An efficient procedure for custom beam-profile convolution in polar coordinates: testing, benchmarking and application in tissue optics

O. Melchert, M. Wollweber and B. Roth

Hannover Centre for Optical Technologies (HOT), Leibniz Universität Hannover, Nienburger Str. 17, D-30167 Hannover, Germany

E-mail: `oliver.melchert@hot.uni-hannover.de`

Abstract. We discuss efficient algorithms for the accurate forward and reverse evaluation of the discrete Fourier-Bessel transform (dFBT) as numerical tools to assist in the 2D polar convolution of two radially symmetric functions, relevant, e.g., to applications in computational biophotonics. In our survey of the numerical procedure we account for the circumstance that the objective function might result from a more complex measurement process and is, in the worst case, known on a finite sequence of coordinate values, only. We contrast the performance of the resulting algorithms with a procedure based on a straight forward numerical quadrature of the underlying integral transform and asses its efficienty for two benchmark Fourier-Bessel pairs. An application to the problem of finite-size beam-shape convolution in polar coordinates, relevant in the context of tissue optics, is used to illustrate the versatility and computational efficiency of the numerical procedure. Further, we address the important issue of testing research code written in the `python` scripting language by using its off-the-shelf unit testing library `unittest`.

Keywords: computational biophotonics, beam-profile convolution, polar convolution, discrete Fourier-Bessel transform, Python, unit testing

PACS numbers: 02.70.-c, 02.30.Mv, 87.64.Aa

1. Introduction

The Fourier-Bessel transform (FBT; also referred to as “0-th order Hankel transform”) and, more generally, the n-th order Hankel transform (HT), represent mathematical tools that appear in numerous computational approaches in science and engineering. Among those are, e.g., applications in atomic scattering [1], electron microscopy [2], tomographic image reconstruction [3], beam propagation through axially symmetric systems [4], and, tissue optics [5]. The underlying theory and the operational rules for use with those transforms are thoroughly discussed in Ref. [6] where also an extensive review of the scientific literature can be found. In addition to the above applications, the FBT allows for the convolution of two radially symmetric functions in polar coordinates [7], a computational tool in its own right. This is viable since the general 2D convolution of two functions can be expressed in terms of their respective Fourier series expansion, exhibiting a nontrivial relation to the n-th order reverse Hankel transform. However, if both functions are restricted to be radially symmetric, their respective Fourier series expansions are nonzero for the term $n = 0$ only, and, consequently, their convolution can be shown to relate to a FBT [7].

In the presented article, we aim to draw some more attention to an efficient algorithm for the accurate evaluation of the discrete Hankel transform (dHT) due to Fisk-Johnson [2]. Albeit the latter reference introduced the dHT algorithm, an in-depth discussion of the discretization scheme for the forward and reverse transformation are provided by Ref. [6]. Our motivation to study the Fisk-Johnson dHT procedure is based on its efficiency for the purpose of polar convolution, requiring a discrete Fourier-Bessel transform (dFBT), i.e. a dHT of order zero, only. As discussed in the seminal article [2], the algorithmic procedure might lead to a significant reduction in computation time, if, subsequent to a HT a follow up reverse transform is required. The specific objective of this study is to explore this potential benefit for applications in computational biophotonics that require a large number of forward and reverse dHTs. We apply the resulting procedure to solve a problem in tissue optics where the task is to convolve the Green’s function response of a multilayered tissue with a custom irradiation source profile to yield the response to a laser-beam of finite extend. Therein, the Green’s function response is obtained from Monte Carlo simulations of an infinitely thin “pencil” laser-beam, incident to a layered tissue sample [8]. It thus results from a complex measurement process that yields the obective function on a finite sequence of, say, equidistant sample points. While this particular use case requires a dHT of order zero only, other examples involving n-th order dHTs can be found, see, e.g. Ref. [4] discussing beam propagation through axially symmetric systems and Ref. [3] discussing the reconstruction of two-dimensional images from their one-dimensional projections in computer-aided tomography.

The article is organized as follows: in section 2 we resume the forward and reverse dFBT procedures that pave the way for an efficient polar convolution algorithm. In order to benchmark our resarch code we assess the accuracy and perfomance of the procedures for two benchmark Fourier-Bessel transform pairs in section 3. In section 4 we then elaborate on the problem of postprocessing a Green's function material response to conform to a spatially extended photon beam in computational biophotonics. In section 5 we summarize and conclude on the presented study. Finally, in Appendix A, we address the important issue of testing research code and implementing a means of quality controll via the `python` scripting language by using its off-the-shelf unit testing library `unittest`.

2. Polar convolution in terms of discrete Fourier-Bessel transforms (dFBTs)

Here we consider a discrete approximation to the Fourier-Bessel transform $F_0(\rho)$ of a function $f(x)$ of a real variable $x \geq 0$, for which $\int_0^\infty f(x)x^{1/2}\ \mathrm{d}x$ is required to be absolutely convergent, defined by [7, 6]

$$F_0(\rho) = \int_0^\infty f(x) J_0(x\rho) x \ \mathrm{d}x. \tag{1}$$

Due to self-reciprocality, its reverse transform reads

$$f(x) = \int_0^\infty F_0(\rho) J_0(x\rho)\rho \ \mathrm{d}\rho. \tag{2}$$

Therein $J_0(\cdot)$ signifies the 0th order Bessel function and, together, f and F_0 comprise a Fourier-Bessel transform pair. Following Ref. [2], the dFBT is based on two reasonable assumptions: (A1) one can give a truncation threshold T above which the objective function vanishes, and, (A2) the Fourier-Bessel series of the objective function might be truncated after N terms. From an applied point of view and so as to yield a finite computational procedure, both assumptions are inevitable and might be satisfied by reasonably large values of T and N. Subsequently, we distinguish the forward transform for continuous objective functions as well as for objective functions known at a finite sequence sample points and allude to their universal backward transformation.

Forward transform for continuous objective functions: For given values of T and N, let $\{j_m\}_{m=1}^N$ denote the sequence of the first N zeros of J_0 in ascending order. Then, the forward dFBT for a continuous objective function, involving the zeros of the Bessel function, derived in Ref. [2], reads

$$F_0(j_m/T) = \frac{2T^2}{j_N^2} \sum_{k=1}^{N-1} \frac{J_0(j_k j_m/j_N)}{J_1^2(j_k)} f(j_k T/j_N), \tag{3}$$

where $J_1(\cdot)$ refers to the first order Bessel function. The above approximation to Eq. (1) is feasible, since, given a continuous objective function, the function values at $\{x_k T\}_{k=1}^{N-1}$ with $x_k = j_k/j_N$ can be computed in a straight forward manner. As a result one obtains the Fourier-Bessel transform of $f(x)$ at the discrete sequence $\{j_m/T\}_{m=1}^N$ of scaled Bessel zeros. Note that the above algorithm terminates in time $O(N^2)$.

Forward transform for discrete objective functions: If the objective function is known for a finite sequence $\{x_kT\}_{k=1}^{M}$ of sample points that do not meet the requirement of $x_k = j_k/j_N$ in Eq. (3) above, we might nevertheless proceed by computing its Fourier-Bessel expansion coefficients to obtain its transform at the same set $\{j_m/T\}_{m=1}^{N}$ of sample points as

$$F_0(j_m/T) = T^2 \int_0^1 x f(xT) J_0(j_m x)\ \mathrm{d}x \tag{4}$$

provided that the number of sample points M is large enough. In our numerical experiments we used a trapezoidal rule to approximate the integral in Eq. (4). Note that under the reasonable assumption $M \gg N$, the above algorithm terminates in time $O(NM)$.

Universal backward transform: If, subsequent to one of the transforms given by Eqs. (3) and (4), an immediate back-transformation is required, arbitrary function values $f(xT)$ for the parameters T and N can be computed by using the sequence $\{F_0(j_m/T)\}_{m=1}^{N}$ of dFBT samples according to [2]

$$f(xT) = \frac{2}{T^2} \sum_{m=1}^{N-1} \frac{F_0(j_m/T)}{J_1^2(j_m)} J_0(j_m x), \qquad 0 \leq x \leq 1. \tag{5}$$

Note that due to (A1) one has $f(xT) = 0$ for $x > 1$. Further, note that the above reverse algorithm terminates in time $O(N)$ for a given value of x.

Polar convolution using the dFBT: As pointed out earlier, from a point of view of computational complexity, the Fisk-Johnson procedure is particularly efficient if a dFBT, resulting in the sequence of transform estimates $\{F_0(j_m/T)\}_{m=1}^{N}$, is followed by a reverse transform based on the summation of F_0 at the exact same sequence of sample points along the transformed domain. Now, considering two radially symmetric functions it is possible to take advantage of the above procedure in order to derive an efficient algorithm for their polar convolution. Let $f(r)$ and $g(r)$ be two such radially symmetric functions. Then, their 2D (polar) convolution $h(r)$, again a function with radial symmetry, can be computed via [7]

$$h(r) = \mathsf{polConv}[f, g](r) = 2\pi \int_0^\infty F_0(\rho) G_0(\rho) J_0(\rho r) \rho\ \mathrm{d}\rho, \tag{6}$$

wherein $F_0(\rho)$ and $G_0(\rho)$ signify the Fourier-Bessel transforms of $f(r)$ and $g(r)$, respectively [9]. A Fisk-Johnson approximation $\mathsf{polConv}[f, g](r; T, N)$ of the polar convolution can thus be formulated as a three step procedure: (i) set the threshold parameters T and N of the Fisk-Johnson procedure, (ii) compute both dFBTs $F_0(\rho_m)$ and $G_0(\rho_m)$ at the same sequence $\{\rho_m = j_m/T\}_{m=1}^{N}$ of samples along the transformed domain using either Eq. (3) or (4), and, (iii) compute the pointwise products $H_0(\rho_m) = 2\pi F_0(\rho_m) G_0(\rho_m)$ followed by a reverse transformation via Eq. (5) to yield $h(xT)$ for $0 \leq x \leq 1$. The resulting Fisk-Johnson polar convolution is thus no more expensive than $O(NM)$ if the number of grid points x_i at which $h(x_iT)$ is sampled is of order $O(M)$.

3. Benchmarking via known Fourier-Bessel pairs

So as to compare the Fisk-Johnson dFBT of an objective function, represented by the sequence of $N-1$ values $\{F_0(j_m/T)\}_{m=1}^{N-1}$, to the exact transform, we need to agree upon a representative sequence of coordinate values of the transformed grid at which to evaluate both. Here we proceed as follows: we consider a further "benchmark" method wich was previously assessed, and, albeit being computationally rather inefficient, reported to be quite precise [10]. We refer to this reference method as the "Cree-Bones" (CB) algorithm, implemented as a numerical integration of the integral transform Eq. (1) using a trapezoidal rule and grid partitioning as reported in Ref. [10]. For comparison, if the objective function is available at M grid points, the CB algorithm terminates in time $O(M^2)$. Subsequently, considering a Fourier-Bessel transform pair, we compute the dFBT using the Fisk-Johnson and Cree-Bones procedures. The latter yields a sequence of coordinates $\{\rho_i\}_{i=0}^{M-1}$ at which we evaluate the exact transform and for which we extrapolate the Fisk-Johnson dFBT using [2]

$$F_0(\rho_i) = 2\sum_{m=1}^{N-1} \frac{j_m J_0(\rho_i T)}{J_1(j_m)(j_m^2 - \rho_i^2 T^2)} F_0(j_m/T). \tag{7}$$

As illustrated in Fig. 1, this not only allows to visually assess the performance of the Fisk-Johnson dFBT procedure for different choices of the truncation parameters T and N, but also allows to quantify the deviation from the exact transform in tems of the relative root-mean-squared error

$$\epsilon_{\rm RMS} = \left(\frac{\sum_i [F_0^{\rm exact}(\rho_i) - F_0^{\rm dFBT}(\rho_i)]^2}{\sum_i [F_0^{\rm dFBT}(\rho_i)]^2} \right)^{-1/2}. \tag{8}$$

dFBT of a jinc *function:* First we considered the Fourier-Bessel pair

$$f(r) = a_0^2\,{\rm jinc}(a_0 r) \quad \Longleftrightarrow \quad F_0(\rho) = \theta(\rho - a_0), \tag{9}$$

for $a_0 = 3$, wherein ${\rm jinc}(x) = J_1(x)/x$ and $\theta(\cdot)$ signifies the Heavyside step function. While here, the Fisk-Johnson algorithm exploits the possibility to compute $f(r)$ at precisely those sample points required by Eq. (3), the Cree-Bones algorithm used an equispaced grid $\{r_i\}_{i=0}^{M-1}$ with, $r_i = r_0 + i\Delta$ and $\Delta = (r_{M-1} - r_0)/(M-1)$ where $r_0 = 0.01$, $r_{M-1} = 20$, $M = 1000$. As usual in Fourier-type function approximation, due to the discontinuous nature of θ, we expect this kind of benchmark transform pair to represent a difficult test for any kind of dFBT. Bearing this in mind, the considered transform pair might be regarded as a worst-case use case that might arise in computational biophotonics since a commonly employed irradiation source profile (ISP), referred to as "top-hat" ISP, exhibits the shape of θ [11, 5]. Consequently, any convolution using such an ISP involves a revese dFBT of the above form.

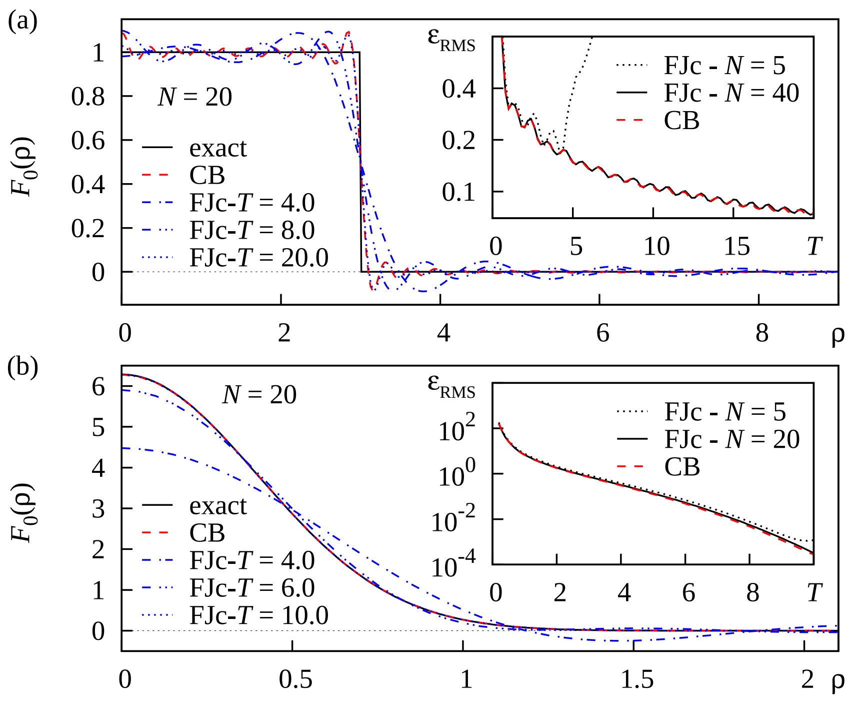

Figure 1. (Color online) Discrete Fourier-Bessel transform (dFBT) for two benchmark transform pairs and different transform parameters T and N, following the method of Fisk-Johnson for a continuous objective function discussed in section 3. (a) Transform of the jinc-function discussed in the text. The main plot shows a sequence of extrapolated function values of the transform for differenct choices ot the truncation parameter T at $N = 20$ (blue dash-dotted curves; labeled FJc). The result obtained by a straight forward numerical integration of the integral transform is shown as red dashed curve (labeled CB). The inset illustrates the root-mean squared error $\epsilon_{\rm RMS}(T)$ for two choices of N. (b) Transform of the Gaussian function discussed in the text. The main plot shows the extrapolated function values for different choices of T at $N = 20$ and the inset illustrates $\epsilon_{\rm RMS}$ for two values of N.

In Fig. 1(a) we show the result of applying the dFBT to the above jinc-function. In the main plot of Fig. 1(a) we explore the effect a finite truncation threshold T has on the transformed function for the summation threshold $N = 20$. Note that for small values of T the Fisk-Johnson dFBT deviates significantly from the exact transform (solid black line). This is due to the assumption that above T the objective function vanishes and, hence, structural details of the jinc-function bejond that threshold are ignored in the transformation process. As the value of T grows larger, the dFBT approximation at sufficiently large N gets increasingly better as shown by the overall decrease of the RMS error in the inset. While, at given T, a too small value of N leads to a huge RMS error, reflecting that the truncated Fourier-Bessel series has not converged as in the case of $N = 5$, the accuracy of the Fisk-Johnson transform at $N = 40$ is similar to that of the (computationally more expensive) Cree-Bones transform. To support intuition, further computer experiments indicate that, e.g., at $T = 10$ there exists a narrow threshold range $N = 6-12$ within which the RMS error decreases by one order of magnitude (not

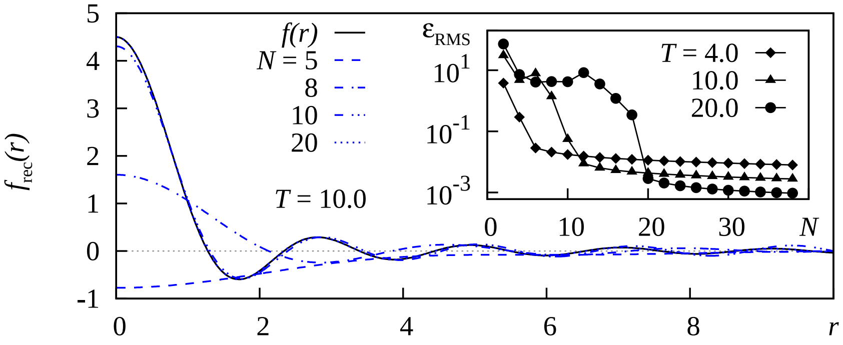

Figure 2. (Color online) Reconstruction f_{rec} of the initial function under a reverse dFBT. The main plot shows the original objective function (solid black curve) and the reconstructed functions at $T = 10.0$ for different values of N. The inset illustrates the relative RMS error (see Eq. 8) for three choices of the truncation threshold T for $N = 2$ through 40.

shown; see discussion below), and where $\epsilon_{\text{RMS}}(N > 12) \approx 0.12$ (cf. inset of Fig. 1(a)). For completeness, note that for $T = 20$ and $N = 20$, the Fisk-Johnson and Cree-Bones dFBT agree well as illustrated in the main plot of Fig. 1(a). Both feature Gibbs ringing artifacts that might be expected for this kind of transform pair.

dFBT of a Gaussian function: Next, we consider the dFBT transform of a Gaussian function

$$f(r) = \exp\{-r^2/(4\pi)\} \quad \Longleftrightarrow \quad F_0(\rho) = 2\pi \exp\{-\pi r^2\}. \tag{10}$$

For the numerical experiments using the Cree-Bones algorithm we again used an equispaced grid $\{r_i\}_{i=0}^{M-1}$ with, $r_i = r_0 + i\Delta$ and $\Delta = (r_{M-1} - r_0)/(M-1)$ where $r_0 = 0.01$, $r_{M-1} = 10$, $M = 1000$. For this kind of smooth benchmark transform pair we expect the accuracy of the tranform to be even better than in the previous case. This type of objective function might be regarded as a best-case use case that might arise in computational biophotonics since another commonly employed ISP has the shape of a simple Gaussian function [11, 5].

As evident from Fig. 1(b), similar to the previous example, if the truncation threshold T is chosen too small, the transfrom deviates from the exact result since vital parts of the objective function beyond T are ignored. To support intuition, note that $f(r)$ drops to its $1/e$-height at $T = 2\pi^{1/2} \approx 3.5$, explaining the deviation of the $T = 4$ dFBT approximation to the exact result. However, a visual inspection of the approximation at $T = 10$, where one finds $f(0)/f(10) \approx 2.8 \cdot 10^3$, indicates that it fits the asymptotic result quite well. This "chi-by-eye" result is supported by the relative RMS error illustrated in the inset. Even at small values of the summation trunction parameter N, the accuracy of the Fisk-Johnson dFBT improves noticably as $T \to 10$ and approaches the approximation error of the CB transform at a given value of T rapidly as N is adjusted to higher values.

Reconstruction of the objective function: Next, we assess the accuracy of a reconstruction of the objective function under a reverse dFBT implemented according to Eq. (5). Therefore, we first compute the dFBT approximation to the jinc-objective function using the sequence of grid samples required by Eq. (3), where we considered the truncation threshold $T = 10.0$ and different values of N. The results of a subsequent reverse transformation, computed for a sufficiently sampling density of x via the Fisk-Johnson procedure are summarized in Fig. 2. As evident from the main plot of the figure, the reconstruction of the objective function seems to be quite accurate once the summation truncation parameter exceeds $N = 20$. This finding can be put on a more quantitative basis by means of the relative RMS error, reported in the inset of Fig. 2. We find that at $T = 10.0$ there exists a narrow threshold range $N = 6-12$ within which the RMS error decreases by almost three orders of magnitude from $\epsilon_{\rm RMS}(N=6) \approx 7.7$ to $\epsilon_{\rm RMS}(N=12) \approx 0.007$. For higher (smaller) values of T, this threshold range can be seen to shift towards higher (smaller) values of N. This is intuitive since at larger values of T more sample points of the transformed domain are necessary to capture the structural details of the underlying function appropriately, thus affecting the convergence of the truncated sums used to approximate the Fourier-Bessel integral transform.

Exemplary polar convolution: Finally, we test the performance of the dFBT for the purpose of 2D polar convolution. Therefore, we consider the two functions

$$f(r) = \begin{cases} 1 & \text{for } r \leq r_0 \\ \exp\{-(r-r_0)^2/a_0^2\} & \text{for } r > r_0 \end{cases}, \tag{11a}$$

$$g_\epsilon(r) = (2\pi\epsilon^2)^{-1} \exp\{-r^2/(2\epsilon^2))\}. \tag{11b}$$

and follow the procedural description detailed in section 2 to compute $\mathsf{polConv}[f, g_\epsilon]$. Note that Eq. (11$a$) represents a "flat-top" ISP, i.e. a top-hat function with a smooth roll-off, consistent with actual beam profiles observed in laboratory experiments, see Refs. [12, 13, 11, 14, 15] that report on flat-top ISPs with parameter ratio in the range $r_0/a_0 = 1-10$. Further, Eq. (11b) signifies a Gaussian approximation to a 2D delta-function, attained in the limit $\epsilon \to 0$. Hence, we expect to find $\lim_{\epsilon\to 0} \mathsf{polConv}[f, g_\epsilon](r) = f(r)$. In this question, Fig. 3 illustrates the accuracy of the Fisk-Johnson convolution procedure for a "steep" example with $r_0/a_0 = 300$ and a "smooth" example with $r_0/a_0 = 1.5$, see Figs. 3(a) and (b), respectively. As evident from the scaling behavior of the associated RMS error between $f(r)$ and $h(r)$ (shown in the inset of the subfigures), the accuracy of the approximation at fixed $T = 1.0$ and given ϵ increases as the summation truncation parameter N increases, saturating at a characteristic limiting value N_ϵ. As ϵ decreases, i.e. the closer $g_\epsilon(r)$ approximates a delta-function, the approximation error of $h(r)$ also decreases. Bearing in mind the above results for the forward and reverse dFBT it does not come as a surprise that the polar convolution of a "smooth" objective function with a delta-function is more accurate than that of a "steep" objective function.

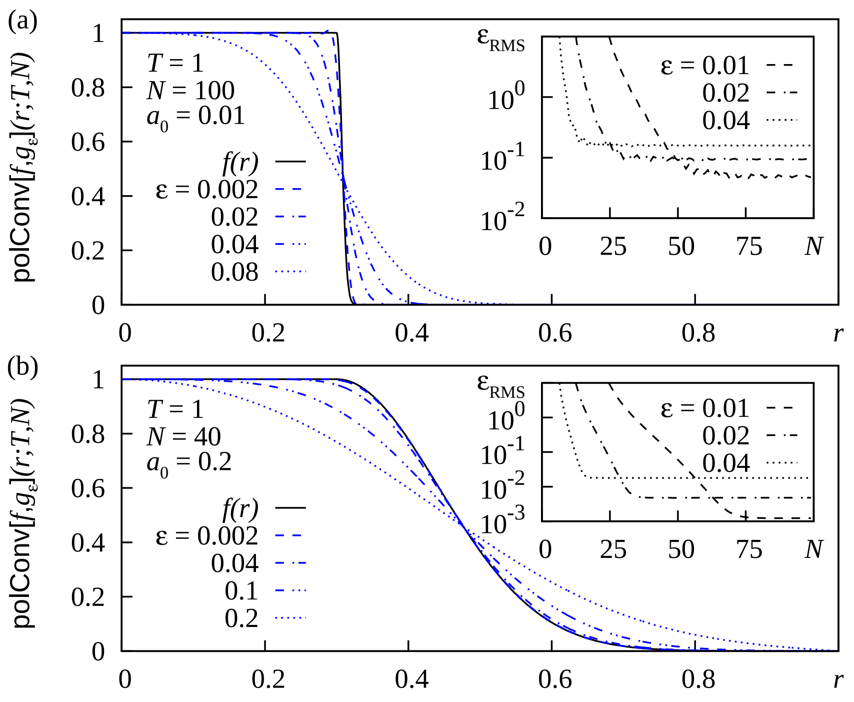

Figure 3. (Color online) Exemplary 2D polar convolution using the Fisk-Johnson procedure detailed in section 2. The figure illustrates the convolution $h(r) = \mathsf{polConf}[f,g](r;T,N)$ of a flat-top profile $f(r)$ with a Gaussian approximation $g_\epsilon(r)$ to the delta-distribution, see section 3. In the limit $\epsilon \to 0$ we expect to find $h(r) \to f(r)$. (a) Convolution using the "steep" flat-top parameters $r_0 = 0.3$, $a_0 = 0.01$ and truncation thresholds $T = 1$, $N = 100$ for different values of ϵ. The inset shows the relative RMS errors for the approximation of $f(r)$ by $h(r)$ as function of the summation truncation parameter N. (b) same as (a) for "smooth" flat top parameters $r_0 = 0.3$, $a_0 = 0.2$ and summation truncation parameter $N = 40$.

4. Beam-profile convolution in polar coordinates

In the remainder we consider a paradigmatic problem in computational biophotonics that requires a large number of polar convolutions, and, consequently, many forward and reverse dFBTs. One might thus benefit from the potential time-efficiency of the algorithmic procedure outlined in section 2 and assessed in section 3. In procedural terms, we provide a general solver for the problem of computing the material response to radially symmetric custom laser beams of finite extend for layered homogeneous media, given the corresponding Green's function response of the medium. To illustrate the computational procedure we considered the simple but paradigmatic case of a semi-infinite medium with a refractive-index-mismatched boundary. For the optical parameters we used the relative refractive indices $n = 1.0$ (for the ambient medium) and $n = 1.37$ as well as the absorption coefficient $\mu_{\rm a} = 0.1\,{\rm cm}^{-1}$, scattering coefficient $\mu_{\rm s} = 10.0\,{\rm cm}^{-1}$ and values of the anisotropy parameter $g \in [0.1, 0.95]$.

Table 1. Characteristic lengthscales [16] for light transport in the considered tissue setup and homogeneous grid parameter for discretization of the source volume using `MCML` [8]. From left to right: anisotropy parameter g, mean free path (mfp) length $\ell_{\rm mfp}$, transport mfp $\tilde{\ell}_{\rm mfp}$, penetration depth $d_{\rm p}$ and grid parameters for the cylindrical sampling lattice.

				z-axis			r-axis		
g	$\ell_{\rm mfp}$ (cm)	$\tilde{\ell}_{\rm mfp}$ (cm)	$d_{\rm p}$ (cm)	N_z (bins)	Δ_z (cm)	$z_{\rm max}$ (cm)	N_r (bins)	Δ_r (cm)	$r_{\rm max}$ (cm)
0.10	0.099	0.110	0.605	363	0.005	1.815	1000	0.002	2.0
0.70	0.099	0.323	1.037	622	0.005	3.11	1000	0.0033	3.3
0.90	0.099	0.909	1.741	1044	0.005	5.22	1000	0.0053	5.4
0.95	0.099	1.667	2.357	1414	0.005	7.07	1000	0.0073	7.3

Monte Carlo modelling of the Greens function response: For our numerical experiments we computed the Green's function G of the absorbed energy density for the above setup as the material response to an infinitely thin "pencil" beam using the publicly available `C` code `MCML` [8]. It solves the problem of steady-state light transport in terms of a Monte Carlo approach to photon migration in layered media and provides the accumulated observables on a homogeneous polar grid, i.e. $G \equiv G(r, z)$. For our numerical experiments we used the simulation parameters listed in Tab. 1. In setting up the discretized source volume we made sure the maximal z-depth $z_{\rm max}$ and r-range $r_{\rm max}$ exceed the penetration depth $d_{\rm p}$ of photons within the medium by a factor of three at least. Note that for extended beam profiles and not too close to the material surface, $d_{\rm p}$ refers to the intrinsic length-scale after which the fluence-rate along the beam-axis reduces to its $1/e$-value [8, 16]. For completeness, one might perform the numerical experiments as well by one of `MCML`s descendants designed for layered homogeneous media, as, e.g., `GPU-MCML` [17].

Material response to laser beams with finite extend: In order to obtain the desired material response $W(r, z)$ to an extended radially symmetric laser beam, the Green's function $G(r, z)$ needs to be convolved using an appropriate transverse ISP (irradiation source profile) $f(r)$. In principle this can be done using the publicly available `C` code `CONV` [5], that implements a top-hat and a Gaussian ISP. However, note that since `CONV` features only these two ISPs it is of rather limited use. Albeit allowing for a highly efficient direct convolution involving the solution of 1D integrals only, both beam profiles are not consistent with actual profiles observed in laboratory experiments, see, e.g., Refs. [12, 13, 11, 14, 15]. Further, on a more general basis, a computationally efficient and more versatile solution procedure that allows for convolution with custom ISPs seems to be of value.

In this regard we follow a different approach by solving the 2D convolution problem

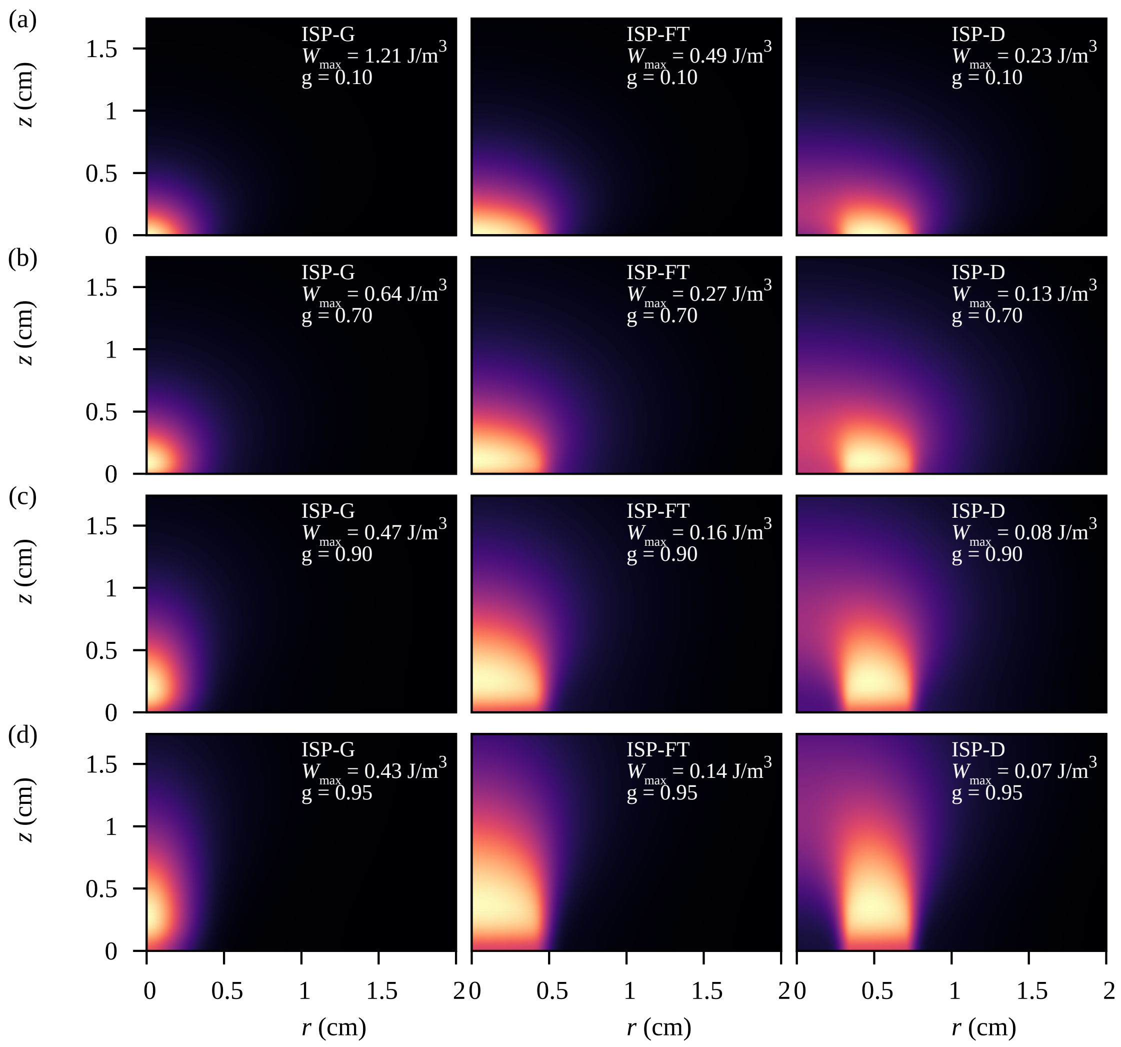

Figure 4. (Color online) Illustration of the beam shape convolution procedure for a Gaussian ISP (G; a special case of Eq. (13) with $r_0 = 0$ and $r_1 = 0$) with parameter $a_1 = 0.25$, flat-Top ISP (FT; a special case of Eq. (13) with $r_0 = 0$) with parameters $r_1 = 0.4$ and $a_1 = 0.1$, and donut (D) ISP with parameters $r_0 = 0.25$, $r_1 = 0.6$ and $a_0 = a_1 = 0.05$ considering four different values of the anisotropy parameter g. (a) from left to right (ltr): G, FT and D ISP for $g = 0.10$, (b) ltr: G, FT and D ISP for $g = 0.70$, (c) ltr: G, FT and D ISP for $g = 0.90$, (d) ltr: G, FT and D ISP for $g = 0.95$. The maximal value of absorbed laser energy $W_{\max}$ (J/m^3) for each configuration, indicated by the brightest color, is listed within the individual subfigures.

in terms of the Fourier-Bessel transform in polar coordinates [5, 7]

$$W(r,z) = \mathsf{polConv}[f,G](r,z) = 2\pi f_0 \int_0^\infty G_0(\rho,z)F_0(\rho)J_0(\rho r)\rho \, \mathrm{d}\rho, \qquad (12)$$

following the discretization procedure detailed in section 2. Therein, G stands for the laser absorption Green's function computed for an infinitely narrow laser beam, normally incident upon the material surface. For our exemplary numerical experiments we used

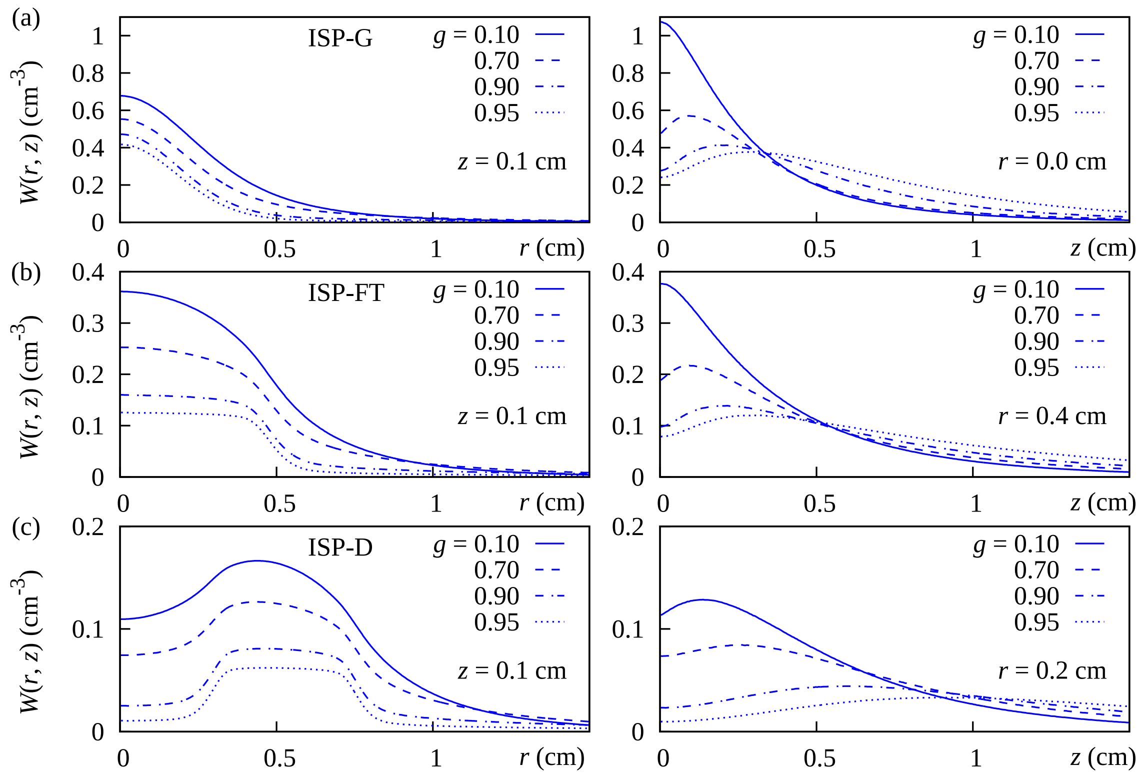

Figure 5. (Color online) Absorbed energy density $W(r,z)$ at fixed z- and r-slices for the three ISPs used in section 4. (a) Gaussian ISP at $z = 0.1\,$cm (left) and $r = 0.0\,$cm (right) considering different values of the anisotropy g, (b) Flat-top ISP at $z = 0.1\,$cm (left) and $r = 0.4\,$cm (right), (c) Donut ISP at $z = 0.1\,$cm (left) and $r = 0.2\,$cm (right).

the custom "donut" ISP

$$f(r) = \begin{cases} \exp\{-(r-r_0)^2/a_0^2\} & \text{for } r < r_0 \\ 1 & \text{for } r_0 \leq r \leq r_1 \\ \exp\{-(r-r_1)^2/a_1^2\} & \text{for } r > r_1 \end{cases}. \tag{13}$$

The respective dFBTs are given by F_0 and G_0. In the above equation, f_0 allows to scale the beam intensity to achieve a total beam power P via

$$f_0 = P\,[2\pi \int_0^\infty r\ f(r)\ \mathrm{d}r]^{-1}. \tag{14}$$

Note that this yields a general purpose routine that allows for quite arbitrary beam profiles, only required to obey the integrability conditions of a Fourier-Bessel transform. As a technicality, note that the dFBT F_0 of the continuous ISP f, computed using the $O(M^2)$ algorithm Eq. (3), can be reused at each value of z. In contrast to the later function, since $G(r,z)$ is known at a finite number of sample points only, its dFBT G_0 is obtained via the $O(NM)$ algorithm Eq. (4).

In Fig. 4 we illustrate the convolution procedure for various anisotropy parameters and three beam shapes: (i) a Gaussian ISP (G), i.e. a special case of Eq. (13) with

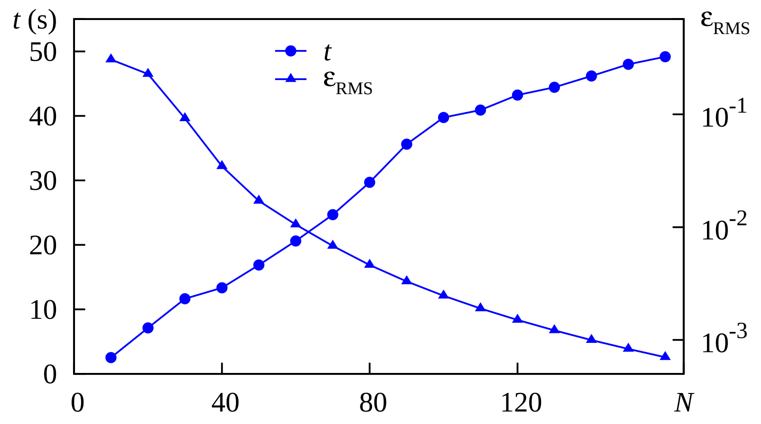

Figure 6. (Color online) Accuracy and computational efficiency of the Fisk-Johnson polar convolution as function of the summation truncation parameter N at fixed $T = 4.0$ for the flat-top ISP. The reconstruction RMS error of the ISP decreases below 10^{-2} at approximately $N = 50$. At this point, the convolution procedure terminates after $t(50) \approx 17\,$s. For comparison: the Cree-Bones procedure used for benchmarking terminates after time $t_{\rm CB} \approx 326\,$s, highlighting the superior performance of the Fisk-Johnson algorithm.

$r_0 = 0$, $r_1 = 0$ and $a_1 = 0.25$ where we used the dFBT parameters $T = 4.0$ and $N = 40$, (ii) a flat-Top ISP (FT), a special case of Eq. (13) with $r_0 = 0$, $r_1 = 0.4$ and $a_1 = 0.1$ using $T = 4.0$ and $N = 80$, and, (iii) a donut (D) ISP with parameters $r_0 = 0.25$, $r_1 = 0.6$ and $a_0 = a_1 = 0.05$ using $T = 4.0$ and $N = 150$. Based on the parameter studies for the forward and reverse dFBT reported in section 3, and by monitoring the rms error for the forward and immediate backtransformation of the beam profile, yielding $\epsilon_{\rm rms} < 10^{-6}$ (ISP-G), $\epsilon_{\rm rms} = 0.003$ (ISP-FT), and, $\epsilon_{\rm rms} = 0.008$ (ISP-D), we opted for the truncation threshold T and summation truncation parameters N listed above. To clarify the behavior of $W(r,z)$ and to illustrate the decrease of $W_{\rm max}$ as function of g, samples of the absorbed energy density at fixed z- and r-slices are shown in Fig. 5. As one might intuitively expect, Figs. 4 and 5 reveal two tendencies: (i) for increasing anisotropy g, the smoothing of $W(r,z)$ due to scattering reduces and its absolute values decreases since backscattering is suppressed, and, (ii) for increasing g, the maximum $W_{\rm max}$ shifts towards deeper values of z since scattering is focused on the forward direction.

Accuracy and computational efficiency: Further, note that time efficiency is an issue: so as to complete the convolution procedure for, say, the sampled source volume at $g = 0.95$, an individual convolution has to be carried out for a sequence of $N_z = 1414$ consecutive values of z, each involving a number of $N_r = 1000$ sample points r, see Tab. 1. For the exemplary case of the previous flat-top beam profile, Fig. 6 reveals that the completion time of the Fisk-Johnson convolution procedure is linear in M with $t(N) \approx 0.34(1)N\,$s. In particular, the reconstruction error of the ISP decreases below 10^{-2} at approximately $N = 50$. At this value of N, the Fisk-Johnson procedure terminates after $\approx 17\,$s. In contrast, note that the Cree-Bones procedure used for benchmarking in section 3 terminates after $\approx 326\,$s, highlighting the efficiency of the Fisk-Johnson polar convolution for the considered application.

5. Summary and conclusions

In the presented article we discussed a time efficient algorithmic procedure for computing a 2D polar convolution of two radially symmetric functions, based on discrete approximations of the forward and reverse Fourier-Bessel integral transform. The specific objective of this study was to exploit the time efficiency of the procedure, and to assess whether it outperforms a naive approach, for applications in computational biophotonics that require a large number of polar convolutions. In procedural terms, we provided a solver for a problem in tissue optics where the task is to convolve the Green's function response of a multilayered tissue with a custom radially symmetric irradiation source profile to yield the response to a laser-beam of finite extend. Note that the presented approach nicely complements the publicly available `CONV` code [5], which is restricted to a top-hat or Gaussian ISP. We found that, from a point of view of computational efficiency, the presented procedure resides between the highly efficient but ISP-restricted direct convolution (implemented by the `CONV` code) and the less efficient but accurate straight forward numerical quadrature used for benchmarking in section 3. Hence, we conclude that, considering realistic custom ISPs, the presented approach to beam-shape convolution in computational biophotonics should be preferred over the assessed alternatives.

For completeness, note that there is also a rather recent method that allows for the calculation of Fourier-Bessel expansions and dHTs of order zero with an asymptotic running time $O(N\log(N)^2/\log\log(N))$ [18]. However, as indicated by the benchmarks and conclusions presented in the latter reference, the elaborate numerical procedure outperforms the naive dHT algorithm only for $N \geq 6000$ ($N \geq 2000$) if a working accuracy ϵ (defined in Ref. [18]) of $\epsilon = 10^{-15}$ (10^{-8}) is aimed for. Note that the application discussed in section 4 has $N \leq 1414$ and is thus solved more efficiently by our procedure than by the asymptotically faster method presented in Ref. [18].

Albeit the scientific literature frequently features new algorithic descriptions to compute the above (and further related) transforms for particular scientific applications, their thorough exploration and implementation in terms of, say, symbolic computer algebra and scripting languages is rather recent [19, 20]. Since the discrete Fourier-Bessel transform and the polar convolution are valuable computational tools for the solution of many physical problems with axial symmetry, and to facilitate transparency and reproducability of results in scientific publications [21, 22, 23], we made the concise research-code used in the presented study publicly available [24].

Acknowledgments

This research work received funding from the VolkswagenStiftung within the "Niedersächsisches Vorab" program in the framework of the project "Hybrid Numerical Optics" (HYMNOS; Grant ZN 3061). The software was developed and tested under OS X Yosemite (Version: 10.10.3) on a MacBook Air featuring a 1.7GHz Itel Core i5 processor and 4 GB DDR3 using `python` [25] version 2.7.6 and `numpy` version 1.8.0rc1 [26]. All figures were generated using `gnuplot` [27].

Appendix A. Unit testing research code

In the presented appendix we address the important issue of testing self-written research code. While it is one thing to check code on a verification base, i.e. to see whether the output of a (possibly) complex algorithmic procedure is in accord with ones expectation, it is generally more beneficial to put atomistic and critical software components under scrutiny. Therefore, it is advisable that any organized scientific software project is complemented by a suit of test cases. Useful test cases usually emerge naturally during the code development and debugging cycle and might be combined as test suite. In the development phase one might run these tests frequently to assess and verify a correct function of the implemented procedures, thus providing a means of quality control. As an aside, note that the excellent writeups of Noble [28], discussing the logistics of carrying out computational experiments, and, Baxter *et al.* [29] and Wilson *et al.* [30], proposing best practices for scientific software development, are a great source of guidelines for any (beginning) scientist working at an interface between science and software engineering.

In our specific example, the `python` standard library, and, beyond that, the full `python` ecosystem, offers a wealth of possibilities that facilitate software testing. In particular, the versatile `unittest` module implements software development tools for constructing test cases and running tests on critical software parts [31]. Below we illustrate how to verify limiting cases and check conservation laws that hold for the implemented procedures via `unittest`. Note that this is in accord with the ideal of planning for mistakes by using an off-the-shelf unit testing library, i.e. best practice 5c of Ref. [30].

In the remainder we therefore discuss two `python` code listings: code listing 1, containing a substantial part of our implementation of the discrete Fourier Bessel transform of order zero for a continuous objective function discussed in sec. 2, and, code listing 2, containing a collection of test cases facilitating quality control. Both code listings comprise a standalone test suite derived from our research code to exemplify the highly rewarding practice of unit testing. For conciseness, the presented listings are not documented thoroughly. In general, this has to be considered bad programming style. However, note that the code in the supplementary material under Ref. [24] is documented to improve its understandability, following best practices 7a-c of Ref. [30].

Comparison to exact results: The benchmarking of the Fisk-Johnson dFBT vs. the Cree-Bones reference procedure, exemplified using two distinct Fourier-Bessel transform pairs in sec. 3, represents a parametric study that elaborates on an atomistic operation of a critical software part that was cast into a unit test early-on in the development phase. The respective test case scrutinizes the forward transform Eq. (3), implemented in lines 5-20 of listing 1, in conjunction with the extrapolation formula Eq. (7), implemented in lines 38-51 of listing 1. The resulting test case, named `test_dFBT_fourierPair` is implemented in lines 27-30 of code listing 2. It evaluates the root mean square error $\epsilon_{\rm RMS}$ of the numeric and exact transform following Eq. (8) and checks whether this falls below the precision threshold of $\epsilon = 10^{-6}$. Given the test fixture implemented by the method `setUp()` (listing 2, lines 19-22), the test is expected to pass since $\epsilon_{\rm RMS} = O(10^{-12})$.

A further immediate test case relates to the reconstruction of the objective function, putting the forward transform Eq. (3) and backward transform Eq. (5), the latter implemented in lines 23-35 of code listing 1, under scrutiny. The respective test case, named `test_dFBT_selfReciprocality` is implemented in lines 32-35 of listing 2. Given the implemented test fixture, the test is expected to pass since, again, $\epsilon_{\rm RMS} = O(10^{-12})$. A parametric study that elaborates on this test case is detailed in sec. 3. Note that both test cases consider the Fourier-Bessel transform pair detailed in Eq. (10), cf. listing 2 lines 7-10.

Verifying conservation laws: A third test case can be implemented on the basis of Parseval's theorem discussed in sec. 8 of Ref. [6]. Therein, if a particular matrix Y (see Eq. (19) of Ref. [6]) is chosen as transformation kernel for the dFBT, and if the scaled coefficients $\tilde{f}_k$ (see Eq. (46) of Ref. [6]) and $\tilde{F}_m$ (see Eqs. (33) and (46) of Ref. [6]) are introduced, Parseval's theorem

$$\sum_{m=1}^{N-1} |\tilde{F}_m|^2 = \sum_{k=1}^{N-1} |\tilde{f}_k|^2 \tag{A.1}$$

should be satisfied. This respective test case is implemented under the name `test_dFBT_generalizedParsevalTheorem` in lines 37-52 of listing 2. Given the implemented test fixture, the test is expected to pass since the difference of both sums is $O(10^{-13})$.

Running the tests: A test case class can simply be created by subclassing the object `unittest.TestCase`. Here, the resulting class `FourierBesselTransformTestCase` (listing 2, line 17) features the above test cases as methods beginning with `test` to inform the test runner about their intended role. While the default setting allows to properly distinguish a success or failure of the testrun, a custom test runner might be passed to `main` in line 56 of listing 2. For each of the above three tests, the two functions `setUp()` (listing 2, lines 19-22) and `tearDown()` (listing 2, lines 24p) handle the preparation of test fixture and cleanup, respectively. Finally, running the tests with some level of detail yields the following output:

```
> python -m unittest -v test_dFBT
test_dFBT_fourierPair (test_dFBT.FourierBesselTransformTestCase) ... ok
test_dFBT_generalizedParsevalTheorem (test_dFBT.FourierBesselTransformTestCase) ... ok
test_dFBT_selfReciprocality (test_dFBT.FourierBesselTransformTestCase) ... ok

----------------------------------------------------------------------
Ran 3 tests in 0.020s

OK
```

```python
import scipy.special as scs
import numpy as np


def fwdTrafo(r,f,T=None,N=10):
    T = max(r) if T==None else min(T,max(r))

    jm = scs.jn_zeros(0,N)
    jp = jm[:-1,np.newaxis]
    jN = jm[-1]
    x = jp/jN

    # DISCRETE TRANSFORMATION EQUATION RELATING PARTICULAR VALUES F(j[m]/T)
    # m=0...N-1 OF TRANSFORMED FUNCTION TO PARTICULAR VALUES f(x[p] T)
    # p=0...N-2 OF OBJECTIVE FUNCTION. SEE EQ. (12) OF REF. [2].
    F0 = 2.0*(T/jN)**2*np.sum(
        f(x*T)*scs.j0(x*jm)/scs.j1(jp)**2,
        axis=0)

    return jm/T, F0, T


def bckwdTrafo(r,F0,T):
    f = np.zeros(r.size)
    jm = scs.jn_zeros(0,F0.size)

    # REVERSE TRANSFORM YIELDING ARBITRARY FUNCTION VALUES f(xT) FROM ITS
    # FOURIER BESSEL TRANSFORM F(j[m]/T) m=0...N-1 AT SCALED BESSEL ZEROS
    # j[m]/T. SEE EQ. (10) OF REF. [2].
    x = r/T
    f[x<1] = 2.0/T**2*np.sum(
        F0*scs.j0(jm*x[x<1,np.newaxis])/scs.j1(jm)**2,
        axis=1)

    return f


def extrapolate(rho,F0,T):
    jFull = scs.jn_zeros(0,F0.size)
    jm = jFull[:-1]
    jN = jFull[-1]
    r  = rho*T/jN

    # EXTRAPOLATION FORMULA EQ. (9) OF REF. [2].
    F0Ex = 2.0*np.sum(
        F0[:-1]*scs.j0(r[:,np.newaxis]*jN)*jm/
        scs.j1(jm)/
        (jm**2-r[:,np.newaxis]**2*jN*jN),
        axis=1)

    return F0Ex
```

Listing 1. Implementation of the discrete Fourier Bessel Transform of order zero (aka Hankel transform) for a continuous objective function in `python` module file `dFBT.py`. The algorithmic procedure follows the method detailed in Ref. [2].

```python
import unittest
import dFBT
import scipy.special as scs
import numpy as np


def FBPair():
    f  = lambda r: np.exp(-r*r/4/np.pi)
    F0 = lambda r: np.exp(-r*r*np.pi)*2*np.pi
    return f, F0


def eRMS(Fn,Fx):
    return np.sqrt(((Fn-Fx)**2).mean()/(Fx*Fx).mean())


class FourierBesselTransformTestCase(unittest.TestCase):

    def setUp(self):
        self.r = np.linspace(0.0, 20.0, 1000)
        self.f, self.Fx = FBPair()
        self.T, self.N = 18.0, 20

    def tearDown(self):
        del self.N, self.T, self.r, self.f, self.Fx

    def test_dFBT_fourierPair(self):
        rhoFJC,F0FJC,T = dFBT.fwdTrafo(self.r,self.f,self.T,self.N)
        F0FJCEx = dFBT.extrapolate(self.r,F0FJC,self.T)
        self.assertLessEqual(eRMS(F0FJCEx,self.Fx(self.r)), 1e-6)

    def test_dFBT_selfReciprocality(self):
        rhoFJC,F0FJC,T = dFBT.fwdTrafo(self.r,self.f,self.T,self.N)
        fFJC = dFBT.bckwdTrafo(self.r,F0FJC,self.T)
        self.assertLessEqual(eRMS(fFJC,self.f(self.r)), 1e-6)

    def test_dFBT_generalizedParsevalTheorem(self):
        j = scs.jn_zeros(0,self.N)
        Fm = np.zeros(self.N)
        Y = lambda m,k: 2.0*scs.j0(j[m]*j[k]/j[-1])/j[-1]/scs.j1(j[k])**2

        rhoFJC,F0FJC,T = dFBT.fwdTrafo(self.r,self.f,self.T,self.N)
        fk  = F0FJC*2./(self.T*self.T*scs.j1(j)**2)

        for im in range(self.N):
            for ik in range(self.N):
                Fm[im] += Y(im,ik)*fk[ik]

        fkScaled = fk /scs.j1(j)
        FmScaled = Fm /scs.j1(j)

        self.assertAlmostEqual(np.sum(FmScaled**2), np.sum(fkScaled**2), 6)


if __name__ == "__main__":
    unittest.main()
```

Listing 2. Collection of testcases contained in `python` script file `test_dFBT.py`. These test are intended to assess and verify a correct function of the procedures implementing the discrete fourier bessel transform in module file `dFBT.py`.

References

[1] J. D. Talman. Numerical Fourier and Bessel transforms in logarithmic variables. *J. Comput. Phys.*, 29:35, 1978.

[2] H. Fisk Johnson. An improved method for computing a discrete Hankel transform. *Comput. Phys. Commun.*, 43:181, 1987.

[3] W. E. Higgins and D. C. Munson. A Hankel Transform Approach to Tomographic Image Reconstruction. *IEEE Trans. Med. Imaging*, 7(1):59–72, 1988.

[4] M. Guizar-Sicairos and J. C. Gutiérrez-Vega. Computation of quasi-discrete hankel transforms of integer order for propagating optical wave fields. *J. Opt. Soc. Am. A*, 21(1):53, 2004.

[5] L. Wang, S. L. Jacques, and L. Q. Zheng. CONV - convolution for responses to a finite diameter photon beam incident on multi-layered tissues. *Comput. Methods Programs Biomed.*, 54:141, 1997. See: http://omlc.org/software/mc/ [Online; accessed 2017-03-30].

[6] N. Baddour and U. Chouinard. Theory and operational rules for the discrete Hankel transform. *J. Opt. Soc. Am. A*, 32:611, 2015.

[7] N. Baddour. Operational and convolution properties of two-dimensional Fourier transforms in polar coordinates. *J. Opt. Soc. Am. A*, 26:1767, 2009.

[8] L. Wang, S. L. Jacques, and L. Q. Zheng. MCML - Monte Carlo modeling of photon transport in multi-layered tissues. *Comput. Methods Programs Biomed.*, 47:131, 1995. See: http://omlc.org/software/mc/ [Online; accessed 2017-03-30].

[9] Note that in Ref. [7], Eq. (6) is formulated in terms of the 2D Fourier transform $F_0^{\rm FT}$ of f, which, in case of a radially symmetric functions is related to the Fourier-Bessel transform via $F_0^{\rm FT}(\rho) = 2\pi F_0(\rho)$.

[10] M. J. Cree and P. J. Bones. Algorithms to numerically evaluate the Hankel transform. *Computers Math. Applic.*, 26(1):1, 1993.

[11] G. Paltauf and H. Schmidt-Kloiber. Pulsed optoacoustic characterization of layered media. *J. Appl. Phys.*, 88:1624–1631, 2000.

[12] G. Paltauf and H. Schmidt-Kloiber. Measurement of laser-induced acoustic waves with a calibrated optical transducer. *J. Appl. Phys.*, 82:1525, 1997.

[13] G. Paltauf, H. Schmidt-Kloiber, and M. Frenz. Photoacoustic waves excited in liquids by fiber-transmitted laser pulses. *J. Acoust. Soc. Am.*, 104:890–897, 1998.

[14] B. D'Alessandro and A. P. Dhawan. 3-D Volume Reconstruction of Skin Lesions for Melanin and Blood Volume Estimation and Lesion Severity Analysis. *IEEE Trans. Med. Imaging*, 31:2083, 2012.

[15] E. Blumenröther, O. Melchert, M. Wollweber, and B. Roth. Detection, numerical simulation and approximate inversion of optoacoustic signals generated in multi-layered PVA hydrogel based tissue phantoms. *Photoacoustics*, 4:125–132, 2016.

[16] B. C. Wilson and S. L. Jacques. Optical reflectance and transmittance of tissues: principles and applications. *IEEE J. Quantum Electron.*, 26:2186, 1990.

[17] E. Alerstam, W. C. Y. Lo, T. D. Han, J. Rose, S. Andersson-Engels, and L. Lilge. Next-generation acceleration and code optimization for light transport in turbid media using GPUs. *Biomed. Opt. Express*, 1:658–675, 2010.

[18] A. Townsend. A fast analysis-based discrete Hankel transform using asymptotic expansions. *SIAM J. Numer. Anal.*, 53:1897–1917, 2015.

[19] E. Dovlo and N. Baddour. Toolbox for the Computation of 2D Fourier Transforms in Polar Coordinates via Maple. *Journal of Open Research Software*, 3:e3, 2015.

[20] N. Baddour and U. Chouinard. Matlab Code for the Discrete Hankel Transform. *Journal of Open Research Software*, 5(1):4, 2017.

[21] N. Barnes. Publish your computer code: it is good enough. *Nature*, 467:753, 2010.

[22] D. C. Ince, L. Hatton, and J. Graham-Cumming. The case for open computer programs. *Nature*, 482:485–488, 2012.

[23] G. K. Sandve, A. Nekrutenko, J. Taylor, E. Hovig, and P. E. Bourne. Ten Simple Rules for Reproducible Computational Research. *PLoS Comput. Biol.*, 9:e1003285, 2013.

[24] dFBT-FJ: Efficient polar convolution based on the discrete Fourier-Bessel transform for application in computational biophotonics. https://github.com/omelchert/dFBT-FJ.git, 2016. [Online; accessed 2017-03-19].

[25] T. E. Oliphant. Python for Scientific Computing. *CS&E*, 9(3):10–20, 2007.

[26] E. Jones, T. E. Oliphant, P. Peterson, and many others. SciPy: Open source scientific tools for Python. http://www.scipy.org/, 2001. [Online; accessed 2017-03-06].

[27] T. Williams, C. Kelley, and many others. Gnuplot 5.0: an interactive plotting program. http://gnuplot.sourceforge.net/, 2015. [Online; accessed 2017-03-06].

[28] W. S. Noble. A Quick Guide to Organizing Computational Biology Projects. *PLoS Comput. Biol.*, 5(7):e1000424, 2009.

[29] S. M. Baxter, S. W. Day, J. S. Fetrow, and S. J. Reisinger. Scientific Software Development Is Not an Oxymoron. *PLoS Comput. Biol.*, 2(9):e87, 2006.

[30] G. Wilson, D. A. Aruliah, C. T. Brown, N. P. Chue Hong, M. Davis, R. T. Guy, S. H. D. Haddock, K. D. Huff, I. M. Mitchell, M. D. Plumbley, B. Waugh, E. P. White, and P. Wilson. Best Practices for Scientific Computing. *PLoS Biol.*, 12(1):e1001745, 2014.

[31] S. Purcell and many others. unittest - Unit testing framework. https://docs.python.org/3/library/unittest.html, 2003. The Python unittest module comprises a software development tool for constructing testcases and running tests. [Online; accessed 2017-03-28].