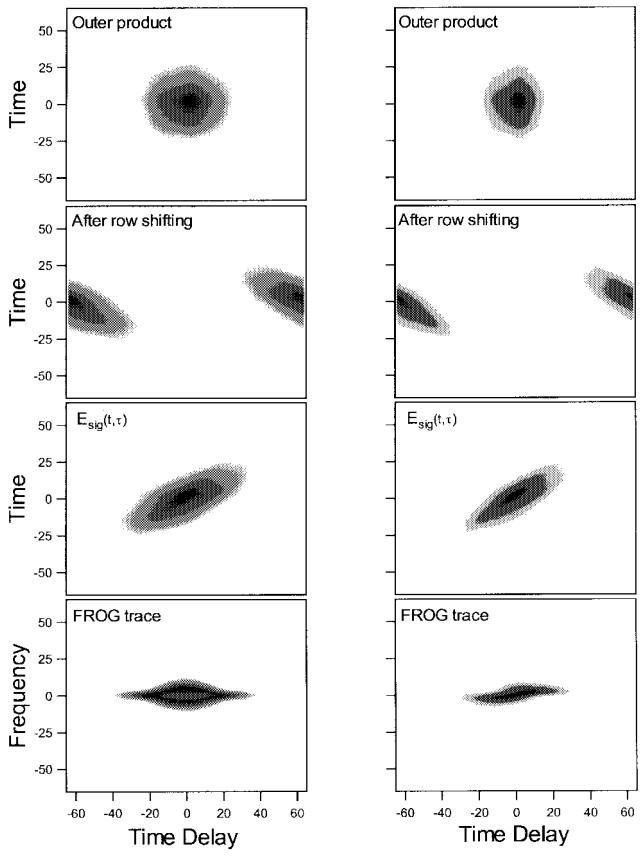


Fig. 4. The different steps in the PCGPA. (a) SHG and (b) PG are shown. The top image plots show the outer product. The next image results from the row rotation depicted in (11). By rearranging the columns, the correctly oriented time-domain FROG trace can be constructed. Fourier transforming the columns produces the FROG traces shown in the bottom image plots.

versa. This transformation can be accomplished by rotating the elements of the rows in the outer product to the left by the row number minus one. Applying this transformation, we obtain (11), shown at the bottom of the page. The $\tau = 0$ column is the first column, where τ is the time delay in increments of Δt, a point-by-point multiplication of the probe by the gate with no time shift between them. The next column is the $\tau = -1$ column where the gate is delayed relative to the probe by one resolution element, Δt. The gate appears to be shifted "up" by one resolution element with the first element wrapped around to the other end of the vector. Column manipulation places the most negative τ on the left and the most positive on the right. Thus, (11) is the *time domain of the spectrogram formed by the multiplication of the probe and gate functions;* a discrete version of the product $E_{\mathrm{Probe}}(t)E_{\mathrm{Gate}}(t-\tau)$. The columns are constant in τ (delay) while the rows are constant in t (time). This gives exactly the same result as calculating the time-domain FROG trace directly by shifting the gate in time and multiplying the shifted gate by the probe. All I am doing is insuring is that there is a reversible way to move between the outer product form and the time-domain FROG trace. By Fourier transforming each column, the Fourier transform of $E_{\mathrm{Probe}}(t)E_{\mathrm{Gate}}(t-\tau)$ is obtained as a function of t. The final step of taking the magnitude of the complex result produces the FROG trace.

IV. PCGPA Inversion

It is easy to imagine an infinite number of complex *images* that have the same magnitude as the spectrogram we wish to invert; however, there is only one image that can be formed by the outer product of a *single pair of nontrivial vectors* that has the same magnitude as the spectrogram to be inverted. In order to find the proper vector pair, the phase of the spectrogram must be determined using a 2-D phase-retrieval algorithm.

Like all FROG trace inversion algorithms, the PCGPA is started using Gaussian pulses with random phase for the initial guess for $E(t)$. The initial gate pulse is derived from $E(t)$ according to the FROG geometry used. A spectrogram is constructed and its magnitude is replaced by the square root of the magnitude of the experimentally obtained spectrogram. The result is converted to the time-domain spectrogram (11) using an inverse Fourier transform by column (see Fig. 4). Next, the time-domain spectrogram is converted to the outer product form (10) by reversing the steps used to construct the time domain spectrogram. If the intensity and phase of the spectrogram are correct, this matrix (the outer product form matrix) is a true outer product and has a rank of one. That is, it would have one and *only one* nonzero eigenvalue and one *right* eigenvector and one *left* eigenvector. The *right* eigenvector, the probe, spans the range of the outer product matrix. The complex conjugate of the eigenvector of the transpose of the outer product matrix (*left* eigenvector) is the gate [24].

The outer product form matrix produced by the initial guess, however, is not rank one and has several eigenvectors. It will probably have (for an $N \times N$ FROG trace) N right eigenvectors and N left eigenvectors (eigenvectors of the transpose): instead of describing of a single line in N space, the matrix represents an ellipsoid in N space. The best

$$\begin{bmatrix} E_1G_1 & E_1G_2 & E_1G_3 & \cdots & E_1G_{N-2} & E_1G_{N-1} & E_1G_N \\ E_2G_2 & E_2G_3 & E_2G_4 & \cdots & E_2G_{N-1} & E_2G_N & E_2G_1 \\ E_3G_3 & E_3G_4 & E_3G_5 & \cdots & E_3G_N & E_3G_1 & E_3G_2 \\ E_4G_4 & E_4G_5 & E_4G_6 & \cdots & E_3G_1 & E_4G_2 & E_4G_3 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ E_NG_N & E_NG_1 & E_NG_2 & \cdots & E_NG_{N-3} & E_NG_{N-2} & E_NG_{N-1} \end{bmatrix}$$
$$\tau = 0 \quad \tau = -\Delta t \quad \tau = -2\Delta t \quad \cdots \quad \tau = 3\Delta t \quad \tau = 2\Delta t \quad \tau = \Delta t \tag{11}$$

next guess may actually be a superposition of two or more different but linearly independent eigenvectors, requiring an optimization such as minimization of the FROG trace error to find the correct superposition.

Fortunately, minimization is not required. Suppose we decompose the outer product form matrix O into three matrices such that

$$O = U \times W \times V^T \tag{12}$$

where U and V^T are orthogonal square matrices and W is a square diagonal matrix. Thus, the matrix O, the outer product form, is decomposed into a superposition of outer products between "probe" vectors (columns of U) and "gate" vectors (rows of V^T). The diagonal values in W (the only nonzero elements of W) determine the relative weights of each outer product and, therefore, how much each outer product contributes to matrix O. If we keep the outer product pair with the largest weighting factor, or *principal component*, for the next iteration of the algorithm, we minimize the function

$$\epsilon^2 = \sum_{i,j=1}^{N} |E^{i,j}_{\text{Outer}} - E^{i}_{\text{Probe}} E^{j}_{\text{Gate}}|^2 \tag{13}$$

where E_{Outer} is the outer product form matrix, E_{Probe} is the probe vector, E_{Gate} is the gate vector, and ϵ is the error [27]. (This is the definition of a projection and is similar to, although not identical to, the projection found in the generalized projections algorithm developed by DeLong *et al*. [16], [19], [21].)

How can all this be accomplished? One elegant, but computationally intensive, means to find the principal vector pair is to use a singular value decomposition (SVD) to decompose O into U, W, and V directly [24], [27]. This approach is convenient because many commercially available mathematical libraries contain routines to compute SVD's. Another way to find the principal vector pair with much less computation than an SVD is to reduce the SVD step to simple low-overhead and fast matrix-vector multiples [28]. For real-time applications, this is the best approach [18].

Rather than finding the eigenvectors of O, the outer product form matrix, and constructing an orthonormal basis from these vectors, an SVD finds the eigenvectors of OO^T (columns of U) and O^TO (columns of V) which are orthonormal [24], [27]. If the columns of U are written as P_i and the columns of V are written as G_i, then they satisfy the equations

$$\begin{aligned} OO^T P_i &= \lambda_i P_i \\ O^T O G_i &= \lambda_i G_i \end{aligned} \tag{14}$$

where λ_i's are the eigenvalues, or "weights," and the superscript T is the transpose operator. O may be constructed by

$$O = \sum_{i=1}^{N} \sqrt{\lambda_i} P_i G_i^T \tag{15}$$

where λ_i, P_i, and G_i are provided by the SVD, but we only need the P_i and G_i corresponding to the largest $|\lambda_i|$, or the principal eigenvectors. Suppose we multiply an arbitrary nonzero vector x_0 by OO^T. Then

$$OO^T x_0 = \sum_{i=1}^{N} \kappa_i \lambda_i P_i \tag{16}$$

where P_i is the eigenvector of OO^T, λ_i the eigenvalues, and κ_i a set of constants. OO^T can be thought of as an operator that maps x_0 onto a superposition of eigenvectors. The process can be repeated resulting in $OO^T \kappa_i \lambda_i P_i = \kappa_i \lambda_i^2 P_i$. Multiplying by $(OO^T)^{p-1}$ gives

$$(OO^T)^p x_0 = \sum_{i=1}^{N} \kappa_i \lambda_i^p P_i. \tag{17}$$

As p becomes large, the largest eigenvalue λ_l dominates the sum so that $(OO^T)^p x_0 \sim \kappa_l \lambda_l^p P_l$. This method is called the *power method* [28]. After a few iterations, a very close approximation to the principal eigenvector (the eigenvector with the greatest eigenvalue) is obtained. Consequently, the next guess for the pulse can be obtained by multiplying the previous guess for the pulse by OO^T. The next guess for the gate can be obtained by multiplying the previous guess for the gate by O^TO. (For polarization-gate FROG, the absolute value of the result for the gate is taken.) While better approximations for the eigenvectors may be obtained by using these operators several times per iteration, once per iteration is adequate in practice [18].

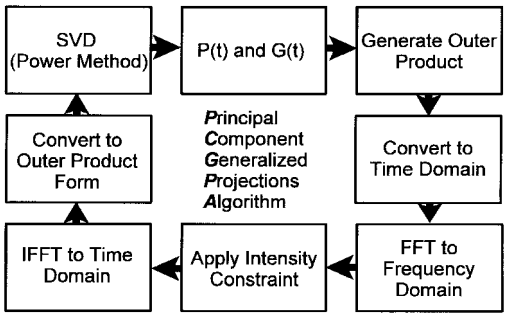


Fig. 5. Schematic of the PCGP algorithm. Transformation from the outer product to the time domain FROG trace (and *vice versa*) may be accomplished via simple permutations (rotations) of each row.

Practically, the power method implementation of the PCGP algorithm (Fig. 5) is very fast and quite robust. Indeed, the power method implementation can loop at nearly 20 iterations/s on a 60 MFLOPS digital signal processor or greater than 30 iterations/s on a 233-MHz Pentium II. Good approximations for the pulse usually occur in about 40 iterations [18].

V. Conversion of PCGPA to a FROG Inversion Algorithm

The PCGPA, as discussed above, is a blind-FROG algorithm. That is, the probe and the gate are completely independent. The only nonlinear interaction assumed is the multiplication of the probe by a gate. How the gate is constructed is of no concern. As a result, some ambiguities can occur. Even though these ambiguities are usually minor,

TABLE I
COMPARISON OF SHG FROG INVERSION ALGORITHM

Algorithm Type	Random Noise Test: Percent Convergence	Random Chirp Test: Percent Convergence	Multipulse Test: Percent Convergence
SHG Vanilla	15 (4)	45 (32)	64 (3)
Femtosoft SHG FROG	56 (48)	80 (52)	92 (0)
Fast SHG PCGPA	78 (77)	79 (70)	95 (10)

Results of a test designed to determine robustness of FROG inversion algorithms. All of the test groups are synthetic. Models for the test groups are discussed in Kane [18[. The percentages given are for a percent of pulses retrieved with an rms FROG trace error of less than 2×10^{-3} in 100 iterations. This was deemed to be the lowest experimental error that can, in practice, be achieved and more aptly defines the usefulness of the algorithm. Percentages given in parentheses are for strict convergence with an rms error of less than 10^{-4} in 100 iterations. The ultimate convergence rates (allowing the algorithm to continue past 100 iterations until stagnation) for the Femtosoft SHG FROG inversion algorithm were 60%, 80%, and 32% for the random noise test, the random chirp test, and the multipulse test, respectively. The ultimate convergence rates of the SHG FROG PCPG algorithm was not determined.

they can produce erroneous results if ignored (see Appendix A) [17], [26]. Spectral constraints can facilitate inversion of FROG spectrograms using the PCGPA [17], [26]. This method has been used extensively to invert experimental FROG traces and blind-FROG traces [29], [30]. Often, however, a spectrum is not available; consequently, we would rather not be required to obtain a spectrum of the pulse in addition to its spectrogram.

The conversion of PCGPA to a FROG algorithm may be accomplished by summing the outer product of the probe and the gate with the outer product of the probe constructed from the gate and the gate constructed from the probe [18]. How the gate is constructed from the probe and *vice versa* is determined from the nonlinear interaction. In the case of SHG FROG, for example, the probe is equal to the gate; thus, the outer product becomes

$$O_k^{ij} = \mathrm{probe}_k^i \mathrm{gate}_k^j + \mathrm{gate}_k^i \mathrm{probe}_i^j \tag{18}$$

forming the FROG trace from the sum of two outer products. Because only the principal outer product pair is used for the next estimate of the electric field, the two outer products are forced to be equal. The only way the outer products can be equal is if probe $=$ gate.

This type of FROG algorithm works very well for SHG FROG (see Appendix B). Table I compares the PCGP-based SHG FROG algorithm to the commercially available Femtosoft SHG FROG program [19], [21], and to the "Basic FROG" or "Vanilla" algorithm [4], [21]. The three algorithms were tested to failure using three synthetically constructed test sets. From the test results, it can be determined that the generalized projections-based algorithms are clearly superior to the "Vanilla" algorithm. In the first test, the random noise test, which determines the overall robustness of the algorithms, the SHG PCGP algorithm performed best. In the other two tests, the PCGP SHG algorithm compares favorably to the Femtosoft algorithm [18].

SHG FROG is a special case, however; (18) is valid only for SHG FROG and must be modified for other FROG geometries. As defined in (2), Γ is the function that produces the gate from the probe $E(t)$; its inverse, denoted Γ^{-1}, produces the probe from the gate. Thus, the gate $= \Gamma^{-1}(\mathrm{probe})$ and the probe $= \Gamma^{-1}(\mathrm{gate})$. Rather than using only the outer product of E_{probe} and E_{gate} to produce the next time-domain FROG trace, the sum of the outer products of $E_{\mathrm{probe}}^i E_{\mathrm{gate}}^j$ and $\Gamma^{-1}(E_{\mathrm{gate}})^i \Gamma(E_{\mathrm{probe}})^j$ is used so that the outer product on the next iteration is given by

$$O_k^{ij} = \mathrm{probe}_k^i \mathrm{gate}_k^j + \Gamma^{-1}(\mathrm{gate}_k)^i \Gamma(\mathrm{probe}_k)^j \tag{19}$$

where O_k is the sum of the two outer products for the kth iteration.

Equation (19) allows PCGPA to be used with any FROG geometries where Γ^{-1} exists. In the PG FROG, however, the inverse of the gate function does not exist. As a result, a pseudo-inverse must be constructed from the square root of the gate intensity and the phase of the pulse. Because the square root can cause small fluctuations in the wings of the gate, producing artifacts in the next guess for the pulse, instabilities may occur in the algorithm. This can be remedied by applying the square root to only well-defined portions of the gate. Where the gate is not well defined (i.e., the intensity is near zero), the intensity (and phase) of the pulse is used. To increase the robustness of the PG algorithm, the pseudo-inverse constraint is applied on alternate iterations. The pseudo-inverse method works well for polarization-gate (PG) FROG on the synthetic test sets, converging to experimental error for 90% of the test pulses, but it has not been tested with experimental data. At this time, the pseudo-inverse method does not appear to work well for self-diffraction (SD) FROG, however.

VI. EXPERIMENTAL—THE FEMTOSECOND OSCILLOSCOPE

The development of the PCGPA can facilitate the inversion of FROG spectrograms in real time [18]. However, building a femtosecond oscilloscope requires more than just a fast inversion algorithm. The data acquisition must be completely integrated with the inversion algorithm. This is accomplished in the demonstration described here by integrating the data acquisition and the inversion engine with a home-built multishot SHG FROG device. The data acquisition and inversion engine utilizes two commercially available digital signal processing (DSP) boards (Fig. 6). This device successfully demonstrates the inversion of experimental FROG traces in real time and can display the inverted pulses (from a 64×64 FROG trace) at a rate of 1.25 Hz, or one every 0.8 s, and 2.3-Hz inversion rates were possible for a 32×32 array.

Using a zero-dispersion pulse stretcher-compressor [31] to vary the pulse dispersion independent from the Ti:sapphire oscillator allows extensive testing of the femtosecond oscilloscope. By translating the lens, dispersion in the beam can be changed enough to more than triple the pulsewidth. The femtosecond oscilloscope can easily track these changes. Also, portions of the spectrum can be blocked to shape the pulse before being sent to the FROG device (Figs. 7 and 8).

Example data obtained using the femtosecond oscilloscope are shown in Fig. 7. The FROG trace shown in Fig. 7(a) was produced by blocking a portion of the pulse spectrum at the Fourier plane of the stretcher-compressor. Also show in Fig. 7 are the retrieved pulse, the retrieved phase, and an example of

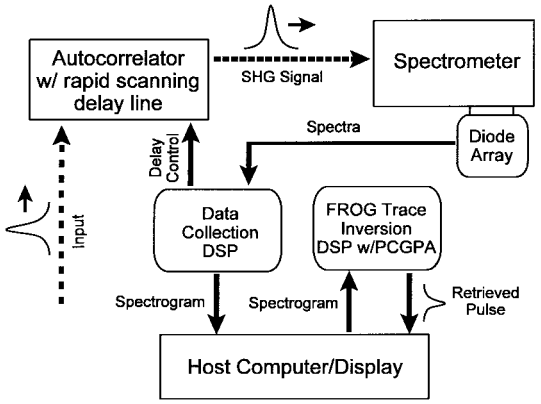


Fig. 6. Schematic of the femtosecond oscilloscope. A multishot SHG FROG device with a rapid scanning delay line acts as the front end. The detector is an EG&G 512 element diode array. The array is read by the data DSP which also controls the delay line. After each spectrum is read, the delay line is incremented by Δt until a complete spectrogram is obtained. The spectrogram is then resampled, filtered, and sent to the host computer. The host displays the spectrogram and sends it to the inversion DSP. After about 15 iterations, the pulse and gate are read by the host and displayed. This device could fully characterize pulses at a rate of 1.25 Hz for 64 × 64 FROG traces and 2.3 Hz for 32 × 32 FROG traces.

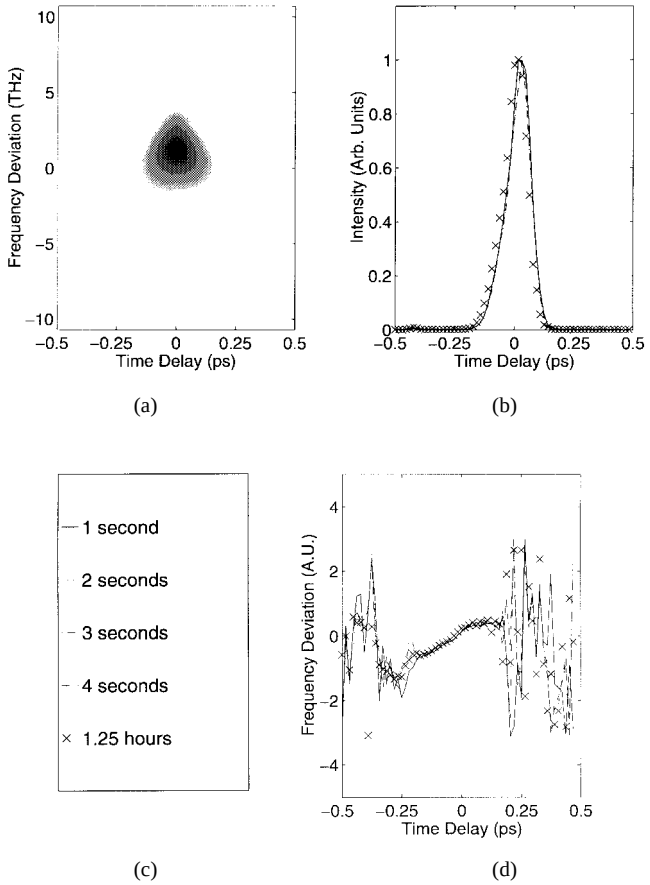


Fig. 7. Data taken using the femtosecond oscilloscope depicted in Fig. 5. (a) The FROG trace: this pulse was retrieved using the PCGPA as the inversion engine implemented on the DSP card. In only one second (20 iterations), the algorithm converged to a FROG trace error of less than 0.5% for a 64 × 64 FROG trace. Also important to the operation of the femtosecond oscilloscope is the stability of the algorithm; results did not change significantly even after thousands of iterations. (b) Pulse. (c) Algorithm/DPS processing time. (d) Pulse phase.

the performance of the DSP/PCGPA combination. After only one second of computational time, the algorithm ran for 20 iterations on the 64 × 64 FROG trace, converging to a FROG trace error of less than 0.5%.

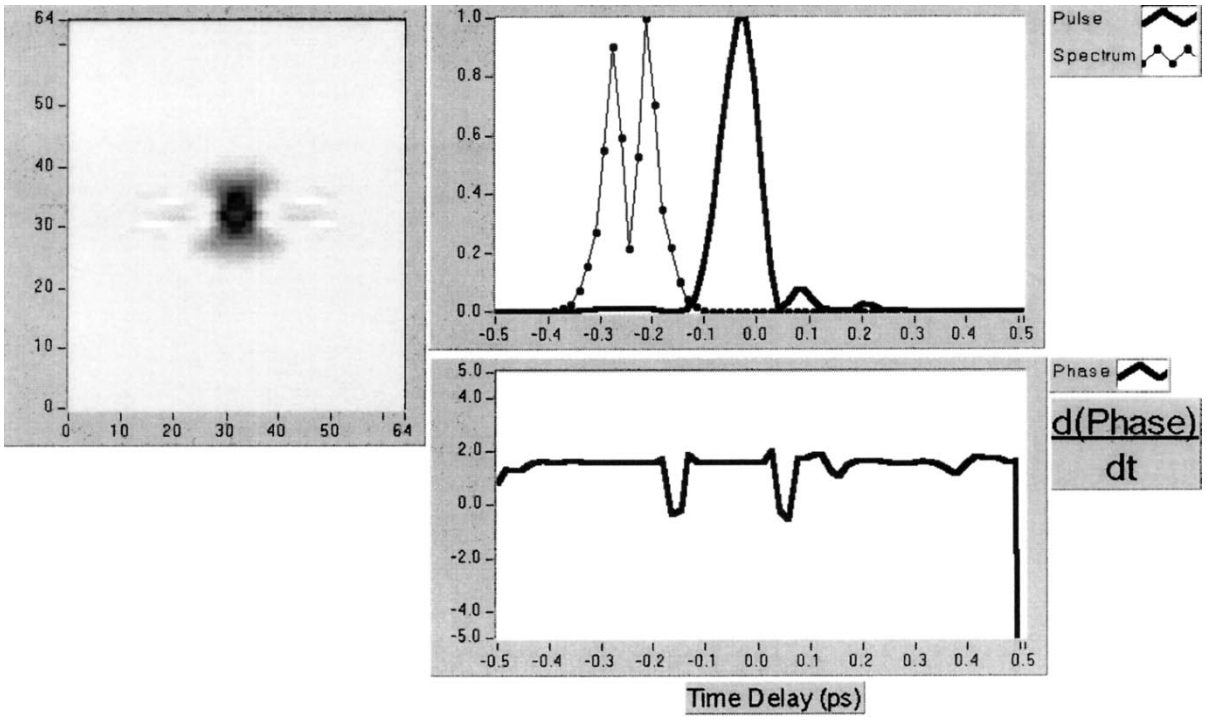
The SHG FROG device splits the input beam into two identical beams by a beam splitter [18]. One beam is sent into a manual delay line used to fine tune the delay between the two beams so that the FROG trace is centered along the time axis for proper operation of the PCGP algorithm. The other beam is sent into a fast scanning delay line based on a General Scanning LT 1000 Z (linear) scanner [18[allowing the delay to be controlled by a voltage (∼2-mm delay/V). The resulting beams are about 8 mm apart and focused by a 250-mm focal length lens into a 200-μm-thick BBO crystal. The spectrum of the second harmonic is measured via a 1/4 m spectrograph and a 512–element EG&G Reticon diode array controlled by the EG&G demonstration board. The resulting electronic signal is filtered by a SRS 560 low-noise differential amplifier before being digitized at 100 kHz (5 ms exposure) by the 16-bit A/D converters on the data collection DSP board. After the diode array is read, the translation stage is set to the next delay via a D/A on the DSP board. Sixty-four spectra are obtained for the 64 × 64 FROG trace and 32 for the 32 × 32 FROG trace.

In addition to digitizing the diode array readout, the data collection DSP board also prepares the raw data for input into the algorithm by resampling the signal vector from the 512-element diode array down to 64 pixels using a 15-element finite impulse response digital filter. The coefficients are chosen to remove all frequencies higher than Nyquist for the resampled vector. After filtering, the background from electronics offset and scattered light is subtracted.

The host computer (166 MHZ Pentium) controlled both DSP boards which are each based on a single Texas Instruments TMS320C32 floating point DSP (Fig. 6) via a host program that polls the data acquisition DSP board for a new spectrogram. When ready, the host reads the spectrogram, frees the data DSP board to read another spectrogram, and displays the spectrogram (Fig. 8). The host then reads the new pulse and gate from the inversion engine DSP board running the SHG FROG PCGPA. The new spectrogram is sent to the inversion engine board. The initial guess used by the algorithm in the inversion DSP for the new spectrogram is the pulse retrieved from the previous spectrogram. The reason behind this step is the assumption that the average pulse will not change too much over one second, allowing real-time updates to occur. This is an important part of the femtosecond oscilloscope. For small pulse changes, the algorithm will track continuously, which is usually the case when adjusting a stretcher-compressor, for example. However, for a step function change in the pulse, such as blocking the input beam momentarily, the algorithm can require 2 s to track the change (for a 64 × 64 FROG trace).

VII. Conclusions

While the PCGPA is naturally a blind-FROG algorithm, it has been successfully adapted to the inversion of FROG spectrograms by averaging the outer product of the pulse and gate with the outer product of the pulse constructed



(a) (b)

Fig. 8. Femtosecond oscilloscope display. (a) The FROG trace resulting from a wire blocking out portions of the pulse spectrum at the Fourier plane of a stretcher-compressor. (b) Both the pulse intensity and spectrum (spectral intensity of the retrieved pulse). Notice that the center portion of the spectrum is missing and ringing of the pulse is clearly visible. To prevent the variable offset in the retrieved phase from changing the plot scaling between updates, the derivative of the phase is displayed.

from the gate with the gate constructed from the pulse. Best suited for SHG FROG, this algorithm is the most robust algorithm available for the inversion of SHG FROG traces. By using a pseudo-inverse, PCGP can also be used to invert PG FROG spectrograms. While not quite as robust as other PG FROG algorithms, it is still a viable alternative when speed, simplicity, or compactness is of concern.

The main hindrance to real-time pulse measurement has been the iterative algorithm required to invert the FROG traces. Even with the fast processors available today, the computational requirements for obtaining the FROG trace, preprocessing the FROG trace, and inverting the FROG trace in real time are too great for most desktop computers. By the addition of an inexpensive but separate DSP, independent of the operating system, real-time pulse measurement can be a reality. Because of the limited memory space on these DSP boards, the compact vector-based fast PCGPA is ideal for this application.

When on a DSP board, the PCGPA using the power method can run at nearly 20 iterations/s which is more than enough to track small changes in the input pulse. For SHG FROG, the PCGPA operates about two times faster than the current generalized projections algorithm while being as robust as the commercially available compound algorithm [21]. The constructed femtosecond oscilloscope obtains SHG FROG traces, updates a spectrogram display, and provides the intensity and phase of the pulse in real time with an update of 0.8 s or 1.25 Hz for a 64×64 array (0.43 s or 2.3 Hz for a 32×32 array). Currently under development is a real time femtosecond oscilloscope based on a single-shot FROG device, a video CCD camera, and a frame grabber.

APPENDIX A
BLIND-FROG PITFALLS, EFFECTIVE AMBIGUITIES

While ambiguities are not a problem when inverting FROG traces, blind-FROG inversions can have ambiguities [17], [26]. The first set of ambiguities is the order of the vector pair. For example, one ambiguity in polarization-gate blind-FROG occurs when there are no phase distortions present in the probe; the blind-FROG trace does not change when the probe and gate are interchanged. As a result, any blind-FROG algorithm may converge with the probe and gate reversed. Minor phase distortions in the probe can prevent this from occurring.

Of greater concern are small variations in the spectrogram that can sometimes produce relatively large fluctuations in the probe and gate vectors; this usually only occurs when the probe and gate are similar in shape and duration. This results in an *effective ambiguity* in polarization-gate blind-FROG. I call this an effective ambiguity because it is not an ambiguity of the technique, but rather manifests itself only when there are noise and/or distortions on the spectrogram. Ideally, the set containing all the spectrograms that have the same intensity as the measured spectrogram and the set containing all of the spectrograms that can be formed from the outer product of two vectors intersect at only one point (Fig. 1). When this occurs, we say the solution is unique. However, when noise and distortions are present on the spectrogram, the two sets no longer intersect at a point, but rather they intersect within a region or, more likely, do not intersect at all. The topology of the two sets indicates how successful the search for a solution will be. If one or both of the sets are pointed

near the intersection point, ambiguities will not pose any concern. However, if the region near the intersection, or closest approach, is broad and flat on both sets, small changes in the error can result in relatively large changes in the probe and gate. One example is when a blind-FROG trace is produced by a Gaussian probe pulse with a FWHM of 100 fs and a Gaussian gate (the actual gate, not the gate pulse) with the same FWHM. Such a trace will differ from a blind-FROG trace produced by a Gaussian probe pulse with a FWHM of 105 fs and a Gaussian gate with a FWHM of 95 fs by an ϵ_{TF} of only 0.000 38. Subtle differences that are present can be obscured by noise. On the other hand, if the probe and gate have different shapes and very different widths, effective ambiguities do not appear to be a problem.

In FROG, as opposed to blind-FROG, because of the *a priori* knowledge of the relationship between the probe and gate, the width ambiguity is not a problem. These ambiguities may be resolved when using a blind-FROG inversion algorithm such as PCGPA by the addition of a spectral constraint on either the probe or the gate [17], [26].

Appendix B
Sample MATLAB Program Listing

```
function [gpulse, ggate] = ispecshg(spectrogram,
  gpulse, ggate, iterations)
% function [gpulse, ggate] = ispecshg(spectrogram,
% gpulse, ggate, iterations)
% ispecshg inverts a spectrogram using the fast
% version. It assumes SHG FROG.
%
% Principal Component Generalized Projections
% (PCGPA). The
% input parameters are:
% spectrogram   The spectrogram to be inverted
%
% gpulse   Vector containing the initial guess
% for the pulse
%
% ggate   Vector containing the initial guess
% for the gate
%
% iterations   Number of iterations
%
% The function returns as soon as the number of
% iterations has been completed.
%
% Make the check for iterations
%
% Initializations

N = max(size(gpulse));   %N = length of gpulse
halfN = N/2;
efrog = zeros(N,N);
% efrog = N x N matrix filled with zeros
temp = zeros(N,N);   % temporary matrix
%
% End Initializations
%
for x = 1:1:iterations
% Because the transpose of a complex quantity
% involves a complex conjugate,
% gpulse = conjugate(ggate) for SHG FROG.

  efrog = gpulse' * ggate + conj(ggate')
    *conj(gpulse);   % ''''' means
% transpose.
% Convert from outer product to frog domain
% First, rotate each row to the left by it's
% row number minus 1
% First index is the row number, second is

% the column
    for j = 2:1:N
        temp(j,1:j-1) = efrog(j,1:j-1);
        % save the left-most part
        efrog(j,1:N+1-j) = efrog(j,j:N);
        % shift the vector
        efrog(j,N+2-j:N) = temp(j,1:j-1);
        % place the left-most part on the right
    end

% switch left and right halves, ":" is an implied
% "for" loop
    temp(:,1:halfN) = efrog(:,1:halfN);
    efrog(:,1:halfN) = efrog(:,halfN+1:N);
    efrog(:,halfN+1:N) = temp(:,1:halfN);
% Now efrog is in the time domain

% Perform "FFT Shift" on each column
    temp(1:halfN,:) = efrog(1:halfN,:);
    efrog(1:halfN,:) = efrog(halfN+1:N,:);
    efrog(halfN+1:N,:) = temp(1:halfN,:);

% FFT columns (MATLAB is column major,
% C is row major)
    efrog = fft(efrog);

% Perform "FFT Shift" on each column
    temp(1:halfN,:) = efrog(1:halfN,:);
    efrog(1:halfN,:) = efrog(halfN+1:N,:);
    efrog(halfN+1:N,:) = temp(1:halfN,:);
% This loop is slow running under MATLAB, for
% faster running under
% MATLAB, vectorize.
    fo  r j = 1:1:N
        for k = 1:1:N
          temps = abs(efrog(j,k));
          if temps ~= 0
             efrog(j,k) = spectrogram(j,k)
               * (efrog(j,k)/temps);
          else
             efrog(j,k) = spectrogram(j,k);
          end
        end
    end
```

```
% Column "FFT Shift" again
      temp(1:halfN,:) = efrog(1:halfN,:);
      efrog(1:halfN,:) = efrog(halfN+1:N,:);
      efrog (halfN+1:N,:) = temp(1:halfN,:);
% IFFT columns
      efrog = ifft(efrog);

% "FFT Shift"
      temp(1:halfN,:) = efrog(1:halfN,:);
      efrog(1:halfN,:) = efrog(halfN+1:N,:);
      efrog(halfN+1:N,:) = temp(1:halfN,:);

% Convert from the frog time domain to the
% outer product
      temp(:,1:halfN) = efrog(:,1:halfN);
      efrog(:,1:halfN) = efrog(:,halfN+1:N);
      efrog(:,halfN+1:N) = temp(:,1:halfN);
      for j = 2:1:N
      temp(j,N+2-j:N) = efrog(j,N+2-j:N);
      efrog(j,j:N) = efrog(j,1:N+1-j);
      efrog(j,1:j-1) = temp(j,N+2-j:N);
   end

% Now the frog trace is in the outer product
% domain

% Find the next estimate for the pulse and gate
      npulse = ((efrog' * gpulse'));
      npulse = (efrog * npulse)';
%
      ggate = (efrog * ggate');
      ggate = (efrog' * ggate)';

% Normalize gpulse and ggate so the peak
% height is one
      gpulse = npulse/max(abs(npulse));
      ggate = ggate/max(abs(ggate));

end
return
```

MATLAB code for the inversion of SHG FROG traces. This program will run without modifications under MATLAB running on any platform. It returns the time reversed pulse. For SHG FROG, this does not matter because of the ambiguity in the direction of time. If this program is to be converted to PG FROG, then this must be kept in mind.

ACKNOWLEDGMENT

The author would like to thank D. Bomse for his helpful suggestions.

REFERENCES

[1] D. J. Kane and R. Trebino, "Characterization of arbitrary femtosecond pulses using frequency-resolved optical gating," *IEEE J. Quanttum Electron.*, vol. 29, pp. 571–579, 1993.

[2] ——, "Single-shot measurement of the intensity and phase of a femtosecond laser pulse," presented at Generation and Measurement of Ultrashort Laser Pulses, Los Angeles, CA, 1993.

[3] D. J. Kane, A. J. Taylor, R. Trebino, and K. W. DeLong, "Single-shot measurement of the intensity and phase of a femtosecond UV laser pulse using frequency-resolved optical gating," *Opt. Lett.*, vol. 19, pp. 1061–1063, 1994.

[4] R. Trebino and D. J. Kane, "Using phase retrieval to measure the intensity and phase of ultrashort laser pulses: Frequency-resolved optical gating," *J. Opt. Soc. Amer. A*, vol. 10, pp. 1101–1111, 1993.

[5] S. Backus, J. Peatross, Z. Zeek, A. Rundquist, G. Taft, M. M. Murnane, and H. C. Kapteyn, "16-fs, 1-μJ ultraviolet pulses generated by third-harmonic conversion in air," *Opt. Lett.*, vol. 21, pp. 665–667, 1996.

[6] P. R. Bolton, A. B. Bullock, C. D. Decker, M. D. Feit, A. J. P. Megofna, P. E. Young, and D. N. Fittinghoff, "Propagation of intense, ultraviolet laser pulses through metal vapor: Refraction-limited behavior for single pulses," *J. Opt. Soc. Amer. B*, vol. 13, pp. 336–346, 1996.

[7] T. S. Clement, A. J. Taylor, and D. J. Kane, "Single-shot measurement of the amplitude and phase of ultrashort laser pulses in the violet," *Opt. Lett.*, vol. 20, pp. 70–72, 1995.

[8] B. Kohler, V. V. Yakovlev, K. R. Wilson, J. Squier, K. W. DeLong, and R. Trebino, "Phase and intensity characterization of femtosecond pulses from a chirped-pulse amplifier by frequency-resolved optical gating," *Opt. Lett.*, vol. 20, pp. 483–485, 1995.

[9] A. Kwok, L. Jusinski, M. A. Krumbugel, J. N. Sweetser, D. N. Fittinghoff, and R. Trebino, "Frequency-resolved optical gating using cascaded second-order nonlinearities," *IEEE J. Select. Topics Quantum Electron.*, vol. 4, pp. 271–277, 1998.

[10] J. N. Sweetser, D. N. Fittinghoff, and R. Trebino, "Transient-grating frequency-resolved optical gating," *Opt. Lett.*, vol. 22, pp. 519–521, 1997.

[11] A. J. Taylor, G. Rodriguez, and T. S. Clement, "Determination of n2 by direct measurement of the optical phase," *Opt. Lett.*, vol. 21, pp. 1812–1814, 1996.

[12] V. Wong and I. A. Walmsley, "Linear filter analysis of methods for ultrashort-pulse-shape measurements," *J. Opt. Soc. Amer. A*, vol. 12, pp. 1491–1499, 1995.

[13] J. R. Fienup, "Reconstruction of a complex-valued object from the modulus of its Fourier transform using a support constraint," *J. Opt. Soc. Amer. A*, vol. 4, pp. 118–123, 1987.

[14] R. P. Milane, "Multidimensional phase problems," *J. Opt. Soc. Amer. A*, vol. 13, pp. 725–734, 1996.

[15] K. W. DeLong and R. Trebino, "Improved ultrashort phase-retrieval algorithm for frequency- resolved optical gating," *J. Opt. Soc. Amer. A*, vol. 11, pp. 2429–2437, 1994.

[16] K. W. DeLong, D. N. Fittinghoff, R. Trebino, B. Kohler, and K. R. Wilson, "Pulse retrieval in frequency-resolved optical gating based on the method of generalized projections," *Opt. Lett.*, vol. 19, pp. 2152–2154, 1994.

[17] D. J. Kane, "New algorithm for the measurement of two ultrashort laser pulses from a single spectrogram," presented at the Conference on Lasers and Electro-Optics, Baltimore, MD, 1997.

[18] ——, "Real time measurement of ultrashort laser pulses using principal component generalized projections," *IEEE J. Select. Topics Quantum Electron.*, vol. 4, pp. 278–284, 1998.

[19] K. W. DeLong, R. Trebino, J. Hunter, and W. E. White, "Frequency-resolved optical gating with the use of second-harmonic generation," *J. Opt. Soc. Amer. B*, vol. 11, pp. 2206–2215, 1994.

[20] J. Paye, M. Ramaswamy, J. G. Fujimoto, and E. P. Ippen, "Measurement of the amplitude and phase of ultrashort light pulses from spectrally resolved autocorrelation," *Opt. Lett.*, vol. 18, pp. 1946–1948, 1993.

[21] R. Trebino, K. W. DeLong, D. N. Fittinghoff, J. N. Sweetser, M. A. Krumbügel, and D. J. Kane, "Measuring ultrashort laser pulses in the time-frequency domain using frequency-resolved optical gating," *Rev. Sci. Instrum.*, 1997.

[22] Y. Yang, N. P. Galatsanos, and H. Stark, "Projection-based blind deconvolution," *J. Opt. Soc. Amer. A*, vol. 11, pp. 2401–2409, 1994.

[23] E. Yudilevich, A. Levi, G. J. Habetler, and H. Stark, "Restoration of signals from their signed Fourier-transform magnitude by the method of generalized projections," *J. Opt. Soc. Amer. A*, vol. 4, pp. 236–246, 1987.

[24] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C: The Art of Scientific Computing*, 2nd ed. Cambridge, U.K.: Cambridge Univ., 1995.

[25] K. W. DeLong, C. L. Ledera, R. Trebino, B. Kohler, and K. R. Wilson, "Ultrashort-pulse measurement using noninstantaneous nonlinearities: Raman effects in frequency-resolved optical gating," *Opt. Lett.*, vol. 20, pp. 486–488, 1995.

[26] K. W. DeLong, R. Trebino, and W. E. White, "Simultaneous recovery of two ultrashort laser pulses from a single spectrogram," *J. Opt. Soc. Amer. B*, vol. 12, pp. 2463–2466, 1995.

[27] A. K. Jain, *Fundamentals of Digital Image Processing*, 1st ed. Englewood Cliffs: Prentice Hall, 1989.

[28] H. Anton, *Elementary Linear Algebra*, 2nd ed. New York: Wiley, 1977.

[29] C. W. Siders, A. J. Taylor, and M. C. Downer, "Multi-pulse interferometric frequency-resolved optical gating: Real time phase-sensitive imaging of ultrafast dynamics," vol. 22, pp. 624–626, 1997.

[30] C. W. Siders, J. L. W. Siders, and A. J. Taylor, "Femtosecond coherent spectroscopy at 800 nm: MI-FROG measures high-field ionization rates in gases," presented at the Ultrafast Phenomena XI, 1998.

[31] J. L. A. Chilla and O. E. Martinez, "Direct determination of the amplitude and phase of femtosecond light pulses," *Opt. Lett.*, vol. 16, pp. 39–41, 1991.

Daniel J. Kane, for photograph and biography, see this issue, p. 420.