JOURNAL OF COMPUTATIONAL PHYSICS 116, 369–379 (1995)

Fast Fourier Transforms of Piecewise Constant Functions

EUGENE SORETS*

Department of Mathematics, Yale University, P.O. Box 208283, New Haven, Connecticut 06520-8283

Received April 8, 1994

We present an algorithm for the evaluation of the Fourier transform of piecewise constant functions of two variables. The algorithm overcomes the accuracy problems associated with computing the Fourier transform of discontinuous functions; in fact, its time complexity is $O(N^2 \log N + NP \log^2(1/\varepsilon) + V \log^3(1/\varepsilon))$, where ε is the accuracy, N is the size of the problem, P is the perimeter of the set of discontinuities, and V is its number of vertices. The algorithm is based on the Lagrange interpolation formula and the Green's theorem, which are used to preprocess the data before applying the fast Fourier transform. It readily generalizes to higher dimensions and to piecewise smooth functions. © 1995 Academic Press, Inc.

1. INTRODUCTION

The fast Fourier transform (FFT) (see, for example, [2, 3, 8]) is a ubiquitous tool of numerical analysis, essential in signal processing, electrical engineering, VLSI circuit modeling, medical imaging, and innumerable other fields. It is a very effective tool when the function to be transformed is smooth; however, if the function has a jump discontinuity, then the accuracy of the FFT is significantly reduced. This loss of accuracy is the result of the simple fact that FFT, from the mathematical point of view, is a collection of integrals evaluated via the trapezoidal rule. Therefore, the numerical error that results from a jump discontinuity is on the order of N^{-1} for a problem of size N. Such errors make even single precision calculations prohibitively costly, especially in higher dimensions. The cost can be reduced with the help of the Richardson extrapolation, but double precision calculations are still virtually impossible.

In this paper we introduce an algorithm for efficient and accurate computation of the Fourier transform of piecewise constant functions. The algorithm uses Green's theorem to replace the area integrals that naturally occur in computing the two-dimensional Fourier transform of piecewise constant functions, with line integrals, thereby significantly reducing the number of nodes required for integration; after that the quadrature weights are redistributed onto the uniform grid with the help of the Lagrange interpolation formula, bringing the data to the form suitable for the FFT.

* Supported in part by the National Science Foundation under Grant DMS901213595.

The paper is organized as follows. In Section 2 we list the relevant mathematical and numerical facts. Section 3 contains the precise statement of the problem and Section 4 contains the notation. A description of the algorithm, along with the error estimates and complexity analysis, can be found in Section 5, and Section 6 contains the results of numerical experiments.

We present the analysis in two dimensions under the assumption that the functions to be transformed are piecewise constant and that the discontinuities occur along a collection of polygons. Neither of these is a serious limitation; the algorithm generalizes quite naturally to higher dimensions and to curved boundaries.

2. MATHEMATICAL AND NUMERICAL PRELIMINARIES

2.1. *Green's Theorem*

We will encounter area integrals of the form

$$\int_D e^{-2\pi i m x} e^{-2\pi i n y}\, dy\, dx, \tag{1}$$

where D is a bounded domain in $\mathbb{R}^2$. We will replace them with line integrals using the following version of Green's theorem in the plane:

$$\int_D \frac{\partial Q}{\partial x}\, dy\, dx = \int_\Gamma Q\, dy. \tag{2}$$

Here $Q : \mathbb{R}^2 \to \mathbb{C}^1$ is a function in $C^2(\overline{D})$ and Γ is the boundary of D travesed counterclockwise. Applying (2) to (1), we arrive at a formula that will be useful to us later:

$$\int_D e^{-2\pi i m x} e^{-2\pi i n y}\, dy\, dx = \begin{cases} \dfrac{1}{-2\pi i m} \displaystyle\int_\Gamma e^{-2\pi i m x} e^{-2\pi i n y}\, dy & \text{when } m \neq 0, \\ \displaystyle\int_\Gamma x e^{-2\pi i n y}\, dy & \text{when } m = 0. \end{cases} \tag{3}$$

2.2. *Lagrange Interpolation*

The Lagrange interpolation formula (see, for example, [1, 4]) approximates a function $f : \mathbb{R}^1 \to \mathbb{C}^1$ by the expression

$$f(x) = \sum_{m=1}^{p} f(x_m) \prod_{\substack{n=1 \\ n \neq m}}^{p} \frac{x - x_n}{x_m - x_n} + R_p(x), \tag{4}$$

where $x_1 < \cdots < x_p$ are points on the real line and $R_p(x)$ is the error term. For each $m = 1, ..., p$, we will denote by δ_m the polynomial defined by

$$\delta_m(x) = \prod_{\substack{n=1 \\ n \neq m}}^{p} \frac{x - x_n}{x_m - x_n}, \tag{5}$$

and observe that

$$\delta_m(x_n) = \begin{cases} 1 & \text{if } m = n, \\ 0 & \text{if } m \neq n. \end{cases}$$

In view of (5), formula (4) can be rewritten as

$$f(x) = \sum_{m=1}^{p} f(x_m)\delta_m(x) + R_p(\mathrm{x}). \tag{6}$$

$R_p(x)$ in Eqs. (4) and (6) is the error term and

$$R_p(x) = \frac{f^{(p)}(\zeta)}{p!} \prod_{n=1}^{p} (x - x_n) \quad \text{for some } \zeta \in (z_1, x_p).$$

In our case, the nodes $x_1, ..., x_p$ are going to be uniformly spaced with the sampling interval $h > 0$, and the point x will lie in the interval of length h centered on the center of the interpolation window $[x_1, x_p]$. Furthermore, the function f will be of the form $f(x) = \exp(2\pi ikx)$. A direct calculation shows that under these conditions $R_p(x)$ satisfies the inequality

$$|R_p(x)| \leq \frac{[\Gamma(p/2 + 1)]^2}{p!} \cdot h^p(2\pi k)^p, \tag{7}$$

where $\Gamma(z)$ is the gamma function: $\Gamma(z + 1) = z!$.

We will also make use of the two-dimensional Lagrange interpolation formula:

$$f(x, y) = \sum_{n=1}^{p} \sum_{m=1}^{p} \delta_m(x) \cdot \delta_n(y) \cdot f(x_m, y_n) + R_p(x, y). \tag{8}$$

Here the points $\{(x_m, y_n)\}_{m,n=1}^{p}$ are the interpolation nodes, the functions $\delta_1, ..., \delta_p$ are the same as in (5), and the error term, $R_p(x, y)$, can be estimated as

$$|R_p(x, y)| \leq \frac{[\Gamma(p/2 + 1)]^2}{p!} \cdot [h_x^p(2\pi k)^p + h_y^p(2\pi l)^p] \tag{9}$$

when the function $f(x, y)$ has the form $\exp(2\pi i(kx + ly))$, the nodes are uniformly spaced with the sampling interval h_x along the x-axis and h_y along the y-axis, and

$$(x, y) \in \left[x_c - \frac{h_x}{2}, x_c + \frac{h_x}{2}\right] \times \left[y_c - \frac{h_y}{2}, y_c + \frac{h_y}{2}\right], \tag{10}$$

with

$$(x_c, y_c) = \left(\frac{x_1 + x_p}{2}, \frac{y_1 + y_p}{2}\right). \tag{11}$$

2.3. *Gaussian Quadratures*

Gaussian quadrature (see, for example, [4, 7]) provides an approximation to the integral of a function $f: [1, 1] \to \mathbb{C}^1$,

$$\int_0^1 f(t)\, dt = \sum_{k=1}^{q} f(t_k)\omega_k + E_q(f), \tag{12}$$

where the points $t_1, ..., t_q$ are the Gaussian nodes on the interval $[0, 1]$, the numbers $\omega_1, ..., \omega_q$ are the corresponding Gaussian weights, and $E_q(f)$ is the error term. The Guassian nodes and weights are chosen to make the approximation (12) exact whenever the function f is a polynomial of degree less than $2q$. Consequently, Gaussian quadrature is effective for functions that are well approximated by polynomials; in fact (see, for example, [1]),

$$E_q(f) = \frac{(q!)^4}{(2q + 1)[(2q)!]^3} \cdot f^{(2q)}(\zeta) \quad \text{for some } \zeta \in [0, 1]. \tag{13}$$

The proof of (13) can be found in [4].

3. STATEMENT OF THE PROBLEM

Our goal in this paper is to find a way to compute Fourier transforms of piecewise constant functions in $\mathbb{R}^2$. In other words, we would like to construct an algorithm that for every piecewise constant function $f: \mathbb{R}^2 \to \mathbb{C}^1$ will compute an approximation to the collection of integrals, to which we will refer as the Fourier transform $\hat{f}$ of f,

$$\hat{f}(m, n) \overset{\text{def}}{=} \int_0^1 \int_0^1 f(x, y) e^{-2\pi imx} e^{-2\pi iny}\, dy\, dx, \tag{14}$$

where m and n are integers such that $-M < m \leq M$ and $-N < n \leq N$. We will refer to the pair (m, n) as the frequency. The pair (M, N), thus, is the highest frequency of interest. The most common approximation to the integrals (14), the discrete Fourier transform [2] is, *essentially*, the *trapezoidal rule*; its standard implementation is the fast Fourier transform (see, for example, [2, 3, 8]). For discontinuous functions the trapezoidal rule has accuracy on the order of N^{-1} for an N by N problem, or, conversely, for accuracy ε, FFT will require on the order of $(1/\varepsilon^2)\log(1/\varepsilon)$ operations; such performance is woefully inadequate in most practical problems. In this paper we solve the following problem.

Problem 1. For any given piecewise constant function $f:\mathbb{R}^2 \to \mathbb{C}^1$ with discontinuities along a finite number of polygons and for any given accuracy $\varepsilon > 0$, compute rapidly all the integrals (14) with accuracy ε.

4. NOTATION

Given a set $S \subset \mathbb{R}^2$, we denote by $\mathbf{1}_S:\mathbb{R}^2 \to \mathbb{C}^1$ the characteristic function of S:

$$\mathbf{1}_S(x,y) \overset{\text{def}}{=} \begin{cases} 1, & \text{if } (x, Y) \in S, \\ 0, & \text{otherwise.} \end{cases} \tag{15}$$

Given a piecewise constant function $f:\mathbb{R}^2 \to \mathbb{C}^1$, we write it as a linear combination of characteristic functions,

$$f(x,y) = \sum_{j=1}^{J} K_j \cdot \mathbf{1}_{D_j}(x,y), \tag{16}$$

where J is a positive integer or $+\infty$, and, for each j, K_j is a complex constant and D_j is a bounded domain in $\mathbb{R}^2$. We will assume that all domains D_j lie within the unit square and their interiors are pairwise disjoint. We will denote the boundary of each domain D_j by Γ_j,

$$\Gamma_j \overset{\text{def}}{=} \partial D_j, \tag{17}$$

and we will assume that each Γ_j is a polygon. Indeed, for each j, let $\{\gamma_l^j:[0,1] \to \Gamma_j\}_{l=1}^{L_j}$ be the set of paths that form Γ_j. Then, our assumptions are

1. $\bar{D}_j \subset [0,1] \times [0,1]$ for all j,
2. $\mathrm{Int}(D_j) \cap \mathrm{Int}(D_k) = \varnothing$ for all distinct j and k,
3. Every path γ_l^j has the form

$$\gamma_l^j(t) = (\alpha_0 + at, b_0 + bt) \quad \text{for } t \in [0,1], \tag{18}$$

where a_0, b_0, a, and b are constants.

Note that since $\gamma_l^j([0,1])$ is a straight line segment inside the unit square, $|b| \le 1$.

We will repeatedly refer to the following special function $F_{m,n}:\mathbb{R}^2 \to \mathbb{C}^1$:

$$F_{m,n}(x,y) \overset{\text{def}}{=} \begin{cases} \dfrac{e^{-2\pi imx}e^{-2\pi iny}}{-2\pi im}, & \text{when } m \ne 0, \\ xe^{-2\pi iny}, & \text{when } m = 0. \end{cases} \tag{19}$$

For every j and l, by $\{(\chi_k^l, \psi_k^l)\}_{k=1}^{q_l}$ we denote the collection of points which is the image under γ_l^j of the Gaussian nodes $t_1, \ldots, t_{q_l}$ on the interval $[0,1]$ (see (12)),

$$(\chi_k^l, \psi_k^l) \overset{\text{def}}{=} \gamma_l^j(t_k) \tag{20}$$

for $k = 1, \ldots, q_l$. We denote the corresponding weights by $\omega_1^l, \ldots, \omega_{q_l}^l$, so that

$$\omega_k^l = \omega_k \tag{21}$$

for $k = 1, \ldots, q_l$, where ω_k is defined in (12) with $q = q_l$.

Let h_x and h_y be the sampling intervals of a uniform grid on $[0,1]^2$ along the x- and y-axes, respectively, and , for a positive real r, let $\lfloor r \rfloor$ denote the greatest integer less than r. For integers $i_1 = 1, \ldots, \lfloor h_x^{-1} \rfloor$ and $i_2 = 1, \ldots, \lfloor h_y^{-1} \rfloor$, by (x_{i_1}, y_{i_2}) we denote the elements of the uniform grid:

$$(x_{i_1}, y_{i_2}) \overset{\text{def}}{=} ((i_1 - 1)h_x, (i_2 - 1)h_y). \tag{22}$$

For any point $(\chi, \psi) \in [0,1]^2$ and integers $j_1, j_2 \in [1,p]$, we define $\bar{x}_{j_1}(\chi)$ by the formula

$$\bar{x}_{j_1}(\chi) \overset{\text{def}}{=} \begin{cases} \left(\left\lfloor \dfrac{\chi}{h_x} \right\rfloor - \dfrac{p}{2} + 1 + j_1\right) h_x & \text{for even } p, \\ \left(\left\lfloor \dfrac{\chi}{h_x} + \dfrac{1}{2} \right\rfloor - \dfrac{p-1}{2} + j_1\right) h_x & \text{for odd, } p, \end{cases} \tag{23}$$

and we define $\bar{y}_{j_2}(\psi)$ by

$$\bar{y}_{j_2}(\psi) \overset{\text{def}}{=} \begin{cases} \left(\left\lfloor \dfrac{\psi}{h_y} \right\rfloor - \dfrac{p}{2} + 1 + j_2\right) h_y & \text{for even } p, \\ \left(\left\lfloor \dfrac{\psi}{h_y} + \dfrac{1}{2} \right\rfloor - \dfrac{p-1}{2} + j_2\right) h_y & \text{for odd, } p, \end{cases} \tag{24}$$

We define the function $W_{(\chi,\psi)}:\{1,\ldots,p\} \times \{1,\ldots,p\} \to \mathbb{Z}^2$ by

$$W_{(\chi,\psi)}(j_1, j_2) \overset{\text{def}}{=} \left(1 + \frac{\bar{x}_{j_1}(\chi)}{h_x}, 1 + \frac{\bar{y}_{j_2}(\psi)}{h_y}\right). \tag{25}$$

For any $(\chi, \psi) \in [0,1]^2$ and integers $j_1, j_2 \in [1,p]$, we define the function $\Delta_{(\chi,\psi)}:\mathbb{Z}^2 \to \mathbb{C}^1$ by

$$\Delta_{(\chi,\psi)}(i_1, i_2) \overset{\text{def}}{=} \begin{cases} \delta_{j_1}(\chi) \cdot \delta_{j_2}(\psi), & \text{when } (i_1, i_2) = W_{(\chi,\psi)}(j_1, j_2), \\ 0, & \text{otherwise.} \end{cases} \tag{26}$$

We will make use of the function $G:\mathbb{Z}^2 \to \mathbb{C}^1$, defined by

$$G(i_1, i_2) \overset{\text{def}}{=} \sum_{j=1}^{J} \sum_{l=1}^{L_j} \sum_{k=1}^{q_l} K_j \cdot b \cdot \omega_k^l \cdot \Delta_{(\chi_k^l, \psi_k^l)}(i_1, i_2) \tag{27}$$

with K_j defined in (16), b defined in (18), ω_k^l defined in (21), (χ_k^l, ψ_k^l) defined in (20), and $\Delta_{(\chi_k^l, \psi_k^l)}(i_1, i_2)$ defined in (26).

Remark 4.1. It is easy to see that for any point $(\chi, \psi) \in [0, 1]^2$, the points

$$(\bar{x}_{j_1}(\chi), \bar{y}_{j_2}(\psi)), \quad j_1 = 1, 2, ..., p, j_2 = 1, 2, ..., p, \tag{28}$$

are the p^2 nodes on the uniform grid satisfying condition (10) with respect to the point (χ, ψ). We will be interpolating functions of the form (19) from the uniform grid onto arbitrary points (χ, ψ) on the square $[0, 1]^2$; in fact, we will be interpolating such functions from the p^2 nodes (28) onto the point (χ, ψ), and we will use the Lagrange interpolation formula (8). Obviously, the interpolation coefficient corresponding to the point $W_{(\chi,\psi)}(j_1, j_2)$ is $\delta_{j_1}(\chi) \cdot \delta_{j_2}(\psi)$ (see (5), (8).) Thus, the function $W_{(\chi,\psi)}$: $\{1, ..., p\} \times \{1, ..., p\} \to \mathbb{Z}^2$ assigns to a pair of integers j_1, $j_2 \in [1, p]$ the location of the interpolation node on the uniform grid indexed by (i_1, i_2), while the function $\Delta_{(\chi,\psi)} : \mathbb{Z}^2 \to \mathbb{C}^1$ provides the interpolation coefficient corresponding to that node.

5. DESCRIPTION OF THE ALGORITHM

5.1. *Informal Description of the Algorithm*

We wish to compute the Fourier transform

$$\hat{f}(m, n) = \int_0^1 \int_0^1 f(x, y) e^{-2\pi i m x} e^{-2\pi i n y} \, dy \, dx \tag{29}$$

of a piecewise constant function f of the form (16) under assumptions 1–3 of Section 4.

Substituting (16) into (29), and using (3), we get

$$\begin{aligned} \hat{f}(m, n) &= \int_0^1 \int_0^1 \sum_{j=1}^{J} K_j \cdot \mathbf{1}_{D_j}(x, y) e^{-2\pi i m x} e^{-2\pi i n y} dy \, dx \\ &= \sum_{j=1}^{J} \int_{D_j} K_j \cdot e^{-2\pi i m x} e^{-2\pi i n y} dy \, dx \\ &= \sum_{j=1}^{J} \int_{\Gamma_j} K_j \cdot F_{m,n}(x, y) \, dy, \end{aligned} \tag{30}$$

with K_j, D_j defined in (16), Γ_j defined in (17), and $F_{m,n}$ defined in (19).

Let $\{\gamma_l^j\}_{l=1}^{L_j}$ be the set of paths that form Γ_j, as defined in (18). Along each γ_l^j we approximate the line integrals in (30) via Gaussian quadratures. Thus, using approximation (12) along each γ_l^j and substituting the result into (30), we get

$$\begin{aligned} \hat{f}(m, n) &= \sum_{j=1}^{J} \sum_{l=1}^{L_j} \int_0^1 K_j \cdot (F_{m,n} \circ \gamma_l^j)(t) \cdot b \, dt \\ &\approx \sum_{j=1}^{J} \sum_{l=1}^{L_j} \sum_{k=1}^{q_l} K_j \cdot F_{m,n}(\chi_k^l, \psi_k^l) \cdot b \cdot \omega_k^l, \end{aligned} \tag{31}$$

with K_j defined in (16), $F_{m,n}$ defined in (19), (χ_k^l, ψ_k^l) defined in (20), ω_k^l defined in (21), and b defined in (18).

Next, we redistribute the weights from the Gaussian nodes (20) to the uniform grid with the help of the Lagrange interpolation formula (8) applied to the function $F_{m,n}$ at $(x, y) = (\chi_k^l, \psi_k^l)$ for every k and l. The final approximation to $\hat{f}(m, n)$ becomes

$$\begin{aligned} \hat{f}(m, n) \approx \sum_{j=1}^{J} \sum_{l=1}^{L_j} \sum_{k=1}^{q_l} K_j \cdot b \cdot \omega_k^l \sum_{j_1=1}^{p} \sum_{j_2=1}^{p} \delta_{j_1}(\chi_k^l) \\ \cdot \delta_{j_2}(\psi_k^l) \cdot F_{m,n}(\bar{x}_{j_1}(\chi_k^l), \bar{y}_{j_2}(\psi_k^l)), \end{aligned} \tag{32}$$

with δ_{j_1} and δ_{j_2} defined in (5), $\bar{x}_{j_1}(\chi_k^l)$ defined in (23), and $\bar{y}_{j_2}(\psi_k^l)$ defined in (24).

Changing the order of summation in (32), and using (26) and (27), we see that

$$\begin{aligned} &\hat{f}(m, n) \\ &\approx \sum_{i_1=1}^{\lfloor h_x^{-1} \rfloor} \sum_{i_2=1}^{\lfloor h_y^{-1} \rfloor} \left(\sum_{j=1}^{J} \sum_{l=1}^{L_j} \sum_{k=1}^{q_l} K_j \cdot b \cdot \omega_k^l \right. \\ &\qquad \left. \cdot \Delta_{(\chi_k^l, \psi_k^l)}(i_1, i_2) \right) \cdot F_{m,n}(x_{i_1}, y_{i_2}) \\ &= \sum_{i_1=1}^{\lfloor h_x^{-1} \rfloor} \sum_{i_2=1}^{\lfloor h_y^{-1} \rfloor} G(i_1, i_2) \cdot F_{m,n}(x_{i_1}, y_{i_2}) \\ &= \begin{cases} \dfrac{1}{-2\pi i m} \displaystyle\sum_{i_1=1}^{\lfloor h_x^{-1} \rfloor} \sum_{i_2=1}^{\lfloor h_y^{-1} \rfloor} G(i_1, i_2) \cdot e^{-2\pi i m x_{i_1}} e^{-2\pi i n y_{i_2}} & \text{when } m \neq 0, \\ \displaystyle\sum_{i_2=1}^{\lfloor h_y^{-1} \rfloor} \left(\sum_{i_1=1}^{\lfloor h_x^{-1} \rfloor} x_{i_1} \cdot G(i_1, i_2) \right) e^{-2\pi i n y_{i_2}} & \text{when } m = 0, \end{cases} \end{aligned} \tag{33}$$

which can be readily computed for all m and n with the help of two- and one-dimensional FFTs.

Remark 5.1. The function G, defined in (27), can be viewed as a weak approximation to the function f, defined in (16). In other words, while G is a very poor approximation to f itself, the Fourier transform of G (in the sense of (33)) makes an excellent approximation to $\hat{f}$. (See Theorem 1 below.)

5.2. *Error Estimates*

In this section we define the error of the computation, E_∞, by the formula

$$E_\infty \overset{\text{def}}{=} \max_{\substack{-M<m\le m\\ -N<n\le N}} \left| \hat{f}(m,n) - \sum_{i_1=1}^{\lfloor h_x^{-1}\rfloor} \sum_{i_2=1}^{\lfloor h_y^{-1}\rfloor} G(i_1,i_2)\cdot F_{m,n}(x_{i_1}, y_{i_2}) \right| \quad (34)$$

and estimate the accuracy of the approximations (31) and (32) made in Section 5.1.

Let γ_l^j be a straight path as defined in (18):

$$\gamma_l^j(t) = (a_0 + at, b_0 + bt) \quad \text{for t} \in [0,1].$$

Then,

$$\int_{\gamma_l^j} F_{m,n}\, dy = \int_0^1 (F_{m,n} \circ \gamma_l^j)(t)\cdot b\, dt. \quad (35)$$

Combining (35) with (12), (18), and (20), we see that the error made by the Gaussian quadrature of the line integral of $F_{m,n}$ along the path γ_l^j equals

$$\int_0^1 (F_{m,n}\circ\gamma_l^j)(t)\cdot b\,dt - \sum_{k=1}^{q_l} F_{m,n}(\chi_k^l, \psi_k^l)\cdot b\cdot \omega_k^l = b\cdot E_{q_l}(F_{m,n}\circ\gamma_l^j), \quad (36)$$

with E_{q_l} defined by (12). The following lemma provides a bound on this error.

LEMMA 5.1. *For all (m, n) such that $-M < m \le M$ and $-N < n \le N$,*

$$|E_{q_l}(F_{m,n}\circ\gamma_l^j)| \le \frac{(2\pi)^{2q_l}(q_l!)^4 q_l}{(2q_l+1)[(2q_l)!]^3} (\sqrt{M^2+N^2}\cdot\sqrt{a^2+b^2})^{2q_l}. \quad (37)$$

Proof. We are going to estimate E_{q_l} with the help of (13). To that end we calculate a bound on the $(2q_l)$th derivative of $(F_{m,n}\circ\gamma_l^j)(t)$. The path γ_l^j, defined in (18), has the form $\gamma_l^j(t) = (a_0 + at, b_0 + bt)$. Thus,

$$(F_{m,n}\circ\gamma_l^j)(t) = \begin{cases} \dfrac{e^{-2\pi i(ma_0+nb_0)}e^{-2\pi i(ma+nb)t}}{-2\pi i m} & \text{when } m \ne 0,\\ (a_0+at)\cdot e^{-2\pi i n b_0} e^{-2\pi i n b t} & \text{when } m = 0, \end{cases} \quad (38)$$

and, consequently,

$$\frac{d^{2q_l}}{d^{2q_l}t}(F_{m,n}\circ\gamma_l^j)(t) = \frac{[-2\pi i(ma+nb)]^{2q_l}\cdot e^{-2\pi i(ma_0+nb_0)}e^{-2\pi i(ma+nb)t}}{-2\pi i m} \quad (39)$$

when $m \ne 0$, and

$$\frac{d^{2q_l}}{d^{2q_l}t}(F_{m,n}\circ\gamma_l^j)(t) = [(-2\pi i n b)^{2q_l} + a\cdot 2q_l\cdot(-2\pi i n b)^{2q_l-1}]e^{-2\pi i n b_0}e^{-2\pi i n b t} \quad (40)$$

when $m = 0$.

Let

$$\overline{M} \overset{\text{def}}{=} (M^2+N^2)^{1/2}, \quad \overline{L} \overset{\text{def}}{=} (a^2+b^2)^{1/2}. \quad (41)$$

Then, it follows from (39) that for $m \ne 0$,

$$\left|\frac{d^{2q_l}}{d^{2q_l}t}(F_{m,n}\circ\gamma_l^j)(t)\right| \le \frac{(2\pi|ma+nb|)^{2q_l}}{2\pi|m|} \le \frac{(2\pi\sqrt{m^2+n^2}\cdot\sqrt{a^2+b^2})^{2q_l}}{2\pi|m|} \le \frac{(2\pi\overline{M}\overline{L})^{2q_l}}{2\pi|m|}, \quad (42)$$

and, it follows from (40) that for $m = 0$,

$$\begin{aligned} \left|\frac{d^{2q_l}}{d^{2q_l}t}(F_{m,n}\circ\gamma_l^j)(t)\right| &\le |(-2\pi i n b)^{2q_l} + a\cdot 2q_l\cdot(-2\pi i n b)^{2q_l-1}| \\ &\le |2\pi n b|^{2q_l-1}\cdot|-2\pi i n b + 2q_l\cdot a| \\ &\le |2\pi n b|^{2q_l-1}\cdot((2\pi n)^2+(2q_l)^2)^{1/2}\cdot(a^2+b^2)^{1/2} \quad (43)\\ &\le (2\pi\overline{M}\overline{L})^{2q_l-1}\cdot((2\pi\overline{M})^2+(2q_l)^2)^{1/2}\cdot\overline{L} \\ &= (2\pi\overline{M}\overline{L})^{2q_l}\cdot\left(1+\left(\frac{q_l}{\pi\overline{M}}\right)^2\right)^{1/2} \\ &\le (2\pi\overline{M}\overline{L})^{2q_l}\cdot q_l. \end{aligned}$$

Since the bound in (43) is always larger than the one in (42), putting (13), (42), and (43) together, we get:

$$|E_{q_l}(F_{m,n}\circ\gamma_l^j)| \le \frac{(q_l!)^4}{(2q_l+1)[(2q_l)!]^3}(2\pi\overline{M}\overline{L})^{2q_l}\cdot q_l, \quad (44)$$

which is exactly the claim of the lemma. ▌

COROLLARY 5.1. *Suppose that $\gamma: [0,1] \to \mathbb{R}^2$ is a straight path defined by (18), $F_{m,n}$ is defined by (19) with $-M < m \le M$ and $-N < n \le N$, and integer $q \ge 1$ and real $\varepsilon > 0$ are such that*

$$q > \max\left\{4\overline{M}\overline{L}, \log\frac{1}{\varepsilon}\right\}, \tag{45}$$

with $\overline{M}$ and $\overline{L}$ defined in (41). *Then,*

$$\left|\int_\gamma F_{m,n}\,dy - \sum_{k=1}^{q} F_{m,n}(\gamma(t_k))\cdot b\cdot\omega_k\right| < b\cdot\varepsilon \le \varepsilon. \tag{46}$$

Here t_k and ω_k are the Gaussian nodes and weights on [0, 1], *respectively.*

Proof. Using Stirling's approximation to $q!$, we see that the right-hand side of (37) is not greater than

$$\frac{\sqrt{\pi q}}{4}\left(\frac{e\pi\overline{M}\overline{L}}{4q}\right)^{2q} \le \left(\frac{2.4\overline{M}\overline{L}}{q}\right)^{2q}. \tag{47}$$

Therefore, the inequality (46) will hold for any q such that

$$q > \max\left\{4\overline{M}\overline{L}, \log\frac{1}{\varepsilon}\right\}. \quad \blacksquare \tag{48}$$

Remark 5.2. The quantity $\overline{M}$ in (41) can be viewed as the frequency of the function $F_{M,N}$ along γ, and $\overline{L}$ as the length of γ. Thus, Corollary 5.1 states that the number of Gaussian nodes needed to compute the line integral $\int_\gamma F_{M,N}\,dy$ with accuracy ε is proportional to the number of periods of $F_{M,N}$ along γ (so long as that number is greater than $\log(1/\varepsilon)$).

Approximation (32) of Section 5.1 was made with the help of the Lagrange interpolation formula applied to $F_{m,n}$:

$$\begin{aligned} F_{m,n}(\chi_k^l, \psi_k^l) = \sum_{j_1=1}^{p}\sum_{j_2=1}^{p} \delta_{j_1}(\chi_k^l)\cdot\delta_{j_2}(\psi_k^l) \\ \cdot F_{m,n}(\bar{x}_{j_1}(\chi_k^l), \bar{y}_{j_2}(\psi_k^l)) + R_p(\chi_k^l, \psi_k^l), \end{aligned} \tag{49}$$

with $R_p(\chi_k^l, \psi_k^l)$ defined in (8), $\bar{x}_{j_1}(\chi_k^l)$ defined in (23), and $\bar{y}_{j_2}(\psi_k^l)$ defined in (24). The following lemma, which is an immediate consequence of (9), provides a bound on $R_p(\chi_k^l, \psi_k^l)$.

Lemma 5.2. *For all points (χ, ψ) satisfying condition* (10) *and all integers (m, n) such that $-M < m \le M$ and $-N \le N$, the term $R_p(\chi, \psi)$ in* (49) *satisfies the inequality*

$$|R_p(\chi, \psi)| \le \frac{[\Gamma(p/2+1)]^2}{p!}\cdot[(2\pi h_x M)^p + (2\pi h_y N)^p] \tag{50}$$

$$\le \frac{\sqrt{\eta p}}{2^p}\cdot[(2\pi h_x M)^p + (2\pi h_y N)^p] \tag{51}$$

for all $p \ge 2$.

The following lemma is an immediate consequence of (51).

Lemma 5.3. *Suppose that under the conditions of the preceding lemma, $\nu \in \mathbb{R}^1$ is defined by the formula*

$$\nu = \nu(p, \varepsilon) \stackrel{\text{def}}{=} 3\varepsilon^{-1/p}, \tag{52}$$

with ε an arbitrary positive real number. Suppose further that h_x and h_y are defined by the formulae

$$\begin{aligned} h_x &= \frac{1}{2\nu M}, \\ h_y &= \frac{1}{2\nu N}. \end{aligned} \tag{53}$$

Then,

$$|R_p(\chi, \psi)| < \varepsilon \tag{54}$$

for all $p \ge 2$.

Remark 5.3. It is easy to see from (53) that ν is the ratio between the actual density of the nodes in each of the directions x, y and the density required by the Nyquist theorem. Thus, we will refer to ν as oversampling.

The following theorem provides the principal error estimate of this paper.

Theorem 1. *Suppose that $\le > 0$ is real, $p \ge 2$ is an integer, ν is defined by* (52), *and E_∞ is defined by* (34). *Then,*

$$E_\infty \le 2\varepsilon\sum_{j=1}^{J}|K_j|\cdot\|\Gamma_j\|, \tag{55}$$

where $\|\Gamma_j\|$ denotes the length of Γ_j, and K_j and Γ_j are defined in (16) *and* (17), *respectively.*

Proof. Substituting (49) into (36), we get

$$\begin{aligned} &\int_0^1 F_{m,n}\circ\gamma_l^j(t)\cdot b\,dt \\ &= \sum_{k=1}^{q_l}\sum_{j_1=1}^{p}\sum_{j_2=1}^{p}\delta_{j_1}(\chi_k^l)\cdot\delta_{j_2}(\psi_k^l)\cdot F_{m,n}(\bar{x}_{j_1}(\chi_k^l), \bar{y}_{j_2}(\psi_k^l)) \\ &\quad + b\cdot R_{q_l}(F_{m,n}\circ\gamma_l^j) + \sum_{k=1}^{q_l} b\cdot\omega_k^l\cdot R_p(\chi_k^l, \psi_k^l). \end{aligned} \tag{56}$$

Denoting by $\overline{R}_p$ the right-hand side of (51), by $\overline{E}_{q_l}$ the right-hand side of (37), and, observing that $\sum_{k=1}^{q_l}\omega_k^l = 1$, we see that the error of integrating along the path γ_l^j (see (56)) is

$$\begin{aligned} &b\cdot E_{q_l}(F_{m,n}\circ\gamma_l^j) + \sum_{k=1}^{q_l} b\cdot\omega_k^l\cdot R_p(\chi_k^l, \psi_k^l) \\ &\qquad \le b\overline{E}_{q_l} + b\overline{R}_p \le 2b\varepsilon \le 2\overline{L}\varepsilon, \end{aligned} \tag{57}$$

if q_l is chosen according to (45) and ν is chosen according to (52). ($\bar{L}$, defined in (41), is the length of γ_l^j.)

Summing up the errors in (57) over all paths γ_l^j that form the boundary Γ_j, we see that integration around each Γ_j results in an error not greater than

$$2\varepsilon \|\Gamma_j\|,$$

where $\|\Gamma_j\|$ denotes the length of Γ_j. The total error of the computation is, therefore, at most

$$2\varepsilon \sum_{j=1}^{J} |K_j| \cdot \|\Gamma_j\|. \quad ∎ \tag{58}$$

Remark 5.4. Theorem 1 provides us with a method of choosing parameters p and ν so that the error of the computation E_∞, defined in (34), will not exceed a given tolerance η. Indeed, given $\eta > 0$, we set

$$\varepsilon \overset{\text{def}}{=} \left(2 \sum_{j=1}^{J} |K_j| \cdot \|\Gamma_j\|\right)^{-1} \cdot \eta, \\ p \overset{\text{def}}{=} \log \frac{1}{\varepsilon}, \quad \nu = 3\varepsilon^{-1/p} = 3e, \tag{59}$$

and Theorem 1 assures us that $E_\infty < \eta$. An examination of Tables II–VII shows that the bound (55) is not optimal; indeed, a more involved analysis shows that ν can be significantly reduced.

The following lemma is an immediate consequence of the definitions (14), (18), (19) and of Eq. (30); it is used in Section 6 to test the numerical accuracy of our algorithms.

LEMMA 5.4. *Under the assumptions* 1–3 *of Section 4,*

$$\hat{f}(m, n) = \sum_{j=1}^{J} \sum_{l=1}^{L_j} K_j \int_{\gamma_l^j} F_{m,n}(x, y)\, dy, \tag{60}$$

where $\hat{f}(m, n)$ is defined in (14), *$F_{n,n}$ is defined in* (19), *and γ_l^j is defined in* (18). *Furthermore, for each $j = 1, \ldots, J$ and $l = 1, \ldots, L_j$,*

$$\int_{\gamma_l^j} F_{m,n}(x, y)\, dy = \frac{(b_1 - b_0) \cdot [e^{-2\pi i(ma_1+nb_1)} - e^{-2\pi i(ma_0+nb_0)}]}{(-4\pi^2 m^2) \cdot [ma_1 + nb_1 - ma_0 - nb_0]} \tag{61}$$

when $m \neq 0$,

$$\int_{\gamma_l^j} F_{m,n}(x, y)\, dy = \frac{a_0}{-2\pi i n} [e^{-2\pi i n b_1} - e^{-2\pi i n b_0}] + (a_1 - a_0)\left[\frac{e^{-2\pi i n b_1}}{-2\pi i n} + \frac{e^{-2\pi i n b_1} - e^{-2\pi i n b_0}}{4\pi^2 n^2 (b_1 - b_0)}\right] \tag{62}$$

when $m = 0$ and $n \neq 0$, and

$$\int_{\gamma_l^j} F_{m,n}(x, y)\, dy = \frac{1}{2}(b_1 - b_0)(a_0 + a_1) \tag{63}$$

when $m = n = 0$. Here, (a_0, b_0) and (a_1, b_1) are the endpoints of the path γ_l^j:

$$(a_0, b_0) \overset{\text{def}}{=} \gamma_l^j(0), \quad (a_1, b_1) \overset{\text{def}}{=} \gamma_l^j(1). \tag{64}$$

We will refer to the algorithm based on Lemma 5.4 as direct. The run times of the direct algorithm are listed in column 6 of Tables II–VII.

5.3. *Formal Description of the Algorithm*

Step 0: Initialization.

Comment: [Choose the accuracy of the computation ε and determine the degree of Lagrange interpolation p and the oversampling factor ν according to (59). Create the function $G(m, n)$, defined in (27), on the uniform grid on $[0, 1] \times [0, 1]$ with the sampling intervals $h_x = (2\nu M)^{-1}$ and $h_y = (2\nu N)^{-1}$. Create a function $G_0(n)$ which will be used in the computation of the Fourier transform for the case $m = 0$. Precomputer the denominators of Lagrange interpolation and the Gaussian nodes and weights on [0, 1].]

```
do $n = 1, \ldots, 2\nu N$
  $G_0(n) = 0$
  di $m = 1, \ldots, 2\nu M$
    $G(m, n) = 0$
  enddo
enddo
```

Step 1: Green's Theorem and Lagrange Interpolation.

Comment: [For each j, integrate the constant function K_j along each of the paths γ_l^j: $[0, 1] \to \Gamma_j$, defined in (18), and then redistribute the weight from each of the resulting Gaussian nodes to p^2 Lagrange nodes from the uniform grid as in (32). Note: the function $U_{(\chi,\psi)}$ below is the second coordinate of the function $W_{(\chi,\psi)}$, defined in (25).]

```
do $j = 1, \ldots, J$
  do $l = 1, \ldots, L_j$
    Using the precomputed nodes and weights on [0, 1]
    calculate the Gaussian nodes $\{(\chi_k^l, \psi_k^l)\}$ and the weights
    $\{\omega_k^l\}$ (defined in (20) and (21), respectively) need to com-
    pute the line integrals along $\gamma_l^j$.
    do $k = 1, \ldots, q_l$
      do $j_1 = 1, \ldots, p$ and $j_2 = 1, \ldots, p$

$G(W_{(\chi_k^l,\psi_k^l)}(j_1, j_2)) = G(W_{(\chi_k^l,\psi_k^l)}(j_1, j_2))$
        $+ K_j \cdot \delta_{j_1}(\chi_k^l) \cdot \delta_{j_2}(\psi_k^l) \cdot b \cdot \omega_k^l$
```

```
$G_0(U_{(\chi_k^l,\psi_k^l)}(j_1,j_2)) = G_0(U_{(\chi_k^l,\psi_k^l)}(j_1,j_2))$
        $+ K_j \cdot \chi_k^l \cdot \delta_{j_1}(\chi_k^l) \cdot \delta_{j_2}(\psi_k^l) \cdot b \cdot \omega_k^l$

      enddo
    enddo
  enddo
enddo
```

Step 2: Fourier Transform.
Comment: [Apply the two-dimensional FFT to the $2\nu M$ by $2\nu N$ array G and only keep the frequencies (m, n) with $-M < m \leq M$ and $-N < n \leq N$. Store the results in the array FG. Similarly, apply the one-dimensional FFT to the array G_0, keep only the frequencies n with $-N < n \leq N$, and store the result in the array FG_0.]

Step 3: Post-processing.
Comment: [For each frequency (m, n) with $m \neq 0$, divide its coefficient by $-2\pi i m$, which brings the computation in agreement with Eq. (33). The case $m = 0$ is already correct and needs no adjustment. After this step FG becomes the desired Fourier transform with the absolute error E_∞ satisfying (55).]

```
do $m = -M + 1, ..., M$ and n $= -N + 1, ..., N$;
with $m \neq 0$
  $FG(m, n) = FG(m, n)/(-2\pi i m)$
enddo
do $n = -N + 1, ..., N$
  $FG(0, n) = FG_0(n)$
enddo
```

Remark 5.5. In certain applications, for example in VLSI modeling, the boundaries of the domains D_j contain vertical and horizontal straightline segments. Contribution from the horizontal segments to (30) is zero, and if S is a vertical segment from (a_0, b_0) to (a_0, b_1), then for $n \neq 0$,

$$\int_S F_{m,n}(x, y)\, dy = \frac{F_{m,n}(a_0, b_1) - F_{m,n}(a_0, b_0)}{-2\pi i n}; \tag{65}$$

that is, computation of a line integral is replaced by evaluation of a sum of two terms of the same type as in the approximation (31). Obviously, this reduces the CPU time requirements of the algorithm; the effect of this reduction on the overall performance of the algorithm is discussed in Section 6.

5.4. *Complexity Analysis of the Algorithm*

Time complexity of the algorithm is summarized in Table I. The estimate for the total operation count is, therefore,

$$O(\nu^2MN + p^2N_g + \nu^2MN \log MN + \nu N \log N) = O(p^2N_g + \nu^2MN \log MN). \tag{66}$$

We now estimate the total number of Gaussian nodes N_g. For a given accuracy ε the number of Gaussian nodes needed for a single path γ_l^j, according to Corollary 5.1, is at most

$$\max\left\{4\overline{M}\overline{L}, \log\frac{1}{\varepsilon}\right\}, \tag{67}$$

where $\overline{M}$ and $\overline{L}$ are defined in (41) and discussed in Remark 5.2. Therefore, the number of nodes needed for each Γ_j is at most

$$\max\left\{4\overline{M} \cdot \|\Gamma_j\|, L_j \cdot \log\frac{1}{\varepsilon}\right\}, \tag{68}$$

where $\|\Gamma_j\|$ is the length of Γ_j, and L_j is the number of paths γ_l^j that form Γ_j. It is now obvious that the total number of Gaussian nodes N_g satisfies the inequality

$$N_g \leq \max\left\{4\overline{M} \cdot \sum_{j=1}^{J} \|\Gamma_j\|, \left(\sum_{j=1}^{J} L_j\right) \cdot \log\frac{1}{\varepsilon}\right\} = O\left(\overline{M} \cdot P + \left(\log\frac{1}{\varepsilon}\right) \cdot V\right), \tag{69}$$

where P is the total perimeter of the domains D_j, and $V = \sum_{j=1}^{J} L_j$ is the total number of the straight paths γ_l^j, or, equivalently, the total number of vertices.

Choosing p and ν according to (59), and putting (66) and (69) together, we see that the time complexity of the algorithm is not greater than

$$O(p^2N_g + \nu^2MN \log MN) = O\left(\overline{M}^2 \log \overline{M} + \left(\log^2\frac{1}{\varepsilon}\right) P\overline{M} + \left(\log^3\frac{1}{\varepsilon}\right) V\right). \tag{70}$$

6. NUMERICAL EXPERIMENTS

We implemented the algorithm of Section 5 both with and without the use of the Green's theorem as described in Section 2.1. The algorithm was implemented in FORTRAN 77 and the implementation was run on SPARCstation 2. All internal calculations were performed in double-precision arithmetic, but the parameter p was relaxed for single-precision experiments.

In the tables below we report the results of the numerical experiments with the following three algorithms. In Algorithm 1 we make use of the Green's theorem as described in Section 5.1 and, in addition, make use of the Remark 5.5 to integrate along the horizontal and vertical lines. Algorithm 2 is also based on the Green's theorem, but without the preferential treatment for horizontal and vertical lines. In Algorithm 3 the area integrals are treated directly without recourse to the Green's theorem.

By the error E_∞ in columns 7–9 of the Tables II–VII, we

TABLE I

Time Complexity of the Algorithm

Step	Operation count	Explanation
0	$O(\nu^2 MN)$	Initialization of all arrays
1	$O(p^2 N_g)$	N_g is the total number of Gaussian nodes created. For each Gaussian node (χ_k^l, ψ_k^l), p^2 Lagrange coefficients $\delta_{j_1}(\chi_k^l) \cdot \delta_{j_2}(\phi_k^l)$ are computed. Since their denominators (see (5)) are precomputed, all p^2 Lagrange coefficients can be calculated in $O(p^2)$ operations
2	$O(\nu^2 MN \log MN + \nu N \log N)$	There is one FFT of a $2\nu M$ by $2\nu N$ array and one FFT of an array fo length $2\nu N$
3	$O(MN)$	For each frequency (m, n), multiplication of its Fourier coefficient by $(-2\pi i m)^{-1}$

mean the error E_∞, defined in (34), with $\hat{f}(m, n)$ computed using the direct method of Lemma 5.4.

Remark 6.1. In the Examples 1–3 we include the performance of a straightforward application of the FFT. We do that for illustrative purposes only; this method cannot be used inpractice unless very crude accuracy can be tolerated, since for discontinuous data, the error of the FFT is on the order of $O(N^{-1})$ at every frequency. This large error is a consequence of the fact that the FFT of a periodic function is mathematically equivalent to evaluating every Fourier coefficient via the trapezoidal rule. Therefore, in order to achieve accuracy ε at any frequency with a straightforward application of the FFT to a discontinuous function we must apply the FFT to an array of size $1/\varepsilon$ by $1/\varepsilon$, regardless of how many frequencies we are actually interested in. The operation count of such a procedure would be on the order of $(1/\varepsilon^2) \log (1/\varepsilon)$, which is prohibitive even for moderate accuracy.

In all examples below $K_j = 1$ for all j. All reported run times include all initializations, but not input and output.

EXAMPLE 1. In Tables II and III we show the results of a computation of the Fourier transform of a single large rectangle (approximately 0.6 by 0.66) using Algorithms 1, 2, 3, the direct method (see Lemma 5.4), and the standard FFT.

EXAMPLE 2. In this example we compute the Fourier transform of 1215 rectangles. Their total area is 0.128 and the perimeter of their union is 60.0. The results of this example can be found in Tables IV and V.

EXAMPLE 3. In this example we calculate the Fourier transforms of a 40 by 50 μm piece of a VLSI mask (courtesy of Eytan Barouch). In most models these masks consist of a large number of simple geometric shapes that represent the integrated circuit. Here we compute the Fourier transform of 1215 small rectangles (the same ones as in Example 2) and 424 triangles. Their total area is 0.132 and the perimeter of their union is 58.9. Tables VI and VII contain the run times and errors for this example.

The following observations can be made based on the numerical experiments above:

1. The errors for all three algorithms are essentially independent of the size of the problem.

2. All three algorithms have approximately the same accuracy.

3. The run times grow approximately like N^2 in Example 1 (in agreement with (70)), while the run times in Examples 2 and 3 appear to grow linearly. This apparent linear growth is the result of the large number N_g of Gauss–Legendre nodes generated, causing Lagrange interpolation to dominate the com-

TABLE II

Run Times and Errors for Example 1; Double Precision

	Double precision: $p = 36$ and $\nu = 3$								
	Run times (seconds)					Error E_∞			
N	Alg. 1	Alg. 2	Alg. 3	FFT	Direct	Alg. 1	Alg. 2	Alg. 3	FFT
32	1.3	5.6	328	0.3	0.6	9.1e-15	9.1e-15	6.9e-15	2.3e-02
64	5.6	14.8	1395	0.6	2.6	4.2e-15	5.1e-15	4.7e-15	9.2e-03
128	24.3	46.7	7037	2.2	10.4	3.1e-15	3.2e-15	2.8e-15	5.0e-03
256	110.7	155.1	28280	8.9	41.5	3.3e-15	3.4e-15	2.0e-15	2.5e-03

TABLE III

Run Times and Errors for Example 1; Single Precision

	Single precision: $p = 16$ and $\nu = 3$								
	Run times (seconds)					Error E_∞			
N	Alg. 1	Alg. 2	Alg. 3	FFT	Direct	Alg. 1	Alg. 2	Alg. 3	FFT
32	1.2	2.3	74	0.3	0.6	1.4e-08	1.4e-08	9.9e-09	2.3e-02
64	5.6	7.7	292	0.6	2.6	6.5e-09	7.8e-09	7.2e-09	9.2e-03
128	25.0	29.3	1191	2.2	10.4	4.7e-09	4.7e-09	4.2e-09	5.0e-03
256	114.0	125.0	6209	8.9	41.6	2.0e-09	2.7e-09	1.9e-09	2.5e-03

TABLE IV

Run Times and Errors for Example 2; Double Precision

	Double precision: $p = 36$ and $\nu = 3$								
	Run times (seconds)					Error E_∞			
N	Alg. 1	Alg. 2	Alg. 3	FFT	Direct	Alg. 1	Alg. 2	Alg. 3	FFT
32	18	254	920	3.4	566	1.0e-14	1.3e-14	5.7e-15	4.2e-02
64	24	401	1990	3.7	2263	1.1e-14	2.4e-14	1.0e-14	3.1e-02
128	46	738	5190	5.0	9054	1.1e-14	2.5e-14	4.5e-15	2.9e-02
256	130	1336	13944	12.2	36133	1.9e-14	1.5e-14	2.7e-15	1.7e-02

TABLE V

Run Times and Errors for Example 2; Single Precision

	Single precision: $p = 16$ and $\nu = 3$								
	Run times (seconds)					Error E_∞			
N	Alg. 1	Alg. 2	Alg. 3	FFT	Direct	Alg. 1	Alg. 2	Alg. 3	FFT
32	5.4	62	207	3.4	567	1.6e-08	2.4e-08	9.1e-09	4.2e-02
64	9.7	95	414	3.7	2262	1.7e-08	3.7e-08	1.6e-08	3.1e-02
128	29.7	161	875	5.0	9060	1.1e-08	3.9e-08	1.1e-08	2.9e-02
256	117.3	400	3083	12.2	36100	3.3e-09	2.5e-08	3.2e-09	1.7e-02

TABLE VI

Run Times and Errors for Example 3; Double Precision

	Double precision: $p = 36$ and $\nu = 3$								
	Run times (seconds)					Error E_∞			
N	Alg. 1	Alg. 2	Alg. 3	FFT	Direct	Alg. 1	Alg. 2	Alg. 3	FFT
32	32	238	1086	4.7	602	9.0e-15	6.7e-14	5.8e-15	4.1e-02
64	44	378	2259	5.0	2397	1.0e-14	6.7e-14	1.0e-14	3.1e-02
128	82	711	5742	6.3	9553	1.1e-14	6.7e-14	4.4e-15	3.3e-02
256	177	1285	14973	13.1	38216	1.9e-14	6.7e-14	2.1e-15	1.8e-02

TABLE VII

Run Times and Errors for Example 3; Single Precision

	Single precision: $p = 16$ and $\nu = 3$								
	Run times (seconds)					Error E_∞			
N	Alg. 1	Alg. 2	Alg. 3	FFT	Direct	Alg. 1	Alg. 2	Alg. 3	FFT
32	9	59	245	4.7	602	1.4e-08	2.4e-08	9.2e-09	4.1e-02
64	15	91	473	5.0	2397	1.6e-08	3.7e-08	1.6e-08	3.1e-02
128	38	157	974	6.3	9553	1.1e-08	3.9e-08	1.1e-08	3.3e-02
256	128	389	3307	13.1	38216	3.4e-09	2.5e-08	3.2e-09	1.8e-02

putation. Since N_g grows linearly with N, the total run times appear to grow linearly as well.

4. Algorithm 2 is at least four times faster than Algorithm 3, even when the ratio of perimeter to area is large and Algorithm 1 is about three times as fast as Algorithm 2 on realistic problems (Examples 2 and 3). On the same realistic problems all three algorithms are faster (for $N \gtrsim 64$) than the direct, closed-form solution (see Lemma 5.4) even when it is available.

5. In Examples 2, 3, the errors for the straightforward FFT-based algorithm appear to decay slower than N^{-1}. The reason for this anomaly is that at the numbers of nodes in the grids used (no more that 512×512) many of the domains D_j (see (16)) are not resolved. Thus, the scheme has not entered its asymptotic regime and is displaying transient behavior. Clearly, this problem does not occur for Algorithms 1–3.

7. CONCLUSIONS

We have presented an algorithm for the evaluation of the Fourier transform of piecewise constant functions of two variables. The algorithm overcomes the accuracy problems associated with computing the Fourier transform of discontinuous functions; in fact, its time complexity is $O(N^2 \log N + NP \log^2(1/\varepsilon) + C \log^3(1/\varepsilon))$, where ε is the accuracy, N is the size of the problem, P is the perimeter of the set of discontinuities, and V is its number of vertices.

The algorithm is based on the Lagrange interpolation formula and the Green's theorem, which are used to preprocess the data before applying the FFT. It admits natural generalizations to higher dimensions and to piecewise smooth functions; this work is in progress and will be reported at a later date.

ACKNOWLEDGMENTS

The author thanks Professors Ronald Coifman and Vladimir Rokhlin for several useful discussions. He also thanks Professor Eytan Barouch for suggesting the problem and supplying the VLSI masks for the experiments.

REFERENCES

1. M. Abramowitz and I. Stegun, *Handbook of Mathematical Functions* (Dover, New York, 1970).
2. E. O. Brigham, *The Fast Fourier Transform and its Applications* (Prentice–Hall, Englewood Cliffs, NJ, 1988).
3. J. W. Cooley and J. W. Tukey, *Math. Comput.* **19** (1965).
4. G. Dahlquist and Å. Björck, *Numerical Methods* (Prentice–Hall, Englewood Cliffs, NJ, 1974).
5. D. Gottlieb, M. Y. Hussaini, and S. Orszag, in *Spectral Methods for Partial Differential Equations,* edited by R. G. Voigt, D. Gottlieb, and M. Y. Hussaini (SIAM, Philadelphia, 1984).
6. I. S. Gradshteyn and I. M. Ryzhik, *Table of Integrals, Series, and Products* (Academic Press, New York, 1980).
7. J. Stoer and R. Bulirsch, *Introduction to Numerical Analysis* (Springer-Verlag, New York, 1980).
8. H. J. Weaver, *Theory of Discrete and Continuous Fourier Analysis* (Wiley, New York, 1989).