Roundoff errors in the problem of computing
Cauchy principal value integrals

Pawel Keller
Institute of Computer Science, University of Wroclaw

Abstract

We show that, when handled properly, the Cauchy principal value
integral \( \int_a^b f(x)(x - \tau)^{-1} \, dx \) \((a < \tau < b)\) can be computed very easily
and accurately using any reliable adaptive quadrature. We give de-
tailed estimations of the roundoff errors in the case the function \( f \) has
bounded first derivative in the interval \([a, b] \).

Keywords: Cauchy principal value, roundoff errors, numerical integra-
tion, adaptive quadrature

Mathematics Subject Classification: 65D30, 30E20

1 Introduction

We consider the problem of numerical evaluation of the Cauchy principal
value integral

\[
I_{\tau,a,b}(f) = \int_a^b \frac{f(x)}{x - \tau} \, dx = \lim_{\mu \to 0^+} \left( \int_a^{\tau - \mu} + \int_{\tau + \mu}^b \right) \frac{f(x)}{x - \tau} \, dx,
\]

where \( \tau \in (a, b) \), and the function \( f \) has bounded first derivative. In general,
the integral (1.1) exists if \( f \) is Hölder continuous. The integrals of the
type (1.1) appear in many practical problems related to aerodynamics, wave
propagation or fluid and fracture mechanics, mostly with relation to solving
singular integral equations.

A great many papers related to numerical evaluation of the integrals of
the form (1.1) have been published so far. Some of them are [2, 3, 4, 6, 7, 8].
A nice survey on the subject, along with a large number of references, is
presented in [1, §2.12.8].

Even though so many algorithms have been proposed, the subroutines
for computing the integrals of the type (1.1) are not commonly available in
systems for scientific computations. This is probably because most of the methods assume some properties of the function $f$, e.g., that $f$ is analytic, $f$ is not rapidly oscillating etc. On the other hand, almost every system is equipped with one or more subroutines for automatic computation of the integrals

$$
\int_{a}^{b} f(x) \, dx.
$$

These algorithms are usually based on the so-called adaptive quadratures and they can compute the integrals of the form (1.2) for a very wide range of integrands. The natural question arises: can an adaptive quadrature be used for computing the integral (1.1)?

In this paper, we give the positive answer to that question. The next section contains the formulation of our algorithm. We apply two known analytical transformations that convert the integral (1.1) into the sum of two nonsingular integrals. In Section 3, we discuss the problem of loss of significant digits that theoretically may appear when computing the transformed integral. In Section 4, we present some numerical examples to validate the usefulness of the proposed method.

## 2 Analitycal tranformations

Without the loss of generality, we may restrict our attention to the case $a = -1$ and $b = 1$. The computation of the Cauchy principal value integral

$$
I_{\tau}(f) \equiv I_{\tau,-1,1}(f) = \int_{-1}^{1} \frac{f(x)}{x-\tau} \, dx
$$

may at first seem quite easy, if we observe that by a simple change of variables, for $\delta = \min\{1+\tau, 1-\tau\}$, we obtain

$$
I_{\tau}(f) = \int_{|x-\tau| \geq \delta} \frac{f(x)}{x-\tau} \, dx + \int_{\tau-\delta}^{\tau+\delta} \frac{f(x)}{x-\tau} \, dx
$$

$$
= \int_{|x-\tau| \geq \delta} \frac{f(x)}{x-\tau} \, dx + \int_{0}^{\delta} \frac{f(\tau + x) - f(\tau - x)}{x} \, dx,
$$

(2.2)

where we use the convention that $\int_{|x-\tau| \geq \delta} \equiv \int_{\tau+\delta}^{1}$, if $\delta = 1+\tau$, and $\int_{|x-\tau| \geq \delta} \equiv \int_{-1}^{\tau-\delta}$, if $\delta = 1-\tau$.

The formula (2.2) was applied for the first time by Longman in [4], and it was derived by splitting the function $f$ into the odd and even parts.
Both integrals on the right hand side of (2.2) exist in the Riemann sense. We should note that if the function \( f \) has bounded first derivative in the neighbourhood of \( \tau \), then this integral is not even singular.

The first integral on the right hand side of (2.2) was commonly ignored, as it is always a proper one. However, if \( \tau \) is close to \(-1\) or \(1\), then this integral is near singular and standard quadratures may fail when applied directly.

In many algorithms, another transformation of the integral \( I_\tau(f) \) is used, usually being called subtracting out the singularity. We have

\[
I_\tau(f) = \int_{-1}^{1} \frac{f(x)}{x-\tau} dx + \int_{1}^{\tau} \frac{f(x) - f(\tau)}{x-\tau} dx \\
= f(\tau) \log \frac{1-\tau}{1+\tau} + \int_{1}^{\tau} \frac{f(x) - f(\tau)}{x-\tau} dx.
\] (2.3)

A direct application of the above formula is commonly not recommended (see, e.g., [3, 5]) due to possible severe cancellation, if a quadrature node happens to be very close to \( \tau \).

The author, however, has never seen the two above approaches being put together. From (2.2) and (2.3) we immediately obtain

\[
I_\tau(f) = f(\tau) \log \frac{1-\tau}{1+\tau} + \int_{|x-\tau|\geq\delta} g(x) dx + \int_{0}^{\delta} h(x) dx,
\] (2.4)

where

\[
g(x) = \frac{f(x) - f(\tau)}{x-\tau} \quad \text{and} \quad h(x) = \frac{f(\tau + x) - f(\tau - x)}{x}.
\]

If the function \( f \) has bounded first derivative, none of the integrals on the right hand side of (2.4) is singular or near singular (unless \( f \) itself has singularities just outside the interval \([-1,1]\)), and, when approximating these integrals numerically, the distance between \( \tau \) and any of the quadrature nodes is never smaller than \( \delta \).

### 3 Estimating roundoff errors

In this section, we will show that the formula (2.4) is numerically safe, i.e., we will prove that if the integral (2.1) is approximated numerically using the equality (2.4), then the absolute error of the approximation, which results from the roundoff errors, is practically very small.
Let us denote by $\varepsilon$ the precision of the arithmetic used, and by $g_\varepsilon(x)$ and $h_\varepsilon(x)$ the numerically computed values of the functions $g$ and $h$ at the point $x$. For a moment, we will assume that the values of the function $f$ are computed exactly.

**Lemma 1.** If we define $D_1 := \max_{|x| \leq 1} |f'(x)|$, then for all $|x| \leq \delta$

$|h_\varepsilon(x) - h(x)| \leq \frac{4\varepsilon D_1}{x} + 4\varepsilon D_1.$ \hspace{1cm} (3.1)

**Proof.** We have

$h_\varepsilon(x) = \frac{f((\tau(1+\gamma_1) + x(1+\gamma_2))(1+\gamma_3)) - f((\tau(1+\gamma_1) - x(1+\gamma_2))(1+\gamma_4))}{x(1+\gamma_5)}$, \hspace{1cm} (3.2)

where $|\gamma_1|, |\gamma_2|, |\gamma_3|, |\gamma_4| \leq \varepsilon$ and $|\gamma_5| \leq 2\varepsilon$. Further,

$$(\tau(1+\gamma_1) + x(1+\gamma_2))(1+\gamma_3) \approx \tau + x + (\gamma_1 + \gamma_3)\tau + (\gamma_2 + \gamma_3)x$$

$$= \tau + x + (\tau + x)\beta$$

for some $|\beta| \leq 2\varepsilon$, and, consequently,

$$|f((\tau(1+\gamma_1) + x(1+\gamma_2))(1+\gamma_3)) - f(\tau + x)| \lesssim |(\tau + x)\beta| D_1 \leq 2\varepsilon D_1, \hspace{1cm} (3.3)$$

as $|\tau + x| \leq 1$. Analogously,

$$|f((\tau(1+\gamma_1) - x(1+\gamma_2))(1+\gamma_4)) - f(\tau - x)| \lesssim 2\varepsilon D_1. \hspace{1cm} (3.4)$$

Now, from (3.2), (3.3) and (3.4) we obtain

$$|h_\varepsilon(x) - h(x)| = \left| h_\varepsilon(x) - \frac{(f(\tau + x) - f(\tau - x))(1+\gamma_5)}{x(1+\gamma_5)} \right|$$

$$= \left| h_\varepsilon(x) - \frac{(f(\tau + x) - f(\tau - x))}{x(1+\gamma_5)} - \frac{\gamma_5}{1+\gamma_5} \frac{(f(\tau + x) - f(\tau - x))}{x} \right|$$

$$\leq \frac{4\varepsilon D_1}{x} + 4\varepsilon D_1.$$

\hfill \Box

**Lemma 2.** If $|x - \tau| \leq 8\varepsilon$, then

$$|g_\varepsilon(x) - g(x)| \lesssim \frac{8\varepsilon D_1}{|x - \tau|}. \hspace{1cm} (3.5)$$
Proof. Similarly as in (3.2), we have
\[ g_\varepsilon(x) = \frac{f(x(1+\gamma_1)) - f(\tau(1+\gamma_2))}{(x(1+\gamma_1) - \tau(1+\gamma_2))(1+\gamma_3)}, \] (3.6)
where \(|\gamma_1|, |\gamma_2| \leq \varepsilon\) and \(|\gamma_3| \leq 2\varepsilon\). Moreover,
\[ \frac{(x(1+\gamma_1) - \tau(1+\gamma_2))(1+\gamma_3) - (x - \tau)}{x - \tau} = \frac{\beta}{x - \tau} \]
for some \(|\beta| \lesssim 3\varepsilon\), which implies that
\[ (x(1+\gamma_1) - \tau(1+\gamma_2))(1+\gamma_3) = (x - \tau)\left(1 + \frac{\beta}{x - \tau}\right). \]
In addition, as \(|x|, |\tau| \leq 1\), we have
\[ |f(x(1+\gamma_1)) - f(x)| \leq \varepsilon D_1 \quad \text{and} \quad |f(\tau(1+\gamma_2)) - f(\tau)| \leq \varepsilon D_1, \]
which together with (3.6) and \(|\beta| \lesssim 3\varepsilon\), \(|x - \tau| \leq \varepsilon\) gives
\[ |g_\varepsilon(x) - g(x)| = \left| \frac{f(x) - f(\tau)}{x - \tau} \left(1 + \frac{\beta}{x - \tau}\right) \right| \leq \frac{2\varepsilon D_1 + \beta D_1}{|x - \tau| \left(1 + \frac{\beta}{x - \tau}\right)} \lesssim \frac{8\varepsilon D_1}{|x - \tau|}. \]

In both lemmas we assumed that the values of the function \(f\) are computed exactly. In practice, this is obviously not true. However, if the computation of \(f(x)\) for every \(x \in [-1, 1]\) is numerically backward stable, then all the above results remain true, only the constant factors change.

3.1 The first approach – cutting of the singularity

From (3.1) and (3.3) we immediately obtain that
\[ E_\varepsilon(f) := \int_{|x-\tau|\geq\delta} |g_\varepsilon(x) - g(x)| \, dx + \int_0^\delta |h_\varepsilon(x) - h(x)| \, dx \]
\[ \lesssim \int_{-1}^1 \frac{8\varepsilon D_1}{|x - \tau|} \, dx \leq 16\varepsilon D_1 \int_0^1 \frac{1}{x} \, dx, \] (3.7)
which may not look like a good approximation of the cumulated roundoff errors. Thus, it seems quite natural to replace the last integral in (2.4) by
\[ \int_{\mu}^{\delta} h(x) \, dx \]
for some very small value of \( \mu \) (\( \mu \leq \delta \)). In such a case, we have
\[ E_{\varepsilon}(f) \lesssim 16 \varepsilon D_1 \log(\mu^{-1}). \]
By \( A(f, a, b) \) we will denote the value of the integral (1.2) approximated numerically by some algorithm \( A \). We will also define
\[ E_A(f, a, b) := \left| \int_a^b f(x) \, dx - A(f, a, b) \right|. \]
Now, as \( |\int_0^\mu h(x) \, dx| \leq 2\mu D_1 \), we obtain
\[ I_\tau(f) = f(\tau) \log \frac{1-\tau}{1+\tau} + A(g_\varepsilon, -1, \tau - \delta) + A(g_\varepsilon, \tau + \delta, 1) + A(h_\varepsilon, \mu, \delta) + \mathcal{E}, \]
where
\[ |\mathcal{E}| \lesssim E_A(g_\varepsilon, -1, \tau - \delta) + E_A(g_\varepsilon, \tau + \delta, 1) + E_A(h_\varepsilon, \mu, \delta) + 16 \varepsilon D_1 \log(\mu^{-1}) + 2\mu D_1. \]

3.2 The second approach – open-type quadratures

If, for approximating the integrals on the right hand side of (2.4), we use a quadrature of the open type, then we are guaranteed that 0 does not belong to the set of nodes of the quadrature \( A(h, 0, \delta) \). Observe that, instead of (3.8), we may write
\[ I_\tau(f) = f(\tau) \log \frac{1-\tau}{1+\tau} + A(g, -1, \tau - \delta) + A(g, \tau + \delta, 1) + A(h, 0, \delta) + \mathcal{E}_2, \]
where
\[ |\mathcal{E}_2| \lesssim E_A(g, -1, \tau - \delta) + E_A(g, \tau + \delta, 1) + E_A(h, 0, \delta) + A(|g_\varepsilon - g|, -1, \tau - \delta) + A(|g_\varepsilon - g|, \tau + \delta, 1) + A(|h_\varepsilon - h|, 0, \delta). \]

Analogously to (3.7), we have
\[ A(|g_\varepsilon - g|, -1, \tau - \delta) + A(|g_\varepsilon - g|, \tau + \delta, 1) + A(|h_\varepsilon - h|, 0, \delta) \lesssim 16 \varepsilon D_1 A \left( \frac{1}{x}, 0, 1 \right). \]

Now, we use the following
Observation 1. Assume that

\[ A(\frac{1}{x}, 0, 1) = \sum_{k=0}^{n-1} G(\frac{1}{x}, s_k, s_{k+1}), \]

where

\[ G(f, s_k, s_{k+1}) = \sum_{j=0}^{m} B_{kj} f(x_{kj}) \]

is a Gauss or Gauss-Kronrod quadrature rule applied to the interval \([s_k, s_{k+1}]\).

If \(x_{0,0}\) denotes the smallest node of the quadrature rule \(G(f, s_0, s_1)\), then

\[ A(\frac{1}{x}, 0, 1) < C_{mn} \log(x_{0,0}^{-1}), \] (3.9)

and \(C_{mn} < 2\) for all \(m \geq 1, n \geq 0\).

The above observation has not yet been proved. However, it was verified experimentally for \(1 \leq m \leq 100\) and thousands different values of \(n\) and \(s_0, s_1, \ldots, s_n\). The greater the value of \(n\) is, the smaller is the constant \(C_{mn}\).

E.g., if \(m \geq 14\), then \(C_{mn} < 1.3\).

From (3.9), we immediately obtain

\[ |E_2| \lesssim E_A(g, -1, \tau - \delta) + E_A(g, \tau + \delta, 1) + E_A(h, 0, \delta) + 32\varepsilon D_1 \log(x_{0,0}^{-1}). \] (3.10)

4 Numerical experiments

4.1 The algorithm

Before we formulate our algorithm, we have to deal with one more problem, which, surprisingly, is usually ignored in the literature. Suppose that \(|\gamma| < \varepsilon\).

In some cases, the exact value of the integral \(I_{\tau(1+\gamma)}(f)\) may be significantly different from \(I_{\tau}(f)\). It means that even if we would have a perfect algorithm for computing the integrals of the type (2.1), we may get wrong results only by rounding the parameter \(\tau\). Such a situation takes place if \(|\tau|\) is close to 1 or if \(f'\) changes rapidly in the neighbourhood of \(\tau\).

We can see that if \(|\tau| \approx 1\), then small changes of \(\tau\) affect mostly the first term on the right hand side of (2.4). If we denote \(L(x) := \log((1-x)/(1+x))\), then

\[ |L(\tau(1+\gamma)) - L(\tau)| \lesssim |\tau L'(\tau)| \]
and, consequently,

\[ |I_{\tau(1+\gamma)}(f) - I_\tau(f)| \lesssim \frac{|f(\tau)|}{\min\{1+\tau,1-\tau\}}. \quad (4.1) \]

The error that results from the rapid changes of \( f' \) in the neighborhood of \( \tau \) seems to be much more difficult to estimate theoretically. However, we have confirmed experimentally that it can be approximated as follows:

\[ |I_{\tau(1+\gamma)}(f) - I_\tau(f)| \lesssim C|\tau|\sqrt{|f''(\tau)|} \quad (4.2) \]

for a properly selected constant \( C \).

The proposed algorithm was programmed in the Matlab language. As the algorithm \( A \) we used the build-in Matlab adaptive integrator \texttt{quadgk}. The approximation to the integral (2.1) is computed as follows:

\[ I_\tau(f) \approx A(g,-1,\tau - \delta) + A(g,\tau + \delta,1) + A(h,0,\delta). \]

In order to estimate the approximation error, we make use of the formula (3.10), where, instead of \( E_A(g,\cdot,\cdot) \) and \( E_A(h,\cdot,\cdot) \), we use the error bounds for \( E_A(g_\varepsilon,\cdot,\cdot) \) and \( E_A(h_\varepsilon,\cdot,\cdot) \) provided, in fact, by the \texttt{quadgk} subroutine. Also, because the last term in (3.10) is rather pessimistic and because we do not know the values of \( x_{0,0} \) and \( D_1 \), we replaced the factor \( 32D_1 \log(x_{0,0}^{-1}) \) by \( 8|f'(\tau)| \). The final error estimation is increased by the formulas given in (4.1) and (4.2) with \( C := 8 \). The values of \( f'(\tau) \) and \( f''(\tau) \) are approximated by simple divided differences.

4.2 Numerical results

As a counterpart we have chosen the algorithm presented in [3], which is based on the idea of Clenshaw-Curtis quadrature, and which we have verified to be fast and accurate. This algorithm is also of an automatic type, i.e., it tries to approximate the integral (2.1) within the prescribed error tolerance and also, together with the computed approximation, returns the estimated error bound.

We performed our test for the following set of examples:

- \( f(x) = e^x, \quad \tau = 0.5 \),
- \( f(x) = \sin(550x), \quad \tau = 0.8 \),
- \( f(x) = \sqrt{2 + \cos(200x)}, \quad \tau = 0.7 \).
\[ f(x) = \log^2(1.0001 - x), \quad \tau = 0.99, \]
\[ f(x) = \sqrt{|\cos(44x)|^3}, \quad \tau = -0.6, \]
\[ f(x) = \sqrt{1 - x^2 \cos(100x)}, \quad \tau = 0.5, \]
\[ f(x) = e^x, \quad \tau = 0.9999999, \]
\[ f(x) = e^{-100(x+0.4)^2} \sin(e^{-10x}), \quad \tau = -0.41. \]

By \( CC \) we denote the algorithm of [3], while by \( A \) the algorithm based on the adaptive quadrature, proposed in this paper. In \( Matlab \), \( \varepsilon \approx 1.1 \cdot 10^{-16} \), while the requested error tolerance was equal to \( 10^{-12} \). The experiments were performed on the computer with Intel Core i5 3.66 GHz processor. In tables, we give the values of the absolute errors of the computed approximations, the estimated error bounds for these errors provided by the algorithms, and the computation times.

Table 1: Comparison (absolute error, error estimation and computation time) of the algorithm of [3] (\( CC \)) and the present method (\( A \)) in the case of the integral \( \int_{-1}^{1} e^x(x - 0.5)^{-1}dx \) (\( 1T = 0.0003s \)).

| algorithm | absolute error | error estimation | computation time |
|-----------|----------------|-----------------|-----------------|
| \( CC \)  | \( 5.6 \cdot 10^{-16} \) | \( 1.4 \cdot 10^{-15} \) | \( 1.0T \) |
| \( A \)   | \( 7.8 \cdot 10^{-16} \) | \( 3.8 \cdot 10^{-15} \) | \( 5.7T \) |

Table 2: Comparison (absolute error, error estimation and computation time) of the algorithm of [3] (\( CC \)) and the present method (\( A \)) in the case of the integral \( \int_{-1}^{1} \sin(550x)(x - 0.8)^{-1}dx \) (\( 1T = 0.001s \)).

| algorithm | absolute error | error estimation | computation time |
|-----------|----------------|-----------------|-----------------|
| \( CC \)  | \( 1.9 \cdot 10^{-14} \) | \( 2.9 \cdot 10^{-14} \) | \( 1.0T \) |
| \( A \)   | \( 2.8 \cdot 10^{-13} \) | \( 5.1 \cdot 10^{-13} \) | \( 4.7T \) |

In Tables [1,4] we present the results in the case the function \( f \) is analytic in a complex region containing the interval \([-1,1]\). As the algorithm of [3] was designed for this class of functions, it performs a little faster than the algorithm presented in this paper. It is not a surprise, as adaptive quadratures are not meant to be daemons of speed, but to compute integrals.
Table 3: Comparison (absolute error, error estimation and computation time) of the algorithm of [3] (CC) and the present method (A) in the case of the integral \( \int_{-1}^{1} \sqrt{2 + \cos(200x)}(x - 0.7)^{-1}dx \) (1T = 0.0029s).

| algorithm | absolute error | error estimation | computation time |
|-----------|----------------|------------------|-----------------|
| CC        | 5.3 \( \cdot 10^{-15} \) | 1.5 \( \cdot 10^{-14} \) | 1.0T             |
| A         | 2.1 \( \cdot 10^{-14} \) | 1.1 \( \cdot 10^{-13} \) | 1.7T             |

Table 4: Comparison (absolute error, error estimation and computation time) of the algorithm of [3] (CC) and the present method (A) in the case of the integral \( \int_{-1}^{1} \log^{2}(1.0001 - x)(x - 0.99)^{-1}dx \) (1T = 0.0018s).

| algorithm | absolute error | error estimation | computation time |
|-----------|----------------|------------------|-----------------|
| CC        | 3.5 \( \cdot 10^{-13} \) | 2.9 \( \cdot 10^{-13} \) | 1.0T             |
| A         | 4.7 \( \cdot 10^{-13} \) | 9.0 \( \cdot 10^{-12} \) | 1.6T             |

Table 5: Comparison (absolute error, error estimation and computation time) of the algorithm of [3] (CC) and the present method (A) in the case of the integral \( \int_{-1}^{1} \sqrt{\cos(44x)}3(x + 0.6)^{-1}dx \) (1T = 2.3s).

| algorithm | absolute error | error estimation | computation time |
|-----------|----------------|------------------|-----------------|
| CC        | 7.3 \( \cdot 10^{-12} \) | 5.0 \( \cdot 10^{-12} \) | 1.0T             |
| A         | 4.9 \( \cdot 10^{-15} \) | 3.0 \( \cdot 10^{-13} \) | 0.007T           |

Table 6: Comparison (absolute error, error estimation and computation time) of the algorithm of [3] (CC) and the present method (A) in the case of the integral \( \int_{-1}^{1} \sqrt{1 - x^{2}}\cos(100x)(x - 0.9)^{-1}dx \) (1T = 3.7s).

| algorithm | absolute error | error estimation | computation time |
|-----------|----------------|------------------|-----------------|
| CC        | 6.7 \( \cdot 10^{-13} \) | 3.7 \( \cdot 10^{-12} \) | 1.0T             |
| A         | 2.0 \( \cdot 10^{-15} \) | 7.1 \( \cdot 10^{-14} \) | 0.0009T          |

for the widest possible class of functions. If the function \( f \) is not analytic (Tables 5 and 6), then the presented algorithm is much more efficient than the one of [3]. The results given in Tables 7 and 8 show that the problem
Table 7: Comparison (absolute error, error estimation and computation time) of the algorithm of [3] (CC) and the present method (A) in the case of the integral \( \int_{-1}^{1} e^x(x - 0.9999999)^{-1} dx \) \( (1T = 0.003s) \).

| Algorithm | Absolute Error | Error Estimation | Computation Time |
|-----------|----------------|-----------------|-----------------|
| CC        | 1.4 \cdot 10^{-9} | 1.4 \cdot 10^{-15} | 1.0T            |
| A         | 1.4 \cdot 10^{-9} | 3.0 \cdot 10^{-9}  | 5.7T            |

Table 8: Comparison (absolute error, error estimation and computation time) of the algorithm of [3] (CC) and the present method (A) in the case of the integral \( \int_{-1}^{1} e^{-100(x+0.4)^2} \sin(e^{-100x})(x + 0.41)^{-1} dx \) \( (1T = 0.003s) \).

| Algorithm | Absolute Error | Error Estimation | Computation Time |
|-----------|----------------|-----------------|-----------------|
| CC        | 1.3 \cdot 10^{-13} | 3.9 \cdot 10^{-16} | 1.0T            |
| A         | 1.2 \cdot 10^{-13} | 4.3 \cdot 10^{-13} | 0.3T            |

discussed in Section 4.1 is very important. When implementing the method of [3], we used only the error estimates presented there. As we can see, if the computations are performed in the fixed precision arithmetic, the additional error estimates similar to the ones given in (4.1) and (4.2) should be a part of any algorithm for computing Cauchy principal value integrals.

The experiments have shown that the very simple algorithm presented in this paper, based on the use of an adaptive quadrature, is very efficient and accurate. It can be applied to a very wide class of integrands and, as it was also proved theoretically, the influence of roundoff error on the accuracy of the result is very small.

References

[1] P. J. Davis and P. Rabinowitz, Methods of Numerical Integration, Second ed., Academic Press, New York, 1984.

[2] D. Elliott and D. F. Paget, Gauss type quadrature rules for Cauchy principal value integrals. Math. Comp. 33 (1979), 301–309.

[3] T. Hasegawa and T. Torii, An automatic quadrature for Cauchy principal value integrals, Math. Comp. 56 1991, 741–754.
[4] I. M. Longman, On the numerical evaluation of Cauchy principal values of integrals, MTAC 12 (1958), 205–207.

[5] G. Monegato, The numerical evaluation of one-dimensional Cauchy principal value integrals, Computing 29 (1982), 337–354.

[6] D. F. Paget and D. Elliott, An algorithm for the numerical evaluation of certain Cauchy principal value integrals, Numer. Math. 19 (1972), 373–385.

[7] R. Piessens, Numerical evaluation of Cauchy principal values of integrals, BIT 10 (1970), 476–480.

[8] C. E. Stewart, On the numerical evaluation of singular integrals of Cauchy type, J. Soc. Indust. Appl. Math. 8(2) (1960) 342–353.