Applied Mathematics and Computation 221 (2013) 691–702

Contents lists available at SciVerse ScienceDirect

Applied Mathematics and Computation

journal homepage: www.elsevier.com/locate/amc

Efficient Filon method for oscillatory integrals

L.F. Shampine

Department of Mathematics, Southern Methodist University, United States

ARTICLE INFO

Keywords:
Quadrature
Oscillatory
Filon
Matlab

ABSTRACT

When ω is large, the integrand of $\int_a^b f(x)\, e^{i\omega x}\, dx$ is highly oscillatory and conventional quadrature programs are ineffective. The few items of software available for the task implement Filon methods. A new program of this kind is developed here that is efficient and very easy to use. It is based on mathematical results and algorithms that are especially well-suited to Matlab.

1. Introduction

Integrals of the form

$$I(f) = \int_a^b f(x)\, e^{i\omega x}\, dx \tag{1}$$

with real $f(x)$ and finite interval $[a, b]$ arise in a number of applications. When the real parameter ω is large, the rapid oscillation of the integrand makes the integral difficult, if not impossible, for standard quadrature programs. An example of this is found in Section 5. The more general problem

$$I(f, g) = \int_a^b f(x)\, e^{i\omega g(x)}\, dx \tag{2}$$

is much more difficult, mainly because critical points where $g'(x) = 0$ affect the behavior of the integral as $\omega \to \infty$. Integrals (1) are often called *regular* oscillations to contrast them with the *irregular* oscillations of (2). Evans and Webster [1] survey many schemes for the approximation of these integrals. Asymptotic and steepest descent methods [2] have been used classically to approximate the integrals for "large" ω and more recently the viewpoint has been useful in developing more broadly applicable methods [3]. This paper is devoted to developing software for approximating regular oscillations that takes full advantage of the popular matlab [4] problem-solving environment (PSE). For the convenience of users and breadth of application, we consider only methods that are accurate for both small and large ω, do not require smoothness of the integrand, and do not require users to supply derivatives. In a seminal paper Filon [5] presents a scheme for regular oscillations that is applicable to functions that are not smooth and produces approximations that are accurate for all ω. He approximates $f(x)$ by an interpolating quadratic polynomial, which others later generalized to a polynomial $p(x)$ interpolating $f(x)$ at nodes $x_1, \ldots, x_N$. After approximating $f(x)$ by $p(x)$, Filon's idea is to form *analytically*

$$Q(f) = \int_a^b p(x)\, e^{i\omega x}\, dx \tag{3}$$

as an approximation to $I(f)$. The various methods of this kind differ in the choice of nodes, how the interpolating polynomial is constructed, and how (3) is evaluated analytically. Nearly all are based on the Clenshaw–Curtis (CC) nodes or some variant

∗ Address: 1204 Chesterton Dr., Richardson, TX 75080, United States.
E-mail address: lfshampine@aol.com

0096-3003/$ - see front matter
http://dx.doi.org/10.1016/j.amc.2013.06.085

thereof, but Bakhvalov and Vasil'eva [6] suggest using Legendre nodes. Evans [7] generalizes the Filon approach to irregular oscillations. Recently Shampine [8] further develops the approach and presents a MATLAB program `quadgF`. Approximating (2) is quite different from approximating (1), so we say no more in this paper about irregular oscillations except in contrast with regular oscillations.

The only *software* for approximating $I(f)$ known to the author are the FSER1 program of Chase and Fosdick [9,10] which implements the original Filon approach of interpolating quadratic polynomials, the program AINOS of Piessens and Branders [11] which interpolates at 13 Clenshaw–Curtis (CC) nodes, and the `osc` program [12] of Shampine which uses a smooth interpolating cubic spline. In this paper we consider the efficient MATLAB implementation of a method based on a polynomial interpolating at Legendre nodes. In Section 2.2 we extend the definition of degree of precision of a formula for a conventional integral to formulas for integrals of the form (1). In this new terminology the Fortran program AINOS is an adaptive implementation of a pair with generalized degrees of precision (8,12). We develop here a MATLAB program, `quadosc`, with a pair that is also of degrees (8,12). Though it provides only tools, the Clenshaw–Curtis rule package [13], which we call here "CC package", is written in MATLAB. It represents a different approach to approximating the integral of a regular oscillation. For example, the `CleCurExpRule` program evaluates an approximation for a number of CC nodes specified by the user. In this way *any* degree of precision is possible. On the other hand, the package does not contain software that computes an approximation to specified accuracy.

We begin our development of software for approximating $I(f)$ in Section 2 with some special, but important matters, namely the nature of the error control, problems with small ω, and $f(x)$ with modest end point singularities. In a certain sense the Chebyshev nodes are optimal for approximating $f(x)$ in the maximum norm. In Section 3.1 we review this standard result and then prove that the Legendre nodes are optimal for least-squares approximation. In Section 3.2 we show how to form the polynomial interpolating at Legendre nodes in a way that is both very simple and fast in MATLAB. A virtue of the Legendre nodes is that (3) has a simple analytical expression. None of the methods discussed in [1] use this expression, including Bakhvalov and Vasil'eva [6], but it makes possible a simple and fast algorithm in MATLAB that we study in Section 3.3. The natural representation of the polynomial interpolating at Legendre nodes also makes possible a simple and effective way of estimating error that we explain in Section 3.4. Section 4 considers how to use the tools developed in previous sections to obtain an efficient and effective adaptive implementation. In this we recognize that vectorization and the use of built-in functions is crucial to speed in MATLAB. The AINOS and `osc` programs are adaptive implementations of Filon methods that process only one subinterval at a time. The new program `quadosc` processes all subintervals at once, which is very advantageous in this PSE. The numerical examples of Sections 2.3 and 5 show that `quadosc` is effective and very easy to use.

2. Preparation

The `quadosc` program uses the MATLAB program `quadgk` [14] to deal with non-oscillatory problems and problems with modest singularities at end points. Here we explain why and how this is done. In this we must discuss the unconventional error control of the program. We also extend the concept of degree of precision of a formula for approximating a conventional integral to a formula for approximating an integral of the form (1).

2.1. Error control

Like the `osc` [12] program, we have implemented a scaled absolute error control in `quadosc`. An integration by parts shows that $I(f)$ is ordinarily $O(1/\omega)$ as $|\omega| \to \infty$. We aim to approximate the integral for large ω, so we recognize the qualitative behavior of the integral by measuring the error relative to $1/|\omega|$. This is a kind of relative error, so we must limit the tolerance in recognition of both the order of the method and the precision used in the computations. The error control also needs to provide for small values of ω. Accordingly, for a tolerance TOL, the approximation Q computed in `quadosc` is to satisfy

$$|Q - I| \leqslant \frac{\mathtt{TOL}}{\max(|\omega|, 1)}. \tag{4}$$

The default tolerance is 10^{-5}. If $|\omega| > 1$, any TOL less than 10^{-10} is increased to 10^{-10}. Roughly speaking, this means that by default the program tries to get 5 correct digits, but it will not try to get more than 10.

2.2. Non-oscillatory

When $\omega = 0$ in (1), we have a conventional quadrature problem. The `quadgk` [14] program exploits fully the MATLAB PSE to solve such problems in a very efficient and robust way. A formula for approximating $\int_a^b p(x)\,dx$ is said to be of degree of precision d if it integrates exactly any polynomial $p(x)$ of degree d. It is a very important measure of the accuracy of a formula. We extend the terminology to oscillatory problems (1) and say that *a formula is of degree of precision d if it integrates exactly* $\int_a^b p(x)\,e^{i\omega x}\,dx$ *for any polynomial* $p(x)$ *of degree d and any real* ω. This is a natural concept for discussing generalized Filon methods because they approximate (1) with an integral of this kind, namely (3), and integrate it analytically.