A Two-Domain MATLAB Implementation for Efficient Computation of the Voigt/Complex Error Function

Sanjar M. Abrarov 1,2,3,*, Rehan Siddiqui 2,3, Rajinder K. Jagpal 2,3 and Brendan M. Quine 1,3

1 Thoth Technology Inc., Algonquin Radio Observatory, Achray Rd., RR6, Pembroke, ON K8A 6W7, Canada
2 Epic College of Technology, 5670 McAdam Rd., Mississauga, ON LAZ 1T2, Canada
3 Department Earth and Space Science, York University, 4700 Keele St., Toronto, ON M3J 1P3, Canada
* Correspondence: sanjar@thothx.com

Abstract: In this work we develop a new algorithm for the efficient computation of the Voigt/complex error function. In particular, in this approach we propose a two-domain scheme where the number of the interpolation grid-points is dependent on the input parameter $y$. The error analysis we performed shows that the MATLAB implementation meets the requirements for radiative transfer applications involving the HITRAN molecular spectroscopic database. The run-time test shows that this MATLAB implementation provides rapid computation, especially at smaller ranges of the parameter $x$.

Keywords: complex error function; Faddeeva function; Voigt function; interpolation

MSC: 65K05

1. Introduction

The complex error function, also known as the Faddeeva function, is defined as [1–4]

$$w(z) = e^{-z^2} \left(1 + \frac{2i}{\sqrt{\pi}} \int_0^z e^{t^2} dt \right),$$

where $z = x + iy$ is a complex argument. The complex error function $w(z)$ is closely related to the complex probability function [2]

$$W(z) = \frac{i}{\pi} \int_{-\infty}^{\infty} \frac{e^{-t^2}}{z - t} dt.$$

The complex probability function can be written in terms of its real and imaginary parts [2]

$$W(z) = K(x, y) + iL(x, y)$$

such that

$$K(x, y) = \text{Re}[W(z)] = \frac{y}{\pi} \int_{-\infty}^{\infty} \frac{e^{-t^2}}{y^2 + (x - t)^2} dt$$

and

$$L(x, y) = \text{Im}[W(z)] = \frac{1}{\pi} \int_{-\infty}^{\infty} \frac{e^{-t^2} (x - t)}{y^2 + (x - t)^2} dt.$$

Both functions $w(z)$ and $W(z)$ are equal to each other on the upper half of the complex plane, when $y = \text{Im}[z] > 0$ [2]. Consequently, it follows that

$$w(z) = K(x, y) + iL(x, y), \quad y > 0.$$
Further we will imply that the parameter \( y = \text{Im}[z] \) is always greater than zero.

The real part of the complex error function \( K(x, y) \) is known as the Voigt function \([1–4]\) that is widely used in Atmospheric Science to describe emission and absorption of the photons by atmospheric molecules \([5–12]\). Specifically, the Voigt function is used to compute wavelength-dependent absorption coefficients by using the HITRAN molecular spectroscopic database \([13]\). The imaginary part of the complex error function \( L(x, y) \) is also used in many applications. For example, it can describe the spectral behavior of the index of refraction in various materials \([14,15]\).

Despite simple representations, the integrals (1)–(3) do not have analytical solutions and must be computed numerically. Many approximations are available in the scientific literature \([16–29]\). Although this problem has been known for many decades, the derivation of new approximations for the complex error function \( w(z) \) and the development of their efficient algorithms still remain an interesting topic \([30–37]\).

In our previous publication we proposed an algorithm for the efficient computation of the complex error function based on a single-domain implementation with vectorized interpolation \([33]\). However, despite rapid performance it has some limitations, including a limited range for the parameter \( x \) as well as restrictions for the input array type. As a further development, in this work we present a new MATLAB implementation without those drawbacks. The numerical analysis and computational tests we performed show that the proposed algorithmic implementation meets all the requirements in terms of accuracy and run-time performance for the efficient computation of the Voigt/complex error function in the radiative transfer applications.

2. Approximations

2.1. Sampling Based Approximation

In our work \([29]\) we have proposed a new sampling method based on incomplete cosine expansion of the sinc function. In particular, it is shown that, using a new sampling method based on incomplete cosine expansion of the sinc function, we can obtain the following approximation

\[
w(z) \approx \sum_{m=1}^{M} \frac{a_m + b_m(z + \frac{i\zeta}{2})}{c_m^2 - (z + \frac{i\zeta}{2})^2},
\]

where the expansion coefficients are given by

\[
a_m = \frac{\sqrt{\pi}(m - 1/2)}{2M^2\hbar} \sum_{n=-N}^{N} e^{\frac{i^2}{4 - n^2}\hbar^2} \sin\left(\frac{\pi(m - 1/2)(nh + \zeta/2)}{M\hbar}\right),
\]

\[
b_m = -\frac{i}{M\sqrt{\pi}} \sum_{n=-N}^{N} e^{\frac{i^2}{4 - n^2}\hbar^2} \cos\left(\frac{\pi(m - 1/2)(nh + \zeta/2)}{M\hbar}\right),
\]

\[
c_m = \frac{\pi(m - 1/2)}{2M\pi},
\]

with parameters \( N, M, \hbar \) and \( \zeta \) that can be taken as 23, 23, 0.25 and 2.75, respectively.

For more rapid performance in algorithmic implementation, it is reasonable to define the function

\[
\Omega(z) = \sum_{m=1}^{M} \frac{a_m + b_mz}{c_m^2 - z^2},
\]

such that Equation (4) can be represented as \([30]\)

\[
w(z) \approx \Omega(z + \frac{i\zeta}{2}).
\]
Overall, approximation (5) is highly accurate. However, its accuracy deteriorates with decreasing parameter $y$. In order to resolve this problem we can use the following approximation \[30\]
\[
  w(z) \approx e^{-z^2} + z \sum_{m=1}^{M+2} \frac{a_m - \beta_m z^2}{\gamma_m - \theta_m z^2 + z^4},
\]
where the expansion coefficients are
\[
  a_m = b_m \left[ c_m^2 - \left( \frac{c_m^2}{2} \right)^2 \right] + i \alpha_m \zeta,
\]
\[
  \beta_m = b_m,
\]
\[
  \gamma_m = c_m^4 + \frac{c_m^2 \zeta^2}{2} + \frac{c_m^4}{16},
\]
and
\[
  \theta_m = 2c_m^2 - \frac{\zeta^2}{2}.
\]
It is interesting to note that this approximation of the complex error function is obtained by substituting approximation (4) into the right side of the identity (see \[30\] for details in derivation)
\[
  w(z) = e^{-z^2} + \frac{w(z) - w(-z)}{2}.
\]
For $|z| > 8$ one of the best choices is the approximation based on the Laplace continued fraction \[1,3\]. In particular, in our algorithm we used
\[
  w(z) \approx \begin{cases} 
  \text{Equation (5), if } (|x + iy| \leq 8) \cap (y > 0.05x), \\
  \text{Equation (6), if } (|x + iy| \leq 8) \cap (y \leq 0.05x), \\
  \text{Equation (7), otherwise.}
  \end{cases}
\]
This algorithm is implemented in MATLAB as a script file \textit{fadsamp.m} that utilizes three approximations (5)–(7) as follows \[30\]
\[
  w(z) \approx \begin{cases} 
  \text{Equation (5), if } (|x + iy| \leq 8) \cap (y > 0.05x), \\
  \text{Equation (6), if } (|x + iy| \leq 8) \cap (y \leq 0.05x), \\
  \text{Equation (7), otherwise.}
  \end{cases}
\]
As we have shown in our publication \[30\], approximation (8) provides highly accurate and rapid computation of the complex error function without poles and can be used to cover the entire complex plane.

2.2. Modified Trapezoidal Rule

In 1945, English mathematician and cryptanalyst Alan Turing, who succeeded to decrypt sophisticated machine codes of the Enigma during the Second World War \[38\], published an interesting paper where he proposed an elegant method of numerical integration for some class of integrals \[39\]. Nowadays, his method of the numerical integration that involves the residue calculus is regarded as the modified trapezoidal rule \[40\] or the generalized trapezoidal rule \[37\]. The comprehensive and detailed description of Turing’s method of integration may be found in literature \[40\].
In 1949, Goodwin showed how to implement Turing’s idea to the integrals of kind [41]

\[
\int_{-\infty}^{\infty} f(x) e^{-x^2} \, dx.
\]

Using the method described by Goodwin, Chiarella and Reichel in their work [42] derived the series expansion for the following integral

\[
\Psi(x, t) = U(x, t) + iV(x, t) = \frac{\Omega(x, t)}{(4\pi t)^{1/2}} \int_{-\infty}^{\infty} e^{-u^2} \, du,
\]

where \(\Omega(x, t) = (1 - ix) / (2t^{1/2})\). In particular, they showed that that the function \(\Psi(x, t)\) can be approximated as a series

\[
\Psi(x, t) \approx \frac{h}{\Omega(x, t)(4\pi t)^{1/2}} + \frac{2h\Omega(x, t)}{(4\pi t)^{1/2}} \sum_{n=1}^{\infty} \frac{e^{-n^2h^2}}{\Omega^2(x, t) + n^2h^2}
+ \frac{\pi e^{\Omega^2(x, t)}}{(\pi t)^{1/2} (1 - e^{2\pi\Omega(x, t)/h})} H\left(t - \frac{h^2}{\pi^2}\right),
\]

where \(h\) is a small fitting parameter and \(H(t)\) is the Heaviside step function defined as

\[
H(t) = \begin{cases} 
0, & \text{if } t < 0, \\
1/2, & \text{if } t = 0, \\
1, & \text{if } t > 0.
\end{cases}
\]

Thus, due to Heaviside step function, Equation (9) can be separated into three parts

\[
\Psi(x, t) \approx \frac{h}{\Omega(x, t)(4\pi t)^{1/2}} + \frac{2h\Omega(x, t)}{(4\pi t)^{1/2}} \sum_{n=1}^{\infty} \frac{e^{-n^2h^2}}{\Omega^2(x, t) + n^2h^2}, \quad t < \frac{h^2}{\pi^2},
\]

\[
\Psi(x, t) \approx \frac{h}{\Omega(x, t)(4\pi t)^{1/2}} + \frac{2h\Omega(x, t)}{(4\pi t)^{1/2}} \sum_{n=1}^{\infty} \frac{e^{-n^2h^2}}{\Omega^2(x, t) + n^2h^2} + \frac{\pi e^{\Omega^2(x, t)}}{(\pi t)^{1/2} (1 - e^{2\pi\Omega(x, t)/h})}, \quad t = \frac{h^2}{\pi^2},
\]

\[
\Psi(x, t) \approx \frac{h}{\Omega(x, t)(4\pi t)^{1/2}} + \frac{2h\Omega(x, t)}{(4\pi t)^{1/2}} \sum_{n=1}^{\infty} \frac{e^{-n^2h^2}}{\Omega^2(x, t) + n^2h^2} + \frac{\pi e^{\Omega^2(x, t)}}{(\pi t)^{1/2} (1 - e^{2\pi\Omega(x, t)/h})}, \quad t > \frac{h^2}{\pi^2}.
\]

Equation (11) deals only with a single point \(h^2/\pi^2\) for the parameter \(t\) and does not represent any practical interest. Therefore, further we will consider only two equations, Equations (10) and (12).

Mata and Reichel [43] showed the relations

\[
K(x, y) = \frac{1}{y\sqrt{\pi}} U\left(\frac{x}{y}, \frac{1}{4y^2}\right)
\]

and

\[
L(x, y) = \frac{1}{y\sqrt{\pi}} V\left(\frac{x}{y}, \frac{1}{4y^2}\right)
\]
that both functions $\Psi(x, t)$ and $w(x, y)$ with each other. Consequently, using these
relations the series expansions (10) and (12) can be reformulated as

$$w(z) \approx \frac{2ihz}{\pi} \sum_{k=0}^{N} \frac{e^{-t_{k}^{2}}}{z^{2} - t_{k}^{2}}$$

(13)

and

$$w(z) \approx \frac{2e^{-z^{2}}}{1 + e^{-2\pi z/h}} + \frac{2ihz}{\pi} \sum_{k=0}^{N} \frac{e^{-t_{k}^{2}}}{z^{2} - t_{k}^{2}}$$

(14)

respectively, where $t_{k} = (k + 1/2)h$ and $h$ can be chosen to be equal to $\sqrt{\pi/(N + 1)}$ [37].

Consider an expansion series for the complementary error function that was reported
by Hunter and Regan [44] (see also [37])

$$\text{erfc}(z) \approx \frac{2hz e^{-z^{2}}}{\pi} \sum_{k=1}^{N} \frac{e^{-(k-1/2)^{2} h^{2}}}{z^{2} + (k - 1/2)^{2} h^{2}} + \frac{2}{1 + e^{2z/h}} \times x < \frac{\pi}{h}.$$  

Using the identity relating complex error function and complementary error function [3]

$$w(z) = e^{-z^{2}} \text{erfc}(-iz),$$

we obtain

$$w(z) \approx \frac{2e^{-z^{2}}}{1 + e^{-2\pi z/h}} + \frac{ih}{\pi z} + \frac{2hz}{\pi} \sum_{k=1}^{N} \frac{e^{-t_{k}^{2}}}{z^{2} - t_{k}^{2}},$$

(15)

where $\tau_{k} = kh$ [37].

Equations (13) and (14) have poles at $z = t_{k}$ while Equation (15) has poles at $z = \tau_{k}$. Furthermore, each of these equations can cover with high accuracy only in the corre-
sponding domain. However, Al Azah and Chandler-Wilde [37] recently proposed the
following approximation

$$w(z) \approx \begin{cases} \text{Equation (13)}, & \text{if } y \geq \max(\pi, x), \\ \text{Equation (15)}, & \text{if } x < (x \cap 1/4) \leq \varphi(x/h) \leq 3/4, \\ \text{Equation (14)}, & \text{otherwise}, \end{cases}$$

(16)

where $\varphi(t) = t - |t| \in [0, 1)$, that appeared to be very efficient since it can be used for rapid
and highly accurate computation without poles at $N = 11$. This is possible to achieve since,
according to approximation (16), Equations (13)–(15) are used interchangeably depending
on the domain over the entire complex plain.

3. Algorithmic Implementation

Previously we have reported a new algorithm based on a vectorized interpolation over
a single-domain [33]. Such an approach provides accuracy better than $10^{-6}$ at $y \geq 10^{-8}$
for the HITRAN [13] applications. However, this implementation has several limitations.
In particular, there is a limitation $|x| \leq 10^{3}$. Although it is possible to increase the range
for more than $10^{5}$, it requires to introduce more interpolation grid-points for precomputation.
Furthermore, this MATLAB implementation accepts an input only as a vector (one
dimensional array) or as a scalar.

One of the ways to implement an efficient algorithm is to use an approximation based
on the two-domain scheme that we proposed in our earlier publication [23]
where the coefficients are [20]
\[
\begin{align*}
a_1 &= y / (2\sqrt{\pi}) + y^3 / \sqrt{\pi} \approx 0.2820948y + 0.5641896y^3 \\
b_1 &= y / \sqrt{\pi} \approx 0.5641896y \\
a_2 &= 0.25 + y^2 + y^4 \\
b_2 &= -1 + 2y^2
\end{align*}
\]
such that
\[
\frac{a_1 + b_1 x^2}{a_2 + b_2 x^2 + x^4} = \text{Re}\left\{ \frac{(i / \sqrt{\pi})}{z - \frac{1}{2} x} \right\}.
\]
Consequently, the complex error function can also be approximated as [33]
\[
w(z) = K(x, y) + iL(x, y) \approx \begin{cases} 
\text{interpolation}, & \frac{x^2}{2\pi^2} + \frac{y^2}{15\pi} \leq 1 \\
(i / \sqrt{\pi}), & \frac{x^2}{2\pi^2} + \frac{y^2}{15\pi} > 1.
\end{cases}
\] (17b)

Although the algorithms for the Voigt and complex error functions, built on Equations (17a) and (17b) can provide rapid computations, their accuracies deteriorate at \( y < 10^{-6} \).

In order to resolve this problem we developed a new algorithm that utilizes the internal MATLAB built-up features. Unlike interpolation algorithms shown in [23, 33], the proposed approach implies that the number of the interpolation grid-points required for precomputation is not constant and dependent on the input parameter \( y \) such that
\[
N_{gp} = \frac{1}{\sqrt{y}} + \delta, \quad (18)
\]
where \( \delta \) is the offset that was found experimentally to be \( 5 \times 10^3 \). The corresponding interpolation grid-points range is bounded by \([-r, r] \), where \( r = 35 \) is the radius that was also found experimentally. The interpolation grid-points are distributed non-equidistantly in logarithmic scale to increase their density near the origin along the \( x \) axis.

Equation (18) is entirely empirical. It is reasonable to suggest that the number of the interpolation grid-points should be increased with decreasing \( y \). However, taking the reciprocal of \( y^n \), such that \( n \geq 1 \), results in a very rapid increase of the interpolation grid-points \( N_{gp} \) with decreasing \( y \). Our experimental results show that a more or less optimal value \( N_{gp} \) can be found by taking the reciprocal of \( \sqrt{y} \). The parameter \( \delta \) in Equation (18) is to provide a sufficient number of the interpolation grid-points \( N_{gp} \) within the internal domain \(|x + iy| \leq r\) as parameter \( y \) increases.

The use of Equation (18) is given as a basic for applications. If higher accuracy is required, the number of the interpolation grid-points \( N_{gp} \) can be further increased by using, for example, a modified form of the equation above
\[
N_{gp} = \frac{2}{\sqrt{y}} + 3\delta.
\]
Such a modification improves accuracy by an order of the magnitude.

The algorithm utilizes two domains, internal and external, that are bounded by a circle of radius \(|x + iy| = r\). The internal domain is situated within the circle while the external domain is situated outside of it.

All points within internal domain \(|x + iy| \leq r\) are computed by the MATLAB built-in interpolation function interp1 though interpolation grid-points that are computed by using the function file fadsamp.m provided in our article [30]. The spline method was found to be the best for interpolation.
All points outside external domain $|x + iy| > r$ are computed by the following approximation

$$w(z) \approx \frac{i/\sqrt{\pi}}{z - \frac{1}{2}z^{-1/2}}$$

that represents a simplified version of Equation (7) above.

The parameter $y$ is dependent on temperature and pressure. Therefore, it is related to the atmospheric height. In a radiative transfer model, the atmospheric column is sliced into layers $[5,6]$. The layers can be considered homogeneous if their thicknesses are small enough. Typically, a thickness of 50 m (or 20 layers per km) is sufficient to consider the layer homogeneous for atmospheric modeling. This implies that, within a given layer, the temperature and pressure are constants. Since the density of air in the mesosphere is extremely low, its contribution to the radiance is practically negligible. Consequently, taking into consideration that only the troposphere and stratosphere, with an atmospheric column corresponding to a height of up to 40–50 km, actually contribute to the radiance, we can conclude that the total number of layers as well as elements for parameter $y$ does not exceed 1000 per gas in atmosphere. However, in many practical tasks, for a single value of $y$ at a given atmospheric temperature and pressure, we may need to cover a wide spectral range with high resolution. This signifies that the number of elements for parameter $x$ can exceed a million. Since the number of elements for parameter $y$ is much smaller than the number of elements for parameter $x$, it is reasonable to implement a configuration in which parameter $x$ is an array and parameter $y$ is a scalar. This approach is more rapid in the MATLAB implementation since the application of a 2D array $M \times N$, where $N$ and $M$ are numbers of elements for parameters $x$ and $y$, respectively, requires considerably more computer memory.

As it has been mentioned above, the computation of the interpolation grid-points in its original version of the function file $w2dom.m$ is performed by external function $fadsamp.m$. However, any other MATLAB function file that can provide highly accurate computation of the complex error function may also be used for computation of interpolation grid-points. For example, the function files such as $fadf.m$ $[27]$ and $fadfunc.m$ $[31]$ can also be used as alternatives. The script of the function file $w2dom.m$ is given in Appendix A.

4. Error Analysis

In order to exclude the rounding and truncation errors in computation by using the most recent HITRAN database $[13]$, the values of the $K(x, y)$ and $L(x, y)$ functions with 6 or more accurate decimal digits in their mantissas are required. Therefore, in radiative transfer applications involving the HITRAN database, the accuracy of computation of these functions has to be better than $10^{-6}$.

Figure 1 shows the absolute difference $|K_{ref}(x, y) - K(x, y)|$, where $K_{ref}(x, y)$ is a highly accurate reference, in the range $-5 \leq x \leq 5$ at $y = 10^{-8}$. As we can see, the absolute difference does not exceed $2.5 \times 10^{-13}$.

Figure 2 shows the absolute difference $|L_{ref}(x, y) - L(x, y)|$, where $L_{ref}(x, y)$ is a highly accurate reference, in the range $-5 \leq x \leq 5$ also at $y = 10^{-8}$. We can see that for the imaginary part the absolute difference also does not exceed $2.5 \times 10^{-13}$. 

Figure 1. Absolute difference for the real part $K(x, y)$ of the complex error function.

Figure 2. Absolute difference for the imaginary part $L(x, y)$ of the complex error function.

For more rigorous error analysis, we can apply the following relative errors

$$\Delta_{\text{Re}} = \left| \frac{K_{\text{ref}}(x, y) - K(x, y)}{K_{\text{ref}}(x, y)} \right|$$

and

$$\Delta_{\text{Im}} = \left| \frac{L_{\text{ref}}(x, y) - L(x, y)}{L_{\text{ref}}(x, y)} \right|$$

for the real and imaginary parts of the complex error function $w(z)$, respectively.

Figure 3 depicts the relative error for the real part of the complex error function $\text{Re}[z] = K(x, y)$ in the domain $0 \leq x \leq 50$ and $0 \leq y \leq 10^{1.699}$, where $10^{1.699} \approx 50$. Inset in Figure 3 shows the smaller domain near the origin $0 \leq x \leq 5.5$ and $0 \leq y \leq 10^{0.74}$, where $10^{0.74} \approx 5.5$. Beyond this smaller domain the observed color is blue indicating that the relative error is below $10^{-12}$. As we can see from Figure 3, the relative error does not exceed $\sim 10^{-10}$. 
Figure 3. Relative error $\Delta_{Re}$ for the real part $K(x,y)$ of the complex error function. Inset shows the relative error $\Delta_{Re}$ in the area near the origin.

Figure 4 illustrates the relative error for the imaginary part of the complex error function $\text{Im}[z] = L(x,y)$ in the domain $0 \leq x \leq 50$ and $0 \leq y \leq 10^{1.699}$. Inset in Figure 4 shows the smaller domain near origin $0 \leq x \leq 5.5$ and $0 \leq y \leq 10^{0.74}$. Beyond this smaller domain the color is blue and the relative error is below $10^{-12}$. As we can see from Figure 4, the relative error is generally lower in the imaginary part and does not exceed $\sim 10^{-11}$.

It is commonly known that the accuracy of $K(x,y)$ and $L(x,y)$ functions tend to deteriorate when $y$ decreases [2,16,21,22,28]. However, from Figures 3 and 4 we can see that the decrease of the parameter $y$ does not deteriorate the accuracy. This is possible to achieve since in accordance with Equation (18) the number of the interpolation grid-points $N_{gp}$ increases with decreasing $y$. Therefore, this technique enables us to resolve efficiently this problem in computation of the complex error function $\bar{w}(z)$.

5. Run-Time Test

The run-time test was performed with MATLAB function files $w\text{Trap.m}$ [37], $\text{fadsamp.m}$ [31] and $w\text{dom.m}$ at equidistantly distributed 10 million points for parameter $x$ at $y = 10^{-8}$. The results of the run-time test are shown in Table 1.
Table 1. Run-time of algorithms for 10 million points at three different ranges.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Run-Time in Seconds</th>
<th>$x \in [-10, 10]$</th>
<th>$x \in [-10^2, 10^2]$</th>
<th>$x \in [-10^3, 10^3]$</th>
</tr>
</thead>
<tbody>
<tr>
<td>wTrap.m</td>
<td>2.41</td>
<td>2.55</td>
<td>2.45</td>
<td></td>
</tr>
<tr>
<td>fadsamp.m</td>
<td>4.14</td>
<td>1.78</td>
<td>1.54</td>
<td></td>
</tr>
<tr>
<td>w2dom.m</td>
<td>1.23</td>
<td>1.04</td>
<td>0.98</td>
<td></td>
</tr>
</tbody>
</table>

It has been reported that both algorithms wTrap.m and fadsamp.m are highly accurate in computation [37]. In particular, the maximum values in relative errors for the algorithms wTrap.m and fadsamp.m are found to be $\sim 10^{-15}$ and $\sim 10^{-14}$, respectively. However, the computational speed of these two algorithms differs depending on the range for the input parameter $x$. In particular, Table 1 shows that at smaller range for parameter $x$ the function file wTrap.m performs computation faster than the function file fadsamp.m. However, as the range for parameter $x$ increases, our algorithm fadsamp.m becomes faster. The run-time test also reveals that the algorithm w2dom.m is always faster regardless of the chosen range. This is particularly evident when the range for the input parameter $x$ is smaller.

Table 2 shows the results of the run-time test at 30 million equidistantly distributed points for parameter $x$ at $y = 10^{-8}$. As we can see, the algorithm w2dom.m remains more rapid proportionally at extended size of the input array as compared to the wTrap.m and fadsamp.m algorithms.

Table 2. Run-time of algorithms for 30 million points at three different ranges.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Run-Time in Seconds</th>
<th>$x \in [-10, 10]$</th>
<th>$x \in [-10^2, 10^2]$</th>
<th>$x \in [-10^3, 10^3]$</th>
</tr>
</thead>
<tbody>
<tr>
<td>wTrap</td>
<td>8.38</td>
<td>8.33</td>
<td>8.42</td>
<td></td>
</tr>
<tr>
<td>fadsamp</td>
<td>14.37</td>
<td>6.02</td>
<td>5.21</td>
<td></td>
</tr>
<tr>
<td>w2dom</td>
<td>3.78</td>
<td>3.07</td>
<td>2.86</td>
<td></td>
</tr>
</tbody>
</table>

The MATLAB code for the run-time test is provided in Appendix B. The scripts of function files wTrap.m and fadsamp.m can be accessed from the cited literature [30,37], respectively.

6. Conclusions

A new algorithm for the efficient computation of the Voigt/complex error function is shown. We propose a two-domain scheme where the number of interpolation grid-points $N_{gp}$ is dependent on parameter $y$. The error analysis shows that our MATLAB implementation meets the requirements for radiative transfer applications utilizing the HITRAN molecular spectroscopic database. The run-time test we performed shows that this MATLAB implementation can provide rapid computation especially at smaller ranges of parameter $x$.

Author Contributions: S.M.A. developed the software, performed data analysis and wrote the manuscript. R.S., R.K.J. and B.M.Q. developed the software and performed data analysis. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: This work is supported by the National Research Council Canada, Thoth Technology Inc., York University, Epic College of Technology and Epic Climate Green (ECG) Inc.
Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

\begin{verbatim}
function FF = w2dom(x,y,opt)

% SYNOPSIS:
% x - array or scalar
% y - scalar
% opt - option for the real and imaginary parts
% %
% % opt = 1 returns value(s) for the Voigt function, K(x,y)-function
% % opt = 2 returns value(s) for the L(x,y)-function
% % opt = 3 returns value(s) for the complex error function
% %
% % This code is primarily intended to work in the range y >= 1e-8 for the
% % HITRAN applications for accelerated computation of the Voigt/complex
% % error function.
% %
% % Example:
% % x = linspace(-10,10,1e7); y = 1e-8; tic; w2dom(x,y); toc
% %
% if ~isscalar(y)
% disp('Input parameter y must be a scalar')
% return % terminate computation if y is not a scalar
% end
% if nargin == 2
% opt = 3;
% disp('Default value opt = 3 is assigned.')
% end
% if opt ~= 1 && opt ~= 2 && opt ~= 3
% disp(['Wrong parameter opt = ',num2str(opt),'! Use 1, 2 or 3.'])
% return
% end
% if y < 1e-8
% if opt == 1
% FF = real(fadsamp(x+1i*y));
% elseif opt == 2
% FF = imag(fadsamp(x+1i*y));
% else
% FF = fadsamp(x+1i*y);
% end
% return
% end
% FF = zeros(size(x));
% radius = 35; % define radius
% ind = abs(x + 1i*y) <= radius;
% gp =[-radius,radius]; % define grid-points
% if isempty(x(ind))
% offset = 5*1e3; % assign offset
% nump = 1/sqrt(y) + offset; % assign number of points
% % For better accuracy use for example nump = 2/sqrt(y) + 3*offset;
% gp = radius*(logspace(log10(1 + eps),log10(2),nump) ... 
% - 1); % notice the log scale
% gp = [flip(-gp),gp];
% end
\end{verbatim}
switch opt  
  case 1
      FF(ind) = real(internD(x(ind), y, gp));
      FF(~ind) = real(externD(x(~ind) + 1i*y));
  case 2
      FF(ind) = imag(internD(x(ind), y, gp));
      FF(~ind) = imag(externD(x(~ind) + 1i*y));
  otherwise
      FF(ind) = internD(x(ind), y, gp);
      FF(~ind) = externD(x(~ind) + 1i*y);
end

function IntD = internD(x, y, gp) % internal domain
    IntD = interp1(gp, fadsamp(gp + 1i*y), x, ...
                   'spline'); % interpolated values
end

function ExtD = externD(z) % external domain
    num = 1:4; % define a row vector
    num = num/2;
    ExtD = num(end)./z; % start computing from the end
    for m = 1:length(num) - 1
        ExtD = num(end - m)./(z - ExtD);
    end
    ExtD = 1i/sqrt(pi)./(z - ExtD);
end
end

Appendix B

clear
clc

%% Table for run-time in seconds
tab{1,1}='Algorithm';
tab{1,2}='[-10 to 10]'; % range 1
tab{1,3}='[-10^2 to 10^2]'; % range 2
tab{1,4}='[-10^3 to 10^3]'; % range 3

for k=1:3 % k is a factor
    for m=1:3 % three ranges
        x0=10^m; x=linspace(-x0,x0,k*1e7); % 1 and 3 are for 1e7 and 3*1e7 ...
        % 1e7 and 3*1e7 points, respectively
    if k==1
        disp(['10 million points in range ',num2str(m)]
    else
        disp(['30 million points in range ',num2str(m)]
    end
    disp('Computing, please wait!')
    tic; for n=1:maxN; wTrap(x+1i*y,11); end; tab{2,m+1}=toc/maxN;
    tic; for n=1:maxN; fadsamp(x+1i*y); end; tab{3,m+1}=toc/maxN;
    tic; for n=1:maxN; w2dom(x,y); end; tab{4,m+1}=toc/maxN;
    clc
end
if k==1; tab1=tab; else tab2=tab; end % assign tab1 or tab2
end
disp('Displaying Table 1:')
disp(tab1)

disp('Displaying Table 2:')
disp(tab2)

References

1. Faddeyeva, V.N.; Terent’ev, N.M. Tables of the Probability Integral $w(z) = e^{-z^2} \left(1 + \frac{2}{\sqrt{\pi}} \int_0^z e^{-t^2} \, dt \right)$ for Complex Argument; Pergamon Press: Oxford, UK, 1961.


