A SOLUTION TO CERTAIN POLYNOMIAL EQUATIONS WITH APPLICATIONS TO NONLINEAR FITTING

CHRIS CONNELL

ABSTRACT. We present a combinatorial method for solving a certain system of polynomial equations of Vandermonde type in $2N$ variables by reducing it to the problem of solving two special linear systems of size $N$ and rooting a single univariate polynomial of degree $N$. Over $\mathbb{C}$, all solutions can be found with fixed precision using, up to polylogarithmic factors, $O(N^2)$ bitwise operations in the worst case. Furthermore, if the data is well conditioned, then this can be reduced to $O(N)$ bit operations, up to polylogarithmic factors. As an application, we show how this can be used to fit data to a complex exponential sum with $N$ terms in the same, nearly optimal, time.

1. Introduction

In this paper we present a combinatorial method for solving the following system of polynomial equations in unknowns $A_i$ and $X_i$ where the elements $F_i$ lie in a (commutative) integral domain $\mathbb{D}$.

\[
\begin{align*}
F_1 &= A_1 + A_2 + \cdots + A_N \\
F_2 &= A_1X_1 + A_2X_2 + \cdots + A_NX_N \\
& \vdots \\
F_{2N} &= A_1X_1^{2N-1} + A_2X_2^{2N-1} + \cdots + A_NX_N^{2N-1}.
\end{align*}
\]

Certainly, any polynomial system which is regular can be reduced to solving univariate polynomials by producing resultants via Sylvester’s elimination method or other algorithms which produce a Gröbner basis. Simply applying the Weak Nullstellensatz gives a criteria for when solutions exist in terms of the associated ideal. The problem is that such general methods do not usually provide the most efficient procedures for accomplishing these tasks. For instance, one can easily check that for the simple case of $F_i = 1 + 1/i \in \mathbb{Q}$ in the above system, at least for $N \leq 5$, one must still root $N$ polynomials with degrees $1, \ldots, N$ even once the Gröbner basis has been computed. This holds regardless of elimination order.

The reduction we present provides an explicit and computationally reasonable method for determining when solutions of this system exist and what they are. We will show that when solutions exist, they lie in an extension (of degree at most $N$) of the fraction field, $\mathbb{K}$, of the integral domain $\mathbb{D}$, although we could (almost) as easily look for solutions in any integral domain extending $\mathbb{D}$. We summarize the solution as follows.

Received by the editor January 18, 2002 and, in revised form, January 6, 2003.
2000 Mathematics Subject Classification. Primary 65H10.
The author was supported in part by an NSF postdoctoral fellowship.

©2004 American Mathematical Society
Theorem I. The problem of solving system (1) is equivalent to (in order) solving a certain linear Hankel system, factoring a univariate degree $N$ polynomial, and solving a linear Vandermonde system. Moreover this reduction is given explicitly.

More generally, for $i = 1, \ldots, N$ choose elements $q_i$ and $r_i$ from the set of exponents of $\mathbb{D}$ (i.e., those elements $y \in \mathbb{D}$ such that $x^y$ is well defined and exists in $\mathbb{D}$ for all $x \in \mathbb{D}$). Then for any positive integer $M$ we may consider the over/underdetermined system of $\mathbb{D}$-rational equations below.

\[
\begin{align*}
F_1 &= A_1 X_1^{r_1} + A_2 X_2^{r_2} + \cdots + A_N X_N^{r_N} \\
F_2 &= A_1 X_1^{r_1 + r_1} + A_2 X_2^{r_2 + r_2} + \cdots + A_N X_N^{r_N + r_N} \\
&\quad \vdots \\
F_M &= A_1 X_1^{r_1(M-1)+r_1} + A_2 X_2^{r_2(M-1)+r_2} + \cdots + A_N X_N^{r_N(M-1)+r_N}.
\end{align*}
\]

After proving the theorem for system (1) in Section 2 we will explain how to solve system (2) in like manner. One item of importance of these particular systems of equations arises from their connection to the problem of fitting a sum of real or complex exponentials. In fact solutions to the above systems result in an efficient method for finding the coefficients $\beta_i$ and exponents $\alpha_i$ which provide a “best fit” approximation of a function $F(t)$, or discrete samples $F = \{(t_1, f_1), \ldots, (t_n, f_n)\}$, by a function of the form

\[
\tilde{F}(t) = \sum_{i=1}^N \beta_i e^{\alpha_i t},
\]

for any choice of $N$. Here the $\beta_i$ and the $\alpha_i$ depend only on the $A_i$ and $X_i$ in a relatively simple way. This will be explored in Section 3.

2. Solving the equations

For the first step in solving equations (1), notice that these equations are linearly dependent on the $A_i$. We can then easily eliminate the $A_i$ from the first $N$ equations. To do this, use the first $N$ equations of (1) to solve for the $A_i$,

\[
\begin{bmatrix}
F_1 \\
F_2 \\
\vdots \\
F_N
\end{bmatrix} =
\begin{bmatrix}
1 & 1 & \cdots & 1 \\
X_1 & X_2 & \cdots & X_N \\
\vdots & \vdots & \ddots & \vdots \\
X_1^{N-1} & X_2^{N-1} & \cdots & X_N^{N-1}
\end{bmatrix}
\begin{bmatrix}
A_1 \\
A_2 \\
\vdots \\
A_N
\end{bmatrix},
\]

and then substitute these into the last $N$ equations of system (1). Recognizing the matrix above as the transpose of the Vandermonde matrix, we can formally compute the entries.

First we establish some notation. Let $\text{Alt}$ be the elementary alternating polynomial in the variables $X_1, \ldots, X_N$. Explicitly,

\[
\text{Alt} = \prod_{k=1}^N \prod_{j>k} (X_k - X_j).
\]
Similarly, let $\text{Alt}_i$ be the elementary alternating polynomial in the variables $X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_N$. Let $\text{Sym}_j$ be the $j$th elementary symmetric polynomial in the variables $X_1, \ldots, X_N$. Explicitly, for $1 \leq j \leq N$,

\begin{equation}
\text{Sym}_j = \sum_{l_1=1}^{N} \sum_{l_2=l_1+1}^{N} \cdots \sum_{l_j=l_{j-1}+1}^{N} \prod_{k=1}^{j} X_{l_k},
\end{equation}

and $\text{Sym}_0 = 1$.

Lastly, let $\text{Sym}_{i,j}$ be the $j$th elementary symmetric polynomial in the variables $X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_N$. We will outline a proof to the following well-known lemma.

**Lemma 2.1.** With the above notation, whenever $X_i \neq X_j$ for $j \neq i$, the $A_i$ satisfy

\begin{equation}
A_i = \frac{(-1)^{N+i+1} \text{Alt}_i}{\text{Alt}} \sum_{j=1}^{N} (-1)^j \text{Sym}_{i,N-j} F_j.
\end{equation}

**Proof.** Consider the entries of the transposed Vandermonde matrix as formal monomials with coefficients in $Z$. The minor $M_{i,j}$ of the transposed Vandermonde matrix corresponding to position $(i,j)$ is an alternating polynomial in all of the variables except $X_j$. Every alternating polynomial $F$ (except in characteristic 2) may be formally expressed as a product, $F = \text{Alt} \text{ Sym}$, of the elementary alternating polynomial $\text{Alt}$ and some symmetric polynomial $\text{Sym}$. Hence $M_{i,j} = \text{Alt} \text{ Sym}_{i,j}$ where for each $i$ and $j$, $\text{Sym}_{i,j}$ is symmetric in all of the variables except $X_j$. For any row $i$ (resp. column $j$), the alternating sum of the $M_{i,j}$ in $j$ (resp $i$) is $\text{Alt}$. One easily checks that this relationship is satisfied when $\text{Sym}_{i,j} = \text{Sym}_{i,N-j}$ (uniquely for coefficients in $Z$).

Recall that a matrix with entries in $D$ is invertible in $D$ if and only if its determinant is a unit in $D$. If that happens, then the inverse matrix is the transpose of the matrix of minors with alternating signs divided by the determinant. It follows that the $(i,j)$ entry of the inverse of the Vandermonde matrix is

\[ \frac{(-1)^{N+i+1} \text{Alt}_i}{\text{Alt}} \text{Sym}_{i,N-j}. \]

So for any fixed $X_1, \ldots, X_N$ belonging to a commutative domain $\mathbb{K}$, if $X_i \neq X_j$ for any two indices $i \neq j$, then $\text{Alt} \neq 0$ and the Vandermonde matrix is invertible in $\mathbb{K}[X_1, \ldots, X_N]$. Thus the formula for the $A_i$ is valid in this domain and hence in the fraction field for $\mathbb{K}$ as well. \hfill \Box

**Remark 2.2.** One approach to finding solutions where some of the $X_i$, say $X_{N-r}, \ldots, X_N$, coincide is to make this substitution to begin with. In this case, system (1) reduces to the overdetermined system (2) with $r_1 = 0$, $q_i = 1$, $M = N$ and $N$ replaced by $N - r$. When solutions to this latter system exist, say in a field $k/K$, there is an $r$ dimensional space (over $k/K$) of solutions to the $A_{N-r}, \ldots, A_N$ in system (1).

Substituting the $A_i$ back into equations (1) then yields relations for the $F_{k+N}$ in terms of symmetric polynomials in the $X_i$ and the $F_i$. More specifically, if for each $j$ and $k$, $C_{k,j}$ denotes the coefficients for $F_j$ in the expression for $F_{k+N}$, then
the second half of the system becomes
\[
\begin{align*}
F_{N+1} &= C_{1,1}F_1 + C_{1,2}F_2 + \cdots + C_{1,N}F_N \\
F_{N+2} &= C_{2,1}F_1 + C_{2,2}F_2 + \cdots + C_{2,N}F_N \\
& \vdots \\
F_{2N} &= C_{N,1}F_1 + C_{N,2}F_2 + \cdots + C_{N,N}F_N,
\end{align*}
\]
where we have
\[
C_{k,j} = \frac{(-1)^{N+j+1}}{\text{Alt}} \sum_{i=1}^{N} (-1)^i \text{Alt}_i \text{Sym}_{i,N-j} X_i^{k+N-1}.
\]

Now we will show that the \( C_{k,j} \) satisfy a recursion relation.

**Proposition 2.3.** The \( C_{k,j} \) satisfy
\[
C_{k,j} = (-1)^{N+j} \text{Sym}_{N-j+1} C_{k-1,N} + C_{k-1,j-1},
\]
with
\[
C_{0,N} = 1, \quad C_{0,j} = 0 \quad \text{for} \quad 0 \leq j < N, \quad \text{and} \quad C_{k,0} = 0 \quad \text{for all} \quad k \geq 0.
\]

**Proof.** To verify this, we simply compute
\[
C_{k,j} - (-1)^{N+j} \text{Sym}_{N-j+1} C_{k-1,N} - C_{k-1,j-1}
\]
\[
= \frac{(-1)^{N+j+1}}{\text{Alt}} \sum_{i=1}^{N} (-1)^i \text{Alt}_i \text{Sym}_{i,N-j} X_i^{k+N-1}
\]
\[
- (-1)^{N+j} \text{Sym}_{N-j+1} \frac{1}{\text{Alt}} \sum_{i=1}^{N} (-1)^i \text{Alt}_i \text{Sym}_{i,0} X_i^{k+N-2}
\]
\[
- \frac{(-1)^{N+j}}{\text{Alt}} \sum_{i=1}^{N} (-1)^i \text{Alt}_i \text{Sym}_{i,N-j+1} X_i^{k+N-2}
\]
\[
= \frac{(-1)^{N+j+1}}{\text{Alt}} \sum_{i=1}^{N} \left( \text{Sym}_{i,N-j} X_i - \text{Sym}_{N-j+1} + \text{Sym}_{i,N-j+1} \right) (-1)^i X_i^{k+N-2} \text{Alt}_i
\]
\[
= 0
\]
since the elementary symmetric polynomials satisfy
\[
\text{Sym}_{N-j+1} = \text{Sym}_{i,N-j} X_i + \text{Sym}_{i,N-j+1},
\]
where we assign \( \text{Sym}_{i,N} = 0 \).

The base cases of the recursion may be verified as follows. To show \( C_{0,N} = 1 \),
we verify by switching variables that the numerator of
\[
C_{0,N} = \frac{-1}{\text{Alt}} \sum_{i=1}^{N} (-1)^i \text{Alt}_i X_i^{N-1}
\]
is an alternating polynomial of total degree \( \binom{N}{2} \) in the variables \( X_1, \ldots, X_N \) and
hence by the principal theorem of alternating polynomials it must be a constant multiple of \( \text{Alt} \). To show that the constant is 1, we simply substitute the values \( X_i = i \) to obtain
\[
C_{0,N} = -\sum_{i=1}^{N} (-1)^i \frac{(-1)^{N-1}i^{N-1}}{i!(N-i)!} = \frac{(-1)^N}{N!} \sum_{i=1}^{N} \binom{N}{i} (-1)^i i^N = 1.
\]
The last equality follows from recognizing the second to last expression as the Sterling number $S_N^M$.

Similarly, for the $C_{0,j}$ with $j < N$ we expand recursively on $j$, using the relations (7), as follows

$$C_{0,j} = \frac{(-1)^{N+j+1}}{\text{Alt}} \sum_{i=1}^{N} (-1)^i \text{Sym}_{i,N-j} \text{Alt}_i X_i^{N-1}$$

$$= \frac{(-1)^{N+j+1}}{\text{Alt}} \left( \text{Sym}_{N-j+1} \sum_{i=1}^{N} (-1)^i \text{Alt}_i X_i^{N-2} \right.$$

$$\left. - \sum_{i=1}^{N} (-1)^i \text{Sym}_{i,N-j+1} \text{Alt}_i X_i^{N-2} \right)$$

$$\vdots$$

$$= \frac{(-1)^{N+j+1}}{\text{Alt}} \left( \sum_{k=1}^{j} (-1)^{k+1} \text{Sym}_{N-j+k} \sum_{i=1}^{N} (-1)^i \text{Alt}_i X_i^{N-1-k} \right).$$

However, we may observe by switching pairs of variables $X_i$ and $X_j$ for all $i \neq j$ that the sum

$$\sum_{i=1}^{N} (-1)^i \text{Alt}_i X_i^{N-1-k}$$

is an alternating polynomial. Formally considering the coefficients to lie in $\mathbb{Z}$, it must then be a product of a symmetric polynomial and Alt. However for $k \geq 1$, its degree is less than Alt so each sum is zero in $D$.

Lastly, $C_{k,0} = 0$ since $\text{Sym}_{i,N} = 0$. \hfill \Box

Using the above recurrence formula, the relationship between the $X_i$ and $F_i$ has been somewhat simplified. For instance, with $N = 3$, the system (8) becomes the following.

\[
\begin{bmatrix}
F_1 \\
F_2 \\
F_3
\end{bmatrix} =
\begin{bmatrix}
\text{Sym}_3 & -\text{Sym}_3 & \text{Sym}_3 \\
\text{Sym}_{3,3} & \text{Sym}_4 - \text{Sym}_{3,3} & \text{Sym}_{4,3} \\
(\text{Sym}_{3,3}^2 - \text{Sym}_{3,3}) & \text{Sym}_{3,3}^2 - \text{Sym}_{3,3}^3 & \text{Sym}_{4,3} + \text{Sym}_{3,3}^3 & \text{Sym}_{4,3}^3 - 2 \text{Sym}_3 \text{Sym}_{3,3} + \text{Sym}_{3,3}^4
\end{bmatrix}
\begin{bmatrix}
F_1 \\
F_2 \\
F_3
\end{bmatrix}
\]

The above proposition allows us to apply symmetric reduction to system (11) since the $X_i$ variables only appear in symmetric polynomials. Next we will show that system (11) has a simpler form.

**Proposition 2.4.** The system (11) may be expressed as the following linear system.

\[
\begin{bmatrix}
F_{N+1} \\
F_{N+2} \\
\vdots \\
F_{2N}
\end{bmatrix} =
\begin{bmatrix}
F_N & -F_{N-1} & \cdots & (-1)^{N+1}F_1 \\
F_{N+1} & -F_N & \cdots & (-1)^{N+1}F_2 \\
\vdots & \vdots & \ddots & \vdots \\
F_{2N-1} & -F_{2N-2} & \cdots & (-1)^{N+1}F_N
\end{bmatrix}
\begin{bmatrix}
\text{Sym}_1 \\
\text{Sym}_2 \\
\vdots \\
\text{Sym}_N
\end{bmatrix}
\]

Moreover, for any solution to system (11), the elementary symmetric polynomials in the $X_i$ satisfy the above linear relation.
Proof. Repeatedly applying the recurrence relation from Proposition 2.3 on the kth equation of (6) for \( k \geq 1 \) we obtain

\[
F_{N+k} = \sum_{i=1}^{N} C_{k,i} F_i = (-1)^{N+1} F_1 \text{Sym}_N C_{k-1,N} + \sum_{i=2}^{N} F_i \left((-1)^{N+i} \text{Sym}_{N-i+1} C_{k-1,N} + C_{k-1,i-1}\right)
\]

\[
= C_{k-1,N} \sum_{i=1}^{N} (-1)^{N+i} F_i \text{Sym}_{N-i+1} + \sum_{i=2}^{N} F_i C_{k-1,i-1}
\]

\[
= \sum_{i=2}^{N+1} F_i C_{k-1,i-1} = \sum_{i=1}^{N} F_{i+1} C_{k-1,i}
\]

\[
= F_2 C_{k-1,1} + \sum_{i=2}^{N} F_{i+1} \left((-1)^{N+i} \text{Sym}_{N-i+1} C_{k-2,N} + C_{k-2,i-1}\right)
\]

\[
= \cdots = F_i C_{k-1,i} = \cdots = \sum_{i=1}^{N} F_{i+2} C_{k-2,i}
\]

\[
= \cdots = \sum_{i=1}^{N} F_{i+k-1} C_{1,i} = \sum_{i=1}^{N} F_{i+k-1} (-1)^{N+i} \text{Sym}_{N-i+1}
\]

\[
= \sum_{i=1}^{N} F_{N-i+k} (-1)^{i+1} \text{Sym}_i.
\]

Therefore these equations reduce to the \( N \) linear equations

\[
F_{N+k} = \sum_{i=1}^{N} F_{N-i+k} (-1)^{i+1} \text{Sym}_i,
\]

as desired.

The above reductions also show that for any solution to system (1) the elementary symmetric polynomials in the \( X_i \) satisfy (8). This is immediate whenever the \( X_i \) are distinct. The coincidence of two or more of the \( X_i \) invalidates formula (5). Nevertheless, Proposition 2.3 implies, in particular, that the \( C_{i,j} \) are polynomial in the \( X_i \). It follows from continuity in the parameters that the elementary symmetric polynomials in the \( X_i \) still satisfy (8) in this case.

The system (8) may be easily solved for the \( \text{Sym}_i \) assuming that the system is nonsingular. Replacing \( \text{Sym}_i \) by \((-1)^{N-i} \text{Sym}_{N-i+1}\) in system (8) and performing the same operations on the corresponding columns of the matrix yields the following
system.

\[
\begin{bmatrix}
F_{N+1} \\
F_{N+2} \\
\vdots \\
F_{2N}
\end{bmatrix}
= 
\begin{bmatrix}
F_1 & F_2 & \cdots & F_N \\
F_2 & F_3 & \cdots & F_{N+1} \\
\vdots & \vdots & \ddots & \vdots \\
F_N & F_{N+1} & \cdots & F_{2N-1}
\end{bmatrix}
\begin{bmatrix}
(-1)^{N-1} \text{Sym}_N \\
(-1)^{N-2} \text{Sym}_{N-1} \\
\vdots \\
\text{Sym}_1
\end{bmatrix}
\]

This we recognize as a Hankel system which, like Vandermonde systems, can be solved much more efficiently than general systems as we will see in Section 2.2.

The original objective was to solve for the \(X_j\) in terms of the \(F_j\). However, treating the \(\text{Sym}_i\) as known constants, the system of equations (9) in variables \(X_1, \ldots, X_N\) has, for each solution, up to \(N!\) redundant solutions corresponding to permutations of the \(X_i\). We are only interested in obtaining all the solutions up to permutation. Hence we wish to reduce the degree of this symmetric system. The following lemma is classical for fields, but nevertheless we will present a short proof to highlight its applicability to commutative rings in general.

**Lemma 2.5.** For elements \(S_i\) in any unital commutative ring \(R\), consider the system of equations \(S_i = \text{Sym}_i\) for \(i = 1, \ldots, N\), where \(\text{Sym}_i\) are given by (9). In any commutative ring \(R'\) containing \(R\), a solution \(X_1, \ldots, X_N\) to this system is also a complete set of roots in \(R'\) for the monic polynomial

\[
P(X) := X^N + \sum_{i=1}^N X^{N-i}(-1)^i S_i.
\]

Conversely every complete set of roots for the polynomial is also a solution to the system.

**Proof.** Recall that \(\text{Sym}_i = \sum_{1 \leq i_1 < \cdots < i_l \leq N} \prod_{m=1}^l X_{i_m}\). For any solution set \(X_1, \ldots, X_N\) of the system \(S_i = \text{Sym}_i\) for \(i = 1, \ldots, N\), substituting \(\text{Sym}_i\) for \(S_i\) into the equation for \(P(X)\), we find

\[
P(X_k) = X_k^N + \sum_{i=1}^N (-1)^i X_k^{N-i} \text{Sym}_i(X_1, \ldots, X_N).
\]

The \((-1)^N \text{Sym}_N\) term cancels with the \(X_k\) term of \((-1)^{N-1} X_k^1 \text{Sym}_{N-1}\) and the \(X_k^2\) terms of this quantity cancel with the \(X_k^2\) terms of \((-1)^{N-2} X_k^2 \text{Sym}_{N-2}\), and so forth, until finally the remaining \(X_k^N\) term of \(-X_k^{N-1} \text{Sym}_1\) cancels \(X_k^N\), showing that \(P(X_k) = 0\).

Conversely, if \(P(X) = \prod_{k=1}^N (X - X_k)\), then expanding this product shows that the coefficient \((-1)^k \text{Sym}_k\) of \(X^k\) is \((-1)^k \text{Sym}_k\) as expected. In these verifications we only use commutativity of the ring operations. \(\square\)

**Completing the proof of Theorem 1.** Now we have the prescription for solving the original system (1). Starting with known \(F_i \in \mathbb{D}\), we use equation (8) to solve for the \(\text{Sym}_i\) which will be rational in the \(F_i\) whenever solutions exist. We then substitute \(\text{Sym}_i \in \mathbb{K}\) for the \(S_i\) in equation (10). For a complete factorization of the polynomial \(P(X)\) we obtain the solution of the \(X_i\) in an extension field \(\mathbb{K}\) of at most degree \(n\) over \(\mathbb{K}\). Then we substitute these values into (8) and solve for the \(A_i\) which also live in \(\mathbb{K}\) whenever solutions exist.

Conversely, given a solution \(X_i, A_i\) for \(i = 1, \ldots, N\) of system (1), Proposition 2.4 states that the elementary symmetric polynomials in \(X_i\) form solve system (8).
and by Lemma 2.3 the $X_i$ form a factorization of $P(X)$. It then follows that the $A_i$ must satisfy (3). Therefore, if either of the two linear systems or the factorization problem fails to admit a solution, then system (1) does not admit solutions. □

Solutions, when they exist, naturally live in the Galois extension $k/\mathbb{K}$ which splits $P(X)$. The problem of looking further for solutions in $\mathbb{D} \subset k/\mathbb{K}$ is then only as hard as the membership problem.

Since system (3) is essentially a Hankel system, much can be said about when solutions exist in terms of $F_1, \ldots, F_{2N}$ (see for instance [Gou97, ISL92, Las90]). Similarly, from equation (5) we see that solutions to the $A_i$ exist whenever the $X_i$ are distinct, or else whenever $X_i = X_j$ for two indices $i, j$, the values $F_1, \ldots, F_N$ satisfy $\sum_{i=1}^{N} (-1)^i \text{Sym}_{k, N-i} F_i = 0$ for $k = i$ and $k = j$.

Let $k$ be any field extending $\mathbb{K}$. The solution variety $V \subset k^{2N}$ for system (1) is fibered in the following way. For each solution $\tilde{s}$ in the affine subspace $S \subset k^N$ of solutions to (3) we consider the (possibly empty) set $L_{\tilde{s}}$ of all distinct (including permutations) complete factorizations of $P(X)$ with coefficients $S_i$ taken from $\tilde{s}$ in order. Each of these forms the coordinates of a point $\tilde{x} \in k^N$. For each $\tilde{x} \in L_{\tilde{s}}$ we have a (possibly empty) affine subspace $T_{\tilde{x}} \subset k^N$ of solutions to (3) for this factorization. Hence

$$V = \bigcup_{\tilde{s} \in S} \bigcup_{\tilde{x} \in L_{\tilde{s}}} (\tilde{x}, T_{\tilde{x}}).$$

Since $T_{\tilde{x}}$ is independent of the order of $\tilde{x}$, the symmetric group $S_N$ acts on $V$ (not necessarily freely) by permuting the first $N$ coordinates and the second $N$ coordinates the same way. The quotient $\tilde{V} = V/S_N$ is the reduced variety and we will say that system (1) is regular if $\tilde{V}$ is exactly one point. Note that if system (1) is regular for any nontrivial field, then the system is regular for any field in which it admits a solution.

2.1. The case of system (2). We can relate the solutions of system (2) in a field $k/\mathbb{K}$ to the solutions of the simplified system (2) where we set $q_i = 1$ and $r_i = 0$ for all $i = 1, \ldots, N$ as follows.

Consider a solution, $A_i$ and $X_i (i = 1, \ldots, N)$, to the simplified system in a field $k'/\mathbb{K}$. For each set of solutions $x_i, i = 1, \ldots, N$, in $k/\mathbb{K}$ of the equations $X_i = x_i^{q_i}$, set $a_i = A_i x_i^{-r_i}$. Since $r_i$ is an exponent, $a_i \in k$. The quantities $X_i = x_i$ and $A_i = a_i$ then provide a solution to (2). Furthermore, all solutions in $k$ arise in this way as $k'$ is allowed to vary.

It remains to discuss the solution to system (1) in the case when the number of equations is $M \neq 2N$. Of course, when we have $M \leq N$, we could solve (2) for the $A_i$’s in $\mathbb{K}$ for any choice of $M$ distinct $X_i$’s which in turn would be a solution to the original system. Therefore we may assume $M > N$.

Calculating $A_1, \ldots, A_N$ as before and substituting these into the second $M - N$ equations, we obtain coefficients $C_{k,j}$ as in (10) for $0 \leq k \leq M - N$ and $1 \leq j \leq N$. Proposition 2.3 still holds when $M > N$, and in fact it gives the formula for $C_{k,j}$ for all $k \geq 1$ and $1 \leq j \leq N$. Hence, we obtain the same system of equations,

$$F_{N+k} = \sum_{i=1}^{N} F_{N-i+k} (-1)^{i+1} \text{Sym}_i,$$

for $k = 1, \ldots, M - N$. 

Lastly in the under-determined case $M < 2N$, the following system
\begin{equation}
\begin{bmatrix}
F_{N+1} \\
F_{N+2} \\
\vdots \\
F_M
\end{bmatrix} = \begin{bmatrix}
F_N & -F_{N-1} & \cdots & (-1)^{N+1}F_1 \\
F_{N+1} & -F_N & \cdots & (-1)^{N+2}F_2 \\
\vdots & \vdots & \ddots & \vdots \\
F_{M-1} & -F_{M-2} & \cdots & (-1)^{N+1}F_{M-N}
\end{bmatrix} \begin{bmatrix}
\text{Sym}_1 \\
\text{Sym}_2 \\
\vdots \\
\text{Sym}_N
\end{bmatrix}
\end{equation}

can be solved (generically) for a $2N - M$ dimensional space of solutions which correspond to completing this system with variables $F_{M+1}, \ldots, F_{2N}$ to transform it into the Hankel system (10).

In the over-determined case $M > 2N$, system (11) may be restricted to any subsystem of size $N$ each corresponding to rows $n$ through $N + n - 1$ for any $n = 1, \ldots, M - 2N$. This way the system will remain essentially Hankel after relabelling. Once any of these systems is solved one may easily check the remaining equations for consistency.

In either case, for each solution to system (11) we obtain the $X_i$ and $A_i$ as before by solving (10) and (3), respectively.

2.2. Time complexity. Now we consider the efficiency for the entire procedure on a random access machine. There are several different cases to consider. First we consider exact solutions in fields. Of course, in general, not much can be said since the fields themselves can be arbitrarily complicated for a computer to handle. That is, one arithmetic operation can take arbitrarily long, such as in an extension field over $\mathbb{Q}$ with an arbitrary large degree. The factorization problem is also very delicate in arbitrary fields. However, some of the most important cases are the finite and $p$-adic cases, $F_p$ and $\mathbb{Q}_p$. Over these fields, both the Hankel system (10) and the Vandermonde system (3) can be solved with $O(N \log^2(N) + N \log p)$ arithmetic operations (BDCMM01). Factoring the polynomial $P(X)$ in $F_p$ takes no worse than $O(N^2 \log p)$ operations (see [vzGP01, Sho95] for subquadratic algorithms when $q$ is relatively small compared to $n$). For not much worse, one can work in extensions of $\mathbb{Q}_p$ (see [Pan01]).

For the case of approximate solutions in $\mathbb{C}$, we wish to find a solution to the system with accuracy $2^{-b}$. We also want to consider an actual computer implementation, so we consider the time in bit operations instead of arithmetic operations. In the case of factorization, the latter usually masks an extra order of $N$ bit operations due to the necessity of high precision.

Both the Hankel system (10) and the the Vandermonde system (3) can be solved in $\mathbb{C}$ with $O(bN \log^2(N))$ bit operations with storage $O(N)$ (see [Pan02]). In [Pan02] it was recently shown that the problem of approximate factorization in $\mathbb{C}$ requires
\[ O\left((bN^2 \log bN \log^2 N \log \log bN)(\log^2 N + \log \log bN)\right) \]
bit operations for accuracy $2^{-b}$. On the other hand, this can be reduced, up to polylogarithmic factors, down to $O(bN)$ bit operations assuming that the zeros of the polynomial are not too tightly clustered ([Pan02]). In particular, the entire time complexity for finding the $X_i$’s and $A_i$’s is dominated by the factorizing problem. Hence up to polylog factors, it runs in $O(N^2)$ bit operations for fixed accuracy and in roughly $O(N)$ bit operations with well-conditioned data.

These running times are optimal up to polylogarithmic factors. In order to verify this, we begin by replacing $\text{Sym}_i$ in (5) by the coefficient $S_i$ of the monic polynomial.
Setting $F_1, \ldots, F_{N-1}$ all to 0 and $F_N = 1$, we may rewrite (8) as the following Toeplitz system.

\[
\begin{bmatrix}
S_1 \\
-S_2 \\
\vdots \\
(-1)^{N+1}S_N
\end{bmatrix} =
\begin{bmatrix}
1 & 0 & \cdots & 0 \\
-S_1 & 1 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
(-1)^{N-1}S_{N-1} & (-1)^{N-2}S_{N-2} & \cdots & 1
\end{bmatrix}
\begin{bmatrix}
F_{N+1} \\
F_{N+2} \\
\vdots \\
F_{2N}
\end{bmatrix}
\]

This can be solved for the $F_{N+1}, \ldots, F_{2N}$ up to polylog factors in $O(N)$ time ([Pan92]). On the other hand, if the $X_1, \ldots, X_N$ belong to a solution of (11) for this choice of $F_1, \ldots, F_{2N}$, then we showed that the $X_i$ represent all the roots of (10). Therefore solving (11) takes asymptotically at least as much time as a complete factorization. On the other hand, the stated $O(N^2)$ worst case time and $O(N)$ well-conditioned case time are optimal, up to polylog factors in $N$, for complete approximate factorization (see [Pan02]). The speed of this algorithm will become even more significant when we explore some applications below.

It is worth noting that the algorithm of [Pan02] is quite complicated and has not been implemented in any numerical package that we are aware of. Moreover, the widely used versions of the Jenkins-Traub factorization algorithm are still likely to be much faster than the near optimal algorithm for $N < 100$.

### 3. Applications

Here we present an application to a generic fitting problem. We will start with a general component minimization scheme before narrowing down to the special case of approximating by exponential sums. In what follows, let $K$ be any complete field with respect to some norm.

Given a measure space $(X, \mu)$, where $\mu$ is a $K$ valued measure, suppose that one has an arbitrary function $F \in L^2_{\text{loc}}(X, \mu)$ taking values in $K$. We may assume that $F$ is restricted to a region $S \subset X$, the sampling domain. We wish to find the best fit of $F$ as a linear combination of $N$ functions from the collection $\{f_\alpha\}_{\alpha \in A} \subset L^2_{\text{loc}}(X, \mu)$ for some parameter space $A \subset K$ and natural number $N$. The approximating function to $F$ will be

\[
\tilde{F}_N = \sum_{i=1}^N \beta_i f_{\alpha_i},
\]

for some choice of constants $\beta_i \in K$ and parameter indices $\alpha_i \in A$ such that $\tilde{F}_N$ is closest to $F$ in a certain sense that we will make precise shortly.

We need to find the best choice of coefficients $\beta_i$ and parameters $\alpha_i$ for $\tilde{F}_N$. To do this, we choose kernel functions $K_n \in L^2(S, \mu)$ based on the collection $\{f_\alpha\}_{\alpha \in A}$ (but independent of $\alpha$) such that for each integer $n = 0, \ldots, 2N - 1$,

\[
\int_S K_n(x)f_\alpha(x)d\mu(x) = G_S(\alpha)P_S(\alpha)^n,
\]

for functions $G_S$ and $P_S$ on $A$. The subscript $S$ is used to indicate that these functions, along with the kernel functions, may depend on the domain as well. Note that if $G_S(\alpha) = 0$ (respectively, $P_S(\alpha) = 0$), then $f_\alpha$ is orthogonal to all of the $K_n$ for $n = 0, \ldots$ (respectively, for $n = 1, \ldots$).
At this point, we put the above into the framework of system (1). Setting
\[ A_i = \beta_i G_S(\alpha_i) \] and \[ X_i = P_S(\alpha_i) \], we may easily verify that,
\[ \int_S K_n(x) \tilde{F}_N(x) d\mu(x) = \sum_{i=1}^{N} A_i X_i^n. \]

As this suggests, we can set \( F_i = \int_S K_{i-1}(x) F(x) d\mu(x) \) for \( i = 1, \ldots, 2N \) and then solve the system (1) for the \( A_i, X_i \). We must assume the \( F_i \) are finite, which holds independently of \( K_{i-1} \in L^2(S) \) if \( F \in L^2(S) \). We must further assume that \( P_S(A) \) contains all possible solution values for the \( X_i \) as the \( F_1, \ldots, F_{2N} \) range over all values produced from the class of functions \( F \) we wish to consider. For instance, this last condition is always satisfied if we choose \( A \) and the family \( \{K_n\} \) so that \( P_S(A) = \mathbb{R} \).

Under these conditions, then at least one solution of \( P_S(\alpha_i) = X_i \) exists with \( \alpha_i \in \mathcal{A} \). If \( A_i = 0 \), then we simply remove the corresponding term from the sum (13); otherwise we set \( \beta_i = \frac{A_i}{\epsilon \alpha_i} \). Substituting these values of \( \alpha_i \) and \( \beta_i \) in (13) will produce a candidate minimizer, \( \tilde{F}_N \), with at most \( N \) terms.

Any such \( \tilde{F}_N \) will satisfy
\[ \int_S K_n(x) \left( F(x) - \tilde{F}_N(x) \right) d\mu(x) = 0, \]
for \( n = 0, \ldots, 2N - 1 \). In other words, \( \tilde{F}_N - F \) lies in \( \text{Span}_{L^2} \{K_0, \ldots, K_{2N-1}\} \). Also note that whenever \( F(x) \) is given by a linear combination from \( \{f_\alpha\}_{\alpha \in \mathcal{A}} \) with at most \( N \) terms, then \( F \) will be included among the possibilities for \( \tilde{F}_N \).

For all \( n \geq 2N \), we have from the definition of \( C_{k,j} \) and the validity of Proposition 2.3 for all \( k \geq 0 \) that
\[ \int_S K_n(x) \tilde{F}_N(x) d\mu(x) = \sum_{i=1}^{N} A_i X_i^n = \sum_{i=1}^{N} C_{n-N+1,i} F_i. \]
The \( C_{n-N+1,i} \) are all homogeneous polynomials in the \( X_i \) of degree \( n - i + 1 \), but they may each have many terms. We will need to estimate the number of terms in \( C_{k,j} \) since we have no priori control over the size of the \( A_i \).

Observe that by iterating the recurrence relation in Proposition 2.3 we obtain
\[ C_{k,N} = \sum_{i=1}^{\min\{k,N\}} (-1)^{i+1} \text{Sym}_i C_{k-i,N}. \]
By induction on \( k \), we obtain that \( C_{k,N} \) is precisely the sum of all distinct monic monomials of total degree \( k \) in the variables \( X_1, \ldots, X_N \). For \( j \neq N \) it is not as apparent how to describe \( C_{k,j} \). Nevertheless, by iterating the recurrence relation twice, we obtain
\[ C_{k,j} = (-1)^{N-j} \text{Sym}_{N-j+1} C_{k-1,N} - \text{Sym}_{N-j+2} C_{k-2,N} + C_{k-2,j-2}. \]
Since \( \text{Sym}_i \) consists of the sum of all products of \( i \) distinct elements from \( X_1, \ldots, X_N \), it follows from the description of \( C_{k,N} \) that every term in the expansion of \( \text{Sym}_{N-j+2} C_{k-2,N} \) is represented in \( \text{Sym}_{N-j+1} C_{k-1,N} \). Since \( C_{1,j} = (-1)^{N-j} S_{N-j+1} \), by induction each term in \( C_{k,j} \) has the same sign \((-1)^{N-j}\).
Hence, mean there is a constant \( k \) or \( K \) hand side tends to 0 as either shows that for each \( j \in \{1, \ldots, N\} \), the quantity \( N \) or \( \frac{1}{2^j} k \) tends to 0 as either \( k \) or \( N \) tends to \( \infty \), so long as \( k \geq N \).

Therefore, if we assume that the sequence \( \{F_i\} \) tends to 0 and that for some \( \epsilon > 0 \), \( |F_S(\alpha)| \leq \frac{\epsilon}{2^{3/2}} - \epsilon \) for all \( \alpha \in \mathcal{A} \), then for all \( n > 2N \),

\[
\left| \int S K_n(x) \left( F(x) - \tilde{F}_N(x) \right) d\mu(x) \right| \leq |F_n| + \sum_{i=1}^{N} (2e^{3/2})^{-n+i-1} c_{n-N+1,i} F_i \leq |F_n| + Ce^{-\epsilon(n-N)},
\]

where \( C > 0 \) is a constant depending only on the largest \( F_i \). Note that the right-hand side tends to 0 as either \( n \) or \( N \) tends to \( \infty \) since \( n - N > N \) by assumption.

Now assume that \( F \in L^2(S) \), so that the \( F_i \) are square summable. Moreover, suppose the sequence \( \{K_n\} \) contains a nondegenerate basis of \( L^2(S) \), by which we mean there is a constant \( C' > 0 \) such that for all \( f \in L^2(S) \),

\[
\sum_{n \geq 0} (K_n, f)_L^2(S) \geq C' \|f\|^2_{L^2(S)}.
\]

In this case,

\[
\|F - \tilde{F}_N\|^2_{L^2(S)} \leq \sum_{n \geq 0} (K_n, F - \tilde{F}_N)_{L^2(S)}^2 \leq C' \sum_{n > 2N} \left( |F_n| + Ce^{-\epsilon(n-N)} \right)^2.
\]

Hence, \( \|F - \tilde{F}_N\|_{L^2(S)} \) tends to 0 as \( N \) tends to \( \infty \). If we only require that \( X_i < 1 \) for \( i = 1, \ldots, N \), then we still may conclude from equation (14) that each \( \tilde{F}_N \) belongs to \( L^2(S) \) for each \( N \).

Let \( H_{F,N} \) denote the Hankel matrix in system (9). Fix \( N \) and suppose \( F \) satisfies the condition that for \( i = N + 1, \ldots, 2N \), we have \( |F_i| < \frac{C}{N} \) where \( C \) is any upper bound for \( \|H_{F,N}^{-1}\| \). Then by system (9) the norm of the coefficients of the polynomial (10) are all less than \( \frac{1}{N} \) and therefore the roots \( X_i \) are all less than 1. In this case, if the \( \{K_n\} \) contain a nondegenerate basis, then \( \tilde{F}_N \in L^2(S) \).

We point out that with obvious modifications, the above analysis applies to the case when the \( K_n \) represent distributions instead of functions. Now we will focus on applying these ideas to the concrete example of exponentials.

4. Fitting exponentials

In terms of the context from the previous section, we set \( \mathbb{K} = \mathbb{C} \) and consider the collection of functions \( f_\alpha(x) = e^{-\alpha x} \) with \( \alpha \in \mathcal{A} = \mathbb{C} \) and any \( S \subset \mathbb{C} \). We may parameterize \( S \) by the variable \( t \). In short we will fit an equally spaced complex time series \( F_1, \ldots, F_{2N} \) to a sum of complex exponentials \( F(t) = \sum_{j=1}^{N} A_j e^{\alpha_j t} \).
We first assume that for $j = 1, \ldots, 2N$, the $F_j$ represent given approximations to $F(t_j)$ where $t_j = j/q + r$ for fixed constants $q$ and $r$. In terms of the previous section, this corresponds to taking $K_j = \delta_{t_j}$ for $j = 1, \ldots, 2N$, where $\delta_{t_j}$ is a Dirac delta at $t_j$. We can then solve for each $A_j$ and each $e^{\alpha_j}$ simply by using our algorithm to solve the corresponding version of (2) with $q_i = q$ and $r_i = r$.

**Example 4.1.** Let $F(t)$ be a series of $N$ exponential functions each of whose coefficients and exponents are randomly chosen using a uniform distribution over the square in $\mathbb{C}$ with opposite corners $\pm(1+i)$. To keep the exponentials from becoming too large or small, we set $q = 2N$ and $r = 0$ so that $F_j = F\left(\frac{t_j}{2N}\right)$ for $j = 0, \ldots, 2N - 1$.

We implemented our algorithm in the *Mathematica* computer algebra system, version 4.1, on a 900 MHz Pentium III machine equipped with 400 megabytes of RAM. Since the purpose of this example is to demonstrate the viability of the algorithm for relatively small $N$, we did not attempt to implement the optimal algorithm. Instead, we chose to use *Mathematica*'s built-in linear solver, which has computational complexity $O(N^3)$, for the two linear systems. We also used *Mathematica*'s implementation of the suboptimal Jenkins-Traub algorithm for the polynomial factorization.

Nevertheless, this implementation could still recover the original $F(t)$ with an accuracy of $10^{-9}$ from the $F_i$ in approximately 2.6 seconds of CPU time for $N = 100$. For $N = 1000$ the entire algorithm takes approximately 1100 seconds. The single most time consuming step is the rooting of the polynomial which was accomplished in 504 seconds of CPU time.

By contrast, *Mathematica*'s nonlinear regression routine, based on the Levenberg-Marquart algorithm, expended 15 seconds of CPU time on the same machine to fit an exponential sum $F(t)$ with $N = 100$, but restricted to real exponents and coefficients. Similarly, the case with $N = 1000$ exhausted 1239 seconds of CPU time. For some reason, the actual time to complete this fit was more than 12 times as long, even without any disk paging.

For many typical numerical applications, the $F_i$ may represent inaccurate or unevenly spaced data. For this situation we may want to rely on averaging techniques. For instance, when $S = [0, \infty)$ and we are modelling decaying exponentials, i.e., Re($\alpha_i) < 0$, then we can use our method when the Laplace transform of $K_n$ has the correct form. Table 1 is a partial list collected from [IK77] for possible choices of $K_n(x)$ along with the corresponding $G_S$ and $P_S$ which fit into the framework of Section 2. For instance, choosing $K_n(x) = \frac{x^{n-1}}{(n)}$ leads to equating the statistical moments of $F_N$ with those of $F$; other choices lead to different notions of approximation.

Here $J_n(t)$ is the $n$th Bessel $J$ function, $L_n(t)$ is the $n$th Laguerre polynomial, and $H_n(t)$ is the $n$th Hermite polynomial. We see from Table 1 that for these kernel functions either $\alpha$ is uniquely determined from $P_S(\alpha)$ for $\alpha$ on $\mathbb{C}$ (with a given branch cut), a half plane ($P_S = \frac{2p}{m^2 + p^2}$), or a strip (in the exponential cases).

For a given choice of the $K_n$ it takes at most $O(Nm)$ time to calculate the integrals $F_i$ for $i = 0, \ldots, 2N - 1$ when $F(t)$ has been sampled at $m$ points. Hence the entire algorithm to compute a fit $F_N$ is, in the worst case, $O(N^2 \log(N) + Nm)$ for a fixed precision. Note that for some choices of $K_n$ the integration can be done in $O(N + m)$ time.
Table 1.

<table>
<thead>
<tr>
<th>$K_n(t)$</th>
<th>$G_S(\alpha)$</th>
<th>$P_S(\alpha)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0  $(0 \leq t &lt; n)$</td>
<td>$\frac{1}{n}$</td>
<td>$e^{-\alpha}$</td>
</tr>
<tr>
<td>1  $(n \leq t)$</td>
<td>$\frac{1}{\sqrt{n}}$</td>
<td>$e^{-\sqrt{\alpha}}$</td>
</tr>
<tr>
<td>$e^{-pt\frac{n-1}{2(\alpha+t)}}$</td>
<td>$\frac{1}{\sqrt{n}}$</td>
<td>$e^{-\frac{1}{\sqrt{\alpha}}}$</td>
</tr>
<tr>
<td>$\frac{1}{\sqrt{n}} \sin(\sqrt{nt})$</td>
<td>$\sqrt{\frac{2}{\alpha + n^2}}$</td>
<td>$e^{-\frac{1}{\sqrt{\alpha}}}$</td>
</tr>
<tr>
<td>$\frac{1}{\sqrt{n}} \cos(\sqrt{nt})$</td>
<td>$\sqrt{\frac{2}{\alpha + n^2}}$</td>
<td>$\sqrt{1 + \alpha^2 - \alpha}$</td>
</tr>
<tr>
<td>$J_n(t)$</td>
<td>$\frac{1}{\sqrt{1 + \alpha^2}}$</td>
<td>$\sqrt{1 + \alpha^2 - \alpha}$</td>
</tr>
<tr>
<td>$\frac{1}{T} J_n(pt)$</td>
<td>$1$</td>
<td>$\frac{1}{p} \left( \sqrt{p^2 + \alpha^2} - \alpha \right)$</td>
</tr>
<tr>
<td>$\frac{1}{n} \frac{1}{\sqrt{n}} J_n(pt)$</td>
<td>$\frac{1}{\sqrt{n}} \frac{1}{\sqrt{1 + \alpha^2}}$</td>
<td>$\frac{2p}{\alpha^2 + p^2}$</td>
</tr>
<tr>
<td>$(\frac{1}{n})^{p/2} J_p(\sqrt{nt})$</td>
<td>$2^{-p/2} \alpha^{-1} (p+1)$</td>
<td>$e^{-\frac{1}{\sqrt{\alpha}}}$</td>
</tr>
<tr>
<td>$\frac{1}{n!} \frac{1}{\sqrt{n}} \frac{1}{\alpha^2 (\alpha+1)^{p/2}} L_n^p(T)$</td>
<td>$\frac{1}{\alpha (\alpha+1)^{3/2}}$</td>
<td>$\frac{\alpha - 1}{\alpha}$</td>
</tr>
<tr>
<td>$\frac{1}{(2n+1)!} H_{2n+1}(\sqrt{T})$</td>
<td>$\sqrt{\frac{2}{\alpha + n^2}}$</td>
<td>$\frac{1}{\alpha}$</td>
</tr>
<tr>
<td>$\frac{1}{\sqrt{(2n-1)!}} H_{2n}(\sqrt{T})$</td>
<td>$\sqrt{\frac{2}{\alpha + n^2}}$</td>
<td>$\frac{1}{\alpha}$</td>
</tr>
</tbody>
</table>

Appendix A. Other noniterative methods in special cases

Here we report on a couple of previously existing methods to handle special cases of the problem of fitting exponential series. The more well-known techniques of nonlinear regression and the Prony method have already been well analyzed in the literature, so we will not discuss them here. Though very effective for small $N$ these inherently iterative methods suffer from being asymptotically inefficient.

The special cases presented here have the potential to run in $O(N \log N)$ operations for well-behaved inputs and are both simple and elegant. Unfortunately, they only recover exponents $\alpha_j$ lying in a particular one real dimensional subset of $\mathbb{C}$. Moreover, their numerical deficiencies make them for the most part impractical.

The general scheme is to find an appropriate operator $\square$ such that $\square e^{\alpha t} = \delta(t-\alpha)$ where $\delta(t)$ is the Dirac delta distribution centered at 0. Thus if $F(t) = \sum_{j=1}^{N} A_j e^{\alpha_j t}$, we may express this as

$$F(t) = \int_{-\infty}^{\infty} f(s) e^{\alpha t} ds$$

for

$$f(s) = \sum_{j=1}^{N} A_j \delta(s - \alpha_j),$$

and $\square F(t) = f(t)$. 

License or copyright restrictions may apply to redistribution; see https://www.ams.org/journal-terms-of-use
When applying discrete versions of such an operator, the inherent numerical inaccuracies do not always make them a good choice for recovery of a best fit approximation, especially in the presence of noise. However, they can at least give a practical means of finding the number of terms, $N$, in the sum, the hope being that the number of peaks $N$ of a numerical realization of $\Box F(t)$ may be discernable even when their relative heights and positions are not.

In the first case, using the notation from Section 3, if $S = \mathbb{R}$ and $\alpha_j \in \mathbb{R} \cdot i$, then we could simply take the Fourier transform of $F$ to obtain a sum of Dirac measures centered at $-i \cdot \alpha_j$ with weights $A_j$. Numerically, this can be accomplished using a nonuniform fast Fourier transform algorithm and taking $A_j$ to be the total mass of each peak which will be centered approximately at $-i \cdot \alpha_j$. This can be done in $O(N \log N)$ time for fixed precision assuming the sampling is less than a fixed multiple of $N$ (DR93 and see NL99, Bey95, CFR84 for consequent improvements). However the constants in the time dependency may be large due to the need to resolve the peaks enough to accurately estimate the $A_j$. We also needed to assume that $F(t)$ can be sampled at an appropriate set of real values of $t$.

Similarly, when $\alpha_j \in \mathbb{R}$ with $\alpha_j < 0$ we could take the inverse Laplace transform
\[
 f(s) = \frac{1}{2\pi i} \int_{\sigma - i\infty}^{\sigma + i\infty} F(t)e^{st}dt \quad \text{for any} \quad \sigma > 0
\]
to obtain a sum of Dirac measures centered at $-\alpha_j$ with weights $A_j$. Numerically this can be done using an algorithm based on the fast Fourier transform (FFT) when $F$ has been sampled along the line $\sigma + i\mathbb{R}$. Otherwise, if $F$ is sampled only for positive real values (which one might commonly expect in practice), then one may use the inversion formula arising from the Mellin transform,
\[
 f(s) = \frac{1}{2\pi i} \int_{\sigma - i\infty}^{\sigma + i\infty} \frac{s^{-p}}{\Gamma(1-p)} \left( \int_0^\infty F(t)t^{-p}dt \right) dp.
\]
Here $\Gamma(x)$ is the Euler gamma function and the function $F$ must also satisfy certain analytic conditions (see AR00) in order for the integrals to converge. It seems that the general method described in [Str92] for fast evaluation of discrete transforms with sufficiently decaying kernels, when applied to the discrete application of the above transform given in [AR00], would give a means of computing this numerically in $O(N \log N)$ time.

The Gardner transform (GGLM59) provides an efficient alternate approach to inverting the Laplace transform in this case when $F(t)$ is sampled at real values only. Its advantage is that it can be simply evaluated from three FFT's.

To solve for $f(s)$ in terms of $F(t)$, we make it a convolution in order to conveniently take its Fourier transform. To do so, we make a transformation of variables in the Laplace transform. From the equation for $F(t)$ above under the assumption $\alpha_j < 0$, we recover,
\[
e^t F(e^t) = \int_{-\infty}^0 f(s)e^{se^t}ds.
\]
Then making a change of variables from $s$ to $-\beta e^{-x}$ (for any positive constant $\beta$) gives us
\[
e^t F(e^t) = \int_{-\infty}^\infty f(-\beta e^{-x}) e^{-\beta e^{t-x}} \beta e^{t-x}dx.
\]
We observe that the integral is a convolution with $K(z) = \beta e^{-\beta e^z} e^z$. Taking the Fourier transform of both sides then gives

$$
\mathcal{F}_{t \rightarrow \omega} \left( e^t F(e^t) \right) = \mathcal{F}_{x \rightarrow \omega} \left( f \left( -\beta e^{-z} \right) \right) \mathcal{F}_{x \rightarrow \omega} \left( K(x) \right).
$$

For the purpose of numerical stability, we multiply the left-hand side by a gaussian factor $e^{-\left( \Phi^2 \right)}$ for some constant $\nu$. Since this is equivalent to convolving $F(t)$ by the same factor at the outset, it acts as a low pass filter which cuts out baseline ripples which are far away from the central peak of $\mathcal{F}_{t \rightarrow \omega} \left( e^t F(e^t) \right)$ (see [SCSB76]). Good values of the constant $\nu$ vary, but often taking $\nu$ to be about half of the length of the interval on which the Fourier transform of $F$ is defined works well.

The final result is

$$
f \left( -\beta e^{-z} \right) = \mathcal{F}_{\omega \rightarrow z}^{-1} \left( \frac{\mathcal{F}_{t \rightarrow \omega} \left( e^t F(e^t) \right) e^{-\left( \Phi^2 \right)}}{\mathcal{F}_{x \rightarrow \omega} \left( K(x) \right)} \right).
$$

The purpose of the constant $\beta$ is to shift the peak of $K(z)$ by a factor of $\log(\beta)$ in order to center it in the domain of $e^t F(e^t)$ so that it goes to zero on either edge of this window. This reduces numerical inaccuracies ([SCSB76]).

The numerical problems with applying this transform mostly stem from the constraint on the domain of the data. For example, the maximum of $e^t F(e^t)$ where $F(t)$ is a single exponential $e^{\alpha t}$ occurs at $t = \log \frac{1}{\alpha}$ and decays very rapidly from that point. Therefore the domain of $F(t)$ should include points nearby to the points $\frac{1}{\alpha j}$ in order to obtain an accurate sampling. Clearly, integrating $e^t F(e^t)$ accurately from sampled data can be problematic since the domain of $F$ is compressed so rapidly.

Since $\mathcal{F}_{x \rightarrow \omega} \left( K(x) \right)$ appears in the denominator in the Gardner transform, there are inaccuracies introduced whenever it decays on the portion of the domain of the numerator representing significant data. By examining Laplace transforms of $K(x)$ on the positive and negative imaginary axes, we find that

$$
\mathcal{F}_{x \rightarrow \omega} \left( K(x) \right) = \beta^{-i2\pi s} \Gamma(1 + i2\pi s).
$$

Therefore it is also important to choose $\beta$ so that the domain where this function has small norm does not overlap with the domain of the data window used in the computation of $\mathcal{F}_{t \rightarrow \omega} \left( e^t F(e^t) \right)$. Since this later function is modelled by

$$
\sum_{j=1}^{N} A_j (-\alpha_j)^{-i(1+i2\pi s)} \Gamma(1 + i2\pi s),
$$

this may be impossible if the $\alpha_j$ are spread apart.

In practice, we found that when applying this algorithm with machine precision of $10^{-16}$, the number of peaks accurately corresponded to the number of exponential terms $N$ except when the $\alpha_j$ were separated by less than $10^{-3}$. Even when choosing optimal values of $\beta$ and $\nu$ with fewer than six $\alpha_j$ uniformly distributed between 0 and 1, the precision of the $\alpha_j$ was usually less than $10^{-3}$. On the other hand, even in cases when the peaks about the $\alpha_j$ are completely separated, it is difficult to estimate their mass to within a factor of 1 of the value of the original $A_j$. Due to the extensive overlap of peaks when two $\alpha_j$ are reasonably close, it is difficult to make any reasonable estimate at all.
ACKNOWLEDGMENTS

The author is grateful for the support of Kay Patterson and the department of anesthesiology at Brigham Women’s Hospital in Boston during the summer of 1993 where several ideas in this paper were first conceived.

REFERENCES


Department of Mathematics, University of Chicago, 5734 S. University Avenue, Chicago, Illinois 60637

Current address: Department of Mathematics, Rawles Hall, Indiana University, Bloomington, Indiana 47401

E-mail address: cconnell@math.uchicago.edu